Anastasiia Voropaeva

Developing a Framework for Enhanced Data Pipeline Quality Management System

Metropolia University of Applied Sciences

Master's Degree

Degree Programme in Business Informatics

Master's Thesis

June 1, 2022

Abstract

Author:	Anastasiia Voropaeva
Title:	Developing a Framework for Enhanced Data Pipeline Quality Management System
Number of Pages:	61 pages
Date:	June 1, 2022
Degree:	Master of Business Administration
Degree Programme:	Business Informatics
Instructor:	Kari Kalliojärvi, Service Manager Antti Hovi, Senior Lecturer

This thesis focuses on the quality of a data pipeline. The thesis was performed in the company that offers the data pipeline as part of a product for developing autonomous solutions for mobile radio networks. Data pipeline aims at delivering data from various sources to a destination in a form optimized for reading, querying, decision making and building data products.

Rationality of decisions and quality of data products are defined by the quality of the data pipeline. Quality monitoring system, currently used in case company, provides means for ensuring quality of the data pipeline and data products. However, with development of the product and gaining more experience from implementing the data pipeline into various environments, it was revealed that the system allows some flaws to pass unnoticed. This thesis aimed at highlighting insufficiently observed stages of the data pipeline and proposed key components of quality management system to implement.

The study was performed using action research methodology and represents applied research, conducted based on qualitative data collection and analysis. The theoretical framework discussed the main components of data pipeline quality management system, such as notifications rules, quality statements, metadata storage, audit storage, incident management system, and quality monitoring system.

The thesis focused on observability problem of a data handling process and proposed the main components that are required for troubleshooting, monitoring, and understanding the health of the data and the pipeline itself. Based on the best practices studied in the thesis, recommended components are inevitable part of trustworthy data pipeline solution of scale. Addressing observability problem and improving means for data pipeline quality monitoring will allow to ensure accuracy of data products and provide solid ground for decision making.

Keywords: Data Warehouse, Data Pipeline, Quality Management System, Metadata, DW, Data Quality, ETL, ELT

Contents

List of Figures

1 Introduction 4

1.1 Business Context 6
1.2 Business Challenge, Objective, and Outcome 7
1.3 Thesis Outline 9

2 Method and Material 11

2.1 Research Approach 11
2.2 Research Design 12
2.3 Data Collection and Analysis 14

3 Existing Knowledge on Data Quality Management in Relation to Automated Data Engineering Pipeline 17

3.1 Data Pipeline 17
3.1.1 ETL and ELT Processes 17
3.1.2 Data Warehouse 18
3.1.3 Data Vault 20
3.2 Data Pipeline Development 22
3.2.1 Data Pipeline Observability and Metadata 23
3.2.2 Quality Tests and Data Profiling 24
3.2.3 Quality Tests Process flow 26
3.2.4 Incident Management 28
3.3 Conceptual Framework 29

4 Current State Analysis of the Data Pipeline Quality Management System at the Case Company 32

4.1 Overview of the Current State Analysis 32
4.2 Case Company 32
4.3 Description of the Data Flow as Part of the Data Pipeline 33
4.3.1 Data types 35
4.3.2 Data structures 36
4.4 Strength and weaknesses of currently used Data Pipeline Quality Management System 36

4.5 Current State Analysis summary 40

5 Development Proposal to Enhance Data Pipeline Quality Management System 42

5.1 Overview of the Proposal Building Stage 42
5.2 Initial Proposal 42
5.2.1 Proposal Element 1: New role within the team 43
5.2.2 Proposal Element 2: Maintaining Documentation 44
5.2.3 Proposal Element 3: Enriched Data Profiling as an Application 45
5.2.4 Proposal Element 4: Metadata and Monitoring 45
5.2.5 Proposal Element 5: Error Event Actions and Alarms 47
5.3 Summary of the Proposal 49
5.4 Findings from Data 2 Collection 50

6 Validation of the Proposal 53

6.1 Overview of Validation Stage 53
6.2 Developments to the Proposal (based on Data Collection 3) 53
6.3 Final Proposal 56

7 Conclusion 58

7.1 Executive Summary 58
7.2 Evaluation of the Thesis 59
7.2.1 Final Words 60

References 1

List of Figures

Figure 1. Simplified autonomous network structure. 5
Figure 2. Data engineering pipeline. 5
Figure 3. Example use case of a data engineering pipeline (Chandrasekar, 2021). ... 6
Figure 4. Action research (Coughlan & Coghlan, 2002). 12
Figure 5. Research design of the thesis. 13
Figure 6. The data warehouse (Kimball & Caserta, What the Data Warehouse is, 2004). 19
Figure 7. Information collection on different data processing steps (Kimball & Caserta, What the Data Warehouse is, 2004). 20
Figure 8. Difference between data warehouse and data vault (CloverDX, 2017). 21
Figure 9. Data pipeline quality components (Bergh, Benghiat, & Strod, 2019). 22
Figure 10. Metadata of various types on different stages of data handling process (Kimball & Caserta, Metadata, 2004) 24
Figure 11. Types of data profiling (Oracle, 2009). 25
Figure 12. Data profiling (Oracle, 2009). 26
Figure 13. Check process flow (Kimball & Caserta, Requirements, Realities and Architecture., 2004). 27
Figure 14. Conceptual Framework of data flow (as a basis for data quality management of the data engineering pipeline). 30
Figure 15. Case company´s offering. 33
Figure 16. Timeline of imports. 34
Figure 17. Components of data pipeline quality management system. 37
Figure 18. Example of entity relationship diagram. 44
Figure 19. Example of data pipeline quality monitoring. 46
Figure 20. Components of data pipeline quality management system addressed in the Proposal. 49
Figure 21. Proposal draft. 50
Figure 22. Final proposal. 56

1 Introduction

Increasing smartphone usage and the widespread availability of wireless mobile networks have supported data traffic growth as consumers have become more engaged with various mobile services such as e-commerce, social networks, entertainment platforms, etc. Traffic growth will continue as modern technologies and services such as solutions and applications for smart homes, cities and buildings are being introduced (GSM Association, 2021).

The telecom networks' expansion and constant technologies development has led to more dense and complex wireless networks which are increasingly difficult and expensive to maintain. Despite constant increase in the number of mobile subscribers, the growth has been slowing down. Revenue and cash flows of telecom companies has dropped by an average of 6% since 2010 (McKinsey, 2017).

Many of processes required to maintain and manage wireless networks are laborious, time consuming and complex, and therefore costly. Quite often new processes and network operations are more complicated and change more often than a human or manual process can cope with.

However, Telecom operators can achieve networks' operational cost savings by transforming their businesses and by managing wireless networks with next-generation technologies. According to (TMForum, 2019) and (Deloitte, 2020), networks operation automation needs to be a main component in mobile operators' transformation. Autonomous networks can take an advantage of Artificial Intelligence (AI), Big Data analytics and provide fully automated, self-optimizing capabilities to manage wireless network operations and maintain good subscribers' experience. Figure 1 shows simplified logic of autonomous radio networks.

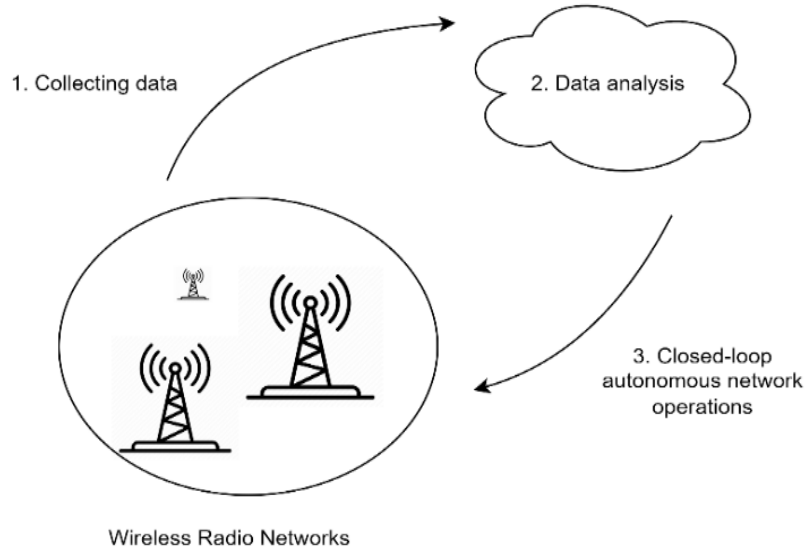

Figure 1. Simplified autonomous network structure.

Analytics and automation require collecting and processing data from various source and implies analyzing the data, visualizing insights, and developing applications (Figure 1). It all comes down to building a data engineering pipeline that extracts, transforms, structures, and stores the data, which can later be accessed for analysis, reporting, applying algorithms or application development purposes (Figure 2). According to (N-iX, 2019), process of extracting, transforming, and loading (ETL process) the data accounts for 80% of any analytics project based on Big Data.

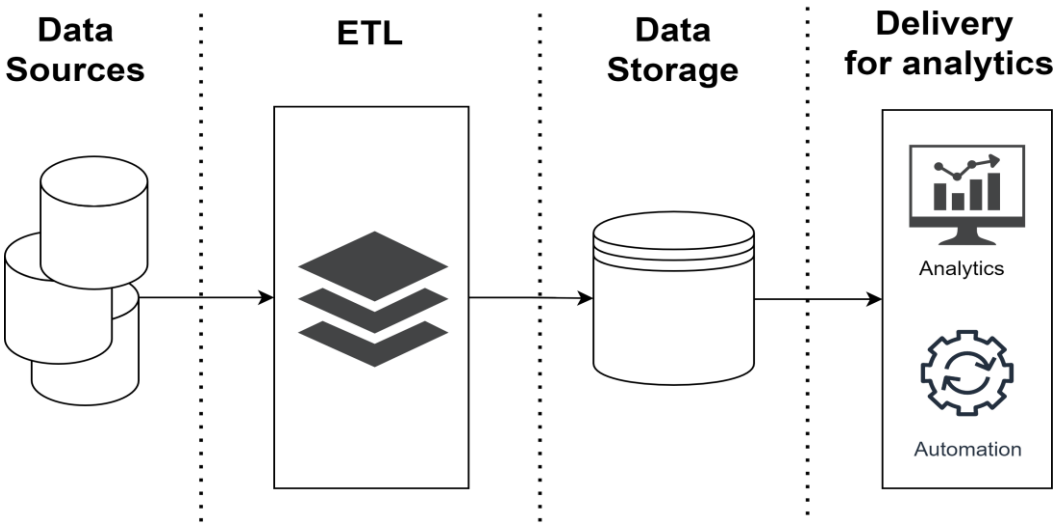

Figure 2. Data engineering pipeline.

Thus, high-quality data is a persistent challenge that defines the usability of a data pipeline, reliability of analysis and accuracy of intelligent applications built on top of it. This thesis is devoted to the analysis and enhancing data pipeline quality management system currently used by the case company in order to improve observability and reliability of the data processing.

1.1 Business Context

The case company of this thesis provides telecommunication operators with off-the-shelf automated solutions for managing, operating, optimizing, and analyzing mobile radio networks. The company offers automated data engineering pipeline to collect data from Radio Access Networks (RAN), Operations Support System (OSS) and Business Support System (BSS), that can be further analyzed and used for closed-loop automatization (without involving a human). Besides, the company offers Software Development Kit (SDK) and capabilities for non-real time (non-RT) applications development.

The case company along with other organizations employs a commonly used big data pipeline that consists of various components from retrieving raw data to supplying refined information to analytics systems. Figure 2 represents overall view and Figure 3 shows an example use case of such data pipeline (Chandrasekar, 2021).

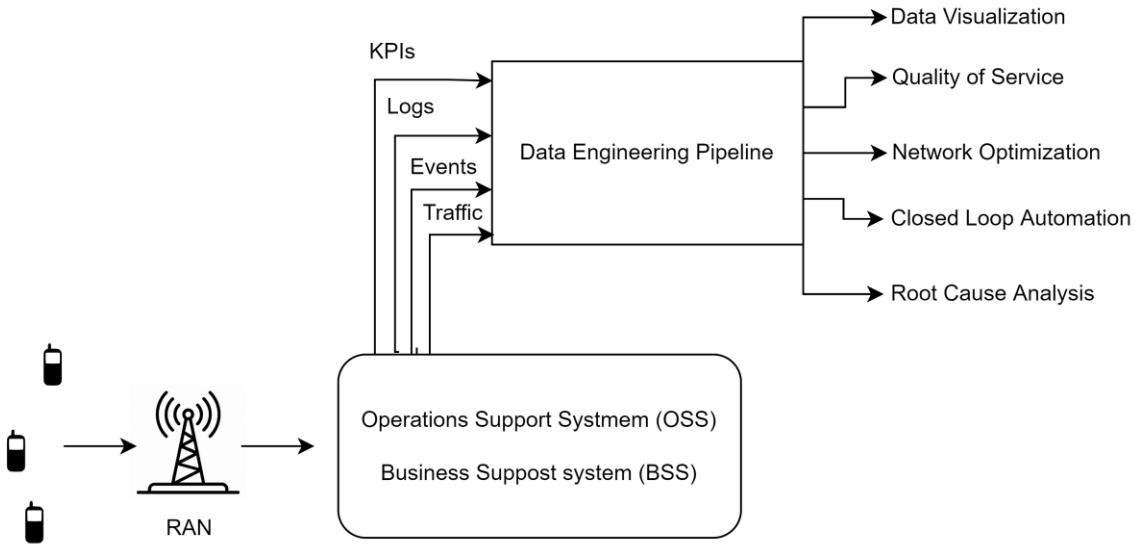

Figure 3. Example use case of a data engineering pipeline (Chandrasekar, 2021).

1.2 Business Challenge, Objective, and Outcome

Big data pipelines have become more and more complex. Source data is ingested, transformed, stored, processed, and delivered with many Application Programming Interfaces (APIs) and integrations between different elements of the pipeline. Multiple stages of data processing and non-trivial dependencies between various data assets have been introduced because of more advanced and distinct tooling. Any change made to data set or to software processing the data might cause unintended impact on the correctness of data assets without being noticed. According to (Moses, Gavish, & Vorweck, 2022) if the data in production cannot be trusted, data products such as Business Intelligence (BI) analytics and various applications build on top of it cannot be trusted as well. By data in production, they imply data that has been ingested by a warehouse or any other data storage, processing solution and delivered to the users.

Correctness of collected data is essential for business of the case company. Using incorrect data when performing analysis of a radio network will lead to wrong conclusions. If recommendations based on inaccurate data are to be implemented, it may cause degradation of the mobile network users´ experience, which is critical for a telecom operator. The data is being injected, collected, and processed to the platform automatically. Therefore, the quality of big data process should be controlled to ensure high quality of the analysis and subsequent recommendation actions.

It should be highlighted here, that in addition to the accuracy of collected and processed data, the quality of recommended actions that are proposed to be implemented to a network is of critical importance as well. The algorithms are automatic and do not imply involving a human. In other words, after performing an analysis an algorithm creates a list of actions and modifications that are to be automatically deployed to a functioning radio network. Any bizarre and unjustified actions, or any anomalies in the recommendations due to, for example, a bug in the algorithm may cause noticeable quality degradation of a radio network and consequently user experience (drop calls, low throughput, etc.). Therefore, it is crucial to ensure that the actions are sensible and justified.

The objective of this thesis is *to propose enhanced data quality management system to increase the reliability of automated data engineering pipeline provided by the case*

company. Enhancing of the system in the context of this thesis implies incorporating additional monitoring, recording, and alerting capabilities to the data pipeline and automation of this process. This will allow to ensure accuracy of business intelligence analysis and allow more trustworthy applications' development both by analysts and developers from the case company and by customers who purchased the product.

To meet the objective stated above, the following actions will need to be performed:

- to classify errors and problems that cause data to be incorrect
- to identify checks that could help discover the flaws
- to detect the best place in the product data pipeline for checking and preventing error/problems
- to perform gap analysis between current checks and needed checks
- to propose methods for the detecting needed checks
- to develop the logic for reporting results of the checks
- to develop the logic for reacting upon detected issues.

The outcome of the thesis is *a framework of enhanced data quality management system of the case company*. The framework will discuss the main components for product data pipeline to capture and to communicate insights of data quality to the stakeholders and decision-makers. The framework will propose means of evaluating the accuracy and completeness of the information available in data warehouse and stability of data products (such as analytical reports) built on top of that data. In other words, the outcome will include:

- methods for quality assurance in data engineering pipeline (various checks, comparisons, etc.)
- recommendations for enhancing data quality monitoring capabilities (e.g., dashboards, warning messages, etc.)
- a proposed incident management process (send an alert email to a responsible person, cancel execution of an application, etc.).

1.3 Thesis Outline

The study conducted within the scope of this thesis focuses on the data quality in a data pipeline used in the case company's business. Data pipeline implies a set of tools and processes for transferring the data from one system to another for storage and further handling (Altexsoft, 2021). The thesis describes the main steps of the data pipeline used in the business. Besides, it examines and captures possible flaws of the pipeline and ways to proactively react and mitigate harmful effects, should any issues occur.

