DATA MANAGEMENT

DATA MANAGEMENT AT SCALE

Note de lecture amendée d'une revue d'état de l'art du Data Management.

Livre écrit par Piethein Strengholt, responsable de la conception de la base No SQL de LinkedIn, large expérience dans le conseil sur des projets Data.

Il aborde le Data Management, un des métiers de la Data peut-être un peu moins connu, qui se trouve être le mien au sein d'Orano.

Pourquoi le Data Management ? Que faire de sa/ses Data = une question qui a toujours existe. La Data ou les données font partie du socle d'une Société Digitale. Les départements Data sont d'ailleurs très souvent rattachés à la Direction de la Transformation Digitale (c'est notamment le cas du mien)

Data is the New Oil, les entreprises cherchent à être/se disent toutes Data Driven.

D'où la nécessité pour les sociétés d'avoir une bonne gestion de ses données, un bon Data Management.

Il présente une approche assez pionnière, basée sur la flexibilité et qui a donc l'avantage de S'adapter au degré de maturité de chaque société.

Toute stratégie Data s'appuie sur une architecture informatique, IS adaptée.

Scaled Architecture ou la scalabilité/l'extensibilité d'une architecture

La capacité finalement d'une architecture à s'adapter à une forte demande, ici sous-entendu à un gros volume de données. Architecture qui diffère des autres par le fait que les parties qui la composent sont independantes et qui prone finalement une approche très incrémentale : "start small, see how things are progressing, and continue " On fera ici le lien avec les méthodes Agile. https://agiliste.fr/introduction-methodes-agiles/

Ce concept de scalabilité se rapproche beaucoup du concept de Data Mesh instaure par Zhamak Dehghani

Pour en savoir plus, un article intéressant sur le Data Mesh https://martinfowler.com/articles/data-monolith-to-mesh.html

Idée derrière = création d'une architecture data en self-service, en traitant finalement la data comme un produit, Assez similaire a une data Fabric = aujourd'hui on parle de data Factory lors qu'on pense industrialisation des cas d'usage, voir si c'est la même chose?

Ci-dessous une illustration de ce qu'est un Data Mesh

Le Data Mesh un bloc beaucoup moins monolithique Source https://martinfowler.com/articles/data-monolith-to-mesh.html

Quelques concepts clés à retenir : Servir plutôt que d'ingérer de la Data Découvrir et utiliser la Data plutôt que d'extraire et charger Méga Diffusion de la Data via des pipelines centralisés Un écosystème de produits autour de la Data basée sur la collaboration et la distribution plutôt qu'une plateforme data centralisée monolithique.

Chapitre 1. Le Data Management et son aspect disruptif

Qu'est-ce que le DATA MANAGEMENT ?

Le Data Management répond à trois challenges majeurs Impact du volume croissant de data, appelé Datafication par l'auteur La démultiplication des technologies Les questions de sécurité

(Dans ce contexte croissant du volume de données, un concept à noter, peut-être moins mis en lumière par l'auteur est l'importance du Change).

Un article intéressant sur le Change et la transformation digitale au sein large, en open source sur Cairn.info La conduite du changement pour et avec les technologies digitales, David Autissier, Kevin J. Johnson, Jean-Michel Moutot)

Pour l'auteur les Datawarehouses et Datalake actuellement en place dans les entreprises ne savent pas répondre à cette Datafication.

Une définition courte du Data Management : Ensemble des politiques et procédures requises pour gérer la Data

Selon la définition du DAMA DBOK "Data management is the development, execution, and supervision of plans, policies, programs, and practices that deliver, control, protect, and enhance the value of data and information assets throughout their life cycles." On gardera en tête la notion clé de valeur/création de valeur autour de la donnée pour y revenir un peu plus tard.

Le Data management répond à un besoin croissance d'analyse de données dit analytics

Le développement de l'Open source a conduit à la création de base de données/Data Bases spécialisées qui permettent maintenant d'analyser des volumes massifs de données aussi appelé Big Data, quelques exemples : Cassandra, HBase, MongoDB, Hive, and Redis.

