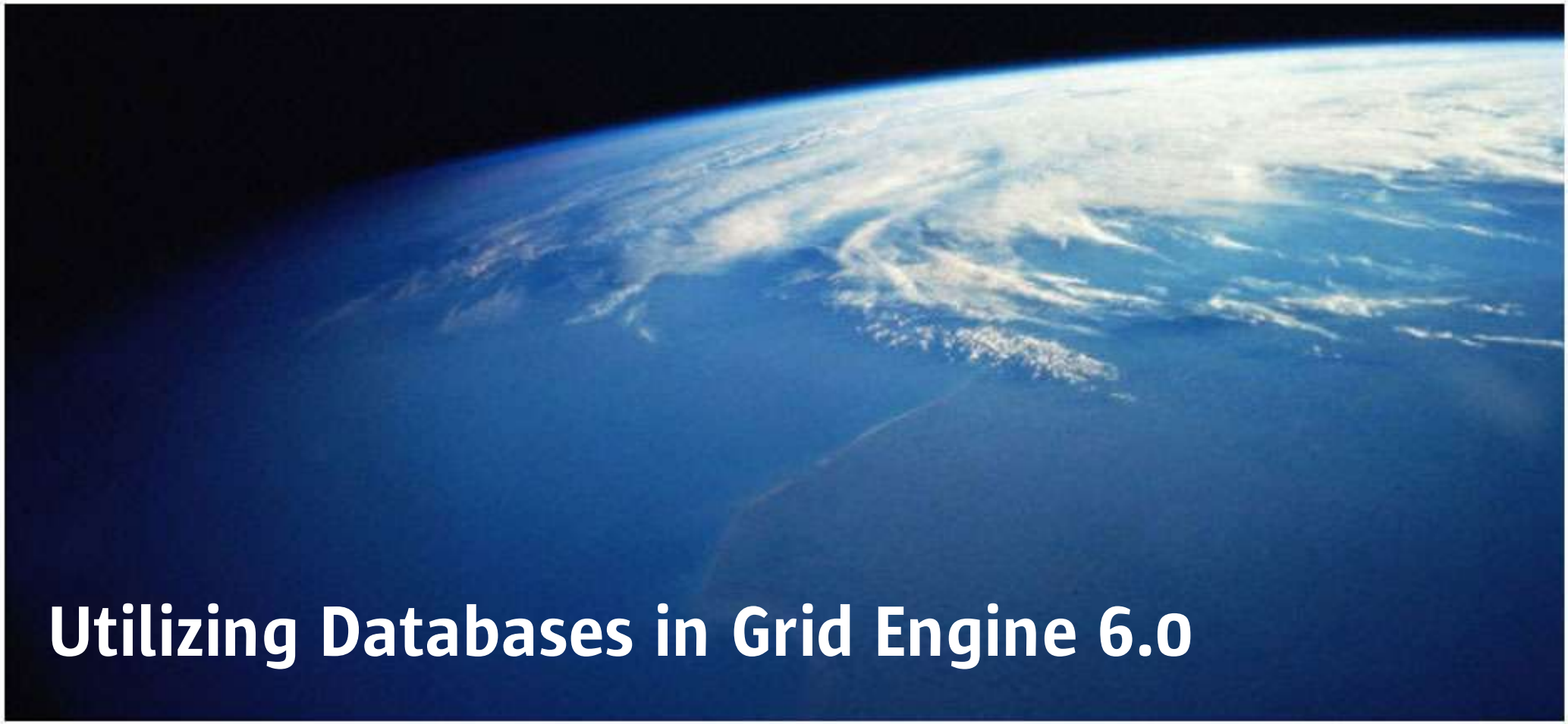


Utilizing Databases in Grid Engine 6.0

Joachim Gabler
Software Engineer
Sun Microsystems

http://sun.com/grid

Sun microsystems
We make the net work.

