Data pipeline monitoring solution and data quality in manufacturing company

Lappeenranta–Lahti University of Technology LUT
Master's Thesis Industrial Engineering and Management
2024
Riku Mattila
Examiner(s): Post-doctoral researcher, docent Lasse Metso, D.Sc. (Tech.)
Post-doctoral researcher, Antti Ylä-Kujala, D.Sc. (Tech.)

ABSTRACT

Lappeenranta–Lahti University of Technology LUT
LUT School of Engineering Sciences
Industrial Engineering and Management

Riku Mattila

Data pipeline monitoring solution and data quality in manufacturing company

Master’s thesis
2024
97 Pages, 19 figures, 12 tables, and 3 appendices
Examiners Post-doctoral researcher, docent Lasse Metso, D.Sc. (Tech.) and Post-doctoral researcher, Antti Ylä-Kujala, D.Sc. (Tech.)
Keywords: Data pipeline monitoring, Data pipeline, Data quality, Data completeness, Real-time monitoring, Error analysis, Design science research

Technology developments have rapidly changed manufacturing in recent years making data a paramount asset. Data pipelines are vital in collecting, transferring, and processing data for ever-increasing business-critical applications. Data quality also plays a crucial role in the effectiveness and efficiency of decision-making processes within organizations, particularly in the manufacturing sector. Companies still face challenges with both hindering the potential of data. The thesis aims to identify common data quality challenges in a global manufacturing company and examine their possible causes. Additionally, it explores the design and development of an automated monitoring solution to address potential data incompleteness issues with the existing data pipeline.

The thesis study is conducted as design science research and consists of a literature review, expert interviews, and empirical research describing the design process. The literature review mainly examines data pipelines, quality, and anomaly detection. The empirical research investigated common data quality problems and the development process for implementing automated data pipeline monitoring. The development process investigates multiple solution possibilities and applicable designs to monitor data loss and completeness of records. The study also composes a general framework deriving knowledge from literature and empirical research.

The study concludes that common data quality problems in the company are outdated data, discrepancies between source systems, missing data, and human-induced errors. The company's probable cause for these data quality problems is lacking data governance. The thesis study develops a customized automated monitoring solution that fits into the existing data pipeline of the company increasing visibility and trust. This solution involves using data from logging mechanisms to establish real-time monitoring, error analysis, and alerting mechanisms notifying relevant stakeholders about potential data loss or anomalies in data pipeline performance. Also, the study introduces a general framework for developing data monitoring to an existing pipeline contributing to the knowledge base. In the future, the general framework can be used by the company and other practitioners to guide pipeline monitoring.

TIIVISTELMÄ
Lappeenrannan–Lahden teknillinen yliopisto LUT
LUT insinööritieteiden tiedekunta
Tuotantotalous

Riku Mattila

Dataputkien monitorointiratkaisu ja datan laatu teollisuusyrityksessä

Tuotantotalouden diplomityö
2024
97 sivua, 19 kuvaa, 12 taulukkoa ja 3 liitettä
Tarkastajat: Tutkijatohtori, dosentti Lasse Metso ja Tutkijatohtori Antti Ylä-Kujala
Avainsanat: Dataputkien monitorointi, Dataputki, Datan laatu, Datan kokonaisuus, Reaaliaikainen monitorointi, Virheanalyysi, Kehittämistutkimus

Teknologian kehitys on muuttanut teollisuutta nopeasti viime aikoina tehden datasta ensisijaisen tärkeän resurssin. Dataputket ovat elintärkeitä datan keräämisessä, siirtämisessä ja käsittelemisessä lisääntyvälle määrälle kriittisiä sovelluksia. Yritystoiminnassa datan laatu on keskeisessä roolissa päätöksentekoprosessien tehokkuudessa ja toimivuudessa. Yritykset kohtaavat edelleen haasteita datan laadussa estäen täyden datan potentiaalin hyödyntämisen. Tämä opinnäytetyö pyrkii tunnistamaan yleiset datan laatuun liittyvät haasteet globaalissa teollisuusyrityksessä ja tutkimaan niiden mahdollisia syitä. Lisäksi työ tarkastelee automatisoidun monitorointiratkaisun suunnittelua ja kehittämistä toimittajien ja yrityksen väliseen dataputkeen mahdollisten datan puutteiden ratkaisemiseksi.

Opinnäytetyö toteutetaan suunnittelutieteellisenä (DSR) kehittämistutkimuksena ja se koostuu kirjallisuuskatsauksesta, asiantuntijahaastatteluista ja empiirisestä tutkimuksesta, joka kuvaa suunnitteluprosessia. Kirjallisuuskatsaus tarkastelee pääasiassa dataputkia, datan laatua ja anomaliatunnistusta. Empiirisessä osuudessa tutkitaan yleisiä datan laatuun liittyviä ongelmia ja kehitysprosessia automatisoidun dataputken monitoroinnin toteuttamiseksi. Kehitysprosessi esittelee ja arvioi useita ratkaisuvaihtoehtoja ja suunnitelmia datan katoamisen ja tietueiden kokonaisuuden monitorointiin. Tutkimuksessa laaditaan kirjallisuudesta ja empiirisestä tutkimuksesta johdettu viitekehys.

Tutkimuksessa todettiin, että yrityksen yleisiä datan laatuun liittyvät ongelmia on vanhentunut data, lähteiden väliset ristiriitaisuudet, puuttuva data ja ihmisen aiheuttamat virheet. Todennäköinen syy löydettyihin haasteisiin datan laadussa on puutteellinen datanhallinta, jonka tulisi standardoida ja ohjeistaa dataan liittyviä prosesseja. Opinnäytetyössä kehitetään myös räätälöity ja automatisoitu monitorointiratkaisu yrityksen tämänhetkiseen dataputkeen lisäten käyttäjien näkyvyyttä ja luottamusta dataan. Ratkaisussa hyödynnetään virheenkirjaamismekanismin kirjaamaa dataa reaaliaikaiseen valvontaan, virheanalyysin ja hälytysmekanismien luomiseen. Hälytykset ilmoittavat asiaankuuluville sidosryhmille mahdollisista datan menetyksistä tai epätavallisuuksista dataputken suorituskyvyssä. Lisäksi tutkimus kokoaa viitekehyksen datan monitoroinnin lisäämisestä olemassa olevaan putkeen, mikä osaltaan tuo kirjallisuuteen uuden käytännönläheisen näkökulmaan. Tulevaisuudessa yritys ja muut toimijat voivat käyttää viitekehystä ohjaamaan dataputkien valvonnan lisäämistä.

ACKNOWLEDGEMENTS

I want to take this moment to thank a few people who have made this thesis possible, supported me, and helped to succeed during this journey. First, my biggest thanks go to the commissioning company and the supportive and inspiring work atmosphere there.

Being part of such a creative and open work community has been a pleasure. Special thanks go to my thesis supervisor: Thank you Vasily for everything! Especially when my ideas kept changing and evolving. Every conversation we had strived and motivated me to work harder. I must also acknowledge my supervisor Klaus for his unconditional support, an example of leadership, and the positive aura that surrounds him.

Next, I would like to thank both Lasse and Antti for their insightful comments and for committing time and ideas to this thesis. At every meeting with them, I had plenty of questions, and after meetings more than enough new ideas and improvements.

Finally, I would like to deeply thank my close ones for supporting and motivating me unconditionally throughout my life. I simply cannot complain about being in a situation where my spouse, family, and friends have been sparring, supporting, and pushing me to succeed throughout my studies and wishing me the best. While this thesis marks the end of my time in Lappeenranta, I am sure that the friends and memories made along the way will last the rest of my days.

Riku Mattila
In Helsinki 9.6.2024

TABLE OF CONTENTS

ABSTRACT

ACKNOWLEDGEMENTS

1 INTRODUCTION 8

1.1 BACKGROUND 8

1.2 RESEARCH PROBLEM AND SCOPE 9

1.3 RESEARCH METHODS AND DATA COLLECTION 11

1.4 STRUCTURE 12

2 MANY CONCEPTS OF DATA: MANAGEMENT, PIPELINES, QUALITY, AND ANOMALIES 15

2.1 COMPASSING DATA: DATA MANAGEMENT AND DATA GOVERNANCE 15

2.2 PREVIOUS RESEARCH ON DATA PIPELINES 20

2.3 WHAT IS DATA QUALITY? 27

2.4 SUMMARY OF ANOMALY DETECTION 34

3 THESIS METHODOLOGY AND CONDUCTED RESEARCH PROCESS 37

3.1 APPROACH TO INTERVIEWS AND BASICS OF DESIGN SCIENCE RESEARCH 37

3.2 DESIGN SCIENCE RESEARCH PROCESS IN GENERAL 40

3.3 CONDUCTED DSR PROCESS 43

4 CASE DESCRIPTION OF THE COMPANY AND DATA QUALITY PICTURE IN THE COMPANY 45

4.1 BACKGROUND INFORMATION OF THE CASE COMPANY 45

4.2 DEFINING OBJECTIVES FOR THE MONITORING SOLUTION 48

4.3 INTERVIEW RESULTS 50

5 DESIGNING, DEVELOPING, AND EVALUATING DATA PIPELINE INGESTION MONITORING SOLUTION 54

5.1 DESIGNING THE DATA PIPELINE INGESTION MONITORING SOLUTION 54

5.2 A GENERAL FRAMEWORK FOR IMPLEMENTING DATA PIPELINE MONITORING 64

5.3 DEVELOPING AND IMPLEMENTING THE DATA MONITORING SOLUTION 67

5.4 TESTING AND EVALUATION OF THE DATA PIPELINE MONITORING SOLUTION 77

6 RESULTS OF DATA QUALITY AND DATA PIPELINE MONITORING IN MANUFACTURING COMPANY...80

6.1 RESULTS OF DATA QUALITY ASSESSMENT AND OVERALL DATA QUALITY PICTURE IN THE COMPANY 81

6.2 RESULTS OF DATA PIPELINE MONITORING SOLUTION AND GENERAL FRAMEWORK .82

7 CONCLUSIONS...85

REFERENCES: 91

APPENDIX 1. INTERVIEW GUIDE..1

APPENDIX 2. FILE EVENT TABLE EXAMPLE (GENERATED).............................2

APPENDIX 3. DATA QUALITY ASSESSMENT PANDAS SCRIPTS 3

FIGURES:

