The Next Generation ARC Middleware and ATLAS Computing Model

To cite this article: Andrej Filipi et al 2012 J. Phys.: Conf. Ser. 396 032039

View the article online for updates and enhancements.

Related content
- arcControlTower: the System for Atlas Production and Analysis on ARC
  Andrej Filipi and the Atlas collaboration
- ARC and gLite interoperability in ATLAS sites
  Andrej Filipcic, Szymon Gadomski, Sigve Haug et al.
- ATLAS Distributed Computing Monitoring tools after full 2 years of LHC data taking
  Jaroslava Schovancová

Recent citations
- Towards a Production Volunteer Computing Infrastructure for HEP
  N Høimyr et al
The Next Generation ARC Middleware and ATLAS Computing Model

Andrej Filipčič¹, David Cameron², Oxana Smirnova³, Aleksandr Konstantinov², Dmytro Karpenko² for the ATLAS Collaboration

¹ Jozef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia
² University of Oslo, P.b. 1048 Blindern, N-0316 Oslo, Norway
³ Lund University, P.O. Box 118, SE 22100 Lund, Sweden

E-mail: andrej.filipcic@ijs.si

Abstract.

The distributed NDGF Tier-1 and associated NorduGrid clusters are well integrated into the ATLAS computing environment but follow a slightly different paradigm than other ATLAS resources. The current paradigm does not divide the sites as in the commonly used hierarchical model, but rather treats them as a single storage endpoint and a pool of distributed computing nodes. The next generation ARC middleware with its several new technologies provides new possibilities in development of the ATLAS computing model, such as pilot jobs with pre-cached input files, automatic job migration between the sites, integration of remote sites without connected storage elements, and automatic brokering for jobs with non-standard resource requirements. ARC’s data transfer model provides an automatic way for the computing sites to participate in ATLAS’ global task management system without requiring centralised brokering or data transfer services. The powerful API combined with Python and Java bindings can easily be used to build new services for job control and data transfer. Integration of the ARC core into the EMI middleware provides a natural way to implement the new services using the ARC components.

1. ATLAS Computing Model

An enormous demand for computing resources needed by LHC experiments to process huge amounts of data led to the establishment of the world-wide LHC Computing Grid (wLCG), which coordinates the development of software and infrastructure through several international Grid-related projects. Grid middleware provides a backbone to access computing resources in a more or less homogeneous way, but it does not offer a complete solution for job management and data organization for the complex demands of big experiments. On top of a core Grid layer, provided by different middleware flavors (Lightweight Middleware for Grid Computing - gLite [1], Open Science Grid - OSG [2] and Advanced Resource Connector - ARC [3]), the ATLAS experiment [4] has developed a comprehensive infrastructure to access computing resources in a transparent and efficient way.

More than 150 sites participate in ATLAS computing, providing 120,000 computing cores and 80 PB of disk and tape storage. Although many of these sites are interconnected with 10 Gb/s network links, this is not sufficient for commodity computing. In early 2012, ATLAS ran on average 700,000 jobs (figure 1) daily, processing several PB of data. This would not have
been possible without well-planned data distribution to computing sites and efficient computing task brokering where jobs are processed on sites with input datasets pre-placed. PanDA [5], the ATLAS management and distribution system for production and analysis jobs, distributes the workload to the sites according to resource and data availability, as provided by the ATLAS data management system (DQ2).

A strict hierarchical model of data distribution and task processing is however not sufficient any more for either central production or user analysis, due to frequent changes in data access patterns or high-priority tasks. ATLAS has thus developed several methods to enhance job throughput, such as PandDA Dynamic Data Placement (PD2P) [6] for user analysis, which triggers on-demand data replication of popular data and site reallocation, and multi-cloud production where jobs can run on sites with no pre-placed input datasets.

The ATLAS computing model is evolving from a strict hierarchical centrally controlled system to a dynamic model for faster task throughput.

2. ARC and ATLAS ND cloud
ATLAS Tier-2 and Tier-3 sites are associated to Tier-1 sites, an entity called an ATLAS cloud, for data distribution and computing task brokering. An ATLAS site registers its computing element (CE) and storage endpoints (SE) to the ATLAS Grid Information System (AGIS) [7].

