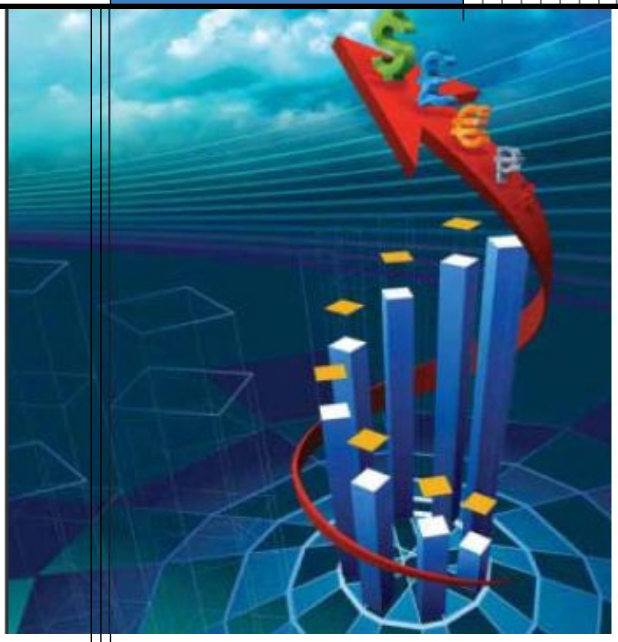

OPEN CIRRUS: A GLOBAL CLOUD COMPUTING TESTBED

Arutyun I. Avetisyan, *Institute for System Programming of the Russian Academy of Sciences*
Roy Campbell, Indranil Gupta, Michael T. Heath, and Steven Y. Ko, *University of Illinois at Urbana-Champaign*
Gregory R. Ganger, *Carnegie Mellon University*
Michael A. Kozuch and David O'Hallaron, *Intel Labs*
Marcel Kunze, *Karlsruhe Institute of Technology, Germany*
Thomas T. Kwan, *Yahoo! Labs*
Kevin Lai, Martha Lyons, and Dejan S. Milojicic, *HP Labs*
Hing Yan Lee, *Infocomm Development Authority of Singapore*
Ng Kwang Ming and Jing-Yuan Luke, *Malaysian Institute of Microelectronic Systems*
Han Namgong, *Electronics and Telecommunications Research Institute, South Korea*
Yeng Chai Soh, *Nanyang Technological University*

Open Cirrus is a cloud computing testbed that, unlike existing alternatives, federates distributed data centers. It aims to spur innovation in systems and applications research and catalyze development of an open source service stack for the cloud.

There is growing interest in cloud computing within the systems and applications research communities. However, systems researchers often find it difficult to do credible work without access to large-scale distributed data centers. Application researchers could also benefit from being able to control the deployment and consumption of hosted services across a distributed cloud computing testbed.

Pay-as-you-go utility computing services by companies such as Amazon and new initiatives by Google, IBM, Microsoft, and the National Science Foundation (NSF) have begun to provide applications researchers in areas such as machine learning and scientific computing with access to large-scale cluster resources. However, system researchers, who are developing the techniques and software infrastructure to support cloud computing, still have trouble obtaining low-level access to such resources.

Open Cirrus (http://opencirrus.org) aims to address this problem by providing a single testbed of heterogeneous distributed data centers for systems, applications, services, and open source development research. The project is a joint initiative sponsored by Hewlett-Packard (HP), Intel, and Yahoo! in collaboration with the NSF, the University of Illinois at Urbana-Champaign (UIUC), the Karlsruhe Institute of Technology (KIT), the Infocomm Development Authority (IDA) of Singapore, the Russian Academy of

 Published by the IEEE Computer Society

Sciences (RAS), the Electronics and Telecommunications Research Institute (ETRI) of South Korea, the Malaysian Institute of Microelectronic Systems (MIMOS), and Carnegie Mellon University (CMU). Additional members are expected to join Open Cirrus later this year.

As Figure 1 shows, the current testbed is composed of 10 sites in North America, Europe, and Asia. Each site consists of a cluster with at least 1,000 cores and associated storage. Authorized users can access any Open Cirrus site using the same login credential.

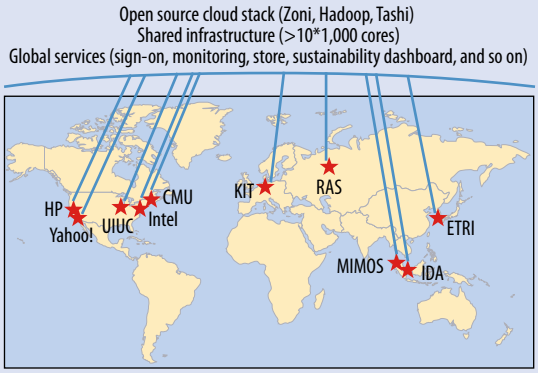

Figure 1. Open Cirrus testbed. Each of the 10 current sites consists of a cluster with at least 1,000 cores and associated storage. The testbed offers a cloud stack consisting of physical and virtual machines and global services such as sign-on, monitoring, storage, and job submission.

