CERN-IT evaluation of Microsoft Azure cloud IaaS

C. Cordeiro[(1)], A. Di Girolamo[(1)],L. Field[(1)], D. Giordano[(1)], H. Riahi[(1)], J. Schovancova[(1)], A. Valassi[(1)], L. Villazon [(1)]

Reviewed and approved by: S. Coriani[(2)], N. Erfurth[(2)], P. Koen[(2)], X. Pillon[(2)], H. Scherer[(2)], R. Stütz[(2)]

[(1)] CERN-IT, [(2)] Microsoft Azure

CERN, 14-02-2016

1 Introduction

This document reports the experience acquired from using the Microsoft Azure cloud Infrastructure as a Service (IaaS) within the distributed computing environments of the LHC experiments. The activity has been conducted in the framework of a joint collaboration between Microsoft and CERN-IT initiated by the round table meeting of 2nd of February 2015 [1].

1.1 Goals

As agreed in the aforementioned round table meeting, the collaboration foresees four phases:

1. informal technical evaluation of Azure,
2. deployment at progressively larger scales,
3. initial assessment of costs and TCO, and
4. investigation of procurement models.

In this document we summarize the experience acquired in the execution of Phase 1. The objective of Phase 1 is "[…] to achieve an informal technical evaluation of Azure and Azure Batch to understand the technology, the APIs, ease-of-use, any obvious barrier to integration with existing WLCG tools and processes. […] the resulting feedback will give both parties a chance to assess the basic technical feasibility for more advanced activities" [1].

[1] *Contact: domenico.giordano@cern.ch, cristovao.cordeiro@cern.ch*

DOI: 10.5281/zenodo.48495

2 Work organization

This activity has been conducted by members of the CERN-IT Support for Distributed Computing group involved in similar projects of integration, deployment and operation of commercial cloud resources within the computing workload systems of the LHC experiments [2]. Guidance and support has been provided by Microsoft Azure Solution Architects (Mr. H. Scherer *et al*.).

2.1 Support meetings

Weekly synchronization meetings (one hour long Skype calls) were held throughout the full duration of the activity to report progress, discuss issues and exchange feedback. Participants:

- CERN: D. Giordano, C. Cordeiro, L. Villazon, and
- Microsoft: H. Scherer, P. Koen, X. Pillons, N. Erfurth.

2.2 Dedicated meetings

Two dedicated topical meetings were also organized.

2.2.1 Microsoft Azure introduction (Apr. 10)

A day-long introduction of Azure IaaS was given by Mr. H. Scherer on site at CERN. Azure offered services were discussed, including Azure Batch, as well as the processes to configure Azure accounts, define storage accounts, import images and manage user data and secrets. Description of the different data centres and locations was also covered. During the tutorial the Azure web portal (https://manage.windowsazure.com) was largely used and illustrated. The future availability of a new portal, Azure Preview Portal (https://portal.azure.com/) was also anticipated.

2.2.2 CernVM deployment support (Aug. 21)

A full afternoon hands-on remote meeting via Skype call, organized to evaluate integration of CernVM image in Azure IaaS.

3 Timeline and milestones

The activity held a kick-off meeting in March 17th, during which the list of test cases was drafted and the initial technical aspects to acquire access to the Azure resources were covered. In particular the creation of a Microsoft Azure Sponsored account was proposed to enable large-scale tests. A Microsoft Outlook account (cern-mscloudtest@outlook.com) has been created and connected to the sponsorship. For the development and small-scale tests, individual CERN MSDN subscriptions ($50 per month of credit) were considered sufficient to start with.

Table 1 reports the major milestones and achievements during the full activity.

Date	Milestone
Feb. 02	Round table meeting
Mar. 17	Technical kick-off meeting
Mar. 19	Configured CERN MSDN accounts ($ 50/month) for initial test activity
Mar. 23	Sponsored Account ready ($10k until end of June). Signature of Loan Agreement needed
Apr. 10	Full day introduction to Azure
May 11	Start migration to Azure Resource Manager
June 16	Start usage of Sponsored Account. Extended Sponsored Account to end of September, increased to $ 40k
Aug. 14	CernVM image for Azure IaaS is available
Aug. 21	CernVM deployment support meeting
Sep. 4	Start large-scale tests
Sep. 22	Extended Sponsored Account to end of November
Sep. 28	About 2,700 vCPUs provisioned across 3 data centres
Nov. 18	Completed large scale test of D3 VM series
Nov. 27-29	Large-scale test: provisioned 4,600 vCPUs. All VMs steadily run workloads of LHC experiments.
Nov. 30	Sponsored Account termination

Table 1: List of major milestones during the full activity

4 Technical design

4.1 Architecture

Figure 1 shows the architecture of the set-up used to provision resources within Azure, to steer LHC experiment workflows to the running VMs and to monitor the VM status and usage accounting. VMs are provisioned using an agent able to interact with the Azure API. Two different agents, communicating with two different APIs, have been used (namely Vcycle and CERN-ARM wrapper, see later) in order to respectively evaluate and adopt the two available Azure provisioning models, Service Manager[2] and Resource Manager[3]. Two different VM images were also used: CernVM [3] and CentOS 6 [4]. In both cases the images included a minimal amount of packages and basic services such as CVMFS [5], the XRootD [6] client and the Ganglia [7] monitoring daemon. The experiment related libraries and configuration data are accessed by the applications at runtime through the HTTP-based CVMFS read-only file system.

[2] https://msdn.microsoft.com/en-us/library/jj838706.aspx
[3] https://azure.microsoft.com/en-us/documentation/articles/resource-group-overview/

The CERN EOS [8] data storage system is used, when needed, to access input files and to store output files, exploiting remote data access across the WAN using the XRootD protocol.

4.2 VM image and size

The size and OS selected for the provisioned VMs is based on the technical specifications already adopted in other commercial cloud contexts, which proved to be satisfactory for running Monte Carlo simulation jobs of LHC events. The minimum size is a single vCPU VM with 2 GB of RAM, 20 GB of free disk space and a public IPv4 address. This last requirement on public IPv4 addresses is based on the similar request made in recent price enquiries for the acquisition of commercial cloud resources, and corresponds to an analogous configuration adopted for the CERN resources. Multi-vCPU VMs have also been tested.