The Nordugrid (ND) cloud, comprising the Nordic DataGrid Facility NDGF-T1 center with its associated Tier-2 and Tier-3 sites, is slightly different from the other 9 ATLAS clouds. As seen by ATLAS it has a single storage endpoint, NDGF-T1, but the data itself is distributed over 4 countries among 9 distinct sites. The distributed dCache [8] has central services at the NDGF-T1 central site while the distributed storage pools provide direct access for the data transfers through gridftp v2, gsidcap or xrootd protocols. Although there is a single point of failure at the central services, past experience has shown it to be extremely stable, with unavailability due

![Completed jobs](image)

Figure 1. Daily ATLAS jobs for the first 4 months of 2012.
to downtimes of less than 1%. Since the distributed storage pools have much higher aggregated
downtime, due to several distinct sites being involved, the NDGF-T1 has 20% of its resources
dedicated to recent-file replication. A new file triggers immediate replication to a dislocated
cache pool. The oldest file replicas are cleaned when the cache partition is full, and the typical
lifetime of replicas is about one month. This mode of operation guarantees file availability for
most of the files even in case of longer pool-site downtimes.

The 13 ARC computing elements (ARC-CE) in 5 countries have no local storage endpoint.
Most of the sites are interconnected with a 10 Gb/s network. A snapshot of the ATLAS monitor
is shown in figure 2. The pilot job model as imposed by PanDA is not an optimal way to use
the worker nodes on ARC clusters, since the worker nodes would transfer the job inputs from
the remote storage. Therefore, the arcControlTower [9] has been developed, which picks the pilot
workload from the PanDA server and submits a fully-defined job to one of the clusters. There
are several benefits to this mixed approach. All the job transfers are performed by the ARC-CE,
so the worker node spends the time on computing only. The job input files are cached in the
ARC-CE cache, so subsequent jobs can re-use them to avoid input re-transfers. The PanDA
server does not assign the jobs to a particular ARC cluster, but arcControlTower does according
to the cluster load and ATLAS software release availability.

In contrast to gLite or OSG, ARC also provides strict job-resource control, such as CPU and
wall-time consumption, per-process or per-job resident and virtual memory limits, multi-core
and multi-node allocation and a job retry mechanism in case of service failures. As opposed
to gLite, where jobs with custom requirements need to be handled through PanDA queue and
CREAM-CE site reconfiguration, ARC with arcControlTower can handle custom jobs on-the-fly.

3. ARC-CE Architecture
The core of the new ARC-CE architecture is composed of three basic services for job submission,
job information and data transfer, as shown in figure 3. All three services provide new web-

service oriented access interfaces (WS), as well as legacy interfaces (pre-WS). ATLAS is still
using the legacy interfaces, but migration to the new services is foreseen for later in 2012. A-
REX, the execution service of ARC-CE, has nevertheless replaced the older Grid Manager on
all ATLAS ARC sites, due to performance improvements and seamless integration within the
existing infrastructure.

The new ARC client (figure 4) has been enhanced to support three different job description
languages, namely XRSL for legacy ARC, JDL for gLite, as well as the new standard JSDL [10]. With the imminent adoption of the EMI Execution Service (EMI-ES) across several middleware flavors, the ARC client will be able to submit jobs to all the three EMI-supported computing elements.

A notable feature of ARC-CE is its portability — it runs on many different flavors of Linux, as well as on Mac OS X and Windows (client only). In addition, it does not require any middleware installation on the worker nodes, although in many sites the nodes have access to client tools to be compatible with gLite.

4. New Features of ARC

The current ARC production version used by ATLAS sites is 11.05 and it provides legacy services for ATLAS job submission through gridftp and an LDAP-based information system. ARC version 12.05 is released through the EMI-2 distribution and is used on one site as pre-release in production. Numerous new features of ARC 12.05 provide much better job control and faster job and transfer throughput.

