Distributed Tracking, Storage, and Re-use of Job State Information on the Grid

Daniel Kouřil, Aleš Křenek, Luděk Matyska, Miloš Mulač, Jan Pospíšil, Miroslav Ruda, Zdeněk Salvet, Jiří Sitera, Jiří Škrabal, Michal Voců
CESNET, Zikova 4, Prague, Czech Republic, and Charles, Masaryk and West Bohemia Universities

24th September 2004

The Logging and Bookkeeping service *tracks job passing through the Grid. It collects important events generated by both the grid middleware components and applications, and processes them at a chosen L&B server to provide the job state. The events are transported through secure reliable channels. Job tracking is fully distributed and does not depend on a single information source, the robustness is achieved through speculative job state computation in case of reordered, delayed or lost events. The state computation is easily adaptable to modified job control flow.*

The events are also passed to the related Job Provenance (JP) service. Its purpose is a long-term storage of information on job execution, environment, and the executable and input sandbox files. The data can be used for debugging, post-mortem analysis, or re-running jobs.

The data are kept by the job-provenance storage service in a compressed format, accessible on per-job basis. A complementary index service is able to find particular jobs according to configurable criteria, e.g. submission time or "tags" assigned by the user.

Both the L&B and JP index server provide web-service interfaces for querying. Those interfaces comply with the On-demand producer specification of the R-GMA infrastructure. Hence R-GMA capabilities can be utilized to perform complex distributed queries across multiple servers. Also, aggregate information about job collections can be easily provided.

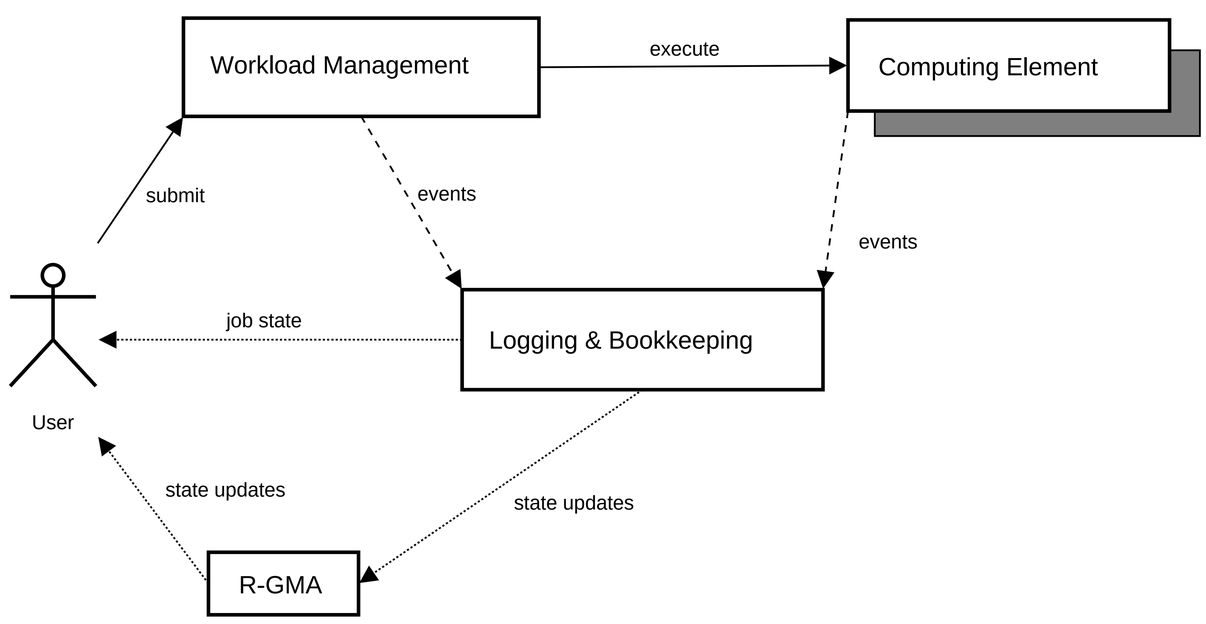
The L&B service was deployed in the EU DataGrid and Cern LCG projects, both L&B and JP will be deployed in the EGEE project.

