The KOALA Cloud Management Service

A Modern Approach for Cloud Infrastructure Management

Christian Baun, Marcel Kunze
Karlsruher Institut für Technologie
Steinbuch Centre for Computing
Hermann-von-Helmholtz-Platz 1
76344 Eggenstein-Leopoldshafen
Germany
[christian.baun|marcel.kunze]@kit.edu

ABSTRACT

While the variety of public and private cloud infrastructure and storage service offerings increases, only few tools exist to efficiently manage hybrid cloud resources of different cloud providers. KOALA is a novel cloud management tool that allows to work with a large variety of services of various public and private cloud providers in a seamless and transparent way.

While most management solutions like command-line tools or browser extensions require a local installation, KOALA follows the cloud paradigm and operates itself as a software service on the basis of a public or private cloud platform. In addition, KOALA executes alternatively standalone with a Linux or Mac OS X system.

The KOALA cloud manager allows to control almost all existing cloud resources with an API compatible to the Amazon Web Services. The users of KOALA have the freedom of choice to use any browser and device to interact with the cloud services. Especially in cases where the users have strong requirements regarding security and privacy, it is a benefit to run the management solution for different cloud services in a private context.

Keywords

KOALA, Cloud Computing, IaaS, PaaS, SaaS, Amazon Web Services, Google App Engine

1. INTRODUCTION

The key advantages of cloud computing are flexibility, scalability and usability. These features originate in the combination of virtualization technologies with web services [5]. In the cloud, network-centric, scalable, abstracted IT infrastructures, platforms and applications are provisioned on-demand for immediate use by the customers [6]. In general, a business model exists to foresee accounting and billing of the actually consumed resources following the *pay as you go* principle [1]. In some offerings the basic usage of the services is granted for free.

One of the most interesting features of cloud computing is rapid elasticity: Resources can be added and removed any time by a customer within minutes. This allows to match workloads with resources much more precisely as compared to the traditional method to plan, build and run IT resources on-premise. However, in contrast to the elasticity and flexibility of the cloud services the traditional management methods and tools seem inappropriate: They usually require local installation of software with the burden of continuous updates and patches. Furthermore, the solutions are very often proprietary and just conform to the cloud service offerings of specific service providers which makes it difficult to work in hybrid cloud systems. A vendor agnostic generic cloud based software service would bear many advantages in this area.

This paper addresses the issue of multi-vendor cloud management and is organized as follows: In Section 2 we present open source solutions to create private cloud infrastructures and storage services. Section 3 compares cloud management tools and discusses their features. Section 4 describes the design of an ideal management tool for cloud services and the implementation of such a tool. The KOALA cloud management service is discussed in Section 5 and evaluated in Section 6.

2. TAXONOMY OF CLOUD SOLUTIONS

The Amazon Web Services (AWS)[1] are a popular collection of public cloud service offerings. Due to their widespread use their API may be considered as a de-facto standard for cloud services. Amongst others, the AWS include the Elastic Compute Cloud (EC2)[2], Simple Storage Service (S3)[3] and Elastic Block Store (EBS)[4]. EC2 implements Infrastructure as a Service (IaaS) to operate scalable virtual server instances. S3 provides a web object data storage for applications and EBS provides persistent block level storage volumes. The AWS comprises a broad range of additional cloud computing web services, but these are not considered in this paper.

A cloud IaaS solution that adheres to the AWS API may

[1] http://aws.amazon.com
[2] http://aws.amazon.com/ec2/
[3] http://aws.amazon.com/s3/
[4] http://aws.amazon.com/ebs/

Table 1: Popular private cloud solutions and their level of compatibility to the AWS

Name	EC2 API	S3 API	EBS API
CloudStack	subset	no support	no support
Eucalyptus	full support	full support	full support
Nimbus	subset	subset	no support
OpenNebula	subset	no support	no support
Tashi	under dev.	no support	no support

benefit from the growing ecosystem of compatible management tools and services. Various open source private cloud IaaS solutions exist (see Table 1) and most of them have at least a subset of the EC2, S3 and EBS APIs implemented.

