

Distributed analysis in the ATLAS experiment

D.L. Adams^a, F. Brochu^b, C. Collins-Tooth^c, S. Hanlon^c, K. Harrison^b,
R.W.L. Jones^d, G. Rybkin^e, C.L. Tan^f

^a *BNL, Upton, NY 11973-5000, USA*

^b *Cavendish Laboratory, University of Cambridge, CB3 0HE, UK*

^c *Department of Physics and Astronomy, University of Glasgow, G12 8QQ, UK*

^d *Department of Physics, University of Lancaster, LA1 4YB, UK*

^e *Department of Physics, Royal Holloway, University of London, TW20 0EX, UK*

^f *School of Physics and Astronomy, University of Birmingham, B15 2TT, UK*

The ATLAS experiment will investigate high-energy proton-proton interactions at the Large Hadron Collider (LHC), due to start operation at CERN, Geneva, in 2007. Interactions with potential physics interest will trigger readout of the ATLAS detector, and data will be recorded for an estimated 10^9 triggers per year. Analysis of the triggered interactions, including associated simulation studies, will require offline processing of a data volume of 5-10 petabytes.

ATLAS involves a collaboration of more than 1800 physicists, from over 150 institutes in 34 countries, spread over 6 continents. The experimental data, recorded at CERN, will be shared between national and regional centres for processing. This includes collaboration-wide production, where interactions are reconstructed, and are tagged according to their characteristics; and analysis, where small work groups and individual physicists are interested in reconstructed interactions with particular tags.

The ATLAS Distributed Analysis (ADA) system is being developed to support the experiment's analysis activities, providing for a worldwide user community that must access distributed data and resources. ADA is required to work across major Grid developments such as LCG, Grid3 and NorduGrid, and emerging new systems, such as gLite and EGEE. ADA must also provide provenance tracking, to ensure that an analysis carried out by one physicist can be readily checked by another.

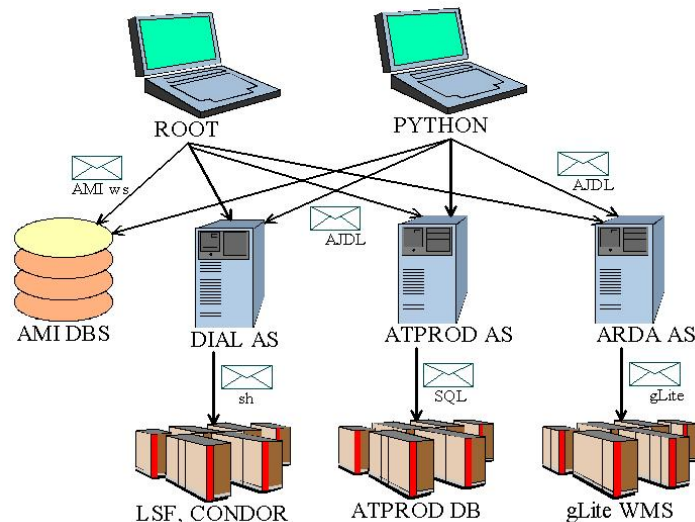


Figure 1: ADA architecture

The main components of ADA (Figure 1) are web services to manage processing, catalogue services to record data and its provenance, clients to interact with these services, and an Abstract Job Definition Language (AJDL) used to format messages between clients and services. The first release of ADA benefits significantly from work in related projects: much of the ADA model and underlying software is inherited from DIAL, catalogue services are being developed in the context of AMI, and a user interface is provided by Ganga.

In AJDL, a job is represented as a transformation acting on a dataset. A transformation, here, is essentially a logical identifier of an application (for example, the ATLAS reconstruction), and a task definition, which is the user's specification of how the application should be run. The task definition can include values for configurable parameters, and source code to be compiled into a shared library that the application should load. At the lowest level, a dataset may be a collection of files. ADA has C++ classes from DIAL for producing and using AJDL descriptions formulated in XML.

Processing in ADA is performed by analysis services. These are batch systems accessed via a web-service interface that accepts jobs represented in AJDL. The analysis service makes use of a lightweight package manager to locate the software needed to apply the transformation specified in the job request, and has tools for retrieving datasets. Analysis services available with the first release of ADA allow submission to LSF, Condor and gLite. Work is in progress on interfacing to the ATLAS production system, which defines a number of useful transformations, and provides access to significant resources.

AMI-based catalogue services are being developed for storage and retrieval of transformations and datasets.

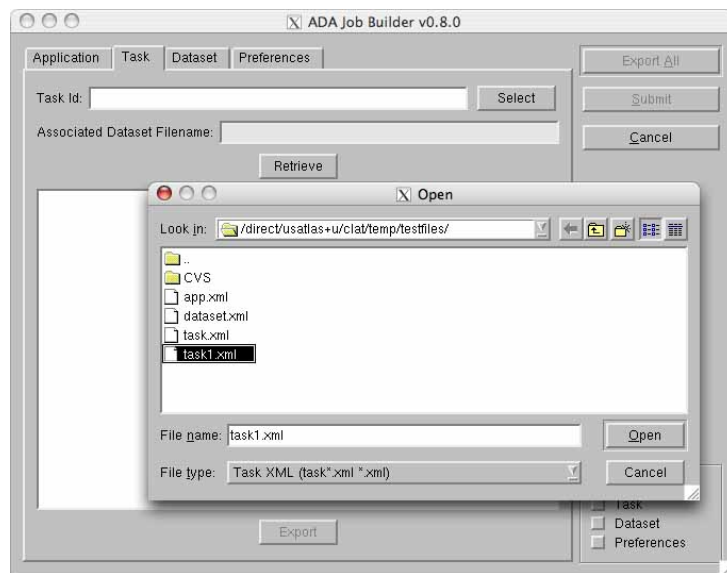


Figure 2: Graphical job builder. The smaller window allows selection of an existing task definition, which may subsequently be edited in the larger window.

Users can interact with ADA either from the ROOT analysis framework, or from Python. In both cases, use is made of dictionaries constructed by parsing DIAL C++ header files. The Python client is maintained in the context of the Ganga project, and is used as the basis for a graphical interface (Figure 2).

This presentation gives details of the ADA design, and reports on the performance achieved with the first release. Emphasis is given to the UK contributions, in the areas of catalogue services, package management, exploitation of the ATLAS production system, and user interface.