AGIS: The ATLAS Grid Information System

AGIS: The ATLAS Grid Information System

A Anisenkov[1], A Di Girolamo[2], A Klimentov[3], D Oleynik[4] and A Petrosyan[4] on behalf of the ATLAS Collaboration

[1] Budker Institute of Nuclear Physics, Novosibirsk, Russia

[2] CERN, ES Department, CH-1211 Geneva 23, Switzerland

[3] Brookhaven National Laboratory, Upton, NY 11973, USA

[4] Joint Institute of Nuclear Research, Laboratory of Information Technology, Dubna, Russia

Email: Alexey.Anisenkov@cern.ch

Abstract. ATLAS, a particle physics experiment at the Large Hadron Collider at CERN, produced petabytes of data annually through simulation production and tens of petabytes of data per year from the detector itself. The ATLAS computing model embraces the Grid paradigm and a high degree of decentralization and computing resources able to meet ATLAS requirements of petabytes scale data operations. In this paper we describe the ATLAS Grid Information System (AGIS), designed to integrate configuration and status information about resources, services and topology of the computing infrastructure used by the ATLAS Distributed Computing applications and services.

1. Introduction

The ATLAS experiment [1] produces petabytes of data per year that needs to be distributed globally to sites worldwide according to the ATLAS computing model [2], which is based on a worldwide Grid computing infrastructure that uses a set of hierarchical tiers. It provides all members of the ATLAS Collaboration speedy access to all reconstructed data for analysis, and appropriate access to raw data for calibration and alignment activities.

The variety of the ATLAS Computing Infrastructure motivated the development of a central information system to store the different parameters and configuration data which are needed by the many ATLAS software components. A centralized information system helps to solve the inconsistencies between configuration data stored in the various ATLAS services and in the different external databases. AGIS is the information system developed to integrate various configuration parameters required to configure and to operate the ATLAS Distributed Computing (ADC) [3,4] systems and services. The information system centrally defines and exposes the topology of the ATLAS computing infrastructure including various static, dynamic and configuration parameters collected from independent sources like gLite BDII (Berkeley Database Information Index), Grid Operations Centre Database (GOCDB) [5], the Open Science Grid Information services (MyOSG, OSG Information Management System - OIM) [6].

Today AGIS is the primary source of information for all the ADC applications including the ATLAS Distributed Data Management (DDM) [7] system and the ATLAS Production and Distributed Analysis (PanDA) [8] workload management system.

2. Information and data sources

The ADC applications and services require the diversity of common information, configurations, parameters and quasi-static data originally stored in different sources. Sometimes such static data are simply hardcoded in application programs or spread over different configuration files.

The difficulty faced by the ADC applications is that ATLAS computing uses a variety of Grid infrastructures (**E**uropean **G**rid **I**nfrastructure (EGI) [9], **O**pen **S**cience **G**rid (OSG) [10] and NorduGrid, ARC-based infrastructure [11]) which have different information services, application interfaces, communication systems and policies. Therefore, each application has to know about the proper information source, data structure formats and application interfaces, security infrastructures and other low-level technical details to retrieve specific data. Moreover, an application has to implement communication logic to retrieve data from external sources that produce a lot of code duplications.

Another point of difficulty in the data organization are the missing links between experiment-specific consumed resources and physical Grid capabilities: the data structures and hierarchy of resources defined in external information services do not fit well ATLAS computing relations. For example, ATLAS can define a few sub-sites behind one physical Grid resource. The primary goal of AGIS is to facilitate, enable and define those missing relationships between computing services *provided by* various sites and the services *used by* the experiment. By providing an abstraction layer from the physical resources, AGIS allows the experiment to define their own real organization of the resources. The system automatically collects information required by ATLAS, caches and keeps it up to date, removing the external source as a direct dependency for clients but without duplicating the source information itself. Collected information is stored and exposed in a way more convenient to the ATLAS experiment. Additional data models and object relations are introduced in the system to fit requirements of the ADC applications and services.