Figure 1: Structure and Progress of Thesis 13
Figure 2: Data Governance framework (DAMA International, 2017) 18
Figure 3: Data Governance Model (Zorrilla & Yebenes, 2022) 19
Figure 4: General model of Data Pipelines (Foidl et al., 2024) 21
Figure 5: Data Quality Dimensions (Albrecht et al., 2023; Fan, W., 2015; Fan, Wenfei & Geerts, 2022) 28
Figure 6: Data Quality process model (Batini et al., 2009) 30
Figure 7: DQA Framework Cai and Zhu (2015) 31
Figure 8: Data Quality Assessment Process (Cai & Zhu, 2015) 32
Figure 9: DSR Knowledge Contribution Framework (Gregor & Hevner, 2013) 39
Figure 10: DSR Process (Peffers et al., 2007) 41
Figure 11: Followed DSR Process 43
Figure 12: Illustration of Data Pipeline 47
Figure 13: General Data Pipeline Monitoring Implementation Framework 64
Figure 14: Solution Components 68
Figure 15: Data Model 69
Figure 16: Supplier X Dashboard 70
Figure 17: Error Percentage by Date 72
Figure 18: Weekly Completeness Percentage 73
Figure 19: Traffic lights (Real time monitoring) 75

TABLES

Table 1: Ingestion Phase in Manufacturing 23
Table 2: Pre-Processing Phase Steps and Techniques 24
Table 3: Data Pipeline Anomalies/Faults Adopted from Aiswarya et al, (2020b) 25
Table 4: Data Pipeline Quality Influencing Factors Foidl et al. (2024) 26
Table 5: DSR Guidelines (Hevner et al., 2004) 40
Table 6: Interview Results, Common Data Quality Problems 51
Table 8: Solution Concept Designs 56
Table 9: Functional Requirements 60
Table 10: Non-Functional Requirements 61
Table 11: Possible Data Sources 61
Table 12: Data Quality Assessment Results 81

1 Introduction

The first chapter is designed to give the reader background information and research context to better understand the investigated field and thesis process. The outcome of this chapter is an outline of the background, research problems, scope, research methods, and structure of the rest of the thesis.

1.1 Background

Changes in competitive space, personnel, and the current economic climate have pushed organizations to develop data-driven solutions and capabilities for a possible source of competitive advantage. The reliability, integrity, and accuracy of data-driven solutions are connected to underlying data quality. Focusing efforts on generating higher-quality data should be a key objective for companies. Data quality issues or missing data are more than likely to occur with severity depending on the use case and mitigation maturity. Low-quality data may lead to ill-advised decision-making and loss of information or profit. Many factors contribute to the quality of data ranging from management to data architecture. Data management should outline and manage the high-level picture of data and affect the overall data quality. Data management is a vast collection of processes, concepts, and practices that aim to leverage data assets (DAMA International, 2017). Data governance is one of those practices that guide organization data policies and standards.

Another example of a concept under data management is data pipeline. Data pipelines in Industry 4.0 drive productivity and profit by collecting, processing, and supplying data for other pivotal applications. Data pipelines play a vital role in modern data-driven systems maintaining the flowing data from systems to use. Automation and robust monitoring are essential to reduce faults and improve performance. Challenges in data pipeline performance can occur from infrastructural, organizational, and data quality issues. Foidl et al. (2024) and Munappy et al. (2020) have both demonstrated the data quality issues and the importance of monitoring the data pipeline and have created foundations for further research. The literature is missing practical designs and cases where monitoring is implemented. Possible explanations for the absence of studies can be the extensive architectural research needed and confidentiality reasons.

The commissioning company is concerned about its data quality and potential loss of information. Although some of the company's information systems and data architecture can be considered state-of-the-art, these solutions are not distributed widely across the wider picture. Issues also arise with the compatibility of legacy systems regarding growing needs for data analytics, process mining, and near real-time information flows while also maintaining continuous data flow to the decision-making. The company has data pipelines from the sources to the destination built and ongoing. The company has also identified a data pipeline that is prone to errors and requires monitoring to be implemented. Some tools can be used to monitor incoming data manually. Highlighted problems and the need for monitoring are a result of the incompleteness of the end user records. The problem domain is complex since the data sources are not located on the commissioning company's premises. Therefore, simply observing the data generation at the source is not applicable.

1.2 Research Problem and Scope

The purpose of this thesis is to investigate the company's data quality issues and design an automated data pipeline monitoring solution. The design of the monitoring solution is focused on a specific supplier testing result pipeline that is known to have problems with the completeness of data. Incompleteness of data has been previously determined to be caused by failed file generation and followed by unsuccessful processing.

The main objective of the thesis is to create an automated solution for data pipeline monitoring in an existing and limited visibility data pipeline and identify data quality-related challenges. Limited visibility in this context refers to the fact that the data pipeline is already established and working, but many parts throughout the data pipeline are inaccessible. The monitoring solution aims to increase the data pipeline visibility and consequently improve the end-user reporting and decision support systems involved. The research problems and objectives can be summarized by the following research questions for the thesis work:

What are the main data quality issues identified in the company and possible causes for the issues?

How can an automated monitoring solution be designed to fit into practice within the existing data pipeline?

What is the general framework for implementing monitoring solutions to an existing pipeline?

The first research question assesses data quality in the company and investigates the potential root causes for these issues. The organization's desired outcomes of the data quality study are a map of the organization's current data quality issues and determining likely causes for those issues. Where data quality can be assessed based on assessing specific datasets more general picture is investigated with the connections to related topics. Other data quality-related topics are data governance and data management as they should create an overall picture and standards for the organizations' data efforts. Data quality and data quality management can be both considered to be part of the wider picture of data management.

The second research question is intertwined with the first one. The outcome artifact of the second research question is designed for monitoring one of the core data quality dimensions completeness alongside other typical monitoring features like real-time monitoring and error diagnosis. For the organization, the main outcome of the artifact is to monitor data ingestion to the data pipeline to catch potential disruptions and assess the completeness of the processed data. Data pipelines have been identified in previous studies to affect data quality. Data processed in the pipeline relates to the organizations' supplier data and product traceability and therefore the full final solution is not going to be available publicly but will be represented clearly and concisely with examples of the components.

The third research question is related to the second one the general objective is to create a generalized framework based on the practical and theoretical knowledge gathered during the design and development of the solution. The generalized framework will exemplify the general process flow and consider the necessary topics for practical monitoring solution implementation. The framework will contribute to the data engineering field of study as an extension of practical and theoretical knowledge.

Data quality scope is limited to practical assessment and cause analysis from the specific supplier pipeline and reassures the general picture analysis with interviews. The company has thousands of different data sources and hundreds of thousands of datasets. Therefore, the need to limit the datasets assessed must be addressed. Since the rest of the thesis generally focuses on a specific data pipeline and few datasets, they are naturally interesting to assess. The scope of the data quality investigation was also set on a specific division inside the company rather than the whole global company.

The scope of the artifact or pipeline monitoring solution in this thesis refers only to monitoring and assessment of the pipeline's performance due to the limitations of the scale of the thesis. Data monitoring solution is not implemented throughout the data pipeline to each phase, but only to the ingestion phase although the completeness of end-user data is also assessed. Through literature review and interviews a suitable solution approach is constructed that considers the limitations of organizations' data governance, information systems, data architecture, and platforms.

1.3 Research methods and data collection

The thesis is executed as Design Science research to establish a practical and customized data pipeline monitoring solution. The thesis also employs qualitative methods like semi-structured interviews to understand the contextual data quality picture and requirements more in-depth. Following the guidelines of Design Science Research (DSR), a practical and usable artifact is constructed in a real-use case to demonstrate its applicability in a real-world scenario. More detailed DSR research methodology for the thesis is reviewed in a separate methodology chapter.

The artifact objectives, requirements, and baseline for conceptual solutions are gathered and synthesized from frameworks, best practices, and process models explained in detail in the literature review. Scientific rigor and understanding are combined with the expertise of the company and practical constraints to construct a feasible artifact. Monitoring methods and suggestions are critically reviewed and compared to the real world to construct an approach that would fulfill the objectives set for the artifact. The output of the examination of literature, interviews, and organizational resources is a list of possible solutions.

These solution options are then developed and designed further resulting in the final artifact that improves the company's visibility and demonstrates possible practical solutions for similar situations. The artifact itself is built to use existing data sources from the data pipeline rather than integrating them into the pipeline. Suggestions and other minor improvements that may solve the data quality challenges in the organization are proposed based on the interview and empirical data. DSR artifacts will be tested in practice and refined based on continuous feedback. The final solution will be evaluated based on key requirements and objectives.

Data collected and used in this thesis is mainly structured SQL data that is formatted in a standardized way. Other data sources for the more abstract problems of general data quality and complex system architecture are the experts interviewed during the thesis. Data collection methods used for expert interviews are semi-structured interviews and meetings. Experts were selected based on their perceived expertise and closeness to the problem domain. Other data sources like internal architecture documentation and process models are used but cannot be stated fully in this thesis documentation to ensure confidentiality.

1.4 Structure

The thesis follows the common DSR publication schema with each chapter linked to others with inputs and outputs. First in the introduction chapter key concepts and research questions are introduced as well as highlighting the theoretical and practical importance of the study. After the introduction chapter, scientific rigor is demonstrated with the literature review of topics related to the thesis.

Literature should constitute an outline for the rest of the study and artifact development. The topics covered by the literature review are basic data concepts, data management, data pipelines, data quality, and anomaly detection. These literature topics are related and intertwined, and a combination of the topics constitutes better requirements for the artifact and wider theoretical understanding.

After the literature DSR methodology is reviewed in detail and highlighting the dominant approaches thesis progresses towards a practical solution. Continuing with the solution, a case description of the company is illustrated, and semi-structured interviews are conducted to better understand the underlying context and limitations. Based on the interviews, context, and literature initial artifact objectives are defined. Objectives guide the design process and different design options are considered. In the design chapter, the design is developed into an actual practical solution using common data analysis techniques, key performance indicators, and data from the logging mechanism.

