Cloud Management: Challenges and Opportunities

Tim Forell, Dejan Milojicic, Vanish Talwar
Hewlett-Packard
[first.lastname]@hp.com

***Abstract*—Cloud computing offers on-demand access to shared resources and services, hosted in warehouse sized data centers at cloud providers. Effective management of these shared resources and services is one of the key requirements for the delivery of cloud computing. However, there are several challenges to achieve effective cloud management. These include scale, multiple levels of abstraction, federation, sustainability, and dynamism.**

In this paper, we outline these challenges, and then describe specific examples of new management architectures that address these challenges. We focus on driving principles for the new designs, and give an illustration of its deployment on OpenCirrus - a research cloud testbed. Overall, the paper provides open issues and challenges in cloud management for the research community to address.

I. INTRODUCTION

Cloud computing is an emerging paradigm, with growing popularity and adoption [1]. Cloud providers host shared servers, and deliver computing, storage, and software to end-consumers as a service. Both Gartner and IDC have estimated healthy growth of cloud computing adoption [2][3]. Cloud services include compute-on-demand, online storage, online/shared office, key value store service, and email, among many others. Examples of public cloud providers are Amazon AWS [4], GoGrid [5], and Rackspace [6]. Several other companies have cloud offerings, such as HP [7], Google [8], IBM [9], and Microsoft [10]. In addition, a number of open source efforts are under way, such as Eucalyptus [11] and Open Nebula [12] as well as numerous research efforts, such as Tashi [13], RESEVOIR [14], and Open Cirrus [15].

Cloud computing is enabled by advances in virtualization, service oriented computing, and utility computing. There are several requirements for cloud computing to be successful. These include low-cost, SLA compliance, security guarantees, high availability, energy efficiency, and accurate accounting. The key to meeting these requirements is effective management of the cloud resources and services. This covers all aspects of the data center lifecycle from bring-up, provisioning, scheduling, monitoring, failure management, and shutdown.

Traditional management solutions are typically centralized, working best at limited scale. They are also silo-ed, with independent controllers for managing applications, virtual machines, platforms, and networks. Large scale, multi-tenant environments, and heterogeneous resources provide challenges for management and monitoring tasks in clouds. Furthermore, multiple data centers and emergence of hybrid clouds demand federated management capabilities for cloud systems.

In this paper, we outline these specific challenges for management solutions in Cloud environments. Specific opportunities identified are those due to scale, multiple abstraction levels, dynamism, and federation. We also give specific examples of new management architectures that address these challenges. These architectures cover the topics of provisioning, monitoring, and sustainability. Finally, we present other open issues for the community to address.

The rest of the paper is organized as follows. In Section II, we provide background on clouds and present challenges for cloud management. Section III describes the OpenCirrus cloud research testbed. Section IV, V, and VI provide principles for design of new management architectures with examples of node reservation, monitoring, and sustainability. Section VII provides a summary of other issues for the community to address, and Section VIII concludes the paper.

II. CLOUD MANAGEMENT

A. Background

Clouds are designed as a layered architecture (see Figure 1, left). Specifically, the cloud architecture consists of Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) layers. IaaS refers to the computing and storage infrastructure provided as a service, PaaS refers to the platform and/or solution stack as a service, SaaS refers to delivering software as a service over the Internet.

Such cloud architectures can be deployed as private, public, or hybrid clouds. Public clouds are provisioned on the Internet. Private clouds are provisioned in-house within enterprises. Hybrid clouds are a combination of public and private clouds (see Figure 1, right).

In all these deployments, a key requirement is efficient management of resources at the various layers of the stack crossing multiple metrics of system operation – performance (SLA compliance), resiliency, and power. This includes provisioning, scheduling, monitoring, failure management, performance optimization, and energy management.

Traditional monitoring and management technologies were developed for enterprise environments, and are typically catered to single customer environments deployed in the range

1530-2075/11 $26.00 © 2011 IEEE
DOI 10.1109/IPDPS.2011.233

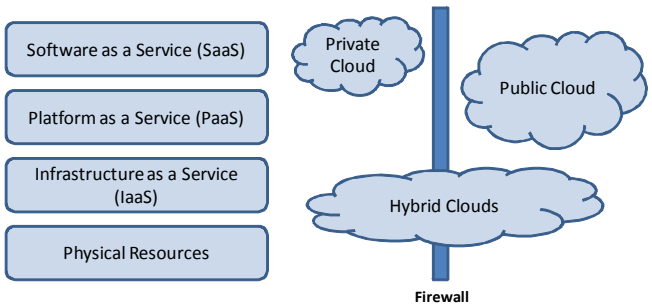

Figure 1. Cloud Architecture and Deployment modes

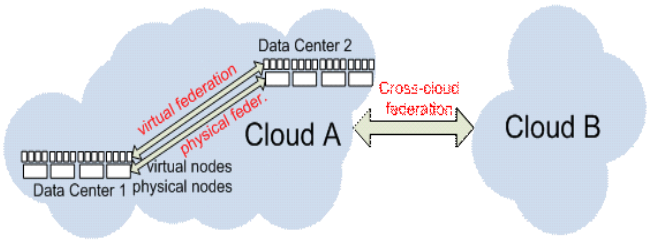

Figure 2. Conceptual Diagram of Federations Types

of thousands of servers. Cloud systems represent multi-tenant environments, and provide computing systems and applications to consumers as a service. These systems are in the range up to hundred thousand servers, serving millions of users. Existing cloud deployments mostly rely on traditional monitoring and management systems not catered to all of cloud needs. Several open research challenges remain to design fully automated, closed-loop management solutions for the cloud meeting customer service level agreements (SLAs) and cloud provider cost metrics. Some of these key challenges are summarized next.

