From Grid to cloud, the STAR experience

Jérôme Lauret, Brookhaven National Laboratory
Matthew Walker, Massachusetts Institute of Technology
Sebastien Goasguen, Clemson University
Levente Hajdu, Brookhaven National Laboratory

Abstract:
In recent years, Cloud computing has become a very attractive paradigm and popular model for accessing distributed resources. The Cloud has emerged as the next big trend after the so-called Grid computing approach. The burst of platforms and projects providing Cloud resources and interfaces at the very same time that Grid projects are entering a production phase in their life cycle has, however, raised the question of the best approach to handling distributed resources. Especially, are Cloud resources scaling at the levels shown by Grids? Are they performing at the same level? Can they be compared? What are their overhead on the IT teams and infrastructure? Since its first use of Cloud resources on Amazon EC2 in 2008/2009 using a Nimbus/EC2 interface, the STAR experiment software team has tested and experimented with many novel approaches: from a traditional native EC2 approach to the Virtual Organization Cluster (VOC) at Clemson university and Condor/VM on the GLOW resources, STAR has ramped up the job scale step by step to achieve stable operation at the 1,000 jobs level. This paper presents an overview of our findings and reports on practical usage of truly opportunistic use of resources.

1 Introduction

Faced with huge data challenges, High Energy and Nuclear Physics experiments and other communities have worked toward the design and engineering of new computing models and paradigms where resources at given sites are seen as part of a global infrastructure. At a higher level, the aim was to present the resources as one virtual entity and thus provide a consistency to the user who may interact with a global pool of resources and not deal with the details. The era of Grid computing was born, and since its inception it has shown its success in data processing and handling over vast distance and across many continents. Grid projects such as the Open Science Grid matured and hardened the technology to provide national grids on which Virtual Organizations (VO) could harvest vast amounts of supposedly widely shared resources. However, the many limitations of Grids were also obvious from the start - with heterogeneity at the heart of its philosophy and the plethora of environments, the exploitation of such global resources became rather challenging: experiment's team spreading thin, the support of complex software stack made the opportunistic usage of sparse and unused resources nearly impossible, and most VOs reverted to the use of pre-installed software with local software support and maintenance. This "modi operandi" allowed minimizing internal cost as the vast amount of available combinations between software, middleware, compilers, libraries and event services would imply a large combination and support platform and hence, a growth in the support team, not to mention the effort in quality control and assurance needed for experimental teams having reached peak productivity. The arrival of Cloud computing has introduced a new reality and feasibility beyond proof of principles: relying on virtualization, a virtual machine can contain all the required components from a tailored and specific operating system (OS, up to the minor revision if need be) to a specific middleware to an experimental software stacks pre-verified, controlled, and approved at their home institution via regression test suites. The burden on the experimental team suddenly appeared to be normalized to their common environment, making the dream of harvesting unused resources "on-the-fly" possible.

However, while Cloud computing models are rapidly reaching maturity, unlike Grids, they do not benefit from a unified approach and standard interface, and many models are available on the market from private commercial clouds to public national laboratory or university community clouds. This paper will present a few models tested by the RHIC/STAR experiment as well as our usage experience.

1.1 The RHIC/STAR Experiment's data challenge

The Solenoidal Track At RHIC (STAR) experiment is a Nuclear Physics experiment located at the Brookhaven National Laboratory (BNL) and part of the RHIC program. At STAR, physicists and skilled specialists from across the world are working hard to understand the nature of the early universe and the tiniest building blocks of matter through the study of nuclear collisions at the highest energies achieved in the laboratory.

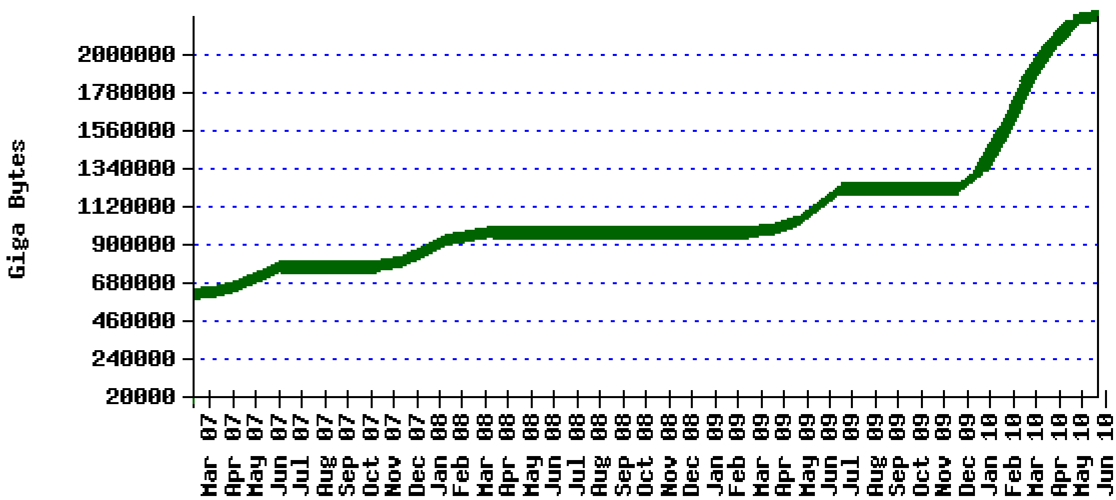

Figure 1: STAR raw data amount stored into archival Mass Storage as a function of time. The curve is restricted to the range 2007 to 2010; the last year, also known as Run 10, has accumulated nearly as much data than all previous years combined with a total of 2.2 PBytes of data.

To achieve this goal, the STAR experiment has developed an ambitious program now entering the Petascale data challenge (Figure 1). In this resource planning, it was shown that only two passes over the raw data and one pass user analysis was possible within the current funding profile; another portion of user analysis as well as all resources to provide simulation capabilities needed to be outsourced and come from other sources than the RHIC funding. Also, with the experiment moving from a statistically challenged data sample to large data sample, it enters an era where systematic uncertainties are predominant and hence, raise concerns on possible (unforeseen) needs for additional simulation for understanding its effect on the detector response, fine grain efficiency, and momentum corrections. Such a simulation will be discussed in Section 2.2.5, constituting a real need for truly opportunistic resources.

To face this challenge, STAR has had a long-standing participation in Grid co-laboratories programs. Since its early phase, STAR has been part of the Particle Physics Data Grid consortium, the Trillium project and its next incarnation, the Open Science Grid. But in parallel, and facing early challenges in porting large application in Physics production mode to heterogeneous platforms, the STAR experiment Software and Computing team have also carried and maintained active participation and testing in alternative distributed computing approaches „ with its first truly last minute opportunistic use of Cloud computing based resources in early 2009. What has driven the experiment to outreach to other distributed computing models? What are the limitations of grids for experimental groups ready to harvest or aggregate their resources?