Figure 1 shows the basic information entities stored in AGIS and represents a schematic experiment view of the physical resources. This unique AGIS information collection enables a mapping of the physical resources (GOCDB/OIM sites, CEs, SEs, LFCs, etc.) to the ATLAS activity end points (ATLAS sites, PanDA queues workload management endpoint, DDM spacetoken endpoints).

3. System architecture

The AGIS architecture is based on the classic client-server computing model. AGIS uses the Django [12] framework as a high-level web application framework written in Python. The Oracle database management system is chosen as a default database backend.

The object relation mapping technique which is built in the Django framework allows to access the content of database in terms of high level models, thus avoiding any direct dependence on the relational database system used. Since the system provides various client interfaces like Application Programming Interface (API), Web user interface (WebUI) and Command Line Interface (CLI) to retrieve and manage the needed data, no direct access to the database from the clients is required.

Figure 2 shows a schematic view of the AGIS architecture. To automatically populate the database with the information collected from external sources a set of collectors runs on the main AGIS server. All interactions with external information services are hidden. Synchronization of the AGIS content with related sources is performed by agents which periodically communicate with sources via standard interfaces and update the AGIS database content. For some types of information AGIS itself is the primary source. The clients are able to update information stored in AGIS through the API and the WebUI. The WebUI is mainly used to define new objects, modify existing properties and easily browse experiment specific resources from various user-friendly views. The AJAX (Asynchronous JavaScript and XML) technology is actively involved to offer efficient interactive access through the WebUI. A Python API and command line tools further help the end users and developers to use the system conveniently.

One of the operation requirements recently addressed to the information system is the possibility of flexible integration of AGIS functionality with the ADC services. It mainly concerns the evolution of API front end to become REST (Representational State Transfer) [13] style oriented. To meet this demand previously used AGIS Python API has been migrated to REST full application interfaces providing data export in JSON (JavaScript Object Notation) format. The variety of developed REST services, implemented filtering support and newer functionality to easily select and use multiple structures of output data (JSON presets suitable for specific client) help to increase the number of clients using AGIS in production.