B. Challenges in Cloud Management

Scale: Managing millions of cores leads to unsustainable administrator costs, requiring automated methods for typical system management tasks. These automated methods have to deal with increased monitoring data sizes of several orders of magnitudes higher than current systems. They are collected across multiple layers of the stack, and at different time intervals. Provisioning, deployment, and scheduling methods too have to work across scale of several orders of magnitude larger than current systems. Traditional monitoring and management systems are typically centralized. These approaches won't scale to potentially millions of management objects in cloud systems. Approaches that are more distributed and have scalability properties that allow easy scale-up and scale-down of the monitoring and management systems to elastically meet cloud requirements are needed.

Furthermore, failure management and performance optimization techniques need scalable and intelligent methods to analyze the large amounts of monitoring data, and to apply smart optimizations to mitigate problems and ensure dynamic resource management in the presence of frequent component failures. Finally, at this level of scale, different system components operate at multiple time scales, further adding to the management challenges.

Multiple abstraction levels and resource types: The separation of functionality across IaaS, PaaS, and SaaS layers provides a good abstraction for cloud systems, however, having separate management systems for these layers results in silo-ed operations and subsystem level solutions. This leads to inefficiency and redundancy, requiring better coordination and exchange of information across the layers. Higher layer SLAs need to be effectively translated to lower layers to make management decisions at its own layer SLA-aware. Similarly, lower layer abstractions can be exposed as appropriately to higher layers in the stack to make them more effective for the cloud operation.

Furthermore, with the emergence of various data intensive cloud services, the cloud resource management systems need to effectively manage nontraditional resources such as I/O bandwidth to meet the demands of data movement. Multiple actuator knobs across the multiple resource types now need to be coordinated for effective actuation of optimization decisions.

Federation: No Cloud provider has unlimited capacity and in case of significant customer demand may have to overflow to another provider. This can take place between private and public clouds, or among private clouds or public clouds. Such federated sites enable much larger pool of resources and therefore more flexible and cheaper use of resources. However, federation creates several challenges for management systems. Monitoring and management techniques would have to be extended across multiple clouds, but still need to provide a single abstraction view to the applications and consumers. Multiple management tools at different clouds sites or providers can add to the complexity, and common standard interfaces are needed to enable federated management. Security and authentication are other concerns in federated environments. Federation can also take place with different patterns: physical resources and virtual resources at same and different sites and federation of different types of Clouds (see Figure 2). The management software in Clouds needs to consider these various types.

Federation is a non-trivial problem and previous systems such as AFS [16], DCE [17], and many others [18], [19], [20], [21], underwent significant development efforts with limited adoption. However, as cloud computing takes off, and the number of providers increases, the need for cross-Cloud usage will increase, and federated management will be a key requirement. Lessons learned from previous systems need to be applied for federated cloud management. A simple scenario of Cloud burst is adopted in nowadays Clouds, but as Cloud computing evolves more general use models will evolve.

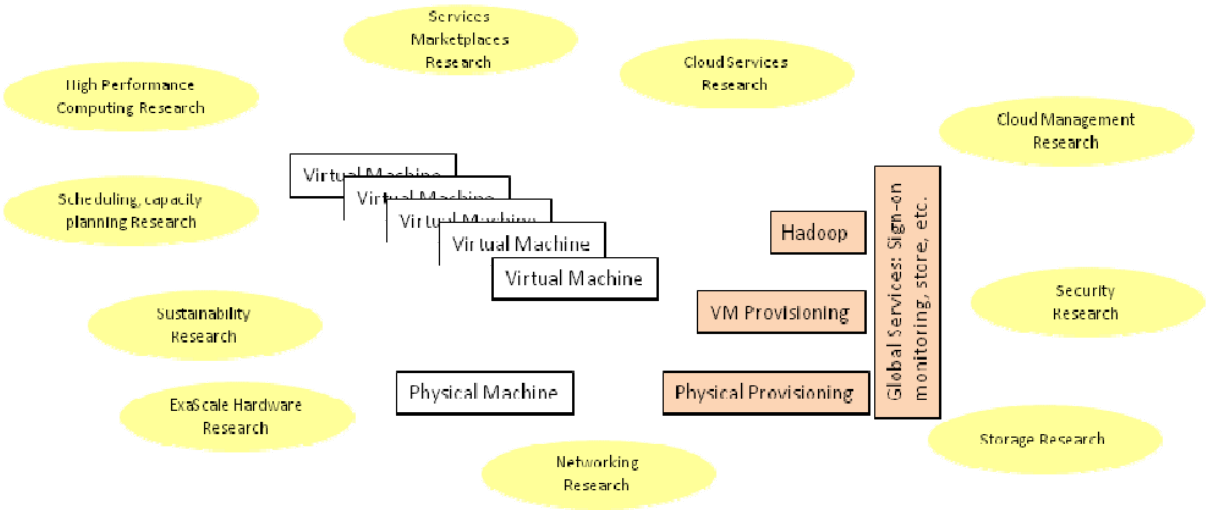

Figure 3. OpenCirrus Node Architecture

Energy Efficiency: Given the scale and scope of cloud data centers, energy efficiency is a critical requirement to meet cost, regulations, and environmental constraints. Power and cooling system is a growing fraction of the costs in data centers, and techniques that minimize power consumption, and use intelligent mechanisms for cooling are needed. In addition, the problem needs an end-to-end perspective wherein sustainability needs to be driving factor during the manufacturing of cloud servers as well as day-to-day operations of cloud data centers.

