Continuous Deployment

LA « BÊTA PERPÉTUELLE »

Le principe de la « bêta perpétuelle » a été introduit pour la première fois par Tim O'Reilly dans le manifeste du Web 2.0, qui expose le principe selon lequel:

> *« Les utilisateurs doivent être considérés comme des co-développeurs, en suivant les principes Open Source (...). Avec le Web 2.0, ce principe évolue vers une position plus radicale, la bêta perpétuelle, dans laquelle le produit est développé de manière ouverte, avec de nouvelles fonctionnalités offertes sur une base mensuelle, hebdomadaire, ou même quotidienne. »*

Le terme « bêta perpétuelle » désigne le fait que notre application n'est jamais finalisée. Elle s'absout des contraintes habituelles liées au cycle de développement en « release » au profit d'une livraison en continu des nouvelles fonctionnalités.

Release early, release often

Derrière ce nouveau concept se cache un concept déjà bien en place chez les agilistes (du point de vue de l'itération

courte) et dans le monde de l'Open Source (du point de vue de la récolte continue du feedback), le « Release early, release often », traduit en français par « Publiez tôt, publiez souvent ».

Cette pratique a été décrite par Eric Steven Raymond dans "La cathédrale et le bazar" où il formulait explicitement:

> *« Publiez tôt. Publiez souvent. Et écoutez vos clients »*

Cette méthodologie vise à réduire les temps de développement et améliorer l'implication de l'utilisateur dans la conception du logiciel afin de créer un produit correspondant à ces attentes et ainsi éviter la création d'un logiciel que personne n'utilisera.

Les services en ligne (Software As A Service)

Le concept de « bêta perpétuelle » a été rendu possible grâce au Cloud Computing et plus particulièrement à la généralisation du service en ligne ou « Software As A Service ». L'hébergement de l'application par l'éditeur permet d'absoudre ce dernier au traditionnel cycle de déploiement d'un logiciel et de ne gérer qu'une seule version de son application.

Les services en ligne sont continuellement mise à jour sans pour autant en informer l'utilisateur. Les nouvelles fonctionnalités, découverte au fur et à mesure par l'utilisateur, permettent un apprentissage progressif des nouveautés applicatives.

La « Customer driven roadmap »

L'hébergement de l'application sur serveur offre à l'éditeur une maîtrise total de sa plateforme de production. Il peut ainsi mettre en place des sondes analytiques afin de récolter des informations sur l'usage de notre application et l'accueil réservé à nos nouvelles fonctionnalités par l'utilisateur.

Les pré-requis

La mise en place d'une stratégie de « bêta perpétuelle » requiert certains pré-requis pour en garantir le succès:

- une intégration continu,
- une livraison continue,
- un déploiement continu,
- une stratégie de type « One click deployment / Rollback » pour une restauration rapide de l'application au dernier état stable.

Conclusion

Le concept de la « bêta perpétuelle » est présent chez de nombreux géants du Web tel que Google, Facebook, Amazon, Twitter, Flickr… Peu en font mention dû à la mauvaise image du terme « bêta », qui pour la conscience collective, se réfère à un produit non fini et peu fiable.

Prenons en exemple Gmail, la boite aux lettres mails développé par Google, qui jusqu'en 2009 intégrait la mention « bêta » dans son logo. De petites fonctionnalités unitaires sont fréquemment proposés aux utilisateurs. En fonction de leur

niveau d'adoption Google les intègrent ou non à la version standard de son service.

ZERO DOWNTIME DEPLOYMENT

Les patterns

Le « Zero Downtime Deployment » (ZDD) s'articule autour de trois grands patterns, le « Blue/Green Deployment », le « Canary Release » et le « Dark Launch ».

Blue/Green Deployment

Le « Blue/Green Deployment » est le pattern de base du ZDD. L'application, hébergée sur au moins deux chaines applicatives, déploie sa version N+1 sur une des chaînes (ou plusieurs) tandis que le service en ligne est maintenu en version N sur les autres chaines applicatives.