The final step of a data pipeline is loading the data to a new destination, which is represented by a data warehouse in the current thesis. According to (Altexsoft, 2021), the data warehouse implies a repository where large volumes of data are stored in a way optimized for reading, accessing (querying) and aggregating. The ultimate purpose of the study is to ensure that the data stored in a data warehouse is accurate, complete, and consistent to guarantee reliable data analysis and trustworthy data products built on top of collected data.

Concepts, processes, and best practices analyzed in this thesis are applicable for any data pipeline and data warehouse. Since the case company's product studied in the thesis is focused on RAN area of mobile operators, general concepts explored in the work might need to be adjusted in accordance with the peculiarities of this field.

The study involved a comprehensive analysis of the product architecture conducted with the support of product developers, architect, and other stakeholders. To capture already known flaws and discover new potential flaws of the data pipeline, interviews and workshops among product developers, analysts, and application developers were conducted. A list of actions was proposed for all identified issues. In the context of development work, an action implies a feature that needs to be implemented into the product, or an additional monitoring dashboard to be build and applied to a particular step of the data pipeline, or an additional check that needs to be incorporated into an application to ensure consistency of its results, etc. All these actions were assigned to a responsible person for further design or implementation. The development work needed within the scope of the current challenge was conducted and is currently being conducted based on Continuous Integration and Continuous Delivery (CI/CD) practices, which aims at minimizing the gap between software development and operation activities and

enforcing the automation of repetitive tasks. In other words, identified flaws and ways to tackle them are not final and will be continuously tuned, tested, complemented, automated, and monitored throughout the development life cycle of the product.

This thesis is divided into 7 sections. Section 1 describes the problem explored in the thesis. It describes why the large volumes data became a question of significant importance for telecom business. Besides, it also discusses the vital importance of quality of collected data for building data products and for automation. Section 2 is devoted to methodologies used for analysis of the challenge and the approach to solve the challenge. Section 3 explores the existing knowledge on data pipelines and data warehouses, the ways to ensure the quality of both within agile software development cycle and CI/CD practices. It also explores recommendations on building data quality management processes. Section 3 ends with the conceptual framework based on which the case company's challenge is later analyzed in more depth. Section 4 reports on the results from the Current State Analysis (CSA) of data quality management process that is currently being used in the case company. This section summarizes all previously faced, newly identified, and foreseen issues with the data pipeline, the ways to monitor them and react to them. Section 5 develops the proposal to tackle the challenge of the case company. This section discusses the best points in data pipeline to proactively identify possible problems, measure and react to them. This section also describes the main stakeholders and their roles and level of involvement into solving the challenge. Section 6 reports on the validation of the proposal and discusses the changes which should be incorporated into the proposal to solve the original problem. This section also provides practical recommendations for proposal implementation. Section 7 evaluates the study and obtained results. This section is intended for critical and grounded look at the proposal and its relevancy.

2 Method and Material

This section describes the research approach applied for discovering, investigating, and addressing the challenge. Collaboration work and the way to collect the data are described in this section as well.

2.1 Research Approach

From the application point of view, two types of research are distinguished: basic research and applied research. Applied research conducted in this thesis aims at solving an immediate problem faced by an organization (the case company) in contrast to a basic (fundamental) research which aims at creating new knowledge or expanding existing knowledge without focusing on its applicability (Kothari, 2004).

From the type of data collected and analyzed in research, three types of research can be recognized: qualitative, quantitative, and mixed. In this thesis, the need faced by the case company related to the current process of data handling and focused on developing a solution that would enhance the quality assurance capabilities of the product. The nature of such a research process implied a holistic view at the entire data pipeline and collecting interview data from stakeholders in a form of semi-structured and non-structured interviews. Thus, the reasoning applied to determine the meaning of the data was primarily inductive (from practical little issues that needed to be addressed to a general solution). According to (Leedy & Ormrod, 2015), all the above are the characteristics of *qualitative* research approach.

The research conducted in the thesis is performed within the scope of the company's in-house project. For achieving the thesis objective, a series of interviews and workshops with stakeholders were conducted. Some of the actions needed to resolve the challenge were determined and unanimously agreed on during the first few workshops. For implementation of these actions, agile software development approach was used, which implies delivering updates to the product in small consumable increments. Requirements, planned actions, and results were evaluated continuously. Besides, the case company applied CI/CD practices to integration and delivering the software. CD part aims at building, configuring, and deploying the software in automated manner (low

cost, high automation (Synopsys, 2018)). This is relevant for the type of the proposal that was developed in this thesis work, because proposed data pipeline quality management system should be automated and should aim at minimizing human involvement. Thus, Action research (AR) methodology applied was considered as appropriate methodology for the study that implied a continuous cycle of analysis, planning, implementation, and evaluation (see Figure 4).

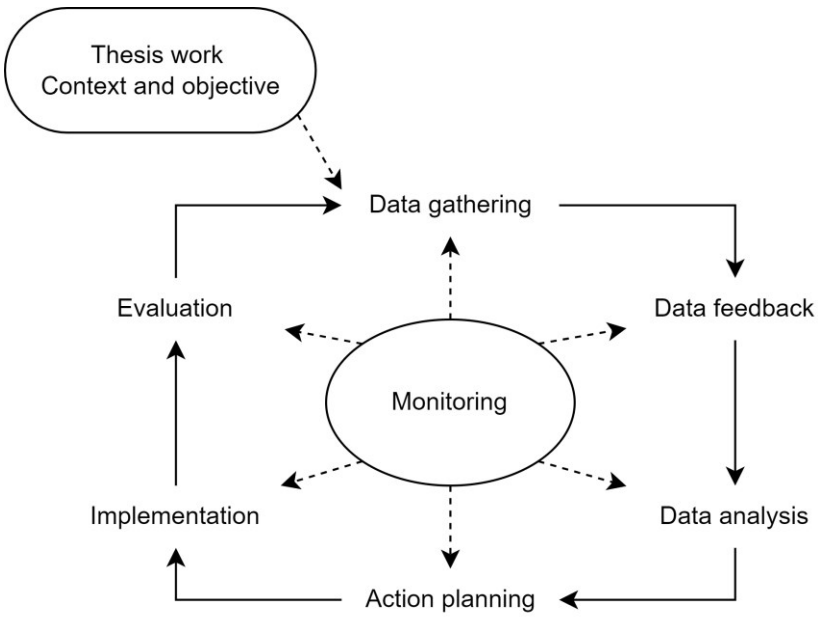

Figure 4. Action research (Coughlan & Coghlan, 2002).

2.2 Research Design

This study was conducted according to the research design presented on Figure 5. The output of each of the phase is highlighted with green color inputs are highlighted with blue color.

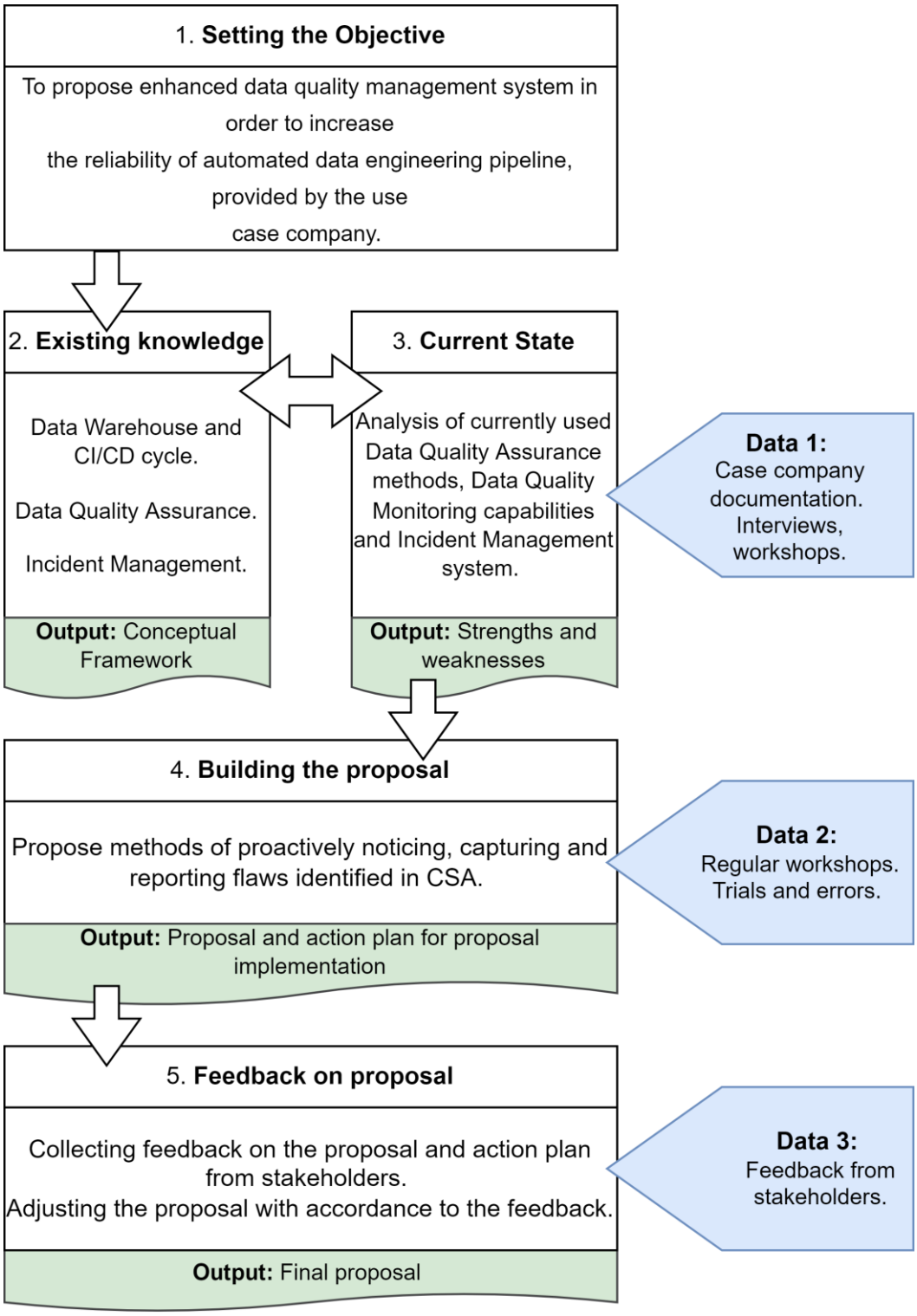

Figure 5. Research design of the thesis.

The objective of the thesis was set clearly at the very beginning. The case company needed to enhance reliability of the data engineering pipeline. In the course of the thesis,

regular workshops and knowledge exchange were conducted between the stakeholders (Data 1 from Figure 5) in order to determine stages of the data pipeline, where flaws have been passing unnoticeably. These flaws were discussed and documented. For every flaw a corresponding stage of the data pipeline where a flaw can be proactively noticed was identified. The next step was to study the best practices, to structure all the findings according to knowledge existing in the industry and to create a framework that would define an overall direction to move to, when implementing means to proactively notice and react to the flaws in the data pipeline.

To perform the current state analysis, extensive study of existing literature, best practices and company's documentation was conducted. A data pipeline and its quality management system are well-known topics. There are comprehensive recommendations available for designing it. However, there are a lot of nuances related to implementation, designing, and maintaining such a system. That is why this subject needed to be carefully explored before putting into practice and adopting to the case company's environment.

After conducting the current state analysis and structuring the findings, proposals were built. Some of the proposals (to be more precise, components of the proposals) were possible to implement immediately. Some of the components were tested in order to assess whether the solution is usable in practice or not. Continuous follow up on the implementations, changes in the approaches and newly emerging solutions were conducted on regular basis (Data 2 from Figure 5). With the conclusion of the thesis work, the stakeholders identified and agreed on the areas to improve and ways to improve them. As the final proposal, the framework of the enhanced data pipeline quality management system was suggested in order to align different stakeholders with final goal to aim to (Data 3 from Figure 5).

2.3 Data Collection and Analysis

The study was performed based on interviews, workshops, and group meetings, conducted with the stakeholders from different departments (architects, software developers, algorithm developers, telecom experts).

Along with the data mentioned above, the internal documentation was used to conduct the research and to prepare the proposal: 1. Product architecture, 2. Data Pipeline, 3. Software/Technology stack used. Architecture of the product and data pipeline description allowed to understand the main components of the product in detail and to link it to the data transformations. Information about utilized software was used to understand possibilities for functionality expansions (e.g., creating additional monitoring dashboards and queries).

Table 1. Workshops and interviews used for data collections 1-3 in this thesis.

Data type	Participants	Topic	Date	Outcome
Data 1				
Interviews	Product Owner, Software Developers, Algorithm developers	Discussions on the need to enhance quality assurance process of the product data pipeline, examples of faced problems.	09-12.2021	List of faced issues. Internal documentation.
Workshop	Product Owner, Architect, Software Developers, Algorithm developers	Generalize known issues, discuss possible unknown issues. Agree on the way to proceed forward.	02.2022	Current state analysis. Refine list of stakeholders and their responsibilities. Document and classify findings and needs. Collect open questions. Close resolved issues.
Workshop	Product Owner, Architect, Software Developers, Algorithm developers	Categorize the findings and needs. Plan how to incorporate them into data pipeline and product architecture.	02.2022	Data pipeline and product architecture with proposed additions. Create list of features that are decided to be implemented. Assign responsibilities on implementations. Collect open questions. Close resolved issues.
Workshop	Product Owner, Architect, Software Developers, Algorithm developers	Further discuss findings and needs. Assess newly implemented features. Refine requirements to new features.	03.2022	Additional features needed. Collect open questions. Close resolved issues. Generalize all features needed.
Data 2				
Workshop	Product Owner, Architect, Software Developers, Algorithm developers	Status check. Feedback.	04.2022	Structure all the features according to the best practices. Propose the main components of the framework and its building blocks. Collect feedback on the framework.
Data 3				
Workshop	Product Owner, Architect, Software Developers, Algorithm developers	Close the project. Plans for further improvements. Stack of technologies/software required for further improvement.	05.2022	Collect open questions. Close resolved issues. Refine framework.

As seen from Table 1, the data collection can be divided into three rounds. The first round (Data 1) was devoted to acknowledging and listing known issues and known flaws of the data pipeline. The issues were structured and generalized based on the best practices. The list of stakeholders and their responsibilities were identified during that round as well. As a result, application developers and data pipeline developers were assigned with features and applications to develop and implement within the scope of quality enhancing project. This round implies trials and errors and refining the approaches to tackle identified issues. During the second round (Data 2), newly implemented features and applications were assessed in terms of whether they help in solving the challenge and whether they are usable on a big scale. Also, these features were categorized. Based on the identified categories and the best practices, the main components of the framework were propose. The third round (Data 3) aimed at concluding the list of features to implement and items to further study in foreseeable future. The building blocks of the framework for the enhanced data pipeline quality management system were refined and concluded.

As it was mentioned above, a detailed study of existing knowledge and best practices was conducted in order to structure topics for workshops, categorize the features to implement and assign them to corresponding stages of the data pipeline. Section 3 explains the terminology used in the thesis, data pipeline structure and other elements of the data pipeline quality management system in more detail.

3 Existing Knowledge on Data Quality Management in Relation to Automated Data Engineering Pipeline

This section discusses the main concepts, established processes and best practices which seems to be useful for understanding and solving the case company's challenge. As discussed in Section 1.2, the challenge is related to observing health of the data pipeline in order to ensure quality of data products and support decision making.

3.1 Data Pipeline

According to (Altexsoft, 2021), a data pipeline is a collection of various tools and processes for moving data from original systems to a destination system for further handling and quick access to combined data. Constructing data pipelines implies designing a software for continuous and automated dataflow (a process of moving data from origin to destination and all the changes the data undergoes along the way).

3.1.1 ETL and ELT Processes

One of the approaches to dataflow is called ETL, i.e., Extract, Transform, Load. The conventional ETL process is intended for combining the data from multiple data sources into one consistent format and includes the three corresponding functions: extract, transform, load. According to (Deloitte, 2019) and (Densmore, 2021), the performance of ETL process starts deteriorating as the amount of data grows over time because of the following reasons: ETL process focuses on relational processing and follows Schema-on-Write approach. Schema-on-Write approach implies that ETL process should start with a reference data set or a schema, that will define acceptable values for the data and the way the information from different sources is related. Then, the data is being transformed and loaded to the destination system, from where the data can be read and viewed in the shape defined by the schema predefined in the first step. The format should be known before starting ETL process. In other words, all the data that does not fit into predefined schemas is eliminated and is not being loaded to the storage.

The alternative to Schema-on-Write approach is Schema-on-Read approach, which implies extracting and storing the data in its original format instead of using predefined schemas. When there is a need to read and use the data, new schemas can be created

to view the data in the accordance with them. In other words, it does not require upfront planning for data extraction. This process is referred as ELT, i.e. Extract, Load, Transform, and is considered to be a way forward for cloud-based data warehouses and data lake infrastructures that supports large storage and scalable computation, according to (Deloitte, 2019), and (IBM, 2021).

