Gearman

pgDay San Jose

July 19, 2009

Eric Day – Sun Microsystems
http://oddments.org/

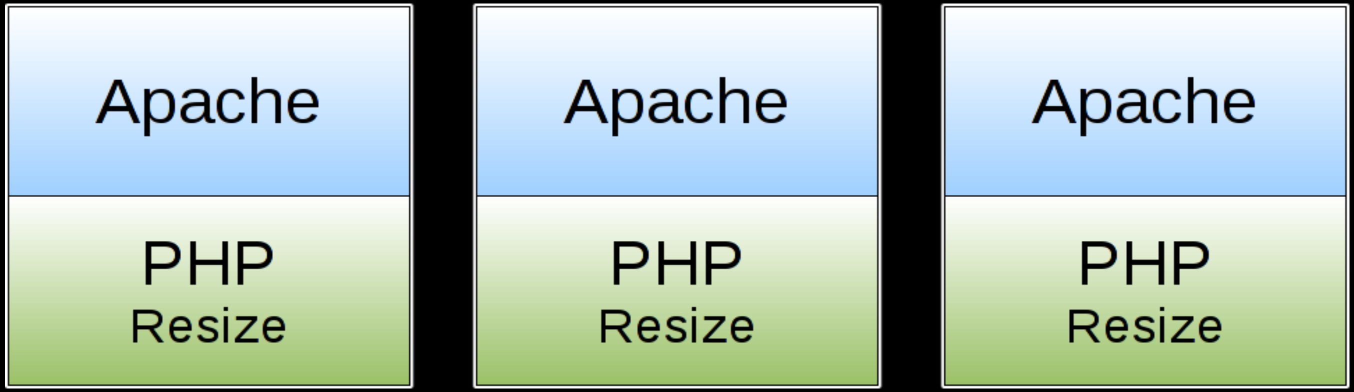
Brian Aker – Sun Microsystems
http://krow.net/