"The growth of analytical techniques means an accelerated growth of data-intensiveness: the read-versus-write ratio is changing significantly"

Plus d'informations sur les bases de données : https://www.lebigdata.fr/base-de-donnees

Une vitesse de livraison de logiciels accélérée

Notamment grâce aux mode de production DevOps. Le DevOps est une approche basée sur l'unification du Dev = développement de logiciels et Ops (toutes les opérations liées à l'information et la technologie (l'administration des infrastructures informatiques), ceci dans le but d'assurer une livraison continue avec des logiciels de qualité.

Vu par certains comme le nouvel Agile, le concept DevOps permet d'accélérer la livraison d'outils et donc de gagner en efficacité.

https://www.silicon.fr/avis-expert/devops-le-nouveau-standard-agile-et-fiable

Une accélération de la vitesse des réseaux

"Gartner Data and Analytics Summit in 2018 where Google demonstrated that it's possible to move hundreds of terabytes of data in their cloud in less than a minute."

La protection de l'information et la sécurité associées aux données font parties des top priorités

Celle si se gère à travers deux volets : Un volet technique autour des architectures = cybersécurité On notera l'explosion des métiers et formations autour de la cyber sécurité. La protection de l'information autour des personnes = e données personnelles Le GDPR en Europe et le CCCA sont aujourd'hui au cœur du débat. Ces règlementations évoluent constamment et demandes aux entreprises une veille et une adaptabilité constante. Par ailleurs ces deux volets sont cruciaux car ils déterminent quels type d'architecture peut être envisagé lors de la mise en place d'une solution.

Chaque société a sa propre politique de classification de l'information. On compte généralement un minimum de 2 niveaux Un niveau non critique qui permet notamment d'envisager des solutions de type Saas sans problème Un niveau plus critique qui pousse généralement vers du On Prem / ou des solutions de type Saas avec des clés de cryptage ou double/triple authentification

Un besoin d'intégration des systèmes opérationnels et transactionnels

Aujourd'hui l'architecture d'intégration doit connecter le système opérationnel et les applications dites analytiques

Une monétisation de la donnée qui fait que les entreprises doivent changer la façon dont elles intègrent la donnée

Utilisation de l'open Data, et les entreprises rendent leurs APIs disponibles au grand public

De nombreuses entreprises disposent d'une architecture obsolète

- Data Warehouse et BI Long à développer, demande certaines connaissances pour le maintenir, un remplacement peut être risqué même si ce système pose des problèmes de qualité de la donnée
- Data Lake Similaire au Data Warehouse mais stocke la donnée avant même que celle-ci soit transformée, nettoyée, structurée

Chapitre 2 ; La scalabilité des architectures : comment la mettre en place

Aujourd'hui les sociétés ont essentiellement besoin d'interfaces faciles à utiliser.

Rappels sur le fonctionnement/la création d'applications

Un bref rappel sur le fonctionnement et la création des applications : Chaque à sa propre base de données et sert un usage propre Etapes de création d'une application Conceptualiser et design thinking Création d'un data modèle dit logical Et enfin la création de l'application

Il est essentiel de savoir à tout moment où est la Data et comment elle est managée. Golden source = base de la Data Gouvernance

Insérer schéma page 84

Afin de comprendre comment transite la donnée, revenons aux applications. La programmation orientée objet est au cœur de la création d'une application On compte plusieurs approches design d'une appli ou d'un logiciel dont une particulièrement intéressante.

La conception pilotée par le domaine dite DDD Domain Driven Design (voir le livre blanc écrit par Eric Evans en 2003) Approche qui s'inscrit dans la continuité de l'approche Agile et cherche à casser les silos. https://blog.myagilepartner.fr/index.php/2020/05/27/ddd-domain-driven-design/

Quelques concept clés à retenir concernant le DDD et qui permettent d'éviter les silos Bounded context = découpage du contexte global en petits contextes logiques Ubiquitous Language = utilisation d'un langage commun compris de tous via les user stories Une modélisation des domaines

Importance de faire le lien entre des business capabilités (cartographies) et mappes Les applications doivent être liées à une business capability instance they should not span across or be linked to multiple capability instances at the same time. "

Une claire délimitation entre les business capabilities et l'architecture des applications est bénéfique et un pré requis à une bonne gouvernance.

