ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

Base de Données Avancées (Ecosystème Big Data)
(4IIR — Spring 2025)

Chapitre 02 :

L'Écosystème du Big Data

Architecture, Intégration et Fonctionnalités pour l'Analyse des Données Massives …

Dr. Ghezlane Halhoul Merabet
G.Halhoulmerabet@emsi.ma

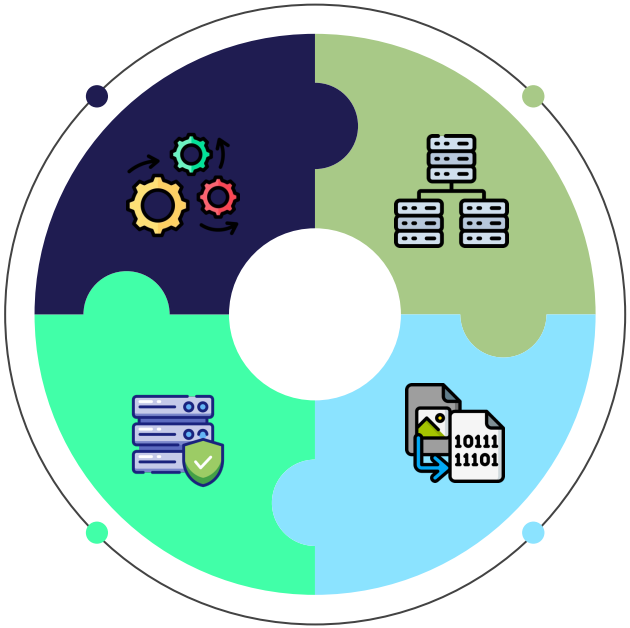
Objectives du Chapitre

1. Comprendre l'**architecture en couches** d'un système Big Data.
2. Identifier les **principales composantes** de l'**écosystème** Big Data et leurs interactions.
3. Expliquer les différentes **stratégies d'intégration** avec les **systèmes existants**.
4. Distinguer les principaux **paradigmes** de **traitement** et **stockage** de données massives

Principes Fondamentaux de l'Architecture Distribuée

Parallélisation Native

Conception des algorithmes pour exécution simultanée sur différentes portions de données.



Distribution Horizontale

Répartition des données et traitements sur plusieurs machines standard plutôt que sur un serveur unique ultra-puissant.

Tolérance aux Pannes

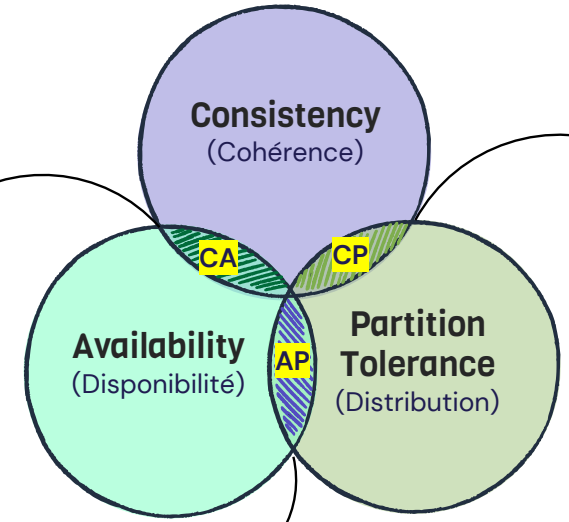
Le système anticipe et gère les défaillances matérielles comme des événements normaux et attendus plutôt qu'exceptionnels.

Localité des Données

Le traitement est déplacé vers les données plutôt que l'inverse, minimisant les transferts réseau coûteux.

