ATLAS TDAQ System Administration: evolution and re-design

S Ballestrero[1], A Bogdanchikov[2], F Brasolin[3], C Contescu[4,5], S Dubrov[4], D Fazio[5], A Korol[2], C J Lee[1,5,8], D A Scannicchio[6], M S Twomey[7]

[1] University of Johannesburg, South Africa
[2] Budker Institute of Nuclear Physics, Russia
[3] Istituto Nazionale di Fisica Nucleare Sezione di Bologna, Italy
[4] Polytechnic University of Bucharest, Romania
[5] CERN, Switzerland
[6] University of California Irvine, United States of America
[7] University of Washington, United States of America

E-mail: atlas-tdaq-sysadmins@cern.ch

Abstract. The ATLAS Trigger and Data Acquisition system is responsible for the online processing of live data, streaming from the ATLAS experiment at the Large Hadron Collider at CERN. The online farm is composed of ∼3000 servers, processing the data read out from ∼100 million detector channels through multiple trigger levels. During the two years of the first Long Shutdown there has been a tremendous amount of work done by the ATLAS Trigger and Data Acquisition System Administrators, implementing numerous new software applications, upgrading the OS and the hardware, changing some design philosophies and exploiting the High-Level Trigger farm with different purposes. The OS version has been upgraded to SLC6; for the largest part of the farm, which is composed of net booted nodes, this required a completely new design of the net booting system. In parallel, the migration to Puppet of the Configuration Management systems has been completed for both net booted and local booted hosts; the Post-Boot Scripts system and Quattor have been consequently dismissed. Virtual Machine usage has been investigated and tested and many of the core servers are now running on Virtual Machines. Virtualisation has also been used to adapt the High-Level Trigger farm as a batch system, which has been used for running Monte Carlo production jobs that are mostly CPU and not I/O bound. Finally, monitoring the health and the status of ∼3000 machines in the experimental area is obviously of the utmost importance, so the obsolete Nagios v2 has been replaced with Icinga, complemented by Ganglia as a performance data provider. This paper serves for reporting of the actions taken by the Systems Administrators in order to improve and produce a system capable of performing for the next three years of ATLAS data taking.

ATL-DAQ-PROC-2015-010
15 May 2015

1. Introduction

After three years of Large Hadron Collider (LHC) beam (Run 1) and two years of upgrades performed during the Long Shutdown 1 (LS1), the LHC has officially restarted.

During the time of the LS1, a tremendous amount of work has been done by the ATLAS [1] Trigger and Data Acquisition (TDAQ) Systems Administrators (SysAdmins). Numerous new

[8] Primary Author

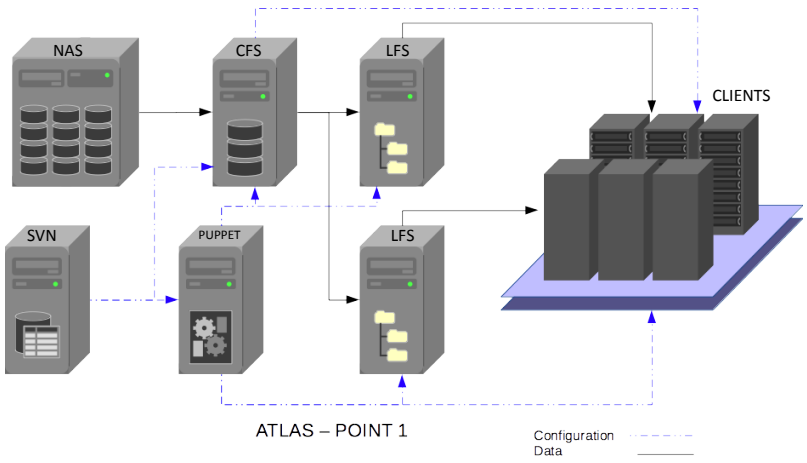

Figure 1. Simplistic layout of the Point 1 system.

software applications have been implemented, operating systems (OS) and most of the hardware has been upgraded, some design philosophies have been changed and the SysAdmins have exploited the High-Level Trigger (HLT) [2] computing farm to be used as a batch system to run specialised offline jobs.

