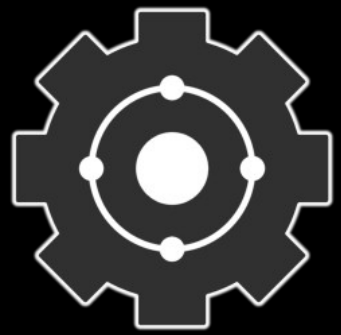
Drizzle



Gearman

Eric Day - http://www.oddments.org/

Patrick Galbraith - http://patg.net/

Drizzle Overview

- Vision
- Community
- Microkernel Architecture
- Modules
  - Storage Engines
  - Protocol
  - Replication
- Roadmap

MySQL CAB '05

Rethink Everything

(but do not assume everything was bad)

Do not play catch-up.
Leap Forward.

Vision

- Announced at OSCON 2008
- Microkernel Architecture
- Infrastructure Aware
- Focus on Web Applications
- Modernize Codebase (C++, STL, OSS libs)
- Multi-Core/Concurrency
- 64-bit, lots of RAM
- UTF-8

Community

- Open Source
- All contributions treated equally
- No contributor license agreements
- Captain system
- All project information is public
- Release early and often (~2 weeks)
- 100+ Contributors
- 500+ On mailing list

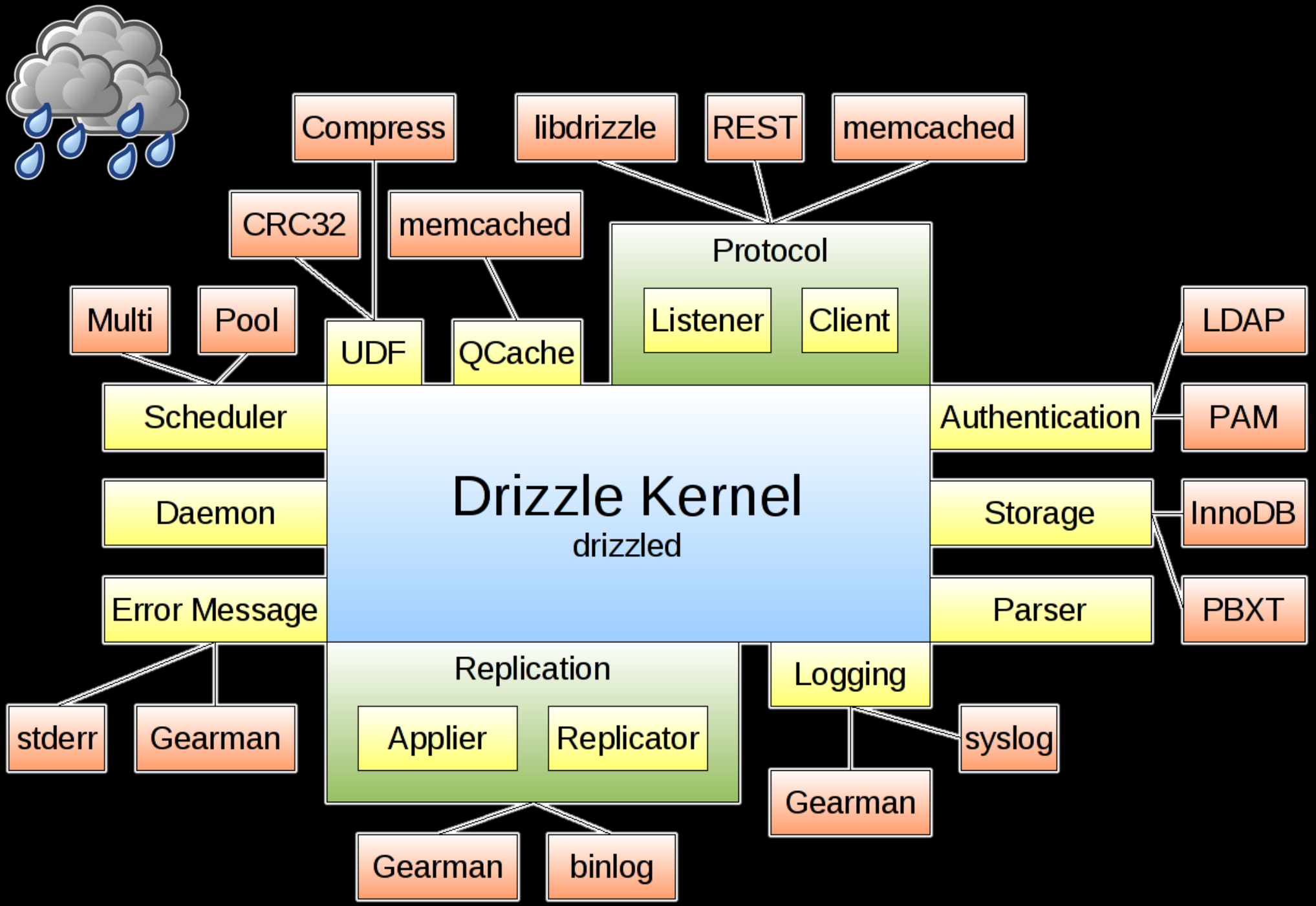
Sun Team Values

- Have open and well-documented interfaces
- Have transparent goals and processes that are communicated publicly
- Have fun and encourage collaboration
- Remove barriers to contribution and participation for anyone
- Enable contributors to build a business around Drizzle

Community Tools

- Launchpad
  - Bugs
  - Blueprints
  - Translations (30+ Languages)
- BZR
- Buildbot
- Hudson

