Scaling Web Apps With RabbitMQ

Alvaro Videla - ECUG Con 2010

About Me

- Development Manager at TheNetCircle.com
- Twitter: @old_sound
- Blog: http://videlalvaro.github.com/
- 老外

Agenda

- RabbitMQ
- AMQP
- Scaling Web Apps
- High Availability

RabbitMQ

RabbitMQ

- Enterprise Messaging System
- Open Source MPL
- Written in Erlang/OTP
- Commercial Support

Features

- Reliable and High Scalable
- Easy To install
- Easy To Cluster
- Runs on: Windows, Solaris, Linux, OSX
- AMQP 0.8 - 0.9.1

Client Libraries

- Java
- .NET/C#
- Erlang
- Ruby, Python, PHP, Perl, AS3, Lisp, Scala, Clojure, Haskell

Docs/Support

- http://www.rabbitmq.com/documentation.html
- http://dev.rabbitmq.com/wiki/
- #rabbitmq at irc.freenode.net
- http://www.rabbitmq.com/email-archive.html

AMQP

AMQP

- Advanced Message Queuing Protocol
- Suits Interoperability
- Completely Open Protocol
- Binary Protocol
- AMQP Model
- AMQP Wire Format

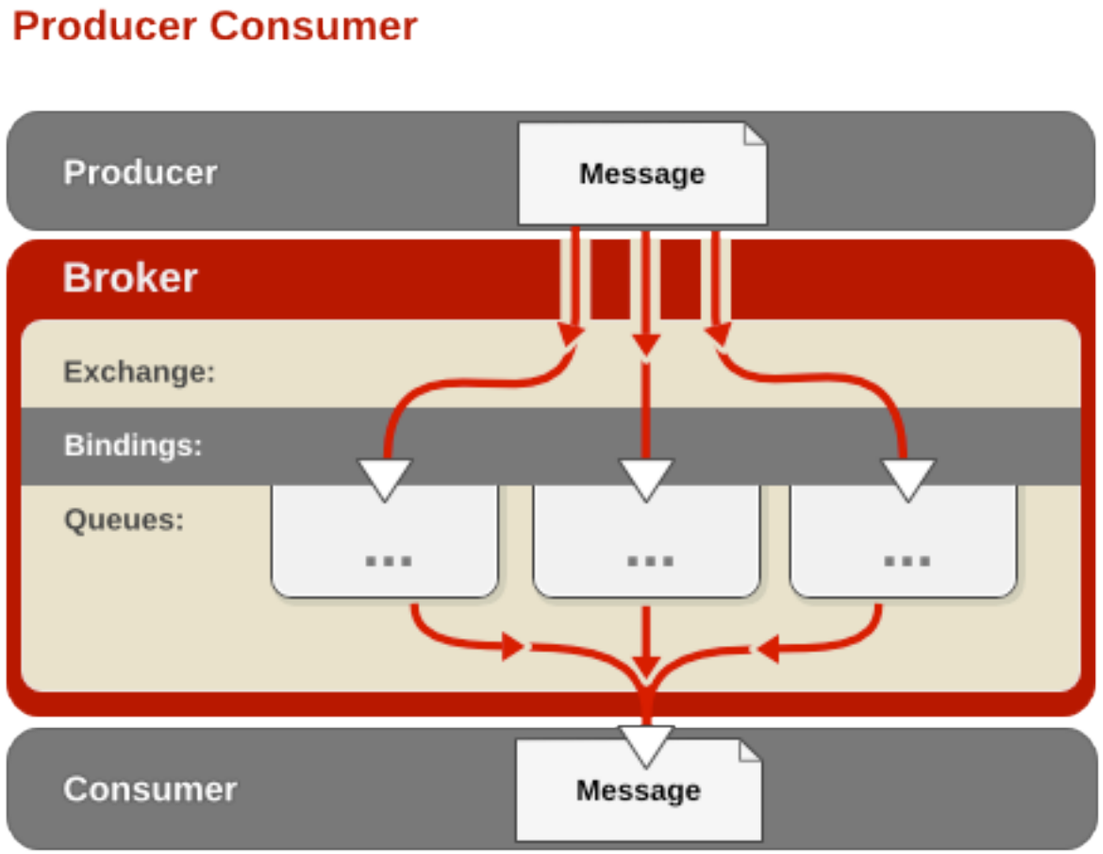
AMQP Model

- Exchanges
- Message Queues
- Bindings
- Rules for binding them

AMQP Wire Protocol

- Functional Layer
- Transport Layer

Message Flow



http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_MRG/1.0/html/Messaging_Tutorial/chap-Messaging_Tutorial-Initial_Concepts.html