L&B service

In a Grid environment, use of advanced Grid services like Resource broker implies lost of direct control over placement of submitted jobs. Security may prevent direct contact with computing element where job is running.

Additional complexity accompanies parallel and multipart jobs as well as job assemblies, where thousand and more tasks may represent one logical job. It is impossible to track all the jobs manually. However, jobs could get lost and be mistreated and some monitoring information is neccessary to track and repair the cause of such an error. While monitoring in general is an on-going activity within the Grid community (see, e. g., [1]), there is no such activity specifically focused on job tracking and monitoring.

The L&B service (see, e. g., [2]) collects events generated as a result of job flowing through the individual Grid middleware components. It supports distributed WMS architecture [3] consisting of various job handling components. The components are instrumented to generate events when a job passes through them, these events are sent through the network to an appropriate L&B database. Managing data not directly accessible otherwise, the L&B provides individual events, a global view on job states, as well as aggregate information on compound jobs.



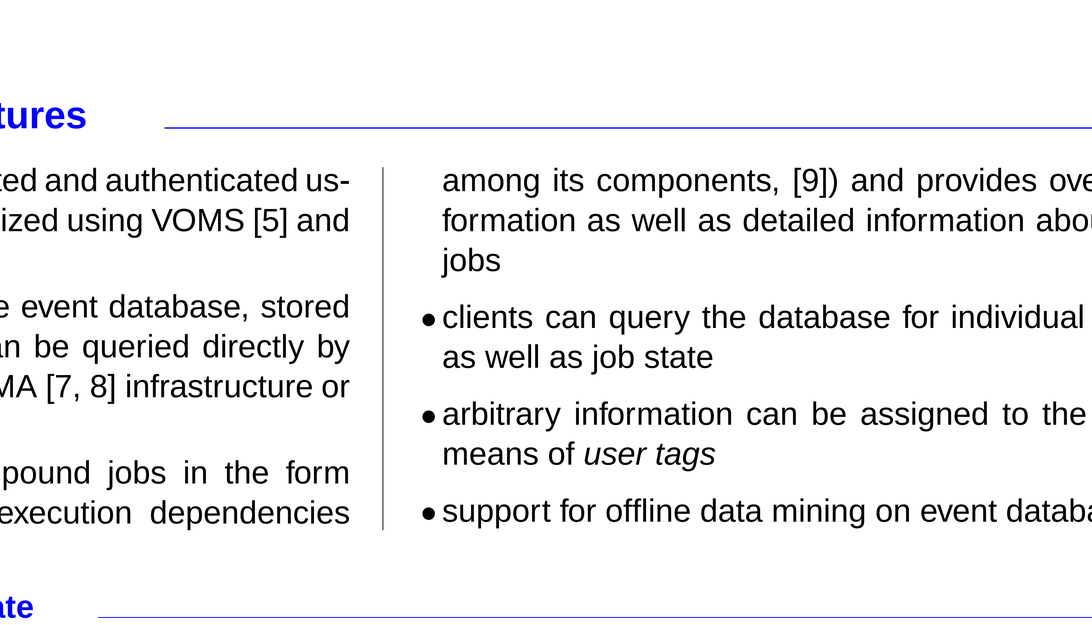
Main features

- L&B service collects, delivers and stores *events* describing the job lifetime
- each job is uniquely identified by its *jobid* and each L&B event is assigned to a single grid job
- information are collected using *push model* (WMS components actively send information)
- logging of data is asynchronous and independent of state of other grid services
- all data transfers are encrypted and authenticated using GSI [4], access is authorized using VOMS [5] and ACLs [6]
- job *state* is inferred from the event database, stored in the L&B databaseand can be queried directly by the clients, fed into the R-GMA [7, 8] infrastructure or announced via *notifications*
- L&B service supports compound jobs in the form of *DAGs* (multi–jobs with execution dependencies among its components, [9]) and provides overall information as well as detailed information about subjobs
- clients can query the database for individual events as well as job state
- arbitrary information can be assigned to the job by means of *user tags*
- support for offline data mining on event database