Eucalyptus is an open source software originally developed at the University of California in Santa Barbara that implements cloud computing on compute clusters. Since April 2009, the software is maintained and supported by Eucalyptus Systems, a company founded by the original authors. Eucalyptus stands for Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems. It implements IaaS that is compatible to the EC2 API and gives the customers the ability to run and control virtual machine instances (Xen or KVM) deployed across a variety of physical resources [17] [18]. Eucalyptus is the only private cloud IaaS solutions with an interface that is fully compatible with EC2 and it contains the storage services Walrus and Storage Controller (SC). Walrus is compatible to the S3 interface and the SC implements a block storage service with an interface compatible to EBS. Both services may reside at an arbitrary node inside the Eucalyptus cluster. Walrus can also be installed and run independent of the other Eucalyptus services.

Nimbus is build on top of the Grid middleware Globus 4 and has currently only a small part of the EC2 API implemented. An EC2 backend also exists that serves as a portal to EC2. Scenarios where Nimbus is used are private science clouds that provide compute cycles in the cloud for scientific communities like the University of Chicago Science Cloud[5]. The IaaS can use Schedulers like Portable Batch System (PBS) or Sun Grid Engine (SGE) to schedule virtual machines [11] [15]. Nimbus includes Cumulus, a storage service that is compatible with the S3 REST API and is used by Nimbus to store the images. However, it does not support the S3 SOAP API. Cumulus can be installed and run standalone without Nimbus.

OpenNebula allows virtual machines to be placed and replaced dynamically in a physical resource pool [21]. The software interoperates with the Xen hypervisor, KVM and VMware and allows to scale private clouds with multiple external AWS compatible private or public cloud infrastructures (EC2 and ElasticHosts) to realize hybrid clouds. Since OpenNebula version 2.0, a small part of the EC2 API has been implemented which enables OpenNebula to be used with the growing tools of the AWS ecosystem. OpenNebula supports the grouping of nodes which is important to realize High Performance Computing as a Service (HPCaaS). In contrast to Eucalyptus and Nimbus, OpenNebula has no storage service included.

[5]http://www.scienceclouds.org

Tashi is a development of the Open Cirrus[6] cloud computing research testbed [4] and supported by the Intel Labs in Pittsburgh. The focus of this IaaS is the scheduling of data in cluster systems. The EC2 API support of Tashi is under development.

Eucalyptus, OpenNebula and Nimbus are used in many research and development projects. Examples include the usage of OpenNebula at CERN with a virtual machine provisioning system that reached a peak load of approximately 16,000 virtual machines [20] and the sky computing projects [12] that use Nimbus to build up a grid of clouds.

3. MANAGEMENT OF CLOUD SERVICES

The existing management tools for cloud infrastructure and storage services can be classified as follows:

- Command-line to work with tools like the API Tools delivered by Amazon, the Euca2ools[7] of Eucalyptus, GSutil[8] of Google Storage (GS) and s3cmd[9]
- Locally installed management applications with a graphical user interface like EC2Dream[10], Gladinet Cloud Desktop[11] and Cyberduck[12]
- Firefox browser extensions like ElasticFox[13], Hybridfox[14] and S3Fox[15]
- Online tools (software services) such as the AWS Management Console[16] offered by Amazon, the Google Storage Manager[17] and Ylastic[18]

The command-line API Tools offered by Amazon only support their own public cloud offerings. The Euca2ools of the Eucalyptus project support both, public and private cloud infrastructure services. The same holds for GSutil of Google Storage that supports public and private cloud storage services. The tool s3cmd works with Amazon S3, as well as the S3-compatible private cloud storage services Walrus and Cumulus. All command-line tools require local installation and therefore may lack ease of use as there is no graphical user interface.