Comment les applications communiquent entre elles ? Via des liaisons dites point à point/point to point Comportent l'inconvénient de limiter le contrôle possible, en résulte une architecture spaghetti Une alternative au point à point serait le silo Un silo comporte l'avantage de permettre de trouver la donnée facilement mais il ne permet pas l'agilité. Il rend l'effort de coordination énorme et quasiment impossible et demande un effort d'intégration important. Par ailleurs dans le cadre d'une bonne politique de la donnée nous cherchons à éviter les silos https://www.lebigdata.fr/silos-de-donnees-tout-savoir Une troisième alternative data hub spoke Similaire a un silo à l'exception du fait que les applications ne dialoguent pas directement et passent par un hub central, ce qui réduit la complexité des interfaces https://www.lebigdata.fr/data-hub-definition

"The first principle of the Scaled Architecture is: only allow golden datasets to be distributed from the golden source systems. A golden source is the application where all authentic data is created" "This unique data will be linked to golden datasets, which are technology-agnostic representations used to classify data and ensure it can be linked to ownership, classifications, context, and so on."

Chapitre 3. Comment gérer un volume de données important : le RDS

Plusieurs types d'architecture

Le RDS ou Remote Desktop Service, trouve ses origines avec la période de création du Datawarehouse. C'est la première architecture d'intégration de distribution de la donnée et la pierre d'angle d'une architecture scalable

Le RDS ou Services Bureau à distance est une architecture centralisée qui permet à un utilisateur de se connecter sur un ordinateur distant utilisant Microsoft Terminal Services.

Il permet une lecture intensive des données et fournit un accès sécurisé à de la Data consolidée, permettant ainsi une large variété de cas d'usage.

Différence principale avec un Data Warehouse est que le RDS préserve la Data dans son contexte original et ne demande pas de transformation du modèle de données de l'entreprise = pas de perte de valeur

Le RDS lui est plus flexible car il permet la duplication des données du coup différents volumes et formats en même temps. Il est important de rendre son RDS performant, notamment au niveau sécurité en ajoutant une couche a son architecture.

Le RDS répond à un besoin vital pour l'entreprise : stocker et historiser la donnée.

Il repose sur la mise à disposition de Meta donnée, un concept clé lorsqu'on aborde la Data Gouvernance et le Data Management. Qu'est-ce qu'une Meta Donnée ? Une donnée qui décrit la donnée.

Copier schema page 165

CQRS ou Command Query Responsability Segregation. Il s'agit d'un pattern d'architecture qui vise à séparer les couches de lecture et d'écriture. Le besoin est né du constat que la plupart des architectures proposent une unique couche d'accès aux données permettant de faire la lecture et l'écriture. Or, dans la plupart des applications, les besoins en écriture ne coïncident pas avec ceux en lecture. https://blog.webnet.fr/cqrs-petites-recettes-pour-vos-architectures/

Il se base sur le fait de copier la donnée pour en permettre une lecture intensive.

La logique de transformation de l'intégration et des étapes de livraison entre la source et le RDS doit être collectée et contrôlée afin de comprendre comment se déplace la donnée. Ce mouvement s'appelle le Data Lineage. En réalité on collecte plus souvent des informations sur le mouvement des Meta données qui nous indique ensuite le mouvement des données.

Intégrer un exemple/visuel de Data Lineage

NB : Bon à savoir, il existe plusieurs outils de Data Lineage sur le marché / Ces outils sont souvent également appelés Data Catalog : Collibra, Meta Analysis

Impossible de parler de données sans aborder la qualité de la donnée dite aussi Data Quality. La qualité de la donnée peut être adressée au niveau de la source ou du RDS. Peut être gérée au niveau de la source ou du RDS. Dans le cadre du data management celle-ci doit être adressée à la source, pour permettre aux consommateurs de data de ne pas avoir à la retraiter à chaque fois.

Afin de pouvoir gérer la donnée au mieux l'auteur commande notamment De considérer d'intégrer des solutions dites off the self D'intégrer des solutions qui ne sont pas sur étagères (Les grosses sociétés marchent généralement avec un système de solutions sur étagères, ce qui n'implique pas toujours d'avoir un SI parfaitement rationnalisé) Une vigilance lors de l'extraction de Data via des API externes et Saas. Parfois l'API ne permet pas de charger toute le volume de données souhaite Certaines API s'avèrent onéreuses, pour chaque appel de données = 1 plan couteux

Pour en savoir plus sur les API https://www.ibm.com/cloud-computing/API_pour_les_nuls_WSM14025FRFR_3_of_5.pdf/

NB : Il est parfois utile d'encapsuler une API quand on requête une solution Saas

Chapitre 4. L'Architecture dite API

API Architecture Basée sur les API, une communication synchrone et un modèle requête-réponse Permet l'échange d'un petit volume de données

L'architecture orientée service SOA = Architecture orientée service (SOAP, JSON ou encore REST) = Communication entre applications à travers des services web Permet aux applications d'être modifiées de façon indépendante.

