Gearman

MySQL Web Seminar

July 14, 2009

Eric Day – Sun Microsystems

http://www.oddments.org/

Gearman Overview

- History
- Basics
- Job Server
- Applications
  - Map/Reduce
  - Log Analysis
  - Asynchronous Queues
  - Narada
- Roadmap

“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, ...

Recent Development

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

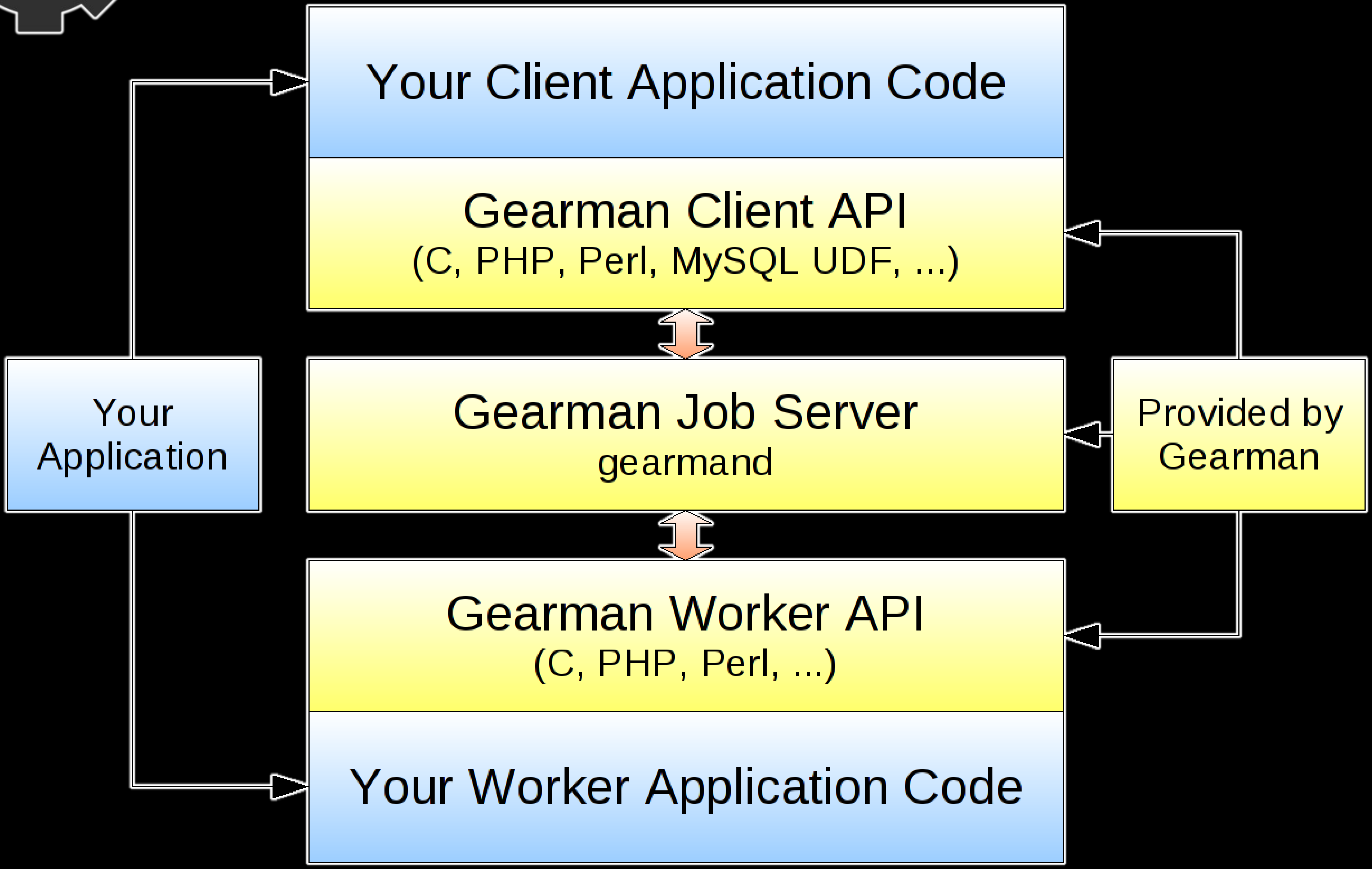
Features

- Open Source (mostly BSD)
- Multi-language
  - Mix clients and workers from different APIs
- Flexible Application Design
  - Not restricted to a single distributed model
- Simple & 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



Your Client Application Code

Gearman Client API

(C, PHP, Perl, MySQL UDF, ...)

