Supervision des Data Pipelines : Revue et Proposition d'Architecture

Nawar TOUMI
Amina BOUHAMRA
Soukaina BOUCETTA
Fadwa EL AMRAOUI

Novembre 2025

Table des matières

Chapitre 1

Introduction

1.1 Introduction

Les pipelines de données sont devenus un composant fondamental dans les systèmes numériques modernes. Ils permettent de collecter, transformer, valider et stocker automatiquement de grandes quantités de données provenant de diverses sources. Ces pipelines facilitent l'alimentation des plateformes d'analyse, des modèles d'apprentissage automatique ou encore des outils de visualisation.

Cependant, la complexité croissante de ces systèmes introduit des risques importants liés à la fiabilité des flux de données. Les interruptions, les erreurs de format, les valeurs aberrantes ou les retards dans les traitements peuvent avoir un impact direct sur la qualité des résultats produits en aval. Dans certains cas, ces défaillances peuvent entraîner de mauvaises décisions ou l'indisponibilité de services critiques.

Face à ces défis, la supervision active des pipelines devient indispensable. Il ne s'agit plus seulement de surveiller la disponibilité des systèmes, mais aussi d'assurer une observabilité approfondie des données en transit, capable de détecter automatiquement les anomalies et de déclencher des alertes ou des corrections.

Plusieurs recherches récentes ont exploré des approches pour renforcer cette supervision. Par exemple, des méthodes basées sur l'apprentissage automatique ont été utilisées pour détecter des anomalies dans les flux complexes [4, 3]. D'autres auteurs proposent des modèles d'observabilité plus globaux, s'appuyant sur la collecte de métriques, de journaux et de traces

pour évaluer la santé d'un pipeline [5]. Enfin, certains travaux visent à formaliser la conception des architectures de traitement pour faciliter le contrôle de bout en bout [6].

Cette section vise à faire le point sur les contributions majeures de la littérature autour de la supervision des pipelines de données, en identifiant les approches utilisées, les outils mobilisés, ainsi que les limites rencontrées par les chercheurs et les praticiens.

Chapitre 2

État de l'art

2.1 Introduction

Les pipelines de données jouent un rôle fondamental dans les systèmes de traitement de données distribués. Ils permettent d'automatiser les flux de données, depuis la collecte jusqu'à l'analyse. Cependant, leur bon fonctionnement dépend de nombreux facteurs, notamment la qualité des données, la stabilité des composants et la capacité à détecter rapidement des dysfonctionnements.

Cette section vise à explorer les approches actuelles en matière de supervision des pipelines de données. Elle s'appuie sur une revue de la littérature scientifique récente afin d'identifier les principales méthodes de surveillance, les outils utilisés, ainsi que les limites des approches existantes.

2.2 Problématique et question de recherche

La supervision des pipelines de données est devenue un enjeu majeur avec la croissance du volume, de la vitesse et de la variété des données. Une interruption, même brève, dans un pipeline peut entraîner des pertes d'informations critiques ou fausser les résultats produits. Les outils classiques de surveillance (logs, alertes système) ne suffisent plus à garantir une vue d'ensemble sur l'état de santé du pipeline et sur la qualité des données en transit.

Face à cela, une question centrale se pose :

Comment concevoir une supervision intelligente, modulaire et adaptable des pipelines de données, permettant une détection efficace des anomalies tout en restant compatible avec les environnements distribués et hétérogènes ?

Cette problématique s'inscrit dans un contexte où les solutions doivent être non seulement efficaces, mais également intégrables avec des outils existants comme Apache NiFi, PostgreSQL, ou Power BI.

2.3 Revue de la littérature

2.3.1 Supervision dans les pipelines de données

Dans un contexte où les volumes de données sont en constante augmentation, la supervision des pipelines de données s'impose comme une exigence fondamentale pour garantir leur fiabilité et leur performance. La supervision peut être définie comme l'ensemble des mécanismes mis en place pour surveiller l'état opérationnel du pipeline, détecter les dysfonctionnements, et permettre une intervention rapide en cas d'erreurs.