Further, energy and power has to be managed across multiple heterogeneous processors, and policies coordinated across the hardware and software (OS) layers to balance performance and power, as well as to efficiently utilize multiple actuators. New hardware architectures will enable turning of different parts of system and increase savings opportunity but also introduce additional management complexity. Examples include balancing the loads across servers to accomplish different tradeoffs of performance/power or turning off individual servers or even racks. This requires moving the load leading to the next challenge, dynamism.

Dynamism: Cloud systems have high churn due to continuous arrival and departure of workloads. With virtualization, workloads can also migrate across the data center. This presents challenges for management software so as to ensure stability in the data center and to ensure low overheads. It also implies that monitoring and analysis methods have to deal with limited knowledge of past behavior to infer the state of operation of the systems.

Next, we present some example management architectures to address a few of these research challenges in the context of OpenCirrus, a cloud computing research testbed.

III. OPENCIRRUS

Open Cirrus is a Cloud Computing Testbed consisting of 14 geographically distributed sites, each comprised of at least 1,000 cores and accompanying memory and storage [15]. Each site is managed independently and the overall testbed is therefore a federation of heterogeneous sites. This was a design choice made initially to enable more comprehensive research in Cloud Services and in particular management. OpenCirrus envisions that, just like Internet today, the future of Cloud Computing will consist of multiple Cloud providers contributing to a seemingly single ubiquitous Cloud that users can get computation, storage, as well as many services from. From today's state of the art, to this vision, there are numerous research questions that need to be addressed and technologies developed. Open Cirrus was created to help with this kind of research.

OpenCirrus is used by several research projects in areas, such as networking, sustainability, exascale computing, storage, service composition, security, etc. Some of these projects require access to the hardware or the systems software of the underlying machines and therefore, it exposes physical machines to the users. This is in contrast to typical cloud offering, where only virtual machines are made available to users. Several administrative tools/services are needed to support the maintenance of the resources at the physical level. In the remaining text, we will address some of these services, in particular Node Reservation System, Scalable Monitoring, and Cloud Sustainability Dashboard. They will motivate requirements for the Research Cloud Computing Testbed.

Figure 3 describes architectural components of Open Cirrus site (Open Cirrus Cloud Computing Stack, and global services, in orange), as well as kinds of research it is currently being used for in HP Labs. Open Cirrus enables a consolidated infrastructure of credible scale for conducting experiments, as well as consolidated users that can result if there is sufficient variety of loads. At the bottom of the stack, the networking researchers can get traces from the testbed, followed by

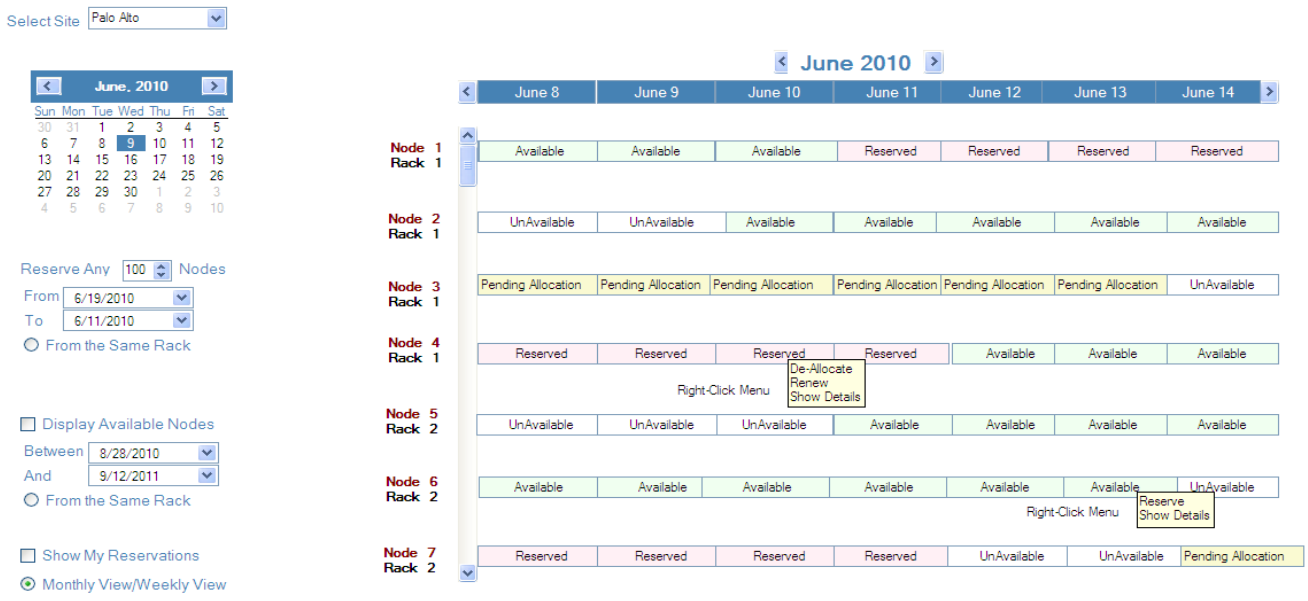

Figure 4. Node Reservation System GUI, currently being developed at HP Labs

hardware designers who can understand requirements and behavior of Cloud services, followed by management research, which can better understand typical processing, memory, storage stack, followed by services research, etc.