Locally installed applications with a graphical user interface in general provide a better level of usability. EC2Dream is a management application for EC2, EBS, Relational Database Service (RDS)[19] and Eucalyptus. The software is open source and runs with Windows, Mac OS X and Linux. Cyberduck is an open source client for various file transfer protocols and cloud storage services. The application

[6]http://opencirrus.org
[7]http://wiki.debian.org/euca2ools/
[8]http://code.google.com/p/gsutil/
[9]http://s3tools.org/s3cmd
[10]http://www.elastdream.com
[11]http://www.gladinet.com
[12]http://cyberduck.ch
[13]http://sourceforge.net/projects/elasticfox/
[14]http://code.google.com/p/hybridfox/
[15]http://www.s3fox.net
[16]http://aws.amazon.com/console/
[17]https://sandbox.google.com/storage/
[18]http://www.ylastic.com
[19]http://aws.amazon.com/rds/

Table 2: Command-line tools

	API Tools	Euca2ools	GSUtil	s3cmd
Provider	Amazon	Eualyptus	Google	Ludvig
Open Source	yes	yes	yes	yes
Charge	none	none	none	none
EC2 support	yes	yes	no	no
S3 support	no	no	yes	yes
EBS support	yes	yes	no	no
ELB support	yes	no	no	no
Eucalyptus	yes	yes	no	no
Walrus	no	no	yes	yes
Nimbus	yes	yes	no	no
Cumulus	no	no	no	yes
OpenNebula	yes	yes	no	no
Google Stor.	no	no	yes	no
Requirements	local installation			

Table 3: Locally installed applications with GUI

	EC2Dream	Cloud Desk.	Cyberduck
Provider	Turner	Gladinet	Kocher
Open Source	yes	no	yes
Charge	none	$59.99/year	none
EC2 support	yes	no	no
S3 support	no	yes	yes
EBS support	yes	no	no
ELB support	yes	no	no
Eucalyptus	yes	no	no
Walrus	no	no	yes
Nimbus	no	no	no
Cumulus	no	no	no
OpenNebula	no	no	no
Google Stor.	no	yes	yes
Requirements	specific OS, local installation		

runs with Windows and Mac OS X and it supports directory synchronization and multiple languages. The proprietary application Gladinet Cloud Desktop supports S3 and Google Storage and integrates them into the Windows Explorer interface.

Browser extension like ElasticFox, Hybridfox and S3Fox only work with the Firefox browser but not with e.g. Internet Explorer, Opera, Google Chrome or Safari. Furthermore, the customers have to install and maintain the management tool on their local computer, a fact that somehow does not reflect the cloud paradigm very well. Recent versions of ElasticFox, Hybridfox can interact with EC2, EBS and Eucalyptus infrastructures. S3Fox is only capable of working with S3.

Online tools are offered as software service and the customers have the freedom of choice to use any operating system and browser. However, Amazon's AWS Management Console is in line with just the AWS cloud service offerings. It is currently not foreseen to configure it in a way to manage other cloud infrastructures. The same holds for the Google Storage Manager that cannot be configured to work with any other cloud service besides Google Storage. Ylastic offers support for most AWS cloud services and Eucalyptus infrastructures but it is not possible to manage other compatible infrastructures e.g. Nimbus or OpenNebula. As the access keys are stored with the provider, the customer also has to trust the provider of the management tool regarding privacy and security.

Table 4: Browser extensions

	ElasticFox	Hybridfox	S3Fox
Provider	Amazon	Community	Suchisoft
Open Source	yes	yes	no
Charge	none	none	none
EC2 support	yes	yes	no
S3 support	no	no	yes
EBS support	yes	yes	no
ELB support	no	no	no
Eucalyptus	yes	yes	no
Walrus	no	no	no
Nimbus	no	no	no
Cumulus	no	no	no
OpenNebula	no	no	no
Google Stor.	no	no	no
Requirements	Firefox browser, local installation		

A trade-off analysis of the available cloud management solutions is shown in Table 2, 3, 4 and 5.

3.1 Mobile device support

While lots of customers of cloud services have mobile devices like smartphones or tablet personal computers with the Android operating system or Apple iOS, the current management tools for cloud infrastructures and storage services more or less lack compatibility with such devices.