Elle englobe des actions telles que la validation du format et du schéma des données, la vérification de la complétude, la mesure des temps de latence entre les étapes, ou encore la détection d'échecs de transformation. Cette supervision peut s'effectuer à plusieurs niveaux : supervision des données elles-mêmes (data-level), supervision des processus (task-level), ou supervision des ressources d'exécution (infrastructure-level).

Dans la littérature, plusieurs travaux proposent des architectures intégrant des modules de supervision temps réel. Par exemple, De Haro-Olmo et al. [1] présentent un pipeline orienté IoT qui inclut des mécanismes de curation des données et de contrôle qualité, capables d'identifier des incohérences ou des interruptions de flux. De leur côté, Hernandez et al. [2] conçoivent une plateforme intelligente (ERAIA) qui supervise automatiquement les données transitant entre capteurs, cloud et interfaces utilisateurs.

D'autres approches privilégient l'intégration de la supervision dès la phase de conception du pipeline. C'est le cas de Pasupuleti [4], qui propose un pipeline augmenté par l'intelligence artificielle, incluant des modules de vérification du schéma, de suivi de performance, et de gestion des erreurs. Une telle approche permet d'automatiser le diagnostic et de faciliter la maintenance du pipeline en production.

En résumé, la supervision ne se limite pas à une simple surveillance passive ; elle implique une instrumentation proactive du pipeline, afin d'anticiper les problèmes potentiels et de garantir une qualité de service continue.

2.3.2 Observabilité et détection d'anomalies

L'observabilité représente une évolution conceptuelle par rapport à la simple supervision. Alors que la supervision consiste principalement à surveiller des indicateurs clés de performance (KPI) prédéfinis tels que la latence, le débit ou le taux d'erreurs, l'observabilité vise à comprendre en profondeur le comportement d'un système à partir de ses signaux internes [5]. Elle repose sur la collecte et l'analyse de trois types de données complémentaires : les *logs* (journaux d'événements), les *métriques* (valeurs numériques mesurées sur le système), et les *traces* (suivi des appels entre composants). Ces données fournissent une vue holistique qui permet aux opérateurs de diagnostiquer les causes profondes des incidents, même lorsqu'ils ne sont pas directement visibles via les indicateurs de haut niveau.

Dans le contexte des pipelines de données, l'observabilité permet notamment de détecter des ralentissements, des blocages, ou encore des incohérences dans les flux. Elle devient essentielle dans les environnements distribués ou temps réel, où les erreurs peuvent se propager silencieusement d'un composant à l'autre. Des plateformes modernes comme DataDog, Prometheus ou OpenTelemetry sont conçues pour faciliter cette collecte, mais elles nécessitent une configuration fine et une compréhension claire des métriques pertinentes.

La détection d'anomalies constitue une application directe de l'observabilité. Elle vise à identifier automatiquement des comportements inhabituels dans les données ou dans le fonctionnement du pipeline. Les méthodes existantes peuvent être classées en trois grandes catégories :

- **Les approches heuristiques**, basées sur des règles fixées manuellement (seuils, règles métiers) ;
- **Les méthodes statistiques**, qui détectent des écarts significatifs par rapport à des modèles de distribution des données (moyenne, écart-type, séries temporelles) ;
- **Les techniques d'apprentissage automatique**, qui apprennent des motifs normaux à partir de jeux de données historiques, et identifient les déviations.

Par exemple, Nasiri et al. [3] proposent un pipeline intelligent combinant

des lois physiques à un modèle supervisé de machine learning pour améliorer la détection d'anomalies dans des environnements industriels à fort bruit de données. Leur approche démontre l'intérêt de coupler des connaissances métiers (physiques) à des algorithmes d'analyse automatique.

Ainsi, l'intégration d'une couche d'observabilité avancée, combinée à des modules de détection d'anomalies intelligents, représente une condition essentielle pour garantir la fiabilité, la traçabilité et la robustesse des pipelines de données modernes.

