Diversity in Computing Technologies and Strategies for Dynamic Resource Allocation

G. Garzoglio[1], **O. Gutsche**[1]

[1]Fermi National Accelerator Laboratory, Batavia, IL, USA

E-mail: garzogli@fnal.gov

E-mail: gutsche@fnal.gov

Abstract. High Energy Physics (HEP) is a very data intensive and trivially parallelizable science discipline. HEP is probing nature at increasingly finer details requiring ever increasing computational resources to process and analyze experimental data. In this paper, we discuss how HEP provisioned resources so far using Grid technologies, how HEP is starting to include new resource providers like commercial Clouds and HPC installations, and how HEP is transparently provisioning resources at these diverse providers.

1. Introduction

High Energy Physics (HEP) strives to develop a detailed mathematical understanding of nature at the smallest elementary level. Its science is based on the interplay between the theory framework that describes elementary particles and elementary forces between them; and the experimental detection of particles and measurements of their interactions. It calls for probing nature at ever increasing detail to unlock the last mysteries of our universe. Also called elementary particle physics, its experimental results are based on the analysis of many individual detector measurements in comparison to corresponding simulations that are based on the current understanding of the theory. Because of this, HEP was and is traditionally a very data intensive and trivially parallelizable science discipline.

We expect that the future will see increases in number and complexity of recorded particle interactions and corresponding simulations. Using the example of the LHC [1], the second data taking period will increase the center-of-mass energy and instantaneous luminosity significantly [2]. In addition, the LHC experiments will collect a higher rate of particle interactions to maximize their physics reach [3, 4]. This translates into increasing CPU resource demands that are needed to perform the simulation and reconstruction of these particle interactions. The future is expected to bring even more increases, we will have to answer the question of how we can provide access to sufficient CPU capacity to be successful in our physics research. We call this the capacity question.

Experience from the LHC also showed that these CPU resource demands are not constant over time. They vary significantly with external triggers like for example the operation schedule of the collider, the conference schedule and vacation schedule.

As an example, Fig. 1 shows the variation of number of active analysis users of CMS over time; and the re-reconstruction passes performed in 2011 leading up to the announcement of

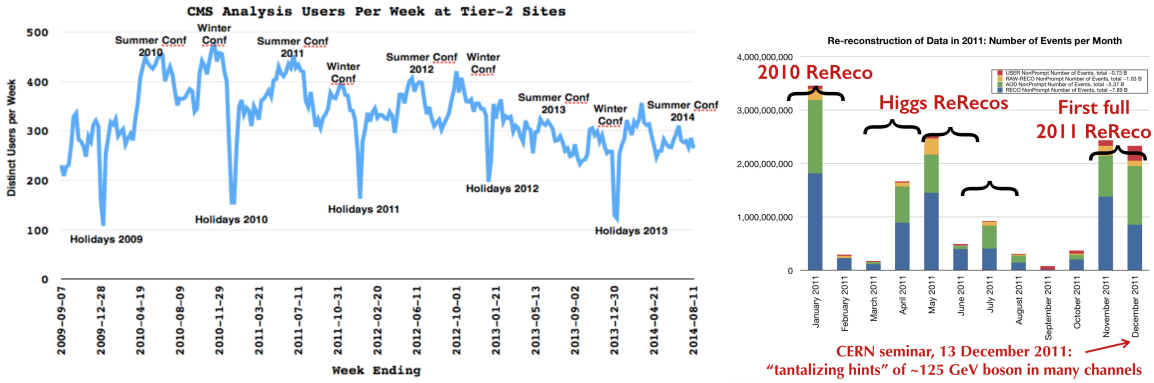

Figure 1. (*left:*) Number of CMS analysis users over time showing the variation of analysis activity, (*right:*) CMS re-reconstruction passes of data in 2011 leading up to announcement of hints for a ∼125 GeV boson at the December 2011 CERN seminar.

first hints for a ∼125 GeV boson, leading to the announcement of the Higgs Boson discovery in 2012 [5]. A computing model that adapts closely to these varying demand is generally called "elastic". In the future, we will have to answer the question of how to introduce more elasticity into the resource allocation. We call this the elasticity question.

In this paper, we want to discuss our view on solutions for the capacity and elasticity question. We want to look at the way HEP is currently provisioning CPU resources using the Grid and look at the new technologies and providers in the form of Clouds and HPC machines.

2. The HEP processing challenge

In general, the HEP processing challenge is trivially parallelizable. The simulation and/or reconstruction of individual particle interactions can be treated separately. The processing of one interaction does not need input from the processing of another interaction. It is one of the best examples of the High Throughput Computing (HTC) paradigm "that focuses on the efficient execution of a large number of loosely-coupled tasks" [6].

A single batch system with access to worker nodes to execute HEP applications was sufficient in the past to realize the HTC paradigm. In most cases, these were installations hosted by universities or research institutes that handled access and support locally. With increasing demand of the community, the need to access more and more resources that are also distributed across locations and administrative boundaries arose. The Grid provided the necessary tools and services to enable easy access to a diverse group of researchers to distributed resources. Different groups of researchers are organized in virtual organizations (VOs). Computing installations at universities or research institutes joined the Grid by allowing all users of a VO to execute applications on their local resources, therefore building a trust federation of computing resources. These Grid sites defined the list of VOs that were allowed access to their local resources. Sites are pledging amounts of their resources for individual VOs, therefore formalizing resource sharing at individual sites. Pledged resources not used by a VO can be used by other VOs and are called opportunistic resources.

