Multi-Threading Approaches in Sun Grid Engine 6.0

Andreas Doerr

Software Engineering

Sun Microsystems Inc

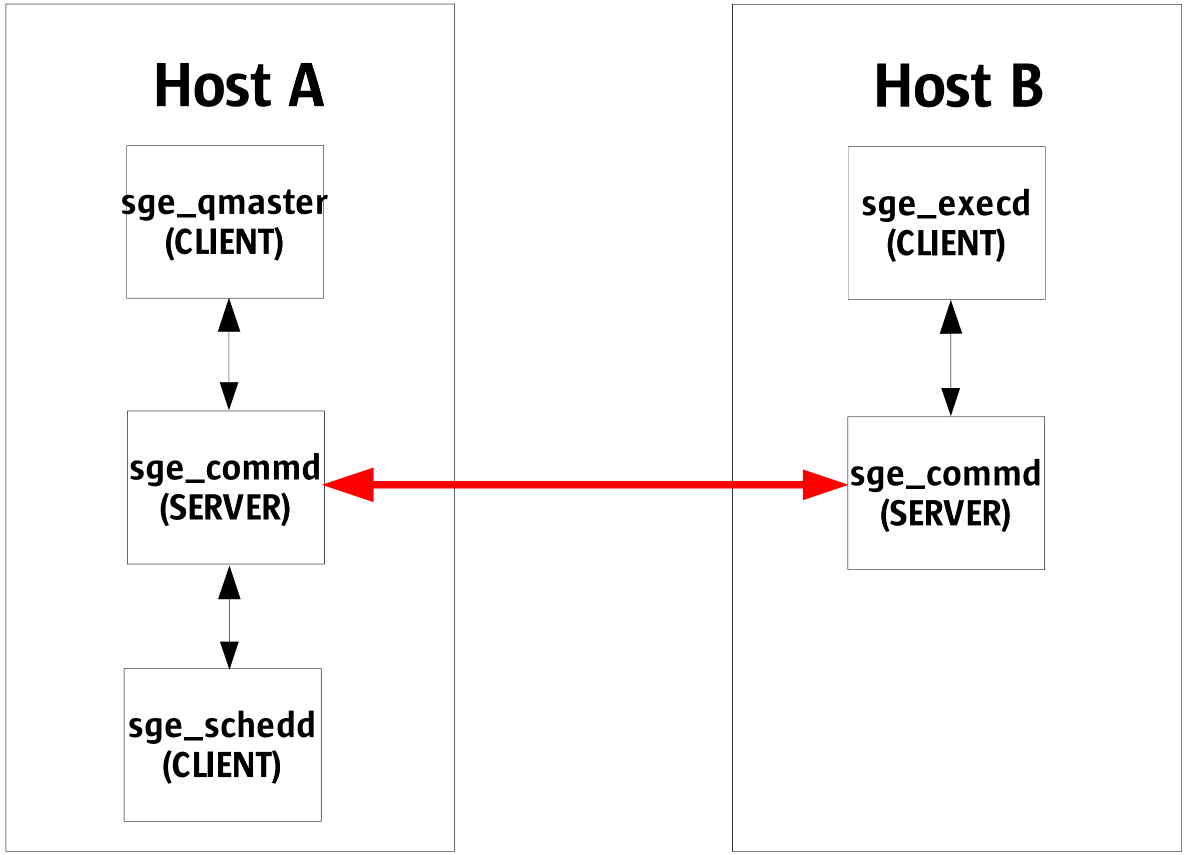
What is it all about?

- Throughput
- Interoperability
- Scalability

Topics

- New communication system and multi-threading
- New message format
- sge_qmaster and multi-threading

