

DATA (/TAG/DATA/) MODERN ARCHI (/TAG/MODERN-ARCHITECTURE/)



Spark expliqué aux décideurs.



© Ippon Technologies

Spark expliqué aux décideurs

👤 Boris Perevalov (/author/boris/) 📅 16 Sep 2019

🗨️ 0 Commentaires (/2019/09/16/spark-explique-aux-decideurs/#disqus_thread)

Problématique

En intervenant sur différents projets Spark (parfois en mode “pompier”) et en recueillant les témoignages de mes collègues, j’ai constaté que les principales causes d’échec sont :

- le faible niveau de compréhension du fonctionnement de cet outil et des contraintes de traitement distribué,
- le manque d’industrialisation et de bonnes pratiques (Tests automatisés, CI/CD, Workflow Git, Code Review, etc.).

Dans certains cas, les projets sont au point mort (rien ne marche, le budget est consommé). Dans d’autres, les défauts du code sont compensés par des clusters surdimensionnés. Dans les deux cas, les pertes financières sont considérables.

La migration vers le Cloud n’arrange pas les choses. En mode *pay-as-you-go*, moins vos applications sont performantes, plus vous payez pour les faire tourner. Le facteur 10 (et même plus) sur le coût du cluster, par rapport à ce que consommerait une application bien optimisée, ne m’étonne plus du tout. Dans ce cas, il est plus pertinent de parler d’efficience que simplement des performances.

Le fait de négliger les aspects de l'industrialisation génère un coût important pour tout projet de développement informatique. Cet effet est encore plus accentué pour un projet Spark en raison de la surconsommation de ressources de calcul (coût du cluster).



Solutions

Formez les développeurs

Je vois souvent des équipes composées de développeurs venant de tous horizons et qui se sont retrouvés à faire du Spark sans y être formés. À mon avis, cela arrive parce que Spark est connu pour être simple à utiliser, ce qui est tout-à-fait vrai, surtout si on le compare à Hadoop MapReduce. Quelqu'un qui ne maîtrise que le SQL peut créer et exécuter un job Spark.

En revanche, développer une application performante et robuste est une toute autre histoire. Il ne suffit pas de savoir se servir des APIs Spark. Il est indispensable de comprendre ce qu'il se passe sous le capot et d'avoir un bon niveau de culture de l'univers du calcul distribué.

Une autre raison est probablement d'ordre financier. Les formations sont chères. On préfère ainsi laisser les développeurs monter en compétences au cours du projet en apprenant de leurs propres erreurs. Avez-vous calculé combien peut vous coûter ce mode d'apprentissage ?

Pour vous en donner une idée, voici ses possibles conséquences :

- surconsommation des ressources,
- faible productivité de l'équipe,
- retard du projet,
- démotivation de l'équipe,
- abandon du projet.

Donnez les moyens pour monter en compétence

Spark est un outil très en vogue. Il existe donc une multitude de livres, d'articles de blog et d'exemples de code sur Stack Overflow. Le problème est que toutes ces sources d'information ne se valent pas. Certaines sont dépassées, d'autres sont difficiles à lire. Il y a même des livres qui contiennent de fausses affirmations.

Ayant lu (ou entamé) plusieurs livres sur Spark, je conseille vivement de privilégier le livre "Spark: The Definitive Guide" (<https://www.oreilly.com/library/view/spark-the-definitive/9781491912201/>) de O'Reilly. À mon avis (mais pas seulement le mien), tout développeur Spark sérieux devrait le lire. Acheter un exemplaire pour chaque développeur de l'équipe sera un investissement très rentable.

Encouragez les développeurs à passer une certification Spark

Comment objectiver le niveau acquis par les développeurs de votre équipe suite à une formation (ou leur autoformation) ? Une des solutions peut être de les encourager (financière évidemment) à passer une certification Spark.



Au delà du niveau de maîtrise acquis, la volonté de passer une certification prouver et l'intérêt qu'ils portent pour cette technologie.

Les certifications (<https://academy.databricks.com/category/certifications>) proposées par Databricks semblent être les mieux réputées.

Assurez-vous que l'équipe développe avec des tests automatisés

Il m'est arrivé d'observer plusieurs fois des développeurs qui testent leurs applications sur le cluster de production tout au long de la phase de développement. Ils buildent et soumettent leurs applications au cluster de production à chaque modification du code.