The Grid was and is very successful and for example enabled the LHC experiments to fulfill all computing demands for the first LHC run. It is though based on batch systems which utilize directly worker nodes installed with a specific OS. Industry went a different way and used virtualization to establish a new resource sharing model: the Cloud. The Cloud replaces the batch system with a system that manages virtual machines on the physical hardware of the site and is run as a business. Commercial Clouds follow a pay-as-you-go model, where

all resources are strongly accounted and a customer pays for what was used. These business models promise near-infinity capacity and elasticity, which allows customers to use significant amounts of resources with very short ramp-up time as well as releasing them again when they are not needed anymore. In the Grid model, VOs plan their resource requests to get their work done in a defined period of time. The resource requests by VOs and the subsequent pledges by Grid sites are provisioned for peak to fulfill the VOs requirements. As shown before, the VOs demands are rarely constant over time and there are periods of lower computational demand by a VO. These free resources can be used opportunistically by other VOs following the sharing principle of the Grid. VOs that benefit from opportunistic resources themselves provide access to their unused resources at other times to the benefit of everyone. If cost effective, elasticity promised by Clouds could help in provisioning less resources permanently through the Grid and in times of demand allow for sufficient resource availability. Some commercial Cloud providers have developed in addition a spot price market, where excess unused capacity in the commercial Clouds can be given to customers at much lower prices through a bidding process. This is the Cloud equivalent to opportunistic usage of Grid sites.

A third new resource provider opening up for HEP applications are HPC installations. HPC stands for High Performance Computing and focusses on the "efficient execution of compute intensive, tightly-coupled tasks" [6]. They can, however, under certain circumstances execute HEP applications that follow the HTC paradigm. In recent time, the usage of HPC installations has become more and more accessible and feasible. HPC installations allocate resources to their users differently than traditional Grid and Cloud resources. Individual researchers or small groups of researchers are granted access to HPC installations through an allocation process. A peer review committee considers proposals designed more for individual researchers than large collaborations. In the end, allocations in time and capacity on HPC installations are awarded to successful proposals.

Table 1 shows an overview of the three resource provider types that we think will be most relevant in the near-term future to provide sufficient resource capacity and elasticity in our field.

Grid	**Cloud**	**HPC**
Trust Federation	Economic Model	Grant Allocation
• Virtual Organizations (VOs) of users trusted by Grid sites • VOs get allocations → Pledges – Unused allocations: opportunistic resources	• Commercial Clouds - Pay-as-you-go model – Strongly accounted – Near-infinite capacity → Elasticity – Spot price market	• Researchers granted access to HPC installations • Peer review committees award Allocations – Awards model designed for individual PIs rather than large collaborations

Table 1. Comparison of the Grid, Cloud and HPC resource provider types.

In the following, we would like to discuss the three resource provider types with emphasis on how we can use them with our HEP applications and how they can be transparently integrated into the current Grid-based setups. As the allocation models of the three provider types are rather different, we discuss how they can be integrated to support HEP needs.

3. The Grid Allocation Model

The Grid is based on a trust federation of resources (see Section 2). It allows transparent access to a large amount of resources for large groups of researchers. Researchers are typically organized in collaborations with many thousand members. The Grid is considered very successful. The prime example being the Worldwide LHC Computing Grid (WLCG) [7], which allowed the LHC experiments to rely on Grid-connected distributed resources from the beginning of their operation.

The Grid infrastructure is based on batch systems on large farms of computers called "worker nodes" that are reachable through Grid interfaces and services. For executing HEP applications, a task is typically split into smaller parts, or jobs, that can be executed in parallel. The Grid provides mechanisms to submit these jobs to a large amount of resources at the same time.

In the early days of the Grid, jobs were submitted directly or through a workload management system to the Grid interfaces of the sites. We call this the push era of the Grid. This evolved into pilot-based submission infrastructures. They are based on lightweight jobs called pilots to claim a job slot on a worker node. After initial checks of the worker node environment to verify basic functionality, the pilot signals the submission infrastructure that it is ready to receive work. It can then be assigned work in the form of a job from a task queue. We call this the pull era of the Grid. Most HEP VOs are now using pilot-based submission infrastructures. This approach allows for very late binding of the processing resource to the job, enabling the system to control scheduling and prioritization on a global scale. It reduces the failure rate of Grid job submission dramatically, because the job execution only starts after the resource was successfully claimed and validated. Pilot-based submission infrastructures allow for easy integration of non-Grid based resources. On the other hand, the infrastructure has generally more components than a push-based model and therefore the debugging can be more complex.

A good example of a pilot-based submission infrastructure is glideinWMS [8], which is based on HTCondor [9]. Fig. 2 shows a schematic view of a glideinWMS submission infrastructure. It is composed of HTCondor submit nodes implementing a queue of jobs; the VO frontend that monitors the submit nodes and initiates pilot submissions to the sites via the factory components; and the central manager that connects pilots that successfully claimed resources with jobs. HTCondor is used to form an overlay pool of all pilots as if all resources are in the same batch system, only spanning multiple distributed sites.