2.3.3 Outils et modèles existants (ETL, ELT, etc.)

Les pipelines de données peuvent être construits selon plusieurs modèles d'architecture, chacun répondant à des contraintes techniques et fonctionnelles spécifiques. Parmi les plus utilisés figurent les modèles ETL (Extract-Transform-Load) et ELT (Extract-Load-Transform). Le modèle ETL est particulièrement adapté aux traitements batch traditionnels, où les données sont extraites de différentes sources, transformées dans un espace intermédiaire, puis chargées dans une base de données cible. À l'inverse, le modèle ELT privilégie un chargement rapide des données brutes dans le système de stockage, suivi d'une transformation différée, souvent effectuée directement dans le data warehouse.

En complément, les architectures orientées streaming, reposant sur des technologies comme Apache Kafka, Apache Flink ou Spark Streaming, permettent de traiter les données en continu, avec une latence minimale. Ce paradigme devient incontournable dans les contextes nécessitant une réactivité en temps réel, comme la détection d'anomalies ou la supervision d'équipements connectés.

Divers outils ont été développés pour implémenter ces modèles. Par exemple, Apache NiFi propose une interface graphique pour la gestion de flux de données, avec un support natif pour la traçabilité, le routage conditionnel et le monitoring basique. Apache Airflow, de son côté, permet l'orchestration de workflows complexes, bien qu'il soit plus orienté batch que streaming. StreamSets Data Collector offre quant à lui une solution hybride, combinant traitement batch et temps réel, avec des capacités d'observabilité renforcées.

Cependant, ces outils ne couvrent pas toujours l'ensemble des besoins en supervision. Dans cette optique, certains travaux de recherche récents cherchent à formaliser les bonnes pratiques de conception à travers des modèles enrichis comme ETLT (Extract-Transform-Load-Transform) et ELTL (Extract-

Load-Transform-Load). Ces patterns visent à améliorer la lisibilité, la modularité et surtout la traçabilité des étapes de traitement dans un pipeline complexe [6]. L'objectif est d'optimiser à la fois l'observabilité et la maintenabilité du pipeline, tout en facilitant l'intégration de composants tiers (machine learning, alerting, visualisation).

Ainsi, le choix d'un modèle de pipeline et des outils associés doit être guidé par les exigences en matière de supervision, de fréquence de traitement (batch vs temps réel), et de flexibilité d'orchestration.

2.4 Limites des approches existantes

Malgré la diversité des travaux menés sur la supervision des pipelines de données, plusieurs limites importantes subsistent dans les approches existantes. Ces limites concernent tant les aspects techniques que méthodologiques, et révèlent un écart persistant entre les solutions théoriques proposées dans la littérature et les besoins concrets des systèmes en production.

Tout d'abord, de nombreuses solutions restent fortement dépendantes du domaine d'application. Par exemple, les pipelines développés pour l'industrie pétrolière intègrent des modules spécialisés basés sur des connaissances physiques du domaine [3]. Si cette spécialisation améliore la précision dans un contexte spécifique, elle limite toutefois la généricité et la réutilisabilité de la solution dans d'autres secteurs.

Ensuite, plusieurs approches mettent l'accent sur la collecte d'informations (logs, métriques, traces), sans offrir de mécanismes intelligents d'analyse ou d'action. Le modèle d'observabilité proposé par Patel [5], par exemple, fournit une base solide pour structurer la supervision, mais n'intègre pas de composants natifs d'alerting ou d'automatisation de la réponse aux incidents. Cette lacune complique la mise en œuvre d'une supervision proactive.

De plus, bien que certains outils comme Apache NiFi ou StreamSets proposent des interfaces de surveillance intégrées, ils nécessitent souvent une configuration manuelle avancée pour prendre en charge des cas complexes (comme la détection d'anomalies multivariées ou la gestion adaptative du flux de données). La flexibilité offerte par ces outils s'accompagne généralement d'une charge opérationnelle importante.

Sur le plan architectural, les modèles classiques de pipeline (ETL, ELT) ne permettent pas toujours une bonne traçabilité des transformations successives, surtout dans des contextes hybrides mêlant données en batch et en

streaming. Des travaux comme ceux de Rucco et al. [6] proposent des patterns de conception alternatifs (ETLT, ELTL), mais ces derniers ne sont pas encore largement adoptés ni intégrés dans les solutions logicielles courantes.