Once the data is transformed and loaded into a destination, it can be easily accessed and used for further analysis, reporting, and developing other data products.

3.1.2 Data Warehouse

According to (Altexsoft, 2021) and (Eckerson Group, 2018), data warehouse is a central repository, where data is stored in a format optimized for reading, aggregating, and querying and represents a destination of ETL process.

In contrast, Ralph Kimball (an original architect of data warehousing) defines a data warehouse as the process of moving data from source to target destination and transforming and organizing in in user-friendly format (see Figure 6). In Section 3.1, this process was defined as a data pipeline. Quoting (Kimball & Caserta, What the Data Warehouse is, 2004):

> A data warehouse is a system that extracts, cleans, conforms, and delivers source data into a dimensional data store and then supports and implements querying and analysis for the purpose of decision making.

Figure 6 below shows the process of moving data from source to target destination, transforming and organizing in in user-friendly format (a data pipeline in the data warehouse).

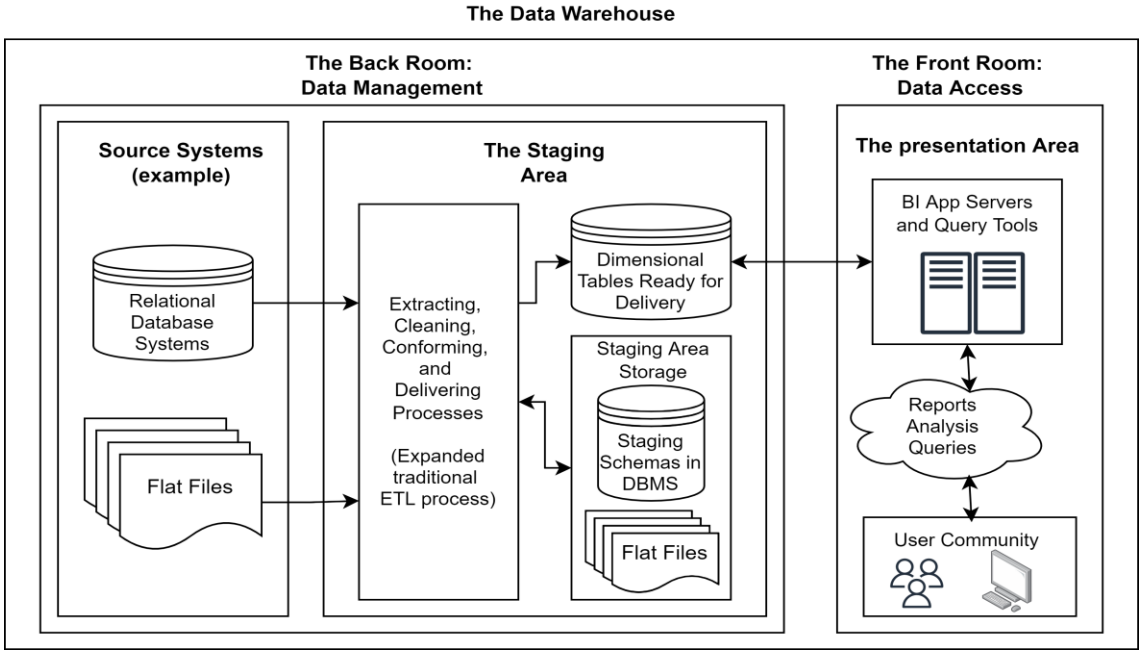

Figure 6. The data warehouse (Kimball & Caserta, What the Data Warehouse is, 2004).

(Kimball & Caserta, What the Data Warehouse is, 2004) details ETL process a bit by splitting transform stage of the ETL process to cleaning and conforming steps:

1. During extracting step, the raw data obtained from various sources (such as databases, XML data sets, txt files, etc.) is written to the data storage with minimal restructuring. Later, this data can be read and transformed multiple times on the need base.
2. During cleaning step, data loaded from the source systems is checked, for instance, for valid values, consistency, and presence of duplicates. Data cleaning step is often irreversible.
3. During conformation step, data from different source is merged according to predefined dependencies.
4. Delivering step is the final stage of ELT process and is intended to make the data ready for querying. Here, the data is being structured according to dimensional models, which simplify application development and reduce querying time.

As seen from Figure 6 above, data warehouse is split is two parts: back room and front room. Usually, the back and front rooms are logically and administratively separated.

According to (Kimball & Caserta, What the Data Warehouse is, 2004), the back room focuses on data management (acquiring, transforming, storing) and delivering data to the front room, where the data can be accessed by a user. The back room is intended for accessing by experienced data integration professional only.

Staging area of the back room is implied for acquiring snapshots of the data on different stages of data processing and/or information about a current step of data handling process. This information can be stored data for further investigation (e.g., for data quality checks or impact analysis) in parallel with transferring the data to next steps (see Figure 7).

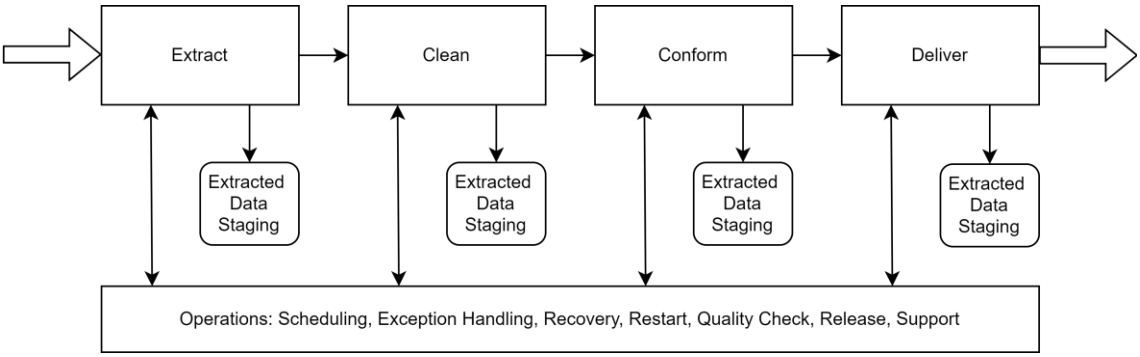

Figure 7. Information collection on different data processing steps (Kimball & Caserta, What the Data Warehouse is, 2004).

As shown on Figure 7 above, information collection is to be performed on all stages of the data processing: extraction, cleaning, conforming, and delivering. Collected information can help to understand how each step of the data processing behaves. For example, how long execution of every step takes, what types of output in produces, etc.

A data warehouse (following Kimball's definition) or a data pipeline can be referred as a data handling process. Each time when there is a new dataset to handle, the entire process should start again.

3.1.3 Data Vault

Storage of a conventional data warehouse contains records of only one set of data. Besides, data that is not needed is viewed as noise and is not being stored in the storage at all. As discussed above, ETL implies eliminating the data that does not fit into

predefined schemas. However, deleted data might be needed in the future. In contrast, Data Vault keeps historical data (old data, new data) and moves transformation stage to data marts (a subset of the data from the data warehouse). According to (Linstedit & Olschimke, 2015), conventional data warehouses store “a single version of truth” of business data, whereas data vaults’ goal is to “provide all the data, all the time” or “single version of facts”. Difference between a data warehouse and a data vault is presented in (CloverDX, 2017) and in slightly modified form on the picture below (Figure 8).

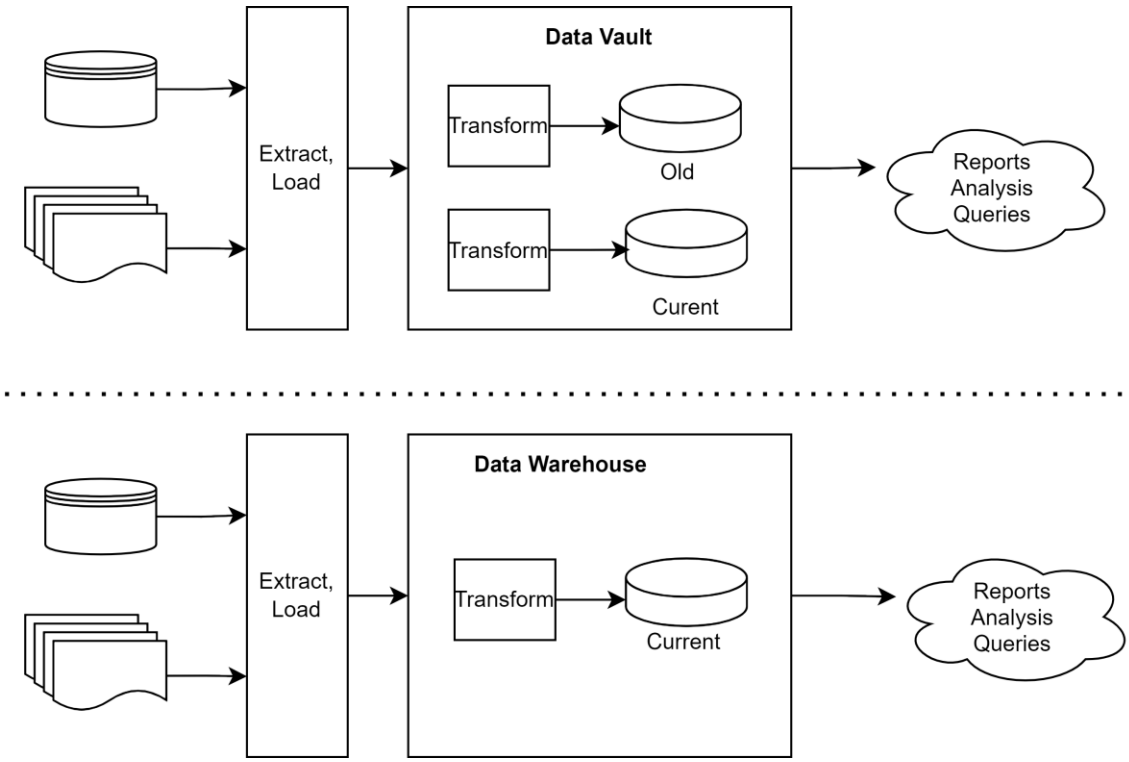

Figure 8. Difference between data warehouse and data vault (CloverDX, 2017).

Figure 8 shows the ELT process (Extract, Load, Transform). As seen from the figure, data vault contains few sets of data: previously loaded and transformed and currently (or recently) loaded and transformed. A data warehouse stores only one version of the data and there is no knowledge on the previously loaded data sets.

Further on, due to various definitions available and for simplicity, it should be stated here that this thesis will be referring to a data pipeline as data handling process and to a data warehouse as a storage (a destination of the data pipeline). With time, data sets change, new sources are being added for transformation, etc. Besides that, the entire data

pipeline may change. For example, new rules on data cleaning or new schemas for data transformation can be introduced.

3.2 Data Pipeline Development

Data handling process have become increasingly complex due to multiple steps of processing and complex dependencies between different data sources (Moses, Gavish, & Vorwerck, What Is Data Quality?, 2022). There are a lot of moving parts, which makes the development of a Data Pipeline complex (Loome, n.d.), for example:

- Sources of original data and its structure are usually not controlled and may change over time.
- Interfaces to raw data sources may change, which can make extraction not possible anymore.
- Deploying changes to ETL process over large data volumes can be problematic and requires a copy of production data for testing.

According to (Bergh, Benghiat, & Strod, 2019), data pipeline maintenance never ends: newly added or updated data source, data schema change, any application development may trigger an update to data warehouse. (Bergh, Benghiat, & Strod, 2019) defines quality as a function of data and code. In other words, quality of raw data and data handling processes are of vital importance (see Figure 9).

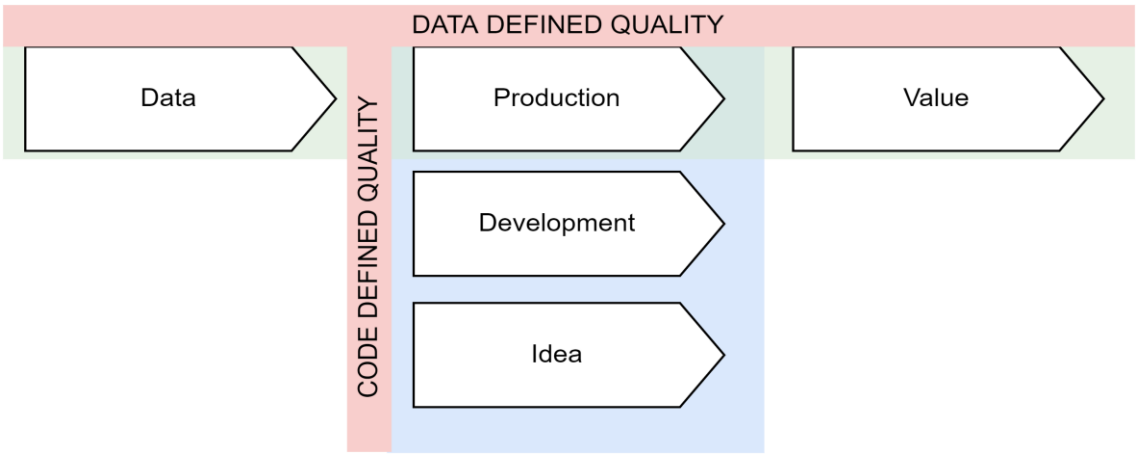

Figure 9. Data pipeline quality components (Bergh, Benghiat, & Strod, 2019).

The changes and the impact of the development must be carefully monitored, so that they do not break the operational analytics and applications. (Moses, Gavish, & Vorweck, Fixing Data Quality Issues at Scale, 2022) proposes to constantly observe the data to ensure its reliability at every stage of the data pipeline.

3.2.1 Data Pipeline Observability and Metadata

As discussed in Section 3.2, data pipeline is a complex combination of software and data. To understand state and health of the system (both data and the pipeline), it need to be monitored and observed. (Eckerson Group, 2018) highlights that observation is an extension of monitoring. Monitoring allows to discover whether system is working as expected, whereas observation is an umbrella term and covers the activities and tools to identify, troubleshoot, predict, and resolve the issues. According to (Databand, 2021) data observability should include following activities: monitoring, alerting, tracking, comparisons, analysis, logging, Service Level Agreement (SLA) tracking. All these activities imply collecting specific metadata (data about data) from different stages of data pipeline.

The data collected on different stages of data handling process is referred as metadata and is described as data that provides information about other data (Merriam-Webster, 2015). According to (Lu & Özsu, 2009), data warehouse metadata may include:

- Information on the content of data in the data warehouse
- Information on a process in the data warehouse (scheduling, loading, data handling process, etc.)
- Information on data schemas (semantics of the data)
- Information on the infrastructure and sources of the data
- Other information such as usage statistics, access rights, etc.

(Kimball & Caserta, Metadata, 2004) distinguishes three types of metadata: business, technical and process execution metadata (see Figure 10). Metadata collected from input and/or output of a stage in data pipeline should be stored in the staging area (see Figure 6) and reported to data warehouse developers to be able to control the operational aspects of the data warehouse. Besides, any metadata that can be useful to an end user

(application developers, analysts, etc.) should come from the back room and should be presented in the front room area of the data warehouse.

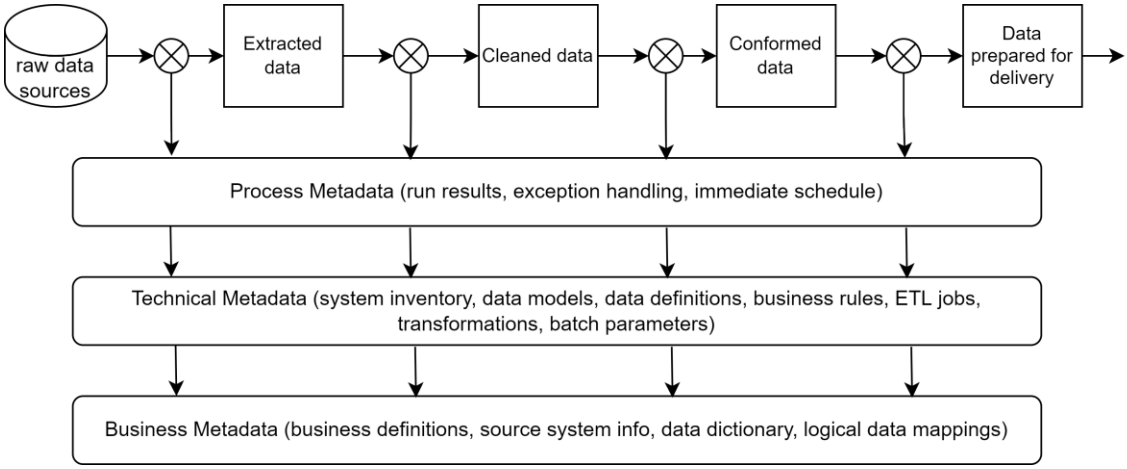

Figure 10. Metadata of various types on different stages of data handling process (Kimball & Caserta, Metadata, 2004)

Impact analysis is one of the advantages which is enabled by availability of metadata. Any changes that happen in data handling process or any other attributes of the data pipeline can be visible, controlled and measured using metadata. For example, results of data models' checks, transformations, etc. can be stored in a form of metadata (in particular, in technical metadata) for further analysis and overviewing the health of the data pipeline in general and execution of each stage of the data handling process. Thus, metadata storage is one of the components of data pipeline quality management system proposed in this thesis.