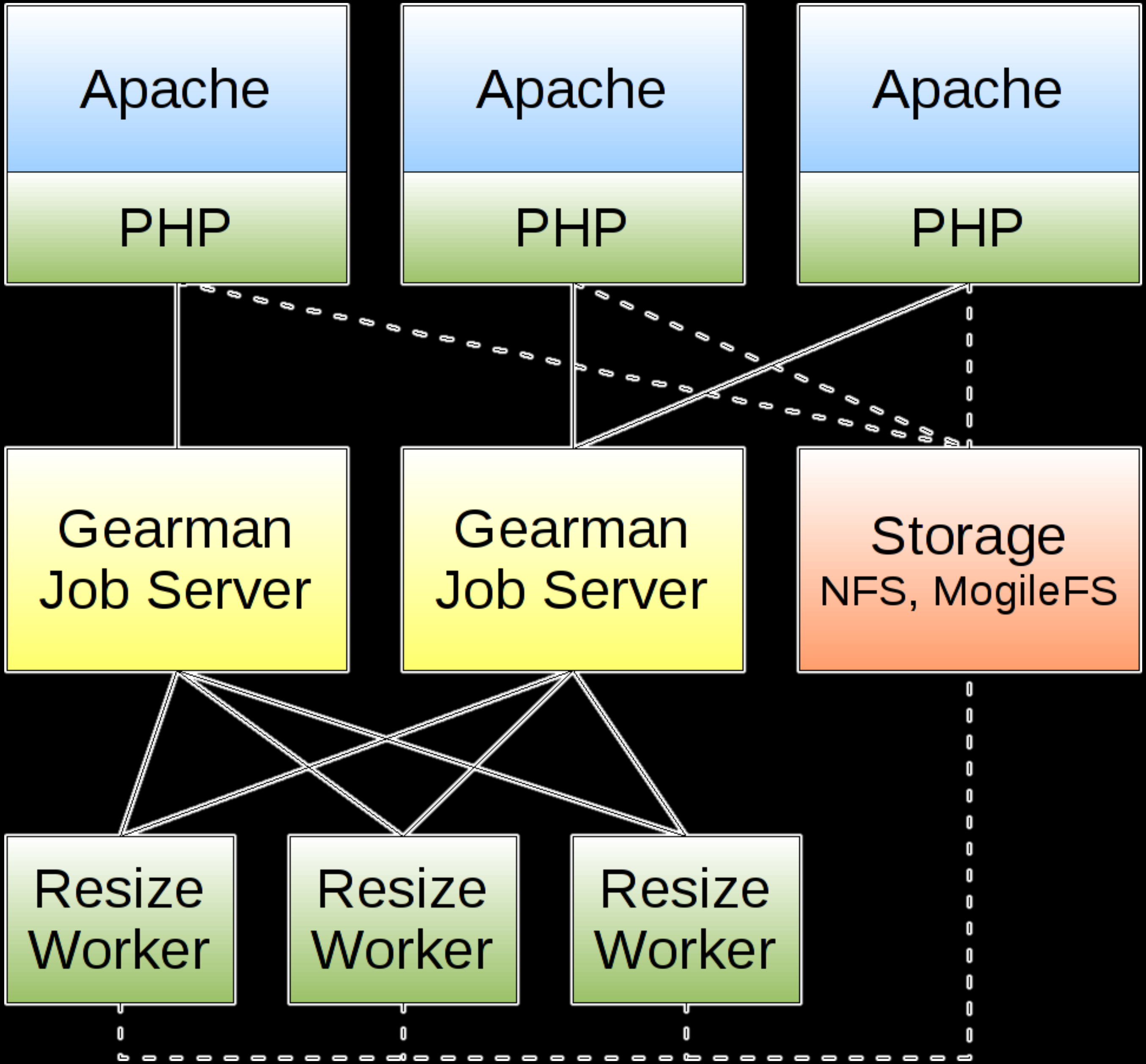
Outline

- History
- Basics
- Example
- PostgreSQL
- Applications
  - Asynchronous Queues
  - Map/Reduce
  - Log Analysis
- Roadmap

Kittens!

(LiveJournal.com Image Processing)


Apache
PHP
Resize
Apache
PHP
Resize
Apache
PHP
Resize


Apache
PHP
Apache
PHP
Apache
PHP
Gearman Job Server
Gearman Job Server
Storage
NFS, MogileFS
Resize Worker
Resize Worker
Resize Worker

“The way I like to think of Gearman is as a massively distributed, massively fault tolerant fork mechanism.”

- Joe Stump, Digg

History

- Danga – Brad Fitzpatrick & company
  - Related to memcached, MogileFS, ...
- Anagram for “manager”
  - Gearman, like managers, assign the tasks but do none of the real work themselves
- Digg: 45+ servers, 400K jobs/day
- Yahoo: 60+ servers, 6M jobs/day
- LiveJournal, SixApart, DealNews, xing.com, Threadless, ...

Recent Development

- Rewrite in C
- APIs
  - PHP, Perl, Java, Python, Ruby, PostgreSQL, Drizzle, MySQL
- Command line tool
- Protocol additions
- Multi-threaded (50k jobs/second)
- Persistent queues
- Pluggable protocol