Microkernel Architecture

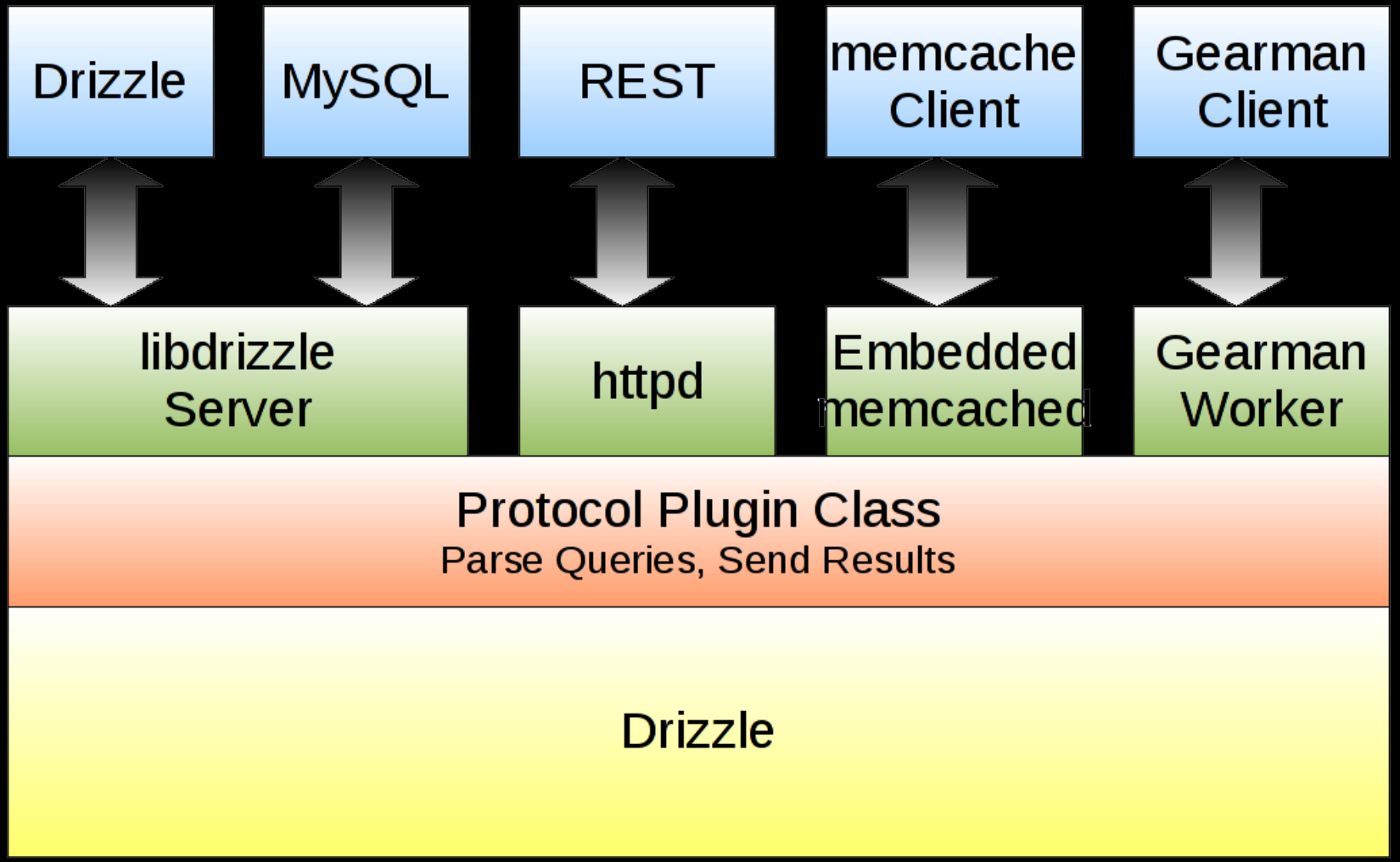

Compress
libdrizzle
REST
memcached
CRC32
memcached
Protocol
Multi
Pool
UDF
QCache
Listener
Client
LDAP
Scheduler
Authentication
PAM
Drizzle Kernel
drizzled
Daemon
Storage
InnoDB
Error Message
Parser
PBXT
Replication
Logging
stderr
Gearman
Applier
Replicator
syslog
Gearman
Gearman
binlog

Storage Engines

- Still have multiple storage engines
- Default to ACID compliant engine
  - InnoDB (we have a PBXT tree too)
- Interface updates
- Let engines supply their own metadata
- MyISAM now temp-table only
  - May be removed at some point

New Protocol

- Asynchronous, Full Duplex
  - Concurrent queries on one connection
- Remove weak attack methods
- Built-in sharding
- Optional checksums
- Pluggable in server


Drizzle
MySQL
REST
memcache Client
Gearman Client
libdrizzle Server
httpd
Embedded memcached
Gearman Worker
Protocol Plugin Class
Parse Queries, Send Results
Drizzle

libdrizzle

- https://launchpad.net/libdrizzle
- BSD License
- Supports both Drizzle and MySQL Protocols
- Provides client and server interfaces
  - SQLite server example
- PHP, Perl, Python, and Ruby extensions
- Concurrent query API
  - Reduce page load times

Replication

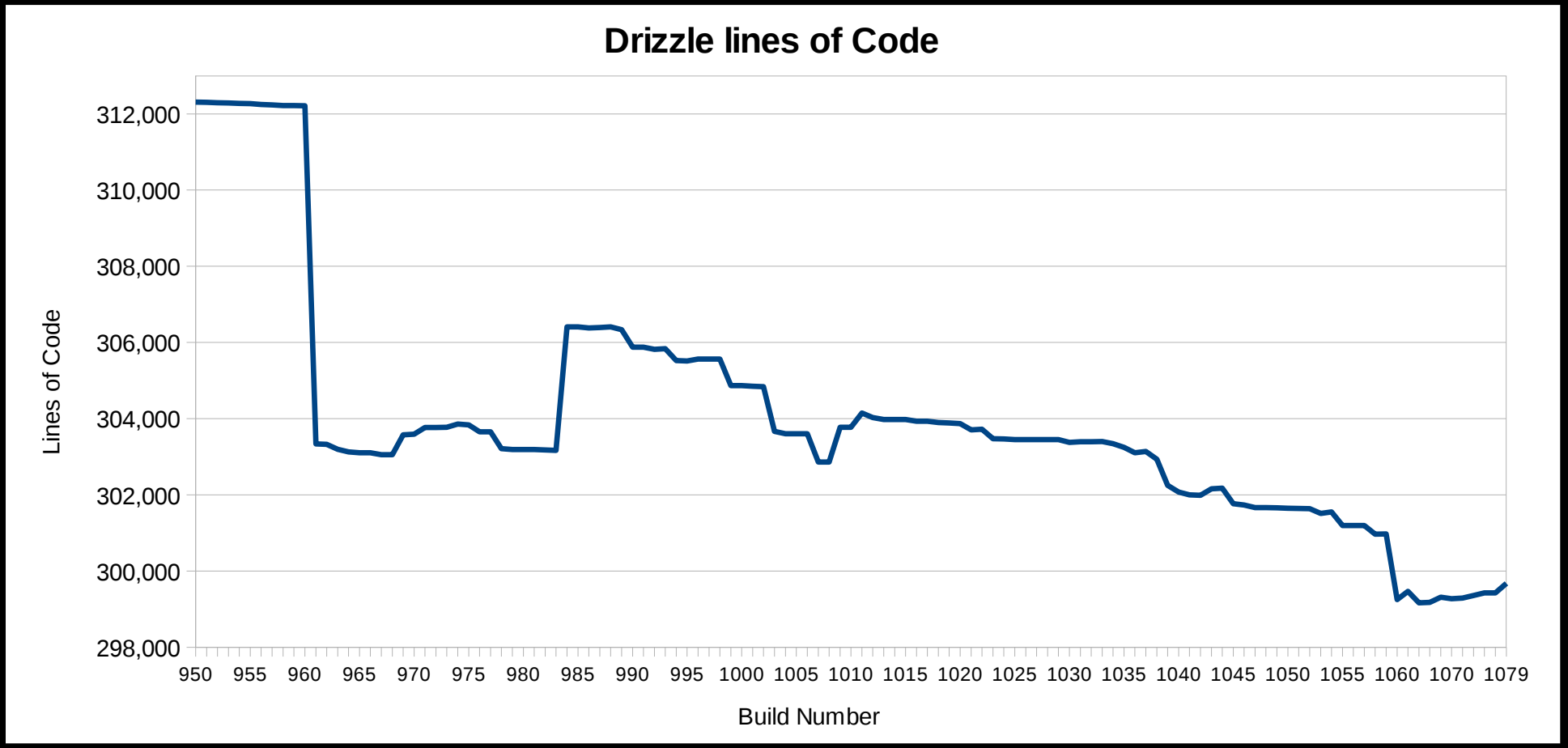
- New API for events
- Google Protobuffer messages
  - Easy to read in any language
- File (binlog) and Gearman plugins underway
- Easy to write your own sender/applier