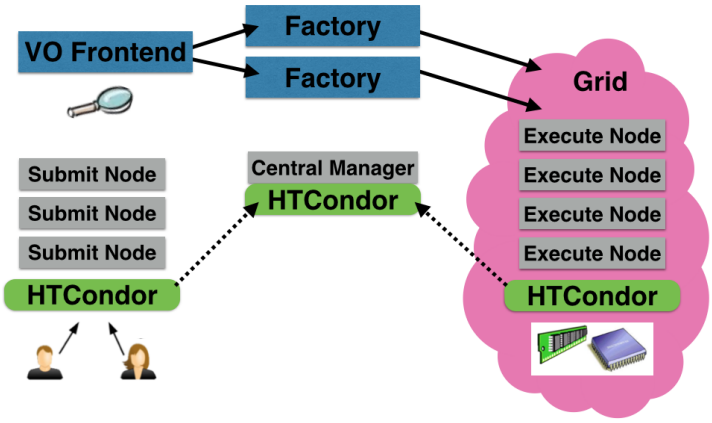

Figure 2. Schematic view of a glideinWMS submission infrastructure.

GlideinWMS is widely used and effectively implements the following concepts:

(i) The provisioning system (factories and central manager) can be shared amongst different communities and VOs.

(ii) Separate overlay pools of resources can be provisioned per community.

(iii) Each community has full control over their policies and priority settings within their pools.

The flexibility and ease-of-use of pilot-based submission infrastructures is important to enable the integration of Clouds and HPC installations for HEP.

4. The HPC Allocation Model

HPC installations have a long history in HEP, they are used for more HPC-like applications such as Lattice QCD [10] and Accelerator Modeling [11]. Recently the interest in the user HEP communities and of the HPC installations increased to also run traditional HEP framework applications. If the HPC installation is using an Intel-based architecture, it is possible to execute HEP applications unmodified. While for non-Intel-based architectures, the cross compilation of HEP applications using native compilers is necessary. In the following section, we give examples of each of the Intel-based and non-Intel-based architecture cases.

In the Intel-based architecture case, CMS received an allocation at the San Diego Supercomputer Center (SDSC) in 2013 to re-process specific proton-proton data [12]. SDSC operates a number of Intel-based HPC clusters ranging from ∼10k to ∼50k cores. Individual principal investigators (PIs) submit proposals and a committee meets every three months to award allocations in the form of CPU hours. Successful proposals have one year to use the awarded allocation. Follow up proposals can be submitted. They need to demonstrate the scientific impact of the previous research. CMS took part in the allocation award procedure at SDSC with the goal to reprocess additional proton-proton data a lot faster and earlier after the LHC run 1 finished. The additional data in question had not been processed during the run itself due to processing capacity reasons and was used to publish additional physics results not reachable by the LHC run 1 data. CMS used glideinWMS pilots submitted through ssh login nodes at SDSC, processed the data, and published more than 11 papers based on the SDSC allocation. CMS is now working on follow-up proposals. As a direct reaction to the CMS/HEP use case, SDSC is preparing to give access to its HPC installations through Grid Compute Elements (CEs), making it even easier to integrate SDSC resources into pilot-based submission infrastructures.

In the non-Intel-based architecture case, Atlas was able to utilize the PowerPC-based Mira Supercomputer at Argonne National Laboratory. The machine has a similar allocation award procedure than SDSC. Proposals are required to demonstrate the ability to enable new science through the usage of Mira. Atlas cross-compiled the Alpgen event generator [13] using the IBM XL compilers for Mira and effectively ran multiple instances of Alpgen in parallel [14]. Miras almost 800k cores are subdivided into nodes and Miras minimal partition size is 512 nodes. This allows Atlas to use backfill queues to run Alpgen jobs on individual free partitions. Currently, jobs are submitted manually through a custom workflow system. In the future, the goal is to integrate Mira into the Atlas pilot-based submission infrastructure.

Both examples show that the usage of HPC installations for traditional HEP applications is possible and we can expect more usage examples in the future.

5. The Cloud Allocation Model

The computing activities of experiments are not constant and, rather, follow peaks and valleys of demand as shown in Fig. 1. These are influenced by external factors, such as instrument operations, social events, conferences, holiday festivities, etc. Until recently, the only feasible approach to satisfy these peaks consisted in building computing centers at National Laboratories and Universities and procuring enough computing resources there. This spurred the creation of

resource federations and sharing agreements, embodied by Grid consortia, so that potential large available off-peak capacity could be utilized opportunistically by all members of the federation [15, 16]. As the needs for peak capacity grows, however, this strategy is becoming cost-prohibitive.

The emergence of Commercial Clouds provides a new solution to this problem. Resources have a cost only when utilized, as if they were rented rather than owned. Commercial providers offer seemingly-infinite resource capacity available on short time scales. As such, the cost of computing time is the same when renting one computing resource for 1,000 hours or 1,000 resources for one hour. There are several challenges for Cloud computing to become competitive with the Computing Centers managed by the scientific community, in terms of cost, reliability, and ease of use. Several HEP experiments and facilities, including Atlas, CMS, STAR, NOvA as well as BNL and Fermilab, are working with Cloud providers to address these challenges [17, 18, 19, 20]. Currently, the areas of work include the development of realistic economic models, resource provisioning, networking, storage, and on-demand services. We will go into more detail for all of these in the following.

5.1. Resource Provisioning

Commercial Cloud providers implement proprietary application programming interfaces (API) to enable the provisioning of resource. To avoid vendor lock-in, many HEP communities rely on commonly used job management layers, such as HTCondor, to abstract access to different providers. HTCondor enables access to different Clouds by supporting the proprietary interfaces of a few Cloud Providers as well as the Amazon EC2 interface. This is a widely emulated interface that enables access to several providers, although with limitations, considering that it is not a standardized interface. This strategy makes provisioning technically possible, but does not alleviate the challenge of balancing demand for computing with cost. Two major challenges for our current technology include

(i) the ability to expand and contract provisioned resources to control cost while the job queue is full;

(ii) fully integrate market price-based solutions to provision Virtual Machines.