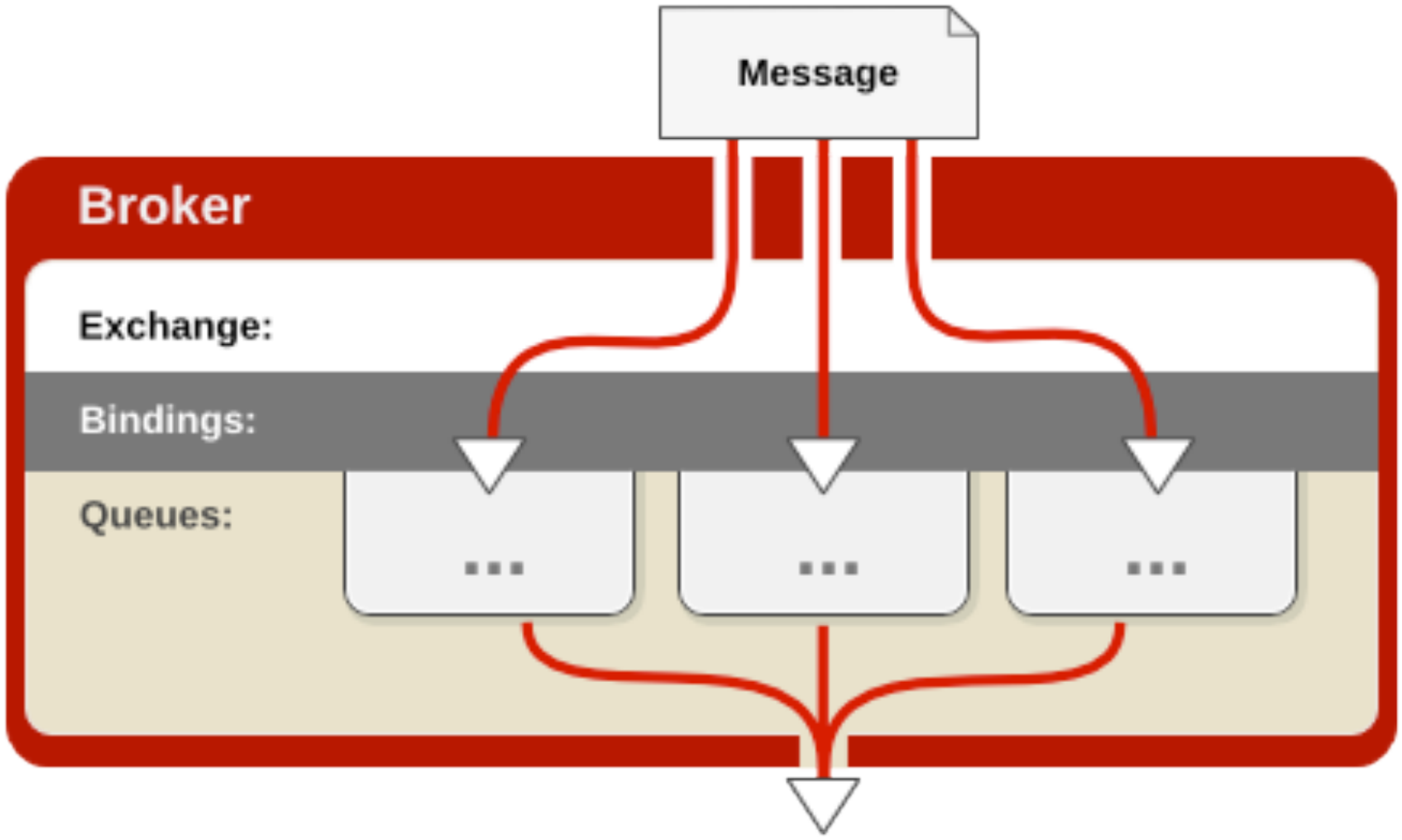
Exchange Types

- Fanout
- Direct
- Topic