Other Plugins

- Logging
  - Gearman, syslog, query analyzer
- Authentication now modular and optional
  - PAM/LDAP, HTTP Auth
- Multi-language support in development
  - Stored procedures (server-side scripting)
  - Plugins from other languages
- Information/Performance Schema
  - No materialization

Where are we now?

- MySQL 6.0.5-alpha: 1,128,112 LoC



Aloha Milestone

- Replication
- New Protocol
- Architectural Cleanup and Performance
- Table Discovery
- New Information Schema
- Plugin Dependency Checking
- Pluggable Configuration

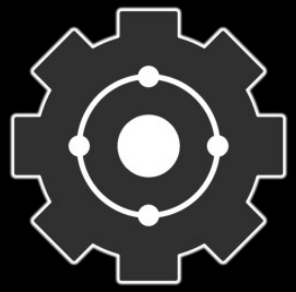
Bell Milestone

- Server Side Scripting
- Performance Schema
- <insert your work> :)

Get Involved!

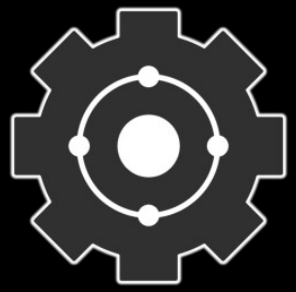
- http://drizzle.org/
- https://launchpad.net/drizzle
- #drizzle on irc.freenode.net
- https://launchpad.net/~drizzle-discuss
- Drizzle jobs available (see mailing list)
- OSCON – July 20th in San Jose, CA
  - 45 Minute Panel
  - Birds of a Feather (BoF)
  - Expo Hall Booth


Gearman



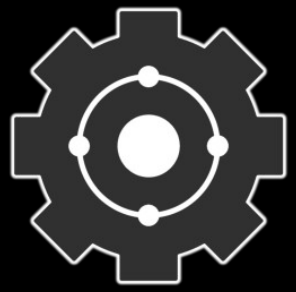
Kittens!

(LiveJournal.com Image Processing)



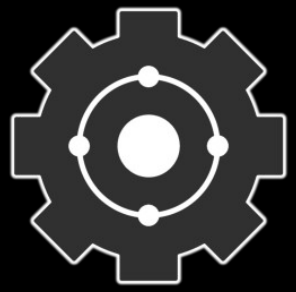
“The way I like to think of Gearman is as a massively distributed, massively fault tolerant fork mechanism.”

- Joe Stump, Digg



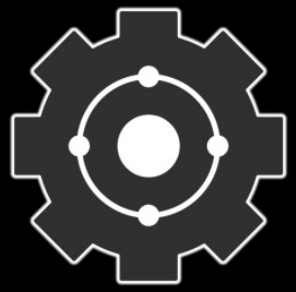
Gearman Overview

- History
- Basics
- Distributed Processing
- Map/Reduce
- Log Collection and Analysis
- Roadmap



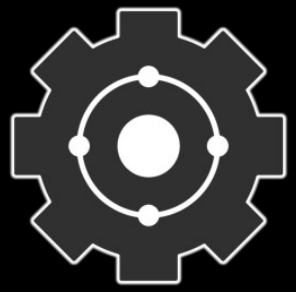
History

- 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
- Many other organizations run it in production



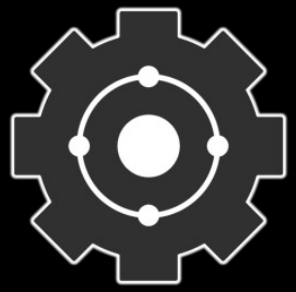
Recent Development

- Rewrite in C
- New Language Bindings
  - PHP ext, Perl XS, Drizzle, MySQL, PostgreSQL
- Command line tool
- Protocol Additions
- Multi-threaded (50k jobs/second)
- Persistent Queues
- Pluggable Protocol