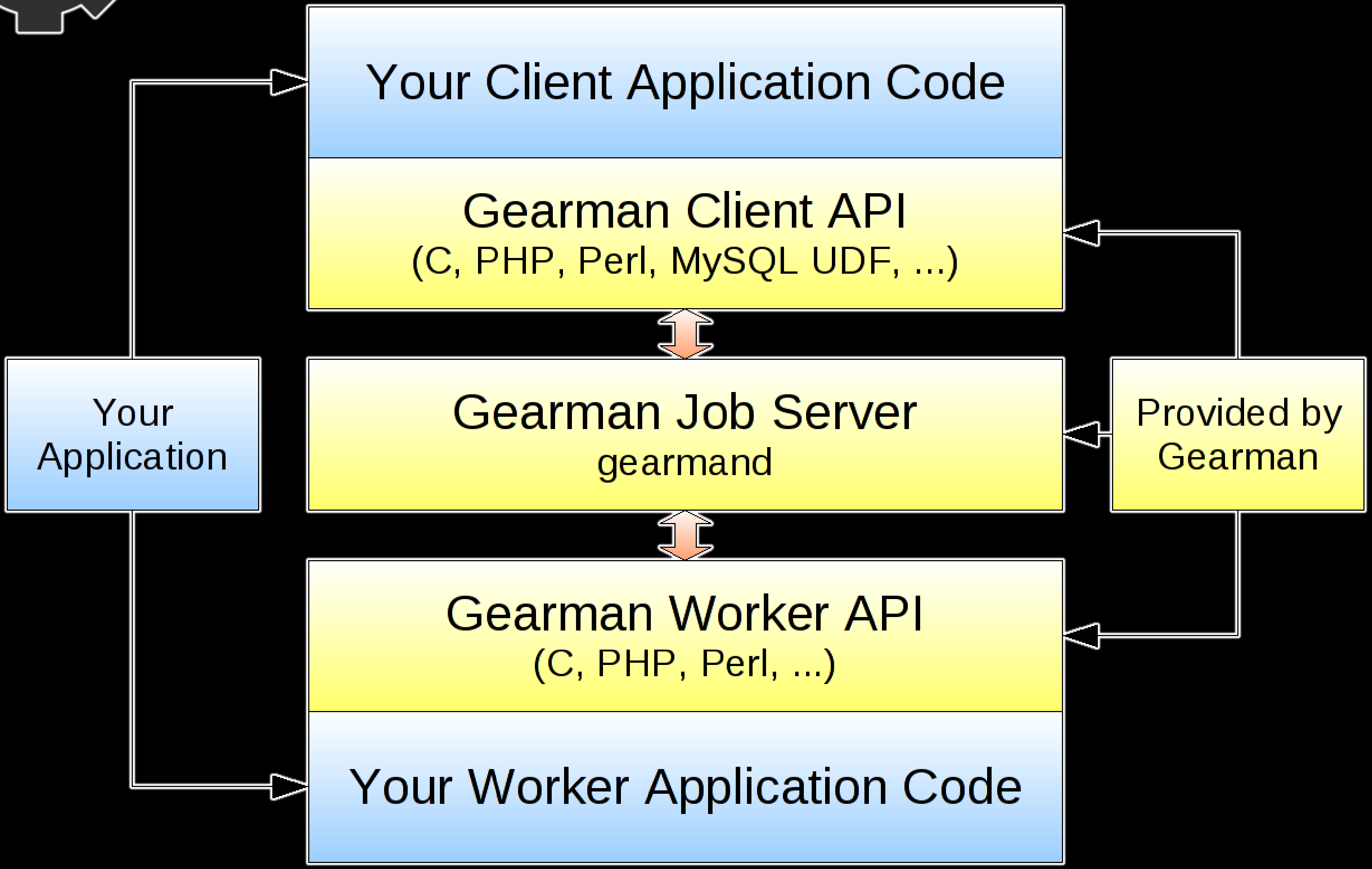
Features

- Open Source
- Simple & Fast
- Multi-language
  - Mix clients and workers from different APIs
- Flexible Application Design
  - Not restricted to a single distributed model
- Embeddable
  - Small & lightweight for applications of all sizes
- No Single Point of Failure