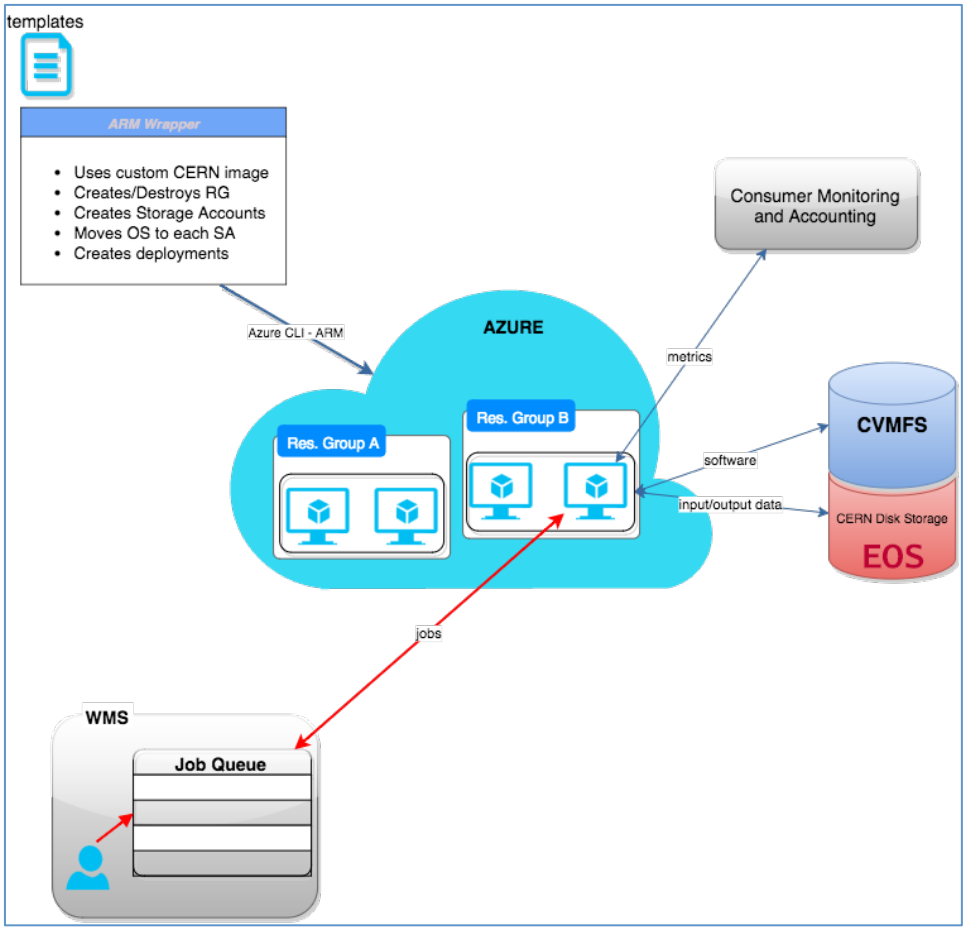

Figure 1. Architecture of the set-up used to provision resources within Microsoft Azure, to steer LHC experiment workflows to the running VMs and to monitor the VM status and usage accounting

4.2.1 CernVM image

CernVM is a virtual machine image based on Scientific Linux 6 combined with a custom, virtualization-friendly Linux kernel. It is based on the µCernVM boot-loader distributed as a read-only image of about 20 MBytes containing a Linux kernel and the CernVM-FS client. The rest of the operating system is downloaded and cached on demand by CernVM-FS. The virtual machine still requires a hard disk as a persistent cache, but this hard disk is initially empty and can be created instantaneously, instead of being pre-created and distributed. Since August CernVM image for the Azure platform has been made available by the CERN CernVM team[4]. The image, imported in the Sponsored Account, has been extensively used for the tests described in Section 6.

4.2.2 CentOS image

CentOS is an alternative image option which is also based on the Linux OS family. In Azure this image is natively provided by OpenLogic in different versions, containing the installation of the Basic Server packages. Among the different versions offered by the provider, CentOS 6.x are the preferred ones according to the technical specifications mentioned above.

Since this image does not include any LHC experiment related environment, an initial setup is required. This can be achieved either through a contextualization process at VM startup or by snapshotting the image after installation of needed packages. To avoid systematic heavy routines running on each VM provisioning and since Azure allows the usage of custom images the latter option was adopted. The resulting snapshot contains the minimal amount of software and configurations taken from previous cloud experiences [9] allowing a correct execution of the experiments workloads.

4.2.3 Microsoft Linux Agent

The Microsoft Azure Linux Agent (waagent)[5] manages the interaction between the virtual machines and the Azure Fabric Controller. It provides many functions for Linux and FreeBSD IaaS deployments in several areas, including Image Provisioning, Networking, Diagnostics and VM Extensions. As the responsible component for the communication between the platform and the VMs, this agent needs to be addressed when working with custom OS images. This means that unlike the CentOS snapshot where the *waagent* was already setup in the base OS, the CernVM image had to be modified in order to include this agent.

[4] http://cernvm.cern.ch/portal/downloads
[5] https://github.com/Azure/WALinuxAgent

4.3 VM Contextualization

As referred above, the experiments' related software and configuration data are setup at runtime, more specifically when the VM starts. This process is addressed as the contextualization stage.

Azure has different ways to provision a start-up script to the VM, depending mainly on the type of OS in use. In the CentOS snapshot use case, the *defacto* contextualization package Cloud-init[6] was not available, leaving then users to rely on Azure's data injection mechanisms like Custom Data[7] and VM Extensions[8]. With CernVM in the other hand the image was prepared in such a way that Cloud-init is also enabled, allowing then for a more versatile generation of the contextualization user data.

With any of the specified mechanisms, the provided user data always perform the same actions changing only its input format and internal interpretation.

4.4 Infrastructure *modus operandi*

The user interaction with the Azure IaaS changes significantly when comparing with other cloud IaaS, in the sense that in Azure there is a larger resource management flexibility given to the user to configure the desired resources. On the other hand this implies that every underlying resource necessary for the successful instantiation of virtual machines must be properly provisioned. For every VM users have to adequately create and link components like Storage Accounts, Network Interface Cards, Dynamic IPs, Virtual Networks and Resource Groups or Cloud Services, which are basically resource containers.

4.5 Azure API

As other cloud providers Azure also offers ways to interact with the infrastructure. The already cited web user interfaces are good options for monitoring and managing resources at small scale, but prove to be suboptimal in case of large-scale deployments and automated deployments. For this use case other options are available, including two REST APIs, a Command Line Interface and a SDK for Python, the latter two being wrappers on top of the REST APIs.

The two REST APIs correspond to two different management mechanisms:

- Azure Service Management (ASM)[2], and
- Azure Resource Management (ARM)[3].

[6] https://help.ubuntu.com/community/CloudInit
[7] https://azure.microsoft.com/en-us/blog/custom-data-and-cloud-init-on-windows-azure/
[8] https://msdn.microsoft.com/en-us/library/azure/dn832621.aspx

The ARM API is newer than the ASM API and cross-compatibility between them is not fully ensured. The ARM API is officially the primary API while the ASM API is called the classic API.

4.5.1 Azure Service Management

The ASM API is often referred to as the "classic" way of handling resources through a dedicated management portal[9]. It is a REST API where all the operations are performed over SSL and mutually authenticated using X.509 v3 certificates. Based on the feedback received by Microsoft Azure Solution Architects, this API should in time be phased out and all ASM provisioned resources migrated to the ARM model.

Within the ASM model, a key component is the Cloud Service that represents a "container" for the application resources. A Cloud Service can host a maximum number of 50 VMs and the maximum number of Cloud Services per subscription is 200 (non modifiable limits)[10]. The Cloud service has a public dynamic IP address and acts as a gateway for all the underlying VMs within its private network, accessible through a 1:N port mapping. This networking model does not fit with the CERN requirement of having all VMs with dynamic public IP addresses per VM, and the workaround of having a single VM per Cloud Service would bring to a maximum capacity of 200 VMs. Most of those limitations are not present in the alternative provisioning model (ARM) therefore large-scale tests have been performed using the ARM API.

4.5.2 Azure Resource Management

The ARM API is a JSON driven REST API and it is linked to its own dedicated web portal[11] as well. The biggest change and advantage with this API is the JSON template deployment model. This template is a JSON file that contains cloud resources definitions and allows the user to make a request for multiple services and resources in one single call.

