OpenStack

Méthodes pour supporter un développement distribué à grande échelle

Artom Lifshitz

OpenStack

- Projet libre d’infrastructure-service/plate-forme “cloud”
- Créé en 2010 par RackSpace et la NASA
- 16000 contributeurs dans 150 pays
- 2 versions stables par année

Comment font-ils?

- Développement piloté par tests (TDD/Test driven development)
- Énorme système d’intégration continue (CI/Continuous integration)

Plan de la présentation

1. Historique
2. TDD et CI
3. Leçons à en tirer

Historique

- Années cinquante: traitement par lots
- Années soixante: temps partagé
- CTSS (Compatible Time Sharing System)
- Projet MAC

Projet MAC

- Machine Aided Cognition
- Multiple Access Computer
- “Notre objectif était l’informatique à la demande” - Bob Fano, un des fondateurs du projet

Projet MAC

- Premier objectif: disponibilité et fiabilité
- "Le système MAC ne marchait pas hier soir. On m’a dit qu’il marcherait ce matin. Il ne marche pas. Que ce passe-t-il ?"
- "On a réussi !" s’exclame Fano
- Informatique à la demande

Informatique à la demande

- Utility computing
- “Si la sorte d’ordinateur que je préconise devient l’ordinateur du futur, alors l’informatique pourrait un jour devenir un service public tout comme le téléphone. L’informatique à la demande serait alors la base d’une industrie importante.” - John McCarthy, 1961

Amazon Web Services

- EC2: location de machines virtuelles
- Gestion via interface web et/ou API
- Utilisateur paye pour l'utilisation, non un montant fixe par mois
- Prédiction de McCarthy devenue réalité

Amazon Web Services

- Au départ: besoin interne
- Les développeurs se heurtaient au “clergé du matériel” - Chris Brown, directeur de développement
- Repenser l’infrastructure pour donner plus de liberté aux développeurs

Amazon Web Services

Mandat donné par Jeff Bezos:

1. Toutes les équipes doivent désormais exposer leurs données et leurs fonctionnalités à travers une interface de service.
2. Les communications entre les équipes doivent se faire à l'aide des ces interfaces.
3. Aucune autre communication entre les équipes n'est permise. Pas d'accès direct à un magasin de données, pas de mémoire partagée, pas de portes dérobées. La seule communication permise est l'appel aux interfaces de service via le réseau.
4. La technologie utilisée n'a pas d'importance. Que ce soit HTTP, Corba, Pubsub ou des protocoles sur mesure, cela n'a aucune importance.
5. Toutes les interfaces de services, sans exception, doivent être conçues de A à Z de façon à être externalisables. Cela signifie que toutes les équipes doivent être en mesure d'exposer leurs interfaces au monde extérieur.

Amazon Web Services

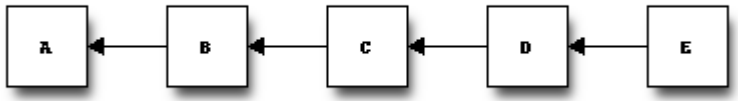
- SOA + virtualization = IaaS

OpenStack - Intégration continue

- Référentiel n’est jamais brisé
- Github
- Remote Gerrit
- https://review.openstack.org/#/c/49555/
- Zuul et Jenkins

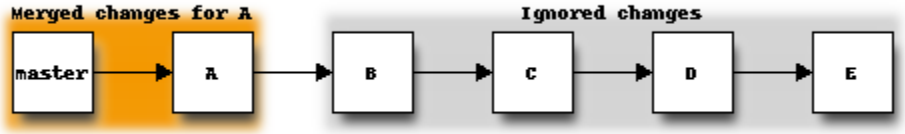
Zuul

- Système de gestion des tâches de test et de leurs interdépendances
- Exemple: les changements A, B, C, D et E sont proposés

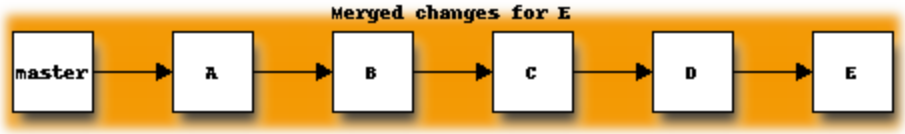


Zuul

Tâches de test pour A:

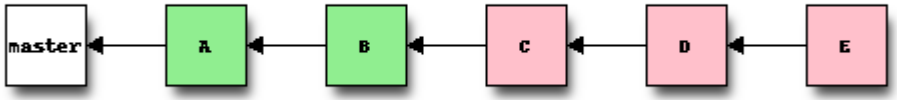


Tâches de test pour E:

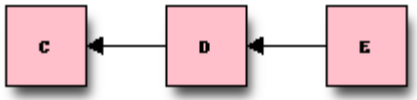


Zuul

Les tests de A et B réussissent, C, D et E échouent:

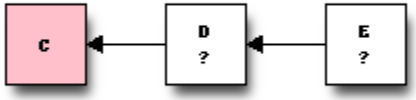


Fusionner A et B, il reste dans la file des tâches:

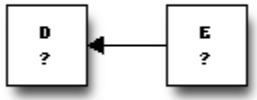


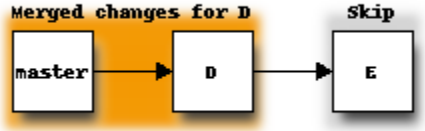
Zuul

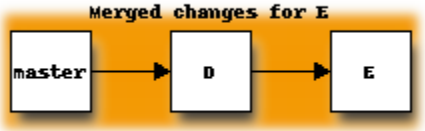
C échoue:



Rapporter que C échoue, continuer les tests pour D et E:







Leçons à tirer

- Zuul marche!
- Zuul est unique à OpenStack
- Conséquence de l’échelle et du rythme de développement d’OpenStack
- Pour justifier sa complexité il faut un projet aussi complexe qu’OpenStack