Une itération dure donc au minimum plusieurs minutes comprenant :

- le temps de build,
- le temps d'attente sur un cluster *on-premise* ou le temps de démarrage d'un cluster éphémère sur le Cloud,
- le temps d'exécution avec les données de production (donc très volumineuses),
- le temps d'accéder aux données en sortie ou à des logs d'erreur.

Dans le pire des cas, ils font tourner le cluster de production pendant plusieurs dizaines de minutes pour se rendre compte d'une faute de frappe dans le nom d'une colonne d'un DataFrame. Je vous laisse prendre la calculatrice et estimer combien cela vous coûte, sachant qu'avec les tests unitaires, détecter une telle erreur ne prend que quelques secondes et ne consomme que les ressources du poste de développement.

Tester le code Spark directement sur les données et le cluster de production n'est acceptable que dans les cas suivants :

- Lorsqu'on est dans la phase d'exploration et de recherche (souvent faite par les Data Scientists en utilisant des Notebooks ou les outils en ligne de commande : spark-shell, spark-sql, pyspark ou sparkR),
- Afin de valider une application déjà testée fonctionnellement et procéder au *fine tuning* des performances.

Lorsque l'on passe à la phase d'industrialisation, il est nécessaire de :

- développer avec des tests automatisés (ou encore mieux en TDD),
- exécuter le code Spark en local avec un échantillon de données représentatif (c'est là où ça coince souvent),



- tuner (si possible) les performances de l'application sur un petit cluster et un sous-ensemble de données proportionnel à la taille du cluster,
- valider les performances de l'application sur le cluster de production (ou un cl et les données de production.



De plus, les tests automatisés rendront le code plus facilement maintenable grâce à la séparation des responsabilités et réduiront le risque de régression.

J'ai une profonde conviction que rien ne documente mieux le code que les tests automatisés.

N'ayez pas peur de Scala

J'ai déjà entendu à plusieurs reprises la phrase suivante :

"Nous n'avons pas retenu Scala parce que nous n'avons pas de compétences en interne, les compétences sont plus rares sur le marché et par conséquent plus chères."

Fausse croyance ! Ce n'est pas de compétences en Scala dont vous avez besoin mais de compétences en Spark. Pour qu'un projet Spark réussisse, il est surtout important que les développeurs aient une bonne compréhension du fonctionnement de Spark et des enjeux d'industrialisation.

Pour faire du Spark, il suffit d'avoir des connaissances rudimentaires en Scala. Il arrive même qu'un expert Scala ne veuille pas faire du Spark parce qu'il s'ennuie.

Un développeur Java suffisamment curieux et ouvert d'esprit tombera amoureux de Scala et du paradigme de la programmation fonctionnelle.

Je suis même convaincu qu'en termes de recrutement, vous allez plus facilement attirer des bons développeurs Java en leur promettant de pouvoir monter en compétence sur Scala.

Les bénéfices de Scala sont évidents :

- Les nouvelles features sont en priorité ajoutées à l'API Scala car Spark est écrit en ce langage,
- Scala est moins verbeux que Java de manière générale, en plus le DSL de Spark en Scala est plus élégant, par conséquent le code est plus facile à écrire et à maintenir,
- Dans certains cas, Spark Scala est plus performant que PySpark,
- L'API Scala est une des plus utilisées (dépassée par PySpark depuis quelque temps), par conséquent elle bénéficie du meilleur support de la communauté que les APIs Java et R,
- Tous les livres sur Spark donnent des exemples de code en Scala,
- La librairie de tests unitaires ScalaTest (<http://www.scalatest.org/>) est un bijou !

Pensez à la CI/CD

Pour la petite anecdote, voici ce que j'ai entendu dire par un décideur lorsque j'intégrais Spark dans une grosse structure, je cite :



"Nous n'avons pas besoin de l'intégration continue parce que notre priorité c'est la qualité. S'il faut passer deux semaines pour mettre en production, on passera deux semaines, mais on le fera bien."

Combien d'incohérences voyez-vous dans cette citation ?

Pensez à mettre en place les pipelines CI/CD dès le début du projet. Cela augmentera la productivité de l'équipe en automatisant les tâches répétitives et réduira le risque de régression du service en production. On ne peut pas parler de l'industrialisation sans évoquer la CI/CD.