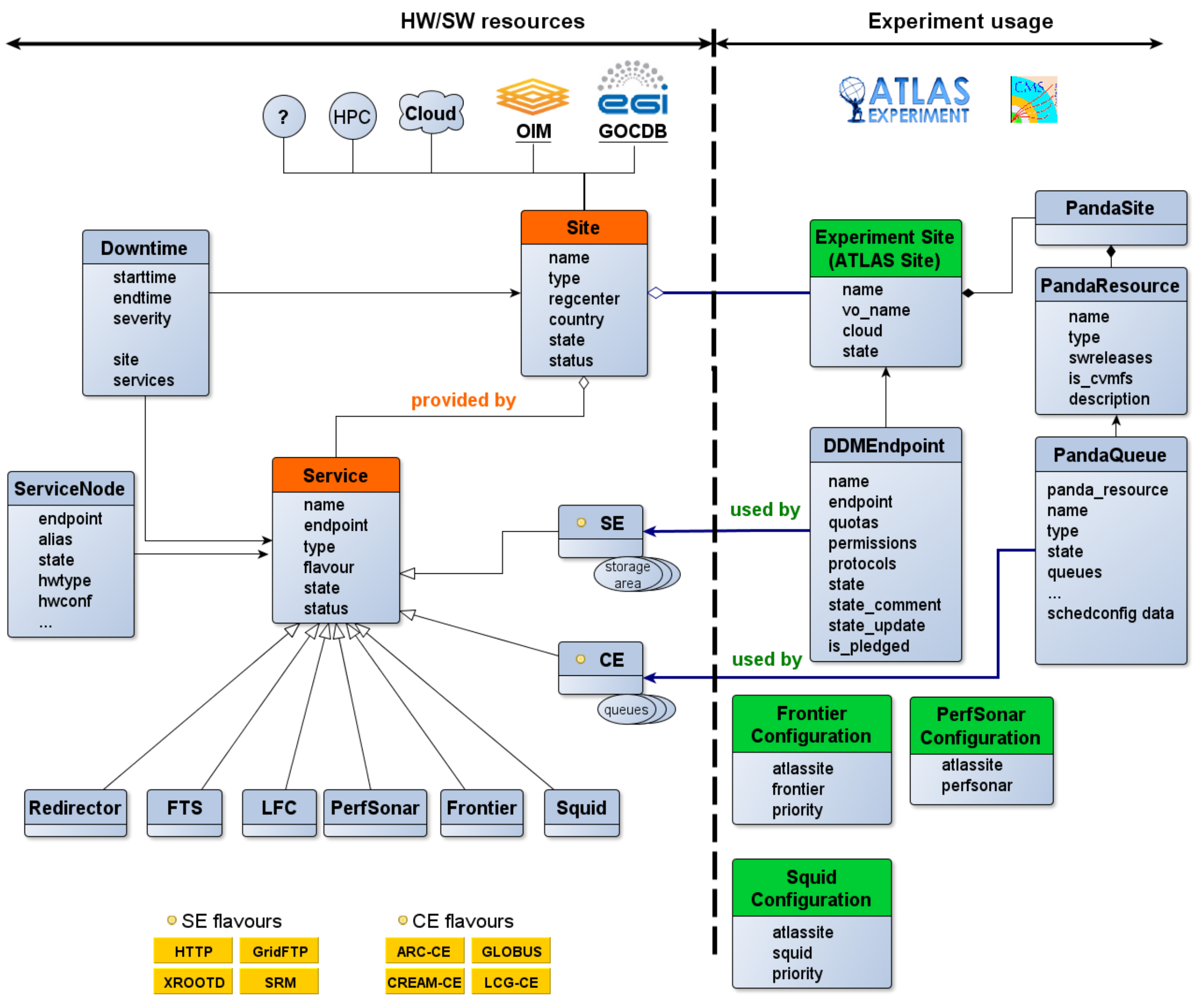

Figure 1. Experiment view of physical resources.

Today, the REST full client API allows users to retrieve data in JSON and, for some applications, in XML formats. For instance, all ATLAS topology can be exported as in an XML feed as in JSON structures suitable for the ADC applications.

4. FAX resources in AGIS

The concept of distinguishing between *'used by'* and *'hosted by'* site resources implemented in AGIS easily allows to introduce and define new services, which can already be used by ATLAS for specific use cases, although not yet integrated in production in WLCG.

An example of recently implemented new data in AGIS is the FAX services organized to federate data storage resources using an architecture based on XRootD with its attendant redirection and storage integration services. FAX storage federation is a way to remotely access ATLAS data without knowing where the data files are physically stored. AGIS provides a set of user-friendly WebUI forms to register and update FAX endpoints, describe redirectors topology, specify and associate various access protocols with the DDM endpoints and finally configure PandaQueue to enable FAX failover for the PanDA Pilots. REST style application interfaces allow to retrieve FAX related data programmatically.