Default Exchanges

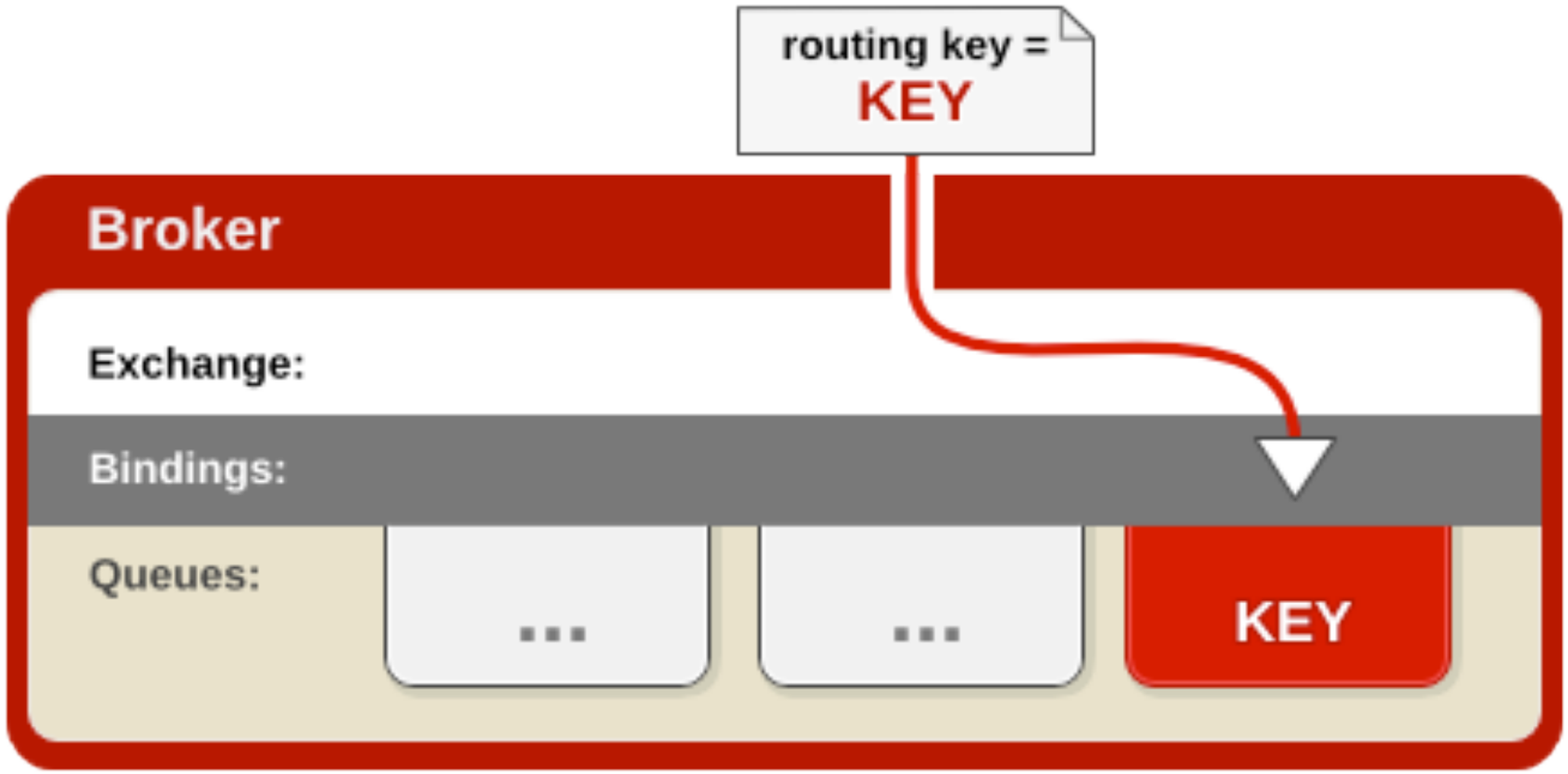
- amqp.fanout
- amqp.direct
- amqp.topic

