DRUPALCON VIENNA

26-29 SEPTEMBER 2017

Drupal 8 in a microservices world

Luca Lusso

DevOps - https://events.drupal.org/vienna2017/tracks#devops

Luca Lusso
Drupal developer

Maintainer of Webprofiler (part of Devel suite), Monolog, XHProf, ...

Developer for Wellnet

Teacher for corsidrupal.it

Lead developer of composy.io

Docker and Go **enthusiast**

@lussoluca - drupal.org/u/lussoluca

Drupal 8 in a microservices architecture

The PHP language isn't the best choice to every computation problem.
SQL isn't the best choice to every storage/retrieval problem

Maybe somewhere on the Internet already exists a service that meets our needs

Drupal 8 could be a part of a more complex and distributed system where the different components communicate (mainly) through HTTP

Drupal 8 in a microservices architecture

On this presentation we'll analyse a system we have developed to solve a common problem using a microservices architecture

Problem: Composer is difficult to setup and learn

Composer requires some degree of knowledge to be used correctly with Drupal 8 but it is the recommended (IMHO the only correct) method to install and manage PHP dependencies

It could be very useful if an user can simply choose the modules and themes he wants to be included in a Drupal 8 website and just push a button to get it build automatically

The problem we want to solve: create a SaaS to configure and run Composer remotely

Solution: a service to run Composer remotely

We've build such a service in a microservices way, using Drupal 8 as a frontend of a more complex system:

10 x Docker containers
4 x Go programs
2 x RabbitMQ queues
1 x Amazon Elasticsearch Service
1 x Redis
1 x Drupal 8

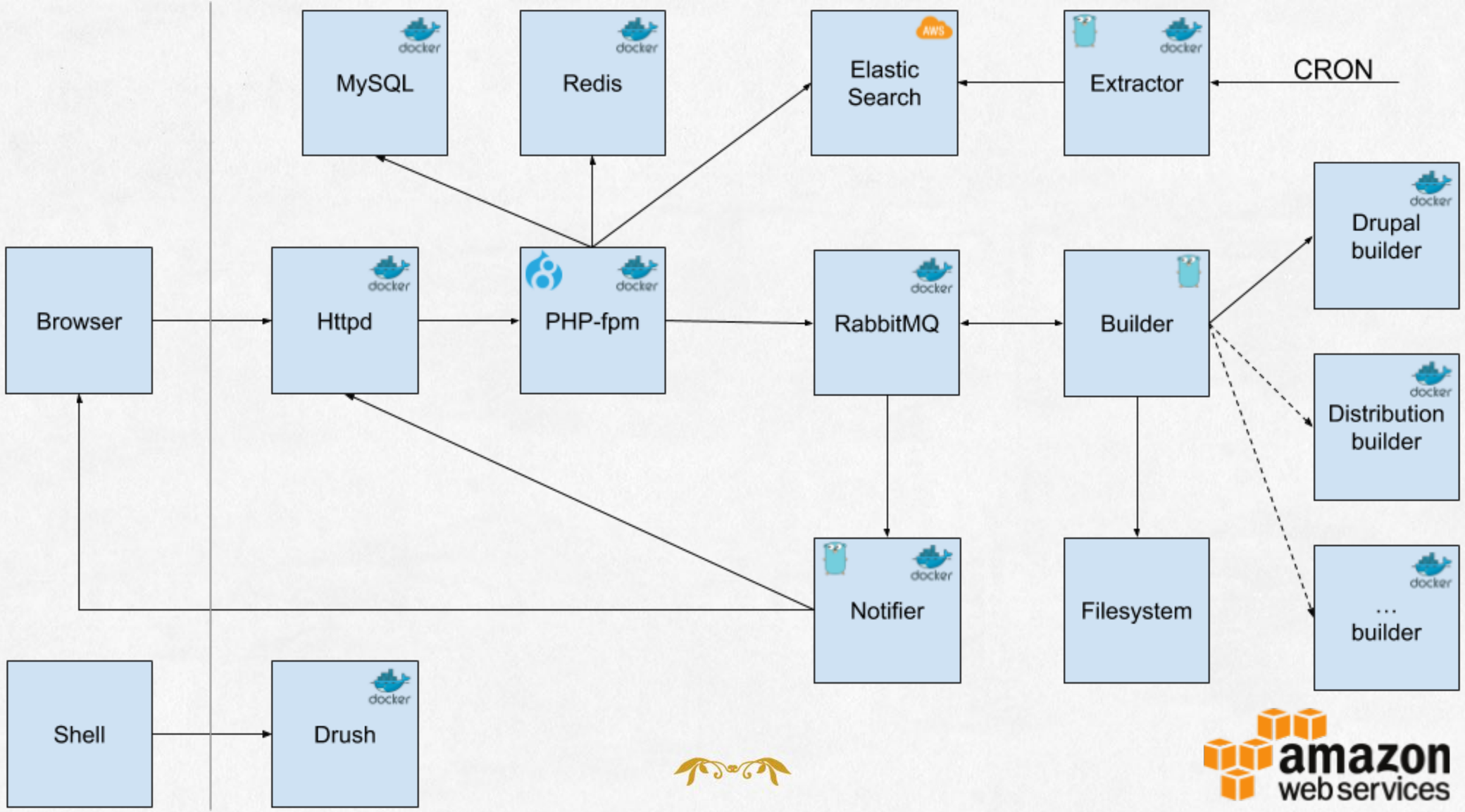
Solution: a service to run Composer remotely