MOTIVATION AND CONTEXT

Open Cirrus has four main goals.

First, the project aims to foster systems-level research in cloud computing. In the current environment, only big service providers such as Yahoo!, Google, Amazon, and Microsoft have access to large-scale distributed data centers to develop and test new systems and services. Most cloud computing researchers must typically rely on simulations or small clusters. Open Cirrus aims to help democratize innovation in this area by providing two unique features essential to systems-level research:

- Open Cirrus sites allow access to low-level hardware and software resources—for example, install OS, access hardware features, and run daemons.
- The testbed comprises heterogeneous sites in different administrative domains around the world, enabling researchers to leverage multiple data centers.

Second, Open Cirrus seeks to encourage new cloud computing applications and applications-level research. Providing a platform for real-world applications and services is an important part of Open Cirrus. Particularly exciting are

- the potential for developing new application models and using them to understand the necessary systems-level support, and
- using the federated nature of Open Cirrus to provide a platform for new kinds of federated applications and services that run across multiple data centers.

Third, Open Cirrus offers a collection of experimental data. Cloud computing researchers often lack data sets with which to conduct high-quality experimental evaluations. Open Cirrus sites will let researchers import, store, and share large-scale data sets such as Web crawls and data-center workload traces. With such facilities, Open Cirrus could become a "watering hole" where researchers with similar interests can exchange data sets and develop standard cloud computing benchmarks.

Fourth, Open Cirrus aims to develop open source stacks and APIs for the cloud. To become widespread, cloud computing requires a nonproprietary and vendor-neutral software stack. Open Cirrus will serve as a platform that the open source community can use to design, implement, and evaluate such codes and interfaces for all cloud stack levels. Open source is as much about community as it is about software, and Open Cirrus seeks to become the foundation of a larger open cloud community.

The Open Cirrus sites are working together to provide a single federated testbed, as opposed to each site building and operating a separate cluster, for three reasons:

- collaborating on a single, larger effort will achieve greater impact than participants could individually;
- testing in the different site environments will improve the quality of software and services; and
- pooling resources will improve efficiency because the sites will be sharing innovations.

One measure of efficiency is management cost. Figure 2 shows ballpark cost figures gleaned from the current Open Cirrus sites. While the costs of running a cloud infrastructure increase with the number of sites, the savings from sharing software development and operational methods reduces overall costs. For example, several participating organizations are prominent developers of the software components in the Open Cirrus service architecture. By sharing these new systems and the lessons learned in deploying them, all of the sites benefit.

ARCHITECTURE, DESIGN, AND IMPLEMENTATION

Several high-level architectural choices drove the Open Cirrus design:

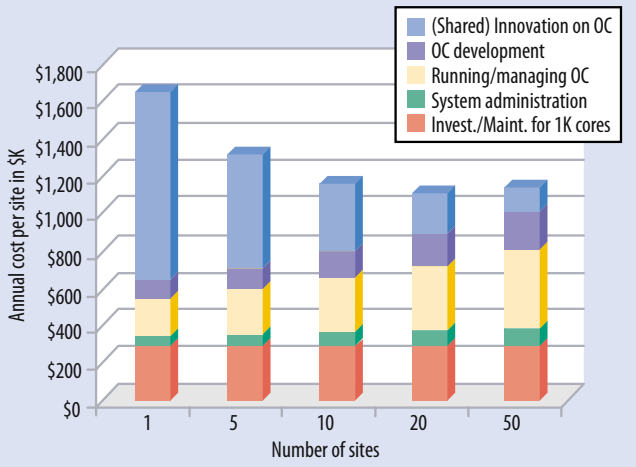

Figure 2. Open Cirrus management costs: annual cost per site for different numbers of sites. While the costs of running a cloud infrastructure increase with the number of sites, the savings from sharing software development and operational methods reduces overall costs.

