La transformation digitale

Vers le continuous delivery ?

Le contexte

- Contexte concurrentiel agressif

- Besoin de réduire le Time To Market
  - Du design à la production

Phase 1

Le constat

On ouvre le capot

Le développement

- Equipes agiles
  - Objectif: autonomie, responsabilisation, maturité…
- L'outillage
  - Tendance avancée à l'automatisation pour l'intégration continue
- Les environnements
  - Pas facile d'être isoprod
  - Casse-tête des données de test
  - Visibilité limitée sur la prod (logs, métriques)
- Livrables hétérogènes (tar.gz, jar) en fonction de la techno

Les tests & qualité

- Manuel
- Prise en compte tardive des besoins de ressources
- Pattern en cornet de glace
- Efficacité pas au rendez-vous

La production

- Exploitation
  - Déployer / Paramétrer / Scheduler : des outils mystérieux
  - Monitoring & Alerte : pas très transparents, alertes mal perçue, pastèques
- Système/Réseau
  - Maitres des ressources et des permissions (VM, BDD)
  - Maitres de l'architecture réseau
- Sécurité
  - Parent pauvre, mal vécue
  - Haute dispo compliquée voir impossible
- Dernier maillon de la chaine
  - Découvertes des besoin au fur & à mesure du staging

La gouvernance

- Parfois, non
- La coordination des équipes
- La gestion des ressources
- La gestion du versioning
- Le complexe du monolithe
- La dette technique

Constat

- Time to market de 6 mois (minimum)
- Complexité croissante, intégration constipée
- Perte de maîtrise du véhicule
- Mise à disposition de l’outillage et des ressources insuffisantes pour le continuous delivery

La cible - ZE Continuous-delivery

- Architecture microservice

- Infrastructure extensible (virtualisation containerisation)
- Digérer du pétaoctet de données (big data, machine learning, AI)

➔ Voilà le projet « transformation digitale »

Phase 2

La reprise en main du véhicule

Réappropriation du SI

- Value Stream Mapping
  - La prise de conscience du serpent de mer des process de la dev à la prod
  - L'occasion de partager une vision du SI avec l'ensemble des parties prenantes

- Le bundle
  - Déployer automatiquement le SI sur un environnement, en mode stop/start/stop
  - Oriente vers la normalisation des configurations et du versioning
  - Approche verticale et monolithique: Complètement antinomique au pipeline et au CD

Gouvernance

- Gestion de la compatibilité des APIs/BDDs
  - Prévisibilité du mode de déploiement
- Centralisée pour initialiser
  - Organisation en communauté de pratique possible
- Communication
  - Propager les usages = faire des ateliers/formation/support
  - Gouvernance documentée !

Constat

- Approche simple et pragmatique
- Mise en place de manière itérative
- On prend conscience du vide
- On amorce l’automatisation de l’infra
- On ne fait pas du tout de Continuous Delivery

Phase 3

Microservices et VMs

Le grand découpage

Architecture Microservice

- Détricotage / Dénormalisation de la grosse BDD
  - Refonte (ou exposition) des procédures stockées/triggers
  - Rupture des intégrités référentiels
- Des services réduits et autonomes
  - Moins de dépendance aux ressources connectées
- Robustesse - designs orientés asynchrone et stateless
  - Utilisation de brokers
  - Orientation vers la HA

Les Machine Virtuelles

- Découpage « physique » des ressources de base CPU/mémoire/HDD
- Reproduction des environnements
- Souplesse dans la mise à disposition
- Gros overhead sur la machine physique (Kernel)
- Granularité trop grosse pour les microservices, pas optimal

Infrastructure: l’Hyperviseur

- Bare metal ou IAAS
- Fournit les VMs avec un OS « enterprise »
- Base de la IAC
- Souplesse dans la mise à disposition des ressources

Infrastructure: Le deployment manager

- Le IAC – la magie de l'idempotence
- Provisioning des VMs
- Configuration centralisée et versionnée
- Maintient en condition opérationnelle
- Monitoring / Alerte

Constat (1/2)

- Chantiers parallélisables
- Pas d’intrusion forte dans le process de livraison existant
- Difficulté de trouver les ressources Ops disponibles à cause du run

Constat (2/2)

- Infra maîtrisée, possibilité dans le cloud voir hybride
- Séparation encore forte Dev/Ops
- Pilotage « semi-automatique » de l'infra (ex: répartition des applis dans les nœuds manuelle)
- On livre encore des artefacts custom et hétérogène

Phase 5

La transformation DevOps

Feature team – Tuer le bundle

- Microservice
  - Découpage du bundle
  - Répartition par feature team

- Intégration d’Ops dans les feature team
  - Découpage des ressources de dev de l’infra par feature team
  - Montée en compétence des autres membres (approche maker checker)

Container

- Basé sur les fonctionnalités d’isolation des process du kernel
- Autonome et isolé comme une VM, utilise le kernel du host
- Dispose de son propre filesystem
- Basé sur des « images »

Image de container

- Filesystem, découpé en layers
  - Stockage naturellement simple à optimiser
- « Légères » (10x moins que les VMs)
- Format pivot de livraison intuitif
  - Pratique à la fois pour les devs et les ops

Orchestrateur

- Aggrège les ressources de plusieurs nœuds
- Gère le cycle de vie des containers et des volumes de données persistants
- Gère automatiquement le load balancing / Réseau / DNS
- Répartition des containers sur les nœuds en fonction des ressources
- Haute Dispo (HA)

Platform As A Service

- S’appuie sur l’orchestrateur de container
  - Tout est container
- Intègre l’outillage pour la stack de monitoring
  - Logs/métrique
- Peut accueillir la stack de dev voir l’encadrer
  - Jenkins
  - SCM
- Gère les permissions au niveau entreprise (ex: role base access control)
  - Sur l’ensemble des environnements

Constat

- Infra virtualisée, souple et robuste
- Feature team autonomes
- Visibilité globale du SI pour tout le monde
- La partie infra VM reste à la marge parfois des équipes
- Continuous Delivery !!!

Démo