*** Evolution naturelle face aux limites des architectures monolithiques vues au Chapitre 01.**

Le 'Théorème CAP' dans l'Écosystème Big Data



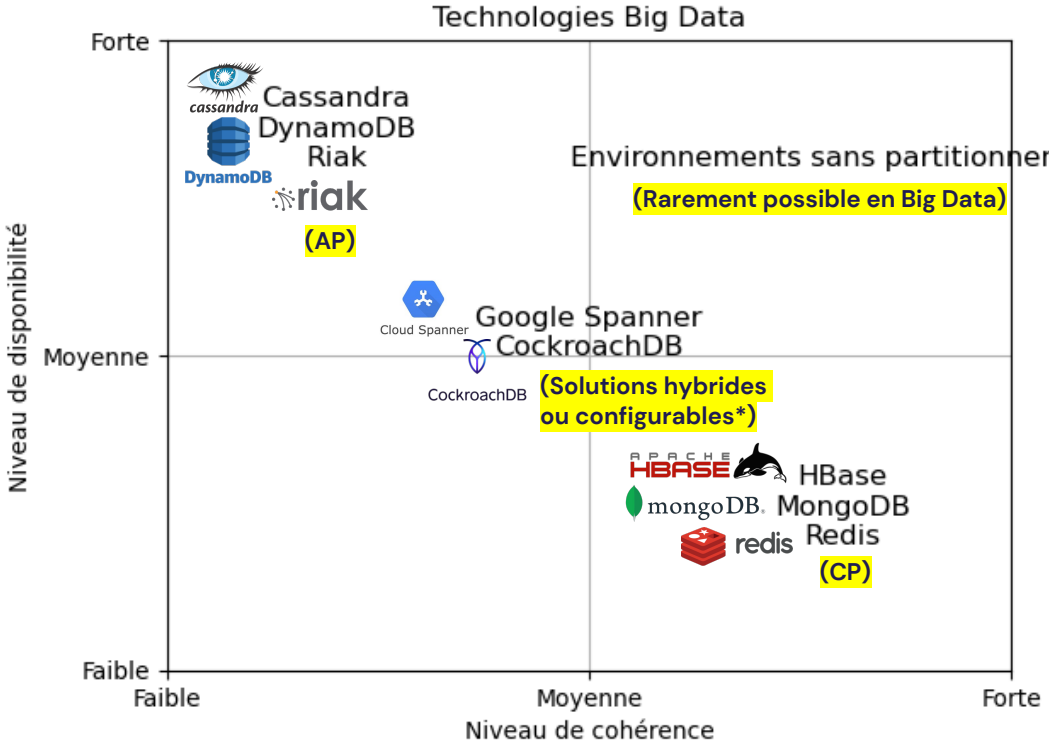
Systèmes Traditionnels
Environnement sans partitionnement (Centralisé)

Systèmes Transactionnels
Gèrent de nombreuses transactions courtes en temps réel pour les opérations quotidiennes (réservations, banque, inventaire).

Systèmes Analytiques
Analysent de grands volumes de données historiques pour l'aide à la décision via des requêtes complexes et agrégations multidimensionnelles.

NOTE :
Impossible d'obtenir les 3 propriétés simultanément en environnement distribué !

Positionnement des Technologies Big Data—Théorème CAP



NOTE :
Le choix dépend du cas d'usage et des exigences métier !

* Systèmes combinant les caractéristiques des bases relationnelles traditionnelles et des technologies NoSQL distribuées.

Exercice : Comparaison des Architectures de Données

Complétez les cases vides du tableau :

Caractéristique	Architecture Traditionnelle	Architecture Big Data
Principe de base	**Centralisation des données et des traitements**	Distribuer les données et les traitements
Unité de traitement	Serveur monolithique	**Multiples serveurs interconnectés (cluster)**
Que se passe-t-il en cas de panne ?	**Interruption complète du service**	Le système continue à fonctionner
Comment augmenter la capacité ?	Acheter un serveur plus puissant (scaling vertical)	**Ajouter plus de serveurs (scaling horizontal)**
Exemple d'utilisation idéale	**Applications transactionnelles, petites bases de données**	Analyse de millions de tweets

Anatomie d'un Système Big Data (Architecture en Couches)

COUCHE De VISUALISATION
- Représentation graphique des insights
- Création de tableaux de bord interactifs
- Exposition via APIs pour intégration
- Alertes et notifications basées sur des conditions

Outils de business intelligence

COUCHE D'ANALYSE
- Requêtage et exploration interactive
- Exécution de modèles analytiques
- Abstraction de la complexité sous-jacente
- Exposition des résultats aux couches supérieures