At a first glance, the Point 1 (The name given to the area in which the ATLAS experiment is located at CERN) system [3, 4] appears to be fairly simple (Fig. 1). A Network Attached Storage (NAS) system holds most of the data, software, images, etc. required for the farm and sub-detector systems. This is exported to a Central File Server (CFS), from where the information is distributed (via rsync) to Local File Servers (LFS). There is one LFS installed per rack for the HLT farm, with a couple being shared between various sub-detector systems. These in turn export, via NFS, what is needed to the clients that each LFS serves.

For configuration management, other than the Configuration Database (ConfDB) [5], the entire system is maintained by code. This is all version controlled via Subversion, and checkouts of which are done on the Puppet [6] server and CFS. These two machines ensure the sane configuration of not only themselves, but all of the Linux machines in Point 1. There are however many more parts to the system (Fig. 2) and this paper will attempt to provide an overview and explain what was done during LS1.

2. Operating system upgrade

Scientific Linux CERN [7] (SLC) is the only officially supported Linux OS in use within ATLAS. It is a RedHat Enterprise Linux (RHEL) based OS, that is maintained and built by the CERN IT department. This has the benefit of being able to obtain direct support from on-site experts who are more "in-touch" with what is required for large scale particle physics experiments. During LS1, all systems were upgraded to SLC Version 6 (SLC6). Version 7, which is CentOS based and is known as CentOS CERN (CC7), has been released in 2014, however SLC6 is expected to remain the OS in use for all of Run 2.

There are a number of Windows OS based systems that are in use within Point 1. Most of these are used by the Detector Control Systems (DCS), and it should be noted that it is only required for one third-party application. During LS1, most of these systems were upgraded to SLC6 Virtual Hosts (VH) running Kernel-based Virtual Machine (KVM). Windows OS is installed as a Virtual Machine (VM) and controlled by the DCS group. The SysAdmins manage the host machine and can therefore use all of their tools and monitoring, which are Linux based,

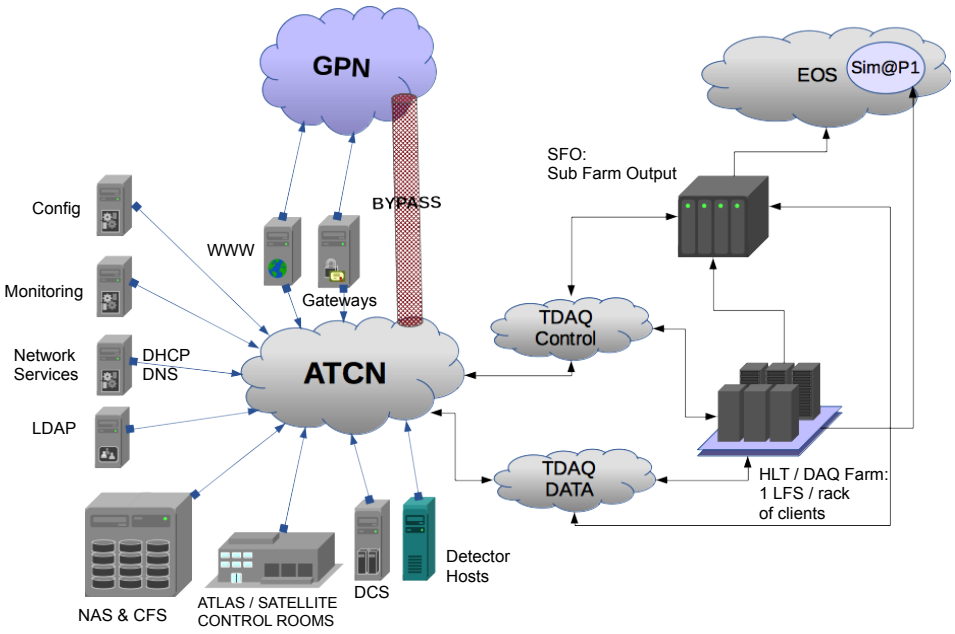

Figure 2. Realistic layout of the Point 1 system.

to ensure that the underlying hardware is performing correctly.