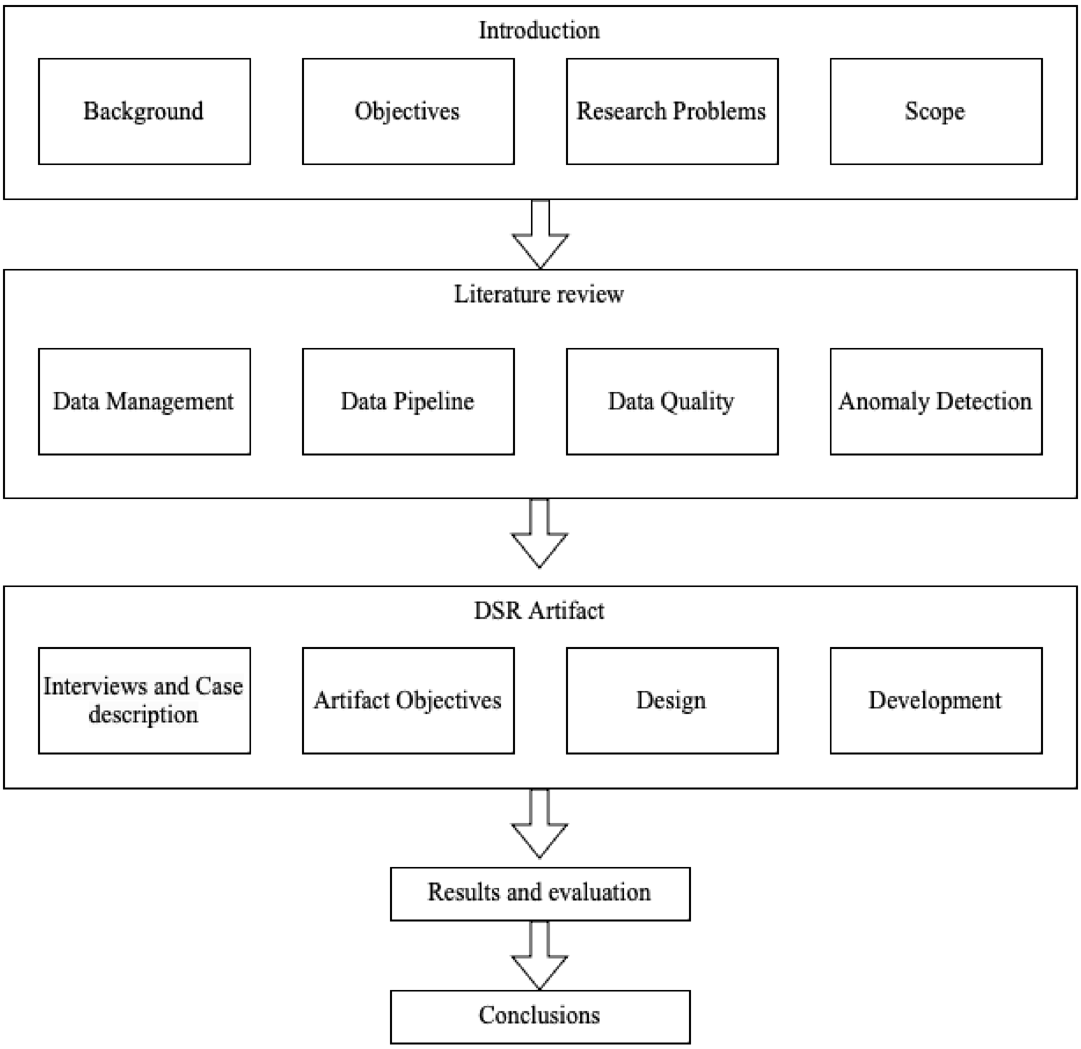

Figure 1: Structure and Progress of Thesis

In the final chapters results of the analysis are combined and the final artifact is evaluated on pre-determined conditions. The thesis structure is represented in Figure 1. In the results and evaluations chapter interview results, data quality assessment, and empirical observations regarding data quality issues are composed of a description of the current state with possible causes and strategies to improve. The developed artifact is reviewed based on the requirements and fulfillment of objectives. Conclusions provide summarized findings of the thesis and illustration of future work. Limitations of the thesis are also reviewed.

2 Many concepts of data: management, pipelines, quality, and anomalies

Manufacturing is rapidly evolving through digitalization and the fourth industrial revolution. Rapidly growing amounts of data generated through a variety of sources and growing needs to utilize the gathered data to create competitive advantage are present in manufacturing. Among many industry-relevant data-intensive applications like predictive maintenance, there exists a need for the constant flow of data. The first sub-chapter briefly introduces a common definition of data and defines and reviews data management and data governance. Later sub-chapters introduce, define, and review data pipeline and data quality theory, practices, and frameworks found in relevant literature.

2.1 Compassing data: Data Management and Data Governance

Data as a concept can be defined as many individual fact-based crumbs of information (Gupta & George, 2016). They can be statistics or measurements from different systems or sources and as the pieces are combined and rearranged information or knowledge can be acquired. Data can be divided into either structured or unstructured data. Structured data is more defined and stored in a set manner whereas unstructured data is less organized and varied. (Gupta & George, 2016; Ottonicar et al., 2019)

Metadata is often defined as data about data according to Perlin (2006) and Mayernik (2020). Metadata is essential in data management and efficient processing of large quantities of data. Rather than processing through a whole database, a search algorithm or analyst can find important pieces of information by simply using the metadata. The purpose of metadata is to help users search, process, and use data. (Perlin, 2006) Metadata can be defined as structured data about an object, context provider, retrieval promoter, or statement (Mayernik, 2020).

Data management is a concept that manages data-related activities and various sub-elements such as data quality, maintenance, and integration. Data management involves ensuring the availability of the right data in the right format at the right time. Effective data management practices are essential for organizations looking to gain a competitive edge. (O'Donovan et al., 2015) These practices involve establishing processes and following policies set by data governance that enable organizations to collect, store, retrieve, analyze, and utilize data in a secure, accurate, and compliant way (Firmani et al., 2021). Data management in the manufacturing industry, particularly in the context of Industry 4.0, involves the collection, storage, and analysis of large volumes of industrial data (Firmani et al., 2021; Raptis et al., 2019). Rather than just being a side-product or metrics of processes, the role of data changes to an asset. The new role of data can potentially be as an integrator between the real and digital world (Raptis et al., 2019). Industry 4.0 is characterized by the rise of advanced technologies such as the Internet of Things (IoT), big data, and artificial intelligence, which have revolutionized manufacturing processes (Tambare et al., 2021).

Much of the literature follows guidelines and definitions set by the Data Management Association in their book Data Management Association- Data Management Body of Knowledge, DAMA-DMBOK for short. Data management is defined as: "development, execution, and supervision of plans, policies, programs, and practices that control, protect, deliver, and enhance the value of data and information assets" (DAMA International, 2017). Similarly, Carretero et al. (2017) express data management's role as implementor, and maintainer of data infrastructure based on business needs and requirements while business rules are actively maintained through data management policies. They also highlight business rules including data quality requirements and data quality management. Organizations have vast amounts of data from heterogeneous sources such as customer interactions, financial transactions, and operational metrics and the management of this data is crucial for the efficient and flexible operation of manufacturing processes (Firmani et al., 2021).

Successful data management requires a strategic approach that goes beyond simply meeting regulatory requirements like ISO9001 data quality standards. It involves aligning data practices with business goals, such as improving customer experience, streamlining operations, and identifying new revenue streams (Firmani et al., 2021). This strategic alignment ensures that data is not just managed but utilized as an asset to drive innovation and informed decision-making. In addition to strategic alignment, data management in Industry 4.0 also involves staying informed of changing technology and data governance best practices according to Koltay (2016). From a data engineering perspective, the integration of structured data management and Industry 4.0 is crucial for the success of this new industrial era (Nasution, 2021). As new data sources and technologies emerge, organizations must continuously adapt their data management strategies to ensure that they remain agile and responsive to change.

Data management in Industry 4.0 goes far beyond the traditional understanding of data maintenance (Nasution, 2021). It requires a holistic, strategic, and future-minded approach that encompasses other related activities like data governance, analytics, and decision-making systems with business objectives (O'Donovan et al., 2015). By embracing these principles, organizations are a step closer to transforming data from a passive resource into a strategic asset or proactive driver of success.

Data governance on the other hand involves establishing policies and standards to ensure the availability, quality, and security of organizational data (Carretero et al., 2017). Seeing data as an asset is one of the central topics that Data governance frameworks propose, and data governance has an important role in establishing policies and standards for the overall data domain (Abraham et al., 2019). Policies and standards are turned into implementable and business objective-aligned strategies (Carretero et al., 2017). Data Governance term has two contexts of use: regulatory or business-related. Regulatory data governance is practiced by regulatory entities like the government or the European Union. A recent example of data governance in a regulatory context is the General Data Protection Regulation Directive better known as GDPR. This thesis will focus on business-related data governance that includes how internal, customer, and stakeholder data is managed. Data Governance approaches data from the point of a strategic asset that should be considered as any other asset that is in a key position for succeeding (Zorrilla & Yebenes, 2022).

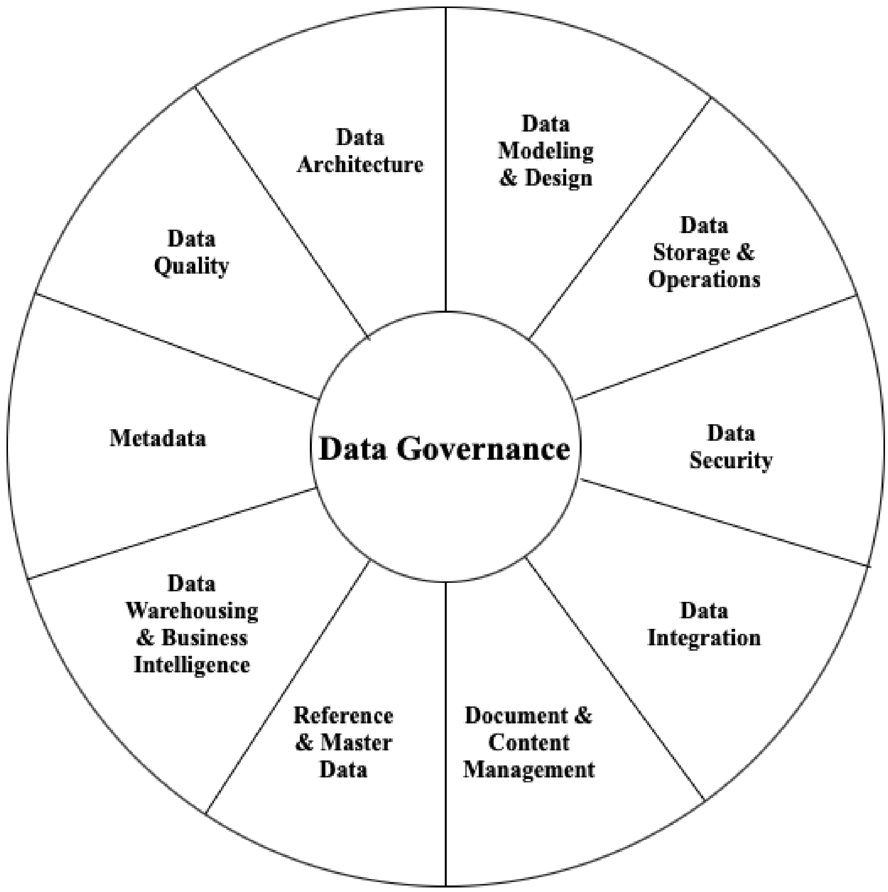

Figure 2: Data Governance framework (DAMA International, 2017)

The data governance framework created by DAMA International (2017) proposes ten different topics that data governance should cover. The framework in Figure 2 shows how holistically data governance should affect organizations' data-related practices. From data architecture to master data, everything related should follow the data governance policies and therefore enable value creation. Data management, data governance, and data quality are closely intertwined topics in the literature. Data management and data governance terms are used in different sources similarly, while the differences come from perspectives of view and context. Koltay (2016) emphasizes the role of data governance in decision-making and defining accountability. Al-Ruithe, Benkhelifa & Hameed (2019) summarize data governance as responsibilities, decision-making rights, and management of data as an asset and underscore the need for robust data governance practices to address the challenges of data complexity found in Industry 4.0. This complexity can emerge from the complex nature of the business and the multitude of legacy systems and custom in-house solutions.