The template provisioning mechanism is a declarative method to instantiate resources, where all the underlying deployment instructions are moved to the infrastructure side. The provisioned resources will fall under containers entitled as Resource Groups, which stand for a logical set of correlated cloud resources.

4.6 VM provisioning system

In this section the two provisioning applications used to automatically interact with the Azure APIs are described.

[9] https://manage.windowsazure.com
[10] https://azure.microsoft.com/en-us/documentation/articles/azure-subscription-service-limits/#subscription-limits
[11] https://portal.azure.com/

4.6.1 Vcycle

Vcycle [10] is an open-source VM lifecycle manager that implements the VAC model on IaaS[12] cloud providers, allowing for an automated creation/destruction of the VMs. It is one of many tools used in the HEP community in order to deliver elastic capacity from cloud providers. Vcycle supervises the VMs and instantiates/shutdowns VMs depending on the state of the experiments central task queue. This approach enables elastic capacity management of the available cloud resources.

Vcycle interacts with different clouds via specific Python plugins (connectors) exploiting the preferred cloud API. We have developed the connector for Azure. This Vcycle connector was developed using the Azure SDK[13] for Python focusing on ASM API. It is available in a public GitHub repository[14].

4.6.2 ARM-CERN wrapper

Another provisioning option is to use what is already provided by Azure. With the ARM API all the provisioning steps are on the provider side, leaving the consumer with JSON template generation only. In other cloud contexts this approach is often defined as “orchestration” because the effective provisioning handling, fault tolerance and retry is moved on the provider side.

In order to use the ARM approach, a custom wrapper (ARM-CERN wrapper) was developed around the Azure XPlat-CLI[15] with the goal to allow a user to build JSON templates and issue the provisioning calls automatically based only on configuration. This wrapper is exclusively prepared for the Azure infrastructure in order to prepare the experiments’ environment in Azure from a custom image. It takes care of making a copy of the OS image in each created Storage Account, when working with custom OS images.

The number of VMs provisioned per Storage Account is configured to respect the suggested maximum of 40 VMs as commented in the Azure Service Limit documentation: “You can roughly calculate the number of highly utilized disks supported by a single storage account based on the transaction limit. For example, a Basic Tier VM, the maximum number of highly utilized disks is about 66 (20,000/300 IOPS per disk), and for a Standard Tier VM, it is about 40 (20,000/500 IOPS per disk). However, note that the Storage Account can support a larger number of disks if they are not all highly utilized at the same time.”

[12] https://www.gridpp.ac.uk/vac/
[13] https://azure.microsoft.com/en-us/documentation/articles/python-how-to-install/
[14] https://github.com/vacproject/vcycle
[15] https://github.com/Azure/azure-xplat-cli

4.7 VM Monitoring

The status of VMs has been monitored in the provisioning and operation phases by means of several monitoring systems, namely the Azure web portals and the CERN instantiated Ganglia monitoring.

4.7.1 Azure monitoring

As reported above, both available Azure portals have been used for the interaction with the IaaS, either to manually provision few resources or to monitor the status of the provisioned resources. The views offered by the monitoring dashboard are extremely detailed, with possibility to navigate down to the level of each single resource component, whilst keeping the full connection to the dependency tree. Availability of alarms and alert reports are beneficial for issue tracking and debugging.

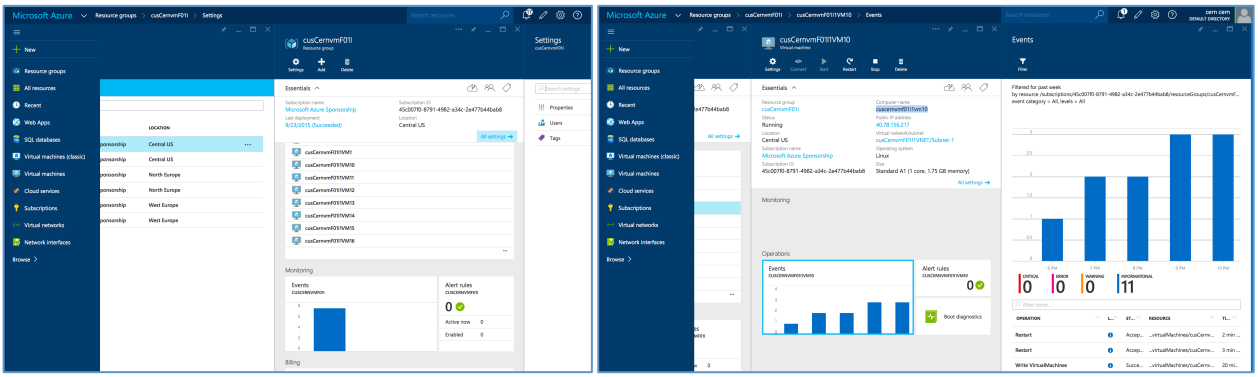

Figure 2. Snapshot of Azure portal to manage the used sponsored account.

4.7.2 Ganglia

Ganglia is a scalable distributed monitoring system for high-performance computing systems such as clusters and Grids. It has already been demonstrated to work for cloud scenarios and how to scale the system is understood [11]. The VMs run a Ganglia *gmond* service to communicate directly with a receiving Ganglia *gmond* collector sitting on the head-node. The Ganglia *gmetad* service points to these collectors as data sources, fetching all the metrics data from them through a TCP channel and storing them in a local Round-Robin database. This data is then interpreted and presented in a web interface provided by Ganglia's Web frontend.

For this specific activity a dedicated Ganglia server has been instantiated[16], monitoring separately Azure VMs provisioned for each experiment (ATLAS, CMS, LHCb). The monitoring metrics per VMs cover CPU related quantities (load, percentage of cpu time spent in idle, system, nice, etc), memory related quantities (free memory, swap, cache, etc), network related

[16] http://azucern-ganglia-monitor.cern.ch

quantities (inbound and outbound traffic) and disk related quantities (disk size, free, full, etc). Figure 3 shows a snapshot of the Ganglia Web UI monitoring VMs provisioned in Azure.

5 Results

This section describes the three test cases performed: scale tests, profiling resources and ability to run experiment's jobs. Three different Microsoft data centres have been targeted, namely Central US, North Europe and West Europe, with maximum allowed capacity of 1500, 1500 and 1000 VMs respectively.