An important point to highlight is that no upgrades are applied to any systems while ATLAS is in data taking. During technical stop periods of the LHC, only critical security updates are applied and this means that the system will remain mostly unmodified until the Long Shutdown 2, expected around 2018.

3. Local boot versus net boot

Within Point 1 there are two different methods used to start the systems. The terms "local boot" and "net boot" are used to differentiate between the two. As of the writing of this paper, there are 664 local boot systems and 2392 net boot systems in Point 1 alone. The local boot systems are mostly servers used by DCS, Trigger and Data Acquisition, and the Systems Administrators, while the net boot systems are implemented on the HLT farm Read Out Systems (ROS), Single Board Computers (SBC), etc.

Essentially, local boot systems are standard Linux installations which boot from their local Hard Disk Drive (HDD). Provisioning is done via the Preboot eXecution Environment (PXE), and a template-based kickstart file that is provided to a new machine from the Puppet server. After the initial OS has been installed, Puppet is started as part of the first boot, and completes the configuration of the machine.

Net boot systems are booted via PXE, and in many cases do not have any HDD. The more components one has in a system, the greater the risk of failure. Hence, the more components one can do without, the more one can increase reliability. This is of particular interest for systems which are hardware bound and critical for ATLAS data taking.

Each reboot of a net boot machine can also be considered a fresh and cleanly installed OS. This has many advantages, in that it is easier to maintain for a small team, and reproducibility is increased on a large scale. It also greatly helps to reduce hardware replacement times. Of course there are some disadvantages as well, as a system like this requires ad-hoc development and support, is not suitable for running servers and is much less flexible, since an entire new

image needs to be built to make any non-trivial change to the system.

A completely new net boot system has been developed during LS1. It is based on the NFSroot [8] system and customised to suit the needs of the SysAdmins. Only special read-write areas are kept in Random Access Memory (RAM). These include directories such as `/etc` and `/var`. For these areas, bind mounts are overlaid on the read only NFS mounts of `/` from the LFS serving the client. Images for deployment are now created by Puppet in a chroot environment and there is no need for a "golden image" [4]. Images can now always be rebuilt from versioned configuration. This frees up more RAM for the host than was possible with the old net boot system used with SLC 4 and 5.

For specialised old hardware that is unable to PXE boot, an ELF (Executable and Linkable Format) image is provided, using a private patch of the `mknbi` [9] package. These systems also require a special 32 bit non-PAE kernel, which is not supported by RedHat and is provided on a best effort basis from both CERN IT and the Sysadmin team.

4. Configuration management systems

As the number of machines in the computing farm increases, it becomes a daunting task to ensure that each and every machine is configured correctly, especially if there are numerous different configurations being deployed throughout. Configuration Management Systems (CMS) have become the only sane way in which farms of this nature can be managed. Quattor [10], which was originally developed in the framework of the European Data Grid project, was dismissed as the CMS of choice as it was obsolescent, and Puppet was chosen [11] as its replacement over Chef [12] and CFEngine [13] which were available at the time. This was partially due to existing experience with Puppet by members of the SysAdmin team and the deployment of Puppet started in 2011. Since then CERN IT has also adopted Puppet as its CMS of choice and many Worldwide LHC Computing Grid (WLCG) [14] applications are also being puppetised.