10 x Docker containers
4 x Go programs
2 x RabbitMQ queues

1 x Amazon Elasticsearch Service
1 x Redis
1 x Drupal 8

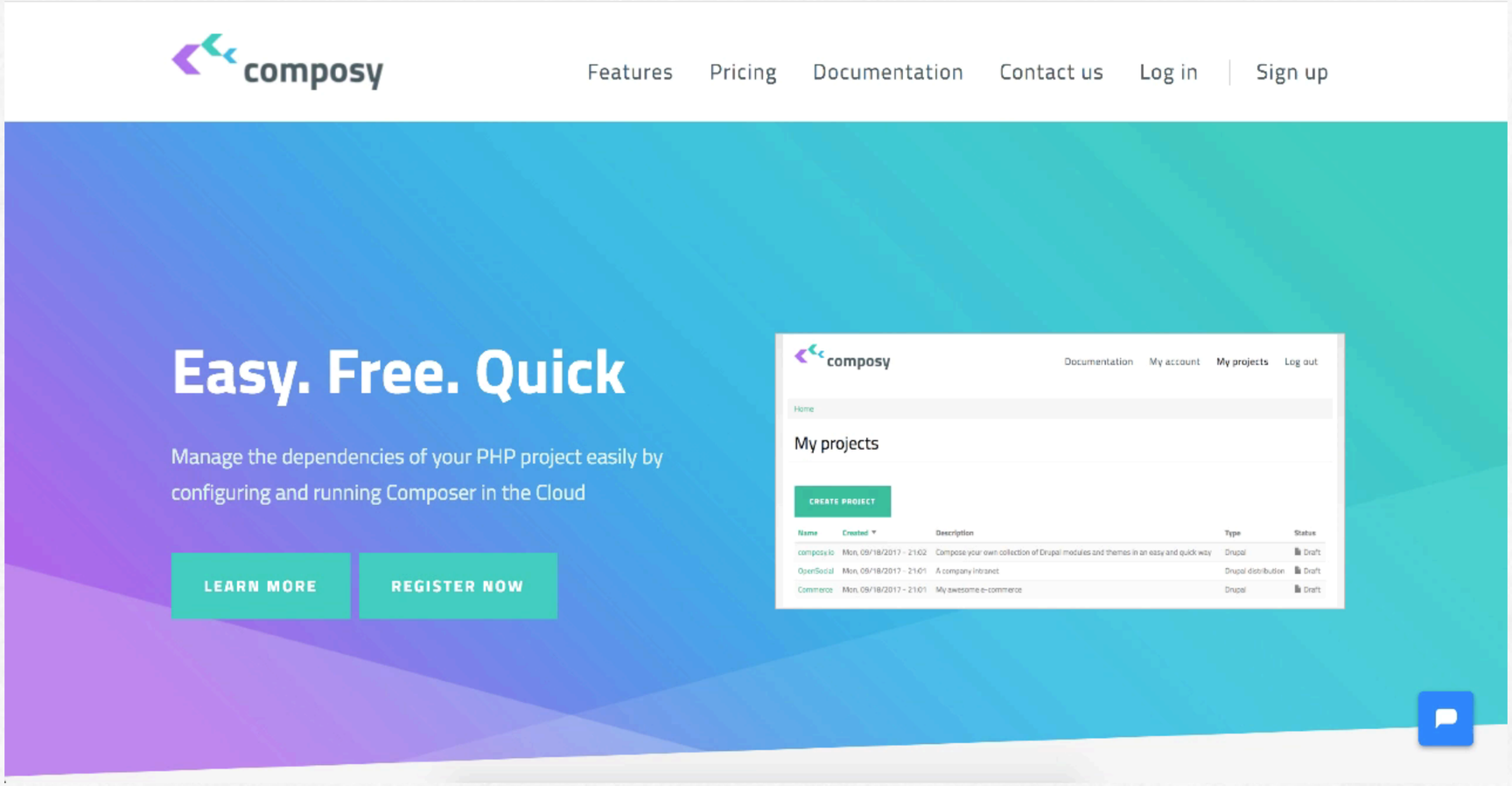
As you see those aren't a lot of things. In effect we built the first working demo in a couple of weeks.
You don't have to think at Netflix, microservices are useful also at a (way more) smaller scale