This type of testbed use results in the need to change and interpose at different levels of stack, such as making changes to the wiring (networking), operating system (hardware research), default monitoring configurations, scheduling, and consequently requires *provisioning of both virtual machines and physical nodes* (former is usual case in cloud computing, latter typical of systems research, such as Emulab). The second requirement, *support for heterogeneity,* derives from the cross regional, global testbeds, differing in terms of export and privacy rules, and preventing from ultimate automation. Because each of the regions (America, EU, Asia-Pacific) has different privacy and export rules, not necessarily better or worse, just different, there is a need for human verification of the users prior to being awarded requested resources. The third requirement derives from the previous ones, which is *the need for federation*. Federation is accomplished at multiple levels, first at the security level, through the global sign on (as well as other services, such as global monitoring), next at the physical provisioning (by extending node reservation system), virtual provisioning (by federating Eucalyptus and eventually Tashi), and individual services level, such as Cloud Sustainability Dashboard.

IV. NODE RESERVATION

While Cloud Computing promises unlimited resources, in practice every Cloud provider has limited hardware, which it shares among the users. In case of a research testbed, the limits are obvious. A typical Open Cirrus site has at minimum 1,000 cores and overall testbed has over 15,000 cores. Many research experiments require significant number of cores to conduct experiments. Node reservation system enables users reserved (guaranteed) access to requested hardware of desired configuration (built-in search for specific resources) and at requested time. See Figure 4 for the User Interface and Figure 5 for the implementation. Furthermore, this is a tool that enables federation at the physical node level, by enabling users to requests machines from other Open Cirrus sites. This service meets all requirements (physical provisioning, heterogeneity, and federation).

The Node Reservation System is designed and implemented in a similar way as other resource reservation systems, such as a room booker. It is aware of all resources in the Cloud instance (single or multi-site), it enables users to search for these resources, and then to make a reservation for a duration of the time. A user can also request recurrent reservations. Node Reservation System is integrated with the Open Cirrus security model, in particular with the user public key which is stored in the Open Cirrus portal. When a user makes reservation, the user's public key gets installed on the reserved nodes. When the reserved time is about to expire, the user is warned of expiration. At the expiration time, the access is forbidden by removing the public key, and after a grace period of time, the node in question is re-provisioned.

There are a couple of use cases for Node Reservation system in production environments. One use case addresses typical deadline situation, for example just prior to a conference submission or a project delivery, when a number of users try to use the Cloud resources. The other use case addresses the need for regular recurrent use of resources, such as the weekly classes with practical student assignments or day in a week reserved for customer demonstrations. In both use cases, if there are unlimited resources, user needs could be met. With limited resources, only a subset of users can be served, or