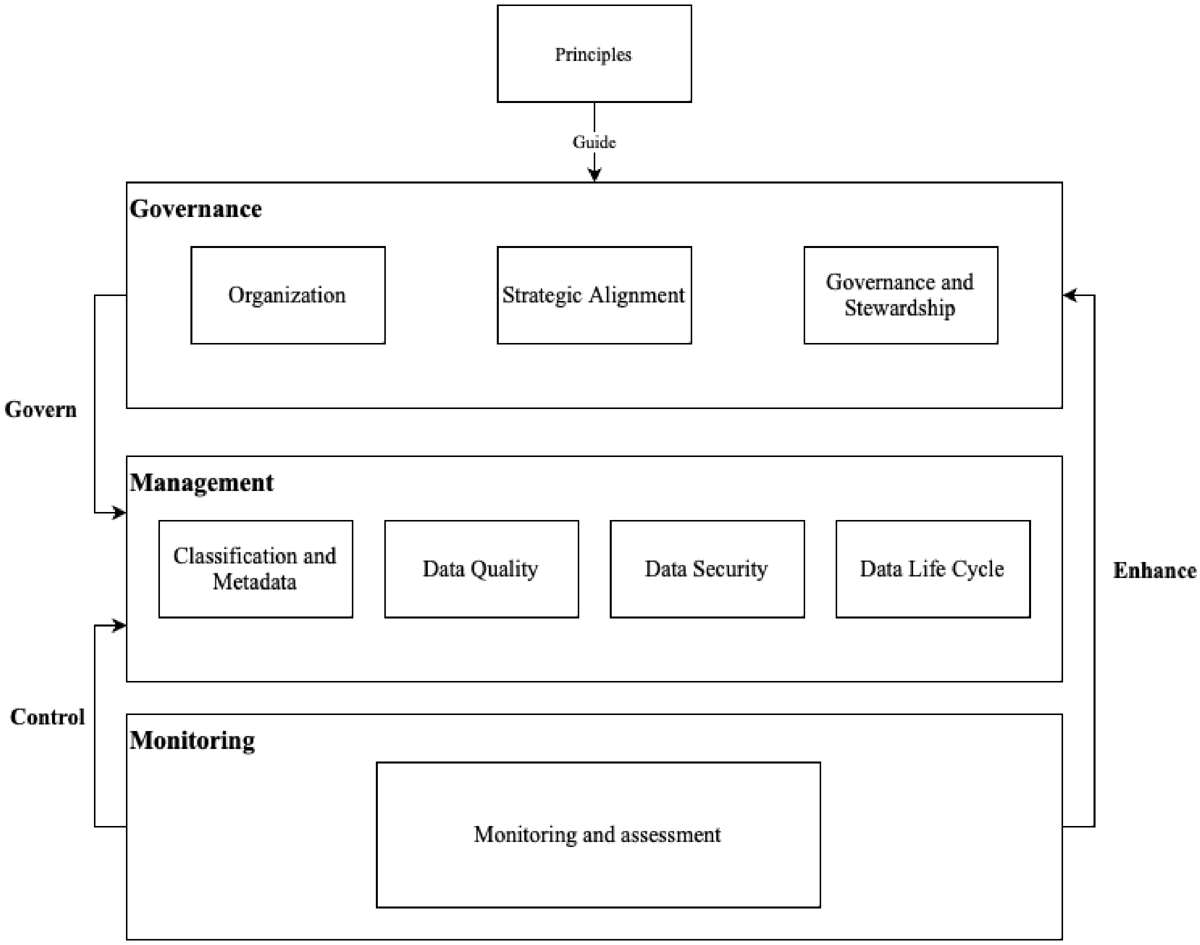

Figure 3: Data Governance Model (Zorrilla & Yebenes, 2022)

The Data Governance model provided by Zorrillas & Yebenes (2022) in Figure 3 includes similar topics as the one proposed by DAMA (2017) in Figure 2, but it defines relationships and divides topics into categories. Where the DAMA (2017) framework only considers technical aspects, Zorrillas & Yebenes's (2022) data governance model also introduces strategic alignment and organization to the picture. The focus on technical and change management is supported by Carretero et al. (2017) while their approach to data governance is notably more weighted towards data strategy than a technical perspective. The data governance model by Zorrillas & Yebenes (2022) shows that data governance should govern, and monitoring should control topics of metadata, data quality, data security, and data life cycle. The data life cycle is defined as the entire process from generation to destruction of data (Rahul & Banyal, 2020). The role of the data pipelines is as accelerators of data lifecycle activities by automating data processing and requiring less human intervention (Munappy et al., 2020).

2.2 Previous research on Data Pipelines

A data pipeline can be defined as a way of moving data from the source or sources to a destination (Munappy et al., 2020). The purpose of a data pipeline is to optimize and transform raw data to a suitable form for analysis and gaining insights or for different applications (O'Donovan et al., 2015). Due to being an automized and continuous process consisting of collecting, processing, and transferring data, they are integral parts of today's data-driven systems yet often unreliable (Foidl et al., 2024). A simple data pipeline architecture can consist of three phases: sources, processing, and destination (O'Donovan et al., 2015). Data Pipeline as a concept is relatively new, one of the earliest uses of the term in articles was only 12 years ago (Klievink et al., 2012). According to Plale & Kouper (2017) data pipeline can be defined as an abstraction that helps understand the data process across the data lifecycle. In literature one can find similar terms in other data-related fields, data science pipelines are an example of this. This thesis considers Machine Learning pipelines as different from Data pipelines, as the work of Biswas et al. (2022) indicates dissimilarity in nature while they do have some similarities.

Standard Extract-Transform-Load (ETL) and Extract-Load-Transform (ELT) processes can be seen as examples of data pipelines by definition (Aiswarya et al., 2020a). The distinction between ETL/ELT and the Data Pipeline concept is the level of abstraction. According to Foidl et al. (2024), data pipelines can contain more meta-level elements than a simple ETL process. Although their objective is similar the two concepts are not equal. Therefore, we consider ETL/ELT as a representation or a tool of data pipeline. Data pipelines can also contain many other frameworks for extracting and transforming data (Dehury et al., 2022).

In the manufacturing industry, O'Donovan et al. (2015) defined that data pipelines should conform to the following requirements: Legacy integration, Cross-network communication, Fault tolerance, Extensibility, Scalability, Openness, and accessibility. The robust structure of data pipelines in the manufacturing industry is crucial for the transition to Industry 4.0. Farooqui et al. (2020) and Ismail et al. (2019) both indicate the importance of data collection and analysis, with Farooqui et al. proposing an event-based architecture and Ismail et al. highlighting the need for customized data pipeline tools. Oleghe & Salonitis (2020) further contributes to this discussion by presenting a framework for designing data pipelines in manufacturing systems, while Helu (2020) focuses on the scalability of data pipeline architecture to support the Industrial Internet of Things.

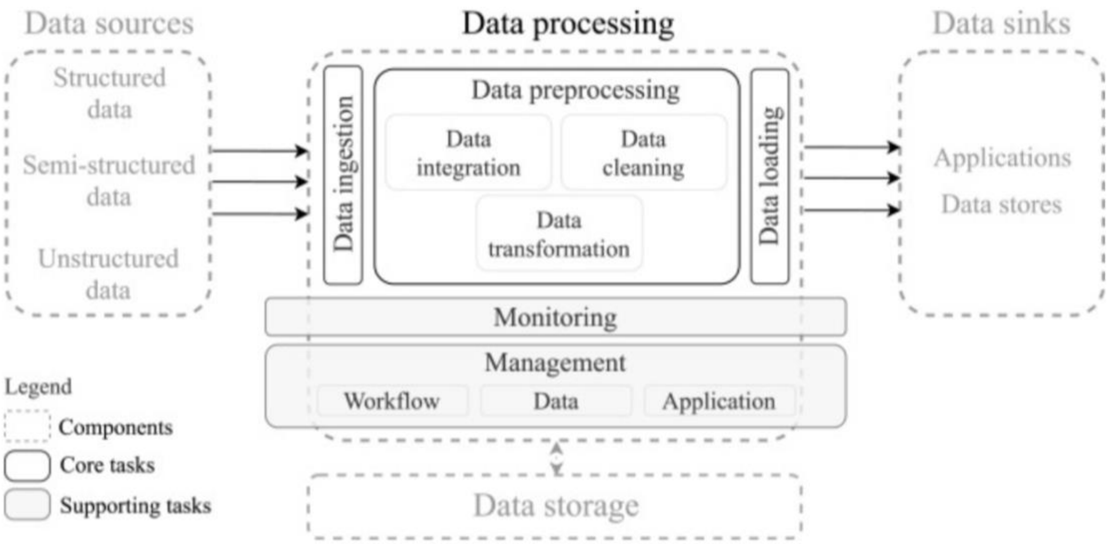

Figure 4: General model of Data Pipelines (Foidl et al., 2024)

Complexity of pipeline structures can occur when sources differ from each other, or the data needs to be vigorously processed. Different processing steps are often customized to suit specific needs (O'Donovan et al., 2015). A formalized and widely accepted general data pipeline architecture framework does not exist since most data pipelines are tailored to a specific case, but currently, one provided by (Foidl et al., 2024) is simple and modifiable and therefore this is used as a starting point in Figure 4. Although the research on data pipelines is heavily focused on processing and machine learning applications, the results are usually applicable to other use cases as well. A complex chain of processes is the reality of the data pipelines for most organizations and most organizations still practice data pipeline management manually. (Munappy et al., 2020)

The data sources phase consists of data gathering and all activities before the actual processing. Generalization can be made but like the rest of the thesis, this phase is context-specific. Ingestion sources can be divided by consumed data type into structured, semi-structured, or unstructured sources (Foidl et al., 2024). The source phase can consist of multiple homogeneous or heterogeneous sources combined in the processing phase through various means. The only limitation that these sources have is that they need to be able to be connected to the rest of the data pipeline through the internet. The manufacturing organization can utilize various data sources for different applications like predictive maintenance or continuous improvement. Data sources in this context can be for example sensors, testing devices, ERP systems, MES, MIS, RFID tags, machinery, and cameras (Oleghe & Salonitis, 2020).