1.2 Grids, success and limitations - a problem analysis

Grids have been successful at aggregating geographically separated computing and storage resources (at sites) and presenting to their users and communities an aggregate or federated "cluster" like component. On such federated resources, the breadth of activities has brought their first benefits, starting from simply enabling data movement across the network and cyber-infrastructure „ and continuing to the development and/or the hardening of middleware and infrastructure, leading to reaching an ever-increasing operational sustainability. From 65% success rate in 2006, 85% in 2006-2007, 90% in 2009 and greater than 97% efficiency on job success using the OSG software stack, the deliverables and benefits are undeniable to first sight and the resource aggregation strategy paid off after long years of efforts. In detail, the STAR experiment seldom uses Grid resources and at the level of 64,000 jobs a week and mostly using dedicated sites that is, sites where the STAR software is pre-installed and maintained locally. This could be easily understood by realizing that STAR, after nearly 10 years of productivity and refinement of its software developed by hundreds of scientists, has reached a point where the stack is composed of 2.5 million line of codes, depends on multiple external libraries (e.g., MySQL, XML, ROOT) and relies on a mix of compilers and languages (from C, C++ and FORtran) requiring Linux core support for those languages (e.g., libstdc++, libg2c). The heterogeneity of Grids does not ensure the existence of such components and compiling "on the fly" is already a no-go due to the lack of existence of compilers. In addition, there are no information systems providing enough information allowing for safely choosing sites fulfilling the proper conditions, and as a consequence, experimental groups have reverted to "pilot jobs" where a simple script runs on the remote resource and checks first the availability of components before downloading and installing the VO software stack and an appropriate payload. Additionally, Grids are very cryptic – troubleshooting is still far from adequate and upon errors, messages from the stack are simply not understandable by the average user (some error may have many meanings requiring investigation). As a result, bursts of resources are hard to acquire and in fact, inspection of other experimental projects reveals little to no opportunistic use of Grid resources and an operation only sustainable to the extent a "support team" exists at these facilities.

In contrast, and while clouds do not offer solutions for adequate monitoring and troubleshooting and cannot be sold as "silver bullets", virtualization offers a "software container" based approach and therefore provides definite advantages while running over heterogeneous resources: the combination of OS, libraries, software stack version is not only under full control of the experiment, but the entire combination can be pre-validated via regression test suite. This in turn saves resources, both human and computing cycles (although the human cost is larger in this case) which otherwise would need to be spent to ensure Physics reproducibility and accuracy over heterogeneous environments. As a bonus, the prepared VM can be archived and serve as a software container repository to ensure later checks of past results or cross comparisons of results obtained with older software often not supportable on newer machines (and newer compilers) without additional porting efforts.

A good question would be whether the Cloud simply displaces the problem: while the software and environment can be preserved, the VM technologies are rapidly evolving and it becomes questionable if today's VM formats would at all be compatible with tomorrow's new virtualization approaches. However, hopes remain as not only tools such as `rBuilder` are being developed in the private sector to address release life-cycles (as well as long term maintainability of a virtualized on-demand platform provisioning approach), but solutions such sa `Xen` VM or `VirtualBox` provide tools to convert from the most common VM formats to their native image format. It is expected that this trend will continue as the need for conversion is already recognized. Certainly, the many advantages of Clouds have been noted by an enthusiastic community in search for true solutions and ,as shown on Figure 2, the interest in Cloud now surpasses that of Grid computing.

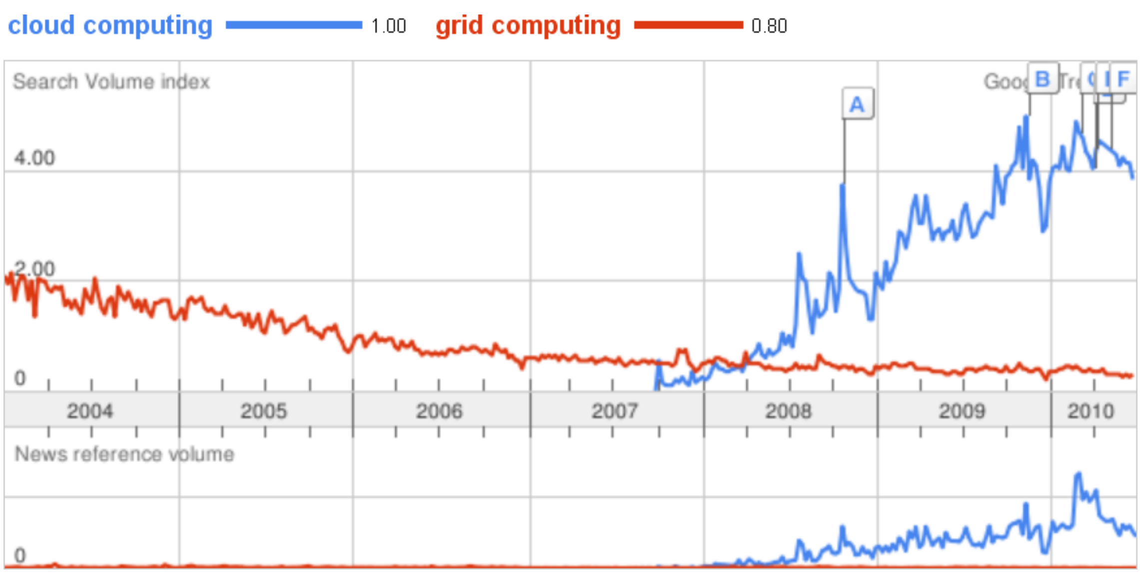

Figure 2: Google search trends for Cloud and grid computing terms. The decrease of interest for grid computing and increase for cloud computing is indicative of a growing community (at large) interest for cloud computing now surpassing Grid computing. The switch of interest happened in 2008.

2 Clouds, virtualization models and STAR tests

2.1 Cloud key features and clouds

We do not intend to describe the anatomy of Clouds in great detail. It is, however, noteworthy to mention that while Grids ask for job slots on Worker Nodes (WN), cloud interfaces are intrinsically requesting VM instances to start on a common infrastructure (regardless of the nature of its "flavor"). In our work, the notions of Software as a Service (SaaS), Platform as a Service (PaaS), or Infrastructure as a Service (IaaS) which could be combined together is not relevant, those definitions and all related notions could have been defined and explained by NIST – the IaaS runs our fully provided and customized VM containing all software provisioning we will need to run and establish a simulation, calibration, real data reconstruction or user analysis.