Solution: a service to run Composer remotely



amazon web services

Solution: a service to run Composer remotely



https://www.youtube.com/watch?v=Zxx6WX6aSHo

DRUPALCON VIENNA

RabbitMQ

RabbitMQ

RabbitMQ is an open source message broker software (sometimes called message-oriented middleware) that implements the Advanced Message Queuing Protocol (AMQP) - *Wikipedia*

https://www.rabbitmq.com

It allows two (or more) microservices to communicate asynchronously by sending messages in a publisher-subscriber model

RabbitMQ

Drupal should delegate long-running tasks or tasks that are more easy/performant to be written in other languages/technologies

We can use RabbitMQ as middleware between our microservices, just let Drupal post a message to a queue and let some other process to receive the message and perform the task

In this way the UX of the Drupal frontend could be better (no wait for the task to be completed) and the external process could send back the results to the client using REST or Websocket (more on this later)

RabbitMQ

In the next example we will define a messages producer as PHP code (in a Drupal custom module) and a messages consumer as a Go process

RabbitMQ

In RabbitMQ we could have different virtualhost, each of them have multiple exchangers that receive messages from channels and dispatch them to queues based on a routing key. (https://www.rabbitmq.com/tutorials/amqp-concepts.html)

In the next example we will use a single virtual host (“/“) and the default exchange (the *direct exchange*) that dispatch all messages sent to a routing key to the queue with the same name (called *builds* in the examples)

RabbitMQ

We need an external PHP library to communicate via AMQP with a RabbitMQ server and, of course, we want to use Composer to manage our dependencies

So in a custom module we have to create a composer.json file with all the required dependencies

RabbitMQ - PHP side

```
1. {
2.    "name": "drupal/custom",
3.    "type": "drupal-module",
4.    "description": "Provides an interface to build Composer projects remotely.",
5.    "require": {
6.        "php-amqplib/php-amqplib": "2.6.3"
7.    }
8. }
```

RabbitMQ - PHP side

```
1. $queue = 'builds';
2. $message_body = [
3.   'type' => 'drupal',
4.   'name' => 'Project name',
5.   'core_version' => '8.4.0',
6.   'path' => '…',
7. ];
```