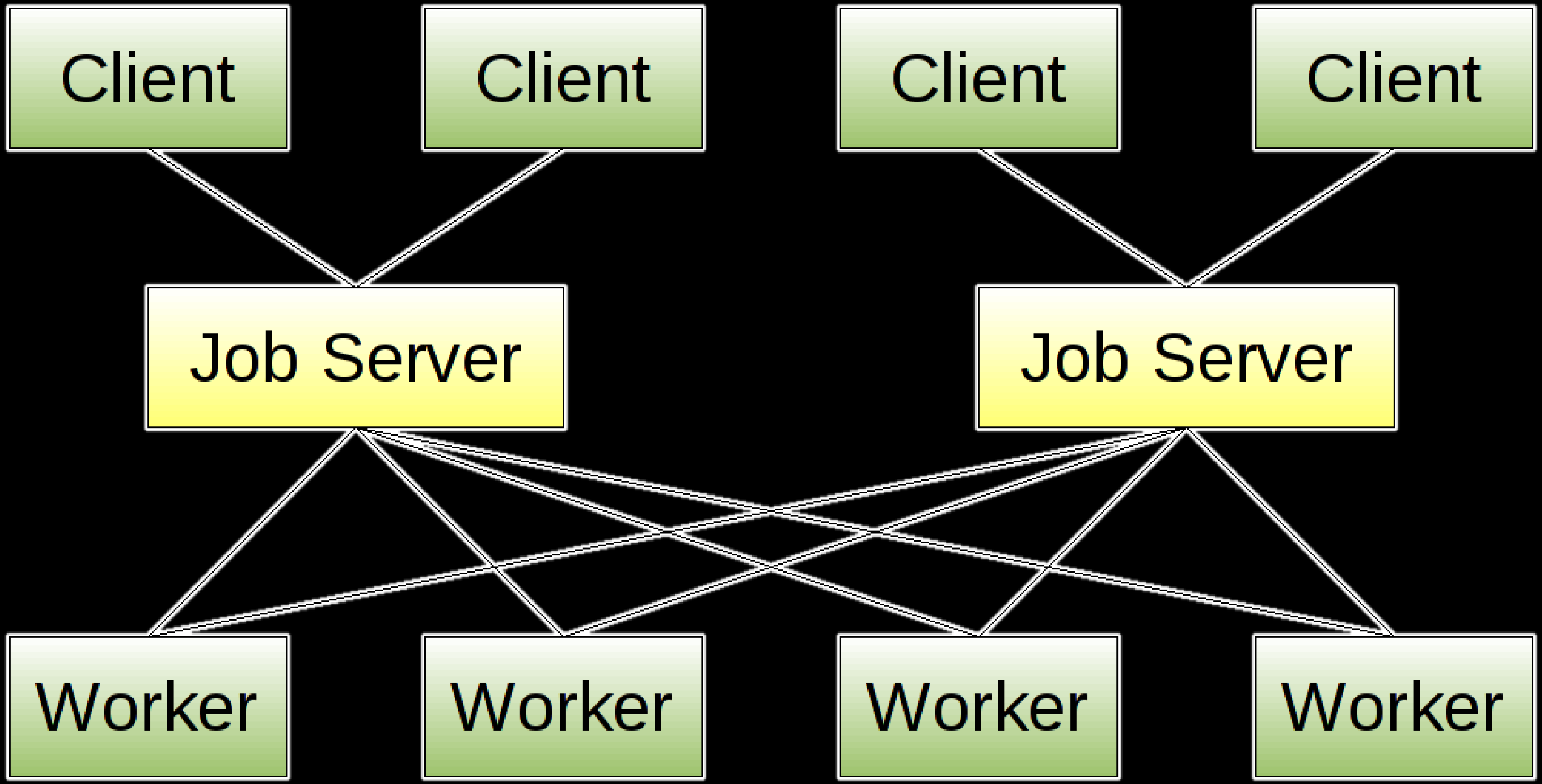
Basics

- Gearman provides a distributed application framework
- Uses TCP port 4730 (was port 7003)
- **Client** – Create jobs to be run and send them to a job server
- **Worker** – Register with a job server and grab jobs to run
- **Job Server** – Coordinate the assignment from clients to workers, handle restarts

Gearman Stack



No Single Point of Failure



Hello World

```
$client= new GearmanClient();
$client->addServer();
print $client->do("reverse", "Hello World!");
```

```
$worker= new GearmanWorker();
$worker->addServer();
$worker->addFunction("reverse", "my_reverse_function");
while ($worker->work());

function my_reverse_function($job)
{
  return strrev($job->workload());
}
```

Hello World

```
shell$ gearmand -d

shell$ php worker.php &
[1] 17510

shell$ php client.php
!dlroW olleH
```

How Is This Useful?