Gearman Job Server

gearmand

Gearman Worker API

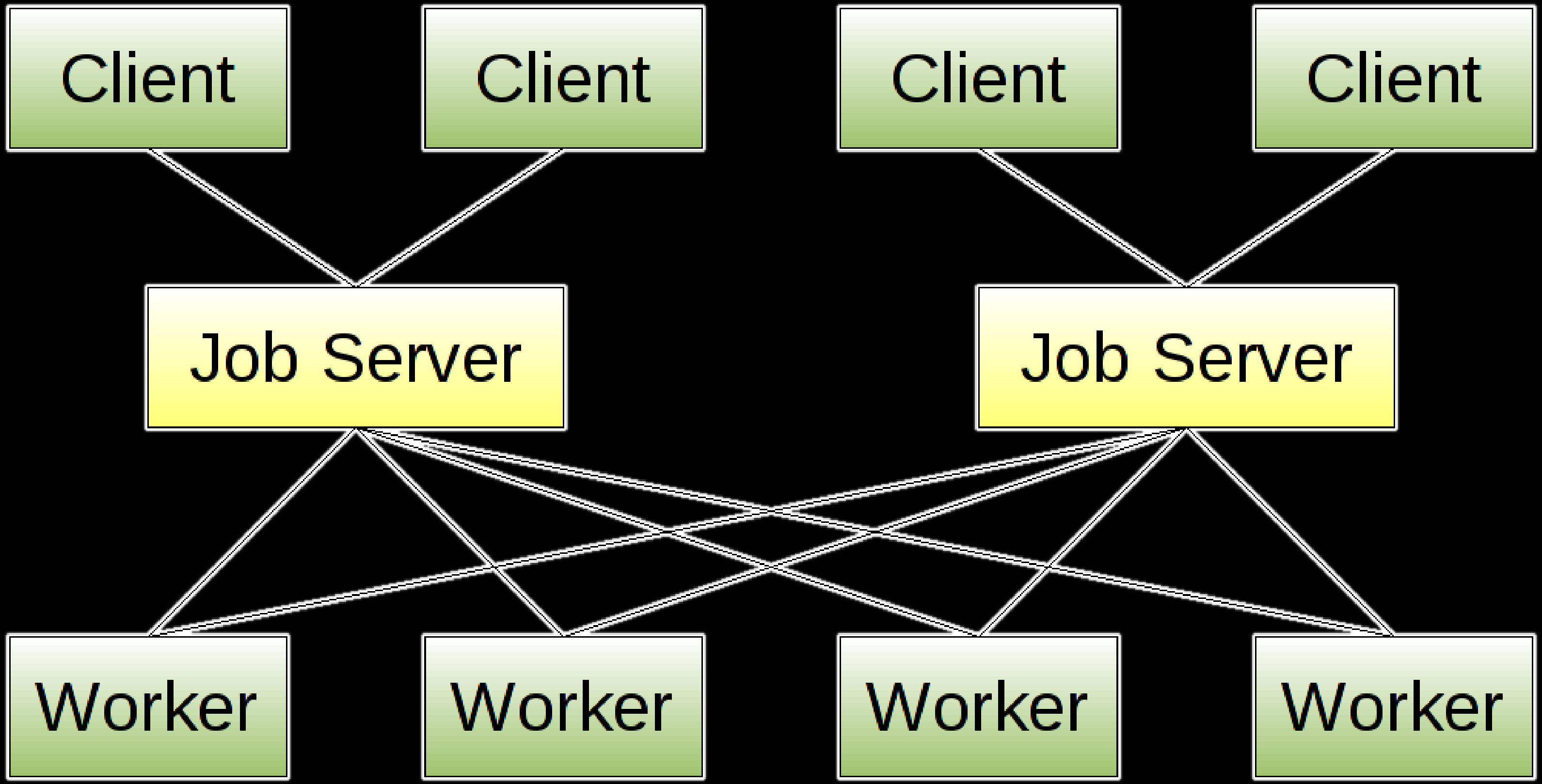
(C, PHP, Perl, ...)