Microservices sont en fait une extension du SOA Il s'agit d'un modèle d'architecture qui décompose les applications en petites parties plus simples, plus faciles à gérer. Il fait écho au concept d'agilité mentionne plus haut = chacun peut travailler sur un petit bout d'application. Chaque partie peut ainsi être testée, développée et déployée individuellement et comporte donc un plus grand degré de scalabilité.

https://nexworld.fr/soa-versus-microservices-quelles-differences/

NB : Ce concept de microservices va de pair avec une autre tendance : les plateformes sans serveur, dites scalable et containerized Kubernetes en est un exemple https://kubernetes.io/fr/docs/concepts/overview/what-is-kubernetes/

Plusieurs concepts clés dans la mise en place de micro-services : Function as a service (FaaS) La fonction en tant que service (FaaS) est un moyen sans serveur d'exécuter des morceaux de code modulaires en périphérie. C'est une forme nouvelle de cloud computing FaaS permet aux développeurs d'écrire et de mettre à jour un morceau de code à la volée, qui peut ensuite être exécuté en réponse à un événement, tel qu'un utilisateur qui clique sur un élément d'une application web. Il est ainsi facile d'adapter le code et constitue un moyen économique de mettre en œuvre des micro-services sans contraintes de maintien d'infastructure.

Quelle différence avec Saas = l'absence de serveur. "functions are spun up on-demand in a pay-per-usage license."

Une façon de contrôler les micro services = service mesh "A service mesh is a layer (proxy) dedicated to handling service-to-service communication"

Différence principale avec un API est que qu'avec le service mesh la communication n'est pas re-reroutée à l'extérieur mais reste dans le périmètre service interne dans lequel opèrent les micro services.

Chapitre 5. Architecture orientée événements et Streaming

Introduction à la Streaming Architecture L'auteur aborde ici la notion d'EDA, architecture orientée événements Une architecture orientée événements est un modèle d'architecture logicielle utilisé pour la conception d'applications. Dans un système orienté événements, la capture, la communication, le traitement et la persistance des événements constituent la structure centrale de la solution. C'est ce qui différencie ce type de système du modèle traditionnel basé sur les requêtes Real time streaming data (en opposition au queuing system) ? La donnée y est livrée sous forme d'un fil d'évènements (streaming data) et peut provenir d'une grande variété de sources : IOT, smartphones, social networks

Quelle différence majeure avec un modèle base sur les API ? On écoute au lieu de parler Communication asynchrone : penser à l'exemple de l'email qu'on envoie sans forcément attendre que l'autre réponde Les applications font la même chose avec des messages ou des événements La communication asynchrone rend une architecture dite scaled plus solide

Quelle différence avec le SOA : On en revient à la qualité de la donnée : "The big difference is that EDA allows a level of data inconsistency by allowing data replication, while within SOA, inconsistency is avoided through service isolation"

Un aparté technologique : Apache Kafka c'est quoi ? Un outil majeur quand on parle Big Data, basé sur la communication asynchrone Solution développée à la base par LinkedIn pour capturer des événements et des données à une échelle incroyablement grande (suite au constat du manque d'outils adaptés) Une plateforme Open source de diffusion de données en continu (qui assure à la fois le stockage et l'échange de données en temps réel de toutes les données utilisées par les entreprises qui l'utilisent ; C'est un outil écrit en Scala "Kafka stocke les messages dans des fichier flats (append-only) immutable (sur un système de fichier distribués).