In this paper, we focus on a Nuclear Physics experiment workflow and its feasibility on the grid: whether or not little or large inputs are needed, whether or not external meta-data (database) information is needed for an event reconstruction producing Physics analysis usable quantities, our workflow does produce a significant amount of output (could be as large as several GB per job) and network transfer in/out of Cloud into a central archival storage system at BNL (For science preservation) was needed to conclude on usability.

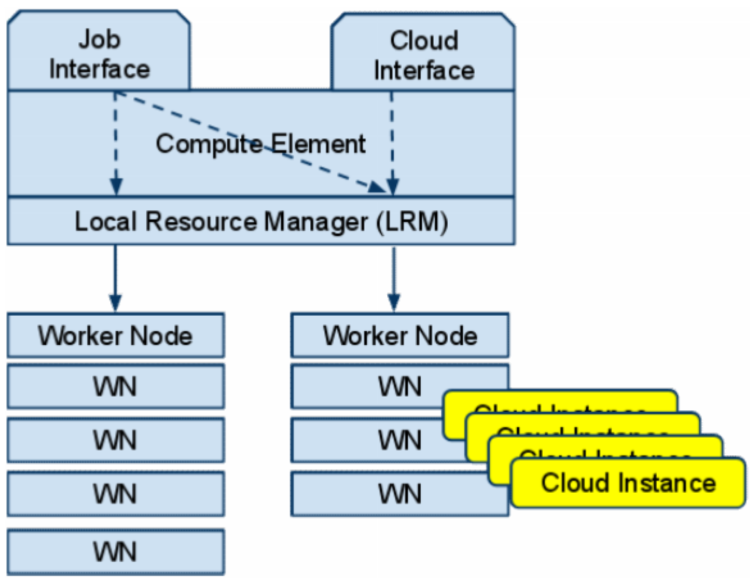

Figure 3: High level representation of access through a Grid or a Cloud interface.

In Figure 3, we present a high level view of access to resources through Grid and Cloud computing. The front-end to users presents itself as a thin layer allowing access to the compute elements and often involves a Local Resource Manager System (LRM or LRMS are often standard batch systems) – as we previously noted, while Grids provide access and scheduling of jobs on a "raw" worker nodes, Clouds instantiate Virtual Machines on worker nodes. Cloud interfaces may present themselves as simple Web front ends to more complex command line grid-like job interface. All tested model fit within this representation and conceptualization, what changes is the domain or boundary for virtualization.

2.2 Virtualization models testing, pro and cons

Many cloud providers currently exist, and we restricted our testing to five models: (1) \ Amazon/EC2 native platform a pure Web based front end, (2) a combination of the Nimbus project and Amazon/EC2, (3) The Clemson University Virtual Organization (VOC) model (,), (4) The Condor/VM model , and (5) The Kestrel model . Due to the different nature of those approaches, the flexibility on the clusters (existing communication in and out of the virtualized world, ability to monitor components outside the VM or not), we could not come up with a consistent set of standard monitoring and test suite, but rather tried to reach the same conclusion using different methods that we will describe in detail in the appropriate sections.

2.2.1 Amazon EC2

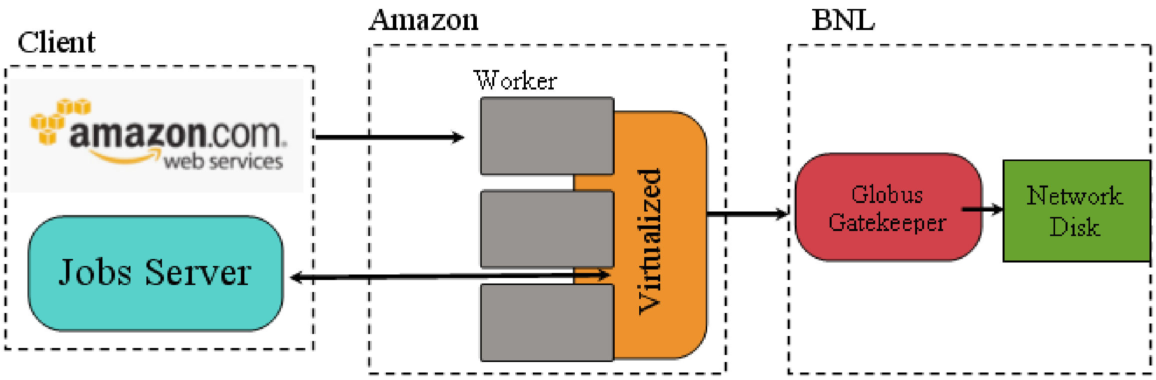

Figure 4: Architecture diagram of the EC2 interface. In this model, images are selected from the Amazon Web Services (AWS) interface and instantiated on the EC2. The images connect to the job server on startup, retrieve any necessary specifications, run their jobs, transfer the results back to the grid, and shut down.

In the native Amazon/EC2 approach, and while command line tools are available, the interaction with, and monitoring of, the system is mainly Web based. The general communication workflow is represented on Figure 4. Amazon EC2 uses a custom image format that can be created from any system using a variety of methods. In this exercise, a VMWare image with Ubuntu and the STAR libraries installed was converted to an Amazon Machine Image (AMI) using the EC2 AMI Tools, which are command-line utilities that help create an image and upload it to Amazon S3. The AMI was deployed using the standard Amazon Web Services (AWS) interface to EC2. Additionally, secure shell custom keys were put inside the image allowing direct connection from the outside in. Data transfer out was handled using either temporary ssh-key based transfer to a low security requirement storage site or using a more secure grid-proxy using myproxy delegation mechanism to a secured site. The system as it stands is not however self-sufficient in the sense the Amazon/EC2 interface allows starting, stopping VMs but does not allow, by itself, to create a virtual cluster equipped with its own RMS (local or global) a-la-Grid gatekeeper. To circumvent this problem, each VM instance connected on startup to a jobs server to receive job parameters, the workflow was a simple Monte-Carlo simulation requiring only a few parameter variation (and statistics were built for each set using a different seed per job). After completing the job and transferring the results back to the main facility at BNL, each instance shutdown.

A first observation is that since we have no access to the underlying infrastructure, the knowledge of efficiency is biased in the sense that whatever we asked as VM started (but we cannot differentiate between an over-commitment of VM "under the hood" or the start and death of some VM affecting farm occupancy). However, once the VMs were started and stable, we observed a 99% efficiency on a 100 VMs order of magnitude. The IO out of each WN was of the order of 5 MB/sec and we did not perform a scaling beyond a few tens of simultaneous transfers from multiple VMs (this data transfer sufficed for our exercise). Performance comparisons were also made between the various types of instances available: using the same program flow (using no more than 1 GB of memory), we tested multiple "instances" namely, the small, medium and large instance. The results were surprising: on a small instance, the ratio of CPU/clock time indicated a 40% CPU usage efficiency (at 0.085$/hour, this leads to a 0.21$/hour effective pricing) while on the medium instance, one of the same program would show 99% CPU efficiency (similar to the large instance performance but costing 0.21$/hour). We also observed that within one medium instance, we could in fact start our program twice in parallel, leading to an effective pricing of 0.09$/hour. Both Medium and Large instances were able to achieve approximately a factor of four increase in throughput compared to small instances.