The processing phase is the heart of any data pipeline, without processing it is just data transferring. Data pipeline literature rightly is the most focused on the processing phase. This phase consists of multiple different and customizable components and tools that are chosen based on the context (Ismail et al., 2019). Core practices involve data handling and manipulation among others. Dehury et al. (2022) describe the processing phase as blocks that take in inputs and deliver outputs to the next block. Each one of the blocks has from 1 to N different services or scripts that are executed to achieve transforming, storing, or processing data.

The first of the steps or blocks is data ingestion also called consumption in some of the sources. The ingestion method varies based on the raw data. The size and format of raw data and load frequencies are the deciding factors when determining the ingestion method (Foidl et al., 2024). The ingestion method as discussed in the data generation could also be batch or continuous. Already in 2016 model of a combination of the methods was modeled (Ji et al., 2016). According to Aiswarya et al. (2020a), the processing phase can ingest data in one of three ways: continuously, intermittently, or as batches. Intermittently meaning alternating or combining the continuous and batch ingestion methods.

Table 1: Ingestion Phase in Manufacturing

Ingestion Method	**Data type**	**Use case**
Continuous	Production confirmations	Continuously updated during production to ensure visibility to production
Intermittent	Production details	Load will differentiate between continuous and batch depending on the current load of pipeline.
Batch	Supplier Information	Loaded from ERP-systems in large batches at specific times.

Table 1 presents examples of different ingestion methods commonly used and possible examples to clarify the type of data that ingestion methods can employed for. Examples of continuously ingested data could be video calls or real-time updating production tracking (Narayanan et al., 2024). Video call data is consumed in (near) real-time and continuously streamed from the source. Batch generated refers to data that is generated in groups and the rest of the process triggers when enough data is sourced, or enough time has passed. Regardless of data type or ingestion method, the main purpose of this core step is to make data available for the rest of the processing. (Foidl et al., 2024)

The Second well-documented sub-step in the processing phase is pre-processing. This step aims to make data as usable as possible, and it has similarities Transform phase in the traditional ETL process (Aiswarya et al., 2020a). The likes of data integration, cleaning, and transformation are present in both methods. It is necessary to point out that not all methods can be applied to all pipelines and each pipeline requires a different combination of sub-steps to achieve its purpose (Dehury et al., 2022). The purpose of data integration according to Munappy et al, (2020) is to merge and combine different ingested data sources with practices like schema mapping or redundancy removal. Data cleaning steps in the pre-processing phases can include scripts performing missing value treatment and noise treatment. The main purpose of data cleaning steps is to exclude errors and inconsistencies. Data transformation steps include statistical techniques from mathematics like normalization and discretization. (Foidl et al., 2024)

Table 2: Pre-Processing Phase Steps and Techniques

Step	**Techniques**	**Industry Example**
Integration	Redundancy removal, Schema mapping	ERP Data and Supplier data combination
Cleaning	Missing value treatment Noise Treatment.	Missing Purchase Order Number
Transformation	Normalization, discretization, Statistical techniques.	Tester result cleaning

The third major step in the basic data pipeline framework is the loading phase where the processed data is loaded to the data sinks. The loading phase is the culmination of the previous steps where the effort bears fruit, usable data. Examples of these data sinks can be data warehouses, artificial intelligence or machine learning applications, or other data-intensive applications. (Aiswarya et al., 2020a; Foidl et al., 2024)

The activity monitoring phase in the framework, provided by Foidl et al. (2024), oversees and controls the pipeline performance. Monitoring is important to ensure the reliability, efficiency, and rapidness of data pipelines. According to Somasundaram (2023), pipeline monitoring helps prevent issues with data quality and failure recovery. Monitoring in this framework will reach all data operations and all different nodes during processing. Pipeline activity monitoring can be performed by using logging mechanisms, creating alerts, data validation, or anomaly detection (Aiswarya et al., 2020b). Somasundaram (2023) suggests using machine learning and indicates that robust monitoring ensures that anomalies and failures throughout the pipeline are noticed. Narayanan et al. (2024) extended alerting to five categories including specialized alerting for latency, data quality, lineage, volume, and consistency. The monitoring phase does not necessarily consist of only one step but utilizes multiple different monitoring categories depending on the subject. Data pipeline monitoring in short is monitoring both the flowing data quality and the performance of the data pipeline (Foidl et al., 2024).

Previous studies have demonstrated the effectiveness of varying methods for validating and monitoring data pipelines. Aiswarya et al, (2020b) emphasize the need for automated fault detection mechanisms in data pipelines, and Table 3 lists possible mitigation strategies.

Redyuk et al. (2021) suggest practitioners take a data-centric approach that automates data pipeline monitoring, using descriptive statistics and machine learning-based anomaly detection. Schelter et al. (2018) present Deequ, an Apache Spark library that allows data assumptions to be specified and automatically validates assumptions made for datasets. Narayanan et al. (2024) linked real-time monitoring as a major contributor to data observability improving data quality and integrity, enabling anomaly detection, and debugging efforts. Dreves et al. (2021) discuss the development of tools for validating datasets and models in continuous ML pipelines.

Table 3: Data Pipeline Anomalies/Faults Adopted from Aiswarya et al, (2020b)

Data Pipeline Stage	**Faults**	**Mitigation Strategies**
Data Generation	Data source failure	Set a proxy which never fails
	Inactive data source	Send notification to restart the source
Data Collection	Authentication Failure	Functional user credentials
	Data Sending job failure	Send notification about failure
	Unexpected data	Send email to flow guardian
Data Ingestion	Incompatible ingestion methods	Log the error Define dedicated ingest methods
	Data Extraction Faults	Conversion to acceptable format, formalize the data, define data extraction method that works for all formats
	Change in data formats	Versioning mechanism
Data Storage	Insufficient storage	Alarm to the developer and then to support team
	Data Duplication	Use of HDFS
	Infrastructure failure	Sending alarm to IT support
Data Processing	Transformation faults	Define lossless approaches
	Unclear definitions and wrong interpretations	Contact SMEs
	Human errors	Data validation
	Schema Errors	Define common schema and common language. Write different parsers
Data Sink	Dirty data	Statistical methods, Data imputation techniques

Even though multiple sources like Aiswarya et al. (2020a, 2020b) and Foidl et al. (2024) highlight the importance of data pipeline monitoring, still there are no existing general models or frameworks on practical guidelines. Helping with assessing the common faults to monitor in different phases of the data pipeline Aiswarya et al, (2020b) have gathered faults and suggested mitigation strategies shown in Table 3. Mitigation strategies during the ingestion phase are error logging, changing methods, conversions, and versioning which could be used in our problem domain. Other notable mitigation strategies suggested could be notification sending and statistical methods.

Data pipeline management is according to Foidl et al. (2024) tasks that are related to the overall data pipeline picture. Management consists of practices related to workflows, applications, or data. Workflow management focuses on data pipeline orchestration and dependency analysis between data processing nodes. Munappy et al. (2020) propose that the purpose of data pipeline management is to A) accelerate data life cycle activities, B) enhance productivity in data-driven enterprises, and C) ensure the automation of the data flow between steps. The application focuses on configuring and maintaining the collection of different tools included throughout the data pipeline. Especially in the context of manufacturing pipelines can have a plethora of heterogeneous tools and data management in this context refers to metadata management, versioning, and cataloging (O’Donovan et al., 2015).

Table 4: Data Pipeline Quality Influencing Factors Foidl et al. (2024)

Category	Description
Data characteristics	
Data dependencies	Instability of data regarding their quality or quantity over time.
Data representation	Data formats, data structures, data types, and further data representation aspects (e.g., data encoding).
Data variety	Heterogeneity (i.e., formats, structures, types) of the data consumed.
Data volume	Quantity and rate at which data are coming into the pipeline.
Data management	
Data governance	Processes, policies, standards, and roles related to managing the data processed by a pipeline (e.g., data ownership, data accessibility, raw data storage).
Data security	Degree to which data are protected during their transmission and processing.
Metadata	Degree to which data consumed and processed are documented (e.g., data catalog, data dictionary, data models, schemas).
Data sources	
Complexity	Number of data models and sources a pipeline must deal with, including the simplicity of merging the corresponding data (e.g., joinability).
Reliability	Degree to which data sources are available, accessible, and provide data of high quality.

Data pipelines can have a multitude of characteristic problems that stem from improper management. Aiswarya et al. (2020b) investigated the complexities, challenges, and benefits associated with implementing and managing data pipelines. Challenges can be categorized as infrastructural, organizational, and data quality challenges. Quality influencing factors of data pipelines in Table 4 were assessed by Foidl et al. (2024) concluding that key factors are dependencies, representation, variety, volume, governance, security, metadata, complexity, and reliability. Especially in data pipeline management challenges can arise from missing data files, operational errors, and logical changes. Automation of the pipeline is essential to reduce possible faults caused by downtimes and performance degradation (Aiswarya et al., 2020b). Munappy et al. (2020) and Foidl et al. (2024) suggested that monitoring practices, regardless of data quality or data pipeline performance, should be integrated throughout the data pipeline to ensure visibility and should be scalable.

2.3 What is Data Quality?

Does your data represent reality? Organizations may gather data to have data rather than have a use for it. This blindness to the relevance of the data may hinder usability and representability of the real world. Analytics hindered by bad data quality is more likely to affect decision-making and transparency negatively (Torres & Sidorova, 2019). Data quality management processes and tasks try to solve issues relating to consistency, accuracy, completeness, timeliness, and uniqueness of the data among other dimensions (Fan, Wenfei & Geerts, 2022).

Data quality management is not once and done kind of project, but rather in general it requires continuous data audition and cleansing efforts from organizations (Albrecht et al., 2023). Effective data quality management will positively impact data analytics and enable organizations to be more data-driven (Kwon et al., 2014). Inefficient or missing data quality management protocols or practices result in non-visibility to data. This will also harm decision-making and strategic planning depending on the data (Glowalla & Sunyaev, 2014). According to Carretero et al. (2017), data quality management is defined as the implementor, and maintainer of data quality culture and guides and develops best practices defined by data management regarding data quality similar to data governance. Data quality improvement strategies can be the following: improving data sources, cleaning up the database, improving the database/architecture, and working with insufficient data quality (Schröer et al., 2023).