- Provides a distributed nervous system
- Natural load balancing
  - Workers are notified and ask for work, not forced
- Multi-language integration
- Distribute processing
  - Possibly closer to data
- Synchronous and asynchronous queues

Back to the Kittens

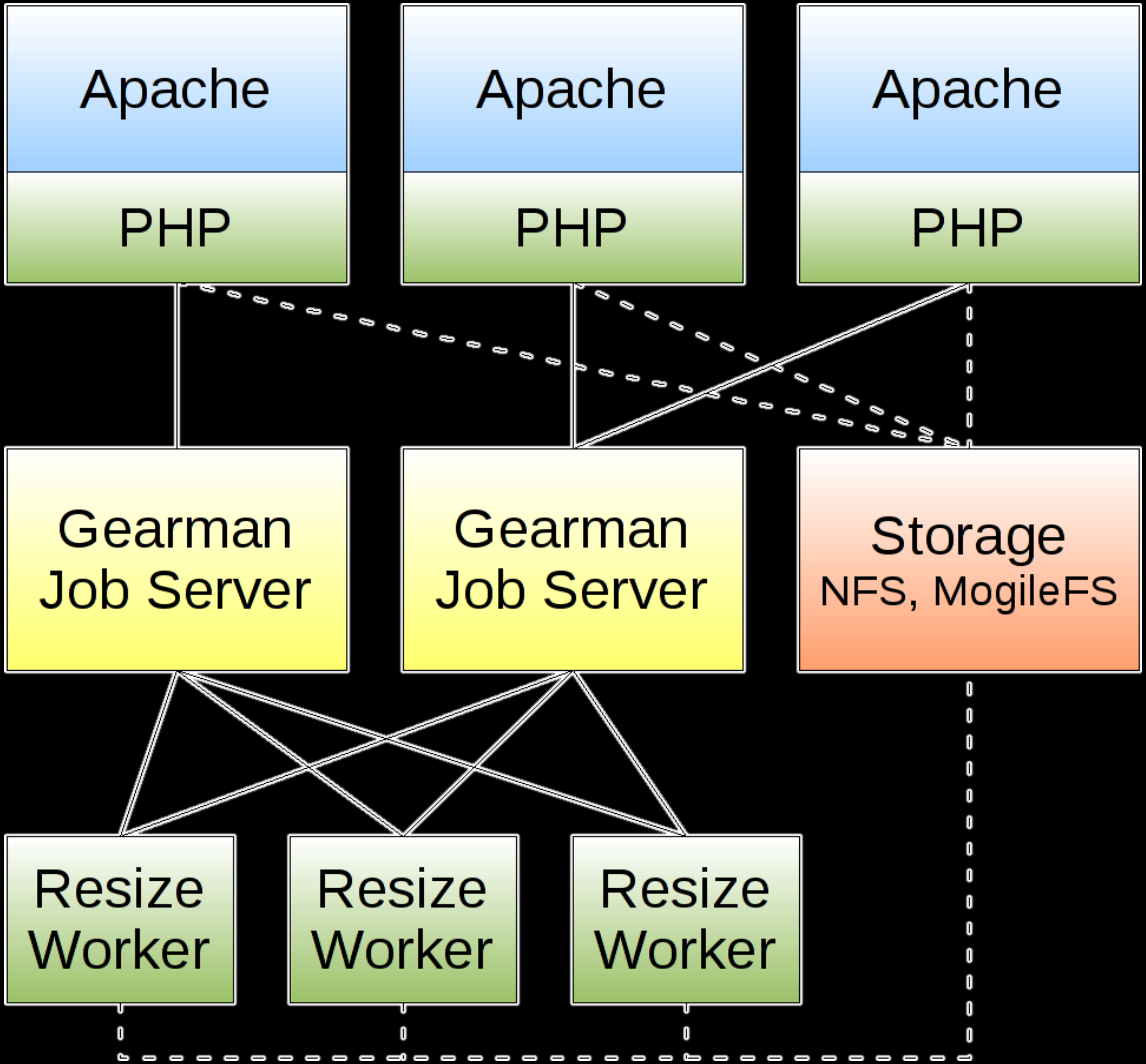

Apache
PHP
Apache
PHP
Apache
PHP
Gearman Job Server
Gearman Job Server
Storage
NFS, MogileFS
Resize Worker
Resize Worker
Resize Worker

Image Resize Worker

```
$worker= new GearmanWorker();
$worker->addServer();
$worker->addFunction("resize", "my_resize_function");
while ($worker->work());

function my_resize_function($job)
{
  $thumb = new Imagick();
  $thumb->readImageBlob($job->workload());
  $thumb->scaleImage(200, 150);
  return $thumb->getImageBlob();
}
```