Current status

- flat file spooling
  - binary format for jobs
  - ASCII format for other objects
- accounting file
- statistics file
- history file

Reasons for database usage

- Scalability
  - bottleneck qmaster spooling
  - accounting
- Accessibility

  SQL, ODBC, JDBC
- ACID

  atomicity, consistency, isolation, durability
- Maintenance

  online backup, truncate historical data

Plans for 6.0

Two separate applications of databases

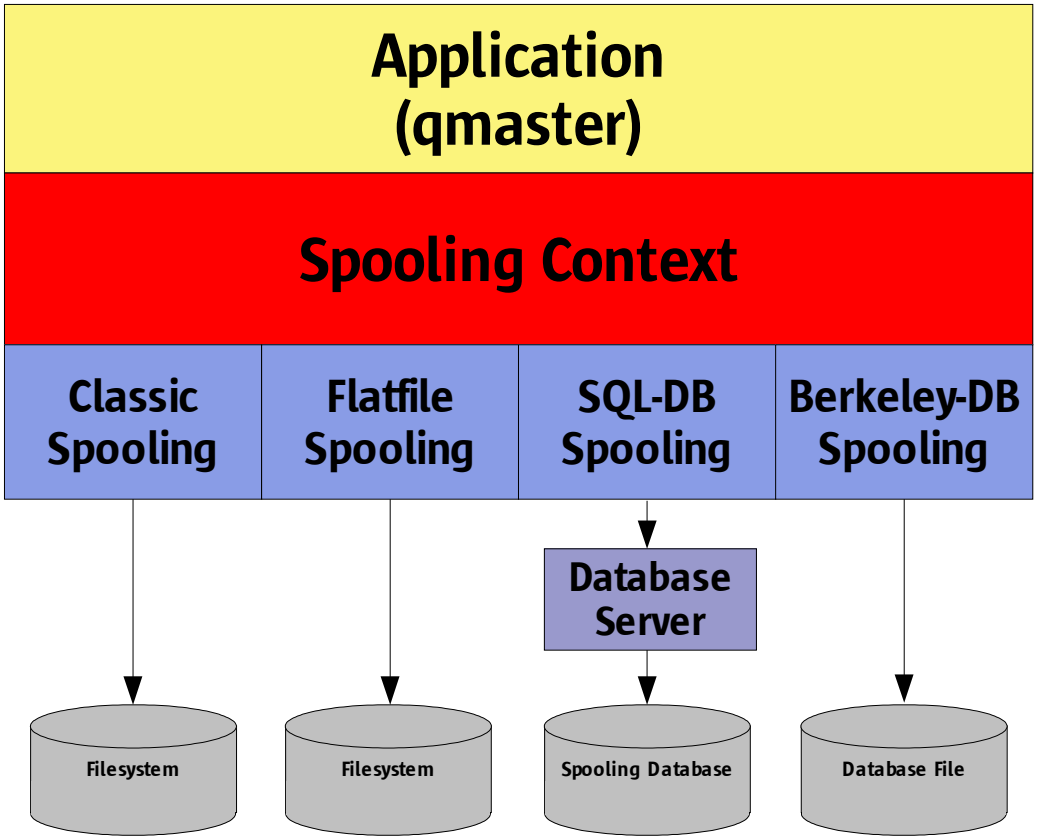
- Spooling Database
  - replace current flat file spooling
  - reflects qmaster's internal data structures
- Accounting and Reporting Database
  - no impact on core system
  - structure suited for report generation
  - precalculated values

Database in the core system

- Spooling Framework
  - abstraction layer
- Multiple spooling methods
  - classic
  - new flat file
  - SQL database (PostgreSQL)
  - BerkeleyDB

The spooling framework

- abstraction layer (spooling context)
- extendable (add shared libs)
- combine spooling methods



Classic spooling

- pros
  - needn't install database server
  - files can be read and edited (besides jobs)
- cons
  - slow, esp. on NFS mounted filesystems
  - difficult to analyze / generate reports
  - hardcoded functions for each data type
  - very limited transaction handling
  - no consistency guarantied

New Flat File Spooling

- pros
  - needn't install database server
  - files can be read and edited (besides jobs)
  - more flexible than classic spooling (generic spooling functions)
- cons
  - slow, esp. on NFS mounted filesystems
  - difficult to analyze / generate reports
  - very limited transaction handling
  - no consistency guarantied