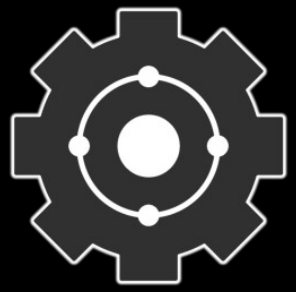
Features

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

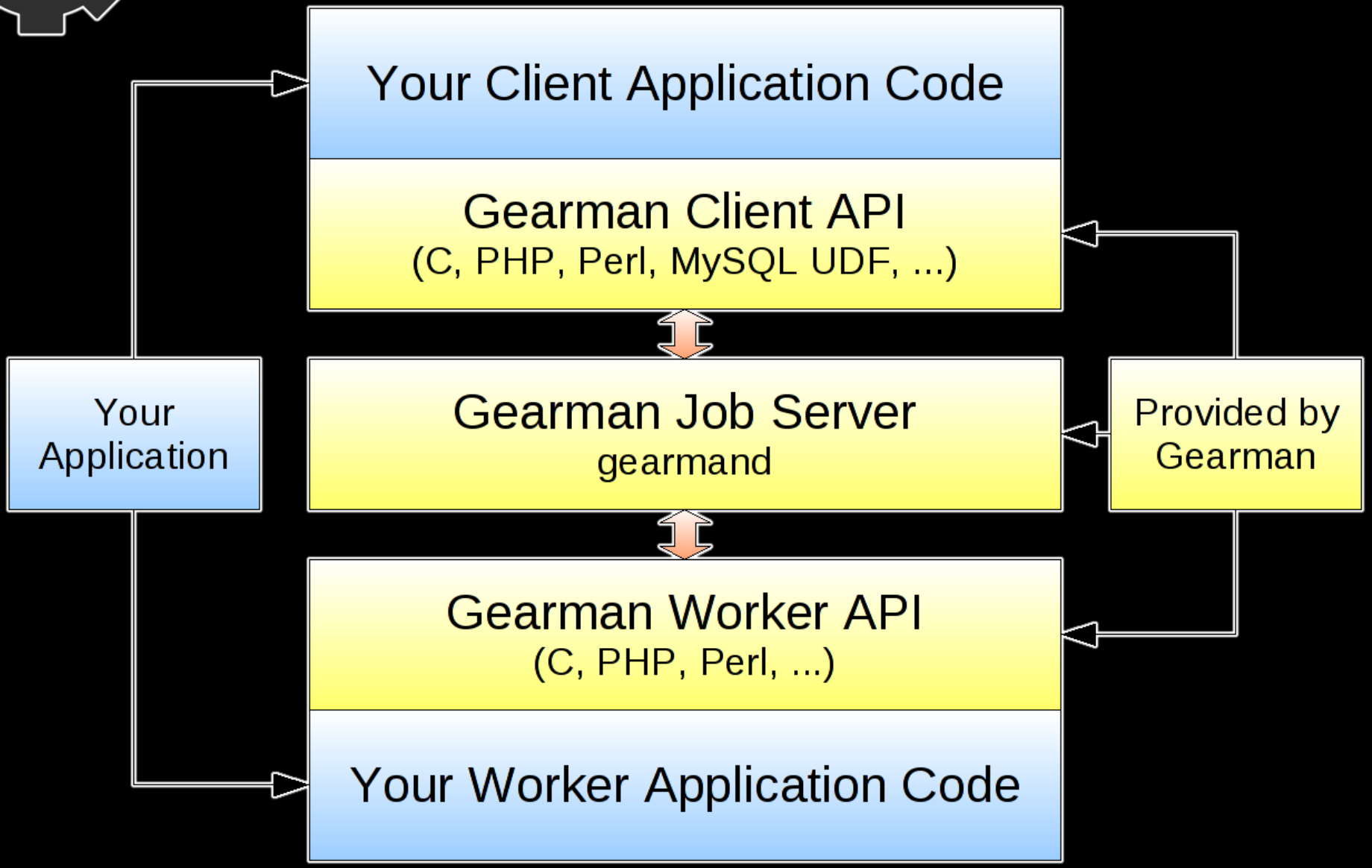


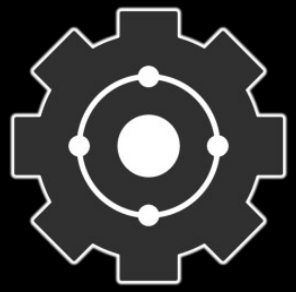
Basics

- 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

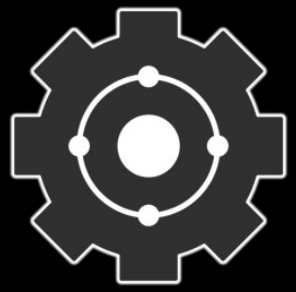




Hello World

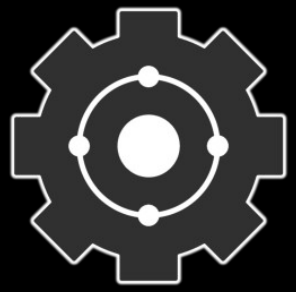
```
$client= new GearmanClient();
$client->addServer();
print $client->do("reverse", "Hello World!");
```