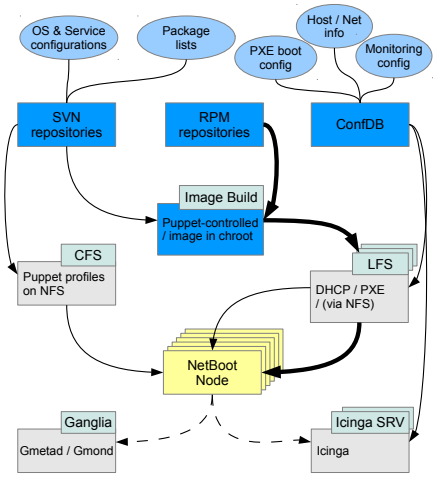

Figure 3. Net booted system schematic.

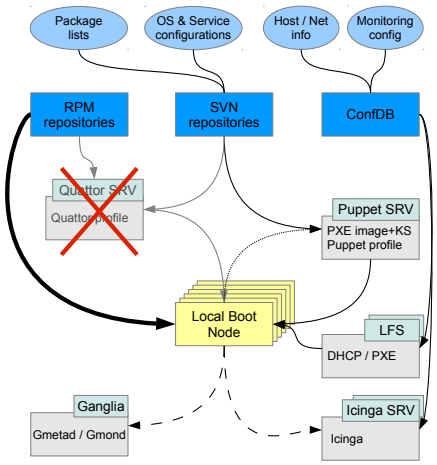

Figure 4. Local booted system schematic.

As can be seen in Fig. 3 and Fig. 4, Puppet is been incorporated into both the net and local boot systems. This now means that the configuration procedure between them is very similar making it considerably easier to maintain, as code can be shared and is reusable between both systems.

For local booted systems, the configuration is ensured every 30 minutes by the local Puppet daemon, which uses the Puppet server as the central information repository. Instead, on the

net booted systems an hourly cron job has been deployed that runs at a randomised time which uses the `puppet apply` functionality. For these, the configuration information is provided from the CFS via an NFS mount, to avoid overloading of the Puppet VM. This NFS-based arrangement has better scalability than the classic one based on a Puppet server. One of the biggest advantages of this new system is that there no longer the need to reboot a net booted machine in order to apply a new configuration, which was previously the case using the Boot-With-Me scripts [11].

5. Configuration database

The only part of the system which is not code, is the configuration database [5]. This is a MySQL database, which contains numerous different settings for different machines. This includes DHCP details of a machine, its operational status, what should be monitored by Icinga, boot options, and other parameters which are not part of the static configuration defined in Puppet. Interaction with this database is done using ConfDB, an in-house application composed of a PHP based web user interface, Python based back-end utilities and a REST API.

The ConfDB UI greatly helps to ease cluster management by providing tools that allow us to issue ssh and ipmi commands to multiple machines. It also provides a means to pull information from CERN IT's LAN Database (LanDB) [15] to ensure that network configurations are always up to date.

During LS1, many new features were added, including an OS versioning system that allows us to configure different OS versions to be run on different machines. This is very useful for testing of a new OS, or rolling back quickly in case of problems. As part of the REST API, a new network interface configuration system (including bonding, bridges and VLAN's) was also included, allowing it to interact with Puppet and other tools to ensure that network settings are applied correctly based on information from CERN IT as well as a feature to configure which operational state the machine should be in, e.g. Maintenance, TDAQ Mode, Sim@P1 (discussed in section. 7), etc.

6. Virtualisation

The last years have seen a general trend towards virtualisation and cloud technologies, as a means to contain the server sprawl, consolidate and better exploit hardware resources, and provide more flexibility and resilience.

In ATLAS, while the SysAdmins have started to virtualise many systems during LS1 and started to explore the use of more virtual machines, this has only been used for the core infrastructure and test machines, and not the DAQ or HLT systems. This is therefore no cloud like approach with shared storage. The SysAdmins have prioritised simplicity, and rely on the multiplicity of lower cost systems instead of a single redundant system.

