Hadoop Workflow Automation

Automic™ Livre blanc

TABLE DES MATIÈRES

Introduction

La visibilité générée par le Big Data est devenue un prérequis essentiel pour que les entreprises restent compétitives. Comme beaucoup de projets Open Source, Hadoop qui était initialement une petite distribution, est devenue partie intégrante de l'écosystème informatique de beaucoup d'entreprises. La vitesse avec laquelle il est possible générer de la visibilité métier à partir d'Hadoop est désormais vitale dans le processus de prise de décision des entreprises. Le défi réside dans la manière d'intégrer de nouvelles applications et processus de Big Data dans des processus IT existants, indispensables aux experts et aux analystes de données, sans provoquer de bouleversement majeur ni impacter le quotidien des opérations. Ce Livre Blanc analyse pourquoi les organisations ont besoin d'un moteur d'ordonnancement pour animer leur environnement Big Data. Il fait le point sur les solutions d'ordonnancement open source disponibles à l'heure actuelle, et sur les limites de ces solutions.

Le Livre Blanc démontre ensuite en quoi Automic est une solution d'ordonnancement indispensable pour les traitements Hadoop et Big Data. Automic introduit une intégration native qui simplifie et accélère la livraison des applications Hadoop et accroît la visibilité. Automic intègre rapidement Hadoop au sein de processus métier existants, mettant le Big Data à la portée des utilisateurs, plutôt que d'être réservé exclusivement aux experts en données.

Pourquoi un moteur d'ordonnancement est-il impératif pour Hadoop ?

Si vous évoluez dans un environnement Big Data, l'abondance de technologies peut faire apparaitre les approches comme particulièrement complexes. Toutefois, la réalité est différente. Lorsque vous devez créer un environnement de Big Data, les principes de base d'un environnement de données standard restent applicables. Vous devez toujours intégrer les données traditionnelles provenant de structures de données relationnelles, aux systèmes Big Data. Et inversement, vous devez faire en sorte que les données provenant de ces systèmes Big Data puissent être intégrées dans l'environnement traditionnel, afin de générer des rapports.

Le message est clair : lorsqu'il s'agit d'exécuter des traitements Big Data, ces derniers associent typiquement les technologies Big Data avec les applications existantes au sein d'un même processus métier.

Alors pourquoi un moteur d'automatisation est-il un élément impératif dans un environnement de développement Hadoop ? Lorsqu'il s'agit de traitements de données, les développeurs Hadoop ont souvent des difficultés à traiter les données dans leur format brut. De multiples opérations de prétraitement, souvent chronophages, telles que les opérations d'extraction, de transformation et de chargement (ETL) doivent être réalisées en amont du traitement physique. Pour rester agiles et répondre aux contraintes des métiers dans les délais impartis, les développeurs Hadoop doivent automatiser ce processus en organisant les différentes étapes en enchaînements réutilisables, pouvant être exécutés autant de fois que nécessaire pour industrialiser et accélérer le processus de développement. En clair, l'automatisation élimine le besoin en écriture de nouveau code.

Planificateur de tâches versus gestionnaire d'enchaînements

Il est essentiel de faire la différence entre un planificateur de ressources (souvent appelé « négociateur») et un gestionnaire d'enchaînements. Les négociateurs de ressources sont un élément important d'Hadoop. Ces derniers pilotent des processus sur différents nœuds, allouant des ressources basées sur des besoins applicatifs ou une capacité cluster. YARN (Yet-Anonther-Resource-Negotiator), qui est intégré dans l'environnement Hadoop, fonctionne de manière transparente pour l'utilisateur, et en règle générale, vous n'avez pas à vous en soucier.

Les gestionnaires d'enchaînements, en revanche, gèrent des tâches Hadoop compliquées. Par exemple, de nombreux traitements s'exécutant de manière séquentielle, en parallèle ou en réponse à des déclencheurs d'évènements. Un traitement peut réaliser de nombreuses tâches, comme exécuter des applications Java individuelles, accéder au système de fichier d'Hadoop / d'autres entrepôts de données, ou encore exécuter des applications Hadoop variées.

La différence ne s'arrête pas là. Les gestionnaires d'enchaînements Hadoop sont également différents en termes de programmation modèle/langage, complexité du code, format de description propriété/paramètre, applications supportées, élasticité, documentation et support.

