PaaSTA - Build, Deploy, Connect, and Monitor Services

Qu'est-ce que PaaSTA ?

PaaSTA est un PaaS développé par la société Yelp qu'il utilise actuellement pour faire tourner leur entreprise. Il permet aux développeurs de la boîte de déclarer, dans des fichiers de configuration, la manière dont ils veulent que les codes ayant été produits doivent être compilé, déployé, acheminé et surveillé.

PaaSTA est donc un système automatisé permettant un gain de temps considérable et limitant les démarches manuelles.

Comment PaaSTA fonctionne ?

PaaSTA est un assemblage de divers modules disponible en open-source qui, à chacun, réalisent différentes tâches :

- **Docker** pour l'acheminement de code et le containment;
- **Apache Mesos** pour l'exécution de code et d'ordonnancement (court conteneurs Docker);
- **Mesosphere's Marathon** de mésosphère la gestion des services à long fonctionnant;
- **Chronos** pour exécuter les choses sur une minuterie (lots de nuit);
- **SmartStack** pour l'enregistrement de service et de découverte;
- **Sensu** pour la surveillance / alertes;
- **Jenkins** pour le déploiement continu.

Problem	PaaSTA Solution
Code containerizer	Docker
Scheduling	Mesos + Marathon
Service Discovery	SmartStack
Monitoring	Sensu
Workflow	Jenkins or CLI + soa-configs

Le développeur commencera par mettre en place un environnement de travail afin d'installer leur service à exécuter dans un container. Cela permet une certaine flexibilité concernant le runtime qu'il pourra configurer à selon ses besoins pour un meilleur confort de travail et fourni à l'infrastructure un "binaire" à exécuter peu importe le système d'exploitation utilisé par la machine de l'utilisateur.

Une fois l'image Docker créée, PaaSTA ou Jenkins se chargera de la mettre en ligne et de l'étiqueter comme étant prête au déploiement.

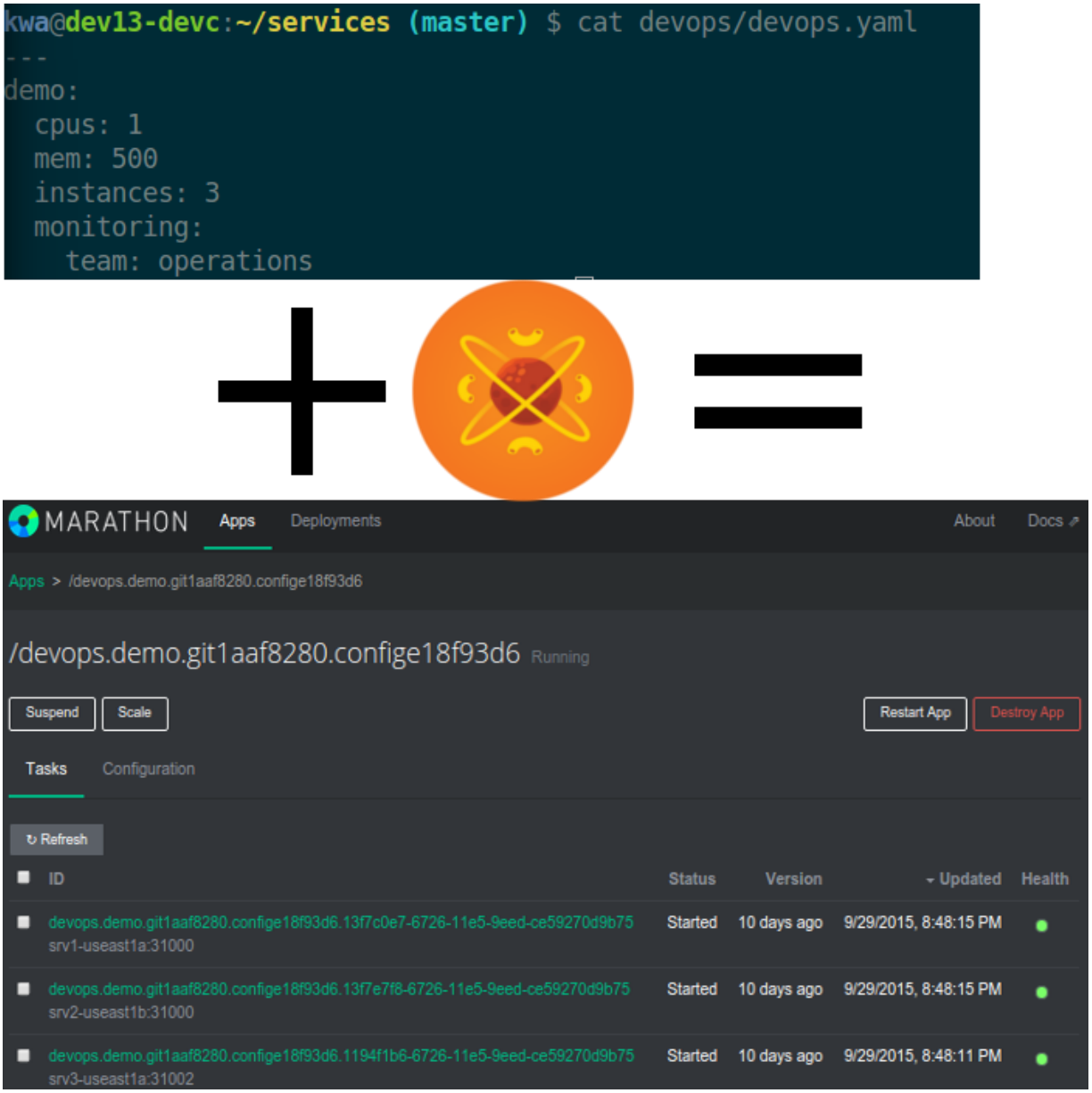
Que faire une fois que l'image Docker est créée ?

Une fois que l'image Docker est prête, le développeur doit déclarer comment et où il veut que son service soit exécuté. PaaSTA fourni un DSL pour le décrire.

Voici un exemple montrant comment définir un service :

```
---
demo:
  cpus: 1
  mem: 500
  instances: 10
  monitoring:
    team: operations
```

L'image est prête et un code décrivant le comportement du service est mis en place. PaaSTA va donc à présent s'occuper du reste. Grâce à Mesos, Marathon et Chronos, PaaSTA peut communiquer aisément avec ses API afin de répondre aux besoins des développeurs. PaaSTA synchronise constamment les déclarations du service avec l'API actuelle.



PaaSTA essayera de prendre en charge les services déclarés une fois exécuter pour les remettre à leur état original défini dans les fichiers de configuration si un problème devait se présenter.

Mon service est lancé. Comment le retrouver ?

Sur n'importe quelle plateforme, le Service Discovery est un outil remarquable. Si un service est déployé de manière automatique, il aura besoin d'être découvert de manière automatique aussi.

C'est là qu'intervient SmartStack. Il prend un fichier de configuration pour référence et configure la répartition des charges en local. PaaSTA guide ensuite SmartStack pour lui indiquer quels services il doit afficher en provenance de Docker, Mesos ou n'importe quelle autre technologie à adopter.

Un Service Discovery ayant un mécanisme fléxible utilisant des éléments "fastidieux" est essentiel à long terme pour l'évolution d'une infrastructure saine. SmartStack a un comportement fermement contrôlable car il finira par lire tous les fichiers de configurations qui seront créés.