Fanout Exchange



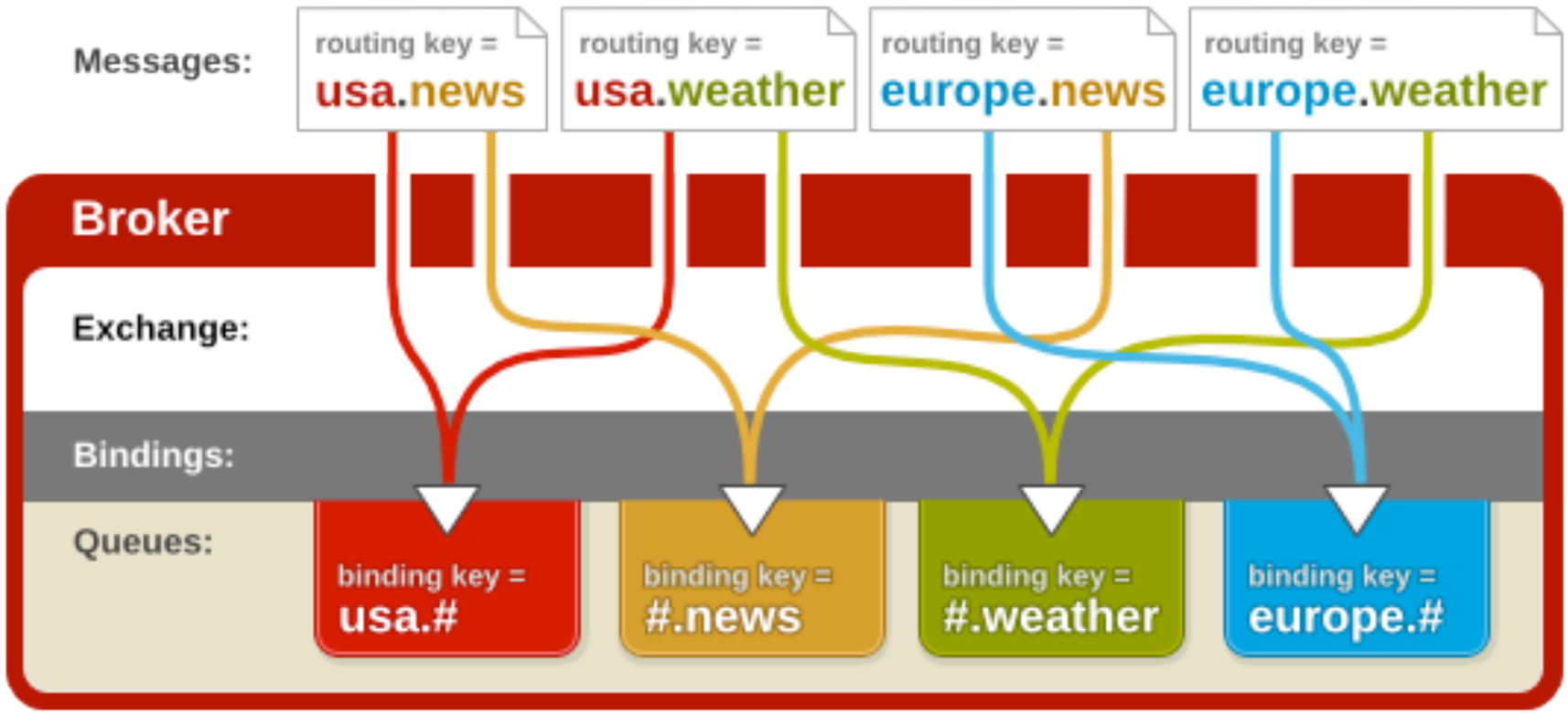
http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_MRG/1.0/html/Messaging_Tutorial/sect-Messaging_Tutorial-Initial_Concepts-Fanout_Exchange.html

Direct Exchange



http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_MRG/1.0/html/Messaging_Tutorial/sect-Messaging_Tutorial-Initial_Concepts-Direct_Exchange.html

Topic Exchange



http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_MRG/1.0/html/Messaging_Tutorial/sect-Messaging_Tutorial-Initial_Concepts-Topic_Exchange.html

Scaling Web Apps

What means “to Scale”