The browser extensions are not compatible with these mobile devices because they do not run the Firefox browser and the use of command-line tools is mostly impracticable. A drawback of native applications for these devices is their limited storage capacity, the need to perform regular updates and the development effort to support more than a single platform. While applications for Android and Blackberry are written in Java, the Apple platform uses Objective-C.

In contrast to browser extensions and command-line tools the software services work with any browser and therefore also with mobile devices.

4. A BETTER MANAGEMENT TOOL

In order to improve the situation, a flexible and open solution should be designed and implemented to satisfy the needs for the management of multi-vendor cloud infrastructures and storage services.

The drawbacks of tools like browser extensions, command-line tools and applications with a graphical user interface that require a local installation indicate that an online tool is best suited to work with multiple cloud service implementations. This should be delivered as a software service that can be used with all kinds of devices and browsers. The software service could be implemented on a scalable platform service such as Google App Engine[20]. It is easy to optimize a software service for mobile devices that lack physical keyboards and have a limited screen size.

If the software service is operated on a public cloud platform, customers have no need to worry about scalability and availability. In scenarios where security or privacy concerns

[20] `http://appengine.google.com`

Table 5: Online tools (software services)

	AWS Console	GS Manager	Ylastic
Provider	Amazon	Google	Ylastic
Open Source	no	no	no
Charge	none	none	$25
EC2 support	yes	no	yes
S3 support	yes	no	yes
EBS support	yes	no	yes
ELB support	yes	no	yes
Eucalyptus	no	no	yes
Walrus	no	no	yes
Nimbus	no	no	no
Cumulus	no	no	no
OpenNebula	no	no	no
Google Stor.	no	yes	no
Requirements	any browser		

exist, it should be possible to operate the management tool on a private cloud platform, too, avoiding to store the access credentials off-premise. In principle all secret keys should be encrypted due to security concerns as they are stored.

The fact that a software service is not functional without a internet connection does not resemble any problem as working with cloud services requires in most use cases to be connected with the internet anyway.

5. KOALA CLOUD MANAGER

KOALA[21] (**K**arlsruhe **O**pen **A**pplication (for) c**L**oud **A**dministration) implements a management tool for cloud services realized as a software service, according to the features discussed above.

Public cloud services such as AWS and private cloud services based on Eucalyptus, Nimbus and OpenNebula are currently supported. Besides infrastructure services compatible to EC2 the software also helps to manage S3 and EBS compliant storage resources and interacts with the Elastic Load Balancing (ELB)[22] service (see Table 5).

As KOALA is not a marketplace for cloud resources it is a pre-requisite that the customers have already access to corresponding services. When customers log in for the first time into KOALA, they need to import their credentials for at least a single cloud infrastructure.

KOALA offers all features that are needed to interact with cloud services that are compatible to the AWS API. The customers can import credentials for AWS, Eucalyptus, Nimbus and OpenNebula public and private cloud infrastructure services as well as the storage service Google Storage.

The KOALA software service has been designed as a Google App Engine application. It has been written in Python using the web framework Django[23]. In order to communicate with cloud services compliant to AWS it uses boto[24], a python interface which is open source and supports EC2, S3, EBS, ELB and more services that are part of the AWS. There is a limited but powerful set of APIs inside the App Engine to use the persistent Datastore BigTable (with a SQL-like query language called GQL) and the fast Memcache. The Datastore is used to store the customers imported credentials and other user data. The user management and authentication is based on Google accounts.

KOALA itself can be installed and run either on the public Google cloud platform or a private PaaS based on AppScale[25] [9][7] or typhoonAE[26]. If KOALA operates inside AppScale or typhoonAE it is possible to control the cloud infrastructure with a management application running inside the same cloud. This helps customers with specific requirements regarding security and privacy. The corresponding APIs and services of the App Engine are supported by AppScale. For instance, the user authentication is implemented with AppScale and there is no need to use any Google accounts for local operation of KOALA. When running KOALA inside AppScale there are no external dependencies and it is possible to perform cloud management operations independently of any provider.