Les outils d'ordonnancement Open Source du marché

Apache Oozie

Lorsqu'un groupe d'ingénieurs de Yahoo! s'est rassemblé autour d'une table à Bangalore, en Inde, afin de trouver une manière d'effectuer des traitements Hadoop plus complexes et à plusieurs étapes, le résultat a été le framework Oozie. Ce projet Open Source, basé sur la technologie Java, simplifie la création d'enchaînements et gère la coordination entre traitements.

Apache Oozie (du nom qu'il a pris par la suite) permet aux développeurs de combiner plusieurs traitements de façon séquentielle, au sein d'une seul unité logique de travail. Les avantages sont légion. Premièrement, Apache Oozie est intégré à Apache Hadoop et supporte les traitements MapReduce, Pig, Hive, et Sqoop. Deuxièmement, le framework Open Source peut être utilisé pour planifier des traitements spécifiques à un système, comme les programmes Java. Troisièmement, les administrateurs Hadoop peuvent créer des transformations complexes de données combinant le traitement de tâches individuelles différentes et même des sous-enchaînements. Le résultat ? Plus de contrôle sur les traitements complexes et une réutilisabilité accrue des traitements à volonté.

Azkaban

Azkaban est un moteur d'ordonnancement conçu pour l'écosystème Hadoop. Développé par LinkedIn et écrit en Java, Azkaban gère les dépendances entre traitements et fournit une interface utilisateur web facile à utiliser, pour gérer et contrôler les enchaînements Big Data.

Il y a un certain nombre de fonctionnalités communes entre Azkaban et Oozie. Les deux sont des moteurs d'ordonnancement Open Source pour la planification des traitements Hadoop et les deux sont écrits en

Java. Les similitudes s'arrêtent là. Azkaban est simple d'utilisation, avec des planifications d'enchaînements faciles à définir, tandis qu'il est plus complexe de définir des enchaînements avec Oozie. L'ordonnancement Azkaban ne supporte que la planification fixe, tandis qu'Oozie supporte à la fois la planification fixe et basée sur les données. Azkaban garde le statut de tous les enchaînements en cours d'exécution en mémoire, alors qu'avec Oozie, le statut de l'enchainement est gardé en mémoire uniquement le temps d'un changement de statut.

Airflow

La plateforme communautaire de réservation de logement Airbnb a récemment mis en open-source Airflow, son propre framework de gestion d'enchaînements, sous licence Apache. Airflow est utilisé en interne chez Airbnb pour concevoir, superviser, et ajuster les processus de données. La plateforme est écrite en Python, tout comme les enchaînements qui s'y exécutent.

Airflow permet aux développeurs de concevoir, de gérer et d'exécuter des enchaînements basés sur une planification périodique. La plateforme interagit avec Hive, Presto, MySQL, HDFS, Postgres et S3. Des APIs spécifiques sont également fournies, afin de rendre le système plus extensible. Airflow fournit une interface en ligne de commande, ainsi qu'une interface utilisateur orientée web, permettant aux utilisateurs de visualiser les dépendances de processus, superviser les progrès et déclencher des tâches.

En quoi Airflow diffère-t-il d'Oozie ou d' Azkaban ? Les processus Airflow sont définis en tant que code, en opposition à un langage « markup » dans Oozie ou Azbakan. De plus, les tâches sont instanciées de façon dynamique, au lieu de de dériver des classes comme dans Luigi (voir ci-dessous). En conséquence, Airflow est parfaitement adapté aux situations où les processus sont générés dynamiquement à partir de fichiers de configuration ou de métadonnées de toute forme.

Luigi

Il s'agit d'un framework de données open source, basé sur Python, visant à concevoir des processus de données complexes et fournissant un support natif à Hadoop. Créé par Spotify en 2009, Luigi est à présent également utilisé par des organisations telles que Foursquare, Stripe, et Asana dans leurs environnements de production. Au lieu d'utiliser une configuration XML/YAML, tous les traitements et leurs dépendances sont écrites comme des programmes Python. L'outil gère de nombreuse tâches, y compris la résolution de dépendances, l'ordonnancement et la visualisation.