Enfin, un nombre limité de solutions adoptent une approche modulaire et interopérable. Dans un environnement où les sources de données, les volumes et les outils évoluent rapidement, la rigidité des pipelines existants devient un frein à l'adaptabilité, à la scalabilité et à l'intégration de technologies émergentes telles que l'intelligence artificielle.

Ces constats soulignent l'intérêt de concevoir une solution unifiée et extensible, capable de combiner supervision, détection d'anomalies et visualisation en temps réel, tout en s'appuyant sur des outils open source flexibles.

TABLE 2.1 – Comparaison des solutions de supervision de pipelines de données

Réf.	**Outil / Système**	**Approche**	**Limites identifiées**
[4]	Pipeline IA	Machine Learning supervisé intégré au pipeline	Nécessite un jeu d'entraînement représentatif ; complexité de déploiement
[5]	Modèle de maturité pour l'observabilité	Collecte de logs, métriques et traces ; supervision globale	Implémentation technique coûteuse ; absence d'alerting intégré
[6]	Modèles ETLT et ELTL	Patterns de design pour la construction de pipelines traçables	Non implémenté dans les outils standards ; manque d'automatisation
[3]	Pipeline physique dans l'industrie pétrolière	Filtrage basé sur les lois physiques + ML	Domaine spécifique (Oil & Gas) ; peu généralisable à d'autres secteurs
[1]	ELI Pipeline	Pipeline IoT avec curation et contrôle qualité des données	Complexité des pipelines IoT ; dépendant du domaine applicatif
[2]	ERAIA Platform	Architecture intelligente pour les applications IoT	Peu d'accent sur la détection d'anomalies ; pas d'alerting intégré

Ce tableau met en évidence la diversité des approches utilisées pour superviser les pipelines de données. Tandis que certains travaux mettent l'accent sur la détection automatique d'anomalies (Pasupuleti, Nasiri), d'autres se concentrent davantage sur l'architecture ou la modélisation du pipeline (Rucco, Hernandez). Toutefois, peu d'entre eux proposent une solution complète combinant acquisition, traitement, supervision et alerting, ce qui justifie le choix d'une architecture modulaire dans notre solution.

2.5 Conclusion

Ce chapitre a permis de mettre en lumière les principales approches de supervision des pipelines de données, ainsi que les outils et modèles utilisés. Malgré les avancées observées, des besoins restent non satisfaits, notamment en termes de modularité, d'interopérabilité et de détection intelligente d'anomalies.

Ces constats ouvrent la voie à la proposition d'une architecture de pipeline supervisé, intégrant des outils open source et des mécanismes d'alerte en temps réel. Cette proposition sera détaillée dans le chapitre suivant.

Chapitre 3

Conception de la solution proposée

3.1 Objectifs de la solution

La solution proposée vise à répondre aux limites observées dans la revue letterature, notamment le manque d'interopérabilité, l'absence d'alertes intelligentes et la complexité d'intégration dans les environnements industriels. L'objectif est de concevoir une architecture de pipeline supervisé, capable de :

- automatiser l'acquisition, le traitement et le stockage des données,
- détecter les anomalies en quasi temps réel,
- générer des alertes,
- et permettre une visualisation intuitive de l'état du pipeline.

3.2 Architecture globale du pipeline de supervision

La figure suivante illustre l'architecture fonctionnelle de la solution. Elle intègre des composants pour la collecte de données via API, leur normalisation, leur stockage, leur agrégation et leur visualisation. Elle repose sur une architecture modulaire facilitant la maintenance et l'extensibilité.