For the core infrastructure and test servers, currently there are around 50 VM's being used in Point 1, and another 50 in the General Public Network (GPN). Notable VM's currently in use within ATLAS are: gateways, domain controllers, DCS windows services, development web servers, Icinga, Puppet, LDAP, and public user nodes. In addition to these, there are around 100 VM's in use by DCS. A completely different use case for virtualisation is that of Sim@P1.

One case of note is how virtualisation helped us to address the scalability limitations of the OpenLDAP service daemon. The four physical LDAP servers used during Run 1 have been replaced by two servers running a total of eight VM's, providing much higher workload capability to support the increased requirements for the Role Based Access Control system [16].

The adoption of virtualisation had also been considered for at least certain parts of the TDAQ infrastructure, but the native application migration capability of the TDAQ Controls & Configuration system already provides cloud-like flexibility and resilience, unlike more common applications which need an OS virtualisation layer to attain these features.

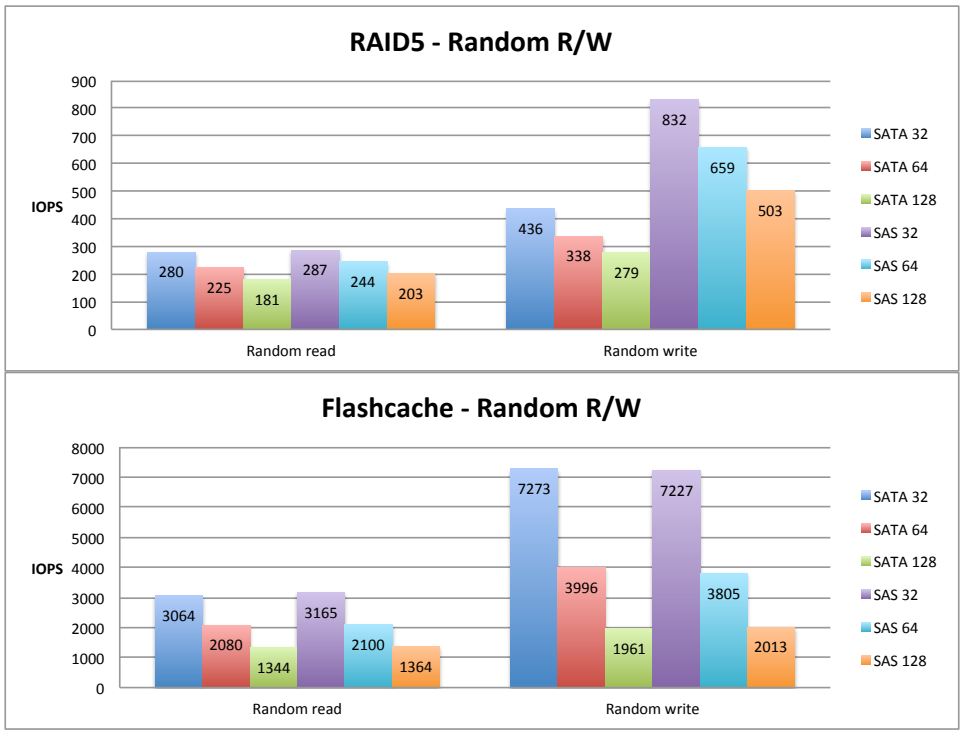

Figure 5. FlashCache performance comparison with normal non-cached disk.

The opportunity was taken during this time to evaluate new hardware that will be available for use as VH's, as well as new software applications. The results of some of the disk input/output (I/O) operations performance tests done using FlashCache can be seen in Fig. 5. This clearly demonstrates the possibilities presented by the use of caching applications to increase I/O performance on systems, while lowering costs.

7. Simulation at Point 1

Simulation at Point 1 (Sim@P1) [17] is the opportunistic use of the existing HLT farm as a batch farm, allowing non-utilised resources to be exploited for ATLAS production jobs. Under the control of an OpenStack [18] cloud management layer, VM's are created on top of the HLT machines, and act as computing nodes interconnected through a virtual LAN (VLAN) on the data network. Through the use of the VM's and VLAN, the SysAdmins are able to completely isolate the Sim@P1 system from the ATLAS Control Network (ATCN), ensuring that security is maintained. Communication to GPN is via a logically separated link to CERN IT, using ACL's to ensure that traffic is only directed to the CERN services (Castor/EOS, Condor, etc).

