POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Ensuring High Data Quality Standards: A Framework for Single and Cross-Enterprise Platforms

Laurea Magistrale in Computer Engineering - Ingegneria Informatica

Author: Leonardo Mandruzzato

Advisor: Prof. Marco Brambilla

Academic year: 2022-2023

1. Introduction

Data platforms have become an indispensable technology for organizations, enabling them to make data-driven decisions. However, the quality of data ingested from multiple sources is a critical factor affecting the reliability and validity of such decisions. This summary outlines a comprehensive framework for enhancing data quality in multi-source analytics platforms across various organizational contexts. By utilizing a proactive approach incorporating prevention and detection techniques, the framework offers organizations a robust tool for safeguarding the quality of their data and, by extension, their decision-making processes.

2. Importance of Data Quality

In the modern enterprise, data is the lifeblood that fuels everything from operational efficiency to customer engagement. Data quality, therefore, directly impacts an organization's bottom line. From ensuring customer satisfaction through personalized services to achieving compliance with regulatory requirements, the importance of high-quality data cannot be overstated. Especially in multi-source environments, poor data quality can result in a "Garbage In, Garbage Out" (GIGO) scenario, leading to a cascade of errors and inefficiencies across multiple organizational functions, up to the point they cause real "datastrophes" [2], bringing enterprises to experience severe repercussions.

3. Addressed Challenges

This framework addresses multiple challenges that organizations face in ensuring data quality. The following subsections detail the critical challenges targeted by the proposed framework.

3.1. From Reactive to Proactive

Traditionally, approaches to data quality are reactive, causing inefficiencies and inaccuracies that affect downstream analytics and decision-making processes. Organizations often wait for data anomalies to be discovered by analysts before taking corrective action. This reactive model needs to be revised. For one, it delays issue identification, exacerbating its impact. Moreover, by the time the issue is identified, the engineers responsible for the particular data pipeline may have moved on to other projects, making remediation challenging. The framework aims to shift this mindset by promoting proactive monitoring and validation of data, allowing organizations to automatically discover quality issues before they escalate.

3.2. Detection and Prevention

Even when organizations adopt automated systems to identify data quality issues, they often focus solely on detection, neglecting the equally crucial aspect of prevention. This unilateral focus can result in a perpetual “fire-fighting” cycle, as problems are continuously detected but not prevented. An essential contribution of this framework is its integrated approach that combines detection with prevention. This dual focus minimizes issues that arise when data producers and consumers make independent changes to data schemas or semantics, ensuring that upstream data changes do not break downstream analytics [5].

3.3. Cross-Enterprise Scenarios

Data quality management becomes exponentially complex in cross-enterprise scenarios, where data producers belong to different external organizations involved in data-sharing collaborations. The risks associated with reactive and detection-only approaches are further amplified in such settings, making a proactive and preventive approach indispensable. The proposed framework addresses this challenge by offering tools and methodologies specifically designed to manage data quality in cross-enterprise collaborations. These tools ensure the data exchange adheres to agreed-upon contracts and standards, safeguarding data quality even in complex multi-organization scenarios.

4. Contribution

The principal contribution of the thesis is the introduction of a comprehensive framework that addresses the multiple facets of data quality management in multi-source analytics platforms. Unlike traditional approaches, the proposed framework introduces a proactive methodology integrating detection and prevention techniques. A unique aspect of this framework is its flowchart-based decision tool that guides organizations in identifying and deploying the most appropriate data quality solutions for their needs. Indeed, the framework is versatile, offering threefold technological solutions that might or might not be implemented according to the organizational context. Finally, the framework is not just theoretical; it has been empirically validated through real-world case studies, proving its practical effectiveness and adaptability.

5. Framework Overview

The core of this framework consists of three pivotal components designed to cater to various organizational data quality needs and intra- and inter-enterprise contexts.