Entrepôt de données avec interface SQL

Interfaces analytiques sur Spark

COUCHE DE TRAITEMENT
- Exécution distribuée des transformations
- Contrôle des tâches
- Gestion des ressources computationnelles
- Optimisation des exécutions parallèles

Modèle de programmation pour le traitement par lots

Moteur de calcul unifié supportant batch, streaming et requêtes interactives

COUCHE DE STOCKAGE
- Stockage résilient et répliqué des données brutes
- Gestion des métadonnées
- Optimisation de l'accès aux données

Système de fichiers distribué optimisé pour grands fichiers et lectures séquentielles.

Bases NoSQL distribuées pour accès aléatoire rapide aux données.

COUCHE D'INGESTION
- Capture de données depuis des sources hétérogènes
- Validation et filtrage préliminaire
- Mise en tampon pour absorber les pics de charge
- Acheminement vers le stockage approprié

Plateforme de streaming distribuée traitant des millions d'événements/sec.

Collecteur de données optimisé pour l'ingestion et le transfert de logs.

SOURCES DE DONNÉES DIVERSES
(Logs, Applis, Capteurs, Bases Existantes...)

Exemples de Technologies Représentatives

Infrastructure Physique et Déploiement

L'architecture physique d'un système Big Data s'organise typiquement en une hiérarchie de nœuds spécialisés:

Nœuds de Contrôle (Maîtres)

- Coordonnent l'ensemble du cluster
- Gèrent les métadonnées et l'état global
- Orchestrent la distribution des tâches
- Généralement déployés en redondance pour la haute disponibilité

Nœuds de Traitement et Stockage (Esclaves)

- Constituent la majorité des machines du cluster
- Stockent les fragments de données distribués
- Exécutent les tâches de calcul parallèle
- Conçus pour être facilement remplaçables

Principes Fondamentaux :

- **Redondance distribuée :** Réplication des données et services critiques
- **Scaling horizontal :** Expansion par ajout de nœuds supplémentaires
- **Réseau optimisé :** Infrastructure réseau adaptée aux transferts massifs de données

Intégration avec les Systèmes Existants

Comment connecter le Big Data avec les systèmes actuels ?

L'adoption du Big Data dans une organisation n'implique pas le remplacement complet des systèmes existants, mais plutôt leur intégration judicieuse avec le nouvel écosystème.

1. **Extraction périodique :** Copier les données des anciens systèmes à intervalles réguliers
   Exemple : Exporter les transactions de la journée chaque nuit

2. **Capture des changements :** Surveiller et copier uniquement ce qui change
   Exemple : Enregistrer chaque nouvelle commande dès qu'elle est passée

3. **APIs :** Interfaces permettant aux systèmes de communiquer
   Exemple : Permettre au système de recommandation d'interroger les catalogues produits

Intégration avec les Systèmes Existants

Cohabitation BI Traditionnelle et Big Data

Les solutions de **Business Intelligence** (**BI**) traditionnelles peuvent coexister et se compléter avec l'écosystème Big Data :

Modèles de cohabitation :

- **Big Data comme source pour BI :** Prétraitement et agrégation dans l'environnement Big Data avant chargement dans le datawarehouse.
- **BI comme couche de présentation :** Utilisation des outils BI familiers connectés directement aux services analytiques Big Data
- **Approche hybride :** Combinaison des deux mondes selon la nature des données et des requêtes

Exemple d'architecture hybride: Un retailer utilisant

- Hadoop pour le stockage et prétraitement des données volumineuses (logs web, données IoT)
- Un data Warehouse traditionnel pour les données critiques et hautement structurées
- Des outils BI connectés aux deux environnements selon les besoins d'analyse

Intégration avec les Systèmes Existants

Batch vs Streaming : Deux façons de traiter les données

Traitement par lots (Batch) :

- Traitement périodique de blocs de données accumulées
- Fenêtres de temps définies (horaire, quotidien, hebdomadaire)
- Optimisé pour l'efficacité et le débit global
- **Inconvénient :** Résultats non immédiats
  Exemple : Rapport quotidien des ventes

Traitement en continu (Streaming) :