More than 1300 physical nodes of the HLT farm, running up to ∼2700 VM's are currently available to run Monte Carlo (MC), high CPU intensive jobs. During LS1 more than 1.7 billion MC events were produced. Availability of nodes was mostly limited by cooling and power maintenance and upgrades.

Switching between Sim@P1 and TDAQ mode is controlled from ATLAS control room. More information on the evolution of Sim@P1 can be found in a dedicated paper [19].

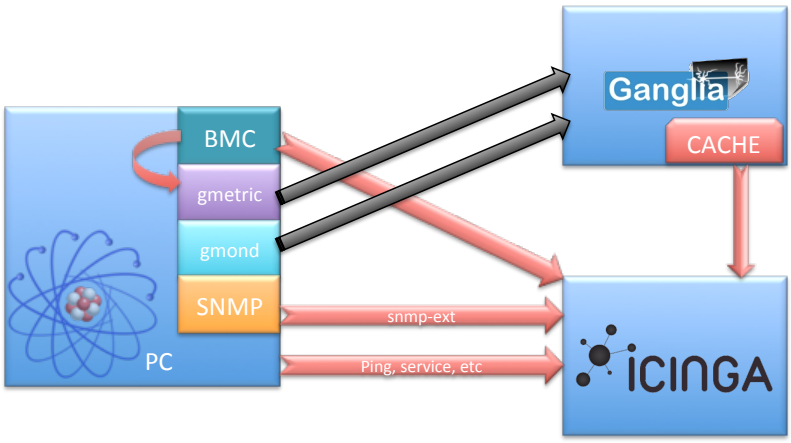

Figure 6. Layout of the monitoring.

8. Monitoring and system logs

Monitoring of any computing farm is a critical part of its operation. It is the first line of defence, letting one know not only when things have failed, but more importantly to give alerts of possible failures before they happen. During LS1 a substantial amount of work was done to migrate away from Nagios towards an Icinga based monitoring system [20]. This is complemented by Ganglia, which is used as collector for performance and health metrics and is highly scalable with the use of rrdcache (a daemon that receives updates to existing round-robin database files). Ganglia-web also provides an advanced user interface, through which historical data from the rrd files can be investigated. Icinga not only provides these critically important alerts and checks, but is able to use Ganglia data for performance alerts. The fact that Icinga was able to use existing Nagios plugins and much of the existing configurations made the migration much easier.

Hardware monitoring is done via IPMI, and a complete re-write of the existing system was started during LS1 and is still ongoing. The new system is based on GNU FreeIPMI [21] instead of IPMITool. This has provided us with much better performance with Sensor Data Record caching and the use of unique sensor IDs versus unstable sensor names.

Icinga, Ganglia and IPMI are all used together to provide a much more efficient system (Fig. 6). Local readout data from IPMI is fed to Ganglia, and the SEL and sensor states are then pulled by Icinga to provide alerting, as well as other information directly from the machine.

A single VM is used to host Icinga for the core systems, which monitors around 570 nodes and performs ∼5000 checks. One physical machine is then used for the rest of the farm, which includes ∼2200 nodes and performs ∼31000 checks. These systems both help to provide not only the SysAdmins, but also the users, with status notifications of problematic machines. Currently testing with Icinga v2 is ongoing for distributed scheduling and to evaluate the configuration and performance thereof.