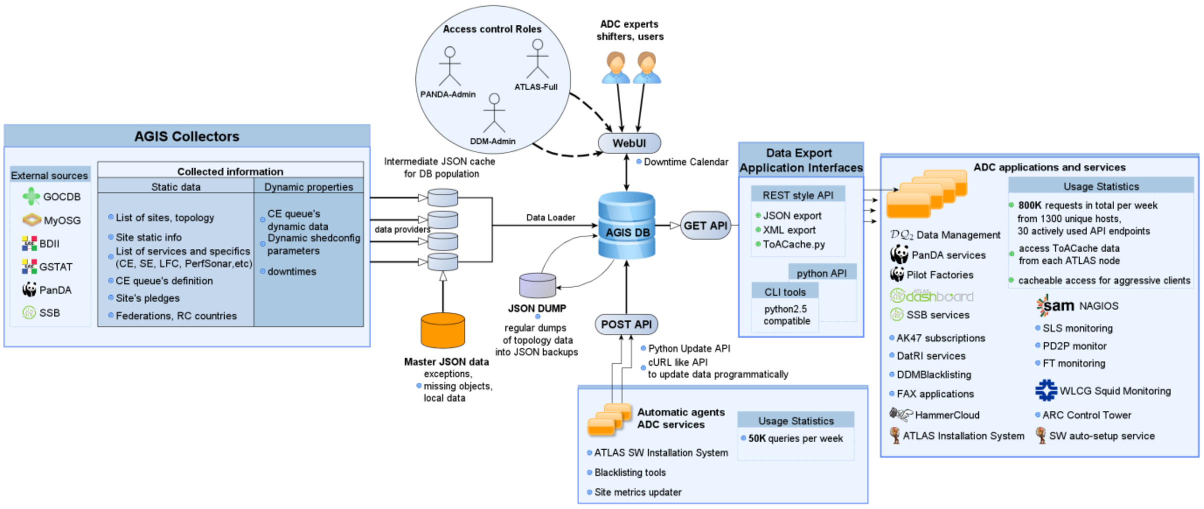

Figure 2. Schematic view of the AGIS client-server architecture.

To support PanDA job brokerage decisions AGIS is extending data models and developing unified interfaces to integrate various site performance indicators and metrics (Sonar, perfSONAR, XRootD measurements, etc) into a common cost matrix. Starting from basic cell granularity on the level of ATLAS site, AGIS automatically propagates and publishes stored network and performance values into a related object level (PandaResource, PandaQueue) used by the workload management system.

5. Stored information

There are two types of information stored in AGIS:

- data retrieved from external sources (PanDA, GOCDB database, BDII, MyOSG, WLCG REsource, Balance and USage (REBUS) source, etc.);
- data completely managed within AGIS.

In its current implementation, AGIS stores all information concerning ATLAS topology: clouds, regional centers, pledges, sites specifics, such as geography, time zone, etc. It also stores resources and services information at sites: the description of Local File Catalog (LFC), File Transfer Server, Computing Element (CE), Storage Resource Manager, Squid, Frontier, perfSONAR services and others.

From the point of view of data synchronization, the whole information stored in AGIS can be classified as static and dynamic. Dynamic data means regular synchronization against the information source from which it is collected. Technically it could be as a set of dynamic properties (for example, CE queue's status, free space at site) or as complete objects (e.g. downtime entries). Objects automatically injected into the system (e.g. GOCDB/OIM sites and services) are registered in the database in DISABLED states (hidden for data export through API), to make such object visible a user has to activate it via WebUI forms.

Key examples of information stored in AGIS are:

- ATLAS topology (clouds, centres, sites and sites specifics);
- PanDA schedconfig parameters;
- site resources and services information, status and its description;

- site information and configuration;
- software installations and tags;
- list of activities and their properties (functional tests, data distribution, etc.);
- global configuration parameters needed by ADC applications;
- user related information (privileges, roles, account info) for ADC applications;
- downtime information about ATLAS sites, list of affected services;
- site blacklisting data.

6. Web user interface

AGIS web user interface allows users to access the information system through their browser providing functionality to modify and visualize data stored in the system. The interface is focused on the use by the ADC community and does not require any special software development knowledge. Built-in tooltips and help messages integrated into the update information forms, completely prefilled example forms, clone object and bulk update functionalities increase usability for the majority of users. Implemented data validation checks, auto-completion of form field values (for example, suggestion to site name or service endpoint) enable user to quickly find and fill in form fields while browsing or updating information stored in AGIS.

The AGIS WebUI main features include:

- a declaration and update information related to ATLAS site topology, DDM storage elements (SE, DDMEndpoint), PanDA computing resources (CE, PandaResource, PandaSite, PandaQueue, schedconfiguration parameters);
- a set of interactive table views to browse the topology of ATLAS sites and their resources;
- tree based hierarchical listing of site resources presented from DDM and PanDA views;
- definition of site services (CE, SE, LFC, Redirector, perfSONAR, Frontier, Squid);
- functionality to perform bulk updates of object properties;
- Frontier-Squid and perfSONAR configurations management;
- XRootD redirectors topology definition;
- CloudResource, top-level regional center site definition;
- comparison and data validation tools (consistency checkers, comparators used to check static data in AGIS against external sources like BDII, GOCDB, OIM).