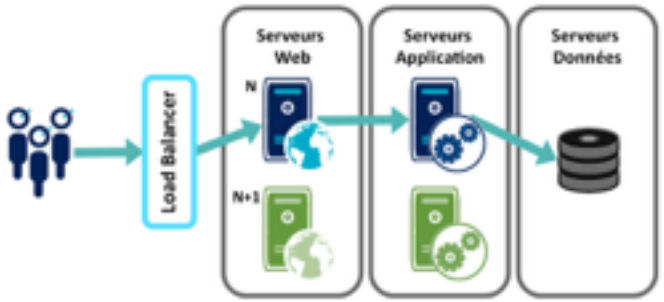


Schéma de pattern Blue/Green Deployment

Canary Release

Une fois le déploiement effectué, notre nouvelle version doit être testé par une population restreinte d'utilisateurs. Le « Canary Release » confronte la version N+1 à une niche d'utilisateurs cibles tandis que la version N reste accessible à la majorité des utilisateurs.

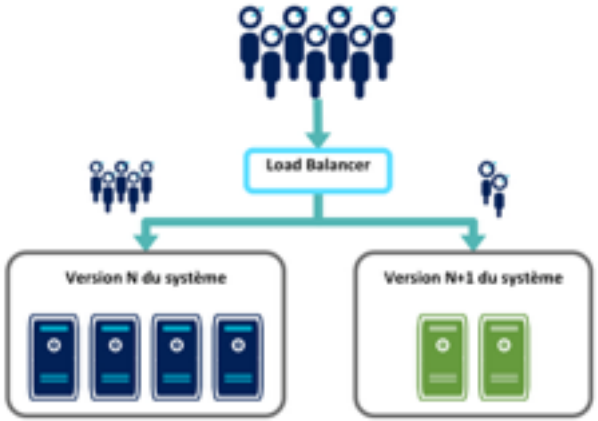


Schéma de pattern Blue/Green Deployment

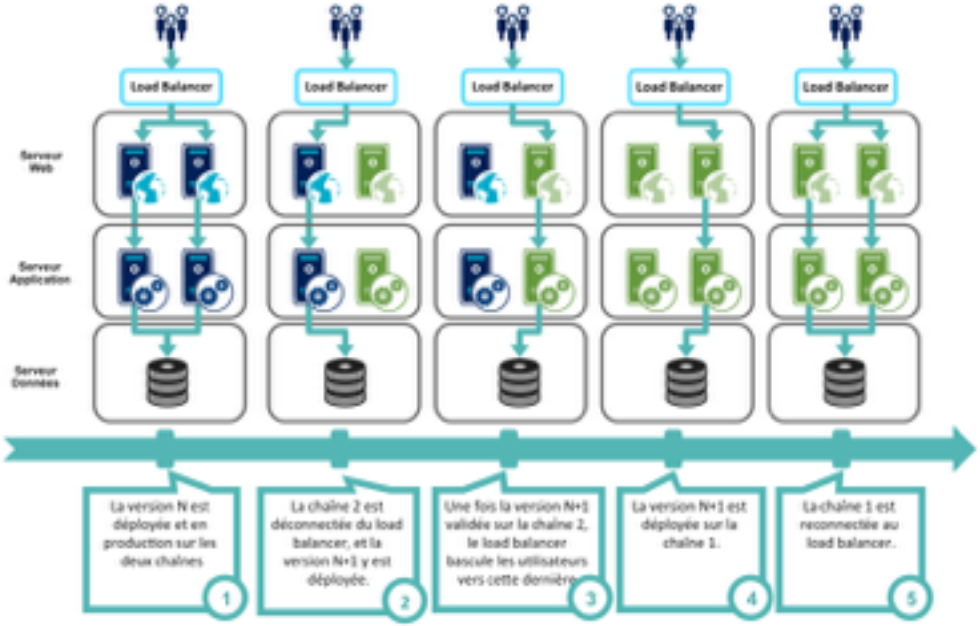
Dark Launch

La dernière étape du ZDD est le test de charge de notre nouvelle application. Le « Dark Launch » stimule progressivement le traffic généré par l'utilisateur afin de valider les performances et la scalabilité de notre plateforme.

La stimulation progressive du traffic permet de préparer et d'optimiser au mieux notre plateforme afin que la mise en production finale de notre application se déroule sans accro.

La mise en oeuvre

Le load balancer



Les sessions

La modification du schéma de base de données

FLIPPING FEATURE