With the upgrade to SLC6, rsyslog has replaced syslogd on all machines and is also used as the active collector for remote logging, replacing syslog-ng. According to their Puppet configuration, clients are able to submit their logs remotely to either an LFS or another central collector. It is also possible to send remote logs to CERN IT for additional security checks, to assist in spotting any potential breaches. All net booted systems retain local logs for two days and send remote logs to their LFS which are held for 30 days before they are rotated. Local boot hosts retain logs for 30 days and all exposed system logs are kept for 12 weeks.

Evaluating the use of a central log collection and analysis tools such as Splunk, ELSA, logstash

and elastic search is still on going.

9. Conclusion

Overall LS1 was a very productive time for the team. Many tasks are now completely automated and require very little human intervention, which makes jobs much easier.

The transition to SLC6 was generally smooth and provided us with a more stable OS. Puppet is easy to maintain. It has been adopted for both local and net boot hosts, and provides a single, clean and simple environment to ensure the sane state of machines, as well as a platform that allows to easily train new SysAdmins in managing the system.

The usage of virtualisation and cloud technologies allowed ATLAS to improve resource usage, resilience and reliability for various services, and to exploit the idle time of HLT computing resources for other tasks.

The new monitoring provides better user interfaces and many more checks than it did before, and it is still rapidly evolving, especially as more systems are added and new checks are required by different hardware.

During this time the system has been streamlined, and an improved environment has been provided that will be easier to maintain and manage for future SysAdmins.

References

[1] ATLAS Collaboration 2008 *The ATLAS Experiment at the CERN Large Hadron Collider JINST* **3** S08003

[2] ATLAS Collaboration 2003 *ATLAS high-level trigger, data-acquisition and controls: Technical Design Report* Technical Design Report ATLAS (Geneva: CERN) URL `http://cds.cern.ch/record/616089`

[3] Adeel-ur-Rehman A, et al 2010 *System administration of ATLAS TDAQ computing environment J.Phys.Conf.Ser.* **219** 022048

[4] Lee C J, et al 2013 *ATLAS TDAQ System Administration: an overview and evolution PoS* **ISGC2013** 006

[5] Ballestrero S, et al 2012 *Centralized configuration system for a large scale farm of network booted computers J.Phys.Conf.Ser.* **396** 042004

[6] PuppetLabs `https://puppetlabs.com/puppet/what-is-puppet`

[7] Scientific Linux Cern `http://linux.web.cern.ch/linux/scientific.shtml`

[8] NFSroot `http://www.tldp.org/HOWTO/NFS-Root.html`

[9] mknbi `http://etherboot.sourceforge.net/doc/html/mknbi.html`

[10] Quattor Overview `http://www.quattor.org/documentation/2012/06/19/documentation-overview.html`

[11] Ballestrero S, et al 2012 *Upgrade and integration of the configuration and monitoring tools for the ATLAS Online farm J.Phys.Conf.Ser.* **396** 042005

[12] Chef `http://www.opscode.com/chef/`

[13] CFEngine `http://cfengine.com/product/what-is-cfengine/`

[14] Worldwide LHC Computing GRID `http://wlcg-public.web.cern.ch/about` accessed: 2015-04-16

[15] Andrade P, et al 2012 *Review of CERN Data Centre Infrastructure J.Phys.Conf.Ser.* **396** 042002

[16] Valsan M L, et al 2011 *Role based access control system in the ATLAS experiment J.Phys.Conf.Ser.* **331** 022042

[17] Ballestrero S, et al 2014 *Design and performance of the virtualization platform for offline computing on the ATLAS TDAQ Farm J.Phys.Conf.Ser.* **513** 032011

[18] OpenStack `http://opensource.com/resources/what-is-openstack`

[19] Ballestrero S, et al 2015 *Design, Results, Evolution and Status of the ATLAS Simulation at Point1 Project. Proceedings of the CHEP2015 conference. J.Phys.Conf.Ser.*

[20] Ballestrero S, et al 2012 *Tools and strategies to monitor the ATLAS online computing farm J.Phys.Conf.Ser.* **396** 042053

[21] GNU FreePIMI `http://www.gnu.org/software/freeipmi/`