- Scale Up
- Scale Down
- Scale to New Product Requirements

Scenario I

Batch Processing

Requirements

Requirements

- Generate XML

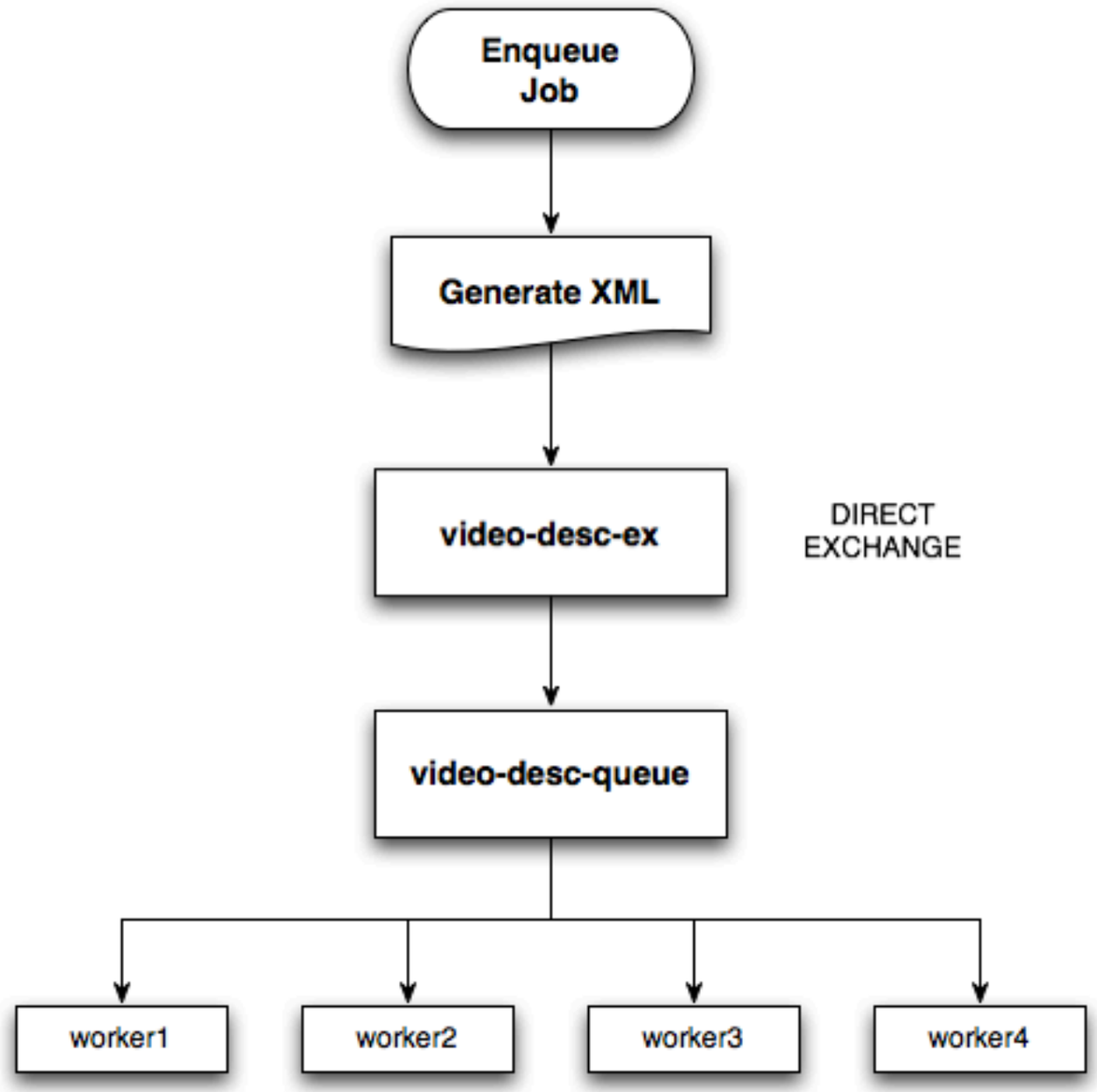
Requirements

- Generate XML
- Distribution Over a Cluster

Requirements

- Generate XML
- Distribution Over a Cluster
- Elasticity - Add/Remove new workers

Design


Enqueue Job
Generate XML
video-desc-ex
DIRECT EXCHANGE
video-desc-queue
worker1
worker2
worker3
worker4

Publisher Code

```
$conn = new AMQPConnection(HOST, PORT, USER, PASS, VHOST);
$channel = $conn->channel();

$channel->exchange_declare('video-desc-ex', 'direct', false,
                true, false);

$msg = new AMQPMessage($video_info,
            array('content_type' => 'text/plain',
                  'delivery_mode' => 2));

$channel->basic_publish($msg, 'video-desc-ex');

$channel->close();
$conn->close();
```

Consumer Code

```
$conn = new AMQPConnection(HOST, PORT, USER, PASS, VHOST);
$channel = $conn->channel();

$channel->exchange_declare('video-desc-ex', 'direct', false,
                true, false);
$channel->queue_declare('video-desc-queue', false, true,
                false, false);
$channel->queue_bind('video-desc-queue', 'video-desc-ex');

$channel->basic_consume('video-desc-queue', $consumer_tag,
                false, false, false, false, $consumer);

while(count($channel->callbacks)) {
    $channel->wait();
}
```