Your Worker Application Code

Your Application

Provided by Gearman

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

Job Server

gearmand

Job Server

- Listens on port 4730
- Clients and workers connect
- Handle job assignment
- Restart jobs on worker failure
- Advanced features
  - Pluggable persistent queue
  - Pluggable protocols

Job Server

- gearmand --help
- Common options

```
-d, --daemon
-h, --help
-l, --log-file=FILE
-p, --port=PORT
-P, --pid-file=FILE
-t, --threads=THREADS
-u, --user=USER
-v, --verbose
```

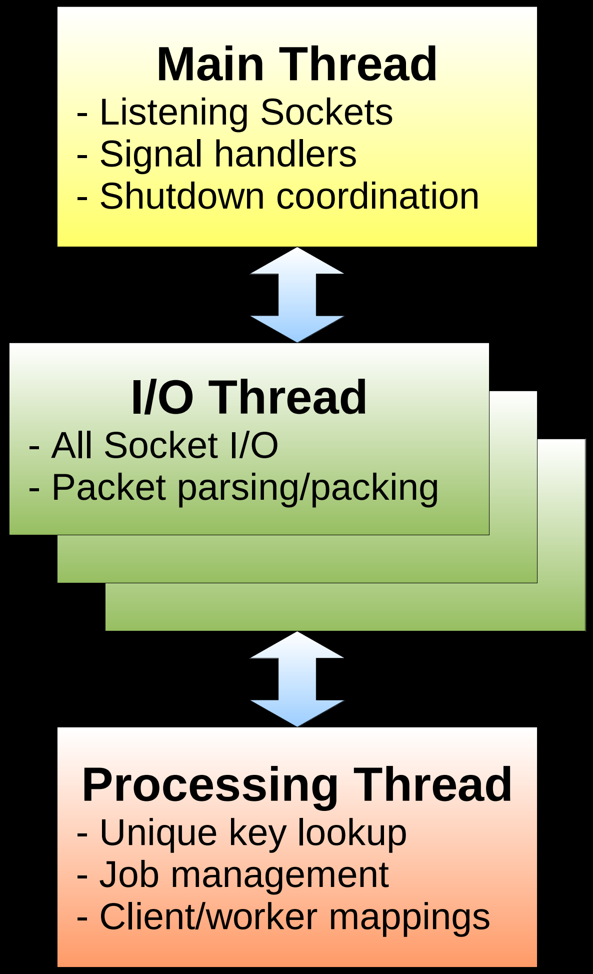
Verbose Option

- Can specify multiple -v
- Very useful for debugging

```
shell$ gearmand -vvvv
 INFO Starting up
DEBUG Initializing libevent for main thread
DEBUG Method for libevent: epoll
DEBUG Trying to listen on :::4730
 INFO Listening on :::4730 (6)
DEBUG Trying to listen on 0.0.0.0:4730
 INFO Creating wakeup pipe
DEBUG Creating 0 threads
 INFO Creating IO thread wakeup pipe
 INFO Adding event for listening socket (6)
 INFO Adding event for wakeup pipe
 INFO Entering main event loop
...
```

Threading Model

- Can run single threaded
- The -t option sets threads
  - I/O thread count
  - Default is -t 0
  - One thread/core
- Processing thread
  - Hashes and other state
  - No systems calls
  - Non-blocking



Main Thread
- Listening Sockets
- Signal handlers
- Shutdown coordination

I/O Thread
- All Socket I/O
- Packet parsing/packing

Processing Thread
- Unique key lookup
- Job management
- Client/worker mappings

Persistent Queues

- Only for background jobs
- Specify -q <queue> option
- libdrizzle module for Drizzle and MySQL

```
shell$ gearmand -vvv -q libdrizzle --libdrizzle-mysql
 INFO Initializing libdrizzle module
 INFO libdrizzle module using table 'test.queue'
...
 INFO libdrizzle replay start
...
DEBUG libdrizzle add: 3ec068d9-293c-4af8-943f-d265138e67f8
DEBUG libdrizzle flush
...
DEBUG libdrizzle done: 3ec068d9-293c-4af8-943f-d265138e67f8
```