The first challenge is mainly related to policy. The priority of computational activities among scientific communities are not always straight forward. Some activities may be urgent but considered low priority. A combination of urgency and priority drives the policy to expand and contract the pool of resources to balance costs. For the second challenge, a popular example of a market-based provisioning solution is Amazon Spot pricing. The user bids the maximum price that he is willing to pay for the resource. Until the market price is below the bid, the user has access to the resource. When the market price goes above the bid, the resource is retired within a few minutes. The price varies following the demand for resources on the market. Many HEP workflows are good candidates to use Spot pricing. The Grid, in fact, implements similar preemption mechanisms when users run on opportunistic resources i.e. resources made available on the Grid, but not owned by the job submitter. On the Grid, preemption is typically implemented by the batch scheduler, which kills the job processes to make the resources available to the higher-priority job (typically a job of the resource owner). To run effectively on the Grid, most computing operations had to be made already resilient to job failure and, thus, could cope with preemption. Considering that jobs are generally submitted in bulk as part of a computing campaign, the commonly used mechanisms to achieve that include

jobs checkpointing: the state of the job is saved and resumed when the failed / preempted jobs are relaunched

bookkeeping: the global state of the computation is saved through appropriate bookkeeping, so that failed jobs in a campaign can be resubmitted and the computation resumed without duplication of work (e.g. SAMWeb database of files already "consumed" in a dataset [21])

stateless jobs: jobs in a campaign are all equivalent to each other (e.g. some Monte Carlo production) and can be simply relaunched

minimal unit of computation: applications process very shorts unit of computation (e.g. 1 event for 10 minutes), thus relaunched computations have minimal duplication (e.g. Atlas Event Service [22])

5.2. Economic model

With commercial Clouds becoming mainstream, computing centers at Laboratories and Universities have the choice to dynamically expand their resource pool. The decision of when to expand the pool depends on several factors, including cost. To properly manage the size of the pool, computing centers face the challenge of fully understanding their costs and compare them with the commercial providers. Preliminary cost estimates to run a "modern" computing core for one hour at National Laboratories, such as Fermilab and Brookhaven, are about $0.03 and $0.04 respectively [23]. For comparison, a basic virtual machine with 1 core at Amazon (m3.medium instance) cost $0.07. The same instance, however, cost as low as less than $0.01 using Spot pricing [24]. In addition to understanding the local cost for computing, however, predicting the costs of Commercial Cloud resources can also be a challenge. To develop an understanding of such costs, we have run computational campaigns with real physics applications on Amazon Web Services (AWS). In 2014, Fermilab has run a few Monte Carlo simulation campaigns for the NOvA experiment [25]. The largest consisted of 3,300 jobs distributed between AWS and the local Fermilab Cloud infrastructure (FermiCloud) for a scale of 1,000 jobs each (see Fig. 3). On AWS, we used dual core virtual machines (at $0.14 / h) running two jobs per machine. The total cost was $449, split between computational charges for $398 and data transfers for $51. Limiting the amount of egress data transfers e.g. by limiting auxiliary information such as log files, was key to contain that cost. Since then, however, AWS has made available to research institutions special data egress fee waivers to further reduce those costs (see Sec. 5.4).

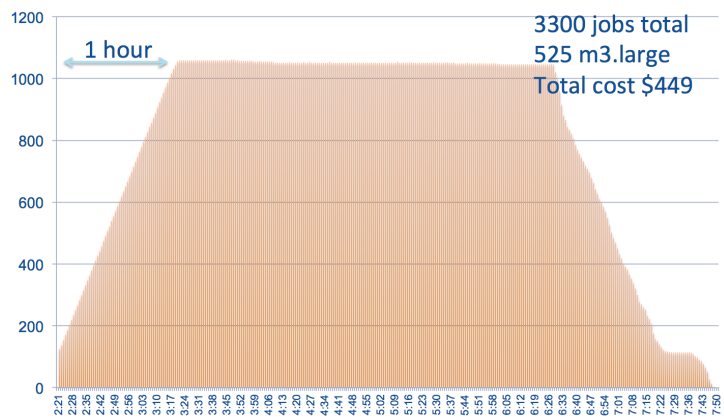

Figure 3. Shown is the NOvA MC campaign running 1000 jobs in parallel on AWS.

To continue the integration of the job management infrastructure with AWS for the NOvA experiment, AWS has awarded an educational grant to Fermilab. The goal of the grant is to demonstrate the continuous availability of the resources at AWS throughout a year. We plan

to run data reconstruction for the 2014 / 2015 NOvA dataset for raw data and Monte Carlo in 16 computational campaign for a total of 2M CPU hours. As our capabilities improve, we aim to demonstrate that using the spot pricing market for this type of physics computation is cost effective and as available as the Fermilab resources.

5.3. Storage

The effective utilization of compute resources depends on the effective handling of data. In general, locality of the data is known to make a difference. In particular for output data transfer, abrupt termination is a concern when provisioning resources with spot pricing. Storage locality, however, is not always more cost effective, according to our model. We consider the case where multiple jobs are submitted to the Cloud for execution and terminate approximately at the same time. We want to transfer the output data back to the home institution. We evaluate two strategies, graphically represented in Fig. 4:

(i) Jobs attempt to transfer data directly to the remote storage at the home institution; the storage system will accept the data transfer with a certain limit on the ingress bandwidth. If data transfer is coordinated among all jobs, some jobs will transfer the data and then terminate, while others will wait in a queue. Irrespectively, virtual machines will be idle i.e. blocked on IO without running any computation for as long as the data transfer last. In addition to the data egress charges, running these idle virtual machines will contribute to the total cost.