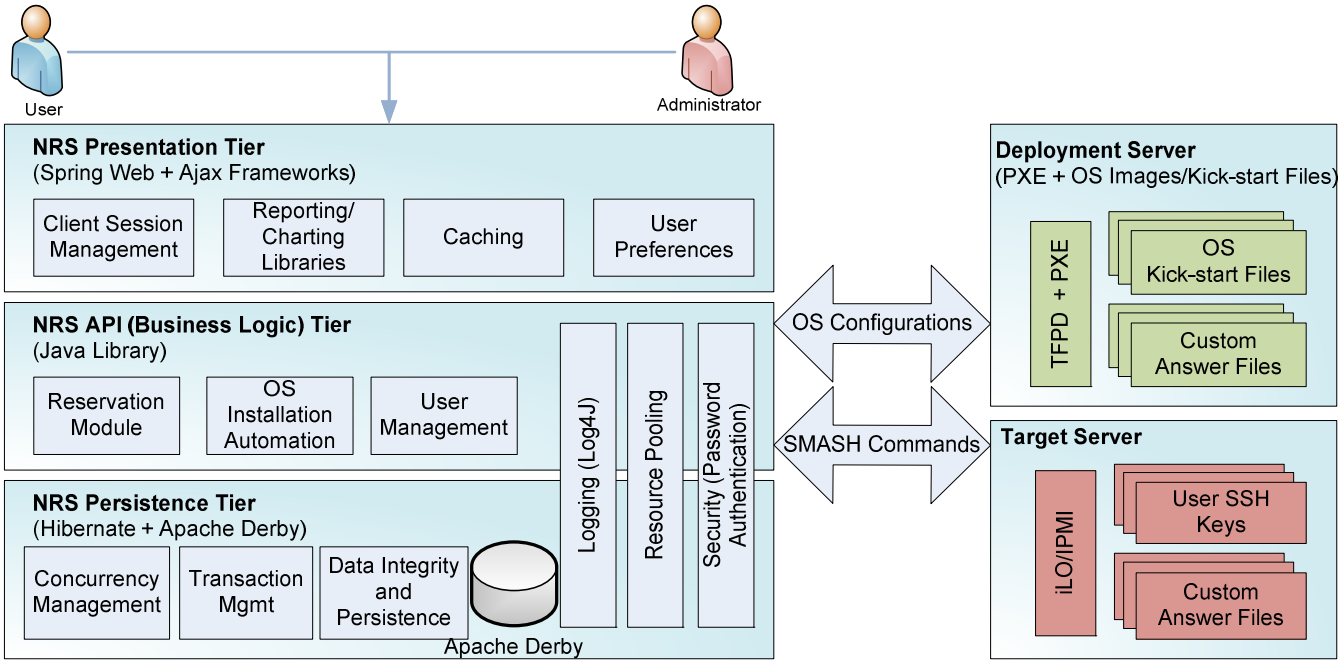

Figure 5. Node Reservation System Architecture

some of the users would be served with reduced performance guarantees.

There are other two uses of Node Reservation System of particular interest in research environments. The fist use is for high performance computing or other topology sensitive applications. Node Reservation System enables users to request nodes in specific enclosure, rack, or site. This way, applications can be configured with topology-awareness to take advantage of location. Components with intensive inter-communication can be co-located, while those which compete for resources (processing, memory, connectivity) can be distributed to different parts of the Cloud. Similarly, if there are specialized resources available in the Cloud (e.g. flash memory (non-volatile memory), GPUs (graphical processing units), or photonic interconnects, the search functionality of Node Reservation System can identify these resources to users. NRS is implemented using SMASH commands at the hardware level and PXE commands for the operating system reboot (see Figure 5).

V. SCALABLE MONITORING

Continuous, online monitoring and analysis is a key component for closed loop management in cloud data centers. Analysis of system monitoring data, e.g. utilization data, enables a better understanding of application and system behavior, helps meet application SLA requirements, and provides insights into assessing and managing data center systems. Existing, traditional approaches today use centralized techniques for data collection, aggregation and analysis across datacenter subsystems and machines. These existing solutions face several scalability challenges in cloud data centers. Large data center sizes would result in huge volume of monitoring data that would be produced across multiple management domains. Traditional approaches would face limitations in terms of:

- high load on central management servers (CMS),
- delayed response time to analysis of critical events, and
- reduced accuracy due to use of larger sampling frequencies to have data processing keep up with data generation.

Another key challenge is to adapt to dynamic change in usage, load, and configuration of data center machines. The monitoring and analysis infrastructure should exhibit small lag and hence predictable latencies under changes occurring due to increased load, or changes in system size, or configuration of data centers. Furthermore, it should also be resilient to failures of monitoring components in the data center.

A scalable monitoring and analytics infrastructure is being experimented on OpenCirrus to address these challenges, and is based on the following general principles [22] [23][24][25].

Data local analysis: With increased scale, monitoring data analytics has to be performed on large volumes of data. Moving data around in large scale can be inefficient; hence, it is advisable to instead move the monitoring analytics and management computation close to the source of monitoring data, i.e. local machines. Analytics can then operate in a distributed and parallel manner across the nodes, and actuation decisions can be made locally. Real-time decisions can thus be achieved with this approach [23].