Persistent Queues

- libdrizzle queue is as robust as your DB
- Can create your own table

```
libdrizzle Options:

     --libdrizzle-host=HOST
     --libdrizzle-port=PORT
     --libdrizzle-uds=UDS
     --libdrizzle-user=USER
     --libdrizzle-password=PASSWORD
     --libdrizzle-db=DB
     --libdrizzle-table=TABLE
     --libdrizzle-mysql

shell$ gearmand -q libdrizzle ...
```

Persistent Queues

- libmemcached
- PostgreSQL
- sqlite3
- Flat file
- Easy to add your own!

Pluggable Protocol

- Handle packet parsing and packing
- Optionally handle raw socket I/O
- HTTP protocol
- Others coming soon
  - memcached
  - XMPP
  - ?

HTTP Protocol

- GET and POST requests
  - Send workload with POST, Content-Length
- Function name from URL
- Optional headers
  - X-Gearman-Unique: <unique key>
  - X-Gearman-Background: true
  - X-Gearman-Priority: <high|low>
- Other headers ignored for now

HTTP Protocol

```
shell$ gearmand -r http &

shell$ nc localhost 8080
POST /reverse HTTP/1.1
Content-Length: 12

Hello World!

HTTP/1.0 200 OK
X-Gearman-Job-Handle: H:lap:1
Content-Length: 12
Server: Gearman/0.9

!dlroW olleH
```

Applications

Asynchronous Queues

- Background Tasks
- They help you scale
- Distributed data storage
  - Eventually consistent data models
  - Choose “AP” in “CAP”
    - Consistency
    - Availability
    - Partitions (tolerance to network partitions)
  - Make eventual consistency work
  - Conflict resolution if needed