Image Resize Worker

```
shell$ gearmand -d

shell$ php resize.php &
[1] 17524

shell$ gearman -f resize < large.jpg > thumb.jpg

shell$ ls -sh large.jpg thumb.jpg
3.0M large.jpg   32K thumb.jpg
```

PostgreSQL

libpq Queue

- Job server queue in memory
- Optional persistent queue for background jobs
- libdrizzle, libmemcached, SQLite, ...
- libpq!
  - Add to queue
  - Flush queue (batching coming soon)
  - Delete from queue
  - Replay queue (on startup)
- Gearman queue is as durable as the database

libpq Queue Startup

```
shell$ gearmand -vvv -q libpq \
       --libpq-conninfo="dbname = test host = 127.0.0.1"
 INFO Initializing libpq module
 INFO libpq module creating table 'queue'
 INFO Starting up
 ...
 INFO libpq replay start
 ...
 INFO Entering main event loop
```

libpq Queue Add

```
shell$ gearman -b -f test Hello
```

```
...
DEBUG libpq add: 69fc531d-96fb-45e4-ae05-6c36624cf2a6
DEBUG libpq flush
...
```

```
test=# \x
Expanded display is on.
test=# SELECT * FROM queue;
-[ RECORD 1 ]-+-------------------------------------
unique_key    | 69fc531d-96fb-45e4-ae05-6c36624cf2a6
function_name | test
priority      | 1
data          | Hello
```

libpq Queue Done

```
shell$ gearman -w -f test
```

```
...
DEBUG libpq done: 69fc531d-96fb-45e4-ae05-6c36624cf2a6
...
```

```
test=# SELECT * FROM queue;
(No rows)
```

pgGearman

- Gearman client functions in SQL
- Submit foreground and background jobs
- Real function is now a Gearman worker
  - Command line tool
  - Shell script
  - Another language
  - Remote machine

pgGearman

- https://launchpad.net/pggearman

```
shell$ make
shell$ make install
shell$ psql <database>
[database]=# \i /usr/local/share/contrib/pggearman.sql
[database]=# gman_servers_set('localhost');
```

pgGearman

- Startup gearmand and worker

```
shell$ gearmand -d
shell$ gearman -w -f test -– wc -w
```

```
shell$ psql <database>
[database]=# gman_servers_set('localhost');
[database]=# select gman_do('test', 'Hello World!');
 gman_do
---------
 2

(1 row)
```