Communication Status Quo



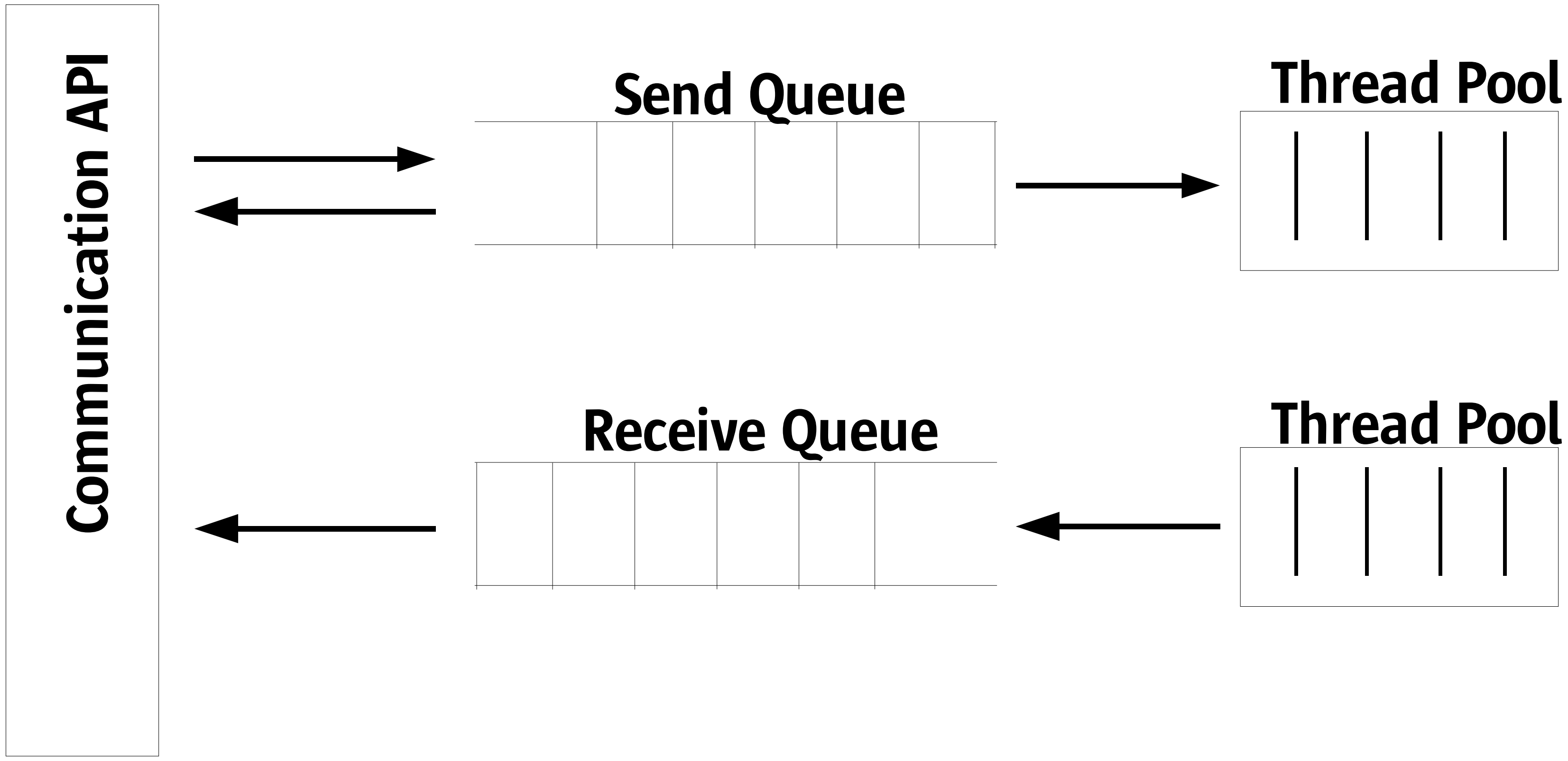
Goals

- Utilize all available CPU's
- Handle message “burstiness”
- Become independent of network protocol

Major Changes

- Eliminate sge_commd
- Utilize threads and message queues
- Abstraction layer to hide network protocol

New Communication System



Communication API

- Blocking send
- Nonblocking send
- Send and receive acknowledgment
- Blocking receive

Topics

- New communication system and multi-threading
- New message format
- sge_qmaster and multi-threading

Current Message Format

- Language dependent
- No explicit message descriptions
- Allows only binary messages

New Message Format

- Each message consists of a common XML-Header and a payload
- Header allows to determine
  - Message type
  - Data format (Binary or XML)
  - Routing information
  - Version
  - Message length

Possible future directions

- Utilize P2P technologies like JXTA
- Additional message delivery guarantees like
  - At most once
  - Exactly once

Topics

- New communication system and multi-threading
- New message format
- sge_qmaster and multi-threading

sge_qmaster Status Quo

- Single-Threaded, iterative Server

```
while (TRUE)
{
    update heartbeat

    deliver events

    receive request

    handle request

    send response
}
```

Goals

- Become a concurrent server
- Take advantage of common SGE usage patterns

Major Changes

- Update-Thread for periodic tasks
- One or more Request-Threads
- Locking-API

Locking Subsystem

- Locking-API hides which lock implementation is used
  - e.g. pthread_mutex_t
- Allows different locking schemes
  - exclusive
  - multiple-reader / single-writer
- Allows additional lock modes
  - e.g. intention locks

Locking API

- sge_lock() / sge_unlock()
- Locktype
  - Master Job List, Event Client List ...
- Lockmode
  - Read, Write (includes Read)
- Locker Id

Restrictions

- Number of threads is limited
- Degree of concurrency depends on type of operation
- No single, best locking scheme
  - For example multiple-reader/single-writer

    Read precedence?

    Write precedence?

Possible Future Directions

- Increased lock granularity
- Distributed sge_qmaster
  - Replicated data
  - Partition requests among nodes

What is it all about?

- Throughput -> New communication System
- Interoperability -> New message format
- Scalability -> Multi-threaded sge_qmaster

Andreas Doerr

Andreas.Doerr@Sun.COM

Sun microsystems

We make the net work.