The main benefit of Amazon/EC2 is its simplicity, its pay as you go feature but also its concept of VM repository (AMI / appliance). This concept of a repository of approved images would be needed for experimental group so the flow of random and un-checked software stack would be prevented. The pricing seem competitive to us as well: at this price, a 100 jobs week long simulation would cost ~ $1,510 (without storage) and a year long would be 79k$ all included (that is, no IT to maintain a cluster, no additional electricity or cooling bill). For university resources, we feel this cost is affordable and may well provide burst of resources required by experimental groups. On the down side, Amazon/EC2 security model has nothing to offer compared to the AAA features of the grid infrastructure (one may even qualify it as weak).

2.2.2 Amazon EC2+Nimbus

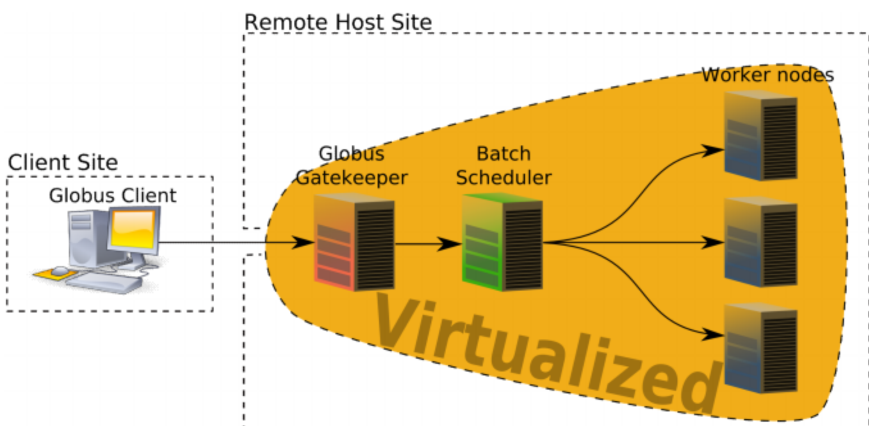

Figure 5: Architecture diagram of the EC2+Nimbus interface. In this model, a VM is started acting as a grid gatekeeper wile Worker Node (WNs) are started to handle the main processing. GK+WNs effectively form a standard Grid (virtualized) cluster and resource.

The Nimbus/EC2 model is similar to the previous model with slight twists and important improvements. The first important change is that not only VMs are started but they also started two different images: one acting as a standard Grid gatekeeper (with the full OSG software stack deployed “within”) and the others (the WNs), having a standard OSS client software stack in addition to the STAR required software + a batch system client registering to a Master running on the Gatekeeper (Grid server software stack) as well as containing a LRMS for batch management. The Worker Nodes’ batch clients subscribe to the gatekeeper and the combination of gatekeeper + WNs appears as a virtual cluster. In this approach, the resource appears as a virtual cluster and to the external user, is simply yet another site as part of federated set of resources as represented on Figure 5. In fact, STAR has used its standard Meta-scheduler (SUMS) with plugins to submit jobs in the most standard way for Grid operations while Nimbus allowed easily configuring and starting the virtual cluster.

To first order, this approach was much simpler of use for adding resources to our Grid operations pool. We scaled to a few hundred VMs but the efficiency of running job after the VM were instantiated dropped slightly to 85% efficiency on first submission, reaching 97% efficiency on second submission of the failed jobs. The drop was mainly due to two factors: a scalability issue in the version of the LRM (PBS in our case) which would drop communication between WN and the Master losing a fraction of the jobs and the interaction between the Grid job submission component (a.k.a. Condor-G) and PBS. After tuning and patching however, the final achieved efficiency is comparable to the one we can observe on the OSG native infrastructure as noted in section 1.2 and the benefits drawn from having a way to easily start/tear down virtual clusters on cloud resources tend to overcome the small loss. A con is that there is a delay between starting a gatekeeper and the WNs: this delay is due to the first identified problem of Cloud – as soon as the model become more complex, virtual machines need a contextualization that is, a mechanism to at least import a-posteriori information within the instance.

In our case, the IP of the master nodes is needed by the client batch system. Nonetheless, the testing and exploitation of such mechanism has allowed STAR to make truly breakthrough real-life data production run in 2009 as reported in reference . This was to our knowledge the very first real usage of Cloud computing resources with results used in an international conference (Quark Matter 2009 in this case).

2.2.3 The Virtual Organization Cluster (VOC) model at Clemson

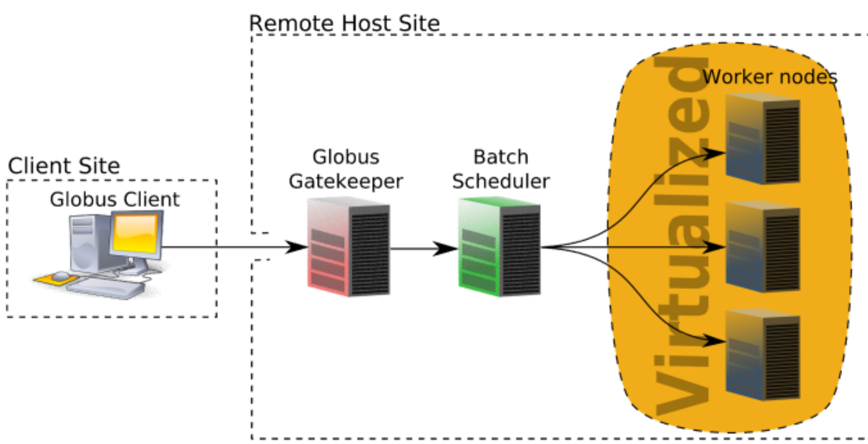

Figure 6: The Virtual Organization Cluster (VOC) model presents itself as a standard Grid job interface while the back-end is virtualized.