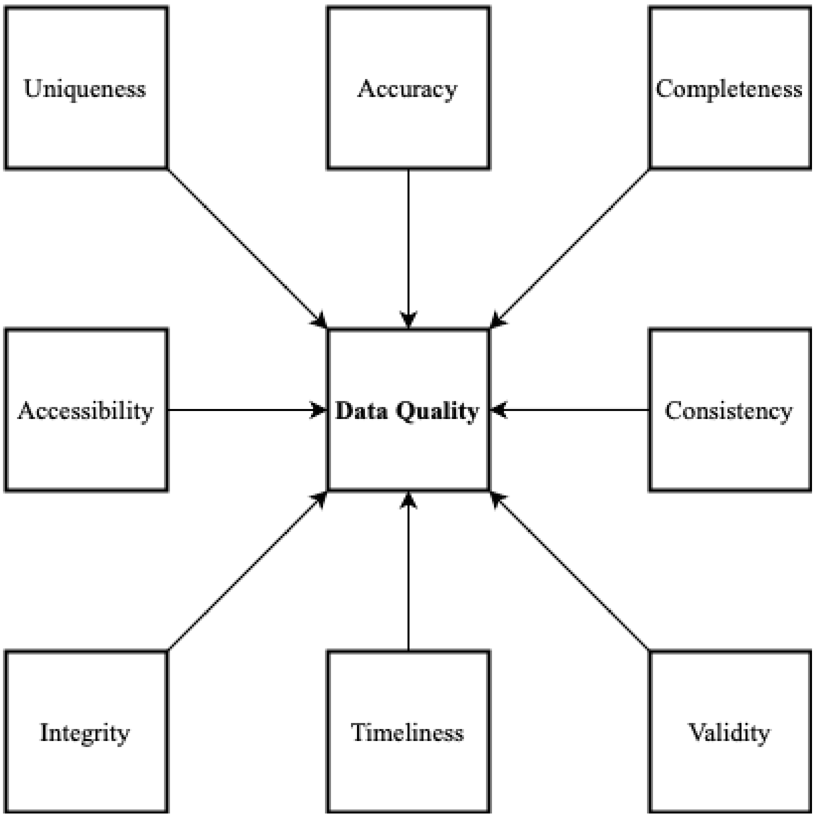

Figure 5: Data Quality Dimensions (Albrecht et al., 2023; Fan, W., 2015; Fan, Wenfei & Geerts, 2022)

The key dimensions of data quality in Figure 5 are essential to understanding and creating data quality solutions. Data consistency, deduplication, completeness, currency, and accuracy are the foundations of data quality according to Fan (2015). Fan's interpretation is concurrent with what Albrecht et al. (2023) theorized about information quality based on previous research. Albrecht et.al (2023) referred to a few previous studies (Madnick et al., 2009; Pipino et al., 2002; Wang, R. Y. & Strong, 1996) and still listed the following six dimensions in their work: accessibility, accuracy, completeness, consistency, integrity, and timeliness. This definition and form are common in literature and often change depending on the context of the research. Changes, mainly detractions and additions to this list, mainly occur to suit the researchers' focus areas. Regardless of the dimensions chosen, many factors can affect data and information quality.

Different dimensions of data quality complement each other and are somewhat overlapping in nature. The accuracy of the data has been traditionally considered and assessed as syntactic accuracy. Syntactic accuracy is the distance between the value and the corresponding domain according to Albrecht et al., (2023). The basis for the accuracy dimension is the reliability of data. The accessibility dimension measures the availability and access easiness. The completeness dimension can be explained as the degree to which the data represents the real world (Madnick et al., 2009). All transactional data in this scope represents real-world events. Uniqueness measures whether data contains multiple instances of the same data. Duplicates might result in errors, difficulties, or inconsistencies in calculations. Consistency refers to contradictions in the data. Contradictions occur if data collection and storing rules are not properly modeled. The integrity dimension refers to violations of semantic rules like formatting. The timeliness dimension considers data quality from the point of data aging. Context matters and states what the time frame should be for the data to be up to date. (Albrecht et al., 2023; Fan, W., 2015; Torres & Sidorova, 2019)

In data quality research, there are many different research topics and fields. Already in 2009, data quality was researched with a focus on: impacts, database-related solutions, in the context of computer science, and curation (Madnick et al., 2009). Not surprisingly recent Data Quality research is focused on Big Data, Automatization, or Machine Learning (Albrecht et al., 2023; Fan, Wenfei & Geerts, 2022; Miksa et al., 2022; Oleghe & Salonitis, 2020; Torres & Sidorova, 2019; Zhang et al., 2023). Data Quality Management is often classified in the newer literature under or overlapping the Data Governance section, but this categorization depends on the context and source (Wang, J. et al., 2018; Weber et al., 2009). Data Quality is somewhat integrated or at least considered through various parts of Data Management and data-related literature.

Different Data Quality dimensions were introduced earlier in the chapter and Figure 5, and they might raise a question about measuring them. Setting measures on a large-scale database or single data table might differ but techniques or methods can be similar. Generality to multiple use cases and sources is also one topic that Albrecht et.al (2023) recommend to practice. The main steps of their DQM methods include Data Quality Definition, Data Quality Assessment, and Data Quality Improvement. Data quality definition is a phase of a cyclical process where experts and key stakeholders define and analyze the existing data quality. Data Analytics, Meta-data, and existing documentation can be used to do state construction as the definition phase (Batini et al., 2009). Data quality assessment is almost a field of its own at this point and time and will be discussed later on a more detailed level, but it measures the quality of data collection regarding the different data dimensions. Data quality improvement, however, focuses on actions and practices that organizations should do (Batini et al., 2009).

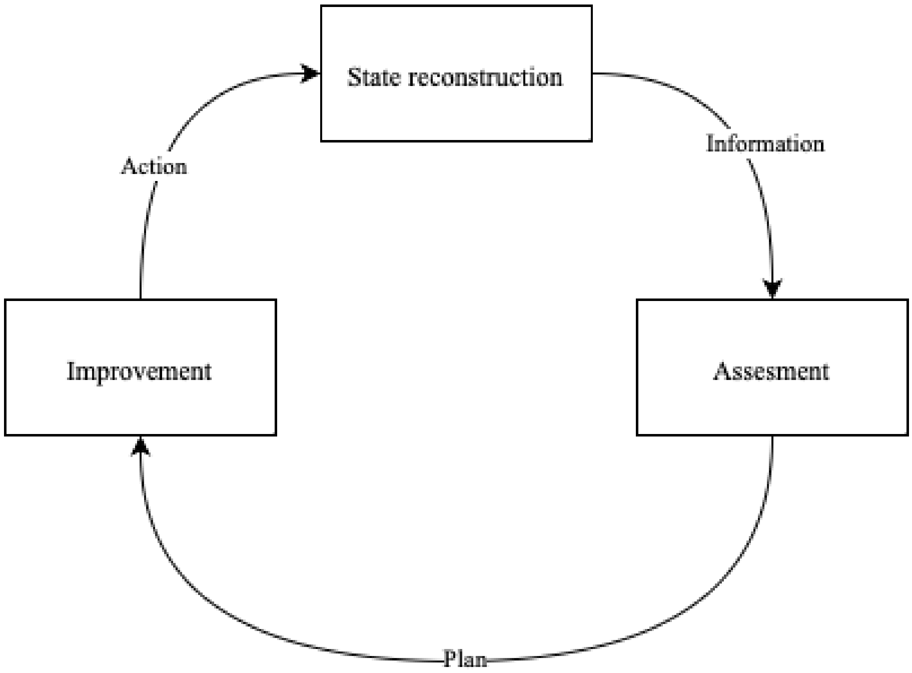

Figure 6: Data Quality process model (Batini et al., 2009)

In Figure 6 above we see a cyclical process based on common data quality process models. These processes have not changed much through the years but are context-dependent and for that reason the differences are common. State reconstruction creates information artifacts for the assessment phase also known as the Quality Assessment phase. The focus of this phase is to gather information about the organization’s processes, data collection, and management procedures. All the information must be in context with the problems at hand and this phase may define quality issues inside the context. Cost analysis or other previous analyses might be fruitful in creating information. The improvement phase ends with steps or strategies to reach quality and cost analysis targets. (Batini et al., 2009)

Data Quality Monitoring is the focus of this thesis work as the theory is near real-life practices. Different data quality dimensions are reviewed, categorized, and measured. Before the measuring can occur limits and measurable attributes must be assessed. Limits are placed according to reference values. One of the more promising frameworks in the focus area of Data Quality Assessment is provided by Cai and Zhu (2015). The framework is based on the hypothesis that data quality is context-related and hierarchical. The theory behind the framework is largely based on previous data quality dimension research conducted by Richard Wang and his colleagues at ACM (Madnick et al., 2009; Pipino et al., 2002; Wang, R. Y. & Strong, 1996). Their framework consists of five quality dimensions and each dimension has different elements and indicators, these indicators to the framework can be found in Appendix 2.

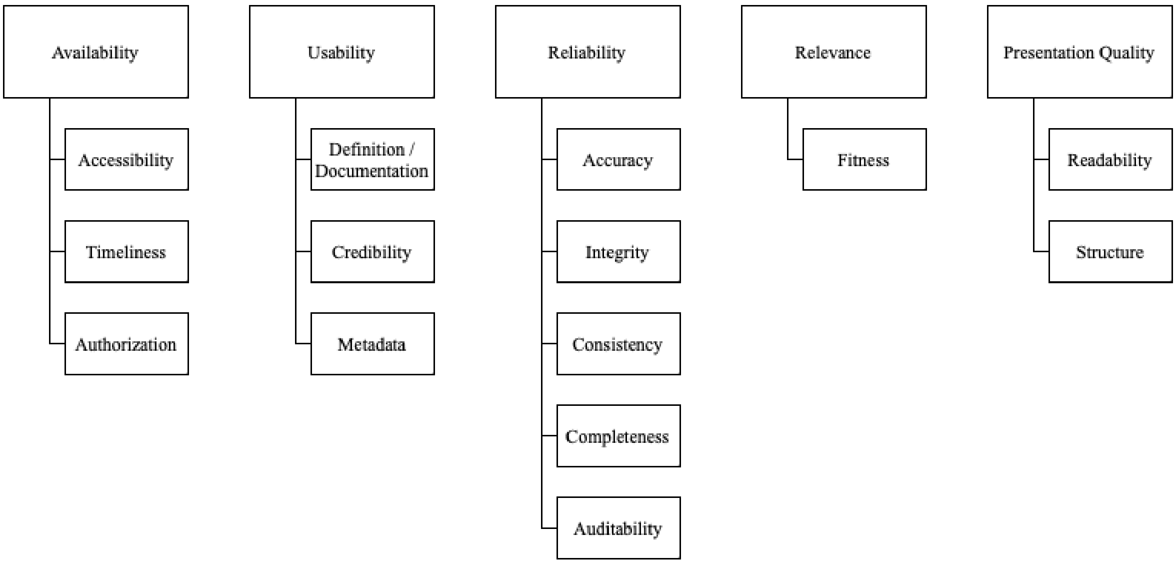

Figure 7: DQA Framework Cai and Zhu (2015)

Two-layer quality standardization in Figure 7 by Cai and Zhu (2015) was originally made for big data but applies to our use case. The organization in question creates a vast amount of data through automized processes. Although discussing and theorizing for example about the accuracy of data might be necessary it is not easy to measure.