Scenario II

Upload Pictures

Requirements

Requirements

- Upload Picture

Requirements

- Upload Picture
- Reward User

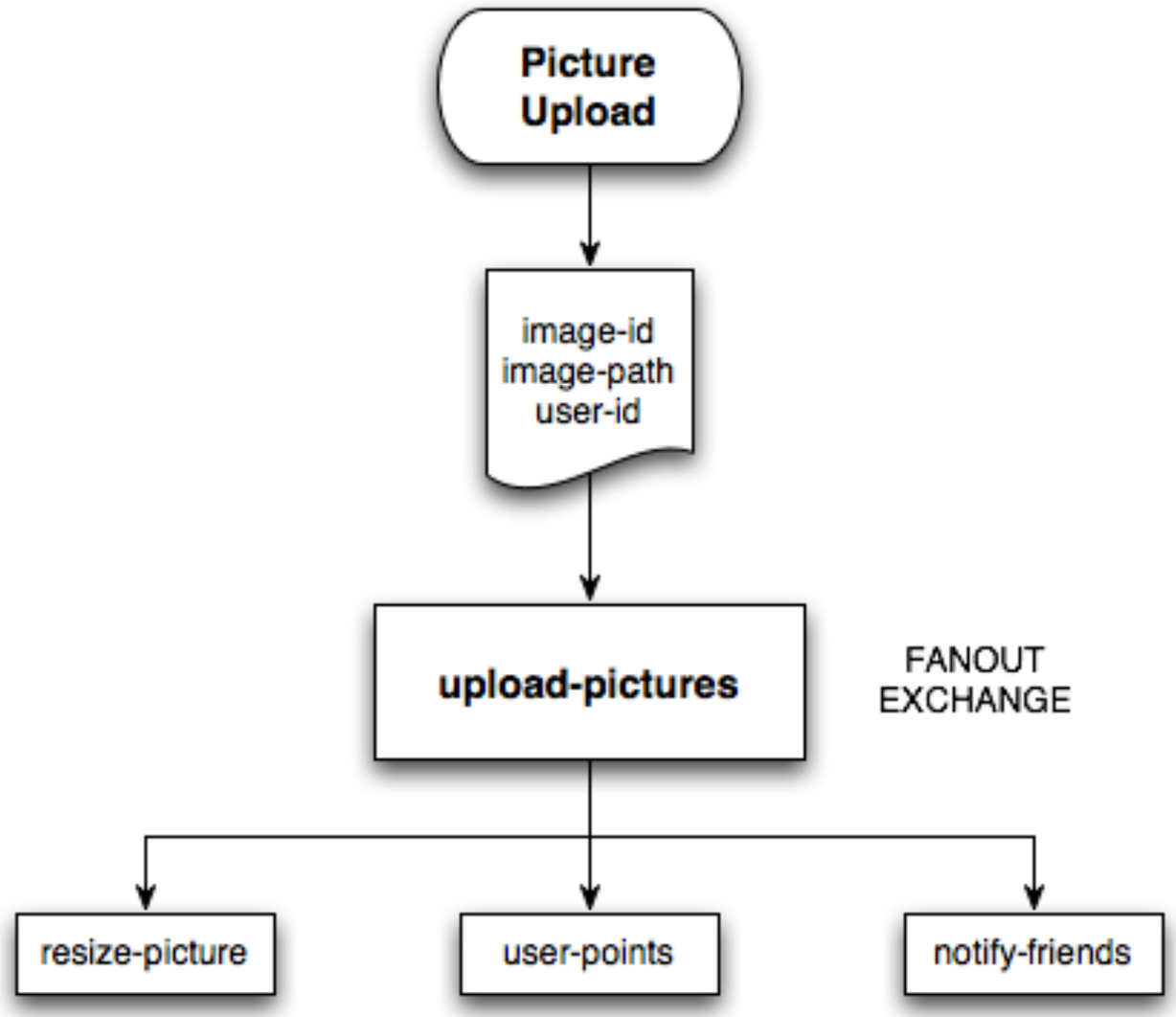
Requirements

- Upload Picture
- Reward User
- Notify User Friends

Requirements

- Upload Picture
- Reward User
- Notify User Friends
- Resize Picture

Design


Picture Upload
image-id
image-path
user-id
upload-pictures
FANOUT EXCHANGE
resize-picture
user-points
notify-friends

Publisher Code

```
$conn = new AMQPConnection(HOST, PORT, USER, PASS, VHOST);
$channel = $conn->channel();

$channel->exchange_declare('upload-pictures', 'fanout', false,
true, false);

$metadata = json_encode(array(
  'image_id' => $image_id,
  'user_id' => $user_id,
  ‘image_path' => $image_path));

$msg = new AMQPMessage($metadata, array('content_type' =>
'application/json',   'delivery_mode' => 2));

$channel->basic_publish($msg, 'upload-pictures');

$channel->close();
$conn->close();
```

Consumer Code

```
$channel->exchange_declare('upload-pictures', 'fanout',
                false, true, false);

$channel->queue_declare('resize-picture', false, true,
                false, false);

$channel->queue_bind('resize-picture', 'upload-pictures');

$channel->basic_consume('resize-picture', $consumer_tag,
                false, false, false, false, $consumer);

while(count($channel->callbacks)) {
  $channel->wait();
}
```

Consumer Code

```
$consumer = function($msg){
  $meta = json_decode($msg->body, true);

  resize_picture($meta['image_id'], $meta['image_path']);

  $msg->delivery_info['channel']->
    basic_ack($msg->delivery_info['delivery_tag']);

  if($msg->body == 'quit'){
    $msg->delivery_info['channel']->
      basic_cancel($msg->delivery_info['consumer_tag']);
  }
};
```