3.2.2 Quality Tests and Data Profiling

(Databand, 2021) highlights 3 different levels of data observability and, consequently, checks: 1. Operational health and dataset monitoring (high level, using process metadata); 2. Column-level profiling (using technical metadata); 3. Row-level validation (for example, using business metadata).

As discussed above, quality of data pipeline is defined by both code and data. When developing a data pipeline, quality tests need to be performed, which can target data or pipeline code. According to (Bergh, Benghiat, & Strod, 2019) code testing can imply: unit

tests, integration tests, functional tests, regression tests, performance tests, smoke test. This thesis is focused on data quality; therefore, code tests will not be discussed in any more detail. As for data tests, (Bergh, Benghiat, & Strod, 2019) distinguishes: location balance tests, historical balance, time balance (or statistical process control). Location balance test checks that the quantity of original data does not change after going through various stages of data pipeline. Historical balance test compares current data to previously obtained values. Statistical test ensures that data values are within acceptable statistical ranges (mean, variance, etc.).

(Oracle, 2009) refers to data checks as profiling of data stored in a data warehouse (as discussed before, some of the experts consider data warehouse to be a structural or semi-structural relational repository, not the entire data pipeline system as per Kimball). (Oracle, 2009) defines 3 main types of profiling: attribute analysis, functional dependency, referential analysis (see Figure 11).

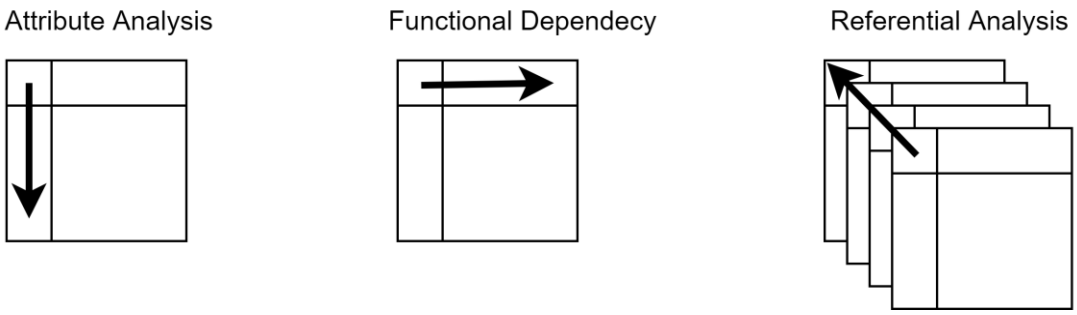

Figure 11. Types of data profiling (Oracle, 2009).

Figure 11 shows that attribute analysis implies checking the content of each column or attribute. Functional dependency is row-level check. Referential analysis implies detecting data objects that refer or does not refer to other objects. In addition to these 3 types, (Oracle, 2009) also highlights that any other custom profiling is possible. All types of analysis aim at discovering various information on the data (see Figure 12).

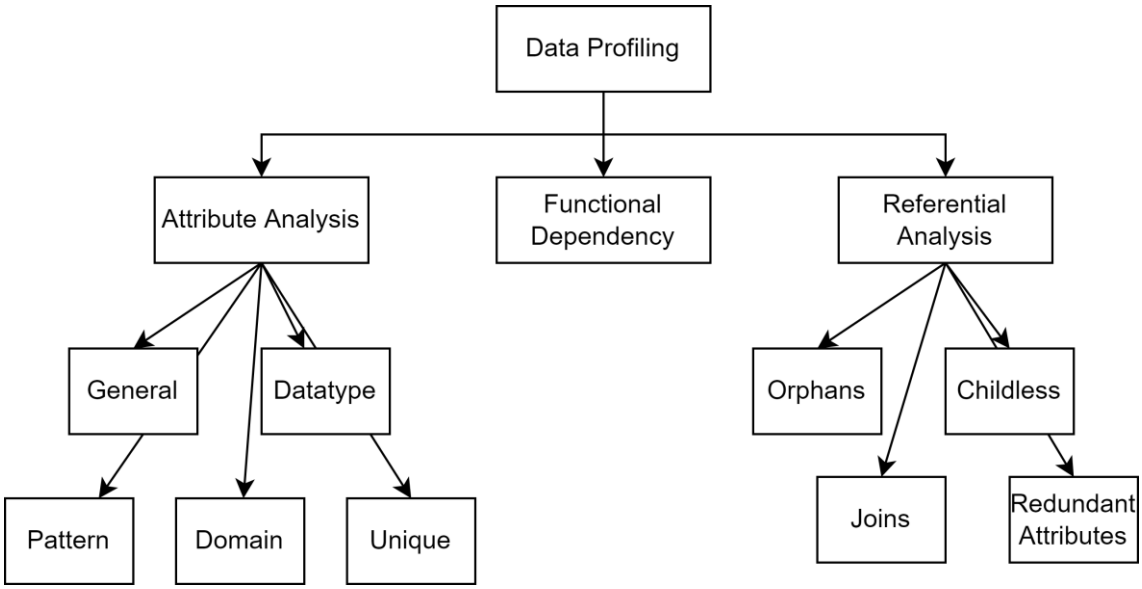

Figure 12. Data profiling (Oracle, 2009).

As seen on Figure 12, attribute analysis aims at discovering patterns in the data records (for example, naming conventions), domains or sets of most common values, data types (for example, precision level of floats, strings, etc.), and uniqueness of data objects. Referential analysis is intended for understanding how various data objects are related to each other.

Thus, there are multiple types of analyses to perform on the data sets. For example, by identifying unexpected values or high number of orphans, it can be concluded that transformation process went wrong, and data was corrupted. Results of such analyses can be presented in form of metadata and allow to get an understanding on content of the data and its health.

3.2.3 Quality Tests Process flow

(Kimball & Caserta, Requirements, Realities and Architecture., 2004) proposes the process flow shown on the picture below for performing various checks (Figure 13).

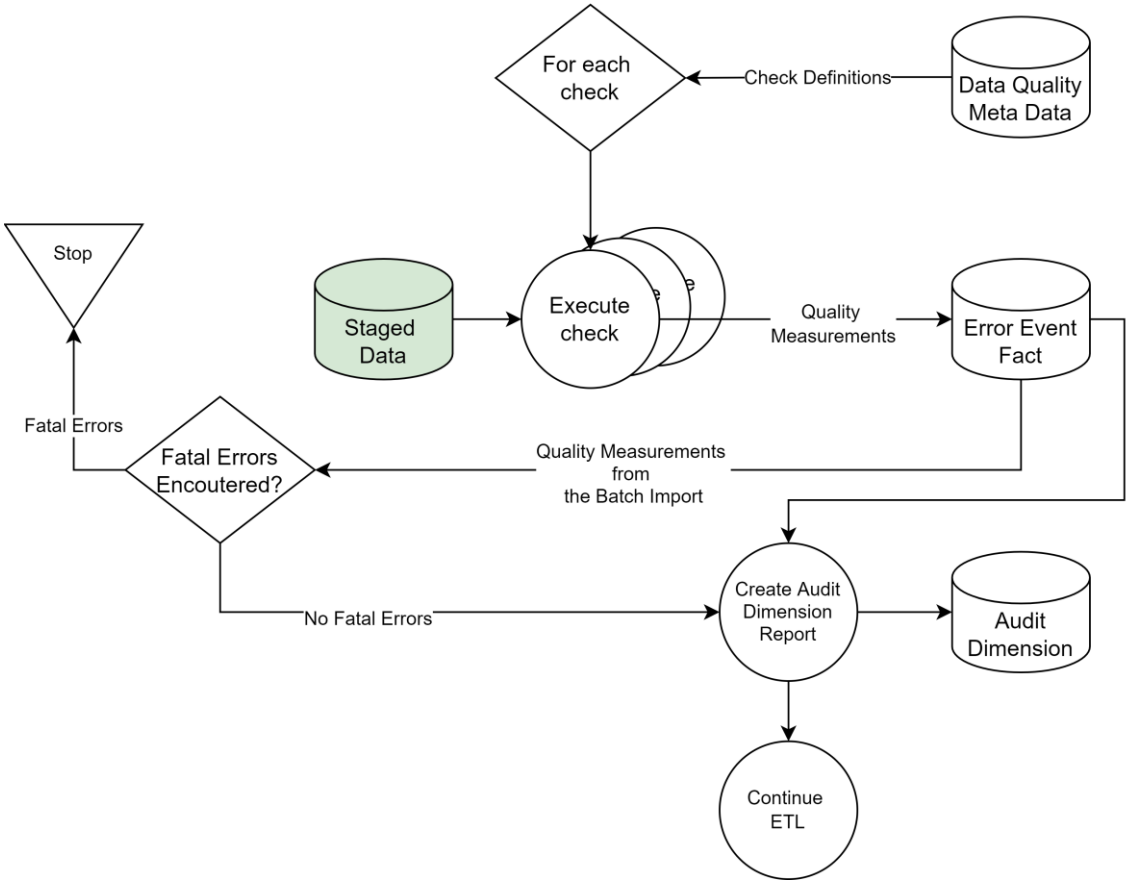

Figure 13. Check process flow (Kimball & Caserta, Requirements, Realities and Architecture., 2004).

The process flow presented on the Figure 13, can be applied to identify and record a quality issue with the data set. For simplicity reasons, concept of “Screen Wave” discussed in (Kimball & Caserta, Requirements, Realities and Architecture., 2004) is not presented on Figure 13 and is not explained in the thesis work.

As it was discussed above (see Section 3.2.1), metadata from every step of data pipeline can be collected and stored in the staging area of a data warehouse. Predefined or historical values can be stored in Data Quality Metadata storage. For example, when talking about process metadata, a rule can be defined for the duration of data extraction stage: if extraction process took more than 1 minute, an error event should be registered. When talking about technical metadata, a rule can be defined when performing historical balance check on the number of unique data objects after conforming stage: if number of unique data objects has changed more than 10% of previous historical value, an error

event should be registered. When talking about business metadata, referential analysis can be applied and if any orphans are found, an error event should be registered.

As Figure 13 shows, multiple quality checks can be run on the data set. The rules for the checks and expected results of the analyses are predefined and stored in Data Quality Metadata storage. If an error event was recorded (e.g., expected value is different from the results of the analysis), this event can be recorded to Error Event Fact storage. If there were many errors recorded or errors that are fatal (see next section) were encountered, the data pipeline can be stopped. Otherwise, an audit report can be created for the current run of data processing and stored in Audit Dimension storage. An Audit Dimension implies a fact table that can be used by a user to track the quality check of current data pipeline run. Audit Dimension is one of the components of data pipeline quality management system proposed in this thesis.

Ideally, when no errors were encountered during data processing, there are no records in audit dimension. When there are many errors encountered, it might not be necessary to stop the data processing. Instead, some other measures might be applied to address increased number of events.

3.2.4 Incident Management

When a check was not passed, severity level of the issue needs to be identified. Thus, for example, when there 2 rows with NULL values out of 1 million rows, the issue might not be considered as critical. In contrast, when a specific stage of a data pipeline has taken considerable amount of time to be executed or a lot of orphan records were found, these types of issues can be considered as fatal, and the data pipeline needs to be stopped to prevent decision making based on wrong data.

According to (Kimball & Caserta, Cleaning and Conforming, 2004), when encountering a problem during data-cleaning step, the data pipeline should not stop or skip the erroneous record. The data issues are something that can be faced daily, and data observability and monitoring system are needed to deal with unexpected conditions and to troubleshoot them. Thus, when performing checks at different stages of data pipeline, it is recommended to record a total score of the set of checks on each stage. Every newly discovered exception during lifecycle of data pipeline can increase or decrease severity

score (Kimball & Caserta, Cleaning and Conforming, 2004). Specific action is assigned to each severity level.

Table 2. Error events severity levels and corresponding actions.

Severity level	**Required Action**
Fatal Error	Stop data pipeline, terminate the application execution.
Warning	Investigate the error and decide whether the pipeline should be stopped or not.
Informational	Be aware, decide on whether checks need to be adjusted.

Further, in some cases immediate actions are required. Therefore, notifications data about pipeline execution and overall health of the system need to be sent to predefined target groups. Besides, it is important to assess the need to send an alarm to a target group. When there are too many alarms sent or the alarms are irrelevant or well-known, with time they will be ignored. That is why, incident management system should include noise suppression (Moses;Gavish;& Vorweck, Fixing Data Quality Issues at Scale, 2022).

Incident management (in terms of pre-defined notification rules, severity levels assessment and actions) is the final component which comprises data pipeline quality management system that is proposed in this thesis.

3.3 Conceptual Framework

Summarizing the key takeaways from the existing knowledge and best practices discussed above, the conceptual framework shown in Figure 14 below highlights the main components of data pipeline quality management system.

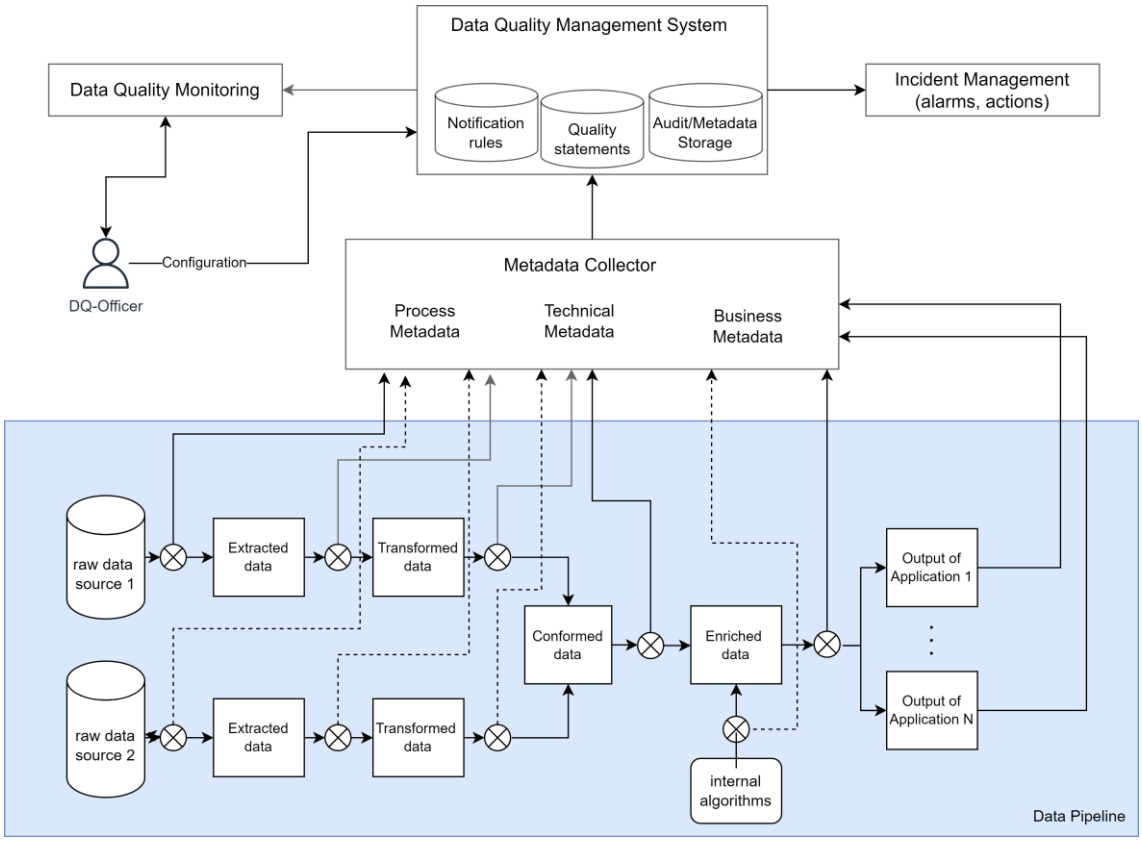

Figure 14. Conceptual Framework of data flow (as a basis for data quality management of the data engineering pipeline).

Figure 14 shows collection of metadata from outputs of applications built using data stored in data warehouse. As discussed above, the data pipeline consists of specific steps that are needed to extract, transform, and load (ETL process) the data to the destination, from where it can be easily accessed and used to build data products such as reports and applications. Transformation of the data in current thesis and in case company implies linking data from different sources and enriching the data with results of internal applications.

As seen from Figure 14, every step of ETL process can be monitored by *collecting metadata*, monitoring, observing, and analyzing it. The final step of the data pipeline is the delivery of the data to a user. That is why metadata can be also collected from the delivery stage as well.