- *Systems versus application-only research.* In contrast to clusters such as Google/IBM, Microsoft Windows Azure, and Amazon EC2/S3, Open Cirrus enables research using physical machines in addition to virtualized machines. This requires provisioning of the bare metal, enabling root access to provisioned servers, providing isolation at the network level, and reclaiming access in case of fraudulent or erroneous behavior.
- *Federated versus unified sites.* In contrast to a unified architecture such as PlanetLab, Open Cirrus federates numerous sites with various hardware, services, and tools. The sites exist on different continents, under different regulations and subject to different privacy concerns. Commonality is enabled by Open Cirrus global services under development, such as global sign-on and monitoring. Some local services may vary across sites, but common practices and regulations will promote consistent administration and oversight.
- *Data-center focus versus centralized homogeneous infrastructure.* Compared to a centralized approach such as Emulab, Open Cirrus revolves around multiple data centers. This data-center focus enables independent research while sharing resources. It has implications for security, enforcing authorizations between users and individual sites, and integration with existing organizational regulations.

Service architecture design

Design of the Open Cirrus service architecture is guided by a desire to create a unified and coherent resource, rather than several completely disjoint clusters.

Direct access to physical resources. Systems research is supported by allowing direct access to physical resources on the machine. For example, researchers can have root passwords, install kernel images, and access processors, chipsets, and storage. However, some resources, particularly network resources needed for proper isolation such as virtual local area network (VLAN) switch configurations, may be virtualized or unavailable.

Similar operating environments. Given that multiple organizations with different practices manage the Open Cirrus sites, it's infeasible for these sites to have identical operating environments. However, it's possible to create similar operating environments by defining a minimum set of services that every site must offer.

Support for near-production use. While supporting cloud computing system software research is Open Cirrus' central mission, robust research often requires access to real-world use cases. Therefore, Open Cirrus sites strive to offer high-quality services to researchers who aren't necessarily conducting research at the systems level. This use provides the workloads, traces, and testing needed for insights into real-world use.

Global services available from any site. A small set of global services are available from any Open Cirrus site. Examples include the single sign-on authentication service, global monitoring, and a moderate-scale storage service for configuration files, intermediate results, or binaries.

Service stack architecture

A typical Open Cirrus site consists of foundation, utility, and primary domain services, as Figure 3 shows.

Zoni. The foundation service for the software architecture is Zoni. Fundamentally, Zoni is the software component responsible for managing physical resources in the cluster and is crucial to providing users with bare-metal server access to conduct software system research. This component provides five key functions:

- allocation of server nodes;
- isolation of node groups, called *domains*;
- provisioning of key software in a domain;
- out-of-band server management; and
- debugging of allocated nodes.

Zoni maintains a database of all available cluster resources. User requests, which are primarily for some number of nodes, are satisfied by allocating resources from that inventory. Users request allocations of physical nodes for various reasons. One common reason is to conduct controlled performance measurements; another is for experimentation with system software that involves aspects of networking—for example, developing a new cluster management system may require controlling a Dynamic Host Configuration Protocol (DHCP) server. Ex-

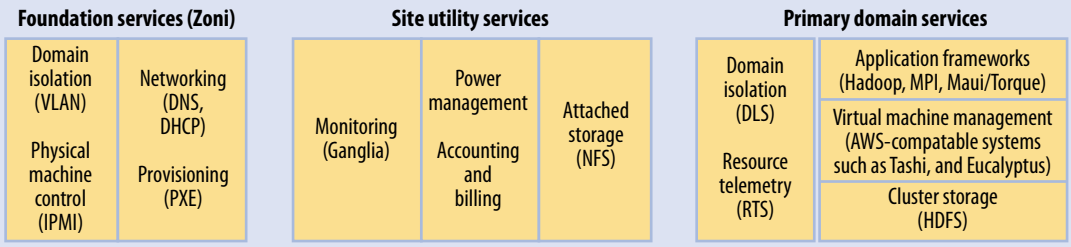

Figure 3. Open Cirrus site services. A typical site consists of foundation, utility, and primary domain services.

periments that belong to this latter group must be isolated from other activities in the cluster to maintain both experimental integrity and cluster stability. Node allocations isolated from the rest of the cluster in this way are domains. The current implementation creates domains by programming the cluster switches to create VLANs.

To bootstrap a domain for remote users, Zoni must also provide a mechanism for provisioning software onto at least one of the allocated nodes. This feature is currently implemented through a process based on Preboot Execution Environment (PXE) booting. However, even though the software may be provisioned properly, there is no guarantee that it will behave as intended or boot at all. Consequently, Zoni also provides users with out-of-band debugging and management facilities. Currently, the key debugging facility is remote console access, and the management facility consists primarily of remote power control. Both are provided through the Intelligent Platform Management Interface (IPMI).