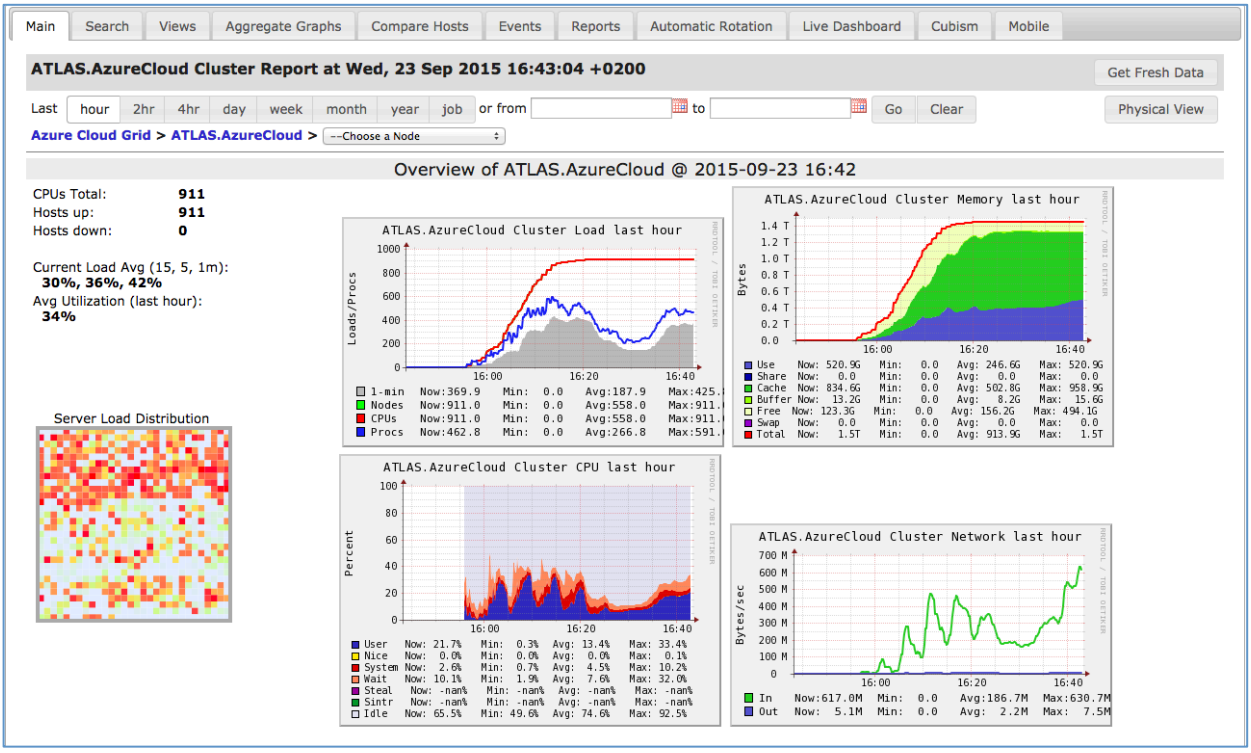

Figure 3 Snapshot of the CERN Ganglia monitoring instance for Azure Cloud. In the plots the Azure resources provisioned for the ATLAS VO during a scale-out test are shown.

5.1 Scale tests

Scale tests have been performed with the goal of testing the infrastructure performance in regimes of large number of VMs provisioned. The capacity targeted is the maximum available capacity of the sponsored subscription per data centre. The adopted approach consists of firing requests for a number of Resource Groups, each containing a fraction of the total number of targeted VMs. The fractioning reflects the suggested limit of 40 VMs per Storage Account and

the default limit of 100 Storage Accounts per subscription. The CernVM image has been used for most of the tests. The small size of the image (around 20 MBytes) eases the process of copying the image into each Storage Account and duplicating it as many times as the number of requested VMs.

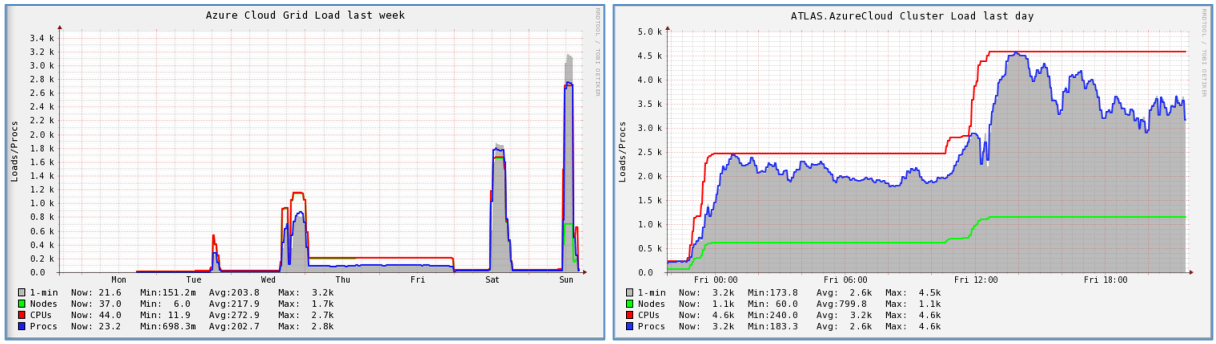

Figure 4. Ganglia plot of the number of single-vCPU provisioned (red curve) for different scale (left). Detail of the largest test is also shown (right).

Figure 4 shows two Ganglia plots of the ramp-up and ramp-down phases in different scale tests. A maximum number of 4,600 vCPUs has been provisioned across the three available data centres, namely 2,000 vCPUs in both North Europe and Central US and 600 vCPU in West Europe.

5.2 Profiling resources

Performance measurements and monitoring are essential for the efficient use of computing resources as they allow selecting and validating the most effective resources for a given processing workflow. In a commercial cloud environment an exhaustive resource profiling has additional benefits due to the intrinsic variability of a virtualized environment. Ultimately it provides additional information to compare the presumed performance of invoiced resources and the actual delivered performance as perceived by the consumer.

In this phase, all provisioned VMs were profiled using different benchmark metrics and the results analysed. The adopted benchmarks span from generic open-source benchmarks (encoding algorithm and kernel compilers) to LHC experiment specific benchmarks (ATLAS KitValidation [12]) and fast benchmarks based on random number generation. Profiling has been performed across the three targeted data centres and different flavors of VMs (Basic A1, Standard A1 and A3, Standard D1 and D3[17]).

Figure 5 shows the comparison among VMs of type Standard_A1 provisioned in three data centres. Data have been aggregated per data centre and per CPU type. The two separate peaks are representative of two different Intel series. Even if all VMs have been provisioned under the

[17] https://azure.microsoft.com/en-us/documentation/articles/virtual-machines-size-specs/

class Standard_A1, it is evident that a set of VMs, mainly provisioned in central US data centre, has a performance that is about 50% worst than the rest of provisioned VMs.

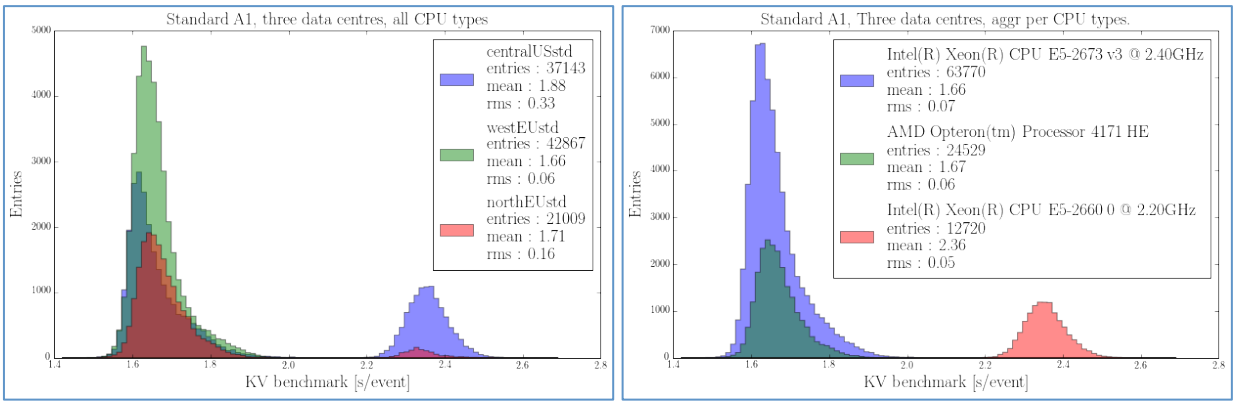

Figure 5. Benchmark performance for VMs of Standard A1 series, aggregated per data centre (left) and per CPU type (right).