KOALA is available as open source and licensed according to the Apache License v2.0. The source code and further information is available at the KOALA project site[27]. KOALA's graphical user interface currently supports the English and German language. Due to the design of the application, it is straightforward to implement additional languages.

6. EVALUATION

KOALA enables cloud customers to work in a seamless and transparent way with different cloud infrastructures and storage services, eliminating the need to install and maintain lots of different tools to achieve a similar functionality.

6.1 KOALA and compute services

With KOALA it is easy for the customers to start, reboot and stop instances in various cloud infrastructure regions and have access to the console output of virtual machines.

Instances need to be assigned to a security group that defines access rules in the firewall. Every customer can create new security groups, erase existing groups and alter the rules inside his security groups.

To log into instances, the customers need at least one key pair that represents a pubic and a private key. The public key is stored with each instance by the cloud infrastructure automatically. Using the private key the customer can access the instances directly by SSH without a password. For this purpose the customer needs to store the private key in a local context on his computer or a portable memory device. With KOALA each customer can create and erase key pairs for use with every cloud infrastructure region he has credentials for.

For all EC2 compliant private cloud infrastructure services the customers can check the available images. As the EC2 public cloud hosts several thousand public images and Amazon currently provides no ability to filter the returned list of images on the server side, KOALA offers the possibility to create a list of favorite images. A collection of quick start images from the Ubuntu project help with the first steps.

Instance storage in a cloud infrastructure is not persistent, i.e. after the termination of an instance its data is lost. Therefore persistent data in the cloud need to be stored inside EBS volumes that are based on S3 storage. With

[21] `http://koalacloud.appspot.com`
[22] `http://aws.amazon.com/elasticloadbalancing/`
[23] `http://www.djangoproject.com`
[24] `http://code.google.com/p/boto/`
[25] `http://appscale.cs.ucsb.edu`
[26] `http://code.google.com/p/typhoonae/`
[27] `http://code.google.com/p/koalacloud/`

KOALA it is possible to create new volumes and erase existing ones. A volume may be attached to a single instance and remapped to other instances at any time. Using a volume is equivalent to the work with an unformatted block storage device. Volumes can be instrumented with any filesystem and attached/detached to/from any instance. Customers can also create snapshots of their volumes at any time. The snapshots can be used to create new volumes and the customer can decide in which availability zone the new volume exists. This feature enables the customers co copy their volumes from one availability zone to another one.

Instances are assigned a public (internal) and a private (external) DNS name automatically in the moment when they are created. However, if an instance is terminated the public and private DNS names both are lost. Thus for sustainable services inside a cloud infrastructure it is mandatory to offer elastic IP addresses that can be associated and disassociated at any time. Inside KOALA it is possible to create elastic IP addresses and to release them, as well as attach and detach them to/from any instance the customer owns.

In order to realize scalable services inside an AWS cloud infrastructure, elastic load balancers can be created and erased as well as their rules can be altered. The private cloud re-implementations Eucalyptus, Nimbus and OpenNebula do not provide elastic load balancers.

6.2 KOALA and storage services

KOALA supports S3, Google Storage[28] and Walrus. It implements two different ways to access these object-based storage services: A pure S3 mode and a mode with more comfort. The pure S3 mode exactly presents the content of each S3 bucket. The list of objects includes for each object the object key, size, date of last write access and the MD5 checksum.

The customers can create and erase S3 buckets, as well as upload and download S3 objects directly via HTTP POST and GET to/from the storage services. It is also possible to check and alter the user access rights – the Access Control List – of each S3 object.

S3 implements a flat name space for the keys and therefore S3 has no support for folders. However, folders can be simulated by clever use of key names. The S3 mode with more comfort in KOALA uses keys that end with the directory placeholder `_$folder$`. The key of an object that is meant staying inside a folder has a name following the schema `folder/subfolder/object`. Based on this approach, folders inside S3 can be emulated the same way S3Fox and the Google Storage Manager works.