Primary domain services. Naturally, not all cluster users are interested in managing a Zoni domain; some are interested in developing higher-level services, and some are interested in simply using the services offered. To serve these last two user groups, one domain in each site is designated the *primary domain* and provides a stable set of services for production use.

To support users working with very large data sets, a cluster storage system, in particular the Hadoop file system (HDFS), is used to aggregate the storage of all the nodes in the domain. A key property of HDFS is that it supports location-aware computing—that is, the file system exports the location of data blocks in such a way that runtime services can schedule a computing task that processes a block on the node that contains that block. This capability reduces pressure on the cluster network.

To support a diverse set of user needs, the recommended primary domain services include a virtual machine management (VMM) layer, which provides a convenient mechanism for allocating resources to various users and services. Hadoop, for example, implements a map/reduce programming paradigm and offers a popular runtime system for building cloud services. The Maui Cluster Scheduler combined with the Torque resource manager is another popular runtime system that provides a job submission service. By providing the two services through a disjoint set of virtual machines, the primary domain's administrator can dynamically adjust the pool of resources available to each service according to need simply by adding or removing VMs. Of course, users whose jobs aren't accommodated by one of the existing application framework services may request a private pool of VMs.

Different sites may select any VMM service as long as it supports the EC2 interface from Amazon Web Services (AWS). Good candidates include Tashi and Eucalyptus. Tashi is an open source cluster management system for cloud computing on massive Internet-scale data sets being developed as an Apache Incubator project; it is designed specifically to complement Open Cirrus. While Tashi is similar to other systems that manage logical clusters of VMs, it was developed to support research in coscheduling computation, storage, and power.

Coscheduling of computation and storage using the location information provided by storage systems such as Hadoop must overcome additional challenges in a virtualized environment. In particular, instances of the Hadoop runtime executing within VMs may be unable to correctly assess the relative distances to data blocks whose locations HDFS described with nonvirtualized location information. Two additional services may overcome this problem: The Data Location Service (DLS) is a clearinghouse for data location information independent of a storage mechanism, and the Resource Telemetry Service (RTS) provides a means to obtain an abstract distance measure between two location identifiers.[1]

Example. Figure 4 illustrates how Open Cirrus services fit together. In this example, the cluster is partitioned into four domains. From left to right, the first domain is used for low-level systems research, where researchers install their own OS kernels and run their own experimental codes and services. The second domain runs a VMM system that provides users with virtual clusters of VMs that share physical nodes and storage in the Zoni domain. Users build their own services and applications on top of these virtual clusters. The third domain provides a dedicated storage service that applications running on the

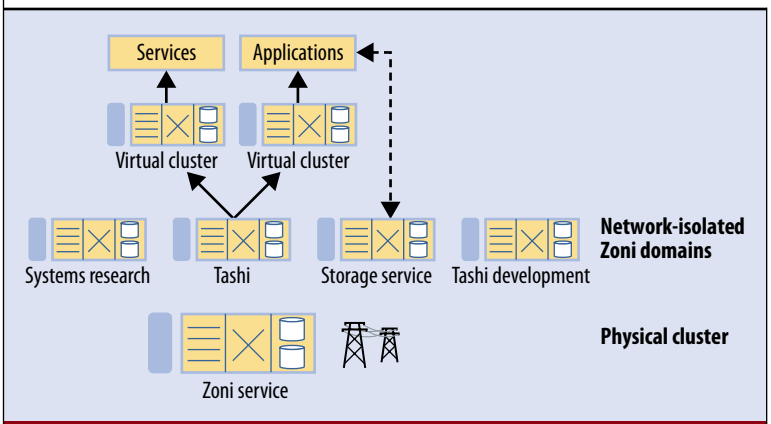

Figure 4. Example service hierarchy possible in Open Cirrus, with the cluster partitioned into four domains.

second partition use. The fourth domain offers a sandbox for developing the next version of the Tashi cluster management component.

Site utility services. To manage an Open Cirrus site easily, many additional, less critical services are required. For example, a monitoring service such as Ganglia not only enables the site administrator to monitor the cluster's health, it also facilitates collection of cluster operational data that may inform future research projects. Some conventional network file system storage is convenient for storing user scripts, small data sets, and small output files. Site utilities also include facilities for tracking resources consumed by users and managing the cluster's power consumption.

Open Cirrus federation

The Open Cirrus testbed provides a unique opportunity for experimenting with issues involving federation in cloud computing. To support these experiments, developers are adding a small set of "global" services—common services that run at participating sites to provide a common infrastructure.

Single sign-on. While users are granted access to each site separately, this authentication clearinghouse service enables users to access all granting sites with the same credentials.