(ii) Jobs transfer data to local storage at AWS (Simple Storage Service - S3). Because of the high level of scalability, all jobs will be able to transfer the data at the same time using high-bandwidth. The full dataset can be transferred asynchronously directly from S3 to the home institution later on e.g. initiating the transfer from the home institution. The data egress charges will be the same as in the previous strategy. This time, however, we pay for storing the data in S3, instead of idle virtual machines.

Depending on the bandwidth available to the storage system at the home institution, the number of running VMs and the amount of data to transfer, one strategy may be more cost effective than the other. For example, Lets assume to run 1,000 jobs on 1,000 VMs of type m3.medium ($0.07/h), each transferring 1 GB of output. The cost of data egress is $120 irrespective of the strategy. The cost of storing the 1 TB data in S3 is generally negligible if the transfer is automatically triggered at the end of the job and then the data is erased. If the transfer is initiated manually, however, storage has a cost. For example, if it takes a week to start the transfer back to the home institution, it is about $8. Adding related costs, such as the costs of Input / Output requests to S3, the total cost would be approximately $132. In comparison, if we transferred the data directly from the VMs to the home institution, the cost would vary (statistically) depending on the aggregate bandwidth to storage. For example, for 20 Gbps, the cost of idle VMs would be $8 for a total of $128; for 2 Gbps the cost would be $78, for a total of $198. We are preparing to measure the cost of these strategies in realistic testbeds in the summer 2015.

5.4. Networking

In the Grid model, participating institutions are connected through scientific networks, such as Internet2 and ESNet in the US. These organizations absorb the cost of data transfer and, as such, this cost is hidden to the end users. This often leads to a feeling among the user community that network is a "commodity", rather than a resource. With the transition to the Cloud model, many of these costs are exposed. Commercial Cloud providers typically allow data ingress for free, but charge for data egress and some internal data transfers [24]. Historically, however,

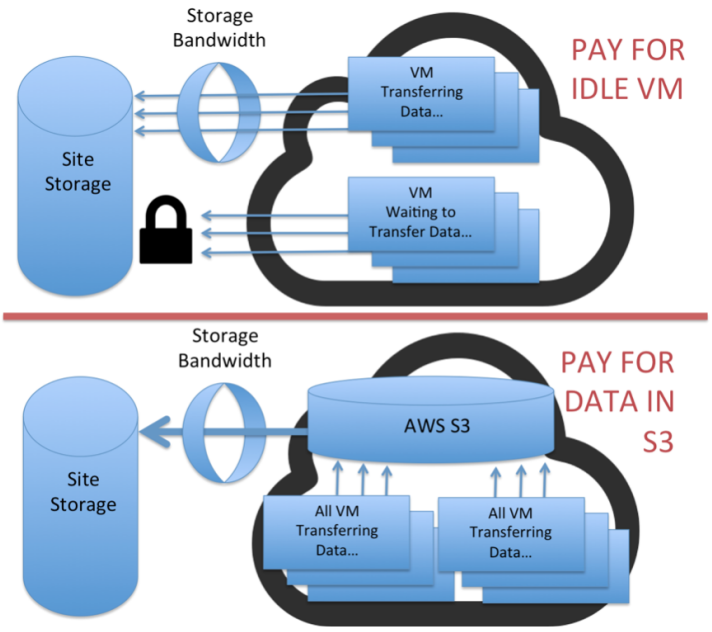

Figure 4. Strategies to effectively handle output data for cloud applications.

the data egress fees have acted as an economic barrier to the adoption of Cloud computing for many scientific communities. Over the past year, the scientific networks have worked to improve their network peering with AWS [26]. Absorbing much of the cost of data transfer, they are in the unique position to negotiate data egress fee discounts for the scientific community. In particular, Internet2 and ESNet have negotiated a data egress fee waiver with AWS, by which data transfers costs below 15% of the total monthly cost are waived. As these agreements are new, some of the contractual terms are still being refined to make this an opportunity for both universities and national laboratories. Together with cost reduction, the scientific networks are working to improve the connectivity to AWS. Today ESnet peers with AWS at three AWS zones in Seattle, Sunnyvale (CA), and Ashburn (VA). Using the default routed network, this peering allows for a connectivity of 10 GE at each point, with a planned 100 GE peering at Seattle to come in the summer. In addition to the general routed network, AWS offers a DirectConnect service, whereby network ports are reserved for certain sites. Through a pilot project, this allows for a dedicated peering of 10 GE with BNL at Ashburn and of 20 GE (2x10GE) with ESNet at Seattle. This reserved bandwidth can be exploited by setting up dedicated circuits between the site and AWS.