Scenario III

Distributed Logging

Requirements

Requirements

- Several Web Servers

Requirements

- Several Web Servers
- Logic Separated by Module/Action

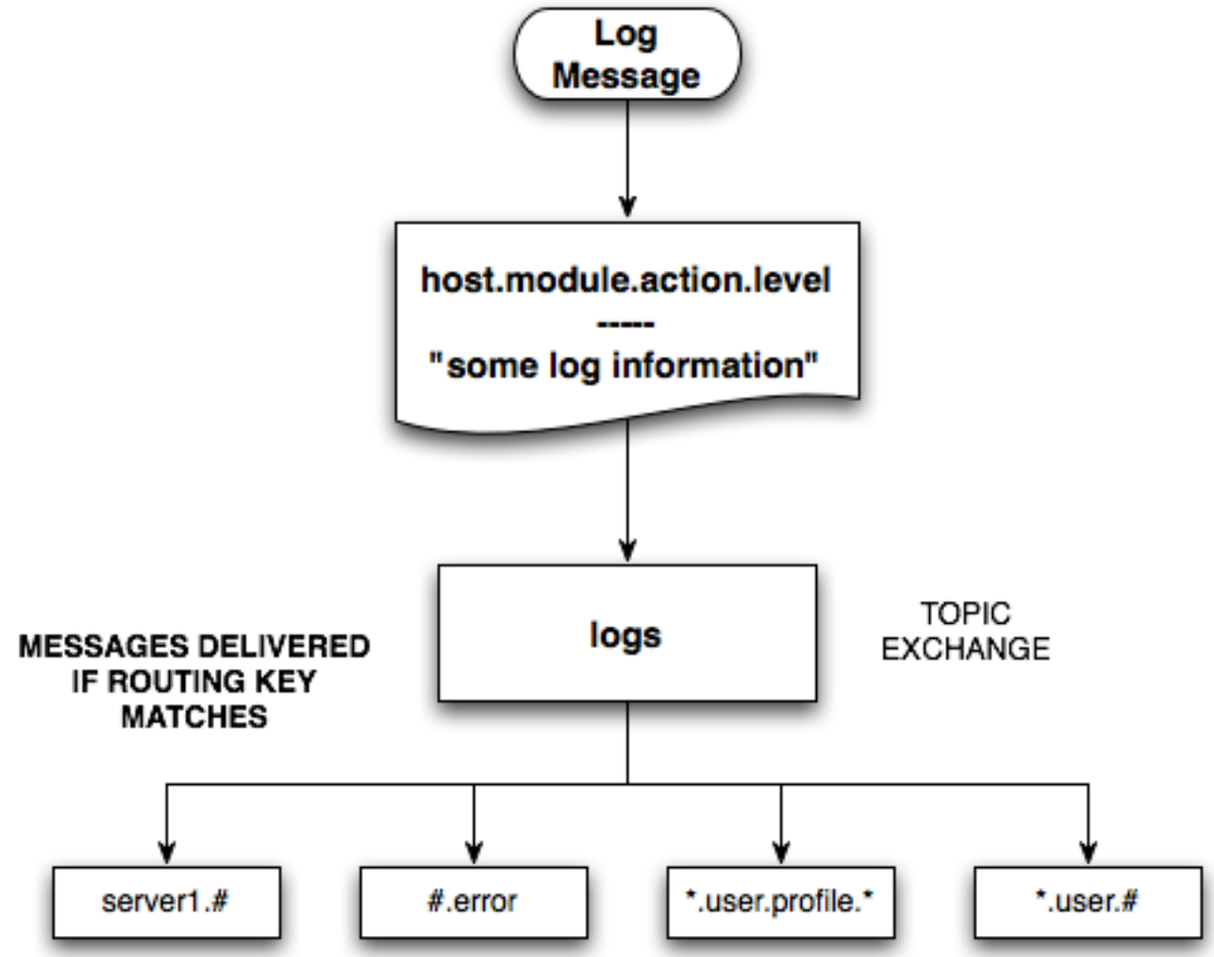
Requirements

- Several Web Servers
- Logic Separated by Module/Action
- Several Log Levels:

Requirements

- Several Web Servers
- Logic Separated by Module/Action
- Several Log Levels:
  - Info
  - Warning
  - Error

Design


Log
Message
host.module.action.level

"some log information"
logs
MESSAGES DELIVERED
IF ROUTING KEY
MATCHES
TOPIC
EXCHANGE
server1.#
#.error
.user.profile.
*.user.#