Global monitoring. By collecting the monitoring output from each site into a central location, administrators can quickly assess the health of the entire testbed, users can determine which sites are busy, and cross-site statistics can be archived for future research.

User directories. Mounting a user directory in a common location at each site facilitates the distribution of small files, such as execution scripts and configuration files, for conducting experiments.

Global storage. Similarly, some data sets may be made available across all sites, either through replication or some other distribution technology, as appropriate.

Scheduling. Ultimately, users may find it convenient to submit a job without having to specify where the job must run. A global scheduling component could automatically dispatch jobs to the optimal cluster according to some criteria: load balancing, minimizing data movement, minimizing energy consumption, and so on.

Resource tracking. As jobs are scheduled across multiple sites, the resources that users consume may need to be tracked and credits exchanged between sites based on resource consumption.

RESEARCH USING OPEN CIRRUS

Table 1 summarizes basic characteristics of the current Open Cirrus sites. More important than physical characteristics are the types of collaboration that Open Cirrus enables.

Approximately 100 research projects at 10 sites use Open Cirrus at the systems and applications levels.

Systems-level projects include robust adaptive routing over redundant layer-2 networks, data computation overlay for aggregation and analyses, and Open Cirrus and PlanetLab federation (at HP); use of optical switches to break data-center networking bottlenecks, power-aware workload scheduling, and optimizing service energy-delay product (at Intel); applying RAID techniques to HDFS, pipelining data between map and reduce stages of Hadoop jobs to improve user interaction, and deploying log-analysis techniques to improve performance of Hadoop clusters (at Yahoo!).

Applications-level projects include computerized language translation, astrophysics, graph data mining, and computer design simulation (at CMU); real-time streaming applications such as gesture recognition from video and simulation of material that changes its shape under software control (at Intel); DNA sequencing, search, and annotation (at ETRI); continuously extracting knowledge from webpages, natural-language processing, large-scale graph algorithms, analysis of Wikipedia group dynamics, statistical machine translation, large-scale document analysis, statistical machine-learning algorithms, and computational sustainability (at Yahoo!).

In most of these research projects, Open Cirrus offers the benefits of running experiments at scale; leveraging commonly shared stack, services, and best practices; and creating synergies across the layers. Systems-level research—for example, networking, power, and cooling—leverages applications to better understand the cloud load, while applications improve in performance and scalability using results from the underlying systems research.

Table 1. Characteristics of current Open Cirrus sites.

Site	Cores	Servers	Public partition	Memory size (Tbytes)	Storage size (Tbytes)	Spindles	Network data rate	Focus
CMU	1,165	159	50	2.40	892	784	1 Gbps	Tashi, distributed file systems, applications/data sets
ETRI	1,024	256	200	0.50	128	256	1 Gbps	Large data sets, cloud infrastructure
HP	1,024	256	178	3.30	632	1,152	10 Gbps internal; 1 Gbps x-rack	Networking, federation
IDA	2,400	300	100	4.80	59+	600	1 Gbps	Applications based on Hadoop, Pig
Intel	1,364	198	198	1.77	610	746	1 Gbps	Tashi, Zoni, MPI, Hadoop
KIT	2,048	256	128	10.00	1,000	192	1 Gbps	Applications with high throughput
MIMOS	1,024	167	16	0.50	36	258	1 Gbps	Platform, tools, testing, security
UIUC	1,024	128	64	2.00	500	258	1 Gbps	Data sets, cloud infrastructure
RAS	1,136	142	600	9.10	36	142	1 Gbps	Hadoop, Tashi, Zoni, Pig, MPI
Yahoo!	3,200	480	400	2.40	1,200	1,600	1 Gbps	Hadoop, Pig

OPEN CIRRUS ECONOMIC MODEL

The emergence of each individual site in Open Cirrus and the expected growth of the federation are driven by the economy in today's cloud computing environment. Explicit break-even points for the choice between renting versus owning a cloud infrastructure implicitly justify Open Cirrus' economic rationale.

Single site

Consider a medium-sized organization—for example, a start-up or a university department—seeking to provide a Web service to a client population. The service will run in a cloud, accessing stored data and consuming CPU cycles. Suppose this service is identical to the UIUC Open Cirrus site: 128 servers (1,024 cores) and 524 Tbytes. Should the organization rent the infrastructure from a cloud provider, such as EC2 (Elastic Compute Cloud) and S3 (Simple Storage Service) from Amazon Web Services, or should it buy and maintain a cloud?