Previously it was discussed that various checks and rules (*quality statements*) can be applied to each stage of a data pipeline. The information on the quality statements should

be stored and accessed when performing a quality test. Results of quality tests execution should have an impact on overall quality score of current health of data pipeline. Depending on the overall score or severity level of a particular check, corresponding stakeholders need to be notified for informational purposes only or to perform a predefined action (troubleshoot, etc.). Responsibility matrix, scoring, severity levels of the exceptions faced, and corresponding actions are to be stored in *Notification Rules* storage. Overall scoring of the data pipeline execution can be stored *in Audit/Metadata storage.*

A dedicated specialist can be responsible for creating, adjusting, and monitoring dashboards, defining notifications rules, quality statements, etc. Helfert and Herrman (Helfert & Herrman, 2002) suggests a role of *Data Quality (DQ) Officer* for proactive data quality management.

It is important to highlight here, that there are two different sources plotted on Figure 14. There can be multiple sources of the original data, every one of each might require specific, for example, extraction and transformation processes and, consequently, separately collected metadata.

The conceptual framework presented on Figure 14 was pulled together from the ideas discussed in Section 3 based on literature and best practice in order to use it as a tool/guidance for analyzing the case company’s challenge in more death and identify the necessary measures and steps that can improve its data engineering pipeline.

4 Current State Analysis of the Data Pipeline Quality Management System at the Case Company

This section discusses the results from the current state analysis of the quality data management system and the data engineering pipeline of the case company. The description is aligned with conceptual framework.

4.1 Overview of the Current State Analysis

As it was described in Section 2.3, workshops with stakeholders were conducted on regular basis. These meetings aimed at developing means for detecting issues within the data pipeline that were faced by the case company previously and foresee other issues. When discussing flaws in the data handling, it became clear that the quality of the pipeline cannot be assessed at the output of the pipeline. Instead, the quality needs to be observed throughout the entire process of data handling (see Section 3.2). All faced flaws in the process could have been discovered at different stages of the pipeline. The current state analysis was built around these stages.

When studying best practices and existing literature, main components of quality management system such as metadata, incident management, audit dimensioning, alarms, etc. were identified. The current state analysis aimed at assessing the status of quality management system components at each stage of the data handling process.

Thus, current state analysis represents a two-dimensional matrix with different stages of the data pipeline on one side and quality management components on another side. Every element of the matrix represents the status of the component: need to be implemented, need to be improved, in a good state.

4.2 Case Company

As mentioned in Section 1.2, the use case company’s business offers a hand-coded automatic platform that collects data from Radio Access Networks (RANs), transforms the data to the unified format, enriches it, so that data product such as analysis reports and applications can be built using this data. As discussed above, source code is one of

the components that defines the quality of data pipeline in general and data it produces. Besides, the case company offers Software Development Kit (SDK) for a customer to build their own applications and to perform their own analysis (Figure 15).

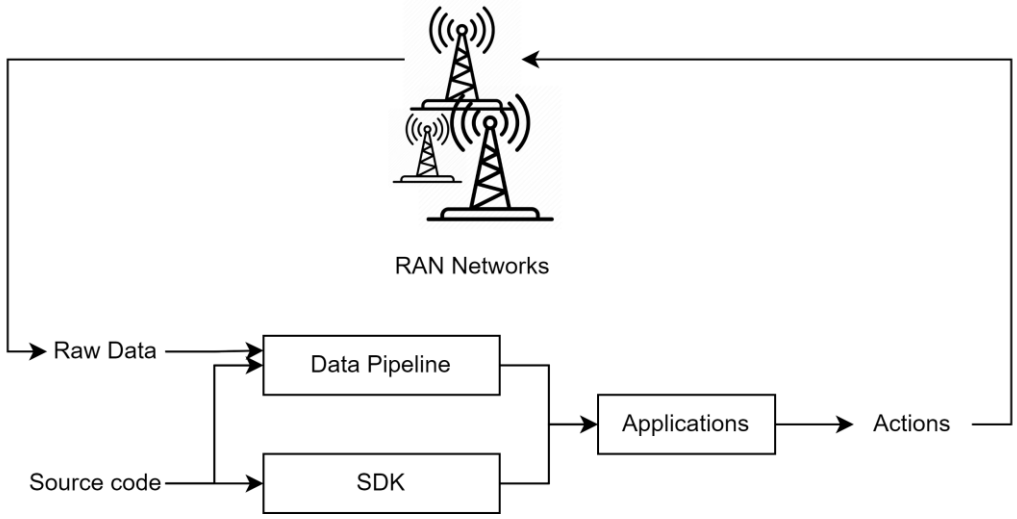

Figure 15. Case company´s offering.

SDK implies programming environment enabling developers to retrieve the data from the data warehouse (data after ETL process is stored in data warehouse) and to create reports, and actions that a radio network can incorporate. This thesis focuses on data pipeline and does not cover SDK and applications development and testing process.

4.3 Description of the Data Flow as Part of the Data Pipeline

The conceptual framework presented on Figure 14 represents one time use of the data pipeline. Thinking of a data pipeline as data processing, it starts with extraction of original data and ends with storing the data in transformed format in the destination (a data warehouse). Whenever there is a new data set to load, data processing should start again. Data pipeline can be executed in real-time or in batches. Batch data import implies that data is moved from the data sources to the destination on scheduled intervals. Batch ingestion is useful when companies need to collect data, for example, weekly. The case company uses batch import on regular basis. The analysis of the data and the algorithms (data delivery) can be happening daily. Therefore, there can be few applications' results delivery per every import (see Figure 16).

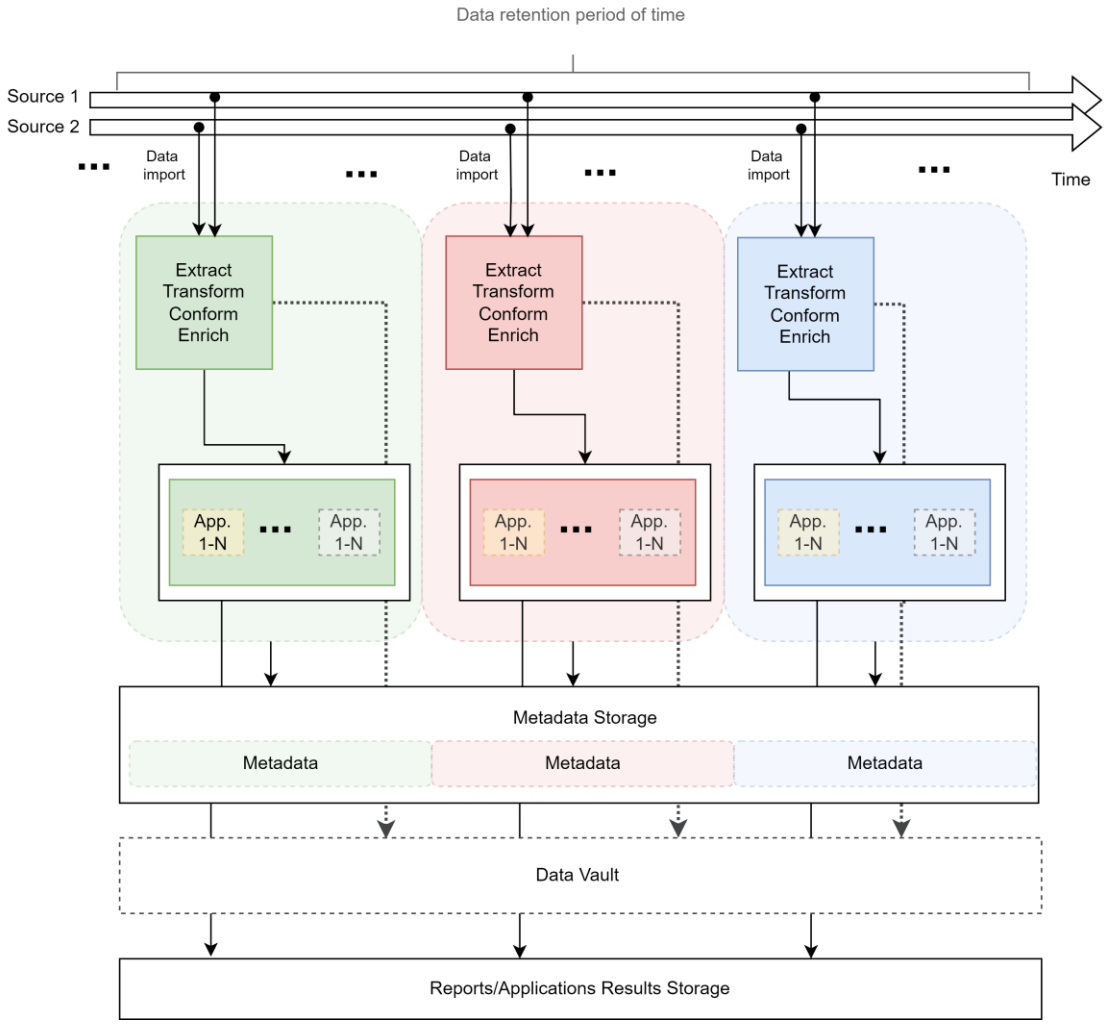

Figure 16. Timeline of imports.

As discussed in Section 3, the main components of data pipeline are extracting, transformation and loading to the destination. In the case company, the destination implies a data vault, where data from previously ran data pipelines is stored. In other words, every time when data extraction from original data sources is scheduled, a data warehouse is being created where transformed and enriched data is stored. All these separated data warehouses represent a data vault. It is possible to access previously created data warehouses within data retention period, after which the data is being removed.

Metadata (discussed in Section 3.2.1) about health of a data set and a pipeline itself that is collected on every stage of data pipeline (extraction, transformation, conforming, enriching, applications) should be stored in separate Metadata Storage and should be

accessible in the next data engineering process for, for instance, performing a historical analysis. Currently, the concept of metadata is not used in the company. As a result, the metadata is not standardized and there is no central repository to access it.

The observability concept discussed in Section 3.2.1, implies (among other things) constant monitoring of the system. For that purpose, there can be few dashboards for different stages of data pipeline which shows for the status of the pipeline: whether it is currently running, or the data transformation has been finalized, what is the amount of data processed and so on. Currently, there are means of monitoring and observing (including alerting) overall health of the pipeline and data. However, it has been discovered and agreed that capabilities for observability should be enhanced. One of the stakeholders commented:

> One of the needs that we want to address is to ensure on an ongoing basis that the data is flowing as expected and the applications have all the necessary data needed to operate correctly.

It should be highlighted here, why metadata is of essential importance: metadata allows to understand whether the data pipeline is properly working or not. The data flowing through data pipeline needs to be inspected for its quality as well. Process of inspecting the data is referred as data profiling. Results of data profiling can be stored in form of, for example, technical metadata. When planning data profiling, it is imported to understand types of the data which are being handled. Various types of the data should be profiled in different ways.

4.3.1 Data types

As mentioned before, there can be multiple sources of original data. The data from various source might need to be processed in different ways and have dedicated storage space. The sources of the data can vary not only based on its location and ways to extract data from them, but also based on its type and format. The case company obtains the data from different sources in terms of its type. Thus, there are two types of data that is being ingested in the data pipeline: timeseries data and cross-sectional data. Timeseries data implies data points listed in time order (for example, how temperature changes within a day). Cross-sectional data represents data collected at one point or period (for example, a student and a course he/she takes).

In case company's data pipeline timeseries data become panel data (also known as longitudinal data) after conformation step. Panel data represents observations for same objects over time. It is important to understand what type of the data is being used in the pipeline, because different data types require different approaches to quality checks and data profiling.

.

4.3.2 Data structures

The case company provides the service to various mobile operators which has their RANs built using hardware from various vendors such as (ZTE, Nokia, Huawei, etc.). The data pipeline that case company offers needs to connect to OSS system of an operator via vendor's and operator's specific API and extract the data. Data structures vary between the vendors. Therefore, once built pipeline for one vendor needs to be adjusted when injecting data from another vendor's RAN. Besides, any update of hardware/software on vendor's or mobile operator's side might have an impact on the data pipeline performance. One of the stakeholders noted:

> One more challenge can be correlation of data, where there are multiple data, sources used. The data might not be from exactly same time

Thus, the data pipeline which was earlier explained as one-time data handling process was put in the context of constant change of the environment, various data types and data structures, development work and other factors contributing into importance of continuous observing of the health of the entire system.

4.4 Strength and weaknesses of currently used Data Pipeline Quality Management System

Based on information collected during the existing knowledge search, as well as based on discussion within stakeholders, and personal observations and experience, the components of data pipeline quality management system presented were identified, as shown in Figure 17 below.

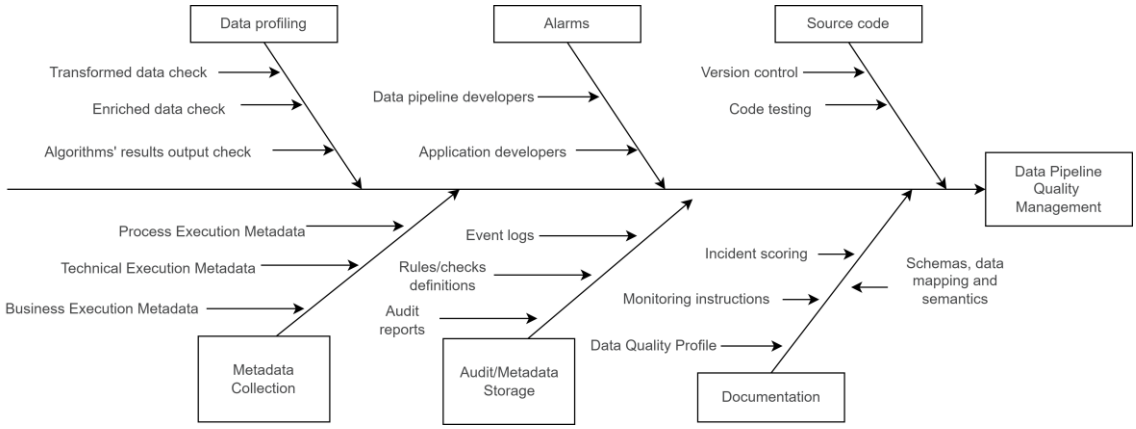

Figure 17. Components of data pipeline quality management system.

The first component of data pipeline quality management system is *Data Profiling*. This component implies attribute analysis, functional dependency, and referential analysis (see 3.2.2). Profiling aims at understanding the data, its types, distribution of values, relations between data objects, etc. At some stages of the data pipeline profiling is not possible, because the data is not presented in a structured format yet or because profiling will not provide any value. For example, the data at Extraction stage can represent multiple text files without clear relation between data elements and clearly defined attributes.

Currently, data is being *Profiled* on some of the stages of the data pipeline (5a from Table 3). However, some additional checks as historical checks could be added to improve quality checking. Besides, the visibility on the results of data profiling is limited since there are no dashboards or repository that would allow to observe results of data profiling (except 5a). It should be noted here that on some stages of the pipeline data cannot be profiled (1a, 1b, 4a, 4b, 5b from Table 3), only overall metadata analysis can be applied for checking the health of theses stages of data pipeline.

The second component is *Metadata Collection*. In this thesis, metadata implies information about the dataset (technical metadata), information about performance of each stage of the data pipeline (process execution metadata), and information about logical connections between data objects, source system information, etc. (business metadata).

Metadata concept is not being used in the case company. There are some KPIs such as time of a process execution (could be classified as process execution metadata) that are being monitored, but they are not classified, are not documented, and are scattered over different dashboards and the 3rd party tools. Some of the stages of the data pipeline are observed better, than other. For example, duration of extraction, transformation, and application stages is presented on the dashboard and can be viewed to detect any misbehavior (1c, 2c, 4c, 6c from Table 3). Besides, the case company also uses some KPIs such as number of unique data entries, percent of orphans, etc., which can be classified as technical metadata (1d, 2d, 5d from Table 3). As for business metadata, it represents the entire data processing logic and is cannot be allocated per every pipeline stage (1-6e from Table 3). It should be noted that there is no central storage location from where metadata, quality statements and notification rules can be accessed (1-6i from Table 3).

The third component is *Audit/Metadata storage*. Metadata collected from various stages of the data pipeline over different scheduled runs can be used to perform, for example, a historical comparison. For example, after implementing changes into an algorithm, process execution metadata from previous version of the data pipeline can be compared with updated version in order to access whether there was a negative impact on the performance or not. Besides, a set of predefined rules can be collected in Quality Statements storage for performing checks. For example, if execution of a process took more than predefined time, an error event can be registered. As discussed in 3.2.3, results of checks performed can be stored and presented in a form of log files to a stakeholder. Further, multiple error events can be registered during the data handling process but none of them could be categorized as critical. Assessing overall score of the data pipeline execution and collecting the reports with the assessment in a form of Audit report, can allow a stakeholder to understand how the process performed over its lifetime.