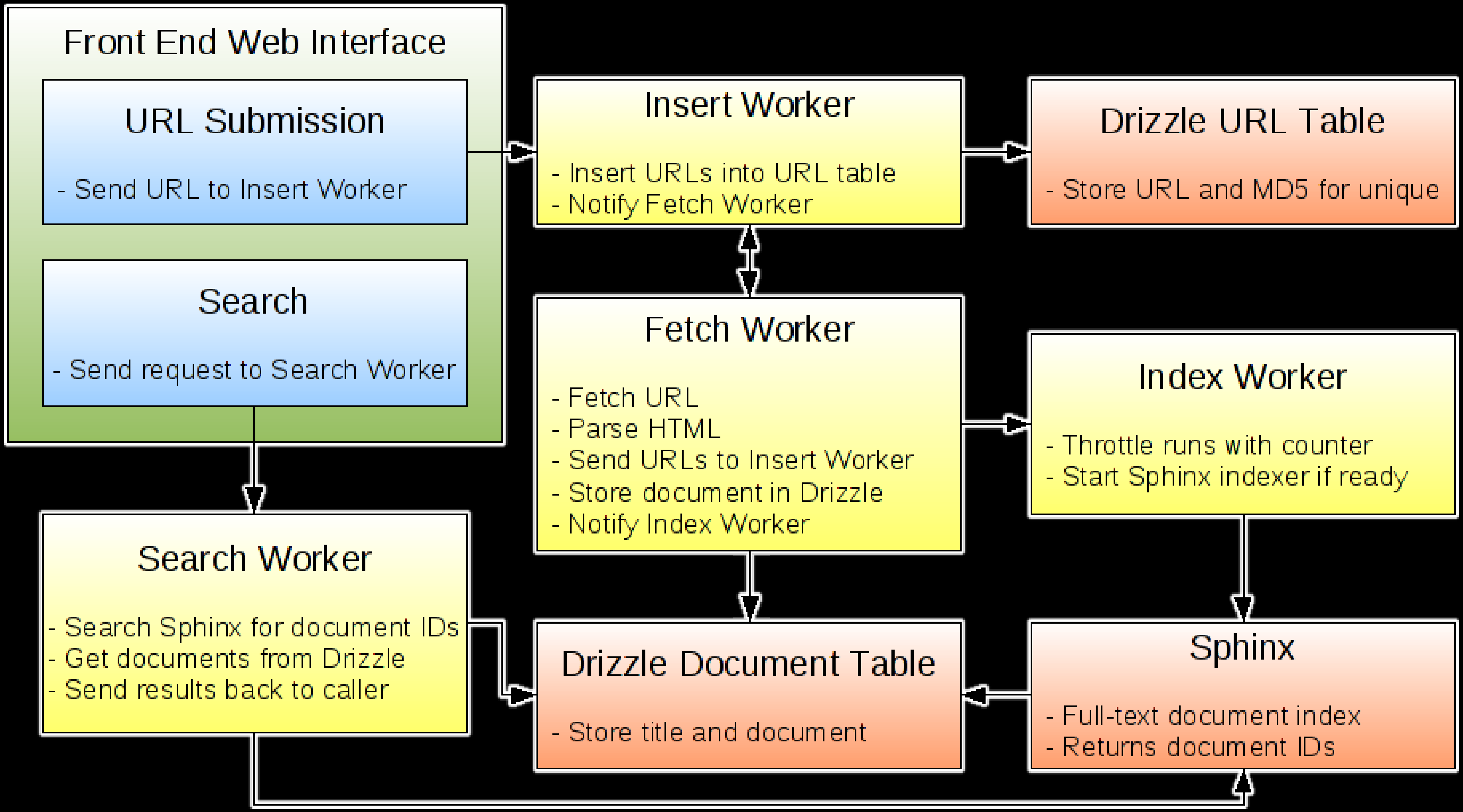
Asynchronous Queues

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

Narada

- Example in Patrick Galbraith'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