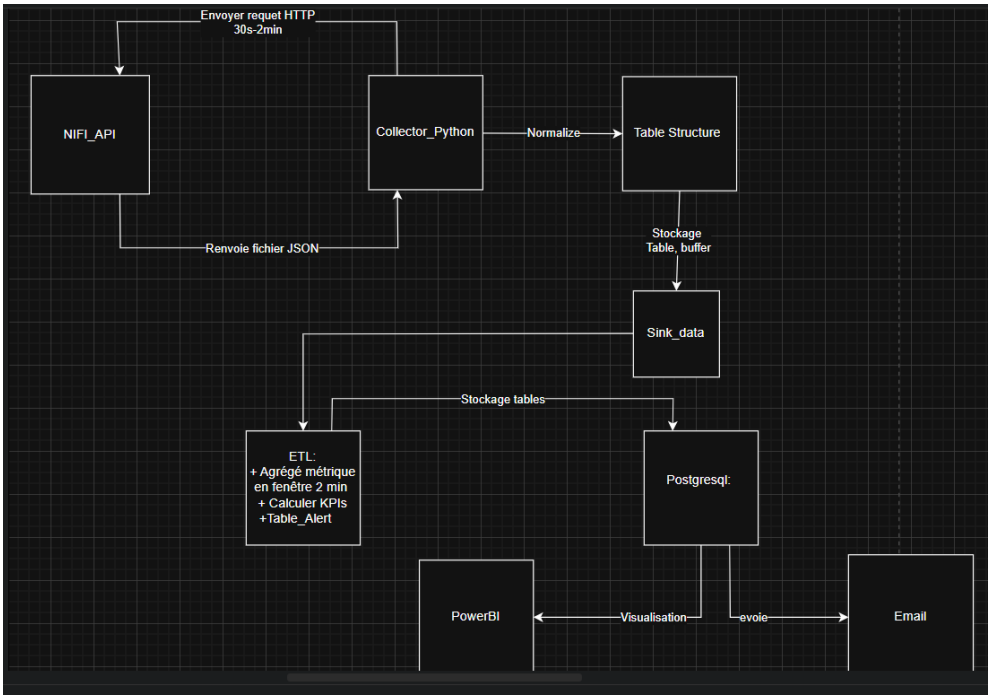

FIGURE 3.1 – Architecture proposée

3.2.1 Description des composants

— **NIFI_API** : envoie régulièrement des requêtes HTTP (toutes les 30s à 2 min) vers des endpoints d'API.
— **Collector_Python** : reçoit les fichiers JSON, normalise les données pour unifier le schéma.
— **Table Structure** : définit les colonnes, types et structures de la table normalisée.
— **Sink_data(à définir)** : tampon temporaire pour éviter les pertes lors de retards dans l'écriture.
— **PostgreSQL** : stockage relationnel permanent des données historisées.
— **ETL** : effectue l'agrégation de métriques (fenêtres de 2 minutes), le calcul des KPIs et la génération de la table d'alertes.
— **Power BI** : outil de visualisation des indicateurs.
— **Email** : système d'alerte automatique basé sur les seuils définis dans la table d'alerte.

3.2.2 Fonctionnement du Data pipeline

Le processus commence par la collecte de données brutes via NiFi, qui sont ensuite traitées en Python pour assurer leur intégrité structurelle. Après normalisation, les données sont temporairement stockées dans un Sinkdata (à définir). Elles sont ensuite transférées dans PostgreSQL. Un module ETL s'exécute toutes les 2 minutes pour : - agréger les données selon des fenêtres temporelles, - calculer des indicateurs clés de performance (latence, débit, erreurs), - générer des alertes si des seuils sont dépassés.

Enfin, les résultats sont visualisés via Power BI, et les alertes sont envoyées par email en cas d'anomalie.

3.3 Choix des technologies

3.3.1 Apache NiFi

Apache NiFi est un outil open-source d'orchestration de flux de données. Il permet la collecte de données à partir de sources variées (API, fichiers, bases de données), avec une interface graphique et un système de logs. Son architecture orientée flux convient parfaitement à notre cas d'usage.

3.3.2 PostgreSQL

PostgreSQL est une base de données relationnelle robuste, idéale pour stocker les métriques historisées et les résultats du traitement ETL. Elle supporte les requêtes complexes et l'indexation temporelle, ce qui est nécessaire pour le calcul des KPIs.

3.3.3 Power BI et alerting

Power BI est utilisé pour créer des tableaux de bord dynamiques qui permettent de suivre l'évolution du pipeline en temps réel. L'alerte est déclenchée via un processus automatisé (script ou webhook) qui lit les données de la table Alert et envoie des emails selon des règles définies.