Figure 6 shows the comparison between two benchmarks used to profile resources, namely a fast benchmark based on random number generation and the KV benchmark that simulates collisions inside the ATLAS detector. The correlation between the two measurements appears to be good, even considering the large measurement range due to different CPU architectures.

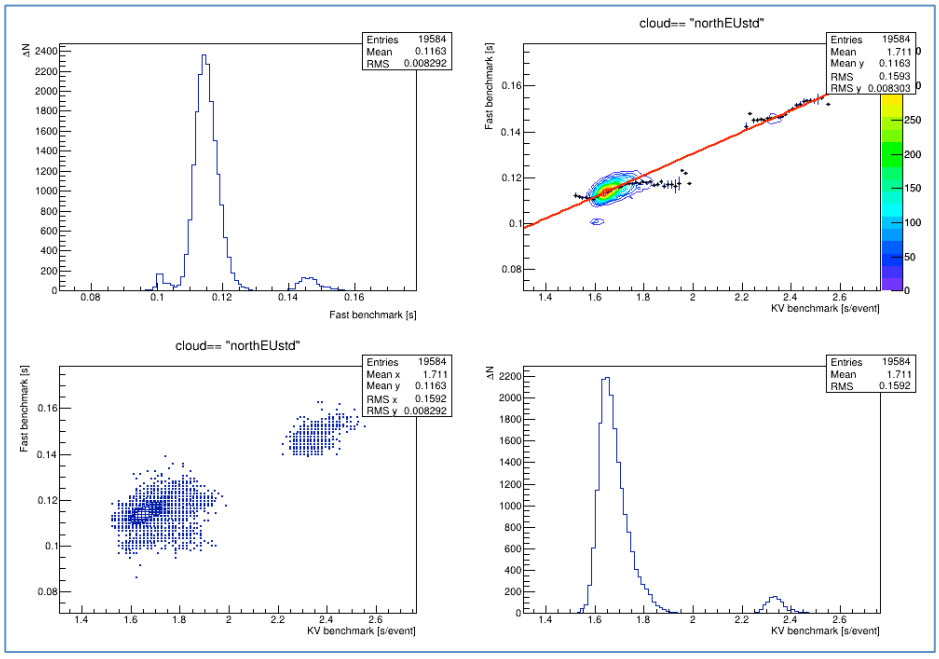

Figure 6. Comparison of two benchmark measurements (Fast benchmark and KV benchmark) performed iteratively in VMs of class Standard_A1 provisioned in the North Europe data centre. Scatter plot of the two variables is shown (bottom left) as well as the two projection on the x-axis (bottom right) and the y-axis (top left). A profile histogram with a linear fit is also shown (top right).

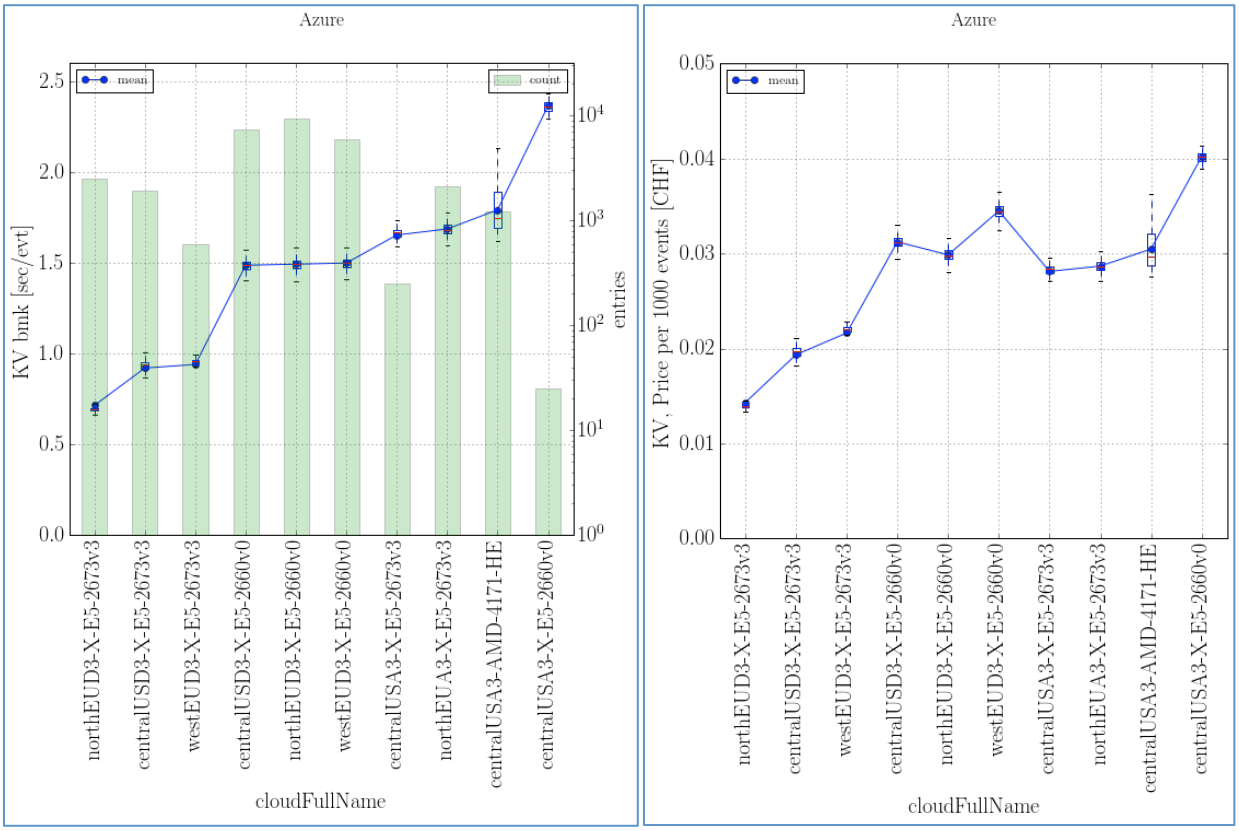

Figure 7. Comparison of VM performance on the basis of the KV benchmark measuring the cpu processing time per event. Benchmarked VMs are classified per data centre (North Europe, West Europe and Central US), VM series (D3 and A3) and CPU model (Intel Xeon E5-2673 v3, Intel Xeon E5-2660 v0, and AMD 4141-HE). The left plot shows the average performance of each VM class, ordered from the fastest to the slowest model (blue line), together with the 50% inter-quartile and the total value range as box plot. The amount of measurements collected for each class is superimposed (green histogram, right y-axis). The cost per event for the same VM classes is also reported (right plot).