Literature shows plenty of different Data Quality Assessment processes (Cichy & Rass, 2019). They usually follow the basic structure of the DMAIC problem-solving technique established and popularized in the industry by Lean Six Sigma. DMAIC stems from the terms Define, Measure, Analyze, Improve, and Control. Quality Management in traditional industrial settings has long traditions in terms of formalized processes and frameworks. Carretero et al. (2017) adapted the development of data management scope from 1950 to 2016 and show that data quality management is relatively new. No wonder that DQM has adapted processes and frameworks from Quality Management. One example of data quality assessment processes found in the literature is by Cai & Zhu (2015). As mentioned before their research is more related to big data, but it is a good example of how simple the process can be under the right circumstances.

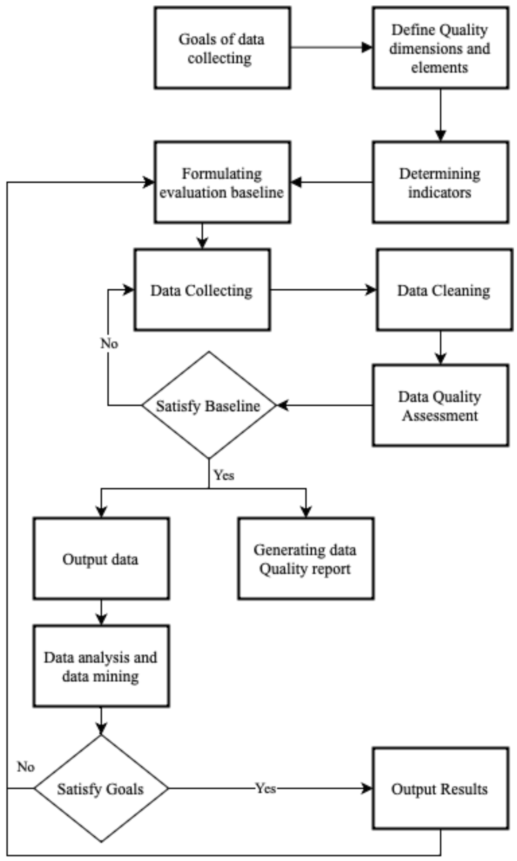

Figure 8: Data Quality Assessment Process (Cai & Zhu, 2015)

Figure 8 resembles a more complex and detailed data quality assessment process. Quality dimensions are there to be used to set the evaluation baseline (Cai & Zhu, 2015). Although iterative this process still lacks re-evaluating of the goals of the data collecting that may change through the process. Data quality is a colorful and unformalized research topic. Most of the literature still refers to data quality dimensions in some sense although there are discrepancies in the importance of each. (Ehrlinger & Wöß, 2022)

The basic steps of most data quality-related processes follow the same pattern: Define, Measure, Improve, and Continuous Monitoring. The define phase is often conducted by gathering metadata (Cai & Zhu, 2015). Metadata can be gathered with various data profiling techniques. Measurement is conducted by evaluating data collections against pre-determined dimensions and attributes. The improvement phase can be accomplished by data cleansing tools temporarily, but if the problem is systematical might be necessary to change the data modeling or structure significantly. Data quality monitoring is a term used in the Continuous Monitoring phase with many definitions. Monitoring refers to either rule checking or ongoing measurements. (Ehrlinger & Wöß, 2022; Fan, W., 2015)

Data profiling is often performed with a variety of tools. Profiling creates an understanding of the data structure, completeness, and distributions. Metadata is often used for easier analysis if available. Metadata includes information about the distinctness, data types, frequency, formatting, and patterns emerging in the data. *"Profiling as a data quality assessment tool reveals data errors, such as inconsistent formatting within a column, missing values, or outliers"* (Abedjan et al., 2015). Statistical analysis is used for distributions, and information can be created using basic statistics like means, medians, and deviations. Although data profiling can be a valuable tool it uses mainly rule-based methods and algorithms. Rule-based tools do not necessarily mean that profiling would be inadequate, but that tool cannot discover anything outside the rules. Especially if the target of data profiling is a vast amount of data, only simple statistics should be used (Schelter et al., 2018). More complex analysis can drain computing resources unnecessarily. (Ehrlinger & Wöß, 2022)

Data quality measurement is one of the more complex topics in this field of research. The main problem is how to measure something that is A) contextual and B) not theoretically agreed upon. Equations are not reviewed in detail in this seminar work instead the possible sources for each dimension equation are referred. Literature provides good explanations and metrics for some of the most researched data quality dimensions. These dimensions are Accuracy, Completeness, Timeliness, and Consistency. (Ehrlinger & Wöß, 2022)

How accurate data should be? Accuracy can be measured syntactically. Hinrichs (2003) proposes a few mathematical equations for Accuracy measurement in his PhD thesis. These equations are discussed, and their validity is proved by further research (Ehrlinger & Wöß, 2022). Timeliness is measured by the decline rate of the context attributes in the time frame. Consistency and completeness dimensions highly correlate with the perceived benefits of data usage with both internal and external data (Kwon et al., 2014). The consistency equation can also be found in the works of Hinrichs (2003) and is calculated by the sum of violation of consistency rules. Completeness is often measured with the percentage of incomplete elements of the whole population of the elements (Pipino et al., 2002).

Data Cleansing is an automatable process of standardization, matching, and ensuring uniqueness (Cai & Zhu, 2015). Most data quality tools have a data cleansing system or protocol. Data cleansing as an automated process is necessary if the scale of data is too large to manage manually. Regardless of the use case data cleansing is limited by rules and modeling of the rules. (Ehrlinger et al., 2023)

Data Quality Monitoring is usually a somewhat automated process that can involve similar elements to the quality measurement phase, but it is done continuously throughout the data life cycle. Monitoring can be done using assessment limits and performance testing. Some of the literature also mentioned using continuous data profiling techniques mentioned before in the monitoring context as well (Ehrlinger & Wöß, 2022). This phase can be as simple as using simple pre-defined metrics to evaluate the metadata created by refreshes. Recent research on the topic shows that automation of data quality monitoring can be created using reference data profiles and machine learning (Ehrlinger et al., 2023).

2.4 Summary of Anomaly Detection

Anomaly detection is a crucial practice for any data system or construct to ensure the intended working and quality of data (Blázquez-García et al., 2021). Even the most robust systems are prone to unexpected things happening, and no system can ever be prepared in a way that would be fully fault-proof (Omri et al., 2021). Including or integrating anomaly detection methods or algorithms into the system will at least create visibility for complex data processes. Industry 4.0 and digitalization are great examples of the changing data landscape that has a growing amount of more customized and diverse systems. Diverse sources and systems need to be monitored and anomaly detection methods can make a difference (Blázquez-García et al., 2021). Prognostics and Health Management (PHM) has recently grown as a fundamental for engineering systems. PHM can help with many real-life problems, like the detection and classification of anomalies (Omri et al., 2021).

The definition of anomaly is a pattern that is not like the common behavior of a system according to Chandola, Banerjee, and Kumar (2009). The normal behavior of the system needs to be defined in detail to allow the creation of automized methods (Chandola et al., 2009). Anomaly has similar properties to an outlier, although outlier is an observation distinctly different from the others (Blázquez-García et al., 2021). It is not important to distinguish between anomaly and outlier since the terms are used interchangeably although Blázquez-García et al. (2021) note that literature regards anomalies as events of interest and outliers as unwanted. Anomalies can also be categorized by their characteristics as point anomaly, contextual anomaly, or collective anomaly (Ahmed et al., 2016; Chandola et al., 2009). Point anomaly would be like an outlier as a single point outside of the normal. Contextual anomalies could be holidays that cause production to halt or similar occurrences. An example of a collective anomaly would be a data instance-like set of test results differing from the rest of the data collection.

Now that anomalies are defined on a conceptual level the focus will shift to common literature approaches and techniques to detect the anomalies. Approaches depend on the context of the problem. According to Chandola et al. (2009) in sensor networks, anomalies can mean that one or more sensors are faulty. Techniques can be grouped by the type of the technique and Chandola et al. divide anomaly detection techniques into statistical, clustering, and machine learning algorithms. Instead, Ahmed et al. (2016) categorized anomaly detection methods into classification groups and statistical techniques. Classification group has techniques like support vector machine, Bayesian network, rule-based, and neural network and Statistical techniques include mixture model, signal processing, time-series analysis, and principal component analysis (Blázquez-García et al., 2021). Sensor networks have similarities to the data pipeline ingestion phase where the monitoring anomaly detection should consider breakages or other anomalies throughout the pipeline. As the data pipeline monitoring usually has a temporal aspect time series analysis like ARIMA could be applicable (Chandola et al., 2009).

To conclude anomaly detection is useful for ensuring the functionality and quality of data systems. An anomaly is a deviation from the typical behavior of a system, often categorized as point, contextual, or collective anomalies. Techniques for detecting anomalies range from statistical methods to complex machine learning approaches, with choices depending on the problem context and available data. Similarly, in data pipeline monitoring, anomalies must be identified throughout the pipeline.

3 Thesis methodology and conducted research process

DSR was chosen as a research methodology to create a practical solution with the combination of the practical expertise of the company and a rigorous literature review to create the best possible result. The result is expected to be a customized solution that should perform in similarly designed environments and with similar architecture or limitations. The expertise of the organization is harnessed with semi-structured interviews.

3.1 Approach to interviews and basics of Design Science Research

Interviews are semi-structured to emphasize for the interviewees to speak their minds on the topic. Semi-structured interviews are designed to illustrate perceptions inside the organization and to better understand the research area (Barriball & While, 1994). To ensure information and organization secrecy none of the actual interview contents are shown and the information gathered in the interviews is generalized. This means that the tools, requirements, or findings are not necessarily the same as the ones used in practice, but they give a generalized understanding of the context. The literature review gives an understanding of the subject matter and the basis of the interview guide that will be used during the interviews. Construction of the semi-structured interview follows the general interview framework represented by Kallio, Pietilä, Johnson, and Kangasniemi (2016). Semi-structured interview framework suggests three main phases: Gathering previous knowledge, Formulating an Interview Guide, and Piloting (Kallio et al., 2016). The formulated interview guide can be found in Appendix 1.