3.4 Apport par rapport à l'état de l'art

La solution proposée se distingue des approches de la littérature par plusieurs points :

- **Interopérabilité** : elle utilise uniquement des outils open-source largement compatibles (NiFi, PostgreSQL, Power BI).
- **Détection rapide** : l'agrégation temporelle et le calcul de KPIs permettent une surveillance fine avec génération d'alertes automatisées.
- **Modularité** : chaque composant est indépendant, ce qui facilite l'extension ou la modification future.
- **Visualisation intégrée** : les utilisateurs finaux peuvent consulter les métriques sans connaissance technique.

3.5 Conclusion

Cette architecture apporte une solution concrète aux limites identifiées dans la littérature. Elle permet de superviser un pipeline de données en assurant la qualité, la disponibilité et la transparence des flux. Le prochain chapitre portera sur l'implémentation technique et les résultats obtenus.

Conclusion générale

Dans un contexte où les volumes de données continuer à s'augmenter, la supervision des pipelines de données devient un enjeu central pour garantir la qualité, la fiabilité et la continuité des flux. Ce mémoire s'est intéressé à la conception d'une solution technique permettant de détecter les anomalies dans un pipeline de données, en s'appuyant sur des outils modernes et des principes d'observabilité.

Dans un premier temps, un état de l'art a permis de mettre en lumière les approches existantes, les limites observées dans la littérature, ainsi que les outils les plus utilisés comme Apache NiFi, PostgreSQL ou encore Power BI. Ce travail a également permis d'identifier le manque de solutions simples, interopérables et capables de déclencher des alertes en temps réel en cas d'anomalies dans les flux de données.

Sur la base de ces constats, une architecture modulaire a été proposée. Elle intègre plusieurs composants : un collecteur de données (via API), un module de normalisation, un tampon temporaire (*Sink_data*), une base de données relationnelle pour l'historisation, un moteur d'agrégation de KPIs, un système d'alerte automatisé, ainsi qu'un tableau de bord de visualisation. Cette solution répond aux exigences de modularité, de supervision automatisée et de traçabilité.

L'implémentation opérationnelle de cette solution pourra permettre de tester son efficacité sur des cas d'usage concrets. À l'avenir, plusieurs pistes d'amélioration peuvent être envisagées : l'intégration de modèles prédictifs pour anticiper les anomalies, l'adaptation de la solution à des environnements Big Data, ou encore la généralisation du système à d'autres types de pipelines (temps réel, batch distribué, etc.).

Ce travail contribue ainsi à la réflexion autour de la gouvernance des données en proposant une solution accessible, robuste et évolutive pour la supervision des pipelines dans un contexte industriel.

Bibliographie

[1] De Haro-Olmo, F. J., Ramos-Munoz, J. J., Baena, E. et al., "ELI : An IoT-Aware Big Data Pipeline with Data Curation and Data Quality," *PeerJ Computer Science*, vol. 9, octobre 2023, p. e1605.

[2] Hernandez, A., Venero, L., et al., "ERAIA - Enabling Intelligence Data Pipelines for IoT-based Application Systems," in *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, IEEE, 2020, pp. 1–9.

[3] Nasiri, M., Chalvatzis, K., et al., "Enhancing Anomaly Detection in Oil and Gas Rotating Machinery through a Physics-Informed Data Filtration Pipeline," Preprint, 2023.

[4] Pasupuleti, S., "AI-Augmented Data Pipelines : Integrating Machine Learning for Intelligent Data Processing," *Journal of Computer Science and Technology Studies*, vol. 7, no. 11, 2025, pp. 276–283.

[5] Patel, H. R., "A Maturity Model for Observability in Big Data Pipelines," *Journal of Computer Science and Technology Studies*, vol. 7, no. 11, 2025, pp. 195–202.

[6] Rucco, C., Saad, M., and Longo, A., "Formalizing ETLT and ELTL Design Patterns and Proposing Enhanced Variants," Working Paper, University of Salento, 2025.