Publisher Code

```
$conn = new AMQPConnection(HOST, PORT, USER, PASS, VHOST);
$channel = $conn->channel();

$channel->exchange_declare('logs', 'topic', false, true, false);

$msg = new AMQPMessage('some log message',
                array('content_type' => 'text/plain'));

$channel->basic_publish($msg, 'logs',
            server1.user.profile.info');

$channel->close();
$conn->close();
```

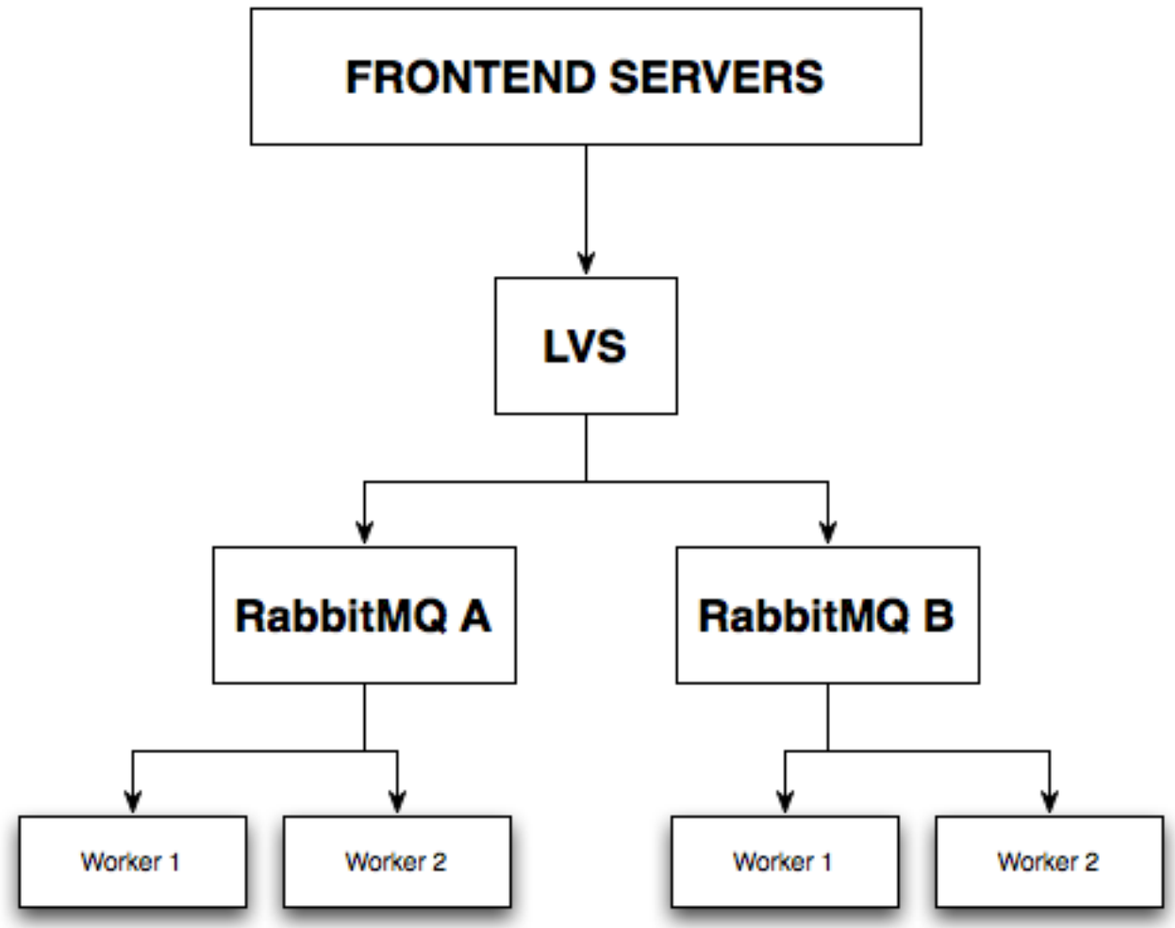
Consumer Code

```
$conn = new AMQPConnection(HOST, PORT, USER, PASS, VHOST);
$channel = $conn->channel();

$channel->exchange_declare('logs', 'topic', false,
                true, false);
$channel->queue_declare('server1-logs', false, true,
                false, false);
$channel->queue_bind('server1-logs', 'logs', 'server1.#');
```

Consumer Code

```
$conn = new AMQPConnection(HOST, PORT, USER, PASS, VHOST);
$channel = $conn->channel();

$channel->exchange_declare('logs', 'topic', false,
                true, false);
$channel->queue_declare('error-logs', false, true,
                false, false);
$channel->queue_bind('error-logs', 'logs', '#.error');
```

One Setup for HA



Questions?

Thanks!

Alvaro Videla

http://twitter.com/old_sound

http://www.slideshare.net/old_sound