RabbitMQ - PHP side

```php
$connection = new AMQPStreamConnection('hostname', 5672, 'user', 'pass', '/');
$channel = $connection->channel();
$channel->queue_declare($queue, FALSE, TRUE, FALSE, FALSE);

$message = new AMQPMessage(
  $message_body, array(
    'content_type' => 'text/plain',
    'delivery_mode' => AMQPMessage::DELIVERY_MODE_PERSISTENT,
  )
);

$exchange = '';
$routingKey = $queue;
$channel->basic_publish($message, $exchange, $routingKey, FALSE, FALSE);
$channel->close();
$connection->close();
```

Go

Go (often referred to as golang) is a free and open source programming language created at Google in 2007 [...]. It is a compiled, statically typed language in the tradition of Algol and C - *Wikipedia*

https://golang.org

Well suited for CLI applications, concurrent applications, servers, ...
Just download the standard toolchain and compile the code in a single statically linked binary file that contains your code, all the dependencies and the Go runtime

Go

Why Go over Node, Java or Python?

- compiled in a single binary file that runs directly to the host machine, there is no need for any dependency
- concurrent by design
- strongly typed but doesn’t need a rigid structure of Classes and Interfaces
- very opinionated
- not so difficult to learn

Go

The Go toolchain is very opinionated and provides standard ways to perform common tasks

go fmt to format code with the Go coding standard.
go build to compile packages and dependencies.
[...]
go get to download and install packages and dependencies.

RabbitMQ - Go side

```
go get github.com/streadway/amqp
```

RabbitMQ - Go side

```
1. type message struct {
2.   Type          string
3.   Name          string
4.   CoreVersion   string `json:"core_version"`
5.   Path          string
6. }
```

RabbitMQ - Go side

```
1.  conn, err := amqp.Dial("amqp://user:pass@hostname:5672")
2.  if err != nil { return err }
3.
4.  ch, err := conn.Channel()
5.  if err != nil { return err }
6.
7.  _, err = ch.QueueDeclare("builds", true, false, false, false, nil)
8.  if err != nil { return err }
9.
10. msgs, err := ch.Consume("builds", "", true, false, false, false, nil)
11. if err != nil { return err }
12.
13. for msg := range msgs {
14.   var m message
15.   err = json.Unmarshal(msg.Body, &m)
16.   if err != nil { return err }
17.   // the m struct now contains the message received
18. }
```

DRUPALCON VIENNA

Elasticserch as common storage

ElasticSearch

Elasticsearch is a search engine based on Lucene. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents - *Wikipedia*

https://www.elastic.co/products/elasticsearch

It is useful as a common data storage between microservices where indexing and searching capabilities are needed

ElasticSearch

In the next example we will define a Go code that store data in Elasticsearch and a PHP code (in a Drupal custom module) that read the data from Elasticsearch

ElasticSearch - Go side

```
go get gopkg.in/olivere/elastic.v5
```

ElasticSearch - Go side

```
1. url := "https://[...].eu-west-1.es.amazonaws.com"
2. indexName := "extensions"
3.
4. client, err := elastic.NewClient(elastic.SetURL(url), elastic.SetSniff(false))
5. if err != nil { panic(err) }
6.
7. _, err := client.Index().
8.    Index(indexName).
9.    Type("extension").
10.     Id("drupal/devel_8.x-1.0").
11.     BodyJson("{Name: \"Devel\", Version: \"8.x-1.0\", Package: \"drupal/devel\"}").
12.     Refresh("true").
13.     Do(context.TODO())
14. if err != nil {
15.    log.Errorf("Error in saving document %s: %e", documentId, err)
16. }
```

ElasticSearch - PHP side

```
1. {
2.   "name": "drupal/custom",
3.   "type": "drupal-module",
4.   "description": "Provides an interface to build Composer projects remotely.",
5.   "require": {
6.     "php-amqplib/php-amqplib": "2.6.3",
7.     "elasticsearch/elasticsearch": "5.3.0"
8.   }
9. }
```

ElasticSearch - PHP side

```
1.  $client = ClientBuilder::fromConfig(
2.   [
3.     'hosts' => ["https://[...].eu-west-1.es.amazonaws.com"],
4.     'retries' => 2,
5.     'handler' => ClientBuilder::multiHandler(),
6.   ]
7.  );
8.
9.  $params = [
10.   'index' => 'extensions',
11.   'type' => 'extension',
12.   'body' => ['query'
13.     =>['bool'=>['must'=>['query_string'=>['query'=>'Name:Devel']]]]],
14. ];
```