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

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

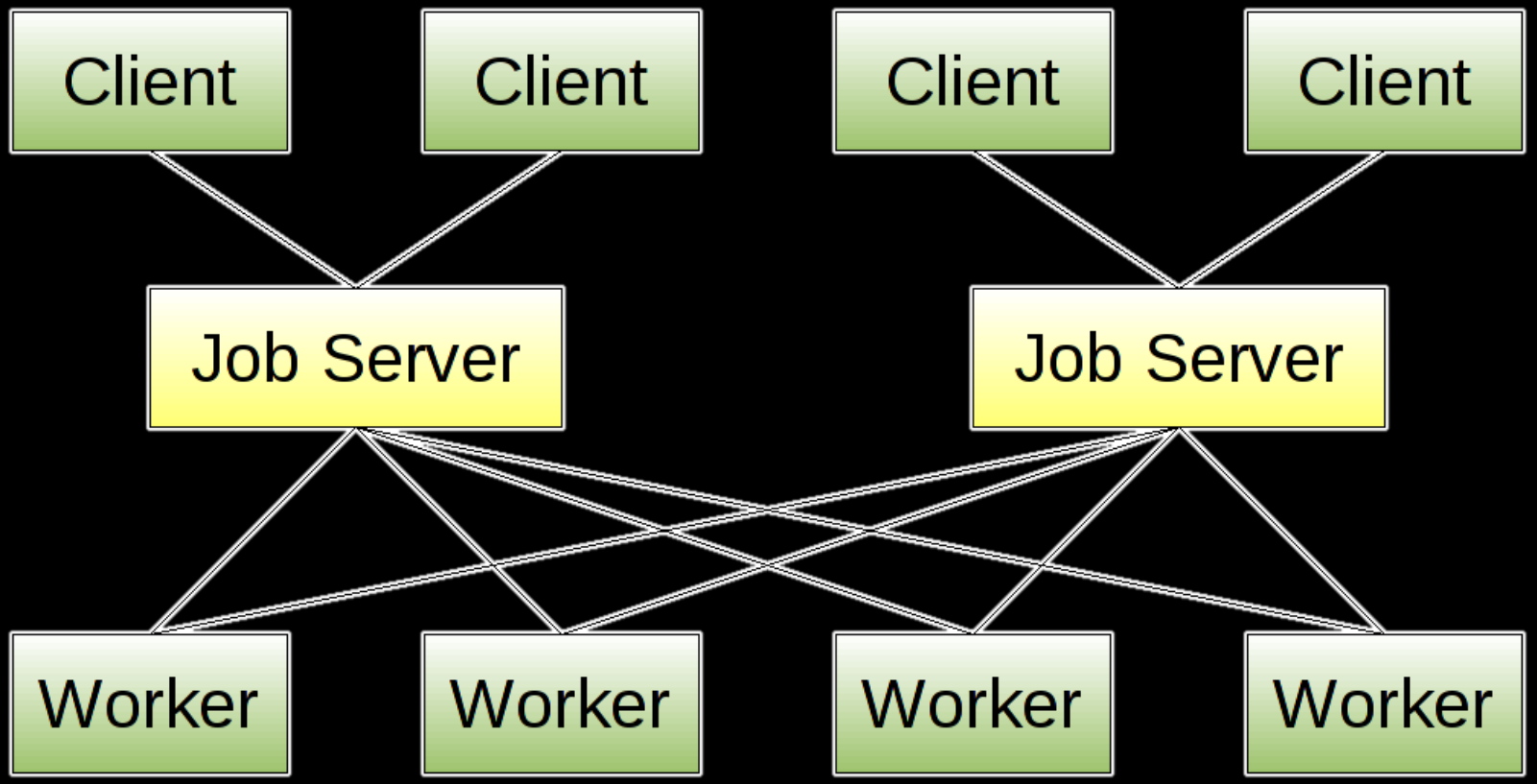


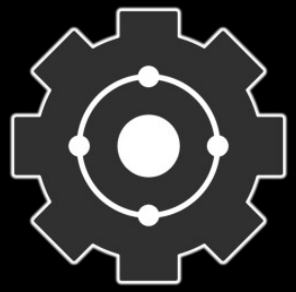
Hello World

```
shell$ gearmand -d

shell$ php worker.php &
[1] 17510

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



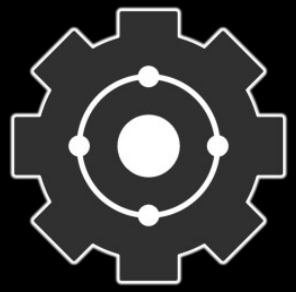
No Single Point of Failure



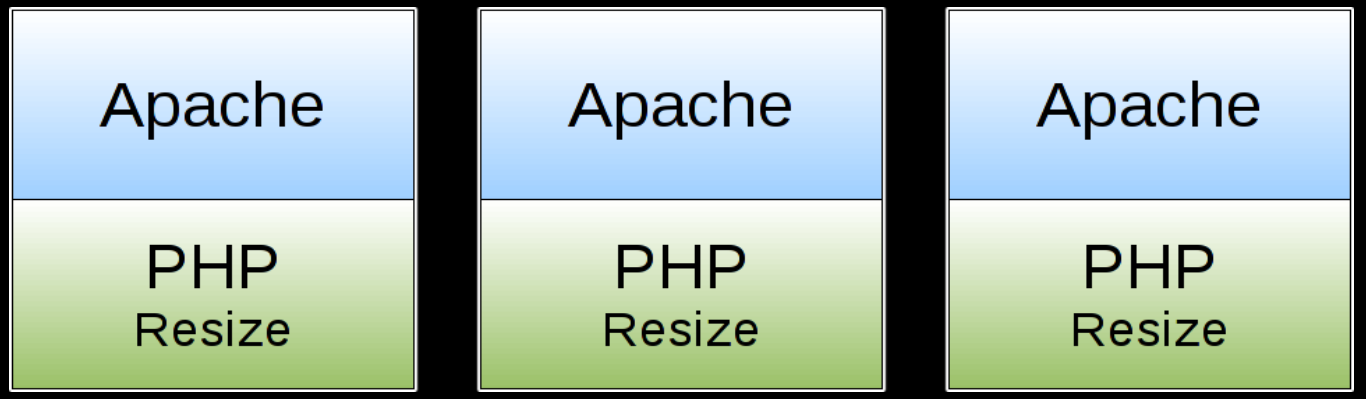


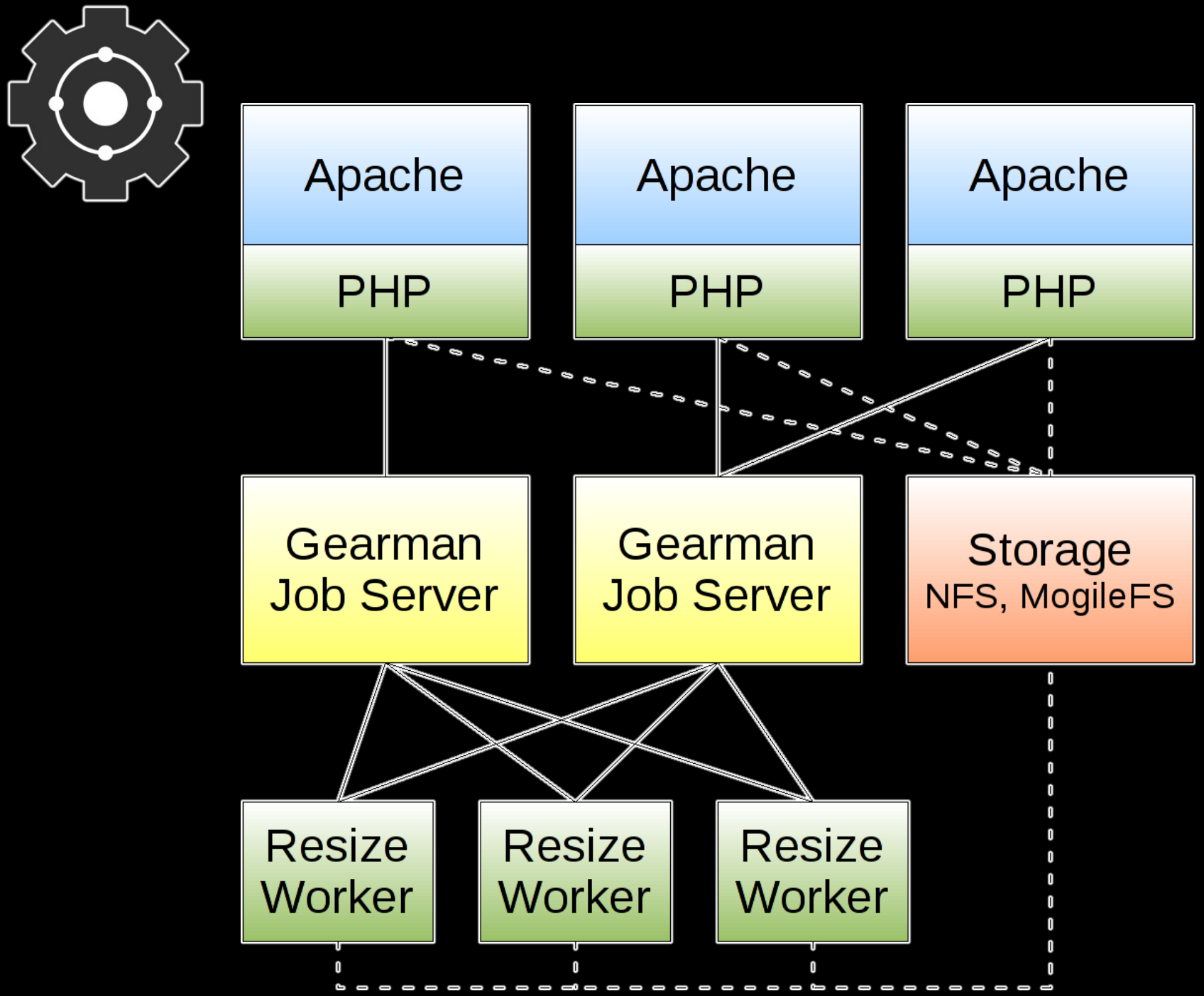
How Is This Useful?

- Provides a distributed nervous system
- Natural load balancing
- Multi-language integration
- Distribute processing
  - Possibly closer to data
- 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

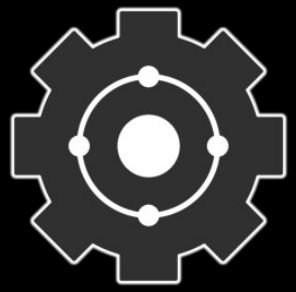


Image Resize Worker