5.5. On-demand Services

Scientific computations rely on several dependent services, such as databases, software distribution, storage, job submission queues, etc. Some of these services, such as the ones offering data caching, are known to improve the efficiency of the computation when local. As the scale of the scientific workflows running on Cloud platforms increases, the ability of instantiating dependent services also on the Cloud becomes important to improve the efficiency of the computation and, ultimately, reduce cost. We refer to these services, which are instantiated following the scale of scientific workflows that are executed on the Cloud, as on-demand services. Through our R&D programs, we have started to experiment with on-demand services such as software distribution and job submission queues. We use the CERN Virtual Machine File

System (CVMFS) [27] for software distribution. The system relies on a network of software repositories made available to remote clients through the HTTP protocol. As such, the system can scale through the adoption of web caching services, such as Squid. Our early attempts to run scientific workflows on AWS used software distribution caches at Fermilab. The lack of cache locality at AWS caused high latencies in the remote access of the software through the Wide Area Network. In addition, it caused a large number of access requests directed at Fermilab, rather than at a local cache, and overwhelmed the Fermilab distribution system. To overcome these limitations, we have developed mechanisms to elastically scale web data caching services and use them for software distribution (see Fig. 5). In short, we run a Squid server in a virtual machine at AWS. The server can be accessed through an Elastic Load Balancer, which defines a single entry point to the data caching system for the clients. The network traffic on the Squid VM is monitored through an AWS service called "Autoscaling Group". As the traffic increases because of demand, the autoscaling group can elastically instantiate additional Squid servers. These, after their cache is loaded, enable the automatic scaling of the data distribution service. In addition, the autoscaling group can retire Squid servers as the load to the system decreases below a set threshold.

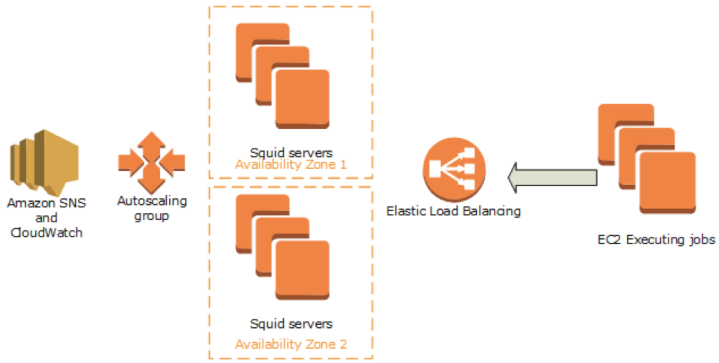

Figure 5. Mechanisms to elastically scale a squid web data caching service in an AWS service called "Autoscaling Group".

Since web data caching is a service with a limited, generally disposable, state, the automated scaling of the service is relatively straight forward. More care had to be taken for the automated scaling of job submission queues. In particular, the scaling down of the service required for the system to wait the draining of the user jobs, a process that may take days. This is an active area of R&D.

6. Virtual Facility

The elasticity promised by commercial Cloud providers can not only be used to the benefit of VOs or science communities. Also traditional Grid sites can benefit from it.

In what we call the virtual facility approach, a Grid site would not provision anymore all needed resources through physical hardware. That hardware would need to be operated and maintained in the sites own data centers. Sites could fulfill their users needs through a combination of owned and rented resources, therefore alleviating the effect of having to provision for peak demand and rather be more elastic and cost effective. Sites would develop a cost model for physical resources and commercial Cloud resources and would optimize costs by choosing a balance between them. The agreement between users and sites about service levels of resources would stay the same. The site, however, would need to make sure that their usage

of Cloud resources would yield in the same service levels as their own physical resources. This would include investigating storage and on-demand auto-scaling service solutions for Clouds as discussed above. In the end, sites could provide complete solutions for their users with their jobs running transparently on physical or rented hardware, while optimizing costs for the sites.

7. Community Solutions

We have discussed three different resource providers and how they can be integrated to run HEP applications. Utilizing these providers efficiently and at high scale however requires technical knowledge and effort. Large VOs, such as the LHC VOs, have their own teams of experts that take care of integration and operations. Not every VO, however, can afford this level of sophistication. To address this limitation, organizations have been funded to provide the community at large, even beyond HEP, with the capabilities, services, and infrastructure to execute their applications at scale on multiple resource providers. One such organization is the Open Science Grid (OSG) [15].

The Open Science Grid was initially founded with the goal to share the infrastructure of the LHC experiments and other Experiments, Universities, and Laboratories in the US. From the beginning, the emphasis was to include scientific communities beyond HEP to transfer the expertise of the LHC experiments to run HTC applications at high scale to multiple scientific disciplines. The community effort is based on the premise that resource owners want to share their resources to maximize the benefit to all without relinquishing control of their local resources. Major clusters at Universities and National Laboratories connect to the OSG and control the sharing policies locally.

One goal of OSG is that researchers use a single interface to all kind of resources: resources they own; resources others are willing to share; resources that they have an allocation on (for example HPC installations); resources they buy from a commercial (Cloud) provider. OSG focuses on making this technically possible.

OSG operates a shared production infrastructure, called the Open Facility. It is based on glideinWMS and enables researchers to easily and efficiently run on different resource providers. OSG also maintains and advances a shared software infrastructure, called the Open Software Stack. It enables researchers to use common tools and techniques to execute their applications at scale on the OSG. In addition, OSG takes care of documentation and training of technologies and techniques to spread the knowledge across researchers, IT professionals, and software developers, creating and Open Ecosystem all research groups to benefit from the advances of the distributed high scale HTC model.

Fig. 6 shows a schematic setup of the OSG Open Facility, where different user and user groups are provided with facilities tailored to their needs to connect to the OSG.

Single Principal Investigators (PIs) can benefit from the OSG Connect service, whereby OSG operates a login node for the researcher and provides disk space and a software repository. Through the common submission infrastructure, OSG assists the PI to provision resources across the OSG facilities. OSG maintains also a dedicated instance of the OSG Connect service to serve the resource needs of researchers from the HPC community. They are awarded allocations on OSG through the XRAC process of XSEDE [28].

