Au secours, mon projet BigData est en production!

Speakers : Vincent Devillers (LAYER4)
Format : Conférence
Date : 19 avril 2019

Vincent travaille sur les solutions Hadoop depuis 8 ans.
L'écosystème Data est très riche, trop. Il est difficile de s'y retrouver.

Besoin métier
Prise comme exemple, la demande client « avoir les données de la table d'opérations dans le datalake de l'entreprise » n'explique pas la finalité, le besoin métier.
On va devoir prendre toutes les données (avec un outil comme Drill). On ne pourra pas préparer les données en amont.
Une préconisation de Vincent consiste à toujours **demander au métier la finalité de la data.** Permet de choisir la techno, l'infra et les outils. Cela permet de diminuer les ressources nécessaires (et donc le cout).

Ingestion
Ingérer des données est relativement simple en 2019 : il existe de nombreux outils, payants comme Open Source. Mais attention : les ratés sur l'intégration (ex : données en doublon) impactent les résultats.
Spark n'est pas utilisé tout seul dans un pipeline d'ingestion des données.
Le data engineering doit garantir la qualité des données ingérées.
Bien trop souvent, Vincent constate que la solution est pensée de manière trop spécifique, et il n'est pas rare de refaire le pipeline d'ingestion tous les 2 ans.

Les tests
Dans le cas de projets Hadoop, on ne cherche pas à vérifier que la requête Hive respecte la syntaxe SQL 2012. On cherche à s'assurer que la requête s'exécute bien sur telles versions de Hive et d'Hadoop.
Créer des tests d'intégration dans Hadoop est un cauchemar car il y'a beaucoup de services à démarrer.
Plusieurs solutions s'offrent au Data Engineer :

1. **Grosse VM** (Cloudera ou HortonWorks) dans laquelle on retrouve tous les services. Cas le plus complet, proche de l'expérience sur un vrai cluster. Par contre, c'est très lent.
2. **Docker/TestContainers** : création d'une stack complète non cantonnée à Hadoop (on peut par exemple y ajouter un Elasticsearch). Difficulté de trouver la bonne image Docker de tel ou tel service avec la version iso-prod.
3. **MiniCluster** : projet conçu par un ex-Horton. Équivalent de HSQLDB et H2 dans le monde Java. On démarre en mémoire sa plateforme Hadoop. Revers de la médaille : le test met du temps à se lancer. On perd également l'accès au shell, bien pratique pour du debug.
4. **Hadoop-Unit** : code repris MiniCluster mais qui démarre dans sa propre JVM. Plus de problème de classpath. Tous les services sont démarrés. On retrouve tous les shells. Un plugin Maven existe. Mais il manque encore quelques services (ex : Roger).

Vincent recommande l'usage d'Hadoop-Unit.

Les technologies Hadoop évoluent rapidement. Il y'a rapidement des morceaux de code obsolètes (API dépréciée puis retirée). Nécessité de régulièrement tester son code sur les nouvelles versions. Plusieurs solutions permettent d'y arriver, par exemple Concourse.

Comment déployer son projet Data ?
Revient souvent à déployer quelques binaires et quelques fichiers de conf, livrer quelques tables …
La Production a besoin d'un pipeline simple et éprouvé. Eviter les gros scripts shells qui deviennent in-maintenable. Privilégier Ansible ou Puppet.
Cas d'un script shell pouvant être remplacé par une classe Java.
Beaucoup de shells : fish, bash, ksh, csh, tcsh, dash, zsh

Scheduling
Pour planifier des traitements, des équipes utilisent trop régulièrement crontab.
Crontab est un SPOF, pas de supervision, ni de stats, de SLA … Il faut le bannir et utiliser un outil adapté : $U, BMC, Airflow, Oozie, CA Technologies …
Bonne pratique : dans le nom du job, ajouter le contexte : job group, date d'exécution, workflow identifier, paramètres d'appel ….

Capacity planning
Bien souvent, tous les jobs sont lancés à minuit. Réflexe de lancer toutes les tâches sur des heures non ouvrées. Ce réflexe vient du mainframe fermé la nuit.
Avec Yarn, on n'a pas ce souci et on peut les lancer en journée. Yarn propose 2 modes : FIFO ordering et FAIR ordering. La queue FIFO trouve sa place pour les jobs de nuit prenant beaucoup de ressources.
Yarn permet de réserver des ressources. Le DSL RDL permet d'exprimer une réservation.