Kafka retient les messages en utilisant un paramètre de temps appelé Time to live TTL = généralement 7 jours Après ce laps de temps les fichiers sont tronqués Kafka est aussi distribué, ce qui signifie que la donnée est répliquée au sein d'un groupe d'instances multiples (brokers) Kafka organise ses données sous forme de sujets, similaires à fils dit message queue Quelle structure derrière Kafka ? Kafka repose sur un log, une structure de données abstraite = sorte d'array de messages au sein duquel les données sont ordonnées en fonction du temps de réception des messages Quelles conditions d'utilisation? "When data is joined in Kafka, you must ensure that your streams and tables are co-partitioned, which means that input records on both sides of the join have the same configuration settings for partitions."

Chapitre 6. Un chapitre pour refaire le point sur les concepts évoqués

L'auteur revient rapidement sur les concepts clés évoqués plus haut.

Toutes les applications qui ont besoin de communiquer et d'échanger de la donnée se retrouvent dans la couche liaison de données et sont découplées.

Qu'est-ce que la couche liaison de données ? = C'est la couche qui transfère les des données entre les nœuds adjacents d'un réseau etendu WAN ou d'un reseau local LAN. Trouver un articleet un graphique expliquant comment marchent sur les reseaux. Graphiques

Voici un retour rapide sur les différents types d'architectures et leurs usages :

Architecture RDS Les + Utile pour analyses BI et des analyses de données Read only data stores et API sont les deux briques cette architecture Basée sur une communication asynchrone, elle joue un rôle important dans la qualité de la donnée et la gestion du cycle de vie de la donnée Les – Peu adaptée quand besoin de données très précises

API Architecture Basée elle sur les échanges en temps réel et la communication synchrone Peut fournir aussi bien des fonctionnalités Data que business

Streaming architecture A travers ce type d'architecture on peut transformer de la Data en temps réel et notifier d'autres applications Envoie de la data à des gens qui « écoutent » = système de communication à sens unique contrairement aux API ou la communication se fait dans les deux sens. Ce type d'architecture permet de réagir rapidement sur les données reçues et surtout "The true potential is that all the different teams of the domains can independently evolve"

Nous avons entrevu le linéage de la Data, les contraintes associées L'auteur met maintenant un pied dans la gouvernance en abordant de fait l'importance des différents rôles que nous verrons plus loin. "The quality of data, its representation, its granularity, and its richness, in this philosophy, must become the responsibility of the respective data owners (product owners)" On les appelle en vérité Data Owners dans un contexte d'entreprise.

A travers ces différents rôles, une même vision : Les données doivent être pensées de façon à pouvoir être réutilisées "Provide data properly once and consume it many times." (Rappelons-nous la réutilisation des services est la base du SOA), et non pour un besoin spécifique Au sein des différents domaines les identifiants au sein des domaines doivent être penses de façon à ce qu'on retrouve la donnée facilement sans avoir à manipuler les clés pour joindre des jeux de données Cela nécessite un certain consensus autour du format mais aussi de la granularité de la donnée = décisions doivent être prises avec les 'sachants', et donc conjointement avec le métier

En plus de ces rôles, il est crucial de Disposer d'un référentiel de données De pouvoir mettre disposition/consultation des Métadonnées via des plateformes

Chapitre 7. Data sécurité et gouvernance pérenne

Application based ownership => a Data based ownership Il s'agit en fait de sortir de la vision outil/application pour adopter une vision Data

Data gouvernance et sécurité de la donnée vont de pair.

Comment définir la Data gouvernance :

L'ensemble des activités qui consistent à implémenter un contrôle sur la Data. Un vrai besoin pour les sociétés notamment lorsqu'elles grossissent, et avec elles le flux de données traitées. Comme vu plus haut, les règlementations/normes liées à la protection des données s'additionnent à la contrainte du volume de Data traité.

La Data gouvernance comprend 5 dimensions - Organisation (culture) via la définition de nouveaux rôles et la mise en place d'une culture change associée - Les process Doivent être audités et contrôles, ceux concernant la sécurité en priorité - Technologie Clé dans la création d'une confiance de transparence autour de la donnée (dans lesquelles les Meta données ont un rôle important à jouer) - Les personnes qui font une société Mettre en place des responsabilités claires et développer une culture d'utilisation de la donnée dans le respect d'une certaine éthique - La Data/les données = classification, lineage

Impossible de parler de Data gouvernance sans parler des rôles ;

Trouver un schéma montrant les différents rôles associés à la Data

Voici les principaux Data Owner aussi appelé Data Trustee ou Process Owner Les deux derniers termes sont en réalité peu utilisés en entreprise. = Celui qui est responsable de la donnée (qualité, définition, classification, usage = but) Data user = celui qui prévoit d'utiliser la donnée, et qui doit notamment clarifier les besoins associes Data Creator = celui qui créée la donnée, sous la supervision du Data Owner Data consumer = celui qui consommé la donnée générée par le Data Creator et/ou le Data Owner Application owner, parfois appelé Data Custodian En charge de maintenir l'information applicative et son accès En réalité un terme qu'on entend de moins en moins, il est remplacé dans les entreprises par le Data Manager ou le RFA.

