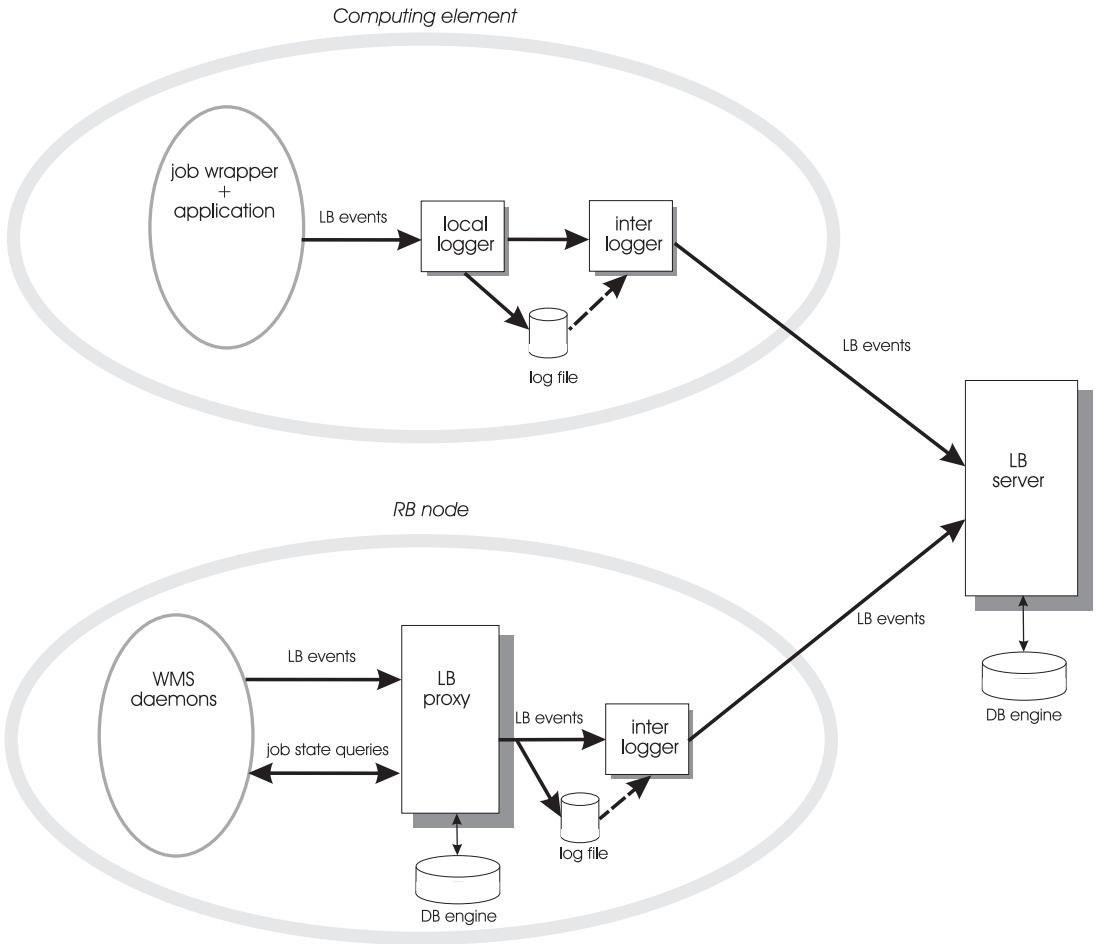
Logging and Bookkeping service – Murphy's design view