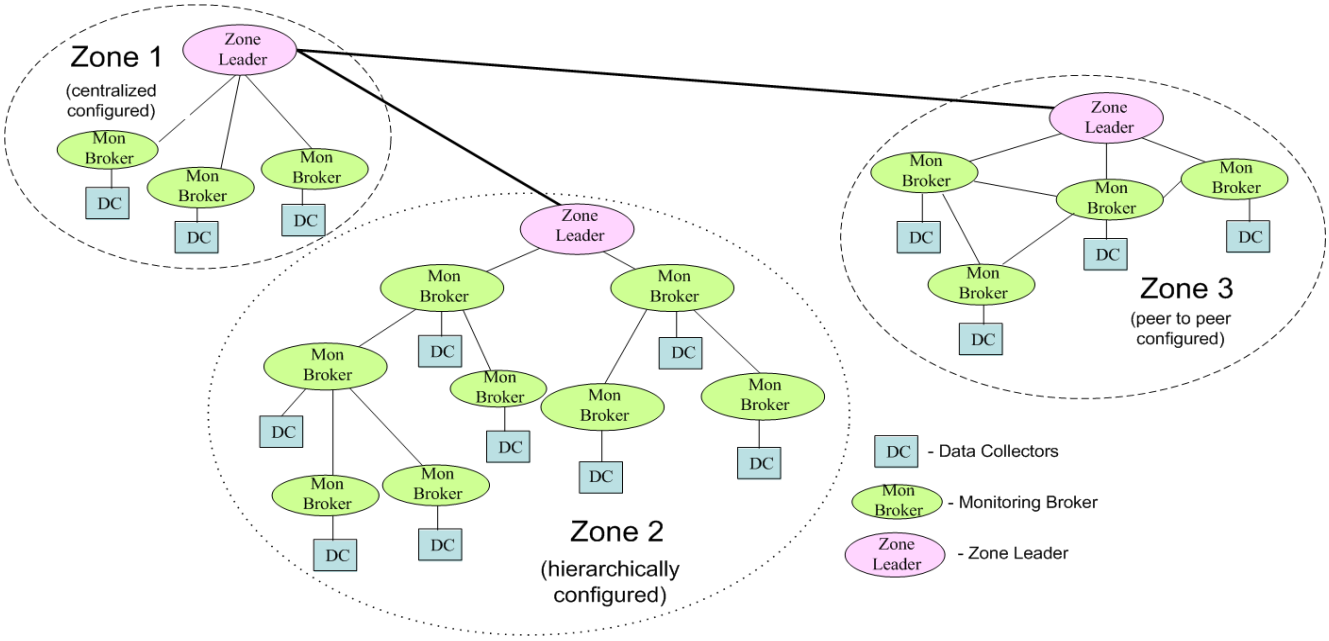

Figure 6. Distributed hybrid structures for scalable monitoring and analysis [23]

Flexible distributed architectures: Monitoring and management systems in cloud data centers have to deal with varying requirements of data analytics and decision making over 'time' and 'space'. Traditional centralized architectures cannot perform well across such large scale and with varying requirements. A distributed architecture is needed. Figure 6 is an illustration of such a distributed architecture. In addition, such distributed architectures must also be flexible to meet varying demands. Flexibility in 'time' can be achieved by enabling on-demand deployment of monitoring and analytics components on sub-set of machines. Flexibility in 'space' can be achieved by encompassing hybrid topology structures across different regions of the data center for different analytics and management operations. Figure 6 illustrates such hybrid topologies, and Figure 7 shows how the flexible architecture can help reduce time to insight into global system behavior at scale.

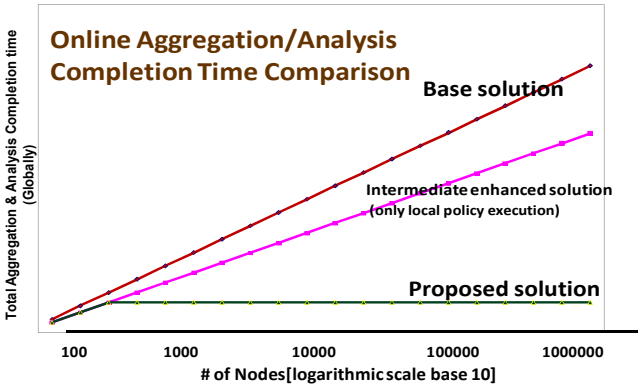

Figure 7. Illustration of reduced time to insight with proposed distributed structures

Control layer for adaptation: The third principle is that of the need for a control layer for distributed monitoring and analytics structures in the cloud so as to adapt to meet bandwidth requirements, failures, and multiple sampling rates. The architecture should be reconfigurable on the fly. Referring back to the distributed structure illustrated in Figure 6, the topology must be adaptable, and satisfy variable resource allocation for specific monitoring and management modules that are collocated with host application.

The above principles are driving the design of the scalable monitoring infrastructure being experimented on OpenCirrus. It does combined monitoring and analytics (termed monalytics) in a distributed manner using the structures mentioned above. The architecture scales-up and down and is configurable based on local optimizations. The software architecture consists of monitoring brokers that aggregate and analyze data from individual data collectors on each node or collection of nodes. Several such monitoring brokers are distributed in the cloud data center each performing data-local analysis, and then are together loosely composed as a computation graph for distributed aggregation and analysis. Different topologies are chosen for the computation graph based on scalability requirements, and it can be reconfigured based on system load and aggregation requirements. The computation graph also uses zone-level isolation to achieve local autonomy while ensuring global aggregation in timely

Open Cirrus Site	Economical ($)					Ecological				Social		
	IT	cooling	Ntwk	support	econo. overall	CO_2 (tonnes-eq)	water (mill. Gal)	Resource Use (GJ-eq)	ecolog. overall	State of devt.	Risk of instability	social overall
Site 1	$0.72	$0.35	$0.16	$0.43		6.0	2.6	83		High	Low	
Site 2	$1.27	$0.59	$0.21	$1.11		6.8	3.3	96		High	Very Low	
Site 3	$1.05	$0.47	$0.12	$1.07		5.9	2.3	81		High	Low	
Site 4	$0.75	$0.35	$0.12	$0.61		6.1	2.7	85		High	Very Low	
Site 5	$0.27	$0.13	$0.05	$0.09		4.3	2.4	59		Low	High	
Site 6	$1.82	$0.77	$0.11	$1.17		10.2	4.3	142		High	Low	
Site 7	$1.23	$0.54	$0.11	$0.98		15.0	4.4	192		High	Low	
Site 8	$0.55	$0.26	$0.10	$0.16		6.9	2.6	95		Med.	Low	
Site 9	$1.01	$0.44	$0.10	$0.83		5.3	2.5	74		High	Very Low	

Figure 8. Cloud Sustainability Dashboard [27]

manner. Overall, the system works for physical and virtualized access, supports heterogeneity in the data center, and also federation across sites.