Data Quality Assessment (DQA) Solution: The baseline unit can be applied to any organizational scenario. It employs Data Quality platforms and tools to continuously evaluate the status of the data flowing into and within the analytics platform. It integrates alerts, key metrics, and dashboards that allow data custodians to automatically spot data inconsistencies and understand the health of data quality across sources. It covers the issue detector role within the framework, and it can be integrated with existing data pipelines and adapt to new data sources effortlessly. For what concerns its applicability, it is highly recommended for organizations that are initiating their journey toward data quality management.

Data Contracts (DC): These are specifically designed for intra-organizational data exchange. Technically, they are “code agreements” between data producers and consumers that mainly define the data type and schema constraints [5]. The Data Contracts must be integrated into a technological infrastructure that allows users to define, enforce, fulfill, and monitor them. When correctly enforced and fulfilled, they prevent two primary things from happening: merging code that potentially breaks downstream pipelines and generating data that is not compliant with the agreed specifications. They are ideal for larger organizations where multiple departments produce data flowing into the analytics platform, ensuring standardization and consistency.

Push API (PAPI): This is a way to facilitate data quality management in cross-enterprise collaborations. The system follows a *push-based approach* [3] where a central entity designs and develops an API that the sources, usually external organizations, will use to share data. They do so via an API request that triggers a back-end logic that automatically enforces the data meets all the specifications collected during the requirements gathering phase. It is applicable in scenarios where organizations collaborate with external partners and must maintain data quality without complete control over external data sources.

5.1. Decision Tool

Complementing these core components is a flowchart-based decision tool (depicted in Figure 1) designed to help organizations navigate through their data quality journey. It walks users through selecting the appropriate components and configuration options, thereby customizing the framework to each organization's unique needs.

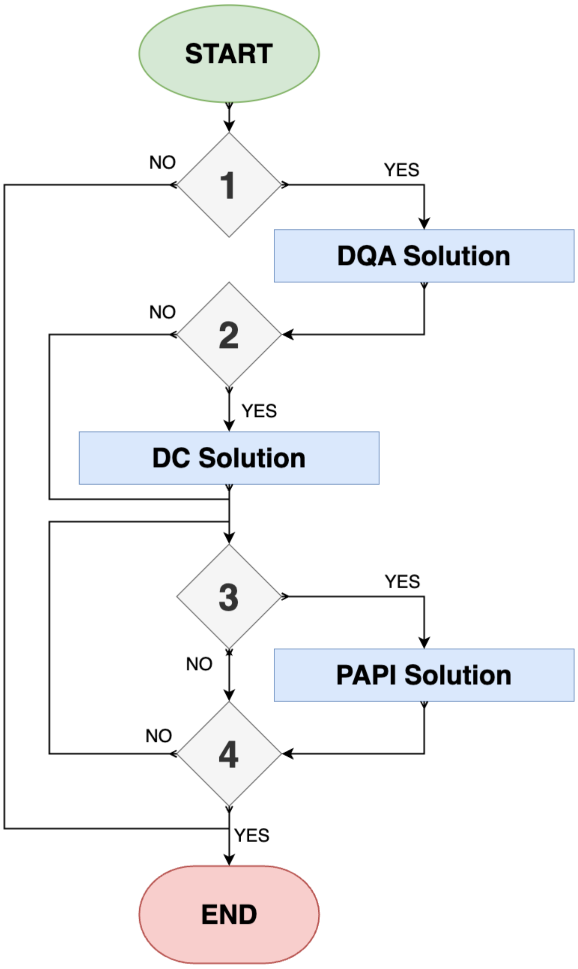

Figure 1: Decision process of the comprehensive framework in the form of a flowchart.

Each *decision symbol* corresponds to a set of questions, and the final decision is derived by combining answers logically via an `AND` operation.

1. "Has the organization reached the proper data maturity stage?" `AND` "Has the organization started a more "enterprisey" approach to data?"
2. "Do you need to integrate multiple internal data sources?" `AND` "Does the data team waste substantial time interacting with upstream software engineers to resolve data issues?" `AND` "Are upstream teams willing to change part of their infrastructure?"
3. "Is there a cross-enterprise data-sharing agreement in place?" `AND` "Can the central entity force the data sources to adopt a specific technological solution to share the data?" `AND` "Is the data shared from the different organizations/sources similar in nature, format, and structure?"
4. "Are the cross-enterprise collaborative data-sharing agreements finished?"