The typical workflow of an ATLAS job on a NorduGrid cluster is

- PanDA job submission by arcControlTower to one of the ARC clusters
- job preparation on ARC-CE and downloading the input files from NDGF-T1 storage to ARC cache (if not already cached)
- job execution by a batch queueing system, arcControlTower monitoring and sending updates on the job status to PanDA
- job post-processing on ARC-CE and transferring the output files to the destination storage endpoint
- finalizing the job by arcControlTower and the output file registration in DQ2

One of the major new features of ARC 12.05 is a new data staging mechanism [11]. It was developed upon an ATLAS request to optimize and maximize the transfers of the job input and output files. The total size of job input files can be several 10GB and with many short jobs, the ARC-CE in legacy transfer mode can reach its job processing limits. The distributed transfer mechanism implements the following:

- A-REX master controls the transfer requests and registers the input files to cache. To speed the transfer preparation, the master resolves the input files in advance and pins them on SRM.
- several data-delivery slaves are configured with local cache disk and receive transfer requests from the A-REX master.
• the master distributes the transfer requests to slaves. The number of transfer slots is kept constant. The input files common to many jobs are handled at once by a single transfer.

• when all the input files are transferred, the master submits a batch job.

Other prominent new features include amongst others:

• web-service job submission interface with much lower latency than legacy gridftp interface
• full support for the standard GLUE2 information schema [12]
• support for the Argus authorization interface [13]
• a SWIG-generated Python library providing comprehensive access to every part of the ARC including low level objects
• Hosting Environment Daemon (HED) [14], the container for all ARC services, which enables building and setting up Grid-related services in a transparent way.
• a powerful API with Python and Java interface to build customized services or client-side systems for complex job workflows
• an A-REX plugin interface to enable custom job processing add-ons at various job state transitions.

5. Advantages of Composite Model
The ATLAS computing model pushes the jobs to sites with input data available on local storage. Even with a new dynamic model, the same paradigm is still valid. The job-oriented dataset transfers are controlled by PanDA, so the central data management system must be able to cope with future increases in transfer activity.

An ARC cluster is able to participate in ATLAS production and analysis without a local storage endpoint. Even if all the data transfers use remote storage in this case, the ARC-CE takes care of controlling and throttling the transfers. The CPU cycles on the nodes are not wasted waiting for the transfers to complete. Since the transfers are handled by ARC-CE on-the-fly, the ATLAS central services are relieved of transfers handling.

ARC supports most of the commonly used data transfer protocols such as LFC, SRM, gridftp, HTTP, HTTPs, FTP and xrootd. In addition, several replicas can be used for inputs with a preferred source replica resolved by site-provided configuration option to provide a fail-over mechanism. For output uploads, a similar fail-over mode can be used. Several storage endpoints can be listed as the output file destinations, so a failure of the primary storage does not lead to a job failure.

The main advantage of ARC for ATLAS is using the cache which reduces the total amount of data transfer by a factor of 2. Typically, the lifetime of an input file cached is two weeks on clusters with 100 GB cache space per core. For big sites, the production and analysis performance is similar for gLite and ARC setups. Smaller gLite sites, especially Tier-3s, typically suffer from insufficient local storage capacity and are not used very efficiently. In the ND cloud, small ARC-CE sites are used to their full extent and rarely experience job failures.

The ATLAS central services experience an increased load with extensions to the computing model. In order to relieve the stress, dynamic services such as the automated on-demand system of the ARC distributed transfer model can play a significant role. The ARC-CE also manages its own cache space, eliminating the need for management by central services.

In order to integrate ARC into the ATLAS computing environment even further, the following activities have been initiated:

• enable full pilot jobs on ARC cluster. The request for data transfers on the nodes will be done dynamically but performed by the ARC-CE staging mechanism. The functionality of the cache and transfer control will be fully preserved.
• enable full support for ARC clusters in the AGIS and the pilot job submission system
• migrate the ATLAS ARC interfaces to the new ARC client library

6. Conclusions
The ARC middleware has proved to be a stable and efficient job-processing solution for distributed sites outside the traditional hierarchical model. In addition, it provides a solution for sites with limited disk resources to participate efficiently in ATLAS production.

The extensions and enhancements to the central ATLAS production systems are limited by the current gLite architecture, such as unimplemented job memory limits. ARC can provide new approaches to the job submission, brokering and data distribution scheme.

References