6.3 Platform-specific issues

If KOALA executes on the basis of Google App Engine the restrictions of this platform service need to be considered. Applications running inside the App Engine can communicate only via URL Fetch, XMPP, E-Mail and the port numbers 80, 443, 4443, 8080-8089, 8188, 8444 and 8990. This does not resemble a problem for public cloud services like the AWS or Google Storage as these services use port 80 and 443 only. However, it is a challenge for the operation of private cloud infrastructures like Eucalyptus, Nimbus and OpenNebula. The default port numbers supported by Eucalyptus, Nimbus and OpenNebula are 8773, 8442 and 4567. Therefore the administrator of a private cloud infrastructure has to route the infrastructure access port to one of the supported ports.

If KOALA is used inside a private cloud platform like AppScale or typhoonAE, the customers are free to use any port number that is allowed by the hosting system.

6.4 KOALA and mobile devices

Users of smartphones with Apple iOS or Google Android are able to use KOALA, too, as the whole graphical user interface has been implemented following the HTML 4.01 transitional standard without JavaScript or proprietary technologies like Adobe Flash.

KOALA includes a customized version for mobile devices that provides an user interface which has been optimized for the display resolutions of smartphones and the usage of touch screens.

7. CONCLUSIONS AND FUTURE WORK

The emerging market of cloud services currently lacks generic management solutions with good usability. In this paper we present the design and implementation of KOALA, an open source management tool to fill this gap. KOALA is a web-based application designed to help the customers to manage AWS compliant cloud infrastructures and storage services. It supports the EC2, S3, EBS and ELB services (see Table 6). The software itself can be installed and operated in public and private clouds. KOALA is not just a new tool or service; it is a new approach to work in a seamless and transparent way with cloud infrastructures and storage services of various public cloud providers as well as private cloud solutions.

Table 6: KOALA Cloud Manager

	KOALA
Provider	SCC, KIT
Open Source	yes
Charge	none
EC2 support	yes
S3 support	yes
EBS support	yes
ELB support	yes
Eucalyptus	yes
Walrus	yes
Nimbus	yes
Cumulus	no
Google Stor.	yes
Requirements	any browser, local installation or PaaS

Our next steps in the development of KOALA foresee the integration of the scalable Domain Name System (DNS) web service Route 53[29] and the private cloud infrastructure solution CloudStack. Support for Cumulus, the S3 compliant storage service of Nimbus, will be implemented, too. Furthermore, KOALA is an interesting basis to study specific issues that arise in the management of hybrid cloud infrastructures. Monitoring and accounting solutions [13] and an on-demand virtual machine allocation method [19] for Eucalyptus could be integrated into the system.

[28] `http://code.google.com/apis/storage/`

[29] `http://aws.amazon.com/route53/`

While some projects have implemented marketplaces for distributed computing (e.g. POPCORN [16] and SPAWN [22]) and for grid computing resources (e.g. Bellagio [2] [3], CATNETS[30] [10], Nimrod/G [8] and Tycoon[31] [14]), still no marketplace for cloud computing services exist. We plan to integrate a component that helps the customers to search a cloud infrastructure that best matches a request regarding functionality and price. This would be an important step towards the formation of a cloud service marketplace.

8. REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. Above the clouds: A berkeley view of cloud computing. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley, Feb 2009.

[2] A. Auyoung, P. Buonadonna, B. N. Chun, C. Ng, D. C. Parkes, J. Shneidman, A. C. Snoeren, and A. Vahdat. Two auction-based resource allocation environments: Design and experience, 2008.

[3] A. Auyoung, B. N. Chun, A. C. Snoeren, and A. Vahdat. Resource allocation in federated distributed computing infrastructures, 2004.

[4] A. I. Avetisyan, R. Campbell, I. Gupta, M. T. Heath, S. Y. Ko, G. R. Ganger, M. A. Kozuch, D. O'Hallaron, M. Kunze, T. T. Kwan, K. Lai, M. Lyons, D. S. Milojicic, H. Y. Lee, Y. C. Soh, N. K. Ming, J.-Y. Luke, and H. Namgoong. Open cirrus: A global cloud computing testbed. *Computer*, 43:35–43, 2010.