Pour optimiser les coûts, pensez aux clusters éphémères

L'utilisation de "clusters éphémères" (également appelés "clusters transients") permet de tirer pleinement bénéfice du Cloud public par rapport au *on-premise*. Il s'agit d'un paradigme qui consiste à créer un cluster dédié à un traitement et à le détruire à la fin de ce traitement. L'article (<https://blog.ippon.fr/2019/04/16/cluster-spark-ephemere-avec-terraform-et-aws-emr/>) de Lucien FREGOSI vous expliquera en détail ce nouveau paradigme et vous donnera un exemple de configuration sur AWS.

La mise en place de ce mécanisme s'inscrit dans le cadre de l'industrialisation de votre projet, car il nécessite l'automatisation de la création de clusters en mode *infra-as-code*. Il n'est plus possible de lancer un job Spark à la main, ni de développer directement sur le cluster.

Faites accompagner votre équipe par des experts

En faisant intervenir des experts dès le début du projet, vous pouvez réaliser plusieurs recommandations de cet article à la fois :

- Former l'équipe à Spark et aux bonnes pratiques de développement,
- Mettre en place l'outillage DevOps (CI/CD, Supervision, Clusters éphémères, etc.),
- Valider les choix d'architecture et le workflow de traitement,
- Tuner les performances (format de données + code + configuration du cluster).

Le ROI positif de cet investissement est indéniable.

Qui a dit "RDD" ?

Si un jour (même après avoir suivi les recommandations de cet article), vous entendez parler de “RDD” dans votre open space, arrêtez tout et faites une réunion d’urgence !



Sans rentrer dans les détails, il s’agit d’une API bas niveau qui était la seule disponible sur les premières versions de Spark. Ensuite, de nouvelles APIs haut niveau ont été construites pour faciliter l’utilisation de RDD en apportant de très nombreuses optimisations et de nouvelles fonctionnalités. Les différences en termes de performance (et donc d’efficience) sont très conséquentes.

Utiliser l’API RDD aujourd’hui, c’est comme coder en Assembleur à la place de C++. Il est possible de faire du code performant, mais il faut avoir un sacré niveau. Descendre au niveau de RDD ne peut être justifié que dans certains cas très particuliers.

Certains projets legacy, développés en RDD sur les premières versions de Spark, ont été migrés vers les nouvelles versions sans réécriture du code. Ils n’ont donc tiré quasiment aucun bénéfice de cette montée de version.

Certains développeurs peu curieux se mettent à utiliser les RDD sans se poser de questions après être tombés sur un exemple de code sur Internet.

Compte tenu de tout cela, si votre équipe parle de RDD, elle vous doit à minima des explications.

Conclusion

Malgré le progrès des outils, le calcul distribué reste une problématique complexe. Le nombre de projets Spark en détresse en témoigne parfaitement.

L’industrialisation est cruciale pour la réussite d’un projet informatique. Les projets Spark n’échappent pas à la règle.

Spark est un bel outil dont je suis fan. Avec cet article, j’espère avoir contribué à ce qu’il soit apprécié à sa juste valeur grâce à l’accroissement du nombre de projets réussis.

You have found this publication useful? Click here

LIVRE BLANC

Explore Big Data.

Ce livre blanc prend “un peu” de hauteur sur les technologies Big Data afin d’aider les décideurs à faire les bons choix techniques et à y voir plus clair sur les opportunités offertes par les différentes technologies.



(<http://blog.ippon.fr/2020/07/31/construire-la-ci-dun-monorepo-les-parent-child-pipelines-de-gitlab-ci/>)



WE ARE IPPON



(<http://www.ippon.fr/nous-rejoindre/>)

Partagez cet article:

- [\(<https://www.facebook.com/sharer/sharer.php?u=http://blog.ippon.fr/2019/09/16/spark-explique-aux-decideurs/>\)](#)
- [\(<https://twitter.com/share?text=Spark%20expliqu%C3%A9%20aux%20d%C3%A9cideurs&url=http://blog.ippon.fr/2019/09/16/spark-explique-aux-decideurs/>\)](#)
- [\(<https://plus.google.com/share?url=http://blog.ippon.fr/2019/09/16/spark-explique-aux-decideurs/>\)](#)
- [\(<http://www.digg.com/submit?url=http://blog.ippon.fr/2019/09/16/spark-explique-aux-decideurs/>\)](#)
- [\(<http://reddit.com/submit?url=http://blog.ippon.fr/2019/09/16/spark-explique-aux-decideurs/&title=Spark%20expliqu%C3%A9%20aux%20d%C3%A9cideurs>\)](#)
- [\(<http://www.linkedin.com/shareArticle?mini=true&url=http://blog.ippon.fr/2019/09/16/spark-explique-aux-decideurs/>\)](#)