One example of analysis performed in the distributed computation graph is anomaly detection. Considering the online nature of the service, the anomaly detectors execute light-weight statistical methods that raise alarms based on metric distributions [24][25]. The monitoring brokers correlate and aggregate data across multiple metrics and multiple nodes and apply statistical methods to analyze for anomalies. The data is aggregated as it propagates through the computation graph till all of the data across the data center is aggregated and analyzed. Alarms depicting anomalous behavior are continuously raised during the process in a distributed manner, thus reducing the time to detect problems. This approach also reduces the load on any single centralized server and different topologies have different tradeoffs for data availability. Existing statistical techniques for analysis are based on Entropy methods [24][25].

VI. Cloud Sustainability

Another service being built on OpenCirrus is Cloud Sustainability Dashboard [27]. It collects information about the resources use, aggregates it at the individual site level as well as across sites. Sustainability is reported to the site managers/owners and potentially to users who want to be sustainability aware about the underlying Cloud where their services execute. Similarly to Node Reservation and Scalable Monitoring systems, Cloud Sustainability Dashboard meets the requirements for the physical and virtualized access, for heterogeneity (different sites have different interfaces for integration), and federation.

Cloud Sustainability Dashboard is built around the models representing the resources in the Cloud. The resource models, together with the monitored data of the resource use, enable us to calculate the sustainability of the Cloud. In particular, we can calculate the IT cost in terms of power consumed by servers and networking, the cost associated with the cooling, with support, etc. These costs represent economical sustainability. Similarly, we can measure the ecological sustainability by measuring and modeling the CO_2, water, and general resources use. These sustainability costs are empirical and objective results of measurements or models. Of interest are also social sustainability aspects, the state of development (e.g. based on GDP) or risk of stability of the country where the data center is located. These are somewhat subjective indicators. All of these factors change over the time with more or less dynamically. Cloud infrastructure and Cloud services providers want to be aware of the sustainability of the data centers hosting the Cloud.

Cloud Sustainability Dashboard can be integrated with other Cloud tools. For example, it can be integrated with the Global Workload Manager, which can move Cloud services based on the criteria of the Cloud service providers. Some providers may be most sensitive to cost and would want to run the service where the cost is lowest. Others may be more ecologically conscious and therefore more sensitive to ecological impact. For similar cost they would want their services to execute where ecological sustainability is optimal. Yet other ones may be most sensitive to social impact and would care most in which social or political region the data center is located. Another use of Cloud Sustainability Dashboard is related to business continuity. It is very important for any provider to assure that their service will continue to execute unimpeded by significant changes in the cost, ecological or socio-political disasters in a region. Cloud Sustainability Dashboard offers alerts to both Cloud Infrastructure and Cloud Service providers in such cases. It also observes sustainability trends and can warn in case that any of the sustainability components undergoes significant changes.

VII. Other Issues

Cross-layer Coordination: Given multiple management controllers operating at various time scales and at multiple

abstraction levels, coordination among them is key for overall *efficient* management in cloud data centers. A classic example is that of power-performance coordination. Traditional power managers that make power optimization decisions based on low level performance counters do not receive direct feedback on the effect of power actuation decisions on application performance. This can lead to inefficiency and in some cases, oscillation between performance and power violations when power and application controllers act independently [22]. Instead, application performance-aware power managers can lead to improved power savings, while at the same time, ensuring compliance to application performance (SLA) goals [22].

Failure Management: The cloud presents several challenges for failure management. Public and private clouds have the common issue of monitoring infrastructure and alerting data center personnel or applications of reliability issues in a highly virtualized environment. In a private cloud, the infrastructure can be more deeply monitored using proprietary agents that can send failure status and cause information up to the application or to a presentation-layer dashboard. Third party cloud monitors often don't have access to agents that are highly instrumented for specific hardware. They depend on productized agents. Public cloud monitors at a minimum will report basic failures such as network outages, server outages and the status of virtual servers but with minimum root cause information. Some third party cloud monitors will have more detailed failure analysis but still using productized agents.

Virtual server issues: A key issue for cloud failure management is the status of the virtual machine (VM). Often hardware status is not available to the VM and therefore the application cannot react to hardware messages that may allow it to recover or shutdown gracefully. One remedy to this is to have a monitor that listens to the hypervisor or hardware state and informs the application in the VM of an impending failure so that proactive measures may take place.

Monitoring groups of resources in the cloud: Another challenge is the scenario where applications in the cloud depend on the full availability of many network, server, storage and software resources. The total health of these resources must be maintained as a group to meet the minimum compute requirements of the application. In the presentation layer of monitors, these grouped resources can be displayed and their failure status updated with alerts.

Failure servicing in the cloud: Cloud infrastructure can be comprised of commodity devices to give good scaling, high rates of provisioning and total ROI to the cloud service provider. In this case, servicing may be as simple as full device replacement on a frequent basis. Diagnostics for this service model are simple indications of device failure and inability to recover. Private clouds give the benefit of more sophisticated and resilient devices and diagnostics. Often mission-critical levels of support can be used to maintain a high level total availability of the cloud infrastructure.

Autonomics: The sheer scale and complexity of the cloud infrastructure requires that analytics operations use intelligent algorithms to understand the behavior of system and application behavior. This needs to feed into policy engines that can reflect on this behavior in a self-aware manner, and make control decisions to bring the system back to desired goal state. Statistical learning and control theory mechanisms can be used and extended to operate across multiple time scales, and multiple metrics of operation [24][25][26][28].