Data steward S'assure que la politique de la donnée produite au niveau groupe et les standards associes soient respectés.

La data gouvernance s'appuie sur ces rôles pour fournir le cadre nécessaire à la gestion de la donnée.

Il est important au sein de ces rôles de définir clairement les taches qui y sont associées. Ces taches se traduisent au sein d'une entreprise par la mise en place d'une matrice RACI, qui peut être amené à évoluer dans le temps.

Coller un exemple de RACI

Le corps Data gouvernance s'assure qu'une gestion de la Donnée/Data management pérenne soit mise en place au sein de l'entreprise.

Cela passe par définir les rôles, les principes, les règles et les standards associes au besoin de l'entreprise et à son gout pour le risque mais aussi et l'auteur semble moins l'aborder ici, au niveau de maturité Data qui est le sien.

- `Définir les rôles/responsabilités associes au sein des processus mis en place dans le cadre du Da`
- `Définir les politiques de gestion de la donnée en accord avec les différentes législations`
- `Définir des niveaux de qualité de la donnée tant à la source (création) qu'au moment de son utili`
- `Définir la granularité du Data lineage`
- `Mettre en place une classification interne/politique de stockage`
- `Etudier l'impact des nouvelles régulations`
- `S'aligner avec l'architecture SI de l'entreprise`
- `Définir et mettre en place des politiques de gestion du cycle de la donnée`
- `Mise en place de convention de nommage pour les Métadonnées via un lexique`
- `Travailler sur la mise en place d'une culture Data`
- `Définir des best pratiques et le maintien d'un glossaire business`
- `S'assurer que les modèles de données sont mis à jour`

- `Alimenter un Data Catalogue et s'assurer que les informations qu'il contient sont correctes et mi`
- `Définir des principes de chargement/consommation de la donnée`

Une gouvernance réussie s'appuie sur : - La possibilité de tracer toutes les applications et leurs propriétaires IT (possibilité d'utiliser Service Now) - Liste des Logs et des jeux de Data - Data sharing agreement administration (DSAA) via une application dite standalone qui enregistre tous les accords de partage de la donnée entre Data Owners et consommateurs de données - Un référentiel de Meta Données Alation ou lumeda, deux Data Catolog, solutions additionnelles vs Meta Analysis et Collibra Les comparer Outils qui permettent de scanner la Data, outils de profiling On trouve ces fonctions dans les data Catalog les plus évolués Autrement Tamr = solution de Data Management la plus évoluée à ce jour

COUCHE DE DATA GOUVERNANCE

Golden Data set, représentations dites technology agnostic, utilisées pour classifier la donnée = chaque jeu de Golden Data sous le contrôle d'un domaine différent

Schéma page 466

ETIQUETTES ET CLASSIFICATION DATA

La classification type de données est un aspect important du Data Management notamment parce qu'elle conditionne ensuite l'accès aux données. Ce qu'on appelle la liste classification sécurité doit être maintenue en interne et ne doit pas être trop longue Cette classification a lieu à la fois sur des données structurées et non structurées.

CLASSIFICATION PAR USAGE

Utile pour les cas d'usage = permet un accès sur mesure à la donnée Ici aussi une liste courte suffit, en voici un exemple : DATA_SCIENCE AD_HOC_USAGE HR_ONLY FINANCIAL_REPORTING DQ_INVESTIGATION

La sécurité autour de la Data Toutes les activités autour de la protection des données : personnes, processus et technologies Un vrai challenge aujourd'hui car la donnée est décentralisée au sein des entreprises. En réalité il existe un gap important entre ce qui est préconisé et appliqué en pratique par les sociétés = nous sommes toujours dans une approche en silos Les consultants aujourd'hui recommandent d'adopter une vision dite data-centric de la sécurité. Les entreprises devraient se baser sur une vision autour de la sécurité des données plutôt que la sécurité des serveurs, réseaux et applications

NB une combinaison qui marche est Apache Atlas (définit les politiques de sécurité) et Apache Range (autorisation) = tous les deux dans l'environnement Hadoop, à faire coïncider avec le référentiel de Meta données car tout ne se base pas sur Hadoop ;

La sécurité doit se baser sur une vision unifiée de la sécurité au sein des architectures et être incorporée dans les architectures dès le départ.