A cost-to-benefit analysis of various VM offers has also been performed. For this purpose VMs of type Standard_D3 and Standard_A3 have been provisioned in the three available data centres, each VM consisting of 4 vCPUs. By provisioning of several hundreds VMs, it has been possible to probe different CPU models, namely Intel Xeon E5-2673 v3, Intel Xeon E5-2660 v0, and AMD 4141-HE. This was an observation during the test phase. According to Microsoft this was due to new hardware deployment for the new Dv2 series. The VM performance has been measured running the KV benchmark. Each KV test consisted in running four parallel instances of the benchmark application, in order to load the 4 vCPUs available in each VM. The measurements have been repeated multiple times in each VM in order to average fluctuations due to uncontrollable effects such as the resource sharing with other customers in the public cloud. Figure 7 (left) shows the average performance of each VM class, ordered from the fastest to the slowest model (blue line), together with the 50% inter-quartile and the total value range as box plot. The amount of measurements collected for each class is superimposed (green

histogram, right y-axis). The plot shows that the performance of D-series VMs depends on the CPU model, with the Intel Xeon CPU E5-2660 v0 @ 2.20GHz being 50% slower than the other probed model Intel Xeon CPU E5-2673 v3 @ 2.40GHz. A-series VMs follow in the performance ranking. In addition, the probed VMs have been compared on the cost/event basis, obtained by multiplying the measured CPU time/event times the VM cost per hour, as reported in the Azure catalogue[17]. The cost per event for each VM class is reported in Figure 7 (right). It shows that the cost per event of Standard_D3 VMs based on Intel Xeon CPU E5-2660 v0 is more expensive than the cost per event processed on Standard_A3 VMs, across the three data centres.

5.3 Experience in executing experiment workloads

5.3.1 ATLAS

The Workload Management System of the ATLAS experiment, PanDA [13], utilizes pilot-based approach to exploit the available resources. In this scale test pilots were provisioned by HTCondor [14]. The AutoPilot Factory submits a pilot wrapper to HTCondor master instance, which then submits a condor job with a pilot wrapper executable to an ATLAS distributed computing centre, where it is executed on a Worker Node (WN). The pilot on a WN contacts PanDA server with a request for a payload, which then executes. The job output is transferred from the WN to a permanent storage, in this case EOS [15] storage at CERN.

Each WN ran a HTCondor client that communicated with the HTCondor master instance. Network-wise, the VMs came with public IP address, however were situated in a private address space. ATLAS leveraged HTCondor Connection Broker (CCB) to successfully operate the available resources with minimal additional operational overhead. With HTCondor CCB, the WNs joined the pool of CERN resources. The relevant PanDA resource was configured in the ATLAS Grid Information System [16].

In the last 3 days of November 2015 ATLAS contributed to the scale test of the Azure IaaS, submitting simulation jobs, characterized by low disk I/O and high CPU time over wall clock time ratio. In total ~2 millions of events were processed, running for the equivalent of 206 k wall-clock hours. At the peak ATLAS benefited from up to 4.6 k cores simultaneously. The resources were stable, with a low rate of failure of ~3.2% of wall-clock time mainly caused by "lost heartbeat" errors between PanDA and the Azure VMs, during controlled VM termination for ramp-down of the cluster.

Figure 8 shows two monitoring plots of the job activity. The hourly number of finished jobs shows the successful and failed jobs, in green and red respectively. The two red peaks

correspond to the external termination of the VMs. On the right the cumulative number of processed events is reported.

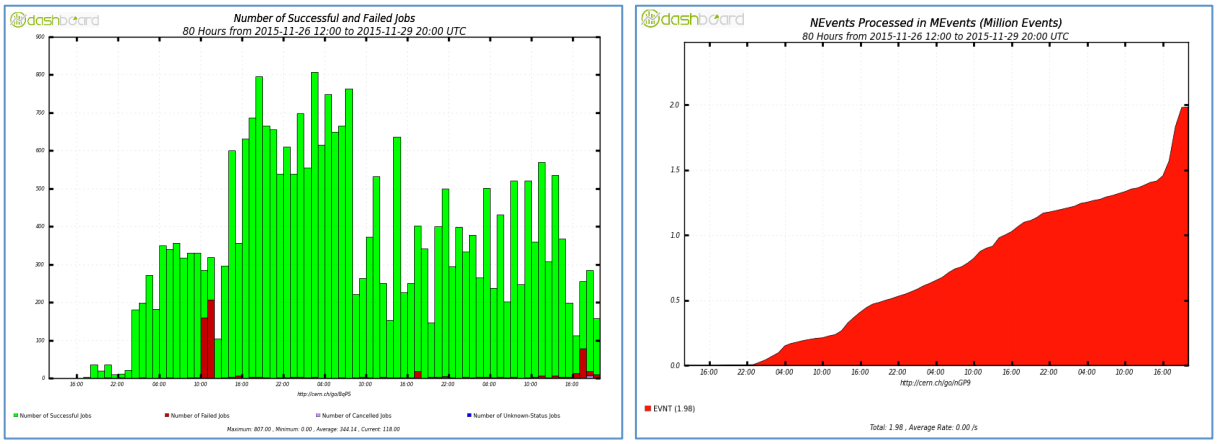

Figure 8. Number of ATLAS finished jobs per hour (left), where successful jobs are reported in green, failed jobs in red. The cumulative number of processed events during the scale test is also reported (right).

5.3.2 CMS

The CMS Workload Management system includes WMAgent [17] for central data production and processing and CRAB [18] for distributed data analysis. It relies on the glideinWMS [19], a pilot-based submission system built upon HTCondor [14]. The main elements of glideinWMS are factories for pilot submission to distributed sites and a glideinWMS frontend to request pilots following the need for resources in the underlying HTCondor pool. The HTCondor pool itself consists of the job queues and a central manager, which matches queued jobs and resources.

Within the third party VM factory model, VMs are provisioned independently from the job queue. The CMS pilot script aka Glidein is downloaded and executed after the contextualization of the VM, and then it retrieves and processes a job from the job queue. A Grid site in CMS provides computing and storage capacities to the experiment. Since the new cloud site hosted in Azure infrastructure is disk-less, it has been configured accordingly in the CMS information system so that the jobs' output is transferred to a persistent remote storage in the Grid.

The workflows supported by CRAB3 are end-user analysis and private Monte Carlo productions. Both have been integrated to run in Azure. The job status is monitored via CMS Dashboard Task monitoring [20] as any other CMS Grid job. Figure 9 shows the execution results of 100 private Monte Carlo production jobs submitted with CRAB3, which were all successfully executed.

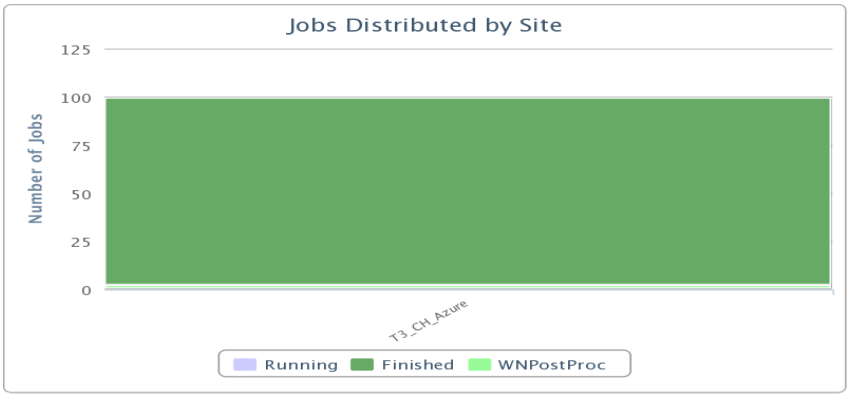

Figure 9. Execution results of 100 CMS Monte Carlo production jobs

5.3.3 LHCb

The Workload Management System of the LHCb experiment is based on the DIRAC [21] community Grid solution and on its LHCb-specific extension, LHCbDIRAC [22]. LHCbDIRAC uses a pilot-based approach to exploit the available resources. Each pilot contacts the LHCbDIRAC server with a request for a payload to be executed. The payload queues for all workflow types and the execution status of each payload can be monitored using the LHCbDIRAC web portal.