[5] C. Baun, M. Kunze, J. Nimis, and S. Tai. *Cloud Computing: Web-basierte dynamische IT-Services*. Springer, 2009.

[6] M.-E. Bégin. An EGEE Comparative Study: Grids and Clouds – Evolution or Revolution. CERN, 2008.

[7] C. Bunch, N. Chohan, C. Krintz, J. Chohan, J. Kupferman, P. Lakhina, Y. Li, and Y. Nomura. An Evaluation of Distributed Datastores Using the AppScale Cloud Platform. In *IEEE Cloud10: International Conference on Cloud Computing*, July 2010.

[8] R. Buyya, D. Abramson, and J. Giddy. Nimrod/G: An architecture for a resource management and scheduling system in a global computational Grid. In *Proceedings of the 4th international conference on High Performance Computing in Asia-Pacific Region*, pages 283–289. IEEE Computer Society Press, 2000.

[9] N. Chohan, C. Bunch, S. Pang, C. Krintz, N. Mostafa, S. Soman, and R. Wolski. AppScale: Scalable and Open AppEngine Application Development and Deployment . First International Conference on Cloud Computing, 2009.

[10] T. Eymann, W. Streitberger, and S. Hudert. CATNETS – Open-Market Approaches for Self-Organizing Grid Resource Allocation. In *Proceedings of the 4th International Workshop on Grid Economics and Business Models*, volume 4685 of *Lecture Notes in Computer Science*, pages 176–181. Lecture Notes in Computer Science (LNCS), 2007.

[30] `http://research.ac.upc.edu/catnet/`

[31] `http://tycoon.hpl.hp.com`

[11] K. Keahey, M. Tsugawa, A. Matsunaga, and J. Fortes. Sky Computing. *Internet Computing, IEEE*, 13(5):43–51, August 2009.

[12] K. Keahey, M. Tsugawa, A. Matsunaga, and J. Fortes. Sky Computing. *IEEE Internet Computing*, 13:43–51, 2009.

[13] K. Knese. *Evaluation und Implementierung einer Accounting-Lösung für die Private Cloud mit Eucalyptus*. Duale Hochschule Baden-Württemberg Karlsruhe, August 2010.

[14] K. Lai, L. Rasmusson, E. Adar, S. Sorkin, L. Zhang, and B. A. Huberman. Tycoon: an Implemention of a Distributed Market-Based Resource Allocation System. Technical report, HP Labs, Palo Alto, CA, USA, Dec. 2004.

[15] P. Marshall, K. Keahey, and T. Freeman. Elastic Site Using Clouds to Elastically Extend Site Resources. April 2010.

[16] N. Nisan, S. London, O. Regev, and N. Camiel. Globally Distributed Computation over the Internet: The POPCORN Project. *International Conference on Distributed Computing Systems*, May 1998.

[17] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov. Eucalyptus: A Technical Report on an Elastic Utility Computing Architecture Linking Your Programs to Useful Systems. In *UCSB Computer Science Technical Report Number 2008-10*, 2008.

[18] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov. The Eucalyptus Open-source Cloud-computing System. In *CCA'08: Proceedings of Cloud Computing and Its Applications workshop*, 2008.

[19] D. Rattu. *On-Demand Virtual Machine Allocation for Clouds*. Hochschule Baden-Württemberg, Karlsruhe, Germany, 2009.

[20] U. Schwickerath, S. Goasguen, B. Moreira, E. Roche, and R. Wartel. CERN's internal cloud infrastructure – Status and plans. 2010.

[21] B. Sotomayor, R. S. Montero, I. M. Llorente, and I. Foster. Virtual infrastructure management in private and hybrid clouds. *IEEE Internet Computing*, 13(5):14–22, 2009.

[22] C. A. Waldspurger, T. Hogg, B. A. Huberman, J. O. Kephart, and W. S. Stornetta. Spawn: A Distributed Computational Economy. *IEEE Transactions on Software Engineering*, 18:103–117, February 1992.