The VOC model has been described in great length in references (,) and is represented on Figure 6. The user interacts and submits jobs to a Globus gatekeeper using standard grid tools; the VOC system and daemons are responsible for dynamically starting (or shutting down) an appropriate number of VMs on the WN to satisfy both demand and resource allocation for a given VO. Note that the switch between native resource to virtualized resources was solely based on authenticating as a member of the STAR VO but other more complex mechanisms could be defined, following this general philosophy. Upon instantiation, a LRM "client" subscribes to the master as in the Nimbus/EC2 case. However, the virtualized domain is only the set of VMs (the gatekeeper may not be virtualized as in the previous model and may share Cloud and Grid submission). The publish/subscribe mechanism in this case provided ultimate transparency – from a job submitter stand point, there are no other differences in submission comparing to a standard Grid submission; the back-end VM ensures proper software provisioning and the user is completely agnostic to the technology used behind the scene.

Our testing was rather limited: from a few tens of VMs, the results can hardly be compared to the other models but rather, features could be extracted. Not surprisingly, the job efficiency (including in this case the ration of requested versus started VMs) was 100%. In this model the contextualization remains to some extent a site specific overhead but, since we made use of KVM , we minimized its burden by heavily relying on the snapshot feature KVM provides. This feature allows re-directing all changes done inside the VM to a local storage hence bypassing the need to mount a specific storage element and reducing local network IO overhead by writing over LAN. We also found in this phase of our testing that pre-caching the images on the local node (as possibly achievable) made the difference between a delayed and an immediate response of the demand satisfaction as supplied VM would start. The total overhead between submitting a job to the gatekeeper to having the job start within the VMs for an extended period of time and different load scenari showed the overhead to be less than 1% of the total workflow lifespan.

2.2.4 Condor/VM

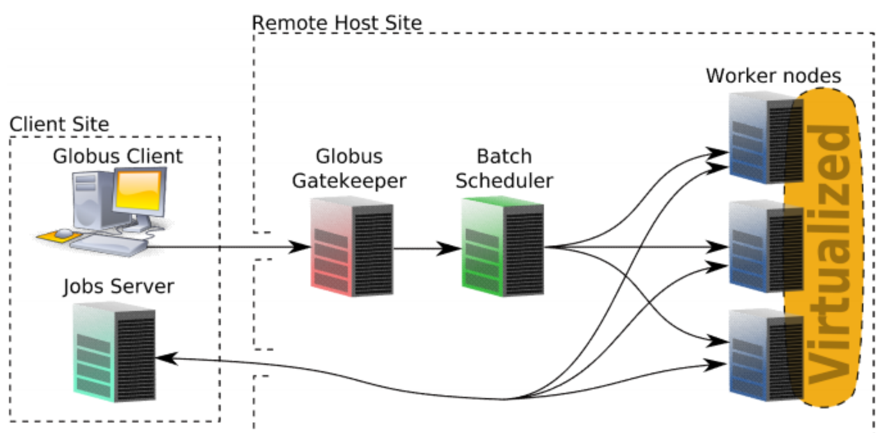

Figure 7: The Condor/VM model is a simple scalable approach whereas the submission to a gatekeeper only requests the startup of VMs onto the cluster. Condor/VM acts as the top layer and controller of the startup and shutdown of VMs.

The Condor/VM model is a simple model where the Grid gatekeeper accepts a standard grid job description but drops all directives but the one requesting to start a VM instance. The system accepts a VMWare image that is then submitted to the Condor cluster as a job. In this mode, a similar model to the Amazon/EC2 was adopted as Condor/VM is also not self-sufficient: after startup, one cannot communicate with the VM and hence, a mechanism to pull a job in has to be supplied (a similar mechanism as the one described in section 2.2.1 was used). Figure 7 represents the communication workflow in this model. All VMs were configured via NAT and had a connection from inside out, allowing transferring, as before, the data out from the cluster to the main STAR facility at BNL. The amount of contextualization in this case is minimal (no batch system to configure as the batch system, Condor, leaves outside the VM starting them as needed; mounting file system could be a standard operation; there was no network / DHCP address lease issues). Since Condor controlled the startup of the VM, a possible problem was that the VMs would expire after a certain pre-defined lifetime determined by the local Condor pool configuration. This time was 24 hours in this case and jobs were calculated to fit within this time modulo uncertainty inherent to Monte-Carlo simulations.

Over a scale of 500 VMs, 10% of the VMs never started on demand causing a net loss of slots at the start (but detectable and recoverable), 15% of the VMs stopped unexpectedly (crashed) and 5% of the jobs were killed due to VM lifetime expiration. The global job efficiency is therefore 73% (with peak at 80-85%). It was also necessary to pre-stage the image to all of the WNs before issuing the Condor job request. This reduced the time to start 500 machines from several hours to about an hour. Here again, the pro is that this Cloud presents itself with a standard Grid interface (but works best for those VO who already have a pull model, where an external job server provides the workflow information to the VMs which may be a con) and the setup of such model requires little contextualization which may be standardized across all supported VOs, reducing overhead on local staffing – the system may also support a vast amount of images (they could later be organized into VO specific repositories of images).