Est ici introduite une notion clé : la notion d'accès via les rôles Comporte le désavantage d'être statique Une évolution vers une gestion de type ABAC Attribute Based Access Control - est alors recommandée.

Comment sécuriser sa couche de données : RBAC = Role Based Access Control

2 solutions: The Security Policy Engine (SPE) The Intelligent Learning Engine (ILE)

Chapitre 8. Data et création de valeur

Lorsqu'on créée des solutions deux types de besoins doivent être considérés

- `des besoins fonctionnels`

  Nécessité de comprendre comment l'information transite et de mettre cette connaissance en relation avec des besoins dits Business. Selon ces besoins, des techniques telles que la BI, le machine learning ou encore l'analyse de données en temps réel peuvent être alors utilisées

- `besoins non fonctionnels`

  Dans certains cas l'utilisation de base de données aux technologies bien particulières peuvent server à traiter certains cas d'usage complexe

Il est recommandé d'utiliser des modèles standardisés, ainsi que des blocs, couplés avec une bonne gouvernance

Avant toute chose, revenons aux modèles de consommation de la donnée via le RDS - Lorsqu'un vrai besoin de créer de la donnée a des fins business puis de la stocker se fait sentir - Lorsque le volume de données traits est tel que le RDS ne suffit plus

D'où l'importance du DDS : Domain data store (self-service) Répond au besoin de stockage et de mise à disposition des données Se situe entre la couche de données et les applications consommatrices de données

Chaque DDS vient avec des outils qui lui permettent de répondre aux divers cas d'usage et doit surtout venir avec un modèle opérationnel cible afin d'éviter la sur complexification des architectures SI Ce modèle opérationnel doit cibler toute une communauté de professionnels : data analysts, data scientists, data engineers et les utilisateurs finaux.

1) Quels sont les besoins dits Business Challenge majeur de la création de la valeur via l'utilisation de la donnée est que les outils de BI sont le plus souvent disséminés à travers une entreprise et le fait que le Data engineering est souvent dissocié de la mise en place de modèles analytiques

Avant toute chose il faut donc déterminer quels sont les besoins Business d'une société et ainsi le volume de données qui va être traité.

2) Besoins non fonctionnels La liste est longue mais voici quelques critères majeurs à considérer (approche top down recommandée)

- `Comment est structure la donnée`
- `Est ce que vos requêtes sont predicted ou unpredicted`
- `Quel type de requêtes est utilisé`
- `Quels sont besoins en ce qui concerne la donnée à ce jour, historisée et archive`
- `Quel arbitrage souhaite entre performance et qualité/précision de la solution`
- `Volume de donnes et Vitesse d'échange`
- `Quels type d'opérations nous souhaitons que notre base de données effectue`
- `Quel est le protocole d'accès à la donnée souhaité`
- `Quelle est le degré de flexibilité souhaité`
- `Et bien sur le cout, le fait que ce soit open source ou pas et l'intégration avec d'autres compos`

3) Construction d'une Data pipeline et d'un modèle de données

Nous avons souvent besoin d'un ETL lorsqu'on déplace la donnée d'un contexte vers un autre

SQL ou non-SQL pipelines ? Une pipeline SQL demande une bonne connaissance des tables et plus d'étapes de validation vs plus de flexibilité

Le pipeline doit adresser les problèmes classiques rencontrés avec les ETL (contraintes, complétude, "propreté" protection de l'information La pipeline toute entière doit se baser sur des Métadonnées et un lineage. Il est intéressant pour cela pour cela de classer les ETL et leurs utilisations dans un portfolio.