SQL Database

- pros
  - fast
  - ACID
  - easy to access with standard tools
  - generic spooling functions
  - open for future development (replication, distributed database ...)
- cons
  - have to install and maintain a database server
  - requires tuning expertise

Berkeley DB

- pros
  - very fast
  - ACID
  - needn't install database server
  - flexible spooling functions
  - open for future development (replication ...)
- cons
  - difficult data access (requires maintenance client)
  - not supported by standard analysis tools

Characteristics of Berkeley DB

- embedded database library
- compact
- high performance
- transaction protected
- simple function call api
- many supported platforms
- commonly available in standard UNIX distributions

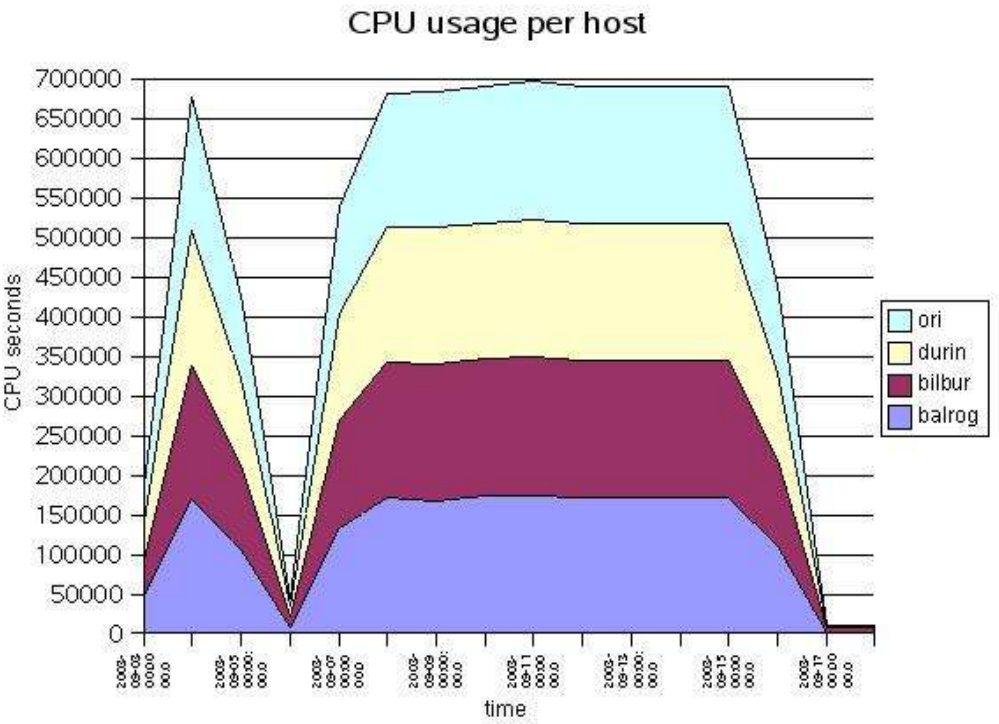
Future plans

- spooling methods (LDAP for user management)
- clients access database instead of querying qmaster
- Future development based on Berkeley DB
  - use replication subsystem
  - use transaction subsystem in SGE components

Accounting and reporting DB

Filter

Sum - cpu	host				
day	balrog	bilbur	durin	ori	**Total Result**
2003-09-03	46487.53	45854.62	46582.15	45762.13	**184686.43**
2003-09-04	169344	169285	169431	169371	**677431**
2003-09-05	105769	107935	107849	106100	**427653**
2003-09-06	8793	13375	12768	9029	**43965**
2003-09-07	134017	135485	135278	134412	**539192**
2003-09-08	171889	170619	169914	168821	**681243**
2003-09-09	168337	171333	172857	171024	**683551**
2003-09-10	174332	172219	170728	174600	**691879**
2003-09-11	174864	175626	172932	175020	**698442**
2003-09-12	172811	172813	172824	172793	**691241**
2003-09-13	172814	172808	172824	172792	**691238**
2003-09-14	172813	172808	172831	172794	**691246**
2003-09-15	172822	172808	172827	172795	**691252**
2003-09-16	109934	110070	109984	110036	**440024**
2003-09-17	101	7200	3601	0	**10902**
2003-09-18	0	7200	3600	0	**10800**
Total Resul	**1955127.53**	**1977438.62**	**1966830.15**	**1955349.13**	**7854745.43**