2.2.5 The Kestrel model

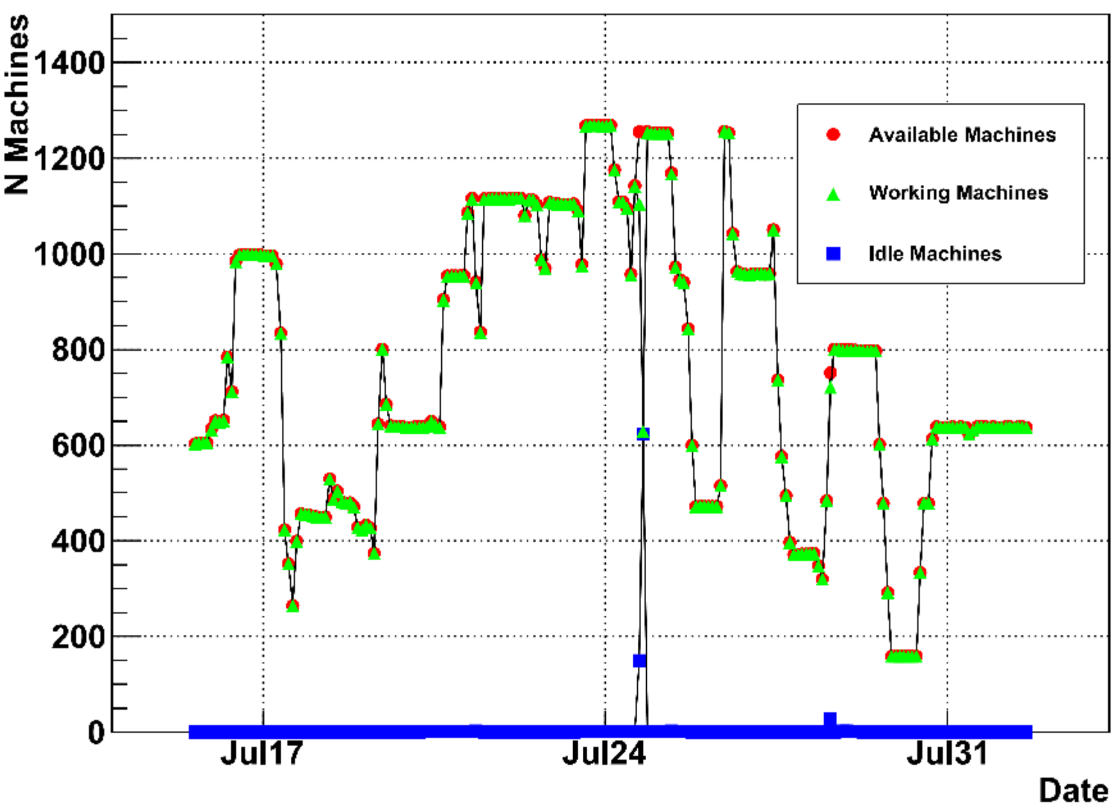

Figure 8: Tracking on the number of jobs as a function of time using the Kestrel system. The number of instantiated VM tracks with the number of available nodes (not used by other demands) indicating a good response of the system overall. A guaranteed allocation of 1,000 slots for a few days around July 21st shows we exceed the number of slots and took advantage of the farm empty slots.

The Kestrel model describes a method for controlling jobs on images across a variety of sites. VMs were started at the Clemson Palmetto cluster and at CERN using PBS [28]. Upon startup, the images connect to the Kestrel Manager, reporting their presence and any attributes that are defined (e.g. which STAR library version). The manager then schedules jobs to workers that satisfy the necessary requirements. Jobs can be submitted to the manager using command line utilities or instant messaging clients that support Jabber . Control of the number of available machines is not yet integrated into Kestrel but a home-made job watcher allowed us to feed the system with jobs. Figure 8 illustrate the response of the system. In red, we represent the number of available machine on a shared resource cluster (the number of machine which may be claimed at a given time) and in green, the number of VMs we could start within minutes. The blue curve represents the number if idle machine – a growth would indicate the Kestrel manager is not able to communicate with the VMs and feed a job. As we can see, a temporary glitch happened on July 25th and need better analysis but overall, the resource availability and our ability to harvest them as opportunity arise is astonishing. The difference between the number of machines started in PBS and those seen by the Kestrel manager on average differed by roughly 1%. The manager was able to utilize newly present machines on the order of seconds, but took on the order of several minutes to notice that a VM had disappeared.

It is important to mention that the usage of the Kestrel Cloud model is beyond an exercise: it represents a complex Monte-Carlo simulation followed by an event reconstruction requiring the full STAR framework to be available within the VM, database included. This simulation is the first use of Cloud for studying detector's systematic effects on DiJet productions. For this effort we added, within the VM, a MySQL server started upon booting the VM. The snapshot database, a 0.5 GB uncompressed file, is de-compressed upon startup before MySQL starts and since we use the snapshot mode of KVM, this leads to the database to appear local to the worker. Each VM/job then has its own database (hence not scalability issue and no network connection issues). This large simulation has been to date our largest use of Cloud computing model, both in simultaneous number of VMs (reaching the 1,000 for days to spanning already over a month length at 700 VMs average). Nearly 12 billion events have been generated by the Monte Carlo event generator using over 40,000 CPU hours. From these events, over 6.5 TB of data have been transferred back to BNL using grid-ftp, which the VMs are able to access using myproxy .

2.3 Current status, projections and perspectives

2.3.1 Current scale and evolution

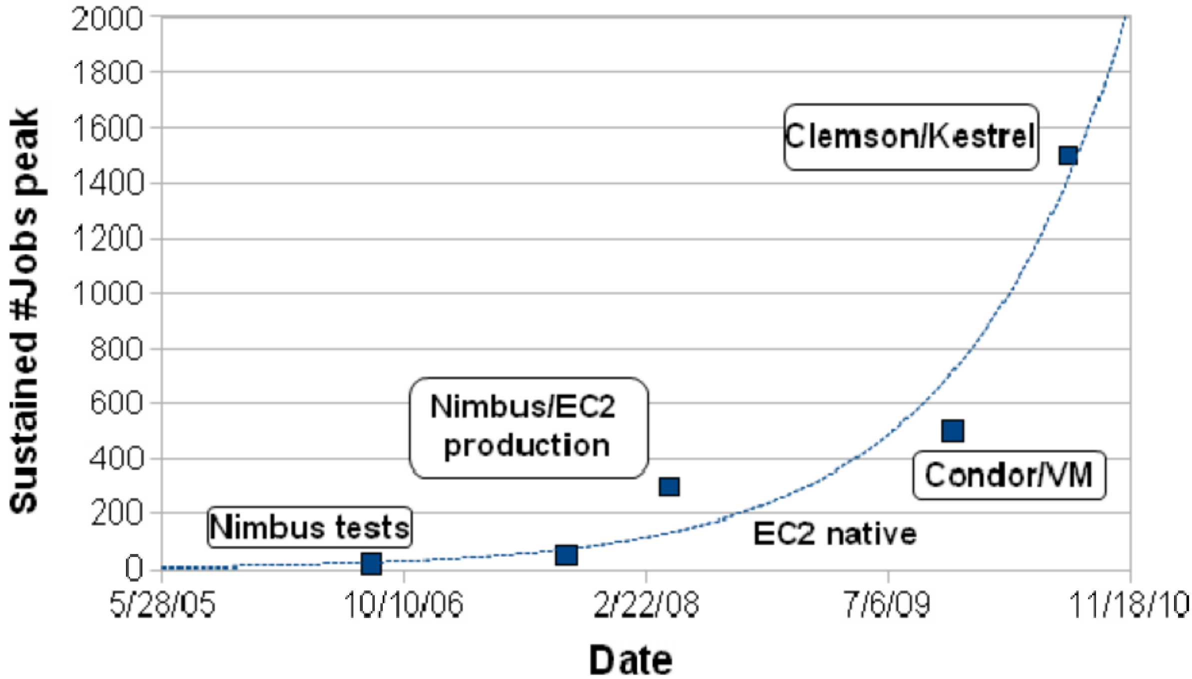

Figure 9: STAR testing scale as a function of time. From the initial Nimbus test in 2006 to the Kestrel model in summer 2010, the scale of sustainable operation has grown by two orders of magnitude, indicating a maturation of the technology and approaches. If this trend persists, we project that within the end of 2010, a number of jobs of the order several 10k to 100k jobs would be feasible and sustainable.