ElasticSearch - PHP side

```php
$results = [];

try {
 $response = $client->search($params);

 foreach ($response['hits']['hits'] as $hit) {
   $results[] = [
     'label' => $hit['_source']['Name'],
     'value' => $hit['_source']['Package'],
    ];
 }
} catch (\Exception $e) {
    return $results;
}

return $results;
```

ElasticSearch

ElasticSearch is useful to produce data for a Drupal autocomplete field

Define a new controller that reads the q argument from the URL, performs a query to ElasticSearch and returns a JsonResponse like:

```
{"label": "Devel", "value": "drupal/devel"}
```

Define a new route in *.routing.yml to map an URL to the controller

Define a form text field to have the #autocomplete_route_name key equal to the route name

DRUPALCON VIENNA

Expose and consume REST

REST

Representational state transfer (REST) or RESTful web services is a way of providing interoperability between computer systems on the Internet. REST-compliant Web services allow requesting systems to access and manipulate textual representations of Web resources using a uniform and predefined set of stateless operations - *Wikipedia*

Drupal 8 API-first initiative: https://www.drupal.org/node/2757967

REST

In the next example we will define a REST endpoint using Drupal core's capabilities and a Go code to post a message to that endpoint

Expose a REST endpoint - Drupal

https://www.drupal.org/docs/8/api/restful-web-services-api/custom-rest-resources

Create a plugin in a custom module (in *Drupal\[...]\Plugin\rest\resource* namespace)

Define a REST resource config (in a YAML file in module's config/install folder)

Expose a REST endpoint - Drupal

```
/**
 * @RestResource(
 *   id = "project_update_status",
 *   uri_paths = {
 *     "canonical" = "/project/{id}/update_status"
 *   }
 * )
 */
class ProjectResource extends ResourceBase {

  public function patch($id, array $payload) {
    $project = $this->entityTypeManager->getStorage('project')->load($id);
    $key = $payload['Status'] ? Build::SUCCESS : Build::ERROR;
    $this->buildManager->changeStatus($project, $payload['Destination'], $key);

    return new ResourceResponse("Success");
  }

}
```

Expose a REST endpoint - Drupal

```
1.  id: project_update_status
2.  plugin_id: project_update_status
3.  granularity: resource
4.  configuration:
5.    methods:
6.      - PATCH
7.    formats:
8.      - json
9.    authentication:
10.     - basic_auth
```

Expose a REST endpoint - Drupal

GET: retrieve a resource
POST: create a resource
PATCH: update a resource
PUT: replace a resource
DELETE: remove a resource

Extract information from Drupal: GET /entity/project/42
Create information in Drupal: POST /entity/project/42
Ask Drupal to do something: POST /entity/project/42/update_status

Consume a REST endpoint - Go

```
1.  url := "https://example.com/project/42/update_status?_format=json"
2.  payload := "{\"Status\": 2}"
3.
4.  req, _ := http.NewRequest("PATCH", url, strings.NewReader(payload))
5.  req.Header.Add("content-type", "application/json")
6.
7.  resp, err := http.DefaultClient.Do(req)
8.  if err != nil { return err }
9.
10. if resp.StatusCode != 200 { [...] }
11.
12. body, err := ioutil.ReadAll(resp.Body)
13. if err != nil { return err }
14. resp.Body.Close()
15.
16. // body contains the response from the server
```

Consume a REST endpoint - Drupal

Of course we can do the opposite and let Drupal consume some REST resources exposed from a microservice

To do that we can use the http_client service that Drupal provides us. http_client is an instance of the Client class from the Guzzle package

http://docs.guzzlephp.org/en/stable

Or we can use the HTTP Client Manager module (https://www.drupal.org/project/http_client_manager) to leverage the Guzzle Service Descriptions feature

