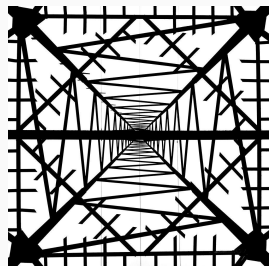


Les serveurs de Messages

Meetup Symfony 13 décembre 2017 - LabSud



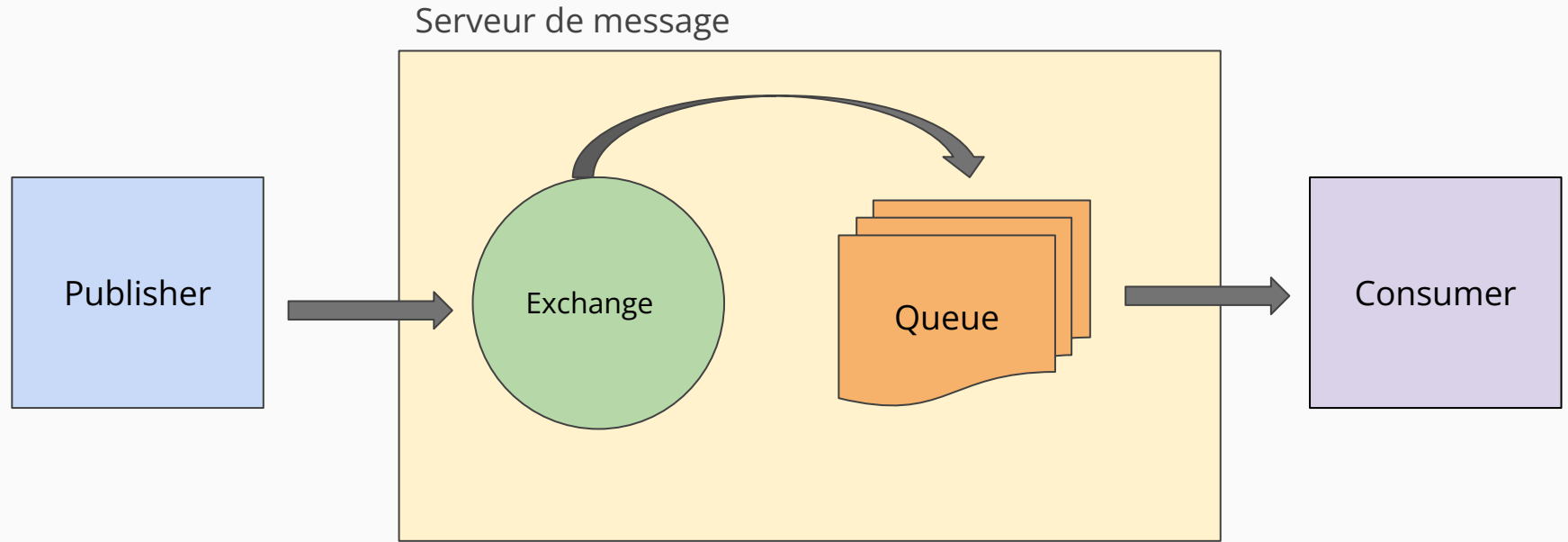
Julien Vinber

@julienVinber sur Twitter
julien@vinber.fr

Développeur PHP / Symfony pour
2S2I solution

Principe

Principe



Et alors?

Asynchrone

Publication rapide

Conçu pour en prendre plein la gueule.

Performance



User: queue
RabbitMQ 3.2.3, Erlang R15B01 [Log out](#)

Overview Connections Channels Exchanges Queues Admin

Overview

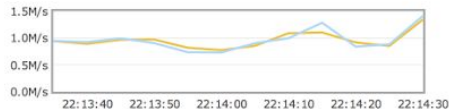
Totals

Queued messages (chart: last minute) (?)



Ready 2,343 msg
Unacknowledged 0 msg
Total 2,343 msg

Message rates (chart: last minute) (?)



Publish 1,345,531/s
Deliver (noack) 1,413,840/s

Global counts (?)

Connections: 12690

Channels: 12690

Exchanges: 194

Queues: 186

Consumers: 10304

Nodes

Name	File descriptors (?)	Socket descriptors (?)	Erlang processes	Memory	Disk space	Uptime	Type
rabbit@b-rabbitmq-queue-1te2	445 262144 available	395 235837 available	4594 1048576 available	252MB 12GB high watermark	6.2GB 48MB low watermark	1h 33m	RAM

<https://content.pivotal.io/blog/rabbitmq-hits-one-million-messages-per-second-on-google-compute-engine>

Des protocoles

AMQP

OASIS 2012



Bien connue et utilisée avec PHP.

Plutôt orienter communication entre logiciel.

Fonctionnalité riche.

MQTT

OASIS 2014



Orienté vers IoT (MQ Telemetry Transport)

Protocole léger

Utiliser par Facebook Messenger

Cas d'utilisation

Cas d'utilisation

- Lancer des traitements de manière asynchrone, exemple envoyer des mail
- Lancer des traitements par lots, exemple l'enregistrement de log en base de données.
- Communication entre application
 - Sans passer par la lourdeur d'une base de données.
 - Sans être obligé d'être 100% up comme une API
 - Sans que l'application soit eux-mêmes des serveurs
- Séparer une application en micro tâche avec des temporalités différentes. (en PHP compensé du multi-thread par exemple)
- Gérer de la scalabilité différente entre ces micros tâche.
- Envoyer des informations en consommant le moins de ressource possible pour l'IoT

Exemple de Code AMQP

php-amqplib/rabbitmq-bundle : config.yml

```
old_sound_rabbit_mq:
  connections:
    default:
      url: 'amqp://guest:password@localhost:5672/vhost?lazy=1&connection_timeout=6'
  producers:
    upload_picture:
      connection:      default
      exchange_options: {name: 'upload-picture', type: direct}
      service_alias:    my_app_service # no alias by default
  consumers:
    upload_picture:
      connection:      default
      exchange_options: {name: 'upload-picture', type: direct}
      queue_options:    {name: 'upload-picture'}
      callback:          upload_picture_service
```

php-amqplib/rabbitmq-bundle : Publisher

```
public function indexAction($name)
{
    $msg = array('user_id' => 1235, 'image_path' => '/path/to/new/pic.png');
    $this->get('old_sound_rabbit_mq.upload_picture_producer')->publish(serialize($msg));
}
```

php-amqplib/rabbitmq-bundle : Subscriber

```
class UploadPictureConsumer implements ConsumerInterface
{
    public function execute(AMQPMessage $msg)
    {
        $isUploadSuccess = someUploadPictureMethod();
        if (!$isUploadSuccess) {
            return false;
        }
    }
}
```

En situation avec MQTT