JE TÉLÉCHARGE ([HTTPS://FR.IPPON.TECH/EXPLORE-BIG-DATA/](https://fr.ippon.tech/explore-big-data/))



VIDÉO MÉTIER

Architecte Data @Ippon c'est ... (avec Arnaud Col)



REX CLIENTS

Ippon & ENGIE Digital : Accélération du produit eCare



POST RÉCENTS

Cucumber et RestTemplate

Aug 31, 2020

(<http://blog.ippon.fr/2020/08/31/cucumber-et-resttemplate/>)

REX, prise en main de Selenium WebDriver

Aug 20, 2020

(<http://blog.ippon.fr/2020/08/20/rex-sur-la-prise-en-main-de-selenium-webdriver/>)

Construire la CI d'un monorepo: les parent-child pipelines de Gitlab-ci

Jul 31, 2020



Ju ([http://www.stumbleupon.com/submit?url=http://blog.ippon.fr/2019/09/16/spark-decideurs/&title=Spark expliqué aux décideurs](http://www.stumbleupon.com/submit?url=http://blog.ippon.fr/2019/09/16/spark-decideurs/&title=Spark%20expliqu%C3%A9%20aux%20d%C3%A9cideurs))



(/author/boris/)

Boris Perevalov (/author/boris/)

📍 LYON

Ippon

Ippon est un cabinet de conseil en technologies, créé en 2002 par un sportif de Haut Niveau et un polytechnicien, avec pour ambition de devenir leader sur les solutions Digitales, Cloud et BigData.

Ippon accompagne les entreprises dans le développement et la transformation de leur système d'information avec des applications performantes et des solutions robustes.

Ippon propose une offre de services à 360° pour répondre à l'ensemble des besoins en innovation technologique : Conseil, Design, Développement, Hébergement et Formation.

Nous avons réalisé, en 2019, un chiffre d'affaires de 42 M€. Nous sommes aujourd'hui un groupe international riche de plus de 400 consultants répartis en France, aux USA, en Australie et en Russie.

📍 FRANCE 🌐 WEBSITE (<HTTP://WWW.IPPON.FR>)

in LINKEDIN (<HTTPS://WWW.LINKEDIN.COM/COMPANY/IPPON-TECHNOLOGIES>)

Post précédent

Scrum Master, challenger技iquement son équipe quand on est plus up-to-dated

(/2019/09/18/untitled-scrum-master-challenger-techniquement-son-equipe-quand-on-est-plus-up-to-dated/)

Poste suivant

Comment réaliser et tester un analyzer Elasticsearch 7.2.0 avec Testcontainers ?



[\(/2019/09/06/comment-realiser-et-tester-un-analyzer-elasticsearch-7-2-0-avec-testcase/\)](/2019/09/06/comment-realiser-et-tester-un-analyzer-elasticsearch-7-2-0-avec-testcase/)

ÉGALEMENT SUR BLOG IPPON TECHNOLOGIES

Cassandra vs ScyllaDB - Partie 1 : Les ...

il y a un an • 1 commentaire

ScyllaDB se veut un challenger de taille face à Cassandra. Cette base ...

AWS re:Invent 2018 - Jour 1

il y a 2 ans • 1 commentaire

Annonces AWS re:Invent 2018 - Jour 1

CD : Snowflake, Sqitch et Gitlab

il y a un an • 1 commentaire

La construction d'un Data Warehouse est assez similaire au ...

Comme que je fa

il y a 2 ans

Un billet a sur mon c tant que d

0 Commentaires

[Blog Ippon Technologies](#)

[Règles de confidentialité de Disqus](#)

1 [S'identifier](#) ▾

[Recommander](#) 1

[Tweet](#)

[Partager](#)

[Les meilleurs](#) ▾

IPPON

(<http://www.ippon.fr/>)

Discovery to Delivery

Ippon est un cabinet de conseil qui accélère les projets innovants des ses clients de la page blanche au Cloud.

Nos équipes dans le monde accompagnent les organisations dans la transformation d'idées innovantes en solutions logicielles de haute qualité avec un focus particulier sur le Time To Market.

© 2020 IPPON (<http://blog.ippon.fr>). Tous les droits sont réservés.