GraphQL- Drupal

A new kid on the block: GraphQL

GraphQL is a data query language developed internally by Facebook in 2012 before being publicly released in 2015. It provides an alternative to REST and ad-hoc web-service architectures - *Wikipedia*

https://www.drupal.org/project/graphql
http://graphql.org

DRUPALCON VIENNA

Realtime websocket notifications

Websocket

WebSocket is designed to be implemented in web browsers and web servers. The WebSocket Protocol is an independent TCP-based protocol. Its only relationship to HTTP is that its handshake is interpreted by HTTP servers as an Upgrade request. The WebSocket protocol enables interaction between a browser and a web server with lower overheads, facilitating real-time data transfer from and to the server - *Wikipedia*

Websocket

In the next example we will define a Javascript code to connect to a remote websocket server implemented in Go

Websocket - server side

```
go get github.com/gorilla/websocket
```

Websocket - server side

```
1.  http.HandleFunc("/ws-endpoint", message)
2.  err = http.ListenAndServe("0.0.0.0:8080",  nil)
3.  if err != nil { [...] }
4.
5.  func message(w http.ResponseWriter, r *http.Request) {
6.    ws, err := upgrader.Upgrade(w, r, nil)
7.    if err != nil { [...] }
8.    defer ws.Close()
9.
10.    ws.WriteMessage(websocket.TextMessage,
11.      []byte(`{"message": "message for the client"}`))
12. }
```

Websocket - client side

```
1.  (function ($, Drupal) {
2.
3.    "use strict";
4.
5.    Drupal.behaviors.websocket = {
6.      attach: function (context, settings) {
7.
8.      var host = settings.webservice.host;
9.      var notifier = new WebSocket("wss://" + host + ":8080/ws-endpoint");
10.     notifier.onmessage = function (event) {
11.       var json = JSON.parse(event.data);
12.       console.log(json.message);
13.     }
14.   }
15.
16.  })(jQuery, Drupal);
```

DRUPALCON VIENNA

Architecture as code

Architecture automation

In microservices architecture, as soon as the system scales, the number of hosts may increase leading to a hard-to-maintain system, where services are scattered across multiple hosts, with each one running multiple services.

Manually managing a microservices architecture would result in an enormous time overhead, since deployment, configuration, and maintenance now extends to each and every service instance and host.

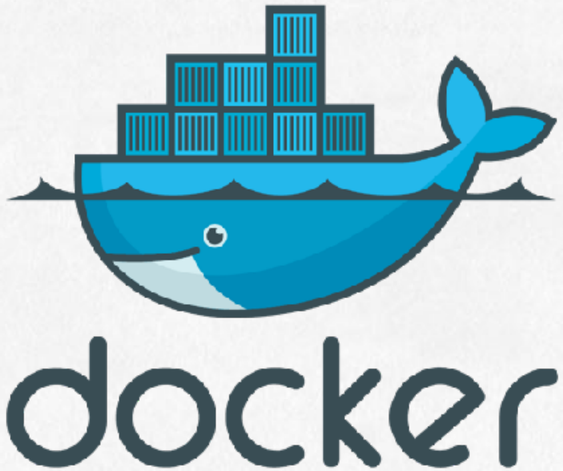
Every time a new service or host is introduced, the system will require an increasing amount of time for manual management.

When standard management activities, such as builds, tests, deployment, configuration, host provisioning and relocation of services are automated, the introduction of new services does not imply a management overhead.

Only maintenance of scripts is required, and developers are expected to manage all the system via automation

From: Microservices: Migration of a Mission Critical System (https://arxiv.org/abs/1704.04173)

Architecture automation



Docker

Docker is a tool that can package an application and its dependencies in a virtual container that can run on any Linux server. This helps enable flexibility and portability on where the application can run, whether on premises, public cloud, private cloud, bare metal, etc - *Wikipedia*

Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a file to configure your application's services. Then, using a single command, you create and start all the services from your configuration - *Docker documentation*