Job state

Every job during its lifetime passes through various phases, associated with job states. Job state is computed from particular events sent by WMS components while handling the job. In general an event indicates a change in job *state*. Job states for the current WMS and recognized by L&B service are defined as follows:

SUBMITTED	. . . job is entered by the user to the grid
WAITING	. . . job has been accepted and is waiting for further processing and/or resource discovery
READY	. . . job has been assigned to appropriate Computing Element
SCHEDULED	. . . job is waiting in the queue on the ComputingElement (this state never occurs with DAG/partitioned job)
RUNNING	. . . job is running or DAGMan is processing its subjobs (if the job is DAG/partitioned)
DONE	. . . job has exited or is considered to be in a terminal state (submission failed unrecoverably)
ABORTED	. . . job processing was aborted by WMS (waiting in the Workload Manager queue or ComputingElement for too long, over-use of quotas, expiration of user credentials, . . .)
CANCELLED	. . . job has been successfully cancelled by user request
CLEARED	. . . output sandbox was transferred to the user or removed due to the timeout



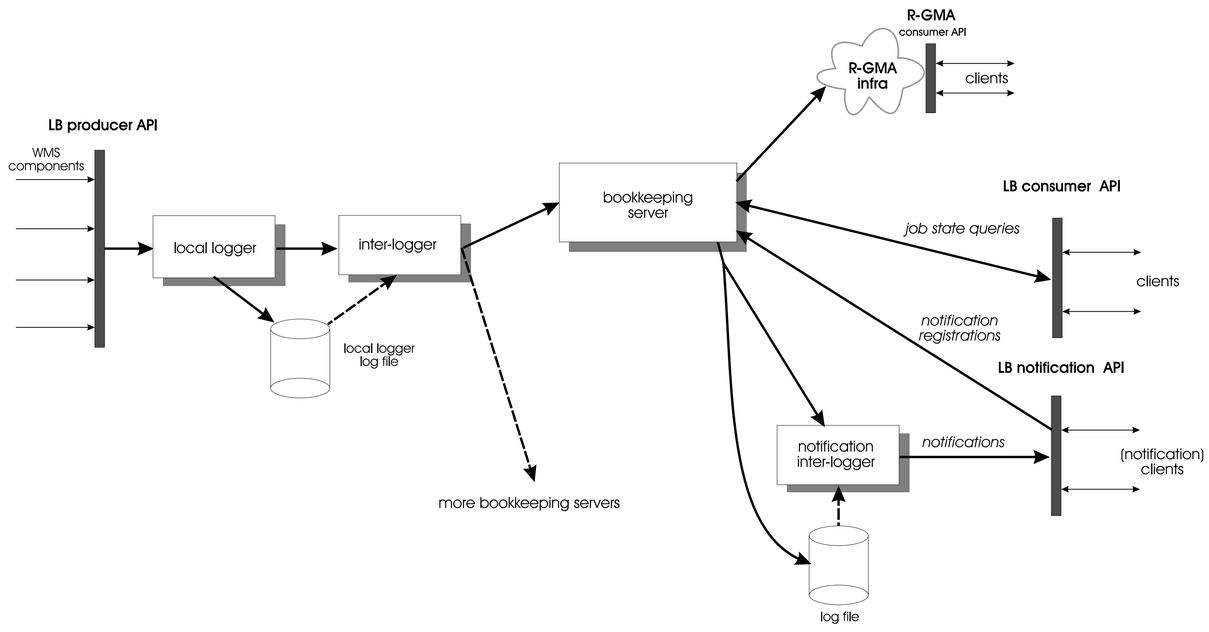
Architecture

API's implemented as a simple libraries provide access to the core L&B services:

- Logging (producer) API is used by the event sources to pass information to the L&B database. It provides locally persistent non-blocking calls regardless of the state of other L&B components, but their synchronous (blocking) counterparts are also available.
- Server (consumer) API serves for querying the contents of L&B database (events and job states).
- Notification API allows clients to register with the L&B database and receive information about changes in job states.

Transport infrastructure is responsible for reliable, secure and fast transfer of events over WAN to the L&B database. The *local logger* daemon accepts messages sent via logging API, stores them into local file and passes them on to the *interlogger* daemon, which tries to deliver the messages to appropriate database until it succeeds. The separation of functionality between the two processes ensures desired properties: reliability and fast response independent of state of other grid services.