On resources where batch systems are installed, batch jobs are submitted with a pilot wrapper executable. In the case of the Azure IaaS, similarly to what is done on other cloud resources available to LHCb, a different approach was adopted, where a process responsible for spawning DIRAC pilots was immediately launched on each VM provisioned, during the contextualization stage. More precisely, on each of the four logical processors available on Azure IaaS VMs, a benchmarking process and a DIRAC pilot pulling single-processor payloads were executed one after the other. When a DIRAC pilot ended, because its payload completed execution or because no payload job was found in the queue, another benchmarking process was started, and so on in an endless loop.

LHCb used the Azure IaaS to execute simulation payloads, characterized by low disk I/O and a high ratio of CPU time over wall clock time. Every payload included the full chain of event generation, detector simulation and event reconstruction, each of these steps involving a different application executable. The output from each application step was transferred from the VM to a permanent storage, in this case EOS storage at CERN.

The largest scale test of the Azure IaaS by LHCb was performed during the last week of November 2015. At the peak, LHCb executed 1.3k single-processor simulation payloads simultaneously on as many cores. This is illustrated by Figure 10, which shows the rapid ramp-up of the number of LHCb simulation jobs as the number of provisioned VMs was increased.

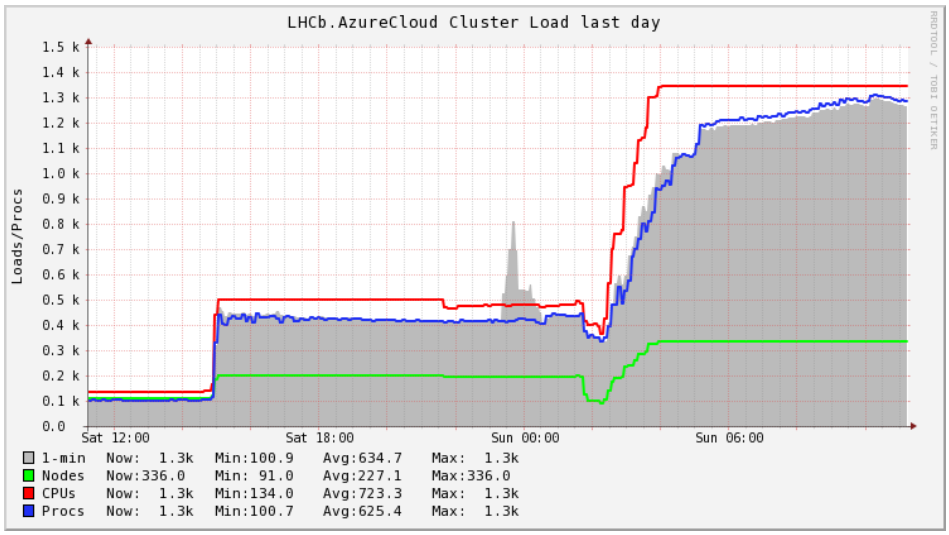

Figure 10. Ganglia monitoring plot for the LHCb scale test on 28-29 November 2015

6 Gap Analysis and Recommendations

In this section we summarize our findings on the usage of Azure resources and integration with the computing frameworks of the LHC experiments. The major showstoppers identified are reported, as well as the mitigation solutions and the forthcoming tools/improvements introduced by the Azure Solution Architects. Table 2 summarizes the major issues faced during the evaluation activity.

6.1 Procurement

6.1.1 Sponsored Subscription

Delays in the initial usage of a Sponsored Subscription have been caused by the need to check and agree on rules and conditions between Microsoft and CERN for sponsorships with Public Sector/Research customers and international organizations. Some of those delays have been caused by the need of introducing a Credit Card number in the subscription portal even for contracts based on an invoice. In our opinion, a prerequisite for the next phase is to tackle the contractual aspects and define a Microsoft – CERN agreement well in advance respect to the technical activity.

Date	Issue	Class	Time to solve
Mar. 23	Sponsored Subscription ready, but pending approval from MS Compliance department	Procurement	1 month
Apr. 20	Loan Agreement CERN – Microsoft for Sponsored Account to be reviewed and signed	Procurement	1 month
Jun. 9	Configuration of Account for Sponsored Subscription requires modification from Credit Card to Invoice method	Procurement	8 days
Aug. 3	Increase subscription default limit on number of cores	Configuration	10 days
Aug. 27	Increase subscription default limit on number of dynamic public IPv4 addresses: default 60, needed 1000 per data centre	Configuration	12 days
Sep. 21	Increase subscription limit on number of Network Interface Cards (NIC) per region per subscription: default 300, needed 1000 per data centre	Configuration	2 days
Sep. 24	Failures in VM provisioning seen systematically in deployments based on CernVM. Failures are ~30% of requests. The problem has been solved properly resizing the CernVM image.	Provisioning	1 month

Table 2. List of major issues faced, with time needed to obtain a working solution.

6.2 Configuration

6.2.1 Subscription limits

As a consequence of the Azure IaaS design described in Section 4.4, subscription limits are applied to essentially each of the multiple resources needed to build a VM (storage, network, IP addresses, etc). Some of those limits cannot be modified, such as the number of Cloud Services in the ASM model. This has been the main reason for the migration to the ARM model. In the ARM model, other limits hold, but most of them can be increased. An overall number of storage accounts accepted is 100 per subscription. Following the suggested practice of having 40 VMs per Storage Account, this brings to a maximum amount of 4000 VMs that can be provisioned

per subscription. Given that a Sponsored Subscription is in general bound to a single subscription, this fixes a limit to the capacity achievable per Sponsored Subscription.

6.2.2 Public IP addresses

As reported in Section 4.2 a public IPv4 address per VM is required. This requirement corresponds to a similar configuration adopted for the CERN resources. A similar request has been made in the recent price enquiries for the acquisition of commercial cloud resources.

In the context of the Azure evaluation, the limitation of public IPv4 addresses has represented a delay for the tests at a reasonably large scale. The initial limit of 60 IPv4 dynamic addresses per region per subscription has been exceptionally increased for a limited amount of time in each region to reach 1500 (both Central US and North Europe) and 1000 (West Europe) IPv4 addresses.

As suggested by the Solution Architects, this limitation will be solved in the future with the adoption of networking virtual gears to leverage VPN and relays/gateways. It has been highlighted that large public IP availability (even with IPv6) is not an option.

6.3 Capacity Management

The ARM model is the suggested approach for the acquisition of large capacity in Azure, through its template functionality that enables a faster VM provisioning. As highlighted by the Azure Solution Architects new features will be made available soon to allow parallel provisioning avoiding the creation of many resource groups. This new capability is named VM Scaleset and is in public preview since Nov-1st for compute base scenario.

6.4 Monitoring

Monitoring of resources and actions through the Azure web portals are very advanced and in line with expectations. Aggregations of metrics per VMs, summarized per Resource Group would be beneficial too. The Azure monitoring used in conjunction with the client-side monitoring tools, such as Ganglia, prove to be effective in monitoring the full VM lifecycle and in verifying that the delivered resources match with the accounted resources.