Universities and laboratories that are connected to the OSG have the possibility to also benefit from unused capacity at other OSG facilities by moving excess local load to the OSG, as well through HTCondor and glideinWMS, therefore virtually expanding their local resources.

LHC experiments and other large VOs use the OSG directly by operating OSG sites and using them through their own submission infrastructures, but gaining access to other OSG facilities as well.

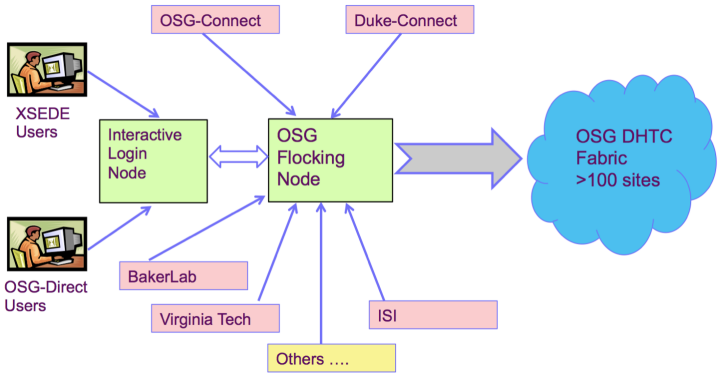

Figure 6. Schematic setup of the OSG Open Facility.

8. Resource Allocation Models

All three presented resource provider types have very different resource allocation models. The Grid allocates resources through pledges given to VOs at sites. These pledges are constant over time and usually given for a year at a time and then renewed. Commercial Clouds follow an economic model where users pay only for what they use. There is no predefined time structure, provisioning 1 CPU for 1000 days costs the same as 1000 CPUs for 1 day. HPC installations grant allocations on their facilities in the form of CPU time that can be used in a given time frame. All three allocation models have different time frames and different mechanisms of defining the amount of resources allocation (Grid: job slots, Cloud: cores, HPC: CPU time). Although still in its infancy, the integration of these allocation models would simplify the operations of the composite cyberinfrastructure. We dont have an immediate solution on how to seamlessly integrate these resource providers and also newer ones that have not been mentioned here, but we think it is important to bring up the issue and pose the question to the community to start the discussion and develop solutions on how to combine these resource allocation models.

9. Summary and Outlook

In this paper, we discuss how the resource usage for HEP and other sciences is changing to include more types of resource providers. The Grid is being augmented by commercial Clouds and HPC providers. Service developed for the Grid, such as workload management systems, are enhanced to integrate the new resource providers through pilot-based submission systems like glideinWMS. The integration of commercial Clouds poses challenges in several areas. As these are addressed, we envision that Clouds will provide an ever larger fraction of the resource pool through the use of cost competitive models, such as the spot market price.

HPC installations are currently used to solve specific problems in HEP computing. As the community develops more experience in the operations of HPC, we envision growth opportunities in the resource pool from this provider type. We discussed the concept of the virtual facility combining owned and rented resources to optimize costs and provide more elasticity for the users. We think that this concept has several benefits for a facility and we expect to hear reports from implementations and modifications to the concept in the future. We also discussed a community solution based on the Open Science Grid, which enables the whole community from individual researchers to large VOs to benefit from the advances in distributed large scale HTC application execution. We think the approach of the OSG is an excellent example how the advances coming from the Grid world combined with new resource providers can be easily utilized by a larger community. In the end, we discussed that although we can use a variety of resource providers transparently through our submission infrastructures, the allocation model

are sufficiently different that new solutions need to be found for a tighter integration.

10. Acknowledgements

We would like to thank the various funding agencies from all over the world that made the research discussed here possible, particularly the Department of Energy in the United States. Many thanks to all our colleagues who helped gathering and organizing information and for fruitful discussions, especially Stuart Fuess, Burt Holzman, John Hover, Bo Jayatilaka, Jim Kowalkowski, Ruth Pordes, Panagiotis Spentzouris, Steve Timm, Margaret Votava, Frank Würthwein.

References

[1] Evans L and Bryant P 2008 *Journal of Instrumentation* **3** S08001 URL `http://stacks.iop.org/1748-0221/3/i=08/a=S08001`

[2] CERN 2014 *5th Evian Workshop on LHC beam operation* (Geneva: CERN) organisers: Lamont, M; Meddahi, M; Goddard, B URL `https://cds.cern.ch/record/1968515`

[3] Artz S, Bauss B, Boterenbrood H, Buescher V, Degele R, Dhaliwal S, Ellis N, Farthouat P, Galster G, Ghibaudi M, Glatzer J, Haas S, Igonkina O, Jakobi K, Jansweijer P, Kahra C, Kaluza A, Kaneda M, Marzin A, Ohm C, Oliveira M S, Pauly T, Pttgen R, Reiss A, Schfer U, Schffer J, Schipper J, Schmieden K, Schreuder F, Simioni E, Simon M, Spiwoks R, Stelzer J, Tapprogge S, Vermeulen J, Vogel A and Zinser M 2015 *Journal of Instrumentation* **10** C02030 URL `http://stacks.iop.org/1748-0221/10/i=02/a=C02030`