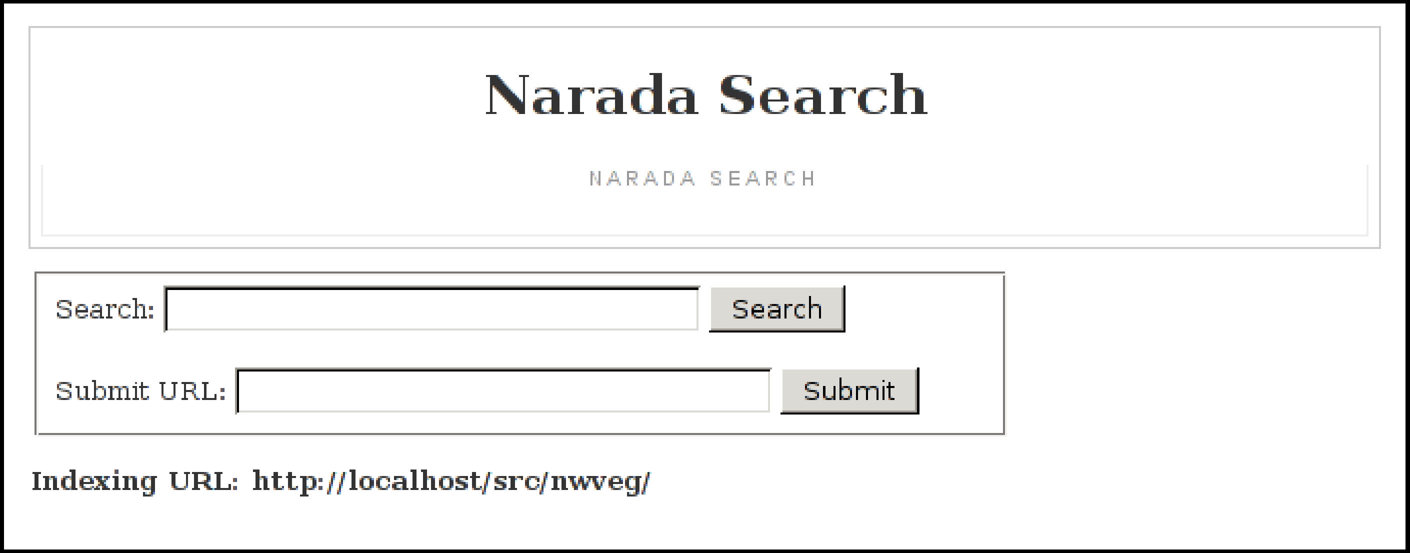
Drizzle Document Table

- Store title and document

Sphinx

- Full-text document index
- Returns document IDs

URL Submission



Narada Search

NARADA SEARCH

Indexing URL: http://localhost/src/nwveg/

Insert Worker

- Insert URL into table
- Notify Fetch Worker

```
shell$ php NaradaInsert.php
Insert URL: http://localhost/src/nwveg/
```

Fetch Worker

- Fetch URL, process HTML, find more URLs
- Store document and notify Index Worker

```
shell$ php NaradaFetch.php
Fetching: http://localhost/src/nwveg/
Protocol: http://
Domain: localhost
Path: /src/nwveg/
File:
Local URL: http://localhost/src/nwveg/index.php
Local URL: http://localhost/src/nwveg/news.php
Local URL: http://localhost/src/nwveg/whyveg.php
Local URL: http://localhost/src/nwveg/board.php
Local URL: http://localhost/src/nwveg/subscribe.php
Local URL: http://localhost/src/nwveg/discounts.php
...
Skipping Remote URL: http://twitter.com/northwestveg
Title: Northwest VEG - Home
```

Insert Worker

- More URLs from Fetch Worker

```
shell$ php NaradaInsert.php
Insert URL: http://localhost/src/nwveg/
Insert URL: http://localhost/src/nwveg/index.php
Insert URL: http://localhost/src/nwveg/news.php
Insert URL: http://localhost/src/nwveg/whyveg.php
Insert URL: http://localhost/src/nwveg/mvp.php
Insert URL: http://localhost/src/nwveg/veg101.php
Insert URL: http://localhost/src/nwveg/join.php
Insert URL: http://localhost/src/nwveg/volunteer.php
Insert URL: http://localhost/src/nwveg/events.php
...
```

Insert/Fetch Workers

- URL Feedback Loop
- Filter at some point
  - Recursion level
  - Restrict to given domain
- Now ready to index
- ...then search!