```
$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();
}
```

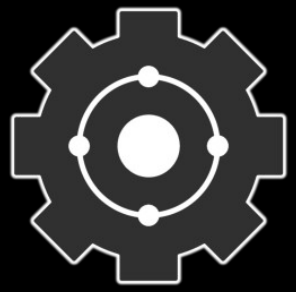
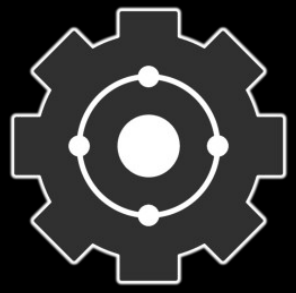
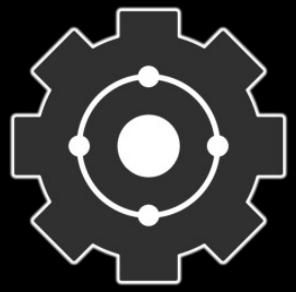


Image Resize Worker

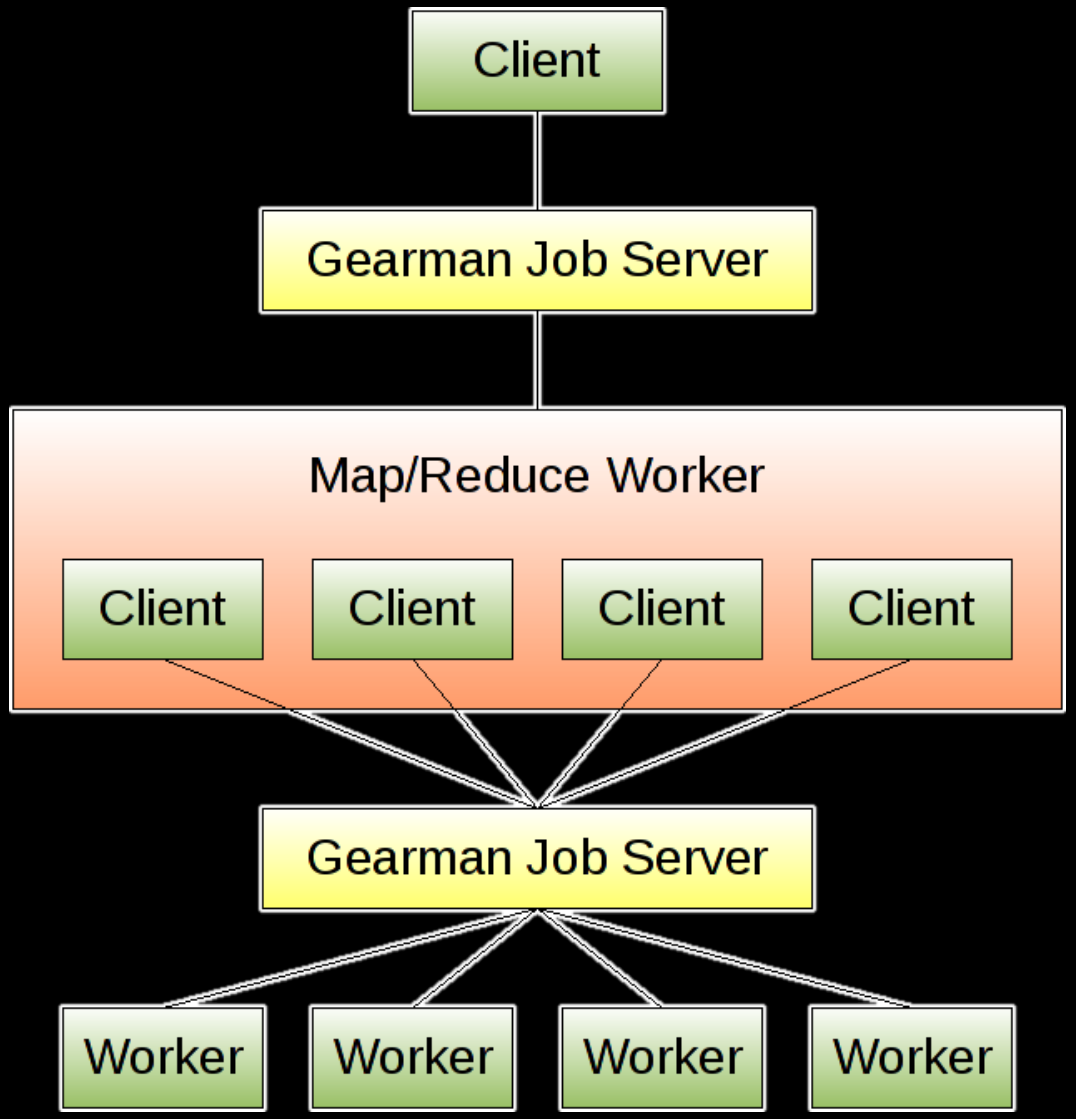
```
shell$ gearmand -d