Audit concept (results of checks, discussed in 3.2.3) is not being used in the case company. The logs from different stages of data pipeline exist (based on these logs the 3rd party tool send alarms to developers), however they are not stored and cannot be easily assessed over time (1-6i from Table 3).

The fourth component is *Alarms*, which implies sending notification after an error event to a target group for further action upon the event. Two target groups were discussed in the thesis: data pipeline developers and application developers. These target groups are responsible for different stages of the pipeline. For example, an application developer should be responsible for troubleshooting and investigating an error event in an application. However, data pipeline developers might get notification as well in order, for example, not to deploy major updates into the source code.

Alarms are being managed on rather good level for certain stages of the data pipeline (1f, 3f from Table 3), implying that target groups are well defined, and they are sufficient. However, there is a very limited amount of target groups set currently. For example, there are no alarms being sent to application developers (6f from Table 3).

The fifth component is *Source Code*. As it was discussed in 3.2, the data pipeline quality is defined not only by quality of the data itself, but also by a source code. The source code here implies a set of instructions and statements written using a computer programming language in order to perform certain operations with the data (like transformation, enriching, etc.). Besides, applications built using the data from the data pipeline relate to the source code as well.

Coding process of the data pipeline in terms of version control is in very good condition. The process of the development is very well established and defined. Some of the issues faced by the case company, due to which need in this project emerged, could have been discovered on during code testing phase instead of data profiling. For example, high number of orphan records (records, which reference non-existing primary data object (Database.Guide, 2016)) arose due to change in unique keys generation of data objects. However, quality of the code is outside the scope of this thesis.

The last but not the least component is *Documentation*. *Documentation* is essential part of any management system. Without clear description of a system and its main components, it is extremely difficult to understand how the system operates, how to access it, and how to plan any changes required. Documentation should cover all the components of data pipeline in detail. It should explain a stakeholder the overall process of assessing the quality of data handling process, guide a user to audit reports and monitoring dashboards, etc. Notification rules, quality statements, alarms, target groups,

actions should be explained in the documentation and a user (with sufficient access rights) should be guided to the place where all these policies are stored.

The *documentation* in the case company on current topic is not in a good condition. During the project execution (during the stakeholders' meetings) the documentation creation has started in order to be able to monitor the flow of proposed actions and changes needed.

Table 3 below aims to visualize the status of every component of quality management system. For every stage of data pipeline, it shows the status: red – need to be implemented, orange – need to be improved, green – in a good state. Column names in the table correspond to the main components of data pipeline quality system.

Table 3. Strength and Weaknesses of Data Pipeline Quality Management System components per each stage of data handling process

		a	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*I*
	Stages of data pipeline	Data profiling		Metadata collection			Alarms	Source code	Documentation	Audit/ Metadata storage
		Cross-sectional data	Time series data	Process Metadata	Technical Metadata	Business Metadata				
1	Extracted data	-	-					Out of scope		
2	Transformed data									
3	Conformed data									
4	Internal algorithms	-	-							
5	Enriched data		-							
6	Applications data									

4.5 Current State Analysis summary

Previous section explains the main components of quality management system of the data pipeline. Section also assesses condition of each component on different stages of the data handling process in case company. The status of each element of the Strength and Weaknesses matrix (Table 3) was identified based on the fact of its existence and

whether there were any proposals for implementation suggested during the workshops with the stakeholders.

One of the stakeholders, after reviewing the results of current state analysis presented in the thesis, provided the following feedback:

> The current state analysis is accurate. The system lacks relevant alarms, and some of the existing alarms are not used for informing correct user group. The data ingestion pipelines are carrying out data validation tasks, but their coverage is not 100%, which means there are needs for improvement on this area. It is true that metadata is not consistently collected, although some of the proposed raw data would be available from the system.

This thesis addressed data pipeline quality and did not intend to propose detailed technical solution or process flow for each component of the quality management system. These components constitute a framework, which can be used by the case company in order to enhance the observability of the data pipeline in consistent and aligned way. It should be noted here that observability implies monitoring, alerting, tracking, comparisons, analysis, logging, etc. To some extent, observability and data pipeline quality management system can be used interchangeably.

Based on the findings from the current state analysis, the thesis proposed improvements and/or features to implement related to metadata collection, alarms, and documentation components of quality management system. Source Code component is left out of the scope of the thesis work due to enormousness of this topic. Data profiling can be viewed as means of obtaining Technical Metadata and, therefore, is not seen as something that requires further analysis and discussion in addition to what was covered in 3.2.2. Audit/Metadata storage is not discussed in further detail as well. The need for collecting metadata and audit reports was clearly established. Further discussion on that topic should focus on technical details of its implementation and planning the process flow of metadata collection and storing.

5 Development Proposal to Enhance Data Pipeline Quality Management System

This section merges the results of the current state analysis, the best practices and on internal co-creation and discussions towards building a proposal. The proposal is built to address observability problem, to enhance data pipeline quality management system and mitigate weaknesses, presented in Table 3. Sections 5.2.1 and 5.2.2 addresses the quality management in general. Section 5.2.3 discusses collection of metadata from, for example, data profiling and process execution and way to monitor it. Section 5.2.5 focuses on alarming.

5.1 Overview of the Proposal Building Stage

This thesis was conducted as part of the internal project in the case company. While working on this project, during workshops the stakeholders went through practical issues that are faced from time to time and ways to identify them and react on them in time. The Proposal presents the results of this co-creation.

This proposal summarizes and generalizes all the findings into a form of a framework, which should be considered when further enhancing the quality management system of the data pipeline. A framework implies supporting blocks out of which such the system can be comprised. These blocks were presented on Figure 17 and include: metadata collection, data profiling, alarms, source code, audit/metadata storage, and documentation. Implementing additional features, additional checks, monitoring capabilities, etc. should be aligned and aim at reaching the predefined end goal. Proposals from this thesis outline such a goal for metadata collection, alarms, and documentation components of the quality management system.

5.2 Initial Proposal

As discussed, observability in data pipelines defines the level of context which is provided on data issues. Observability is an umbrella term and implies monitoring, alerting, analysis, comparisons, etc. Without standardizing and centralizing activities on observability, teams working on data pipeline development and users of the data and data products cannot have satisfactory level of awareness for proactive high-quality maintaining activities.

To address enhancing the data quality management system in general, the stakeholders proposed to assign a dedicated person responsible for that (5.2.1), to create and maintain related documentation (5.2.2), to collect metadata and properly present it (5.2.3), and to reassess responsibilities and actions upon error events (5.2.5).

5.2.1 Proposal Element 1: New role within the team

The data pipeline is an overly complex system that requires specialized knowledge and expertise. Currently, the case company employs the product owner, architect, data pipeline and SDK tool developers and application developers (business analysts).

The recommendation is to hire or to assign a role of *a data administrator* (or data warehouse quality engineering manager), whose responsibility would be as listed in Table 4 below.

Table 4. Responsibilities of a data administrator (or data warehouse quality engineering manager).

1	Maintaining and administrating case company data assets
2	Standardizing data models and data attributes
3	Maintaining metadata repository
4	Monitoring and reporting data quality
5	Root cause analysis of issues with the data and data pipeline
6	Overseeing data on front-end and back-end stages of the data warehouse (applying Kimball's terminology)

In other words, the responsibility would include the work that is done in this thesis work on daily basis and on more detailed level with close collaboration with management team (who needs to understand the quality of the data pipeline), developers, architect, and product owner.

5.2.2 Proposal Element 2: Maintaining Documentation

As mentioned in the current state analysis, the documentation on the data pipeline quality system does not exist. It is recommended to create the documentation and maintain it. Every step of the data pipeline should be explained with the references to the code, checks, quality statements and references to the dashboards where every step is monitored. Otherwise, only a data pipeline developer knows what exactly is done between source systems and presentation area. Figure 18 shows an example how relations between data objects can be documented.

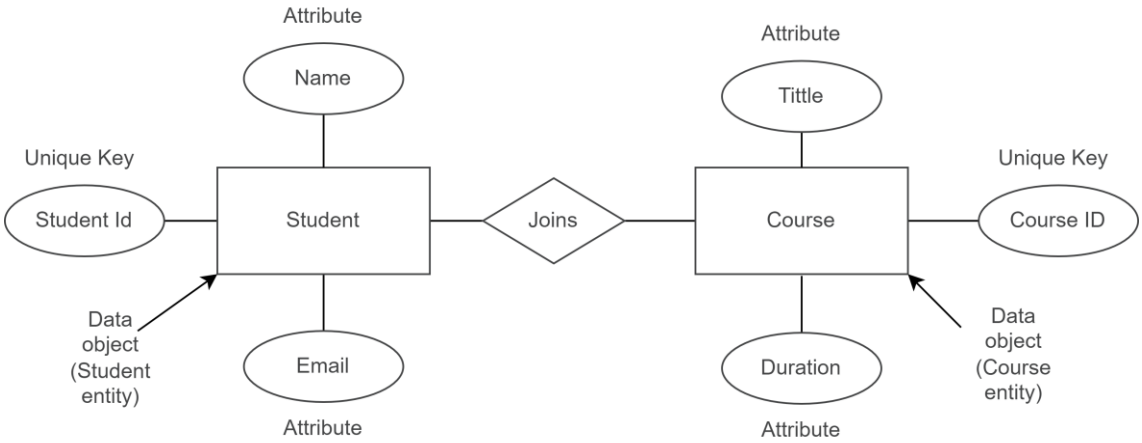

Figure 18. Example of entity relationship diagram.

For example, after transformation stage of data pipeline, it is recommended to collect following metadata: 1. Source tables; 2. Surrogate key generators; 3. Relations between data objects; 4. Filters; 5. Transformations (like renaming). After the conforming stage, a relationship diagram should be created and visualized in a format that is easy to understand and study by application developers (see Figure 18). Unique identifiers and surrogate keys, relations between tables, cardinality of relations (one-to-one, one-to-many, many-to-many) should be shown on that diagram. The documentation package was exemplified on *the relationship diagram* above.

In addition to a relationship diagram, a *logical data map* should be created and maintained. The logical map should include following components: source database of file name, source table name, source column name, target table name, target column

name, transformations. The logical map is needed in order to track, explore and troubleshoot the data.

Besides, it is important to enforce *standardization* to all stages of data pipeline, including application development. For example, standardize naming conventions of data attributes. Completed and updated documentation can help to eliminate bus factor of one person (Dhingra, 2022) and to help with onboarding new team members.

5.2.3 Proposal Element 3: Enriched Data Profiling as an Application

Due to the lack of means to visualize Technical Metadata and due to its limited collection, it is proposed to create an application that would perform profiling of enriched data. Meaning that an application would query all the data available and perform attribute analysis, functional dependency, referential analysis (section 3.2.2).

5.2.4 Proposal Element 4: Metadata and Monitoring

As discussed in the previous sections, metadata from different stages of the data pipeline should be collected, stored in central repository, and presented on a dashboard for monitoring, observability, and quality assessment purposes.

It is proposed to have a single dashboard where all relevant metadata is presented, and quality of the pipeline can be accessed. An example of the dashboard is given on Figure 19. As it was shown on Figure 16, raw data import can happen, for example, every week. To perform historical balance test (comparing current data to previously obtained values), metadata data from previous batch imports should be stored and should be accessible during the check. The dashboard presented on Figure 19 is the missing comparison page, which would show how the performance changes over different batch imports.

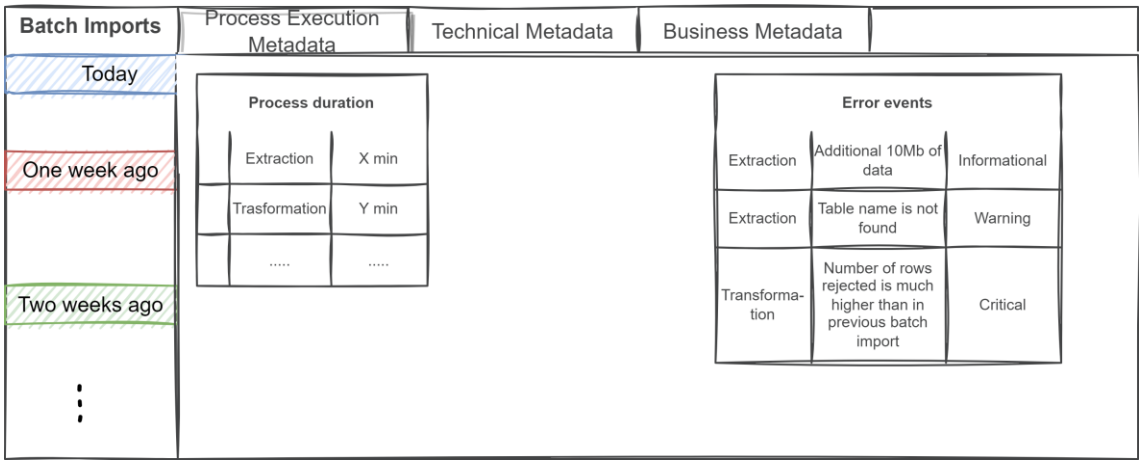

Figure 19. Example of data pipeline quality monitoring.

As shown from Figure 19, the metadata collected on various stages of the data handling process and results of various checks can be presented on a central monitoring dashboard. For every run of the data pipeline (for example, today, one week ago, two weeks ago, etc.) such information can be presented on separate pages of the dashboard. The impact of changes in the raw data or impact of the pipeline code development that did not trigger an error event could be clearly seen from this comparison page. Comparison is to be done over all metadata available.

Tables 6 and 7 below shows the checks/KPIs that are proposed to be monitored and presented on the dashboard. Process execution metadata is aimed at provided overall understand of data pipeline performance and storage needed. Technical metadata implies attribute analysis, functional and referential analysis discussed in Quality Tests and Data Profiling section.

Table 5. Process Execution Metadata.

	Extracted data	**Transformed data**	**Conformed data**	**Enriched data**	**Output data from algorithm**
Cross sectional data	Time taken for data extracting. Warnings, errors generated.	Time taken for data parsing. Warnings, errors generated.	Time taken for data conforming. Warnings, errors generated.	Time taken to enrich the data (access metadata from previous imports, run internal algorithms). Warnings, errors generated.	Time taken to generate the output. Warnings, errors generated.
Timeseries data	Time taken for data extracting. Warnings, errors generated.	Time taken for data parsing. Warnings, errors generated.			

Table 6. Technical Metadata.

	Extracted data	**Transformed data**	**Conformed data**	**Enriched data**	**Output data from algorithm**
Cross sectional data	Number of rows in the files.	Number of rows loaded. Number of rows rejected.	Number of unique entities. Distribution of property values.	Comparison of values to previous values.	Values within the range. Comparison to previous values.
Timeseries data	Number of rows in the files.	Number of rows loaded. Number of rows rejected. Null values	Data types of properties. Number of unmapped data entities. Number of orphan records. Number of relationships per data entity.		

As for Business Metadata, it is difficult to allocate it per data pipeline stage. List of business metadata recommended for collection by (Ponniah, 2001) include: (a) data transformation rules, (b) logic data map, (c) data ownership, (d) entity relationship map, and (e) report distribution information.

5.2.5 Proposal Element 5: Error Event Actions and Alarms

Results of every check performed for every batch import should be stored in form of logs so that they can be later accessed if needed. Logs that are classified as informational, critical, or fatal should be presented on data pipeline quality dashboard for further investigation or immediate reaction. As discussed in CSA section, currently there are no alarms that are being sent to application developers. The reports and applications (the presentation area of data warehouse) are something that customers mostly interact with.

A flaw in the current data pipeline might be unnoticeable for an end user (for example, incorrect transformation of the data attribute that is rarely used), but a flaw in results of an application that a customer will interact with is critical for customer satisfaction.

Besides, when talking about closed-loop algorithms, all the changes are being done to RAN without involving a human. Therefore, applications need to pass the checks designed for them and notify an application developer immediately. In case of critical errors (for example, number of results has changed drastically) an application should be switched from closed-loop to open-loop (that will not do any changes to RAN).

Alarms should target different user groups (data pipeline developers (DPD), application developers (AD), product owner (PO), architect (A), etc.). Error events and actions they require needs to be classified and documented in a form of responsibility matrix (Table 7).

Table 7. Example of responsibility matrix.

Stage	Check example	Alarm type	Target group	Action
Extracted data	Size of data extracted increased by 10 Mbytes	Informational	DPD, A	Be aware, decide on whether checks need to be adjusted.
	Part of source are not accessible	Fatal Error	DPD, PO, A	Stop the data pipeline
Transformed data	Number of rows injected grew by 10%	Warning	PDP	Investigate the error and decide whether the pipeline should be stopped or not.
	Values are outside predefined statistical intervals by 5%	Warning	PDP, AP	Investigate the error and decide whether the pipeline should be stopped or not.
Conformed data	Orphan records number decreased by 5%	Informational	DPD, AP	Be aware, decide on whether checks need to be adjusted.
Internal algorithms	Execution time took 10% more time	Informational	DPD, AP	Be aware, decide on whether checks need to be adjusted.
Applications data	An algorithm's results number has changed by 90%	Fatal Error	AP	Stop the algorithm execution (change from closed-loop to open-loop)

Table 7 shows an example of checks that can be performed at each stage of data handling process; types of the alarms that can be generated; target groups for these alarms; and actions that are to be taking because of an alarm.