[4] Bawej T A, Behrens U, Branson J, Chaze O, Cittolin S, Darlea G L, Deldicque C, Dobson M, Dupont A, Erhan S, Forrest A K, Gigi D, Glege F, Gomez Ceballos G, Gomez-Reino Garrido R, Hegeman J G, Holzner A G, Masetti L, Meijers F, Meschi E, Mommsen R, Morovic S, O'Dell V, Orsini L, Paus C M E, Petrucci A, Pieri M, Racz A, Sakulin H, Schwick C, Stieger B B, Sumorok K, Veverka J and Zejdl P 2014 The New CMS DAQ System for Run 2 of the LHC Tech. Rep. CMS-CR-2014-082 CERN Geneva URL `https://cds.cern.ch/record/1711011`

[5] Chatrchyan S *et al.* (CMS) 2012 *Phys.Lett.* **B716** 30–61 (*Preprint* `1207.7235`)

[6] EGI Glossary: `https://wiki.egi.eu/wiki/Glossary_V1`

[7] Bird I, Bos K, Brook N, Duellmann D, Eck C *et al.* 2005

[8] Sfiligoi I, Bradley D C, Holzman B, Mhashilkar P, Padhi S and Wurthwein F 2009 *Proceedings of the 2009 WRI World Congress on Computer Science and Information Engineering - Volume 02* CSIE '09 (Washington, DC, USA: IEEE Computer Society) pp 428–432 ISBN 978-0-7695-3507-4 URL `http://dx.doi.org/10.1109/CSIE.2009.950`

[9] Thain D, Tannenbaum T and Livny M 2005 *Concurrency - Practice and Experience* **17** 323–356

[10] Blum T, Van de Water R, Holmgren D, Brower R, Catterall S *et al.* 2013 (*Preprint* `1310.6087`)

[11] Amundson J, Macridin A and Spentzouris P 2014 *IEEE Comput.Sci.Eng.* **16** 32–41

[12] Press Release: SDSCs Gordon Supercomputer Assists in Crunching Large Hadron Collider Data `http://ucsdnews.ucsd.edu/pressrelease/sdscs_gordon_supercomputer_assists_in_crunching_large_hadron_collider_data`

[13] Mangano M L, Moretti M, Piccinini F, Pittau R and Polosa A D 2003 *JHEP* **07** 001 (*Preprint* `hep-ph/0206293`)

[14] Childers T, Le Compte T, Uram T and Benjamin D 2015 *21st International Conference on Computing in High Energy and Nuclear Physics (CHEP2015)* URL `https://indico.cern.ch/event/304944/`

[15] Pordes R, Petravick D, Kramer B, Olson D, Livny M, Roy A, Avery P, Blackburn K, Wenaus T, Wrthwein F, Foster I, Gardner R, Wilde M, Blatecky A, McGee J and Quick R 2007 *Journal of Physics: Conference Series* **78** 012057 URL `http://stacks.iop.org/1742-6596/78/i=1/a=012057`

[16] Kranzlmller D, de Lucas J and ster P 2010 *Remote Instrumentation and Virtual Laboratories* ed Davoli F, Meyer N, Pugliese R and Zappatore S (Springer US) pp 61–66 ISBN 978-1-4419-5595-1 URL `http://dx.doi.org/10.1007/978-1-4419-5597-5_6`

[17] Timm S, Garzoglio G *et al.* 2015 *21st International Conference on Computing in High Energy and Nuclear Physics (CHEP2015)* URL `https://indico.cern.ch/event/304944/session/7/contribution/448`

[18] Taylor R *et al.* 2015 *21st International Conference on Computing in High Energy and Nuclear Physics (CHEP2015)* URL `https://indico.cern.ch/event/304944/session/7/contribution/146`

[19] Balewski J, Lauret J, Olson D, Sakrejda I, Arkhipkin D, Bresnahan J, Keahey K, Porter J, Stevens J and Walker M 2012 *Journal of Physics: Conference Series* **368** 012011 URL `http://stacks.iop.org/1742-6596/368/i=1/a=012011`

[20] Colling D *et al.* 2015 *21st International Conference on Computing in High Energy and Nuclear Physics (CHEP2015)* URL https://indico.cern.ch/event/304944/session/7/contribution/230
[21] Mengel M, Norman A *et al.* 2015 *21st International Conference on Computing in High Energy and Nuclear Physics (CHEP2015)* URL https://indico.cern.ch/event/304944/session/4/contribution/463
[22] Wenaus T *et al.* 2015 *21st International Conference on Computing in High Energy and Nuclear Physics (CHEP2015)* URL https://indico.cern.ch/event/304944/session/4/contribution/463
[23] Ernst M, Hogue R, Hollowell C, Strecker-Kellog W, Wong A and Zaytsev A 2014 *Journal of Physics: Conference Series* **513** 062053 URL http://stacks.iop.org/1742-6596/513/i=6/a=062053
[24] Amazon EC2 Pricing http://aws.amazon.com/ec2/pricing/
[25] Norman A 2015 *21st International Conference on Computing in High Energy and Nuclear Physics (CHEP2015)* URL https://indico.cern.ch/event/304944/session/9/contribution/465
[26] Hover J 2015 *HEPiX Spring 2015 Workshop* URL https://indico.cern.ch/event/346931/session/9/contribution/20
[27] Blomer J, Buncic P, Charalampidis I, Harutyunyan A, Larsen D, and Meusel R 2012 *Journal of Physics: Conference Series* **396** 052013 URL http://stacks.iop.org/1742-6596/396/i=5/a=052013
[28] XSEDE: Extreme Science and Engineering Discovery Environment https://www.xsede.org/