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
```

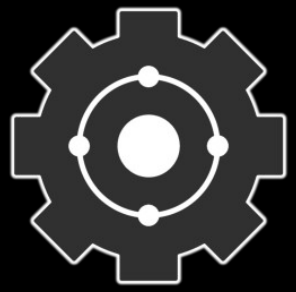


What else?



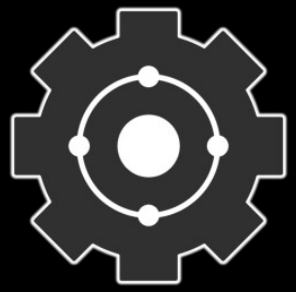
Map/Reduce


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



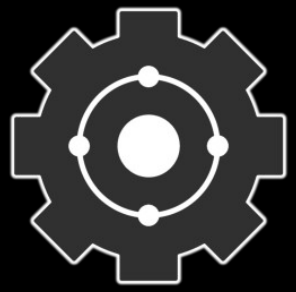
Log Processing

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

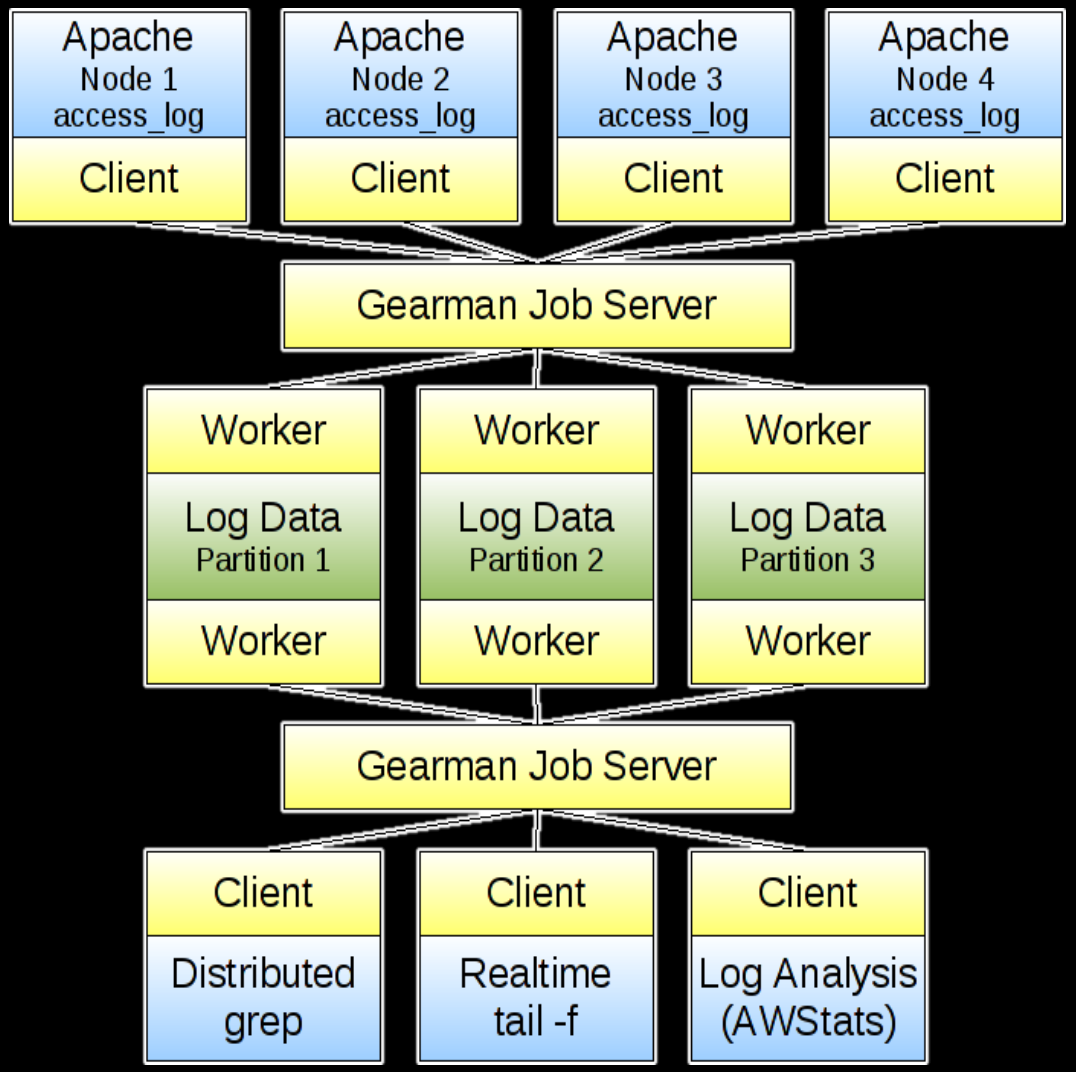


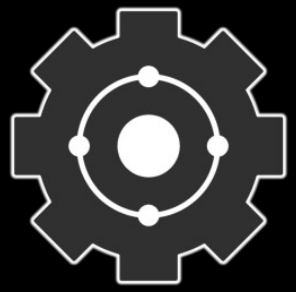
Log Processing

- 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



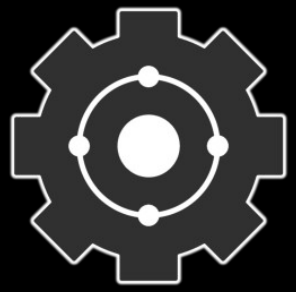
Log Processing





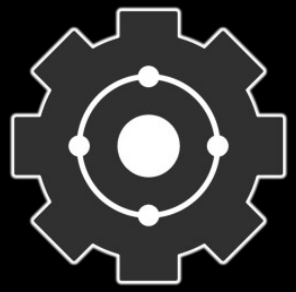
Asynchronous Queues

- They help you scale
- Distributed data storage
  - Eventually consistent data models
  - Choose “AP” in “CAP”, make EC work
- Background Tasks
- Narada



What's Next?

- More protocols (memcached, XMPP, ...)
- TLS, SASL, multi-tenancy
- Replication
- More language interfaces
  - JMS, C-based Python
- Improved statistics reporting
- Event notification hooks
- Monitor service

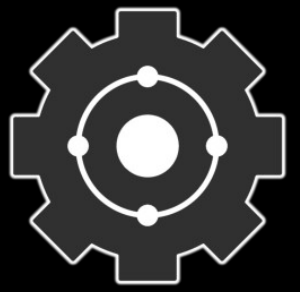


Get Involved!

- http://www.gearman.org/
- #gearman on irc.freenode.net
- http://groups.google.com/group/gearman