At average AWS rates of US$0.12 per Gbyte/month and $0.10 per CPU-hour, renting the cloud infrastructure would incur a monthly storage cost of 524 × 1,000 × $0.12, or $62,880; the total monthly cost would be $62,880 + 1,024 × 24 × 30 × $0.10 = $136,608. In the case of owning a cloud, amortized monthly costs would be split among the hardware, power, and network 45 percent, 40 percent, and 15 percent, respectively.[2-4] If the service lifetime is M months, it would incur a monthly storage cost, assuming $300 1-Tbyte disks and scaling for power and networking, of 524 × $300/0.45/$M$, or $349,333/$M$; the total monthly cost, based on actual systems cost and the salary of one system administrator for about 100 servers,[3,4] would be $700,000/0.45/$M$ + $7,500, or $1,555,555/$M$ + $7,500.

The break-even point for storage would be $349,000/$M$ < $62,880, or M > 5.55 months; the overall break-even point would be $1,555,555/$M$ + $7,500 < $136,608, or M > 12 months. Thus, if the service runs for more than 12 months, owning the cloud infrastructure is preferable to renting it. Similarly, it's better to own storage if you use it for more than six months.

Clouds are typically underutilized.[2] With X percent resource utilization, the break-even time becomes 12 × 100/X months. Given the typical hardware lifetime of 36 months, the break-even resource utilization is 12 × 100/X < 36, or X > 33.3 percent. Even at the current 20 percent CPU utilization rates observed in industry, storage utilization greater than 47 percent would make ownership preferable, as storage and CPU account evenly for costs.

Federated sites

Federation can help absorb overloads due to spikes—for example, at conference deadlines—or underprovisioning.[2,4] Figure 5 plots the costs incurred by a single underprovisioned cloud for three options: offloading only to AWS, offloading to five federated clouds and AWS, and offloading to 49 federated clouds and AWS.

A federation of six sites can defer costs up to 250 percent overload, while with 50 sites the break-even point is roughly 2,500 percent. (This assumes that other sites are utilized 50 percent and not idle; otherwise, the break-even point would be 500 percent and 5,000 percent, respectively). The detailed data and spreadsheet for this calculation are available at http://opencirrus.org. Note that this is only a starting step—the calculation can be expanded by accounting for the economic costs of disasters, such as massive failure, project cancellation, and start-up time.

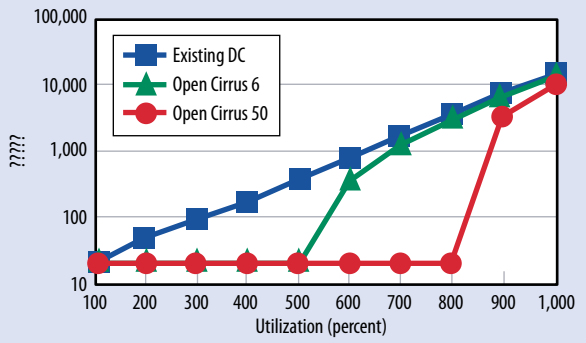

Figure 5. Costs incurred by a single underprovisioned cloud for three options: offloading only to Amazon Web Services (existing DC), offloading to five federated clouds (Open Cirrus 6) and AWS, and offloading to 49 federated clouds (Open Cirrus 50) and AWS.

RELATED WORK

We can broadly divide existing cloud computing testbeds into those that mainly support applications research and those that can support systems research. Table 2 compares the most prominent testbeds.

The Google/IBM cluster (www.google.com/intl/en/press/pressrel/20071008_ibm_univ.html), TeraGrid,[5] and Microsoft Windows Azure platform (www.microsoft.com/windowsazure) all focus on supporting computing applications research. Thus, these testbeds don't enable access to bare-metal hardware or root access to the OS; instead, services such as MPI and Hadoop are installed to ease access to the resources. For example, the Google/IBM cluster is configured with the Hadoop service and targets data-intensive applications research such as large-scale data analytics. TeraGrid is a multisite infrastructure mainly used for scientific research, and Microsoft provides a Windows Azure interface but no access to systems-level data.

The Open Cloud Testbed (www.opencloudconsortium.org) focuses on cloud computing middleware research and is currently configured as a smaller-scale testbed with four 32-node sites.

Testbeds such as PlanetLab,[6] Emulab,[7] DETER (cyber-DEfense Technology Experimental Research),[8] and Amazon EC2 (http://aws.amazon.com) are designed to support systems research but with diverse goals.

PlanetLab consists of a few hundred machines spread around the world and is mainly designed to support wide-area networking and distributed systems research. Although it doesn't provide access to bare-metal hardware, it does provide root access to the OS through a lightweight virtualization similar to FreeBSD jails.