Figure 9 summarizes the scale and dates at which we performed the diverse exercises reported in this paper. While we have now reached and demonstrated the level of a 1,000 VMs and project that if this trend persists, the 10,000 to 100,000 levels will be reached by the end of 2010, an operational scalability which may satisfy STAR's need.However, we are to date a long way away from the sustainability provided by the OSG infrastructure, peaking at a near 800 k jobs .

2.3.2 Evolution of Cloud infrastructure, advantages and possible benefits

Nonetheless, Cloud computing shave generated not only a community wide interest but is likely at the peak of expectations . In parallel, the private sector such as Amazon.com, is aggressively addressing the demand of the HPC community by providing offers suited for large scale computational work : new "Cluster compute" instances on EC2 now provide 1.6 TB of storage, 23 GB of memory and 10 Bg network connection to the outside world for a $1.60/hour quoted price. It remains to be seen if this pricing is beneficial but once promises are finally delivered by Cloud and virtualization technology the possibility to really reach the ultimate goal of harvesting opportunistic resource will have arrived. The technology is so mature that several projects have also seen birth over the past few years (DOE Magellan project , NSF Azure Cloud) with aim to evaluate and/or create a national Cloud infrastructure for frontier science. It is to be noted as well that in our testing of Amazon/EC2, the network data transfer seemed inadequately low to also sustain a full data production operation requiring in STAR a total of nearly 3 Gb/sec transfer rate. The Cloud-ification of national laboratories could alleviate this problem by creating Cloud infrastructure on already network provisioned infrastructure. National Laboratories also provide low cost large data storage solution (mass storage, archival storage, disk storage) which we omitted in our price estimates (Amazon S3 is not currently competitive to the cost at National Laboratories for Peta-Scale storage, and lifetime storage guarantees are not yet part of many commercial cloud's real objectives). Several advantages of Cloud and/or virtualization appears to our community.

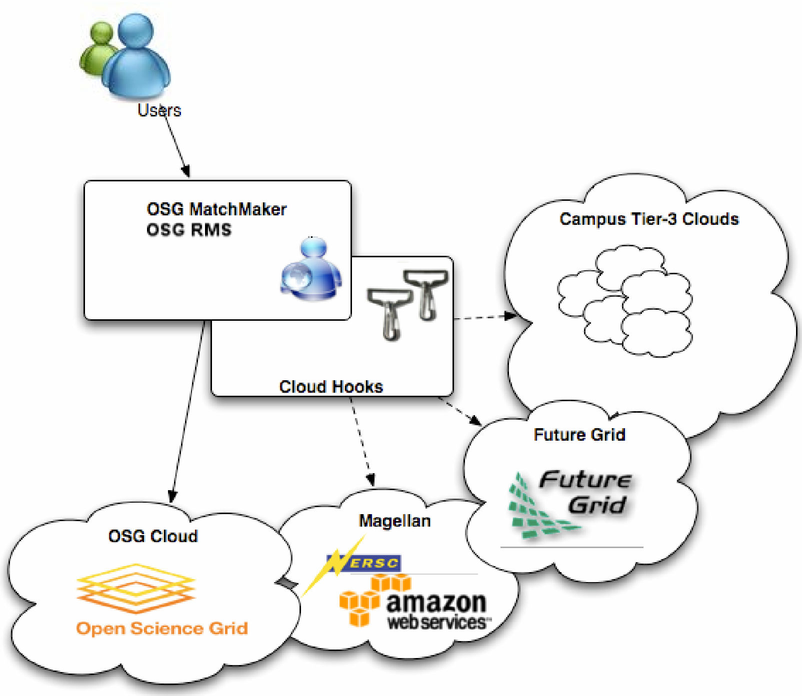

Figure 10: At the highest level of abstraction, a possible interface to the Open Science Grid could be a standard OSG Match Maker or standard GRMS with plug-and-play hooks to Cloud resources. While the Grid layer would act as a standard globus stack, hooks would handle submission to Clouds such as Amazon/EC2 (perhaps leveraging the EC2/Nimbus approach), Magellan Cloud (Eucalyptus plug-in).

Beyond National Laboratories, the ability of virtualization to carry along software and environment does not seem beneficial only in a context of Cloud and distributed computing. For a typical embarrassingly parallel community with simple workflows, and as we noted in section 1.2, maintenance of a vast variety of platforms is costly. In a typical Tier architecture model of distributed computing, cost goes beyond Tier-0 centers and span to the Tier-2 support where software installation is often done by a volatile workforce. Easy software provisioning with no need for post-installation validation would serve as an asset to the creation and use of local resources at Tier-2 centers.

The role of Cloud computing in the era of Exa-Scale computing is also to be understood – what used to be named "off the shelf" commodity hardware composed of nodes equipped with a few cores and nearly no parallel processing capabilities may become obsolete as the hype of Exa-scale and the multi-core era ramps up to take on the community's full attention. Cloud would allow sites to drop cost ineffective clusters and fold them onto larger multi-core resources (providing IO and inefficiency are not concerns) changing dramatically the notion of "Clusters" – in this scenario, a "Cluster" may simply be a piece of a larger resource pool provisioned by Cloud technologies.

[1] Cloud interfaces as we have seen remains however heteroclitic and this may reduce the penetration, early adoption or hardening of Cloud technologies in experimental groups. However, several projects are already on the way to provide unified interfaces to the cloud; example of such work is the incubator libcloud project , Delta-Cloud or StratusLab . A possible high level diagram of how to integrate clouds to the main stream projects such as the OSG is presented in Figure 10. Our experience with Cloud computing shows many approaches already provide (following the discussion in section 2) plug-and-play or transparent submission features (VOC, Nimbus) while others need to be supplemented by additional components to offer the full versatility the Grid provides. Such a proposal would need to be evaluated within the OSG consortium and may leverage the work and deliverables from the ExTENCI project

ExTEN, a joint TeraGrid and OSG project aimed to explore the use of VMs across TeraGrid and OSG test beds.