5.3 Summary of the Proposal

The proposals in this thesis targeted metadata collection, alarms, and documentation components of quality management system (see Figure 20). As it can be seen from Figure 20, Source Code component is left out of the scope of the thesis work due to enormousness of this topic. Data profiling can be viewed as means of obtaining Technical Metadata and, therefore, is not seen as something that requires further discussion in addition to what was covered in 3.2.2. Audit/Metadata storage is not discussed in further detail as well. The need for collecting metadata and audit reports was clearly established. Further discussion on that topic should focus on technical details of its implementation and planning the process flow of its collection and storing.

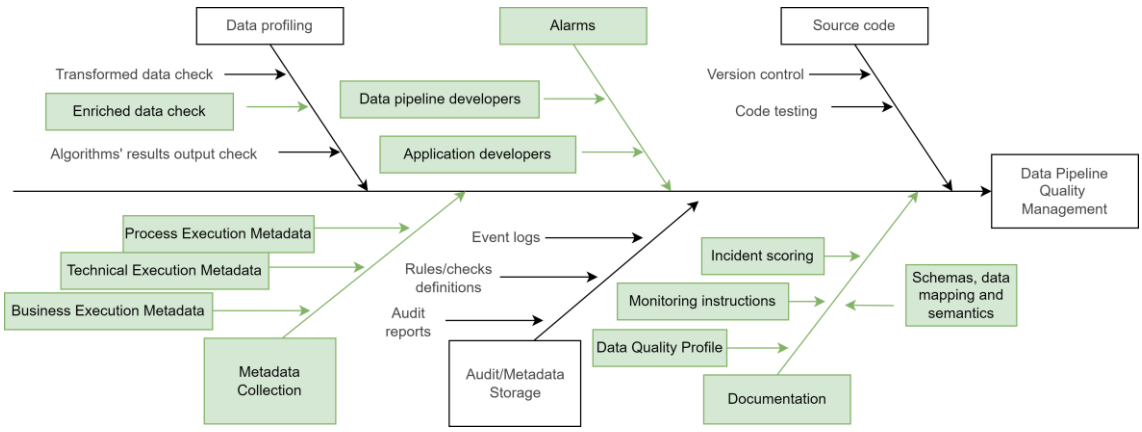

Figure 20. Components of data pipeline quality management system addressed in the Proposal.

Returning to the Conceptual Framework presented on Figure 14, the elements that were addressed in the proposal can be visualized in a clear way (Figure 21).

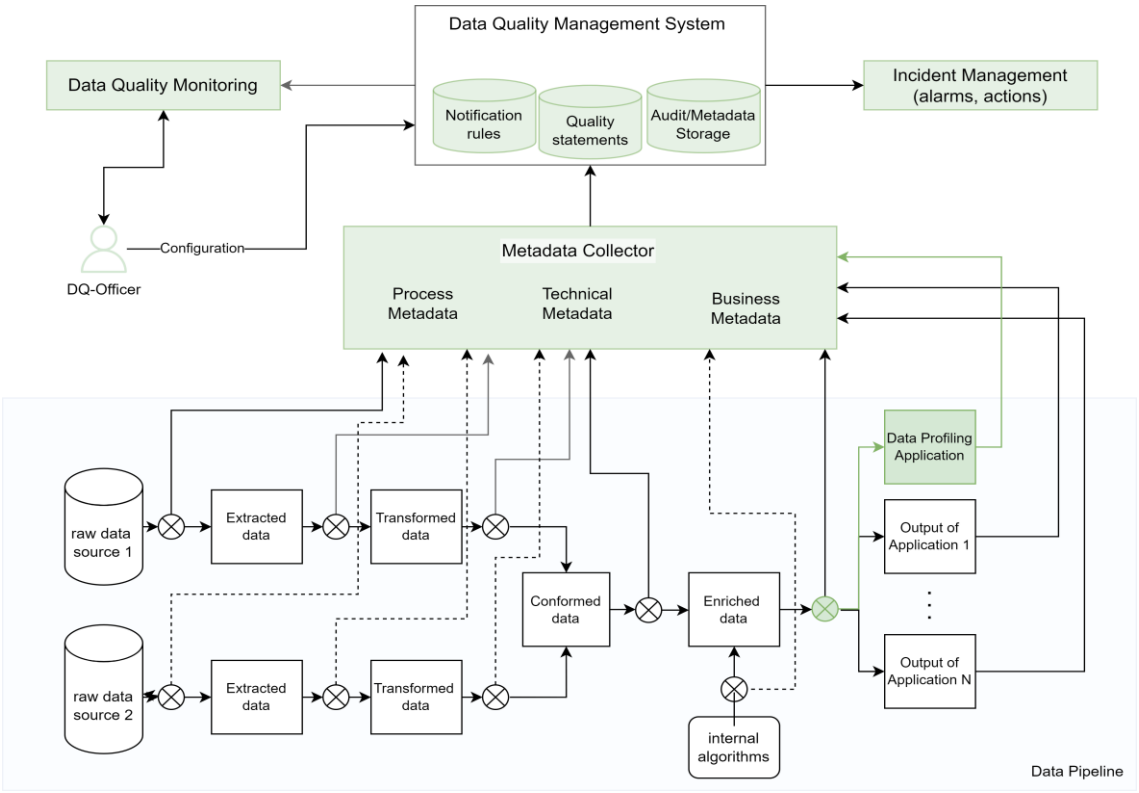

Figure 21. Proposal draft.

The Proposal draft depicts the discussed solutions. Figure 21 includes, for example, the proposed application for enriched data stage of the data pipeline. Instead of performing queries directly in a database that stores enriched data, it was proposed to create an application. The main motivation for that are visualization capabilities which are available for an application by default. When performing queries to the database, means for visualizations should be developed. The Proposal draft picture keeps it in place to re-consider it at the next stage of development.

5.4 Findings from Data 2 Collection

Table below aims at putting together Conceptual framework (CF), Current state analysis (CSA), and feedback about the proposals that were obtained from stakeholders.

Table 8. Key stakeholder suggestions (findings of Data 2) for Proposal building in relation to findings from the CSA (Data 1) and the Conceptual framework.

Proposal theme	*Input from literature **(CF)***	*Key focus area from **CSA** (from **Data 1**))*	*Suggestions from stakeholders for Initial Proposals improvement, summary **(from Data 2)***
1.New role in the team (Data warehouse quality engineering manager)	(Helfert & Herrman, 2002)	Not discussed in CSA. The role does not exist, and was discussed as part of the Proposal	The proposal was commented by one of the stakeholders: *"Here I got a bit confused on product development company vs. company using the product (for example "maintaining the case company data assets"). Where would this new role be? Also, since this is "on daily basis", would not it be a bit in conflict with the target to provide an automated solution?"*
2.Documentation	(Kimball & Caserta, Requirements, Realities and Architecture., 2004)	Status based on CSA: – need to be developed	The proposal was commented by one of the stakeholders: *"The standardization of data models etc. would in my view be more like developing overall software engineering practices, not so much specific to data quality, but quality in general." (Stakeholder)*
3.Enriched Data Profiling as an Application	Enriched stage of the data pipeline is one of the stages after which metadata collection is proposed	Status based on CSA: – need to be improved	To be tested and verified in the next stage.
4.Metadata Collection and Monitoring	(Kimball & Caserta, Metadata, 2004)	Status based on CSA: – overall concept is not used, though some metrics are collected	The proposal was commented by one of the stakeholders: *"This is true, it is not used as a concept (if not considering the data import metrics that there might be some). One aspect that needs to be considered, is for automated gates is that data to compare to. There may be "bad" (metadata) versions, for example due to software bug, and one cannot always compare to previous import, but needs to compare to some previous "accepted" / considered-to-be-good import." (Stakeholder)*
5.Alarms	(Kimball & Caserta, Cleaning and Conforming, 2004), (Moses, Gavish, & Vorweck, Fixing Data Quality Issues at Scale, 2022)	Status based on CSA: need to be developed for some of the stages of the data pipeline	The proposal was commented by one of the stakeholders: *"I would adjust this a bit further to consider from product deployment point of view. When an application is in production, the operations and support people are the ones who would first need the notifications, and then may be customer people or product support people. True, they will after that contact the application developers, but still there is a difference." (Stakeholder)*

As seen from Table 9, there was a wide discussion, and the proposals were mostly found relevant on a general level. However, more accurate and detailed planning was requested by stakeholders. For example, it was suggested that the new role in the team and alarms should be defined properly and addressed from different angles: from the case company side (when the data pipeline is under development) and from a customer side (when the data pipeline is deployed to customer's environment).

Next, the parts of the proposal that were possible to test and validate (for example, *Proposal Element 3: Enriched Data Profiling as an Application*) were moved forward to validation. Such proposals as *Proposal Element 5: Error Event Actions and Alarms* and *Proposal Element 4: Metadata and Monitoring* could not be tested as such. Instead, they were recognized as the direction to move to. Changes related to *Proposal Element 5* and *Proposal Element 4* are to be incorporated and refined step by step starting with suggestions in corresponding tables (see Table 5, Table 6, Table 7). *Proposal Element 2: Maintaining Documentation* is viewed as a call to action and is to be prioritized in daily activities. *Proposal Element 1: New role within the team* is to be described in detail on how to incorporate this position into current processes and to be decided on higher levels. The Final proposal addressed these concerns to the extent that was possible to address at this framework level. The Final proposal, based on the results of valuation, is shown at the end of Section 6.

6 Validation of the Proposal

This section reports on the results of the validation stage and points to further developments to the initial Proposal.

6.1 Overview of Validation Stage

Validation of the proposal started by experimenting for Proposal Element 3: Enriched Data Profiling as an Application. The idea was implemented and assessed during the workshops with stakeholders.

Some of the proposals, which are conceptual and high level, are not possible to be validated by piloting or experimenting, instead they were evaluated based on feedback from experts and managers. Validation of Proposal Elements 1, 2, 4, 5 was held in a form of general level discussion and receiving feedback. It should be noted here that these proposals were built by generalizing issues faced by the case company. Therefore, the proposals are not “greenfield.” Greenfield implies lack of components imposed by prior work or experience. Instead, the proposals are based on daily work experience, observations, communications, and best practices applied to data pipelines in general.

Additionally, the proposal correlates quite a lot with customer requests. Customers have requested to implement historical balance tests to application stage of data pipeline with an action to change closed-loop to open-loop in case of error event. Besides, there were a few customers who have requested *data pipeline checks* and *audit reports.* These customer requests were also included as part of Data 3 at the Validation stage.

6.2 Developments to the Proposal (based on Data Collection 3)

Table below summarizes the inputs for developing final proposals that were obtained from stakeholders in Data 3 collection, the validation stage.

Table 9. Key stakeholder suggestions (findings of Data 1, 2 and 3) for Proposal building.

Proposal theme	*Key focus area from **CSA** (from **Data 1**))*	*Suggestions from stakeholders for Initial Proposals improvement, summary **(from Data 2)***	*Further developments to the Initial proposals **(from Data 3)***

1.New role in the team (Data warehouse quality engineering manager)	Not discussed in CSA. The role does not exist, and was discussed as part of the Proposal	The responsibilities were recognized as needed by the stakeholders, but the role should be carefully planned.	Too detailed level. Cannot be further addressed within the framework.
2.Documentation	Status based on CSA: – need to be developed	The need for documentation was approved by the stakeholders. Documentation should start with proposed items in the thesis.	Too detailed level. Cannot be further addressed within the framework. Main components of the documentation presented in 5.2.2 were approved.
3.Enriched Data Profiling as an Application	Status based on CSA: – need to be improved	When trying out and discussing this proposal, it was assessed as not a very efficient way forward.	When trying out and discussing this proposal, it was assessed as not a very efficient way forward. Try the 3rd party tools.
4.Metadata Collection and Monitoring	Status based on CSA: – overall concept is not used, though some metrics are collected	The stakeholders approved the metadata collection and monitoring with various comments. Collection of relevant metadata should be expanded, starting with proposed data in the thesis.	Too detailed level. Cannot be further addressed within the framework. Examples of data to collect presented in Table 5 and Table 6 were approved.
5.Alarms	Status based on CSA: need to be developed for some of the stages of the data pipeline	The stakeholders approved the extension of alarm groups with various comments. Extension of target alarm groups should be incorporated, starting with proposed target groups in the thesis.	Too detailed level. Cannot be further addressed within the framework. Examples of data to collect presented in Table 7 were approved.

During the very first workshops, the need to implement *additional features to the data platform* were identified to be able to incorporate some of the proposals. For example, to perform historical balance test on application stage of data pipeline, it was required to add possibility for an application developer to access results from previous application run (inside the same import batch or from the previous). The feature has been implemented and currently is being tested by an application developer.

Besides, for example, *Data profiling* for enriched step of data pipeline was proposed to be implemented in a form of an application. However, that ended up being not feasible because of the data storage limitations. Proposed application would need to access data vault, obtain data from previous batch import, store this data within its operational memory, obtain data from current batch import, store this data operational memory, and perform historical balance test. In that case, an application would need to store something that is already being stored in the data vault within its own memory, which is not an optimal way of using available storage capacity. Therefore, another way of addressing this step from practical perspective need to be figured out. Currently, it is decided that an application should perform historical balance test of its own results. In fact, (Hinrichs & Aden, 2001) and (Kimball & Caserta, Cleaning and Conforming, 2004) specify that data profiling should happen before storing the data in the data warehouse. This best practice was ignored when proposing such type of application. It should be mentioned here that the idea behind that proposal was to present a customer with a proof

of correctly working data pipeline (original data is not distorted) and consistent data. Considering unavailability of metadata in general, that was considered as an option.

Even though the concept of *Metadata* is not used in the case company, based on CSA results, the process execution data collected on such stages of data pipeline as extraction and applications is in a decent shape. This data is being *monitored* and presented on different dashboards. However, this information is scattered. A global dashboard that would collect and present *all metadata* and *error events* is rather promising idea from data pipeline quality monitoring perspective; however, it is difficult to implement. Even though data pipeline is hand-coded, it is still integrated with various 3rd party tools. For example, cross-sectional data and timeseries data are stored in different databases. Created data quality statements (metrics) for every database, transferring the metrics from data quality statements storage to the place where checks are run and transferring the results of the checks to a dashboard that can process various formats is not a trivial task and requires constant development and maintenance. With the currently available resources it is rather a challenging task. That is why proposed *Role of data warehouse quality engineering manager* should also imply responsibilities related to data pipeline monitoring development and knowledge of such tools as, for example, Grafana, Prometheus, etc.

In general, the stakeholders concluded that development of *data pipeline quality management system* should be done by studying capabilities of the currently used 3rd party tools and assessing the proposal in terms of whether these tools can be used for proposals' implementation. However, the focus of this thesis was to develop a framework for enhancing quality system with the emphasize on data checks. The thesis is intended for strategy alignment. The technical assessment and implementation part is outside the scope of this thesis and was left for further work in scope of the internal project. Besides, revealing used tools, detailed analysis of their capabilities and exact descriptions of the metrics that need to be applied is rather sensitive information and cannot be disclosed.

Additionally, a good indication of a valid proposal is that it correlates well with customers' requests. Customers have requested to implement historical balance tests to application stage of data pipeline with an action to change closed-loop to open-loop in case of error event. Besides, there were a few customers who have requested *data pipeline checks* and *audit reports.*

One of the stakeholders, after reviewing proposals provided the following feedback:

> The proposed items in the Developmental Proposal are all sound and relevant. Data engineer role is relevant, and the proposed tasks can be all be considered as mandatory in big data type of system. How the role is realized requires careful planning. Documentation is obviously very important in ensuring the data quality. As the system has been under rapid development, the documentation has been lagging behind, but this will improve when the pipelines mature. Part of the proposed Metadata already exists within raw data, but it has not been fully utilized. This will be improved according to the proposal in order to improve the monitoring capabilities of the system. Error event actions and alarms is also an active area of development in the system. This proposal brought valuable insight on how to improve the targeting of the alarms so that their impact is maximized.

6.3 Final Proposal

Based on the feedback collected, no notable changes were imposed on initial proposals. The main components of data pipeline quality management system proposed are relevant and required further detailed development in terms of their implementation. The only exception is *Proposal Element 3: Enriched Data Profiling as an Application*. It was suggested to put this on hold at the moment and consider available 3rd party tools for data profiling and for presenting results of this profiling on a dashboard, which potentially can be developed using 3rd party tools' capabilities (see Figure 22).