Index Worker

- Don't run indexer for every document

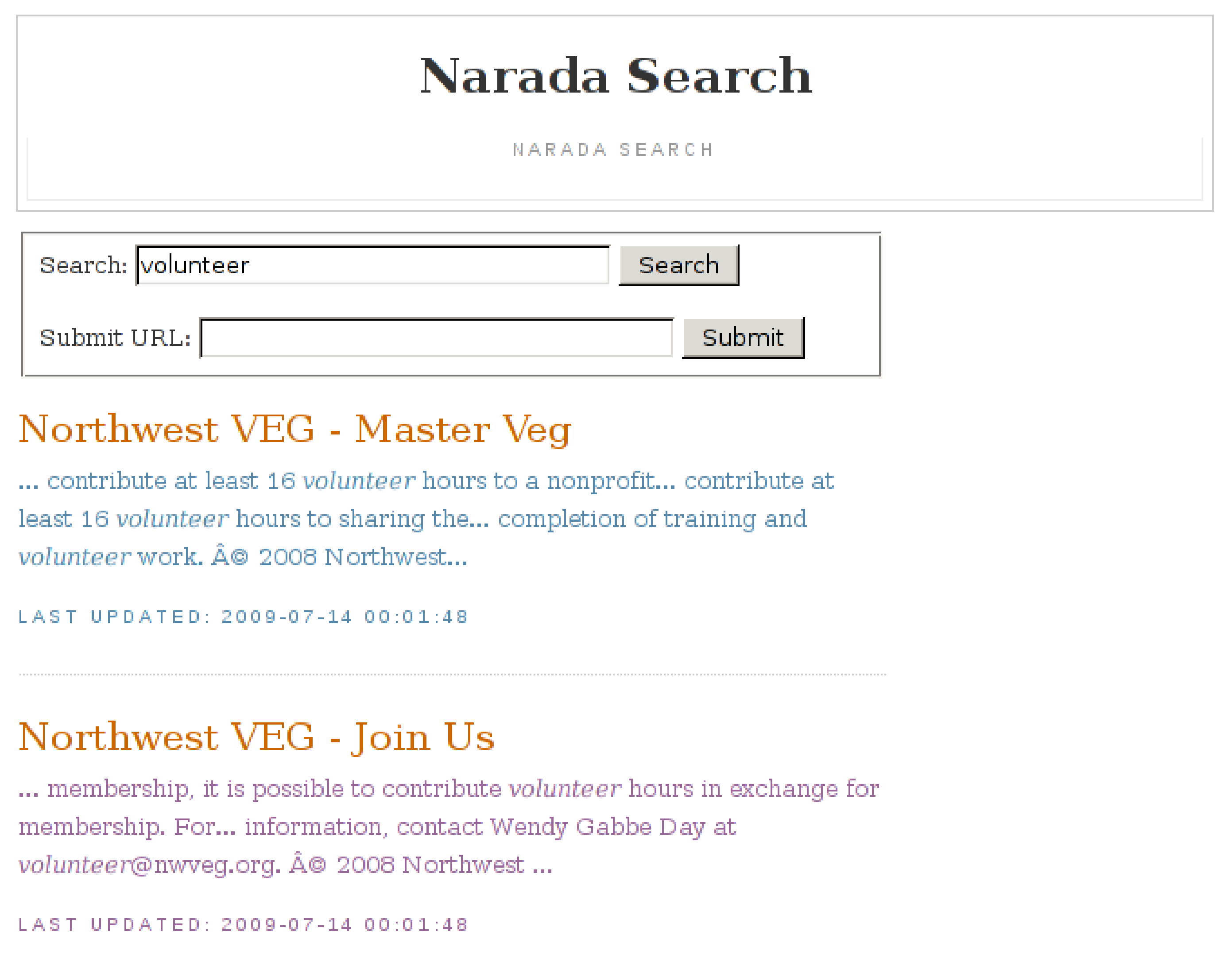
```
shell$ php NaradaIndex.php
Index Count: 1
...
Index Count: 10
Running Index
Sphinx 0.9.9-rc2 (r1785)
Copyright (c) 2001-2009, Andrew Aksyonoff

using config file '/home/eday/other/sphinx/etc/sphinx.conf'...
indexing index 'url'...
collected 25 docs, 0.1 MB
sorted 0.0 Mhits, 100.0% done
total 25 docs, 61524 bytes
total 0.085 sec, 718134 bytes/sec, 291.81 docs/sec
indexing index 'url_delta'...
collected 0 docs, 0.0 MB
total 0 docs, 0 bytes
total 0.045 sec, 0 bytes/sec, 0.00 docs/sec
distributed index 'dist_url' can not be directly indexed; skipping.
total 3 reads, 0.000 sec, 30.4 kb/call avg, 0.0 msec/call avg
total 9 writes, 0.000 sec, 8.3 kb/call avg, 0.0 msec/call avg
rotating indices: succesfully sent SIGHUP to searchd (pid=17010).
```

Search Worker

- Query Sphinx for document IDs
- Get documents from Drizzle or MySQL
  - Soon will get from memcached too

```
shell$ php NaradaSearch.php
1 Documents matched 'nonprofit'
2 Documents matched 'board'
5 Documents matched 'oregon'
3 Documents matched 'vegfest'
7 Documents matched 'volunteer'
```

Search



Narada Search

NARADA SEARCH

Northwest VEG - Master Veg

... contribute at least 16 *volunteer* hours to a nonprofit... contribute at least 16 *volunteer* hours to sharing the... completion of training and *volunteer* work. Â© 2008 Northwest...