6.5 Accounting

The accounting report of used resources is available through the Azure portal[18]. The daily and hourly usage breakdown per computing resources (CPU, storage, and network) is available in csv format. The cost of the used resources was not available in the sponsored subscription and therefore, neither in the CSV nor in the summary billing history session of the portal (see Figure 11). In general prices are subject to contract and specific per volume. The Monthly Sponsorship Statement is received on the 18th of every month and covers the activity of the previous month.

[18] https://account.windowsazure.com/Subscriptions/billinghistory

This will change once we move from a sponsored subscription to an Enterprise Agreement contract.

As stated in the Azure Sponsorship Offer: *"The special pricing will terminate and your subscription(s) under the Microsoft Azure Sponsorship offer will be converted automatically to the Pay-As-You-Go offer upon the earlier occurrence of (1) when your total cumulative usage reaches the Usage Cap (specified above) at standard Pay-As-You-Go rates prior to application of any discount or (2) when you reach your End Date (specified above)."*

Given the large-scale tests CERN executes, in our opinion this lack of ongoing feedback on the remaining credit can represent a financial liability for CERN in case that the full credit is prematurely exhausted.

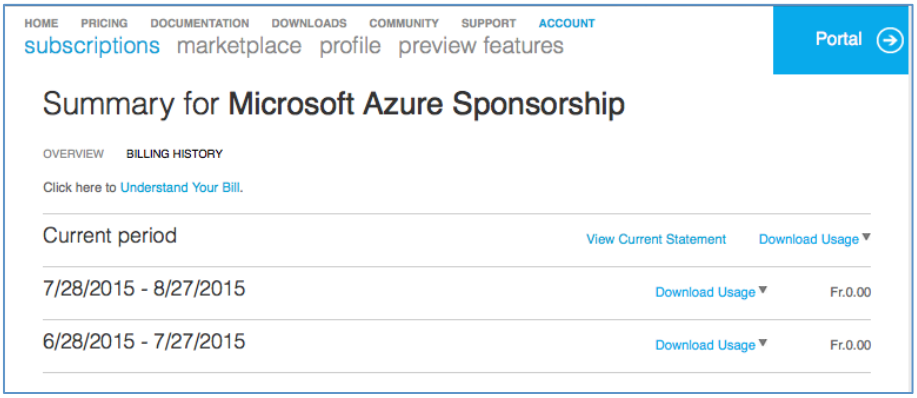

Figure 11. Snapshot of the reported billing history for the used resources.

6.6 Azure Batch

Azure Batch[19] introduces a concept well known in the workload management systems for HEP computing that is workload splitting and scheduling. Azure Batch includes the concept of jobs and tasks, job splitting, scheduling and data driven scaling-out. Figure 12 shows an example of parallel workload on Azure Batch as reported in the documentation[19]. Even if we consider this approach interesting to investigate, we consider its integration within the current workload management system of the experiments an additional indirection layer that will introduce more complexity than benefits. Each experiment already has adopted a system to schedule resources and split workloads in jobs. Their interplay with a similar system, Azure Batch, would be not trivial. The alternative of adopting Azure Batch to schedule resources and split workloads in jobs within each experiment will introduce more benefits.

[19] https://azure.microsoft.com/en-us/documentation/articles/batch-technical-overview/

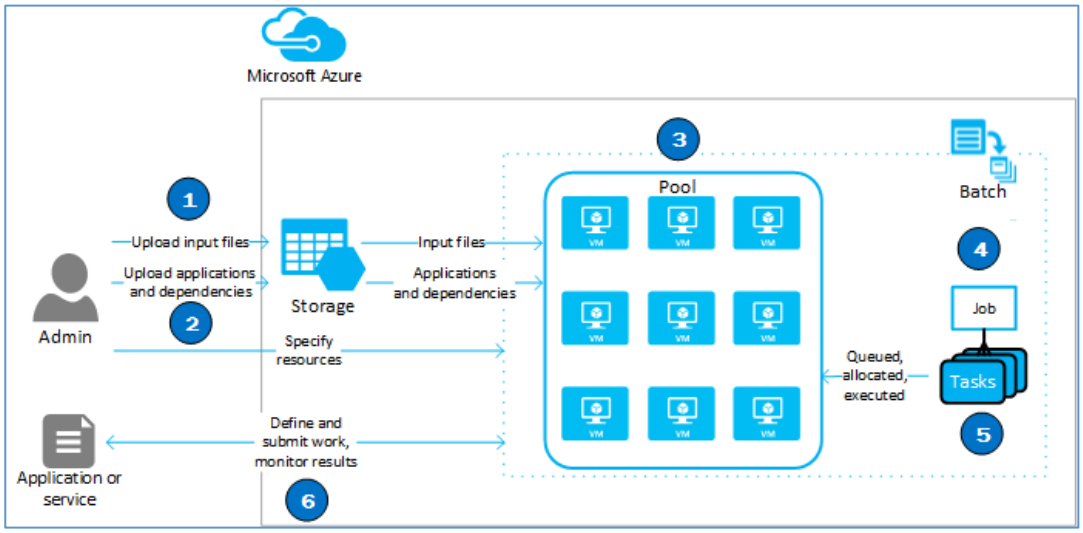

Figure 12. Scale out a parallel workload on Batch.

7 References

1 A. Di Meglio, CERN IT - Microsoft meeting minutes (restricted access)
2 CERN-IT SDC Cloud resource integration
https://twiki.cern.ch/twiki/bin/view/ITSDC/WLCGResourceIntegration
3 CernVM http://cernvm.cern.ch/portal/
Image download page http://cernvm.cern.ch/portal/downloads
4 https://www.centos.org/
5 P. Buncic et al., 2010 J. Phys.: Conf. Ser. 219 042003
6 F. Furano and A. Hanushevsky, 2010 J. Phys.: Conf. Ser. 219 072005
7 M. Massie et al 2012 Monitoring with Ganglia (O'Reilly Media)
8 A. J. Peters et al., 2014 J. Phys.: Conf. Ser. 331 052015
9 D. Giordano et al., 2015 J. Phys.: Conf. Ser. 664 022019
10 A. McNab et al., 2014 J. Phys.: Conf. Ser. 513 032065
11 C. Cordeiro et al., 2015 J. Phys.: Conf. Ser. 664 022013
12 De Salvo A and Brasolin, 2010 J. Phys.: Conf. Ser. 219 042037
13 T. Maeno et al., 2008 J. Phys.: Conf. Ser. 119 062036
14 E. Fajardo et al., 2015 J. Phys.: Conf. Ser. 664 022014
15 X. Espinal et al., 2014 J. Phys.: Conf. Ser. 513 042017
16 A. Anisenkov et al., 2012 J. Phys.: Conf. Ser. 396 032006
17 E. Fajardo et al., 2012, J. Phys.: Conf. Ser. 396 042018
18 M. Mascheroni et al., 2015 J. Phys.: Conf. Ser. 664 062038
19 J. Letts et al., 2015 J. Phys.: Conf. Ser. 664 062031
20 E. Karavakis et al., 2010 J. Phys.: Conf. Ser. 219 072038
21 A. Tsaregorodtsev et al., 2014 J. Phys.: Conf. Ser. 513 032096
22 F. Stagni et al., 2012 J. Phys.: Conf. Ser. 396 032104