VIII. CONCLUSIONS

Cloud is an emerging paradigm with several management challenges. These include scale, multiple abstraction levels, federation, and dynamism. We presented these challenges in this paper, and also principles based on distributed management, federation, and sustainability. Several open issues remain though for cloud management, and it is rich area for further research, so as to make cloud computing a success in the future.

ACKNOWLEDGMENTS

The scalable monitoring efforts are being pursued jointly with Karsten Schwan at Georgia Tech through an HP Innovation Grant. The Node Reservation efforts are being pursued jointly with Shoumen Bardhan and Vishwanath RN from HP Enterprise Services. The sustainability efforts are being pursued jointly with Cullen Bash, Tahir Cader, Yuan Chen, Daniel Gmach, Richard Kaufmann, Amip Shah, and Puneet Sharma at HP Labs.

REFERENCES

[1] M. Armbrust et al., "Above the Clouds: A Berkeley View of Cloud Computing," tech. report UCB/EECS-2009-28, 2009; www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf.

[2] Gartner. http://www.gartner.com/it/page.jsp?id=1389313; http://www.gartner.com/it/page.jsp?id=1454221

[3] IDC, http://www.idc.com/research/cloudcomputing/index.jsp

[4] Amazon AWS, http://aws.amazon.com/

[5] GoGrid, http://www.gogrid.com/

[6] RackSpace, http://www.rackspacecloud.com/

[7] HP Cloud, http://www8.hp.com/us/en/solutions/solutions-detail.html?compURI=tcm:245-300983&pageTitle=cloud

[8] Google Apps, http://www.google.com/apps/intl/en/business/index.html

[9] IBM Cloud, http://www.ibm.com/ibm/cloud/

[10] Microsoft Cloud, http://www.microsoft.com/en-us/cloud/

[11] Eucalyptus, http://www.eucalyptus.com/

[12] Rafael Moreno-Vozmediano, Ruben S. Montero, and Ignacio M. Llorente, "Elastic management of cluster-based services in the cloud", ACDC '09.

[13] Mike Kozuch, et al., Tashi: Location-aware Cluster Management," ACDC'09 June 2009, Barcelona Spain.

[14] RESERVOIR, www.reservoir-fp7.eu

[15] Avetisyan, A., et al., "Open Cirrus A Global Cloud Computing Testbed," IEEE Computer, vol 43, no 4, pp 42-50, April 2010

[16] John H. Howard, Michael L. Kazar, Sherri G. Menees, David A. Nichols, M. Satyanarayanan, Robert N. Sidebotham, and Michael J. West. "Scale and performance in a distributed file system", ACM Trans. Comput. Syst. 6, 1 (February 1988).

[17] Ward Rosenberry, David Kenney, Gerry Fisher, "Understanding DCE," O'Reilly, 1992.

[18] Amit P. Sheth and James A. Larson, "Federated database systems for managing distributed, heterogeneous, and autonomous databases", *ACM Comput. Surv.* 22, 3 (September 1990)

[19] Abhilasha Bhargav-Spantzel, Anna C. Squicciarini, and Elisa Bertino. "Establishing and protecting digital identity in federation systems", *J. Comput. Secur.* 14, 3 (May 2006)

[20] Ian Foster, "Globus Toolkit Version 4: Software for Service-Oriented Systems," IFIP Network & Parallel Computing, LNCS3779.

[21] Atul Adya, William J. Bolosky, Miguel Castro, Gerald Cermak, Ronnie Chaiken, John R. Douceur, Jon Howell, Jacob R. Lorch, Marvin Theimer, and Roger P. Wattenhofer, "Farsite: federated, available, and reliable storage for an incompletely trusted environment", OSDI 2002.

[22] Sanjay Kumar, Vanish Talwar, Vibhore Kumar, Parthasarathy Ranganathan, and Karsten Schwan, "vManage: loosely coupled platform and virtualization management in data centers", ICAC 2009.

[23] Mahendra Kutare, Greg Eisenhauer, Chengwei Wang, Karsten Schwan, Vanish Talwar, and Matthew Wolf, "Monalytics: online monitoring and analytics for managing large scale data centers", ICAC 2010

[24] Chengwei Wang, Krishnamurthy Viswanathan, Lakshminarayan Choudur, Vanish Talwar, Wade Satterfield, Karsten Schwan, "Statistical Techniques for Online Anomaly Detection in Data Centers", IM 2011.

[25] Chengwei Wang, Vanish Talwar, Karsten Schwan, Parthasarathy Ranganathan, "Online Detection of Utility Cloud Anomalies Using Metric Distributions", NOMS 2010.

[26] Ramya Raghavendra, Parthasarathy Ranganathan, Vanish Talwar, Xiaoyun Zhu, Zhikui Wang, "No Power Struggles: A Unified Power Management Architecture for the Data Center," ASPLOS 2008.

[27] Cullen Bash, et al., "Dynamically Assessing Sustainability of Data Center and Cloud," HP Internal Technical Conference, 2011.

[28] Marco Comuzzi, Constantinos Kotsokalis, Christoph Rathfelder, Wolfgang Theilmann, Ulrich Winkler, and Gabriele Zacco, "A framework for multi-level SLA management", ICSOC/ServiceWave'09.