pgGearman

- Run background jobs
- Return immediately, don't wait for worker

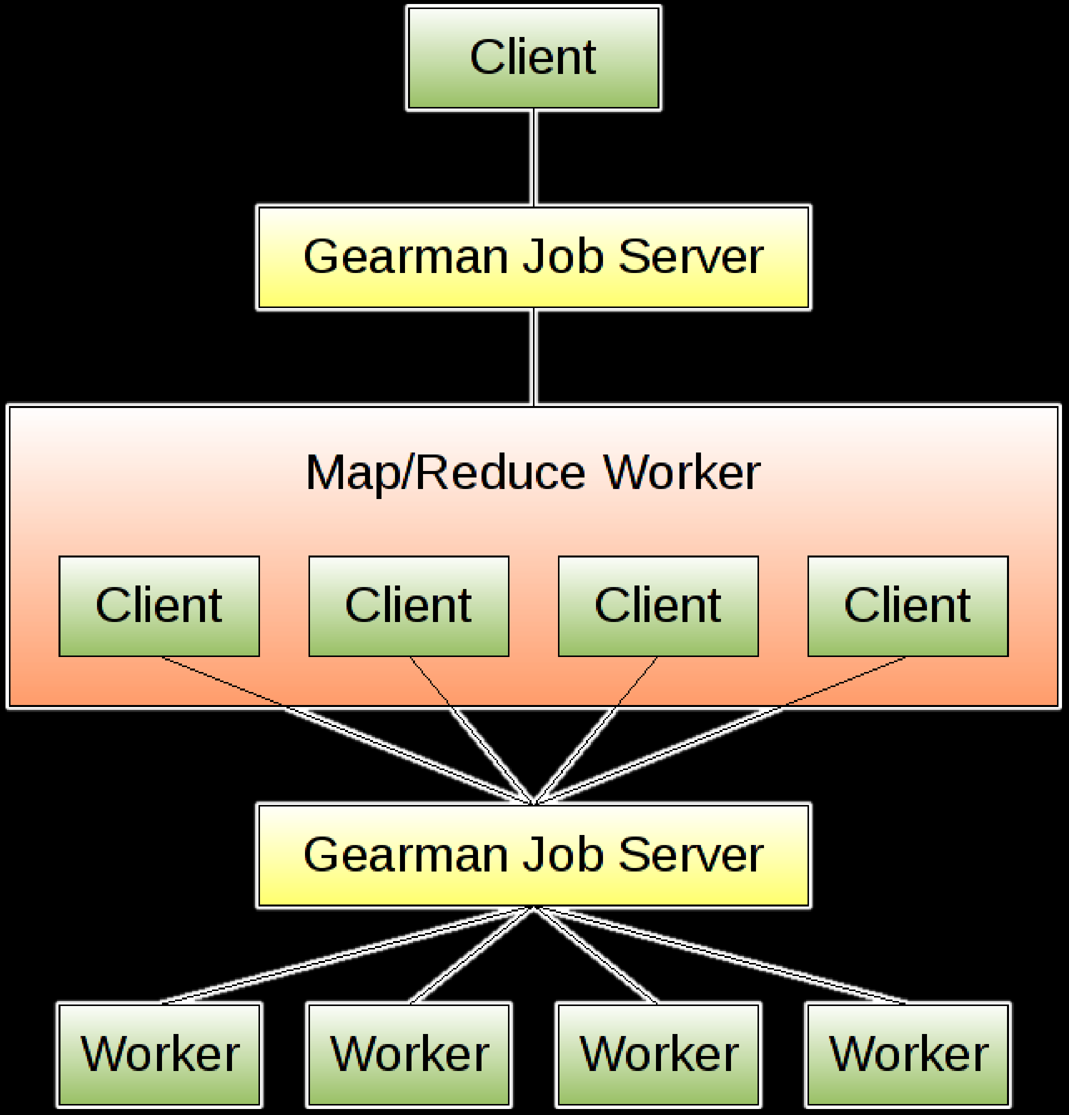
```
shell$ psql <database>
[database]=# gman_servers_set('localhost');
[database]=# SELECT gman_do_background('world_peace',
[database](#                           'forty-two');
 gman_do_background
--------------------
 H:lap:8
(1 row)
```

Applications

Asynchronous Queues

- They help you scale
- Not everything needs immediate action
  - E-Mail notifications
  - Tweets
  - Certain types of database updates
  - RSS aggregation
  - Search indexing
- Allows for batch operations

Map/Reduce


Client
Gearman Job Server
Map/Reduce Worker
Client
Client
Client
Client
Gearman Job Server
Worker
Worker
Worker
Worker

Map/Reduce

- Can be multi-tier
- Fan-out depends on application
- Don't need to follow strict MR model
  - Not Google API (paper) or Hadoop
  - Can be ad-hoc
- Use any Gearman API (mix/match)