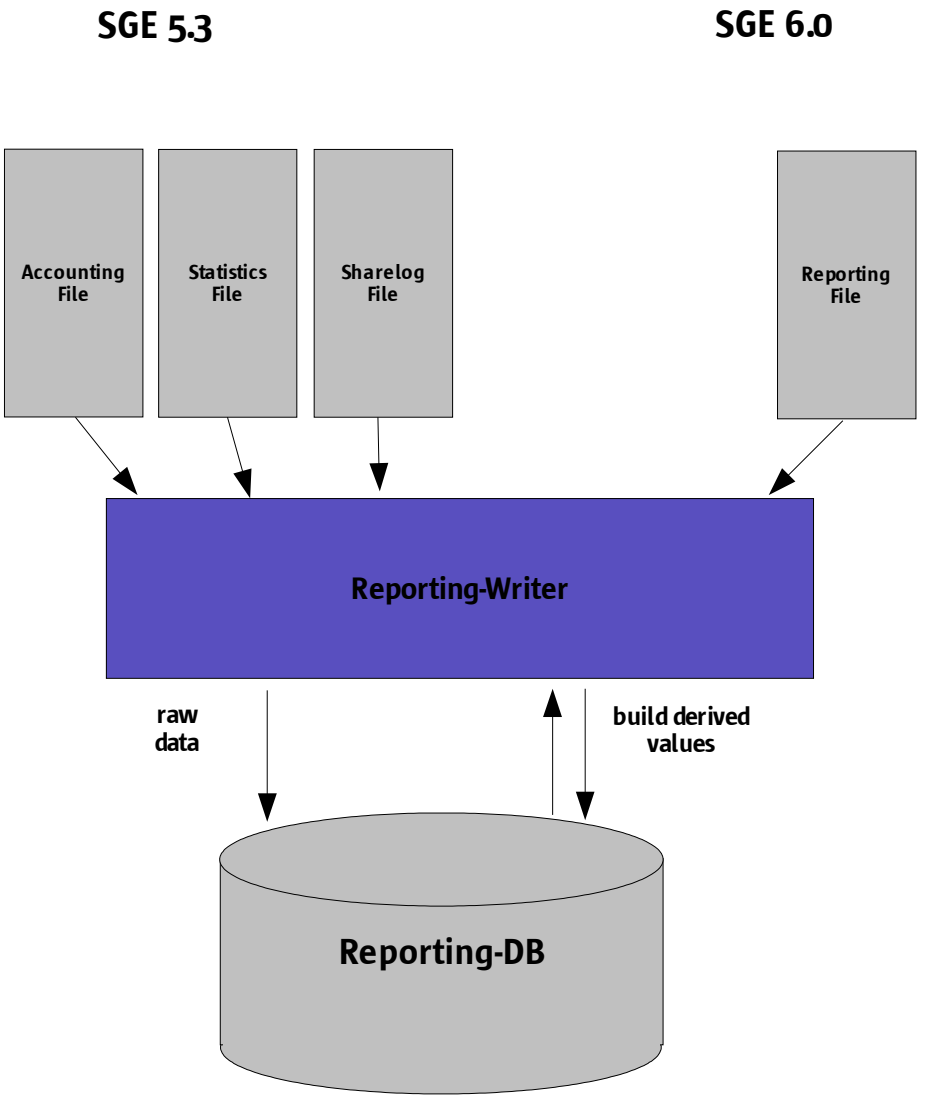


Why two separated databases?