Bookkeeping server stores events delivered from WMS components into local database, performs on–line computation of job state and informs the subscribed clients about job state changes using either R-GMA or L&B notification infrastructure. It also responds to direct user queries about job states and events.



Job Provenance

The purpose: To provide long lived (years) storage for definition of submitted jobs, execution conditions and environment, and important points of the job life cycle. Only information on completed jobs (regardless of their completion status) is provided. Stored data can be used for debugging, post-mortem analysis, comparison of job execution within an evolving environment, as well as assisted re-execution of jobs.

It is practically impossible to record the entire job execution environment—the ideal case would encompass an entire Grid snapshot. We restrict the recorded data to those that are processed or somehow affect processing of the Workload Management and Computing Element services. Snapshots of the state of other Grid services are not done, namely queries to the Data Catalogue and their results are not stored, as well as contents of data files down- and uploaded from and to Storage Elements.

Data required by the JP service are gathered by various Grid middleware components, using both the "pull" and "push" modes.

Job Record is a collection of following parts, containing all the relevant information about a particular job.

Job life log taken over from the L&B database. It contains the complete definition of the job (in terms of the submitted JDL), various timestamps, information on the chosen CE (or more of them, if the job was resubmitted), result of execution (including return code, if possible), the user-defined tags specified either statically upon job submission, during its execution, or overridden after the job terminates.

Executable file(s) provided by the user upon submission. For the sake of storage effectivity, JP has to allow sharing a stored executable among multiple jobs.

The executable and input sandbox is managed by the submission interface of the Workload Management service, instrumented to cooperate with the JP.

Execution environment consists of information about operating system and installed software versions, and specific configuration information (including values of environment variables) of the particular worker node where the job has been executed.

The primary source of the most up-to-date information of this type is the worker node itself, the script which wraps invocation of the job executable is being instrumented to log the required information. However, certain information can be also retrieved from CE configuration management service.

Input/output sandbox contains miscellaneous files required for execution and miscellaneous output (debug logs, core file). Staging the sandboxes in and out is managed by the Network Server.

Custom data may be provided by the user upon submission, gathered during job execution (via plugin script called by the job wrapper), or even after the job termination, through the interaction with JP itself.