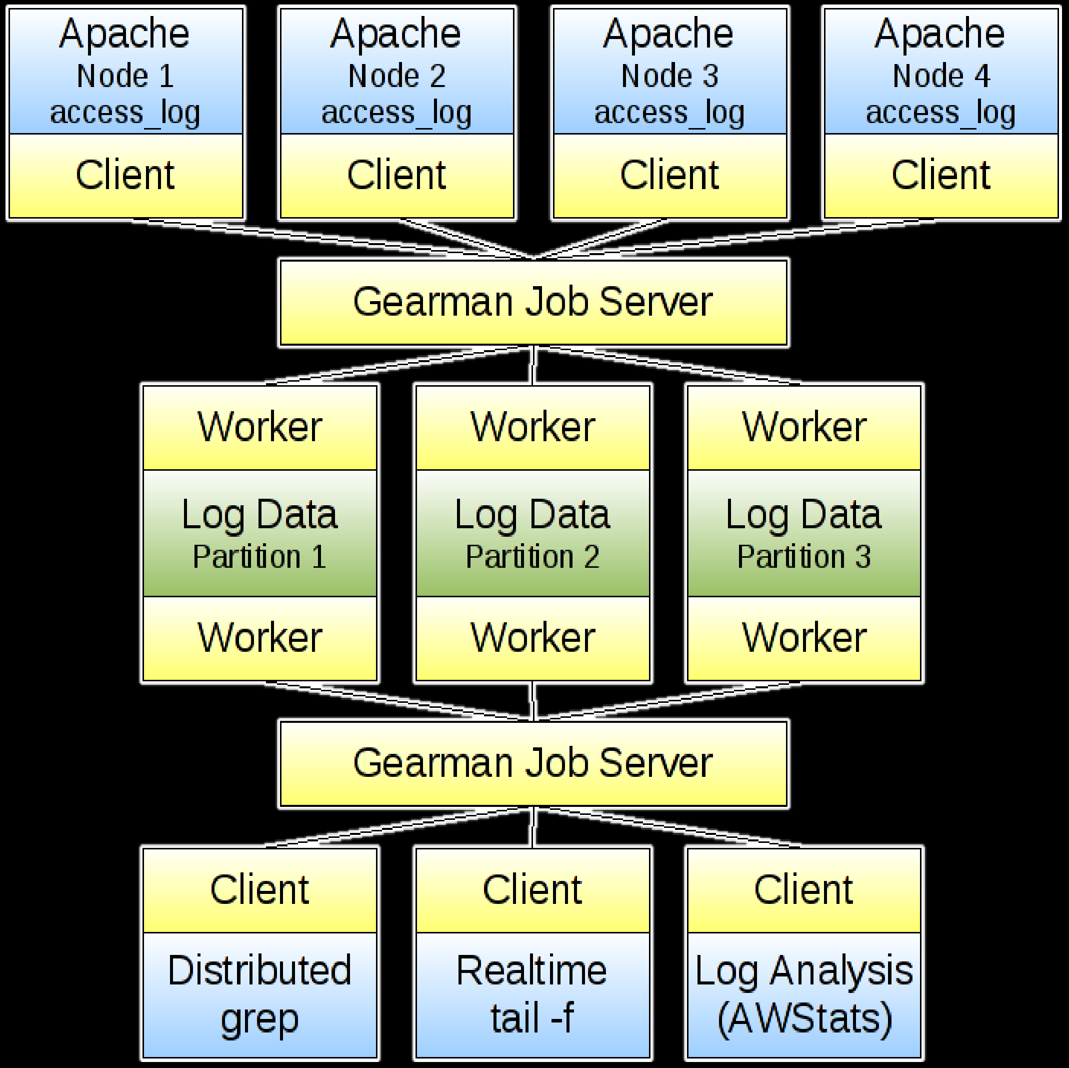
Log Processing

- Bring Map/Reduce to Apache logs
- First, get log storage off Apache nodes
- Distribute processing to log storage nodes
- Combine data in some meaningful way
  - Summary
  - Distributed merge-sort algorithms

Log Processing

- Collection
  - tail -f access_log | gearman -n -f logger
  - CustomLog "| gearman -n -f logger" common
  - Write a Gearman Apache logging module
- Processing
  - Distributed/parallel grep
  - Log Analysis (AWStats, Webalizer, ...)
  - Custom data mining & click analysis

Log Processing



Other Applications

- MogileFS
- Distributed e-mail storage
- Gearman Monitor Project
  - Configuration management (elastic)
  - Statistics gathering
  - Monitoring
  - Modular (integrate existing tools)
- What will you build?

What's Next?

- More protocol and ~~queue modules~~
- TLS, SASL, multi-tenancy
- Replication/subscription/job relay
- Job result cache (think memcached)
- Improved statistics gathering and reporting
- Event notification hooks
- Monitor service

Get involved

- http://gearman.org/
- #gearman on irc.freenode.net
- http://groups.google.com/group/gearman
- Gearman @ OSCON
  - 3 Hour tutorial - Tuesday
  - 45 Minute Session (similar material) - Wednesday
  - Birds of a Feather – Wednesday @ 7pm
  - Expo Hall Booth