Le framework Luigi est relativement petit (quelques milliers de lignes de code) et par conséquent facile à appréhender. En outre, le gestionnaire d'enchaînement fournit un moyen abstrait mais simple de définir un processus de données en termes de tâches et de cibles, s'occupant également de gérer les dépendances. Luigi a de nombreux avantages : il gère les dépendances, réduit le volume de code standard requis pour la vérification des erreurs et gère également la reprise en cas de problème. La solution permet également aux développeurs de suivre une structure claire et précise durant le développement de processus de données.

Pinball

Pinterest a récemment proposé Pinball, en tant que gestionnaire d'enchaînements élastique et open source. La solution supporte une grande variété de processus de données, de traitements composés de simples scripts Shell à des enchaînements Hadoop élaborés. Pinball est largement utilisé par des équipes d'ingénieurs et gère des centaines d'enchaînements traitant les données quotidiennement dans les clusters Hadoop. Les plus gros enchaînements comptent jusqu'à 500 traitements.

Nativement, Pinball supporte une syntaxe de configuration d'enchaînements basée sur Python. Il fournit également un certain nombre de modèles de traitements pour configurer de simples scripts Shell et d'autres traitements plus élaborés sur la plateforme Hadoop. Avec Pinball, les utilisateurs ont toute latitude nécessaire pour modéliser les dépendances de données entre traitements – pensez par exemple à un traitement retardé jusqu'à que les données dont il a besoin soient disponibles.

Limites de ces solutions open source

Lorsqu'il s'agit de choisir une solution d'ordonnancement open source comme celles abordées ci-dessus, quelques précautions sont à prendre :

- Chacune de ces solutions open source ont été conçues par des entreprises jeunes et en croissance rapide - Airbnb, Spotify and Pinterest, par exemple. Ce type d'organisation a rarement besoin d'intégrer le Big Data dans des environnements de production existants. Pourtant, la plupart de entreprises plus classiques ont besoin d'intégrer le Big Data à des applications traditionnelles de type ERP ou entrepôt de données.

- Chacune de ces solutions d'ordonnancement open source identifiées ci-dessus sont conçues pour les ingénieurs de données ou les développeurs, pas pour les analystes de données ou les utilisateurs informatiques lambda. Or de nos jours, la plupart des cas d'utilisation courantes du Big Data concernent les utilisateurs non-techniques comme les analystes de données ayant besoin de capacités de libre-service pour réaliser des analyses de données dans un cadre plus agile.

- Une autre limitation de ces solutions open source concerne la conformité et la gouvernance de données, fonctionnalités qui manquent ou requièrent un développement complémentaire. En s'inscrivant dans une stratégie de GRC (Gouvernance, Risque, Conformité), les organisations informatiques doivent s'assurer que le métier dispose de données fiables, conformes et indiscutables, protégées de toute perte causée par inadvertance ou malveillance.

Trop d'outils = trop de gestion

Si vous adoptez trop d'outils au sein d'un processus, la charge en termes de gestion peut s'avérer écrasante. Pour chaque outil ou application que vous ajoutez dans vos traitments Big Data, de multiples tâches doivent être intégrées, synchronisées et coordonnées entre ces nouveaux outils et une multitude d'autres déjà utilisés dans le système.

Ce processus est compliqué à concevoir et à gérer. Visibilité et intelligence nécessaires à la compréhension de ce qui se passe à d'autres endroits du système font défaut à la plupart des outils. Le choix de l'outillage,

l'intégration et la gestion sont des problématiques récurrentes dans l'industrie de l'informatique. De nombreuses organisations perdent beaucoup de temps, d'argent et de ressources à analyser les outils qu'ils souhaitent utiliser, pour ne découvrir que plus tard qu'il leur en faudra bien davantage pour comprendre comment les utiliser, les intégrer, et les maintenir.

Et ce n'est pas fini. Une fois qu'une organisation a résolu ces problématiques, il faudra encore beaucoup de temps et d'efforts pour concevoir les éléments nécessaires à l'intégration de l'outillage. L'explosion de nouveaux outils et applications Big Data n'a fait qu'accélérer et accroître le degré de ces problématiques.

Pourquoi Automic est-il un moteur d'ordonnancement indispensable pour Hadoop ?