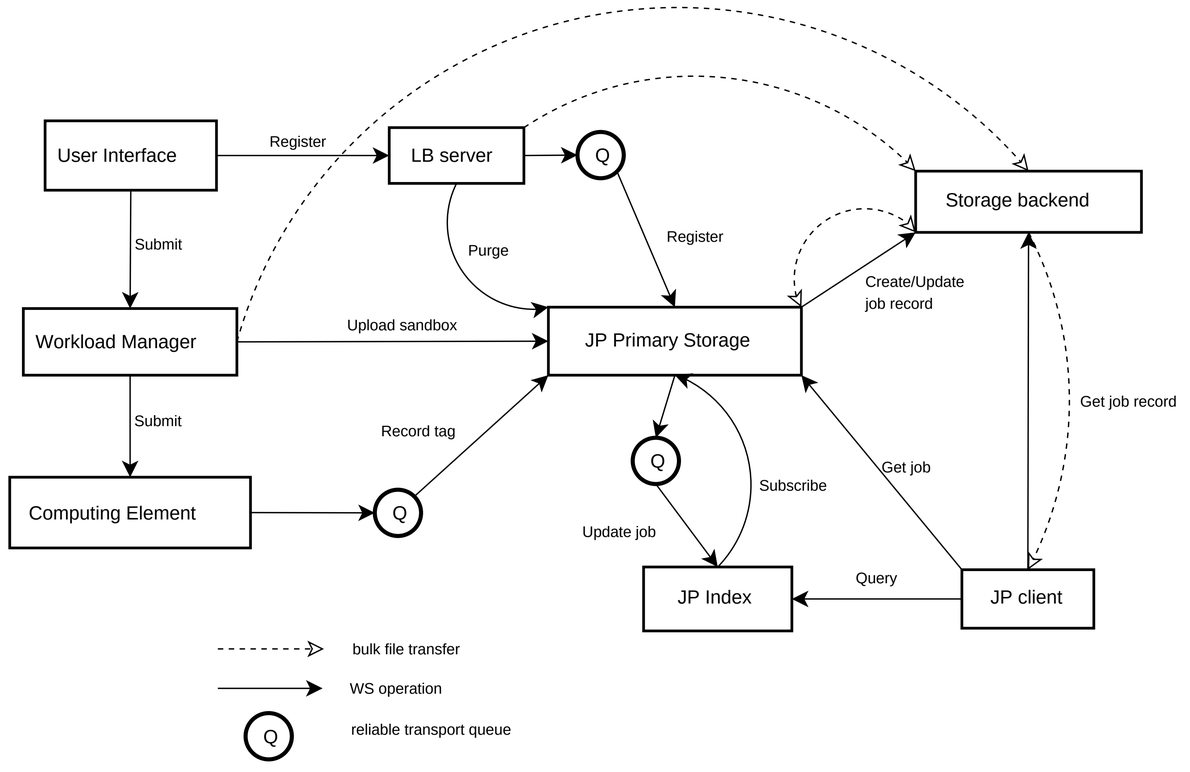
Architecture

Primary storage servers

- keep all the job JP data in the most compact and economic form of *Job Record*, use of metadata enables sharing of large files among jobs
- provide public interfaces for storing and retrieval of job records and registration of *index servers*
- job record is accessed using *jobid* (jobid serves as a primary key) or additional metadata (currently submit time and job owner)
- access control mechanism is based on ACL's taken from L&B data
- may use different backends for job record storage (embedded gridftp server, SE)

Index servers

- provide a limited data mining capability on the JP data; each index server is configured to support set of *queryable attributes* (subset of L&B attributes, user tags and input sandbox filenames), some of them marked as *indexed*
- provide query interface with approximately the same strength as the L&B server (query may refer only to queryable attributes and must contain indexed attribute for performance reasons), server responds with list of jobids and primary storage server contacts
- receive data from primary servers either as *batch* of dumped L&B data (for jobs in given time interval or when queryable attributes change) or *incrementally* (when new data arrive to JP storage)



References

[1] B. Tierney et al: *A Grid Monitoring Architecture*. Global Grid Forum Working Draft, `http://www-didc.lbl.gov/GGF-PERF/GMA-WG//papers/GWD-GP-16-2.pdf`

[2] D. Kouřil et al: *Logging and Bookkeeping Service for the DataGrid*. EU DataGrid project document, `http://server11.infn.it/workload-grid/docs/DataGrid-01-TEN-0109-1_0.pdf`

[3] G. Avellino et al. *The EU DataGrid Workload Management System: towards the second major release*. Computing in High Energy and Nuclear Physics (CHEP03), USA, 2003.

[4] I. Foster, C. Kesselman, G. Tsudik and S. Tuecke: *A Security Architecture for Computational Grids*. Proc. 5th ACM Conference on Computer and Communications Security Conference, 1998.

[5] R. Alfieri , R. Cecchini , V. Ciaschini , L. dell Agnello, A. Frohner, A. Gianoli, K. Lorentey, and F. Spataro: *VOMS, an Authorization System for Virtual Organizations*. 1st European Across Grids Conference, Santiago de Compostela, 2003

[6] A. McNab: *Grid-based access control and user management for Unix environments, Filesystems, Web Sites and Virtual Organisations*. CHEP 2003, La Jolla, California, 2003.

[7] A. Cooke at al.: *Relational Grid Monitoring Architecture (R-GMA)*. Presented at UK e-Science All-Hands meeting, Nottingham, UK, September 2-4, 2003. `https://edms.cern.ch/file/400756/1/rgma.pdf`

[8] A. Křenek, L. Matyska, Z. Salvet, J. Sitera. *Efficient Distribution of Grid Jobs State Data via Monitoring Infrastructures*. Submitted to TNC 2004.

[9] F. Giacomini, F. Prelz, M. Sgaravatto, I. Terekhov, G. Garzoglio, T. Tannenbaum: *Planning on the Grid: A Status Report*. INFN-GRID Project Technical Report, 2002.