SLAC-PUB-13549

GLAST (FERMI) Data-Processing Pipeline

Daniel L. Flath,[1] Tony S. Johnson, Massimiliano Turri, and Karen A. Heidenreich

SLAC National Accelerator Laboratory, Menlo Park, CA, U.S.A.

Abstract. The Data Processing Pipeline ("Pipeline") has been developed for the Gamma-Ray Large Area Space Telescope (GLAST) which launched June 11, 2008. It generically processes graphs of dependent tasks, maintaining a full record of its state, history and data products. The Pipeline is used to automatically process the data down-linked from the satellite and to deliver science products to the GLAST collaboration and the Science Support Center and has been in continuous use since launch with great success. The pipeline handles up to 2000 concurrent jobs and in reconstructing science data produces approximately 750 GB of data products using 1/2 CPU-year of processing time per day.

1. Introduction

GLAST, now called Fermi, is a high-energy gamma-ray observatory launched in June of 2008. In order to provide prompt processing of science data, a distributed computing facility is required (Dubois 2009). The Fermi Data-Processing Pipeline provides a generic means by which users can define arbitrarily complex processing work-flows ('Tasks'), manage and monitor these processing tasks and easily interface with the Fermi Data Catalog.

The Pipeline software must be capable of processing downlinked Level-0 (raw) instrument data into Level-1 (reconstructed) science products and delivering these to the Fermi Science Support Center within 24 hours of Level-0 receipt. The Level-0 data arrive in eight daily downlinks and the software is officially required to handle 2× this nominal load in the event that a delivery interruption occurs. The system must be capable of recursively splitting a set of events into parallel streams of processing. A very low failure rate and high throughput are required to push thousands of jobs through the system prior to receiving the next downlink.

2. Architecture and Technologies

The Pipeline processing facility uses a three-tiered architecture as shown in Figure 1.

[1]For the LAT Collaboration. SLAC is operated by Stanford University for the US Department of Energy.

Work supported in part by US Department of Energy contract DE-AC02-76SF00515.

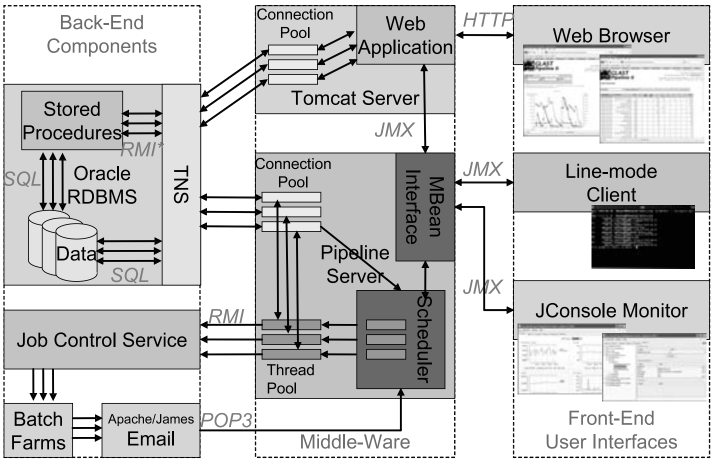

Figure 1.: Data-processing pipeline architecture.

2.1. Back-End Components

Oracle Database We run a pair of Sun Niagara-class, 64-thread servers in a primary/secondary redundant configuration. All processing state, past and present, is stored in database tables. We make extensive use of Oracle technologies. The highly-configurable Oracle scheduler is used to run periodic jobs tabulating various measures of system throughput, including resource usage of the Oracle Server itself. These quantities are then made available to users via trending plots in the web front-end. We use stored procedures in both Java and PL/SQL (Oracle's proprietary language) to perform query-intensive tasks. For example, the rules-engine which evaluates dependencies upon process completion and determines which among the dependent processes should be executed or skipped is run entirely within the database.

Job Control Service and Batch Farms We currently perform processing on two farms; the Platform-LSF[2] cluster at SLAC and the BQS[3] cluster at Lyon, France. Each cluster-specific tool set is wrapped to provide a uniform interface (*SubmitJob, GetJobStatus, KillJob, etc.*). Applications known as a Job Control Services publish the uniform interface though Remote Method Invocation (RMI[4]) and handle requests from the Pipeline Server.