- standardized access (SQL, ODBC, JDBC)
- simpler, easier to use database structure
- historical data (not needed in core system)
- preprocessed data (sums, averages ...)
- queries don't affect core system
- lower requirements on availability

Architecture

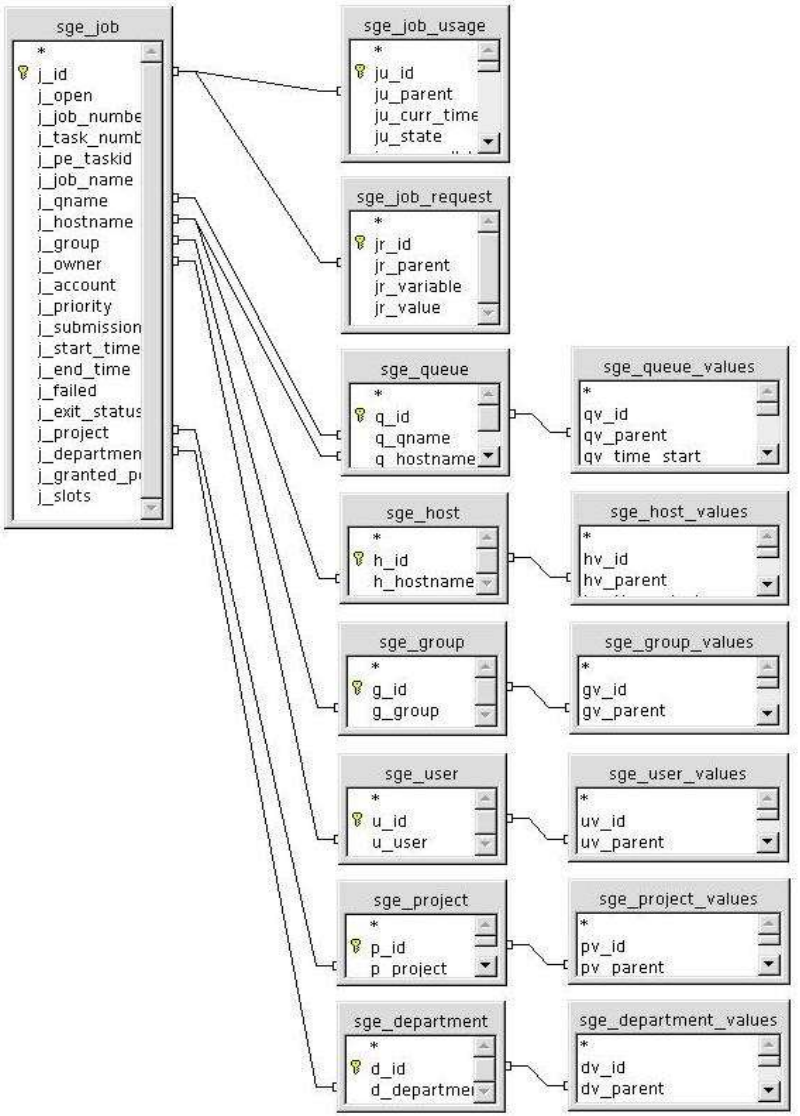
- Java application
- Database access through JDBC
- loosely coupled to the SGE system



Supported Databases

- PostgreSQL
- MySQL
- Oracle
- “any relational Database providing JDBC interface and standard SQL”

Database Schema



Stored Data

- Job related information
  times, user, project, exit status ...
- Host and queue related information
  load information, consumables ...
- Sharetree
  configured shares, actual shares ...
- Precomputed, derived values
  sums, averages per host, queue, user, project ...
- Daemon statistics

Metrics (examples)

- Average (min, max) job wait time
- Number of jobs completed
- Total (avg, max) cpu time consumed
- Total (avg, min, max) job runtime
- Utilization of slots, cpus, hosts, queues
- Cluster availability, outage times
- Share utilization
- Memory abuse

Summary

- Improve scalability and performance of the core system through integration of Berkeley DB
- Provide easy access to a reporting and analysis facility through use of a standard SQL database

Joachim Gabler

joga@sun.com

http://sun.com/grid

Sun microsystems

We make the net work.