Docker

```
1.  version: "2"
2.
3.  services:
4.
5.    httpd:
6.      image: httpd:2.4
7.      ports:
8.        - 80:80
9.        - 443:443
10.     restart: always
11.     volumes:
12.       - ./httpd-prod.conf:/usr/local/apache2/conf/httpd.conf:ro
13.       - ./fullchain.pem:/etc/ssl/fullchain.pem:ro
14.       - ./privkey.pem:/etc/ssl/privkey.pem:ro
15.       - .:/var/www:ro
```

Docker

```
1.  db:
2.    environment:
3.      - MYSQL_USER=drupal
4.      - MYSQL_DATABASE=drupal
5.      - MYSQL_PASSWORD=drupal
6.      - MYSQL_ROOT_PASSWORD=root
7.    image: mariadb:latest
8.    ports:
9.      - 3306:3306
10.   restart: always
11.   volumes:
12.     - ./custom.cnf:/etc/mysql/conf.d/custom.cnf:ro
13.     - db_data:/var/lib/mysql
```

Docker

```
1. php:
2.   image: wellnetimages/php:7.0.14
3.   restart: always
4.   volumes:
5.     - ./php.ini:/usr/local/etc/php/conf.d/php.ini:ro
6.     - .:/var/www
```

Docker

Container for:

- Httpd
- PHP-FPM
- MySQL
- Redis (for Drupal cache)
- RabbitMQ
- Notifier (Go process system notification via REST and Websocket)
- Drush (to execute sshd)

In dev also:

- Mailhog
- Elasticsearch
- Kibana
- blackfire.io

Ansible

Ansible is an open-source automation engine that automates software provisioning, configuration management, and application deployment - *Wikipedia*

With Docker and Docker Compose it will allow you to describe all your services architecture in code so you can version control it

Ansible

Ansible is agentless, no software is needed on managed machines other than Python and a SSH connection

Ansible reads a list of hosts from an inventory and performs a set of tasks defined in a playbook

Playbook uses modules to describe the operations to be executed on every hosts in the inventory

If Ansible modules are the tools in your workshop, playbooks are your instruction manuals, and your inventory of hosts are your raw material

Ansible

```
1. ---
2. - name: System setup
3.   hosts: all
4.   become: true
5.   become_user: root
```

Ansible

```
1.  tasks:
2.    - name: Install Python setuptools and Docker
3.      yum:
4.        name: "{{ item }}"
5.        state: present
6.      with_items:
7.        - python-setuptools
8.        - docker
9.    - name: Install pip
10.     easy_install: name=pip
11.   - name: Install docker-compose
12.     pip:
13.       name: "docker-compose"
14.       version: 1.15.0
15.       state: present
```