3 Conclusions

Cloud computing is an emerging technology with virtualization at its core. While Cloud computing offers an attractive model with its service components (PaaS, SaaS, IaaS) and may lack features such as providing a repository service for securely depositing approved VO-specific VMs, adequate network bandwidth for HPC, possibly common interfaces (and an easy way to contextualize a custom image) or strong security model, the virtualization capability alone offers a unique ability, long time ago foreseen as a benefit by experimental groups such as STAR, to "can" their environment in a fully consistent container with operating system, software, middleware, and all services embedded and hence, helping provision remote resources with tested and approved software for Physics result reproducibility and integrity. In our testing of Cloud resources and approaches, we found that Amazon/EC2 (before the appearance of their latest HPC instance support) provides competitive pricing in offering CPU power comparing to University clusters but insufficient data transfer bandwidth to satisfy the demand of HPC. The hybrid approach (Virtualization, Cloud-ification of Grid resources or clusters) provided by Nimbus and the VOC system offers the very attractive feature of allowing to present a standard grid interface to the end-user therefore, allowing for immediate integration of the Cloud paradigm in experimental production workflows with little to no changes. In contrast, condor/VM and Kestrel use a different approach and need to be supplemented with an external job feeder system, but their scalability to larger number of jobs and burden on the local team (contextualization) is heavily reduced by the simplifications. The Kestrel model though has allowed STAr to run stably an operation of a 1,000 jobs scale for a month in a full data mining context (full workflow with runtime gathering of Meta-data as real data production would require). We also observed that the community is rapidly researching and developing interfaces allowing for smooth integration of Cloud and Grid and highlighted a few usage cases of Cloud computing for HPC from partitioning of virtual clusters in the Exascale, easy software provisioning of Tier centers, and an easier distributed computing model for experimental groups focused on delivering stable World-wide renown science for the past decade. We infer Cloud and Grid are not orthogonal approaches and that a merging of both technologies is bound to happen at a fast pace. We predict 10k to 100k Cloud-based production operations would be reachable by the end of 2010. Finally, STAR is already running the most complex simulations and workflows on Cloud, requiring Meta-Data access (database) as well as full event reconstruction, similar to real-data production. In short, and modulo resolving the massive data transfer required for the input, STAR is one step away from being able to run real data production and reconstruction on Cloud in a truly and fully opportunistic manner.

References

[2] The Anatomy of the Grid: Enabling Scalable Virtual Organizations. I. Foster, C. Kesselman, S. Tuecke. *International J. Supercomputer Applications*, 15(3), 2001

[3] The Open Science Grid consortium, http://www.opensciencegrid.org/

[4] Knowledge management and virtual organizations. Yogesh Malhorta**,** Idea Group Inc (IGI), 2000 - Business & Economics, ISBN 1-878289-73-X

[5] The STAR collaboration: J. Adams et al., Experimental and theoretical challenges in the search for the quark gluon plasma: The STAR collaboration's critical assessment of the evidence from RHIC collisions. Nuclear Physics A, 757:102, 2005

[6] The Relativistic heavy Ion Collider - http://www.bnl.gov/rhic/

[7] The STAR computing resource plan 2009, STAR Notes CSN0474.

[8] The Particle Data Grid (PPDG) Collaboratory Pilot, http://www.ppdg.net/

[9] SunGrid and the STAR Experiment, http://www.sun.com/service/sungrid/brookhaven.pdf + Experience with on-demand physics simulations on the Sun Microsystems computing facility (SunGrid) at network.com, J. Lauret *et al* 2008 *J. Phys.: Conf. Ser.* 119 052024

[10] Integrating X-Grid into the HENP distributed computing model, L Hajdu, A Kocoloski, J. Lauret and M. Miller, 2008 J. Phys.: Conf. Ser. 119 072018

[11] Nimbus and Cloud Computing Meet STAR Production Demands, HPC wire April 02 2009 + Clouds make way for STAR to shine, iSGTW April 8th 2009 + Number Crunching Made Easy, Cloud computing is making high-end computing readily available to researchers in rich and poor nations alike, Newsweek May 2nd 2009 + Computing for the RHIC experiments, CHEP 2009 invited contribution.

[12] STAR Physics - Utilizing the Grid, PPDG report May 2004

[13] From BNL to KISTI: Establishing High Performance Data Transfer From the US to Asia, RHIC News January 13th 2009.

[14] Setting up a STAR Tier 2 Site at Golias/Prague Farm, P. Chaloupka et al 2010 J. Phys.: Conf. Ser. 219 072031

[15] rBuildeer.com, a product from rPath Inc. http://www.rpath.com/

[16] Xen hypervisor, an OpenSource industry standard for virtualization, http://www.xen.org/

[17] VirtualBox, a general-purpose full virtualizer for x86 hardware, http://www.virtualbox.org/

[18] National Institute of Standards and Technology, CS division, Cloud computing definitions

[19] Amazon Elastic computing Cloud, Amazon EC2, http://aws.amazon.com/ec2/

[20] Nimbus, an OpenSource toolkit for enabling IaaS on clusters, http://www.nimbusproject.org/

[21] M. A. Murphy and S. Goasguen “Virtual Organization Clusters: Self-Provisioned Clouds on the Grid submitted to Elsevier Journal of Future Generation Computer Systems special issue on Cloud Computing, (In Press)

[22] Contextualization in Practice: The Clemson Experience, M. fenn, S. Goasguen, J. lauret, proceedings of 13th International Workshop on Advanced Computing and Analysis Techniques in Physics Research (in press)

[23] The Condor project, a high throughput computing scheduler system and its Virtual Machine Applications,

[24] Kestrel: an XMPP-based framework for many task computing applications, L Stour, M. Murphy, S. Goasgen, In Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers (Portland, Oregon, November 16 - 16, 2009). MTAGS '09. ACM, New York, NY, 1-6. DOI= http://doi.acm.org/10.1145/1646468.1646479

[25] MyProxy, an OpenSource software for managing X509 Public Key management Infrastructure (PKI)

[26] Amazon/EC2 instances, http://aws.amazon.com/ec2/instance-types/

[27] Conceptual Grid Authorization Framework and Classification, Globus Grid Forum report GFD-I.038

[28] The STAR Meta-Scheduler, http://www.star.bnl.gov/public/comp/Grid/scheduler/

[29] The Portable Batch System (PBS), Enabling on-demand computing,

[30] The Kernel based Virtual Machine (KVM) system, http://www.linux-kvm.org/page/Main_Page

[31] OSG Gratia monitoring, http://gratia-osg-prod-reports.opensciencegrid.org/gratia-reporting/

[32] Gartner's 2009 Hype Cycle Special Report Evaluates Maturity of 1,650 Technologies,

[33] New Amazon EC2 Instance Type - The Cluster Compute Instance, addressing HPC needs

[34] Magellan Project, Exploring CLOUD Computing for DOE’s Scientific Mission.

[35] Computing in the Cloud (CiC), http://www.nsf.gov/pubs/2010/nsf10550/nsf10550.htm

[36] libcloud, a unified interface to the cloud, http://incubator.apache.org/libcloud/

[37] Delta.Cloud, many clouds one API, http://deltacloud.org/

[38] StratusLab, Enhancing Grid Infrastructure with Cloud Computing, http://www.stratuslab.eu/doku.php

[39] Jabber Technology, http://www.cisco.com/web/about/ac49/ac0/ac1/ac258/JabberInc.html

[40] ExTENCI: Extending Science Through Enhanced National Cyberinfrastructure.