- Traitement continu des données à mesure qu'elles sont générées
- Latence minimale entre génération et traitement
- Adapté aux cas d'usage temps réel ou quasi-temps réel
- **Inconvénient :** Plus complexe à mettre en œuvre
  Exemple : Détection de fraude bancaire immédiate

Intégration avec les Systèmes Existants

Batch vs Streaming : Deux façons de traiter les données

Approche hybride (Lambda) :

- Combinaison des deux approches pour différents cas d'usage
- **Couche streaming** pour résultats immédiats approximatifs
- **Couche batch** pour résultats précis et complets

Exemple : Un système de détection de fraude bancaire utilisant :

- Streaming pour l'analyse en temps réel des transactions suspectes
- Batch pour l'entraînement périodique des modèles sur l'historique complet
- Approche Lambda pour combiner alertes immédiates et analyses approfondies

Fonctionnalités de Base du Big Data

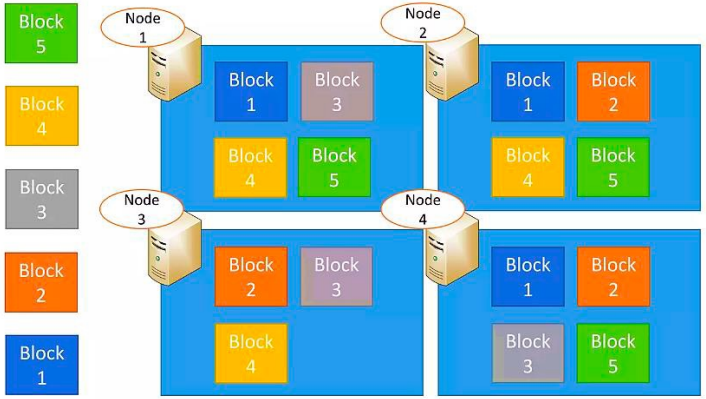
1. Stockage Distribué

Comment stocker plus de données que ne peut en contenir un seul disque dur?

Principe : Diviser les fichiers en **morceaux** (blocs) et les répartir sur plusieurs machines

Caractéristiques Importantes :

- **Réplication :** Chaque bloc est copié sur plusieurs machines.
- **Tolérance aux pannes :** Si une machine tombe en panne, les données restent disponibles
- **Localité :** Le système essaie de traiter les données sur la machine où elles sont stockées



Exemple de

Exemple : Un fichier de 1 Go pourrait être divisé en 8 blocs de 64 ou 128 Mo, chacun stocké sur 3 machines différentes.

Fonctionnalités de Base du Big Data

2. Traitement Distribué (ex. MapReduce technique)

MapReduce est un modèle de programmation fondamental en **Big Data** qui permet de traiter d'énormes volumes de données en parallèle.

Fonctionnement en 3 étapes simples:

1. **Map** (Diviser et traiter)
   - Divise le problème en sous-tâches indépendantes
   - Chaque machine traite une partie des données

   Exemple : Compter les occurrences de chaque mot dans sa portion de texte

2. **Shuffle/Split** (Regrouper)
   - Réorganise les résultats pour que toutes les données liées soient ensemble

   Exemple : Rassembler tous les comptages du même mot

3. **Reduce** (Combiner)
   - Combine les résultats partiels pour obtenir le résultat final

   Exemple : Additionner tous les comptages pour chaque mot

Fonctionnalités de Base du Big Data

3. Types de Stockage Big Data

En fonction des besoins, différentes solutions de stockage sont utilisées :

1. **Systèmes de fichiers distribués** (ex. : HDFS*)
   - Stockent de grandes quantités de données brutes
   - Optimisés pour la lecture séquentielle de fichiers volumineux
   - **Cas d'usage :** Stockage de logs, fichiers texte, images

2. **Bases NoSQL** (ex. : MongoDB, Cassandra)
   - Plus flexibles que les bases SQL traditionnelles
   - Différents types selon les besoins (clé-valeur, document, colonne, graphe)
   - **Cas d'usage :** Profils utilisateurs, catalogues produits, réseaux sociaux

*** HDFS : Hadoop Distributed File System.**