Email Messaging Email is used to provide asynchronous, persistent messaging from batch-jobs to the pipeline server. Email is sent at the beginning of each

[2]http://www.platform.com/Products/platform-lsf

[3]http://cc.in2p3.fr/rubrique351.html

[4]http://java.sun.com/javase/technologies/core/basic/rmi/index.jsp

batch job to inform the pipeline server that the job has started, which node accepted the job, and any other pertinent information. Another email is sent when the job finishes to transmit the return-code, resource-usage, elapsed wall-clock time, any variables the process wanted to pass to it's dependents, and system commands the process wants executed by the server (e.g.: create four streams of the "Reconstruction" sub-task.) Because the system sends tens of thousands of email messages per day, we maintain a dedicated email server running the free Apache JAMES[5] software.

2.2. Middle-ware

Pipeline Server The Ring-Master of this three-ring circus, the Server maintains two pools of Java threads, a work pool and an admin pool.

The work pool is used to manage processes. When a batch-process becomes available to run on one of the farms, a thread is allocated to perform the submission using the appropriate Job Control Service. Email status reports from the batch jobs are also handled by these threads which make updates to the database tables that record processing state. Users may also define Jython 'scriptlets' which are run in these threads. These Jython scriptlets are provided APIs to both the Pipeline Server and the Fermi Data Catalog. The Pipeline Server API allows querying of other processes, creation of sub-streams, getting/setting variables, etc. The Data Catalog API allows registration of output datasets and querying for input datasets.

The admin thread pool runs tasks which find work and delegate it to the worker pool. These include gathering status messages from the mail server and querying the database for processes which are ready to run.

A subset of the pipeline API is also made available via Java Management Extensions (JMX[6]) providing a call interface to the various user-interface applications.

Web Applications A cluster of Apache Tomcat servers host pages written in JSP. These pages provide a graphical user-interface across platforms and available world-wide. The applications read data for display directly from Oracle using SQL queries. Where interaction with the Pipeline Server is required, Java Tag Libraries are used to make JMX requests. Tag libraries also provide more complex functionality such as plotting[7].

2.3. Front-End User Interfaces

User interfaces are provided through the Internet using the web-applications discussed above as well as through command-line interfaces. In each case JMX is used to communicate user-requests to the pipeline server. CAS[8] user authentication, linked to the laboratory's user database, coupled with a tiered

[5]http://james.apache.org/

[6]http://java.sun.com/javase/technologies/core/mntr-mgmt/javamanagement/

[7]http://aidatld.freehep.org/

[8]http://www.ja-sig.org/products/cas/

user-privilege model allows fine control over the operations any user is allowed to perform.

3. Performance and Reliability

The pipeline software shepherds thousands of jobs per day with a daily average throughput of about 1/2 CPU-year of processing. The peak usage to date has been 45,000 jobs in a single day and 30 CPU-years of processing in a single month. Remarkably, the Level-1 Data Processing task experiences a overall job-failure-rate of only 0.13%. This rate is effectively reduced to 0.03% (or ≈ 1 failure requiring operator intervention per week) by automatically re-running failed jobs to reduce the contribution of transient failures such as I/O errors.

The reliability of the system is due in no small part to heavy stress testing of simulated data for several months prior to launch which fleshed out performance bottlenecks and suggested tuning of code and related database table-structure and queries.

4. Portability

Other than the tight coupling to Oracle Database, the Fermi Data-Processing Pipeline uses only open-source third-party components. The system may be easily extended to other batch-processing systems by implementing a simple Java Interface. Condor is an obvious (free and light-weight) candidate, as are the so called “Grid-tools.”

Acknowledgments. The *Fermi* LAT Collaboration acknowledges generous ongoing support from a number of agencies and institutes that have supported both the development and the operation of the LAT as well as scientific data analysis. These include the National Aeronautics and Space Administration and the Department of Energy in the United States, the Commissariat à l’Energie Atomique and the Centre National de la Recherche Scientifique / Institut National de Physique Nucléaire et de Physique des Particules in France, the Agenzia Spaziale Italiana and the Istituto Nazionale di Fisica Nucleare in Italy, the Ministry of Education, Culture, Sports, Science and Technology (MEXT), High Energy Accelerator Research Organization (KEK) and Japan Aerospace Exploration Agency (JAXA) in Japan, and the K.A. Wallenberg Foundation, the Swedish Research Council and the Swedish National Space Board in Sweden.

Additional support for science analysis during the operations phase from the following agencies is also gratefully acknowledged: the Istituto Nazionale di Astrofisica in Italy and the K.A. Wallenberg Foundation in Sweden for providing a grant in support of a Royal Swedish Academy of Sciences Research fellowship for JC.

References

Dubois, R. 2009, in ASP Conf. Ser. 411, ADASS XVIII, ed. D.A. Bohlender, D. Durand & P. Dowler (San Francisco: ASP), 189