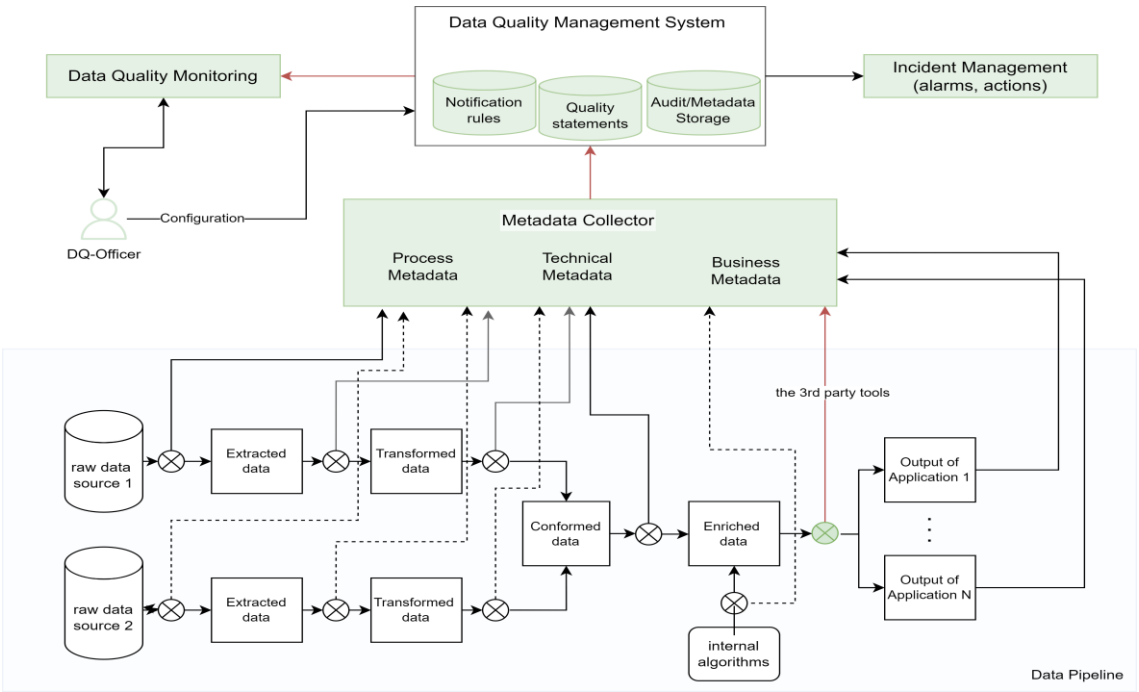

Figure 22. Final proposal.

As seen from Figure 22, the main change in the proposals is related to moving data profiling from application stage to enrich stage and using the 3rd party tools for that. There is still a possibility to implement that in a proposed format, but it might require additional feature implementation to the product in order to speed up queries and optimize data storage.

The goal of the thesis was to develop a framework (layered structure of elements of a system under development) for enhancing data pipeline quality management system. The main components of the framework that were discussed are: metadata and metadata storage, notification rules, quality statements, audit storage, data profiling, incident management. The goal of the thesis was reached, the main components were discussed, references from the best practices are gathered and can be used for further detailed planning of every component.

7 Conclusion

This section highlights the relevance of the research conducted, lists the proposals, and discusses concluding thoughts on the thesis work.

7.1 Executive Summary

This thesis focused on assessing quality management system of the case company, more specifically, its data pipeline. The thesis was performed in the company that offers the data pipeline as part of a product for developing autonomous solutions for mobile radio networks. Data pipeline aims at delivering data from various sources to a destination in a form optimized for reading, querying, decision making and building data products.

Rationality of decisions and quality of data products are defined by the quality of the data pipeline. Quality monitoring system, currently used in case company, provides means for ensuring quality of the data pipeline and data products. However, with development of the product and gaining more experience from implementing the data pipeline into various environments, it was revealed that the system allows some flaws to pass unnoticed. This thesis aimed at highlighting insufficiently observed stages of the data pipeline and proposed key components of quality management system to implement.

The study was performed using action research methodology and represents applied research, conducted based on qualitative data collection and analysis. As a result of the current state analysis, it was discovered that some of the stages in data processing are extremely hard to monitor, in case of an error event it is hard to troubleshoot the issue and to notice it on time before any harm is done. This is especially important due to the nature of the applications that are built using processed data: applications work in a closed loop, any inconsistency in the data might lead to dramatic changes in RANs and quality degradation of customers' mobile subscribers.

To approach the challenge, the theoretical framework of the thesis discussed the main components of data pipeline quality management system, such as notifications rules, quality statements, metadata storage, audit storage, incident management system, and quality monitoring system.

The thesis focused on observability problem of a data handling process and proposed the main components that are required for troubleshooting, monitoring, and understanding the health of the data and the pipeline itself. As the data pipeline is a quite complex system, therefore various experts are required to develop and maintain it.

The following was proposed in this thesis based on the investigation, literature review and co-development with the stakeholders:

First, create and maintain detailed documentation of quality management system. Otherwise, it is difficult to understand how reliable it is and to assess the quality of the system.

Second, introduce and incorporate concept of metadata, which will help to understand health of the data on every stage of data pipeline lifecycle.

Third, implement checks and verification steps for those stages of data pipeline, where they are missing.

Fourth, reassess actions in case of error event and target group which receive a notification about the error event.

Fifth, hire a data warehouse quality engineering manager or assign this role to one of the existing members of the team, who will be managing all the above.

Thus, the thesis focused on observability problem of a data handling process and proposed the main components that are required for troubleshooting, monitoring, and understanding the health of the data and the pipeline itself. Addressing observability problem and improving means for data pipeline quality monitoring will allow to ensure accuracy of data products and provide solid ground for decision making.

7.2 Evaluation of the Thesis

The thesis implies repetitive approach (plan -> implement -> evaluate), therefore it is difficult to explicitly evaluate the thesis work outcome. Considering, that most of the proposals are based on best practices, it is unlikely they can be called irrelevant or not useful.

Significant part of the thesis work was devoted to existing knowledge gathering. It was needed to gain an understanding on best practices and recommendations. Data engineering pipeline and data warehouses are rather widespread topics and there are a lot of information available on that regard. Besides, the best practices can be applied to any data pipeline no matter what industry it is being used for.

During proposals validation, it was stated that the thesis lacks software and technology stack assessment. For a thesis to be applicable, this is, of course, important. However, agreeing on the framework and the direction to move to is also valuable. Therefore, it seems sufficient to limit the scope of the thesis to framework creation.

Section 1 outlined that the outcome of the thesis will include: methods for quality assurance in data engineering pipeline; recommendations for enhancing data quality monitoring capabilities; propose incident management process. It can be stated that declared outcome has been achieved. For example, various checks and test were discussed (historical balance test, statistical tests, attribute analysis, functional dependency, and referential analysis) and recommended for those stages of the data pipeline, where they can be applied. To enhance the monitoring capabilities, the concept of metadata was introduced, and a sample overview dashboard was presented. Incident management was discussed in terms of introducing additional target groups for receiving notifications in case of error events. Besides, severity level of error event was proposed with specific consequent actions.

7.2.1 Final Words

The research and development conducted in this thesis was based on extensive review of existing knowledge and best practices. Studied literature is full of smart but scattered details that are difficult to comprehend at the first glance. Interestingly, some of the best practices can be fully understood only after trying another approach and facing an issue. That happened with the case described in Section 6, when Data profiling for enriched stage of data pipeline was proposed to be implemented in a form of an application (on application stage of a data pipeline).

Action research approach (plan - implement - evaluate) proved to be perfect for solving the selected business challenge of the case company. It should be mentioned here that

the work performed in the thesis, research approach and proposals are aligned with ISO 9001:2015 standard on Quality Management System (QMS). The action research approach applied in the thesis (see section 2) is also aligned with PDCA (plan-do-check-act) cycle discussed in the standard.

Implementation of the proposed ideas will require leadership, commitment, and allocation of resources. To reach the final goal in a form of centralized storage of metadata and an overview dashboard presented in Section 5.2.3, a lot of planning and design work should be conducted. However, based on obtained understanding of the best practices this approach is commonly used and is inevitable part of trustworthy data pipeline solution of scale.

References

Altexsoft. (2021). *Data Engineering and Its Main Concepts: Explaining the Data Pipeline, Data Warehouse, and Data Engineer Role*. Retrieved April 2022, from https://www.altexsoft.com/blog/datascience/what-is-data-engineering-explaining-data-pipeline-data-warehouse-and-data-engineer-role/

AWS. (n.d.). *What is a data lake?* Retrieved March 2022, from https://aws.amazon.com/big-data/datalakes-and-analytics/what-is-a-data-lake/

Bergh, C., Benghiat, G., & Strod, E. (2019). *Tha DataOps Cookbook.* Retrieved 2022, from https://31m79x47w6ko3tyiwd35dmbf-wpengine.netdna-ssl.com/wp-content/uploads/2020/11/DK_dataops_book_2nd_edition.pdf

Chandrasekar, A. T. (2021, October). Telco Data Analytics using Open-Souce Data Pipeline: Detailed Architecture and Technology Stack. *International Journal for Research in Applied Science and Engineering Technology, 9*(X). Retrieved April 2022, from https://www.academia.edu/67596892/Telco_Data_Analytics_using_Open_Source_Data_Pipeline_Detailed_Architecture_and_Technology_Stack

CloverDX. (2017). *A Data Vault, Warehouse, Lake, And Hub Explained*. Retrieved April 2022, from https://www.cloverdx.com/blog/a-data-vault-warehouse-lake-and-hub-explained

Collins Dictionary. (2022, May). *framework.* Retrieved from Collins Dictionary: https://www.collinsdictionary.com/dictionary/english/framework

Coughlan, P., & Coghlan, D. (2002). Action research for operations management. *IJOPM, 22*(2), 220-240. Retrieved from http://www.dep.ufmg.br/old/disciplinas/epd804/artigo3.pdf

Databand. (2021). *Data observability: Everything you need to know*. Retrieved April 2022, from https://databand.ai/data-observability/#:~:text=%E2%80%9CData%20observability%E2%80%9D%20is%20the%20blanket,issues%20in%20near%20real%2Dtime.

Database.Guide. (2016). *What is an Orphaned Record?* Retrieved May 2022, from https://database.guide/what-is-an-orphaned-record/#:~:text=An%20orphaned%20record%20is%20a,to%20be%20an%20orphaned%20row.

Deloitte. (2017). *A network of networks.* Retrieved February 2022, from https://www2.deloitte.com/content/dam/Deloitte/tr/Documents/technology-media-telecommunications/DUP_Network-of-networks.pdf

Deloitte. (2019). *The Future of Extract, Transform & Load tools | Part I*. Retrieved April 2022, from https://www2.deloitte.com/nl/nl/pages/technology/articles/future-etl-extract-transform-load-big-data.html

Deloitte. (2020). *The Age of Telecom Network Automation.* Retrieved April 2022, from https://www2.deloitte.com/content/dam/Deloitte/pt/Documents/technology-media-telecommunications/PoV_Network%20Automation.pdf

Densmore, J. (2021). Common Data Pipeline Patterns. In *Data Pipelines Pocket Reference.* O'Reilly Media.

Dhingra, D. (2022). *Bus Factor in Software Engineering*. Retrieved May 2022, from https://medium.com/analytics-vidhya/bus-factor-in-software-engineering-d3316279ea41

Eckerson Group. (2018). *Data Observability - A Crucial Property in a DataOps World*. Retrieved May 2022, from https://www.eckerson.com/articles/data-observability-a-crucial-property-in-a-dataops-world

Eckerson Group. (2018). *The Complexities of Modern Data Pipelines*. Retrieved April 2022, from https://www.eckerson.com/articles/the-complexities-of-modern-data-pipelines

GSM Association. (2021). *The Mobile Economy Europe 2021.* Retrieved February 2022, from https://www.gsma.com/mobileeconomy/wp-content/uploads/2021/09/GSMA_ME_Europe_2021_R_Web_Singles.pdf

Helfert, M., & Herrman, C. (2002). *Proactive Data Quality Management for Data Wahehouse Systems*. Retrieved from http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-58/herrmann.pdf

Hinrichs, H., & Aden, T. (2001, June 4). An ISO 9001:2000 Compliant Qiality Management System for Data Intergation in Data Warehouse Systems. *Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW'2001)*.

IBM. (2021). *ETL (Extract, Transform, Load)*. Retrieved April 2022, from https://www.ibm.com/cloud/learn/etl

Kimball, R., & Caserta, J. (2004). Cleaning and Conforming. In *The data Warehouse ETL Toolkit: Practical techniques for Extracting, Cleaning, Conforming, and Delivaring Data.* Wiley.

Kimball, R., & Caserta, J. (2004). Metadata. In *The Data Warehouse ETL Toolkit: Practical Techniques for Extracting, Cleaning, Conforming and Delivering Data.* Wiley.

Kimball, R., & Caserta, J. (2004). Requirements, Realities and Architecture. In *The Daata Warehouse ETL Toolkit: Practical Techniques for Extracting, Cleaning, Conforming and Delivering Data.* Wiley.

Kimball, R., & Caserta, J. (2004). What the Data Warehouse is. In *The Data Warehouse ETL Toolkit: Practical Techniques for Extracting, Cleaningm Conforming and Delivering Data.* Wiley. Retrieved from https://learning.oreilly.com/library/view/the-data-warehouse/9780764567575/ch01.html#what_the_data_warehouse_is

Kothari, C. (2004). *Research Methodology. Methods and Techniques.* New Delhi: New Age International (P) Ltd. Retrieved April 2022, from https://research-methodology.net/research-methodology/research-types/applied-research/#_ftn1

Leedy, P. D., & Ormrod, J. E. (2015). *Practical Research. Planning and Design.* (11th ed.). Harlow: Pearson Education Limited. Retrieved from https://pce-fet.com/common/library/books/51/2590_%5BPaul_D._Leedy,_Jeanne_Ellis_Ormrod%5D_Practical_Res(b-ok.org).pdf

Linstedit, D., & Olschimke, M. (2015). The Enterprise Data Warehouse Environment. In *Building a Scalable Data Warehouse with Data Vault 2.0.* Morgan Kaufmann.

Loome. (n.d.). *CI/CD for Data Watehousing and Analytics*. Retrieved April 2022, from https://www.loomesoftware.com/ci-cd-for-data-warehousing

Lu, L., & Özsu, M. T. (2009). *Encyclopedia of Database Systems.* Boston, MA: Springer.

McKinsey. (2017, February 13). *A future for mobile operators: The keys to successful reinvention.* Retrieved February 2022, from https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/a-future-for-mobile-operators-the-keys-to-successful-reinvention

Merriam-Webster. (2015). *Metadata*. Retrieved April 2022, from https://www.merriam-webster.com/dictionary/metadata

Microsoft. (2021). *Online analytical processing (OLAP)*. Retrieved March 2022, from https://docs.microsoft.com/en-us/azure/architecture/data-guide/relational-data/online-analytical-processing

Microsoft. (2022). *Big data architecture style*. Retrieved March 2022, from https://docs.microsoft.com/en-us/azure/architecture/guide/architecture-styles/big-data

Moses, B., Gavish, L., & Vorweck, M. (2022). Fixing Data Quality Issues at Scale. In *Data Qiality Fundamentals.* O'Reilly Media, Inc.

Moses, B., Gavish, L., & Vorwerck, M. (2022). What Is Data Quality? In *Data Quality Fundamentals.* O'Reilly Media, Inc. Retrieved from https://learning.oreilly.com/library/view/data-quality-fundamentals/9781098112035/

N-iX. (2019). *Big data and predictive analytics for data-driven insights in telecom*. Retrieved April 2022, from https://www.n-ix.com/telecom-analytics-big-data-predictive-analytics/

Oracle. (2009). *Warehouse Builder User's Guide. Understanding Data Quality Management.* Retrieved May 2022, from https://docs.oracle.com/cd/B31080_01/doc/owb.102/b28223/concept_data_quality.htm#BGBEFJCA

Ponniah, P. (2001). Business Metadata. In *DATA WAREHOUSING FUNDAMENTALS: A Comprehensive Guide for IT Professionals.* Wiley-Interscience.

Synopsys. (2018). *What's the difference between agile, CI/CD, and DevOps?* Retrieved April 2022, from https://www.synopsys.com/blogs/software-security/agile-cicd-devops-difference/

Technopedia. (2020). *Semantic Data Model*. Retrieved March 2022, from https://www.techopedia.com/definition/30489/semantic-data-model

TMForum. (2019, May). *Autonomous Networks: Empowering Digital Transformation For the Telecoms Industry.* Retrieved February 2022, from https://www.tmforum.org/wp-content/uploads/2019/05/22553-Autonomous-Networks-whitepaper.pdf

Vaughan, J. (2019). *Data Quality*. Retrieved March 2022, from https://www.techtarget.com/searchdatamanagement/definition/data-quality

Williams, C. (2007, March). Research Methods. *Journal of Business and Economic Research, 5*(3), 65-72. Retrieved from https://clutejournals.com/index.php/JBER/article/view/2532/2578

XENONSTACK. (2021). *Data Ingestion: Pipeline, Architecture, Tools, Challenges*. Retrieved March 2022, from https://www.xenonstack.com/blog/big-data-ingestion