5.2. Integration

These components can operate individually or can be integrated into a unified solution. When used in concert, they provide a robust and comprehensive approach to improving and maintaining data quality in complex, multi-source analytics platforms.

6. Case Studies

This section presents three real-world case studies. Each of them overviews one of the three pivotal components of the final framework: Data Quality Assessment solution, Data Contracts, and Push API.

6.1. Case Study 1: Startup with a limited portfolio of data products

Context: As a startup, the organization was in the early stages of analytics adoption, with a limited portfolio of data products [1]. A detection-centric approach proved adequate for kickstarting their data quality enhancement journey.

Technology: The Data Quality Assessment solution came into play here. The organization integrated a data quality tool into its analytics platform [6] and, more precisely, into its ETL processes; this allowed real-time validation against semantic and schema specifications established during the requirement collection

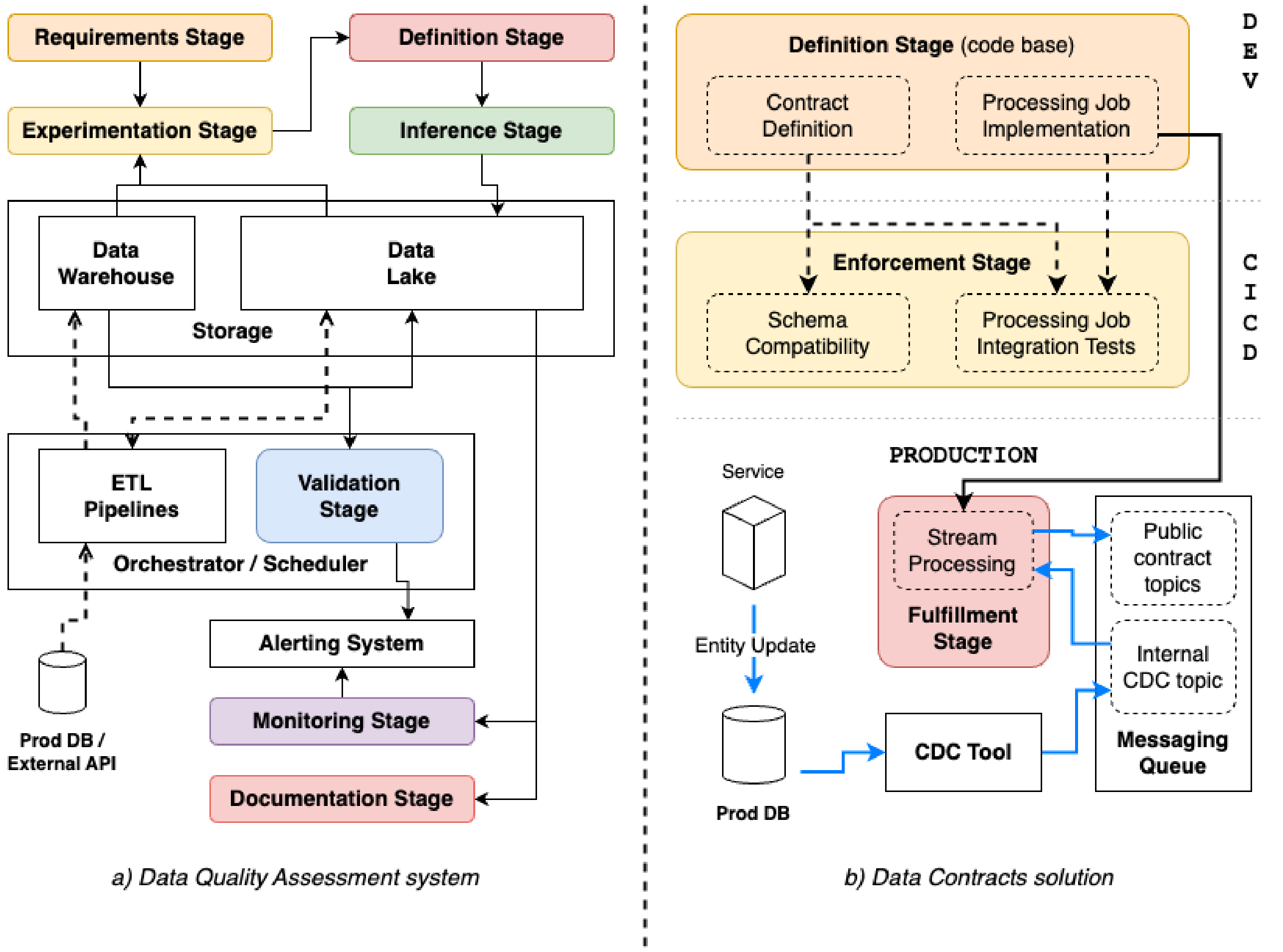

Figure 2: Architectural schema for two out of three components of the final framework: a/ the Data Quality Assessment system, and b/ the Data Contracts solution.

phase. In addition, an alert system, essential metrics, and dashboards automatically updated stakeholders on the data platform's health.

Architecture: The implemented DQA system architecture (depicted in Figure 2a) encapsulated seven distinctive stages. After the initial four—i.e., requirements collection, expectations experimentation, contract definition, and artifacts inference with the consequent loading into the storage layer—the process moved forward with the system's core: the *validation stage*. This pivotal stage was deeply embedded in the ETL system and thus entirely governed by an orchestrator. Its purpose consisted in checking that the considered data asset met the expectations specified in the contract. In case of eventual data inconsistencies, it alerted the data custodians so they could react based on the relevance of the discovered issue. Moreover, it enabled effective monitoring and documentation of the checks performed on the different data assets. This way, both technical and non-technical profiles could always access an overview of the DQ health status of the whole data platform.