Aleš Křenek, on behalf of EGEE JRA1 CZ cluster

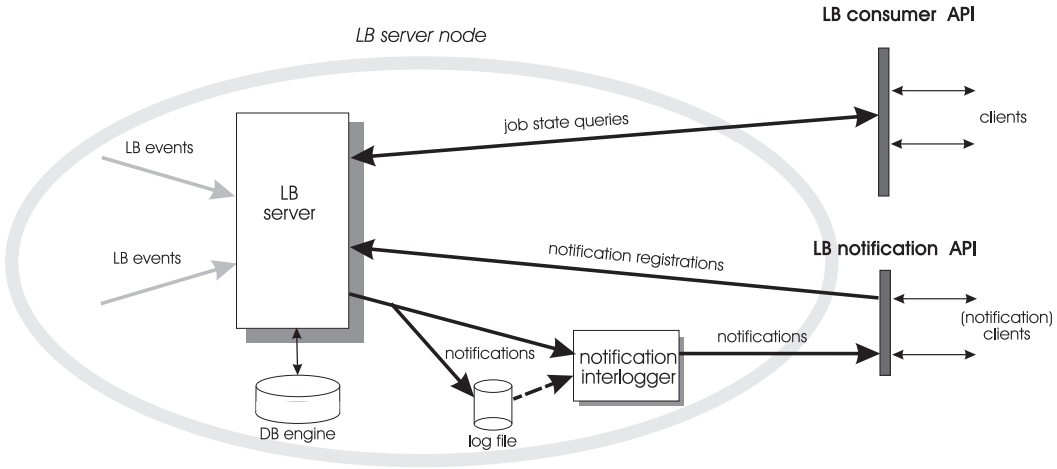
L&B architecture in a nutshell

- Grid components generate **L&B events**
  - any important point in job life
  - eg. transfer between components, start running, done, ...
- events are delivered to **L&B server**
  - uniquely and immutably assigned to job (jobid prefix)
- L&B server cooks raw events into user-digestable **job state**
- users pose **queries**, subscribe for **notifications**
- passed stress-tests of >1M jobs/day

L&B architecture in a nutshell (2)



L&B architecture in a nutshell (3)



Asynchronous message delivery

Local logger

- local delivery semantics
  - accept event, store in local file, return OK
  - no latency chaining (don't wait for other service to complete)
- close to event source
  - the same machine, cluster front-end, ...
  - minimize network outages and latency
- one file per job
  - locked wrt. interlogger
  - append only

Asynchronous message delivery (2)

Interlogger

- read job files, deliver to final destination
  - maintain pointer "start delivery here" (aux file)
  - read only
  - purge "all delivered" files eventually
- one thread per destination
  - hanging connection does not block others
- home-made implementation
  - messaging systems (e.g. JMS) would provide all functionality
  - tracing problems under production load is almost impossible

L&B server internals

Processes

- dispatcher
  - accept TCP connection
  - monitor load, send reject message eventually
  - forward connections to slaves
  - restart terminated slaves
    - bullet-proof software error handling (SEGV etc.)
- slave
  - do the workload, including SSL handshake
  - keep idle connections for while, ready to preempt
  - terminate voluntarily after some # of requests
    - upper bound on resource leakage
  - dispatcher-slaves structure in reusable library

L&B server internals (2)

Job state machine

- compute job state from raw events
- high redundancy
  - 3 events on job transfer between WMS components
    - start,accept by other side, OK.
- don't rely on timestamps
  - complex hierarchical sequence code
  - logical timestamp, works asynchronously, deals with failures
- don't be too general
  - reasonably tailored to gLite job life cycle
  - can be extended
  - multiple different state machines (PBS, Condor)

L&B server internals (3)

Critical sections

- no need for read locks
  - DB-level isolated transactions or home-made tricks
- coarse-grain write locks
  - job level
  - small set of semaphores
  - jobid hashed to semaphores evenly and randomly
  - configurable # of semaphores (= # server slaves by default)
- fast, resilient to failures
  - any held semaphore is released when slave dies

L&B server internals (4)

L&B proxy

- restricted mode of L&B server
- accept events and queries over local socket
- no SSL (Unix authz only)
- local view on job state, synchronous semantics
- used by WMS to avoid keeping persistent info on its own

GSS layer

- usual networking API
  - accept/connect, read/write, close
- on top of GSS-API
  - currently using GSS/GSI from Globus
  - any GSS implementation can be plugged in
- strict timing out
  - any network operation can take indefinite time
  - firewalls :-(
  - any network operation (including GSS transport) guarded by `poll(2)`
  - DNS – use of `c-ares`
  - better control wrt. usual thread-per-connection

Conclusion

"Anything that can go wrong, will."

Conclusion

"Anything that can go wrong, will."

- The Grid is weird, we don't control it
  - service may not be available when needed :-(
  - it is likely to turn up eventually :-)
  - **asynchronous invocation wherever possible**
- No software is perfect
  - unexpected failures and leaks occur
  - don't expect to ever fix them all (may not be yours)
  - **refresh processes, bind resource usage**
- The simpler the better
  - externals: avoid bleeding edge technology, not stable in general
  - yours: don't plan for next-decade requirements, they'll change
  - **complex systems are difficult to debug, don't make it worse**