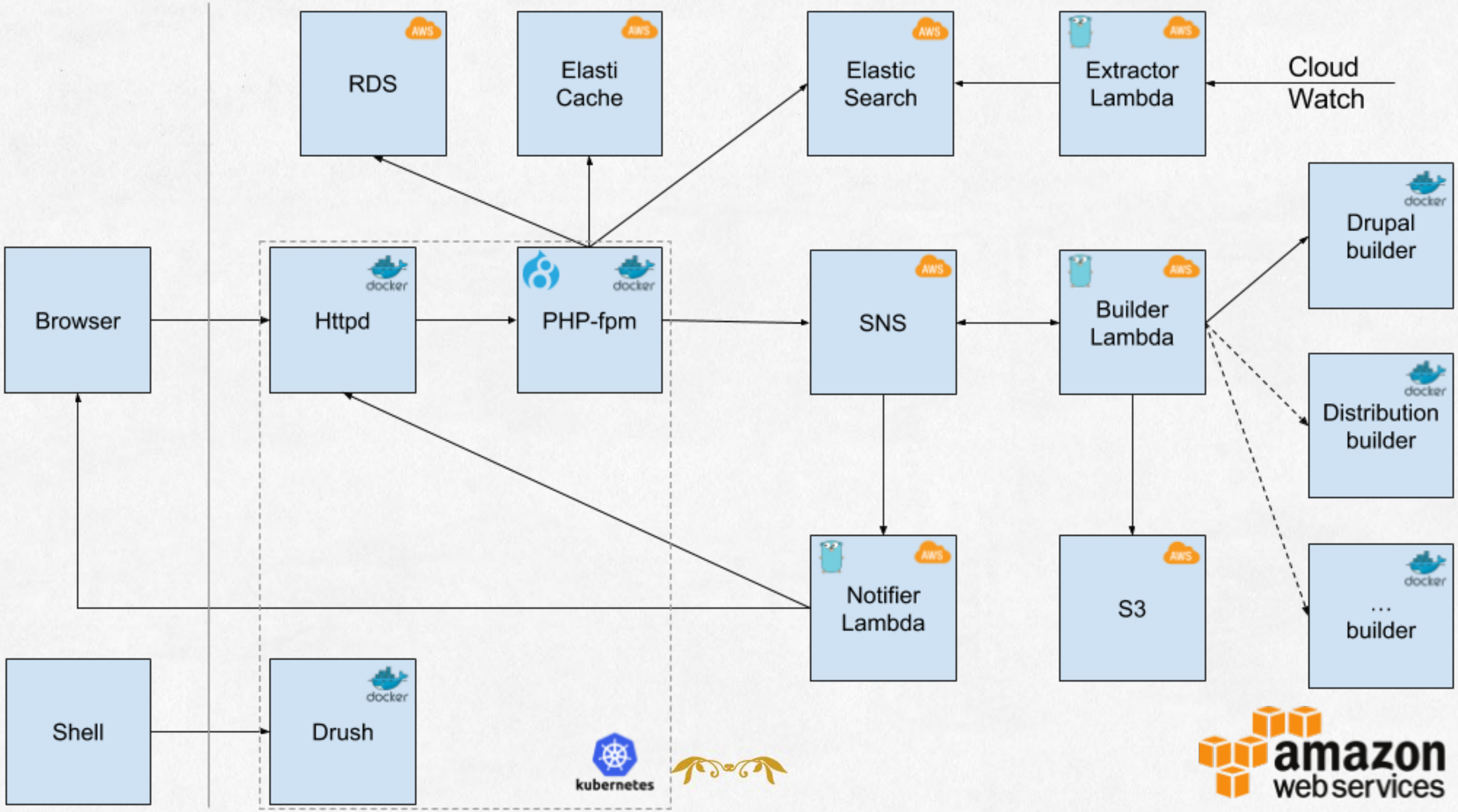
Ansible

```
1. - name: Start Docker
2.   service:
3.     name: docker
4.     state: started
5. - name: Add ec2-user to docker group
6.   user: name=ec2-user
7.         group=docker
```

Ansible

```
1.  - file:
2.      path: "/app"
3.      state: directory
4.  - name: Upload docker-compose files
5.    synchronize:
6.      src: "compose"
7.      dest: "/app"
8.  - name: Start docker compose
9.    docker_service:
10.     project_src: "/app/compose"
11.     restarted: true
```

Next step: use other AWS services



DRUPALCON VIENNA

Takeaway

Takeaway

1. don’t try to do anything with Drupal
2. use Drupal for its best features: CMS, user management, template, …
3. try Go
4. start small (all services in Docker containers on one server)
5. put your infrastructure under version control

composy

composy.io

compose core, modules and themes

download a zip with all inside

build distributions (Contenta, Lightning, Thunder, OpenSocial, ...)

choose folder layout, standard or flat

subscribe for automatic updates

Wellnet is hiring!

- Drupal 7/8
- Go
- Java
- Javascript (Node, React, ...)
- AWS

info@wellnet.it

DRUPALCON VIENNA

JOIN US FOR CONTRIBUTION SPRINT

Friday, September 29, 2017

Mentored Core Spint

9:00-12:00
Room: Stolz 2

First time Sprinter Workshop

9:00-12:00
Room: Lehgar 1 - Lehar 2

General sprint

9:00-12:00
Room: Mall

#drupalsprints


DRUPALCON VIENNA

WHAT DID YOU THINK?

Locate this session at the DrupalCon Vienna website:

http://vienna2017.drupal.org/schedule

Take the survey!

https://www.surveymonkey.com/r/drupalconvienna


THANK YOU!