Results: The immediate benefit was quickly detecting data issues immediately after deploying a data pipeline. The DQA Solution flagged inconsistencies right after an ETL process was productionized, thus eliminating scenarios where faulty data remained undetected for months.

6.2. Case Study 2: Large Corporation with multiple departments

Context: With multiple departments generating various data products, this corporation struggled with data inconsistencies and quality breakdowns originating upstream and breaking downstream pipelines.

Technology: The corporation adopted Data Contracts (DC) to rectify this. These are formal agreements, in the form of code, between data-producing and data-consuming departments that outline schema and data type specifications. A real-time streaming processing job then acted as an abstraction layer, transforming incoming data events to comply with these contract specifications [7].

Architecture: The DC infrastructure (illustrated in Figure 2b) comprehended four distinct stages. The initial stage—the *definition stage*—involved collaboration between data producers and consumers to establish a data contract and develop the streaming processing job that fulfilled the same contract. Afterward, the *enforcement stage* validated that the streaming processing job aligned with the agreed-upon contract. It also ensured compatibility with previous versions of the contract schema. These checks were integrated into the CI/CD pipeline, and a failure in either of the two will prevented code deployment, thereby safeguarding downstream jobs and pipelines. The *fulfillment stage* was a purely automated operational phase in which the production service generated events that were captured by the Change Data Capture (CDC) solution and finally transformed by the previously implemented processing job to meet the contract specifications. As a result, the output inherently conformed to the established format and schema, allowing downstream consumers always to receive consistent data.

Results: The abstraction layer represented by the streaming processing job effectively decoupled systems [4] and safeguarded downstream pipelines from upstream changes, thus achieving a harmonized data flow across departments.

6.3. Case Study 3: Cross-Enterprise data sharing collaboration

Context: Faced with the challenge of aggregating data from multiple external enterprises, the central organization consulted with the Politecnico di Milano to create a standardized solution.