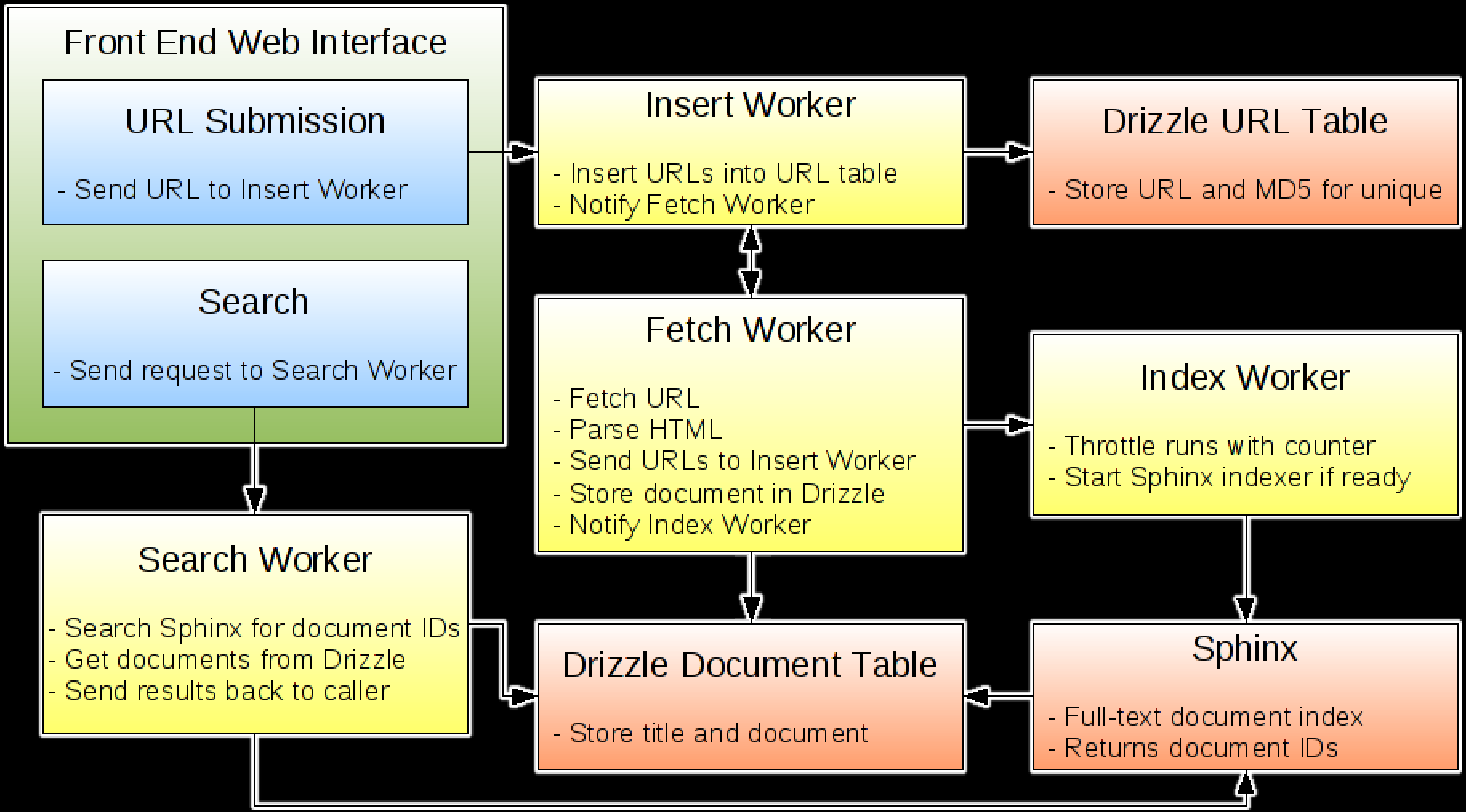
LAST UPDATED: 2009-07-14 00:01:48

Northwest VEG - Join Us

... membership, it is possible to contribute *volunteer* hours in exchange for membership. For... information, contact Wendy Gabbe Day at *volunteer*@nwveg.org. Â© 2008 Northwest ...

LAST UPDATED: 2009-07-14 00:01:48

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

What's Next?

- More protocol and queue modules
- TLS, SASL, multi-tenancy
- Replication (job relay)
- 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
  - July 20-24 in San Jose
  - 3 Hour tutorial
  - 45 Minute Session (similar material)
  - Birds of a Feather (BoF)
  - Expo Hall Booth