EmuLab, the original Zoni service, is a single-site testbed where each user can reserve a certain number of machines (typically a few tens) and get exclusive access to bare-metal hardware. Emulab also provides mechanisms to emulate different network characteristics. Open Cirrus provides Emulab-like flexibility for systems research with federation and heterogeneity, which are crucial for cloud computing.

Table 2. Comparison of cloud computing testbeds.

Characteristics	Open Cirrus	Google/ IBM cluster	TeraGrid	PlanetLab	Emulab	Open Cloud Testbed	Amazon EC2	LANL cluster
Type of research	Systems and services	Data-intensive applications	Scientific applications	Systems and services	Systems	Interloper across clouds using open APIs	Commercial use	Systems
Approach	Federation of heterogeneous data centers	Cluster supported by Google and IBM	Multisite heterogeneous clusters for super-computing	Nodes hosted by research institution	Single-site cluster with flexible control	Multisite heterogeneous clusters	Raw access to virtual machines	Reuse of LANL's retiring clusters
Participants	CMU, ETRI, HP, Intel, IDA, KIT, MIMOS, RAS, UIUC, Yahoo!	Google, IBM, MIT, Stanford Univ., Univ. of Washington	Many universities and organizations	Many universities and organizations	Univ. of Utah	Four centers	Amazon	CMU, LANL, NSF
Distribution	10 sites	Central-ized—one data center in Atlanta	11 partners in US	More than 700 nodes worldwide	More than 300 machines	480 cores distributed in four locations	Several unified data centers	Thou-sands of older but still useful nodes at one site

The DETER testbed is an installation of the Emulab software and is mainly used for security research—for example, colleting a large-scale worm trace. Consisting of two heterogeneous sites, DETER may be viewed as a federated Emulab installation. However, the two sites are tightly coupled: The controller resides in one site and controls physical resources in both sites. In Open Cirrus, all sites are loosely coupled.

Amazon EC2 provides VMs on a pay-as-you-go basis. Although it allows complete control over the VMs, users can't control network resources, reducing its flexibility as a systems research testbed.

Finally, CMU's Garth Gibson is leading an effort to recycle Los Alamos National Laboratory's retiring clusters (typically with a few thousand machines) by making them available for systems research.

Other related cloud computing efforts such as Reservoir (http://sysrun.haifa.il.ibm.com/hrl/reservoir) and RightScale (www.rightscale.com) aren't proper testbeds. Reservoir (Resources and Servers Visualization without Barriers) is an EU-funded grid computing project that enables massive-scale deployment and management of IT services across administrative domains. RightScale is a cloud services management platform.

Open Cirrus offers unique opportunities for cloud computing research that no existing alternatives offer. It federates heterogeneous sites, systems and applications research, and data sets. In addition, it provides an open source service stack with nonproprietary APIs for cloud computing. And through shared innovation, Open Cirrus will have a greater impact on research communities around the globe.

While working on Open Cirrus during the past year, we realized the value of common stacks and services, but even more, we came to appreciate the benefits of a research community working together toward the same goals. Heterogeneity of the individual sites has contributed to the diversity of solutions and has strengthened our approaches, even though it does make global services more complex to develop, deploy, and maintain.

Future work on Open Cirrus will revolve around increased use across the sites. In particular, we're exploring applications that can be used on multiple sites to increase their performance, scale, and reliability. Another area of interest is standards, but we need more experience in using stacks and services before exploring standardization. C

Acknowledgments

Partial funding for the Open Cirrus UIUC site was provided by NSF. Funding for the Open Cirrus CMU/PDL site was provided by Intel, NSF, ARO, and the PDL Consortium. Open Cirrus is a trademark of Yahoo! Inc. Several people made significant contributions to Open Cirrus, including A. Chien, R. Gass, K. Goswami, C. Hsiung, J. Kistler, M. Ryan, C. Whitney, and especially J. Wilkes, who was instrumental in formulating the original Open Cirrus vision as well as the concept of a management layer like Zoni.