- OSCON – July 20th in San Jose, CA
  - 3 Hour Tutorial
  - 45 Minute Session
  - Birds of a Feather (BoF)
  - Expo Hall Booth

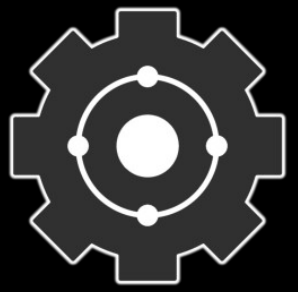
Narada?

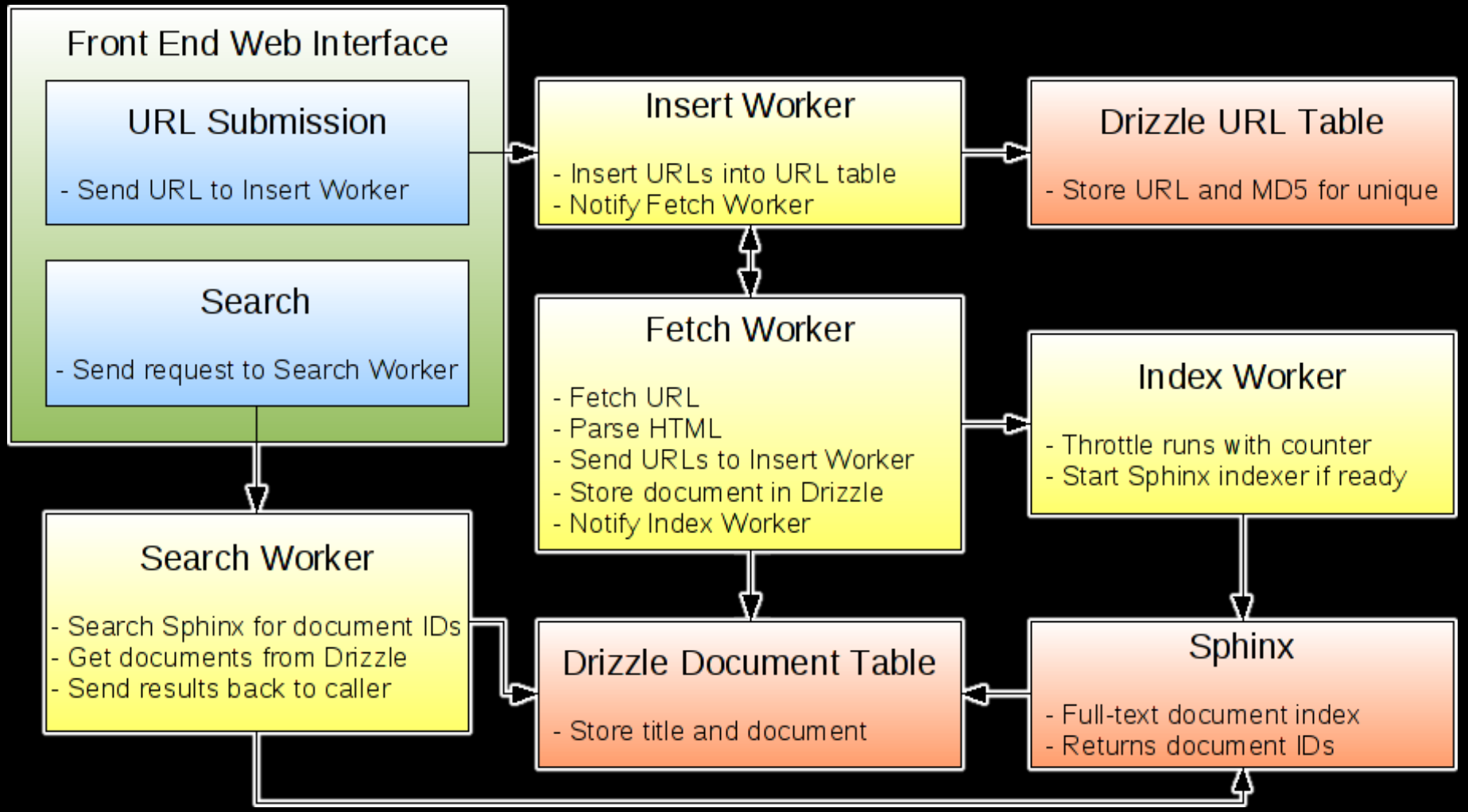
Narada



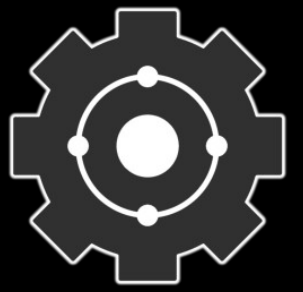
- Started from example in Patrick's book
- Custom search engine
- Perl, PHP, and Java implementations
- Asynchronous queues
- Drizzle or MySQL
- Optionally use memcached
- Easy to integrate into existing projects
- https://launchpad.net/narada

Narada




Front End Web Interface
URL Submission
- Send URL to Insert Worker
Search
- Send request to Search Worker
Insert Worker
- Insert URLs into URL table
- Notify Fetch Worker
Drizzle URL Table
- Store URL and MD5 for unique
Fetch Worker
- Fetch URL
- Parse HTML
- Send URLs to Insert Worker
- Store document in Drizzle
- Notify Index Worker
Index Worker
- Throttle runs with counter
- Start Sphinx indexer if ready
Search Worker
- Search Sphinx for document IDs
- Get documents from Drizzle
- Send results back to caller
Drizzle Document Table
- Store title and document
Sphinx
- Full-text document index
- Returns document IDs

Narada



Demo!