Si vous recherchez une solution d'automatisation de traitements Hadoop, conçue pour répondre aux exigences métier d'une entreprise moderne, il est capital d'examiner les fonctionnalités clé suivantes :

- Une capacité de planification Hadoop native
- Agents pour les systèmes d'exploitation et les Webservice (SOAP & REST) pour sécuriser tous les futurs développements Hadoop
- Transferts de fichiers intégrés
- Architecture Haute Disponibilité et élastique
- Mises à jour transparentes, sans aucune interruption de service
- Support de l'orientation objet afin de réaliser des enchaînements rapides, souples, et réutilisables
- Support d'applications tierces natif, y compris les ERP et le BI
- Automatisation dynamique et évènementielle
- Equilibrage de charge intelligent
- Capacités SLM intégrées, pouvant donner une perspective à la fois au métier et à l'IT
- Outils d'analytique et de reporting intégrés, fournissant de la visibilité sur les activités d'enchaînements
- Capacité de libre-service, pour rendre les utilisateurs métier plus autonomes
- Sécurité et de conformité des processus au travers de l'entreprise
- Expérience de l'automatisation dans tout type de secteur

Bien qu'il existe d'autres facteurs à prendre en compte, une solution qui englobe déjà les fonctionnalités clé citées ci-dessus vous garantira de répondre aux exigences sans cesse croissantes de l'entreprise.

Automic fournit toutes ces fonctionnalités. Notre solution d'automatisation Hadoop introduit une intégration native qui simplifie et accélère livraison d'applications Hadoop.

Elle intègre rapidement Hadoop dans des processus métier existants, mettant le Big Data à la portée des utilisateurs, plutôt que d'être réservé exclusivement aux experts en donnée

Comparaison de fonctionnalités Automic

Fonctionnalité	Automic	Oozie	Azkaban	Airflow	Luigi	Pinball
Mission de l'outil	Enchaînements métier	Enchaînements Hadoop	Enchaînements Hadoop	Enchaînements Python	Enchaînements Python	Enchaînements Python
Déclencheurs d'enchaînements	Temps, Prédécesseurs, Fichier, Application, Evènement	Temps, Prédécesseurs	Temps, Prédécesseurs	Temps, Prédécesseurs	Temps, Prédécesseurs	Temps, Prédécesseurs
Dépendances cross-cluster Hadoop	Oui	Non	Non	Oui	Oui	Oui
Map Reduce / Yarn	Oui	Oui	Oui	Oui	Oui	Oui
Pig & Hive	Oui	Oui	Oui	Oui	Oui	Oui
Sqoop	Oui	Oui	Oui	Oui	Oui	Oui
Haute Disponibilité	Native, OOTB	Equilibrage de charge	Non	Non	Non	Non
Configuration interface	GUI & Scriptage	XML	Fichiers propriété Java	Code Python	Code Python	Code Python
Utilisateur cible	Nonn Technical to Technical	technique	technique	Développeur	Développeur	Développeur
Interface graphique	Dev, Monitoring, Admin, Ops, Catalogue de service	Supervision	Supervision, Ops, Admin	Supervision, Admin	Supervision	Supervision
Transferts de Fichiers	Natif/Intégré	Non	Non	Code nécessaire	Code nécessaire	Code nécessaire
Support OS	Etendu (15+)	Limité (Linux)	Limité (Linux)	Python	Python	Python
Intégration Application	Etendu (SAP, Oracle, Informatica ...)	Non	Non	Non	Non	Non
Intégration ITSM	Oui	Non	Non	Non	Non	Non
Gestion de version des enchaînements	Oui	Non	Non	Non	Non	Non
Libre-service GUI avec prompts utilisateur	Oui	Non	Non	Non	Non	Non
Pré et Post conditions pour chaque traitement	Oui	Non	Non	Code nécessaire	Code nécessaire	Code nécessaire
Fonctionnalité de reprise d'enchainement	Oui	Non	Oui	Code nécessaire	Code nécessaire	Code nécessaire
Fonctionnalité de retour arrière sur enchaînement	Oui	Non	Non	Non	Non	Non
Mise à jour sans interruption de service	Oui	Non	Non	Non	Non	Non
Gouvernance & Conformité de données	Native	Non	Non	Code nécessaire	Code nécessaire	Code nécessaire
SLM intégré	Oui	Non	Non	Non	Non	Non