References

1. M.A. Kozuch et al., "Tashi: Location-Aware Cluster Management," *Proc. 1st Workshop Automated Control for Datacenters and Clouds* (ACDC 09), ACM Press, 2009, pp. 43-48.
2. M. Armbrust et al., "Above the Clouds: A Berkeley View of Cloud Computing," tech. report UCB/EECS-2009-28, EECS Dept., Univ. of California at Berkeley, 2009; www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf.
3. J. Hamilton, "Internet-Scale Service Efficiency," keynote at 2nd Large-Scale Distributed Systems and Middleware Workshop (LADIS 08), 2008; http://mvdirona.com/jrh/TalksAndPapers/JamesRH_Ladis2008.pdf.
4. A. Greenberg et al., "The Cost of a Cloud: Research Problems in Data Center Networks," *ACM SIGCOMM Computer Communication Rev.*, Jan. 2009, pp. 68-73.
5. C. Catlett et al. "TeraGrid: Analysis of Organization, System Architecture, and Middleware Enabling New Types of Applications," *High Performance Computing and Grids in Action*, L. Grandinetti, ed., IOS Press, 2007, pp. 225-249.
6. L. Peterson et al., "A Blueprint for Introducing Disruptive Technology into the Internet," *ACM SIGCOMM Computer Communication Rev.*, Jan. 2003, pp. 59-64.
7. B. White et al., "An Integrated Experimental Environment for Distributed Systems and Networks," *ACM SIGOPS Operating Systems Rev.*, winter 2002, pp. 255-270.
8. T. Benzel et al., "Design, Deployment, and Use of the DETER Testbed," *Proc. DETER Community Workshop Cyber Security Experimentation and Test*, Usenix Assoc., 2007; www.isi.edu/deter/docs/200708-usecdw-deter-design-deploy.pdf.

Arutyun I. Avetisyan *is deputy director of the Institute for System Programming of the Russian Academy of Sciences (ISP RAS), RAS's representative in Open Cirrus. Contact him at arut@ispras.ru.*

Roy Campbell *is Sohaib and Sara Abbasi Professor in the Department of Computer Science at the University of Illinois at Urbana-Champaign (UIUC). Contact him at rhc@illinois.edu.*

Indranil Gupta *is an associate professor in the Department of Computer Science at UIUC, where he also heads the Distributed Protocols Research Group. Contact him at indy@illinois.edu.*

Michael T. Heath *is a professor and Fulton Watson Copp Chair in the Department of Computer Science at UIUC, where he's also director of the Computational Science and Engineering Program. Contact him at heath@illinois.edu.*

Steven Y. Ko *is a postdoctoral research associate in the Department of Computer Science at Princeton University who worked on Open Cirrus as a PhD student at UIUC. Contact him at steveko@cs.princeton.edu.*

Gregory R. Ganger *is a professor in the Department of Electrical and Computer Engineering at Carnegie Mellon University (CMU), where he's also director of the Parallel Data Lab. Contact him at ganger@ece.cmu.edu.*

Michael A. Kozuch *is a principal researcher for Intel Labs Pittsburgh working in the area of computer architecture and system software. Contact him at michael.a.kozuch@intel.com.*

David O'Hallaron *is the director of Intel Labs Pittsburgh and an associate professor in the Department of Computer Science and the Department of Electrical and Computer Engineering at CMU. Contact him at david.ohallaron@intel.com.*

Marcel Kunze *heads the Department of Integration and Visualization at the Steinbuch Centre for Computing, Karlsruhe Institute of Technology, Germany, and is a technical lead in Open Cirrus. Contact him at marcel.kunze@kit.edu.*

Thomas T. Kwan *is director of research operations at Yahoo! Labs, where he manages multiple cloud computing research partnerships for Yahoo!. Contact him at tkwan@yahoo-inc.com.*

Kevin Lai *is a research scientist in the Social Computing Lab at HP Labs and has worked on various projects at the intersection of operating systems and incentive engineering. Contact him at klai@hp.com.*

Martha Lyons *is a Distinguished Technologist at HP Labs focused on incubating and delivering innovative capabilities to HP's customers. Contact her at martha.lyons@hp.com.*

Dejan Milojicic *is a scientist and senior researcher manager in the Strategy and Innovation Office at HP Labs and director of Open Cirrus. Contact him at dejan.milojicic@hp.com.*

Hing Yan Lee *is program director of the National Grid Office at the Infocomm Development Authority (IDA) of Singapore. Contact him at lee_hing_yan@ida.gov.sg.*

Ng Kwang Ming *is director of the Grid Computing Lab at the Malaysian Institute of Microelectronic Systems (MIMOS). Contact him at kwang.ming@mimos.my.*

Jing-Yuan Luke *is a staff engineer in the Grid Computing Lab at MIMOS. Contact him at jyluke@mimos.my.*

Han Namgong *is a director in the Cloud Computing Research Department of the Software Research Laboratory at the Electronics and Telecommunications Research Institute (ETRI), South Korea. Contact him at nghan@etri.re.kr.*

Yeng Chai Soh *is a professor in the School of Electrical and Electronic Engineering at Nanyang Technological University, Singapore. Contact him at eycsoh@ntu.edu.sg.*