7. Access control management

Special attention is given to the implementation of user roles and privileges allowed to separate access within user groups using AGIS in production. The permissions are checked to determine a user's authorization to perform actions in the WebUI or API services. AGIS is able to determine what operations are allowed on given objects by using access control list definitions of an object that can be associated either with specific user or with user group. All actions to update information require *"write"* permissions to be granted to users. AGIS uses X509 user certificates (VOMs proxy certificates included) for authentication and authorization, taking into account certificate's DN values to identify a user.

The following groups are predefined in AGIS to simplify user management:

- *ATLAS-full* group includes modification privileges for any object in AGIS except user management itself;
- *DDM-Admin* group defines the list of update actions typically required to configure DDM related resources (SE, DDMEndpoint, DDMGroup, etc);
- *PANDA-Admin* group provides predefined set of privileges to manage the sched configuration and PanDA queues related objects.

Furthermore, two special root roles *is_superuser* and *is_staff* are defined to permit all actions on any objects of data models management and user management accordingly. Currently, more than one hundred users are registered in AGIS and granted administrator privileges for data modification.

8. Conclusions

AGIS has been developed to provide, in a single portal, the topology and resources information to configure the ATLAS computing applications. The system is currently in production and represents the primary source of information for all the ADC applications and services.

AGIS functionalities allow the ADC community, experts and shifters to configure and operate production ADC systems and Grid applications. AGIS is continuously extending data structures to follow new demands of ADC needs. The functionality to handle FAX storage federation services, implemented recently, is an example of fulfilled requests from ADC users.

The AGIS design and basic principles included in the architecture allow the use of the AGIS core part by several experiments. Recently implemented AGIS updates (Cloud Resources definition and top-level (GOCDB, OIM) site objects declaration, VO separation within the system, the functionality to define non ATLAS computing resources) open possibility to extend the system application for other HEP experiments. Some CMS computing resources are already defined in AGIS and used by PanDA. AGIS is evolving towards an experiment-non-specific information system.

References

[1] The ATLAS Collaboration 2008 The ATLAS Experiment at the CERN Large Hadron Collider *JINST* **3** S08003
[2] Jones R and Barberis D 2008 The ATLAS Computing Model *J. Phys.: Conf. Ser.* **119** 072020
[3] ATLAS Distributed Computing, web page, https://twiki.cern.ch/twiki/bin/view/AtlasComputing/AtlasDistributedComputing
[4] Anisyonkov A, Klimentov A, Krivashin D and Titov M 2010 ATLAS Distributed Computing *Nuclear Electronics & Computing: Proc.of the XXII International Symposium on Nuclear Electronics & Computing (NEC'2009) (Institute for Nuclear Research and Nuclear Energy, Varna, Bulgaria, September 7-14, 2009)* E10 11-2010-22 (Dubna) pp 45-49, http://nec2011.jinr.ru/pdf/nec2009.pdf
[5] Grid Configuration Database (GOCDB), https://wiki.egi.eu/wiki/GOCDB
[6] Open Science Grid Information service (MyOSG), http://myosg.grid.iu.edu/about
[7] Branco M et al. 2008 Managing ATLAS data on a petabyte-scale with DQ2 *J. Phys.: Conf. Ser.* **119** 062017
[8] Maeno T et al. 2008 PanDA: Distributed production and distributed analysis system for ATLAS *J. Phys.: Conf. Ser.* **119** 062036
[9] European Grid Infrastructure (EGI), http://www.egi.eu/
[10] Open Science Grid (OSG), http://www.opensciencegrid.org/
[11] NorduGrid project, http://www.nordugrid.org/
[12] Django project, http://www.djangoproject.com
[13] REST interfaces, http://en.wikipedia.org/wiki/Representational_state_transfer