Technology: A Push API (PAPI) provides a possible answer. The central entity should implement a RESTful API that external data providers can use to submit their data; this gives the central entity the power to dictate the terms for accepting incoming data, thereby ensuring its quality.

Possible Results: The system is supposed to act as a data quality gateway, screening all incoming data against pre-defined requirements. With the inherent scalability of the solution, the system could potentially integrate an unlimited number of data sources with minimal maintenance.

7. Limitations and Future Work

This section elaborates on the limitations inherent to the proposed framework and its components and outlines possible avenues for future work.

7.1. Individual Components Limitations

Data Quality Assessment (DQA) solution: While it offers a robust mechanism for the real-time detection of data quality issues, it primarily serves as a diagnostic tool and does not encompass preventive measures. Organizations looking for an end-to-end solution that includes prevention may find the DQA system to be lacking in this aspect.

Data Contracts (DC): They offer a structured approach to managing data quality, but the effectiveness of this solution is conditional upon two main factors. First, the complexity of designing the right system and the effort required to implement the underlying infrastructure. Second, successful implementation often requires software engineers' willingness and active participation, particularly those responsible for producing the data. Therefore, an implementation might be challenging when engineers are reluctant to extend the existing infrastructure.

Push API (PAPI): It is designed to work optimally for only a niche of use cases. It requires the central entity to be in a prevailing position over the data sources; it should be able to force the sources to conform to the provided API. Additionally, the PAPI solution assumes a degree of uniformity in the data nature, structure, and format across different sources; this limits its applicability in scenarios involving heterogeneous data from multiple parties.

7.2. General Limitations

Scope of Framework: The framework currently caters to three distinct collaborative scenarios, thus leaving other possible contexts unaddressed. Organizations operating in scenarios not covered by these three solutions may not find the framework directly applicable. Including new technological solutions for the missing organizational contexts is the first dimension toward which future works should focus.

Multi-tenancy: The framework focuses on multi-source situations, leaving unexplored terrain in multi-tenant environments. This emphasis may curtail its relevance in scenarios characterized by complex N-to-M relationships between sources and consumers. Developing strategies for data quality maintenance in these settings could enhance its practicality and comprehensiveness. Extending the framework to encompass multi-tenant scenarios is another direction to extend the current work.

8. Conclusion

This executive summary outlines the principal contributions of the thesis *Ensuring High Data Quality Standards: A Framework for Single and Cross-Enterprise Platforms*, which proposes a proactive framework that tackles data quality from both preventive and diagnostic angles. Three real-world case studies corroborate the framework's effectiveness and adaptability. With its limitations acknowledged, the framework opens several pathways for future research, reinforcing its utility and relevance in the ever-evolving data quality landscape.

References

[1] Zhamak Dehghani. How to move beyond a monolithic data lake to a distributed data mesh. `https://martinfowler.com/articles/data-monolith-to-mesh.html`, May 2019.

[2] Andy Petrella. Datastrophes: Il buono, il brutto, il cattivo. `https://andy-petrella.medium.com/datastrophes-il-buono-il-brutto-il-cattivo-c0bad03e8dcb`, Mar 2021.

[3] Joe Reis and Matt Housley. *Foundamentals of Data Engineering*. O'Reilly Media, Inc., June 2022.

[4] Chris Riccomini. Kafka change data capture breaks database encapsulation. `https://cnr.sh/essays/kafka-change-data-capture-breaks-database-encapsulation`, Nov 2018.

[5] Chad Sanderson. The rise of data contracts. `https://dataproducts.substack.com/p/the-rise-of-data-contracts`, August 2022.

[6] Chad Sanderson and Daniel Dicker. Data contracts for the warehouse. `https://dataproducts.substack.com/p/data-contracts-for-the-warehouse`, Jan 2023.

[7] Chad Sanderson and Adrian Kreuziger. An engineer's guide to data contracts - pt. 1. `https://dataproducts.substack.com/p/an-engineers-guide-to-data-contracts`, Oct 2022.