Voici les schémas de modèle de données les plus utilisés : - Un Data Store dit transactionnel - BI stores - Des Data Stores dits analytiques - Des Data Stores dits harmonized

Chapitre 9. Gestion des Data assets

Qu'est-ce que le Master Data Management

Le MDM permet de gérer et distribuer les données critiques ou clés afin d'en assurer la qualité, la fiabilité et la cohérence/pérennité dans le temps Se concentre sur détecter et résoudre les irrégularités liées à la Data, et la gestion des données de référence mais surtout des Master Data ? Il vient généralement avec un livre blanc qui donne les grands principes de ce MDM.

Plusieurs styles de Master Data Management, qui ne sont pas exclusifs et peuventêtre combinés

Consolidation style Très lié au DataWarehouse Généralement implémenté pour de la BI et du reporting

Registry Le modèle le plus simple, caractérisé par un tableau croisé des références qui permet d'identifier les doublons

Centralized style Coexistence Style Utilisé dans les situations où les données de références ne peuvent pas être utilisées de façon centrale et doivent être réparties à travers plusieurs endroits dans l'entreprise

L'architecture du MDM Pour permettre le MDM, intégrer la donnée est fondamental. Le MDM doit reposer sur les mêmes bases que celles utilisées pour distribuer et intégrer toutes les données. Il est très important de ne pas faire des connexions point à point entre le MDM hub et les applications génératrices et consommatrices de données

Il est important de différencier quels jeux de données font partie de l'entreprise et lesquels sont propres à des domaines spécifiques (via l'usage de Meta Donnée°

MDM et Data Quality as a service L'auteur recommande de fournir le MDM as a service

Curated Data Le procédé qui consiste à collecter de la donnée de sources diverses puis à l'intégrer afin que celle-ci prenne plus de valeur que lorsqu'elle est gérée individuellement. Différence principale avec le MDM = n'indique pas si la donnée existante doit être modifiée et à quel niveau Le MDMD n'est pas supposé changé le sens de la donnée ou en créer de la nouvelle, ainsi les Data Owners ont toujours la possibilité de retracer le chemin de la donnée. Data curation peut impliquer des nouveaux rôles, de créer ou de donnée un nouvel usage à la donnée.

Chapitre 10. Démocratisation de la donnée via les Metadonnées

Gestion des Metadonnées Donnée qui écrit la Donnée «Data About Data », terme né dans les années 1990 Construction d'un modèle de données qui représente toute l'entreprise et les relations qui la composent Identifier et définir les Meta données critiques et l'approche architecturale la plus appropriée Démocratiser leur utilisation, via notamment des portails en libre-service

Le modèle de Metadonnées

Afin de le construite, voici la question primordiale Quelle Meta donnée business est la plus importante/la plus critique ? Afin de construite les bases de son Meta Modèle, il faut commencer simplement : Lister les applications, Les sources de données, les schémas d'interface, les propriétaires de la donnée et la sécurité

Au cœur du modèle de Meta données sont : Les jeux de données et les objets Data Data Models ou modèles de données Les modèles d'interface et le linéage La Data Gouvernance et la sécurité

Gestion des Meta données : l'approche architecturale L'auteur met en avant le besoin de créer une couche de Meta données qui permettre l'échange des Meta Données entre les différentes parties. Il existe ainsi plusieurs modèles : Le modèle centralisé ou consolidé Toutes les Metadonnées sont gérées dans une base de Meta données centralisée Le modèle fédéré ou décentralisé Les Meta données sont gérées localement et regroupées uniquement lorsqu'on en a besoin Une approche hybride dite partagée

DICTIONNAIRE DES ACRONYMES

ABAC Attribute Based Access Control

API Application Program Interface

BI Business Intelligence

ETL Extract Transform Load

CCCA Chartered Compliance and Cyber Analyst Program

CQRS Command Query Responsability Segregation

DAMA DBOK Data Management Body of Knowledge

DDD Domain Driven Design

DDS Domain data store

DSAA Data sharing agreement administration

GDPR General Data Protection Regulation

ILE The Intelligent Learning Engine

IS Information System

LAN Local Area Network

MDM Master Data Management

RDS Remote Desktop Service

SOA Service Oriented Architecture

RBAC Role Based Access Control

RACI Responsibility Assignment Matrix

RFA Responsable Fonctionnel d'Application

SAAS Server As A Service

SPE The Security Policy Engine

SQL Structured Query Language

WAN Wide area Network