Semi-structured interviews were chosen as the research method for all the research questions for the following reasons. Data quality is always context specific therefore we need to understand the context and use of the data to assess it. Semi-structured interviews are useful for knowledge gathering in complex and context-specific research. For the artifact, the interviews provide more perspectives and the generation of ideas from the experts that may lead to different design choices or new requirements for artifacts. Similarly, to the data quality the data pipeline monitoring solution created is context-specific and multifaceted, hence the use of semi-structured interviews is reasonable.

Design Science Research (DSR) is a research methodology that explores the knowledge generation process in the creation of artifacts, combining practicality with scientific rigor (Vom Brocke et al., 2020). According to Hevner et al. (2004), DSR aims to extend human and organizational capabilities by developing and constructing new and innovative artifacts and is common in Information Systems research. The DSR process centers around the creation of a design artifact, which in turn leads to the development of a design theory for action and use (Siedhoff, 2019). Järvinen (2007) illustrates DSR objectives as finding and developing new properties of social, informational, or technical resources by creating artifacts. This definition covers an object that implements a practical and embedded solution to a well-defined research problem. This approach is also supported by Van Aaken (2004) and adds a recommendation of defining a general solution model based on the created artifact.

DSR application domains, for example, include Information Technology related fields for example engineering, management, and information systems (Gregor & Hevner, 2013; Hevner et al., 2004). Designed artifacts and the design process should be illustrated in a way that they can be recreated easily (Peffers et al., 2007). In contrast to traditional behavioral science paradigms that aim to validate theories or concepts, the design science paradigm aims to create applicable, practical, and effective solutions (Hevner et al., 2004). Hevner et al. also remark that both paradigms are important to creating relevant and effective research outcomes.

DSR aims to contribute in two ways design artifacts and design theories (Baskerville et al., 2018). According to Hevner et al. (2004), artifacts can be broadly classified into constructs (vocabulary and symbols), models (abstractions and representations), methods (algorithms and practices), and instantiations (implemented and prototype systems). Later to this list was added design principles, technological rules, and design theories (Gregor & Hevner, 2013). Weigand et al. (2021) argue in their ontological research that 1) DSR practitioners should aim to create a universal artifact rather than context-specific to better contribute to the theory and 2) that instantiations are not artifacts, but rather evaluations of existing ones. Their guideline for designing DSR artifacts is straightforward forward “Make a plan – use a plan”. Artifacts and theories are the main contributions of DSR, and they can be classified and labeled in multiple ways. Gregor & Hevner (2013) introduced a maturity-based knowledge contribution framework.

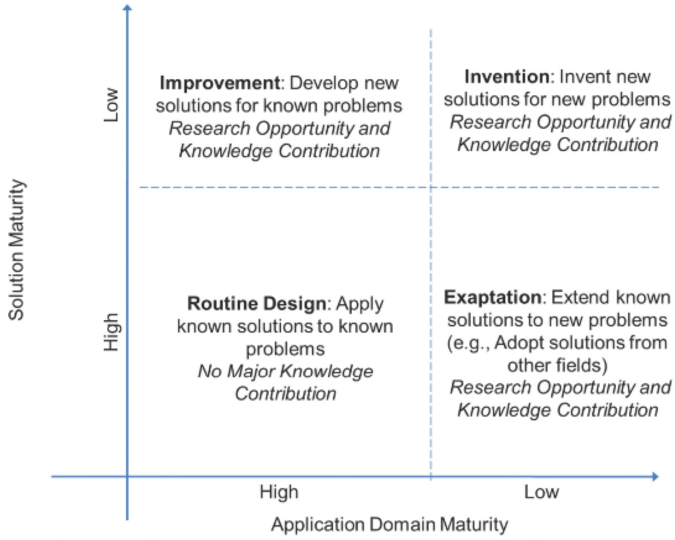

Figure 9: DSR Knowledge Contribution Framework (Gregor & Hevner, 2013)

Research contributions in the DSR paradigm can be classified by solution maturity and application solution domain maturity. Matrix framework in Figure 9 defines solutions potential contributions depending on both: solution and domain maturity. Routine designs that have a high solution and domain maturity will not have a major contribution but rather can affirm an already-made knowledge base. Exaptation is for example adopting solutions from other fields where improvements are new solutions to well-known problems. Inventions are the solutions where neither the solution nor domain maturity is high. (Gregor & Hevner, 2013)

Design Science Process is by core definition iterative for practical use (Hevner et al., 2004; Kuechler & Vaishnavi, 2012). Design Science Processes have a few other key steps other than the iterative nature. The design Science Process is based on guidelines for DSR defined by Hevner et al. (2004). Guidelines define the well the outlines and following the general guidelines ensures the research is on the right track.

Table 5: DSR Guidelines (Hevner et al., 2004)

Guideline	**Description**
Guideline 1: Design as an Artifact	Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation.
Guideline 2: Problem Relevance	The objective of design-science research is to develop technology-based solutions to important and relevant business problems.
Guideline 3: Design Evaluation	The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods.
Guideline 4: Research Contributions	Effective design-science research must provide clear and verifiable in the areas of the design artifact, design foundations, and/or design methodologies.
Guideline 5: Research Rigor	Design-science research relies upon the application of rigorous methods in both construction and evaluation of the design artifact.
Guideline 6: Design as a Search Process	The search for an effective artifact requires utilizing available means to reach desired end while satisfying laws in the problem environment.
Guideline 7: Communication of Research	Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences.

Design Science Research is based on principles that guide the research. Hevner et al. (2004) defined seven guidelines to guide researchers shown in Table 5. These seven guidelines ensure the validity and relevance of the research. Järvinen (2007) explored these guidelines and created suggestions for improving and refining them. Järvinen (2007) would rather formulate the first guideline as “Design of Innovation” and would formulate the 2nd and 6th guidelines as more opportunity-focused rather than problem-based.

3.2 Design Science Research process in general

The first step in DSR is often understanding or defining the problem (Järvinen, 2007; Kuechler & Vaishnavi, 2012; Peffers et al., 2007). This step should contain a formal definition of the problem and researchers should be able to show the importance of the problem domain (Peffers et al., 2007). Current problem state analysis and general state analysis will outline the sector of possible solutions and create a basis for objective setting

(Vom Brocke et al., 2020). The first DSR process steps show the audience the wide understanding needed since the problem domain is often very complex and needs to show importance (Peffers et al., 2007).

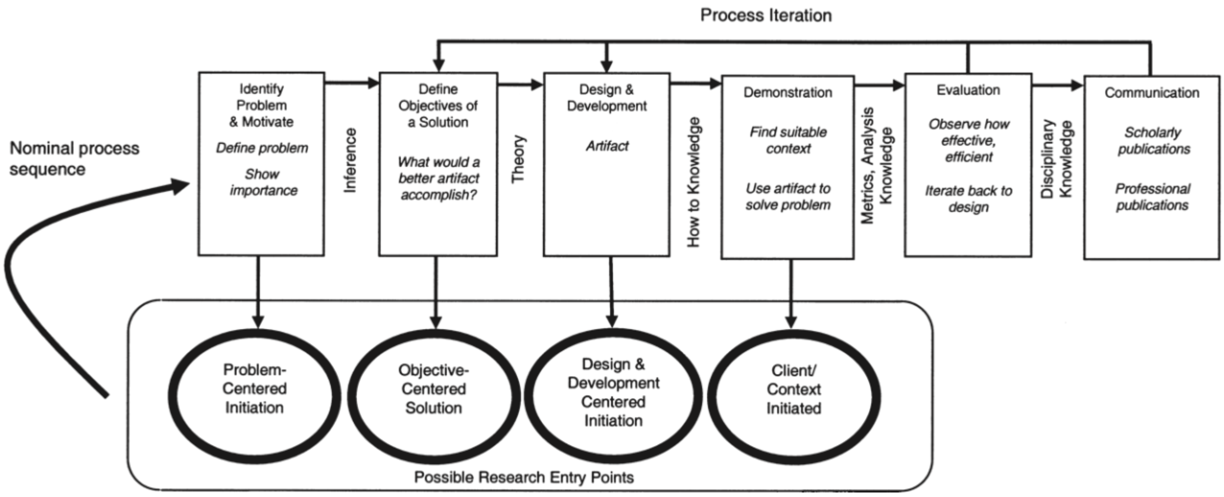

Figure 10: DSR Process (Peffers et al., 2007)

The next step in the DSR Process in Figure 10 is defining the objectives of a solution (Peffers et al., 2007). According to Vom Brocke et al., (2020), this process step needs information on the state and scope of possible solutions, and the inputs should be inferred to rational objectives. This is a part of the process where previous foundational theories or “kernel theories” should be reviewed (Kuechler & Vaishnavi, 2012). Objectives can be defined as quantitative or qualitative. Quantitative objectives can be measurable metrics that illustrate the improvement and qualitative objectives describe the planned effect of the artifact (Vom Brocke et al., 2020). The output of this phase is refined to a clear objective that states what will be accomplished with the designed artifact (Peffers et al., 2007). Sonnenberg & Vom Brocke (2012) propose that the following methods could be applied: expert interviews, assertions, literature review, focus groups, practitioner initiatives, and surveys.

The third step of the DSR process consists of the design and development of the artifact based on the rigorous literature review (Peffers et al., 2007). The design and development step usually refines the initial objectives since all requirements cannot be foreseen. Weigand et al (2021) deem it necessary to create “a well-described design specification” that refers to a similar concept which is classified as rigor by Hevner (2004). In this phase, similar artifacts and theories from the literature should be reviewed and requirements can be derived from them (Kuechler & Vaishnavi, 2012). Designs can be initially evaluated by assertions, mathematical proof, logical reasoning, simulations, expert interviews, or focus groups (Sonnenberg & Vom Brocke, 2012). A researcher should also define the needed abstraction level that is based on the artifact type created for example instantiation or construct (Gregor & Hevner, 2013).

The actual design process and development process are not formalized but should include common software development best practices like requirements gathering and use case demonstration or illustration of the needed functionality (Peffers et al., 2007). Requirements gathering can be performed through various techniques including interviews, questionnaires, and prototyping, as well as using tools for elicitation, analysis, and verification (Sajid et al., 2010; Shah et al., 2017). The design and development phase should be documented in a way that makes the artifact possible to replicate with relative ease (Peffers et al., 2007). Hevner et al. (2004) indicate that design specification and justification should occupy a significant amount of the length of the research paper and formatting can be varied.

The demonstration process step is where the effectiveness of the artifact in solving a problem is made clear to the reader (Peffers et al., 2007). If the process was initiated without context, then applying the artifact to a new context must be demonstrated and the suitability indicated with for example artifact prototypes (Sonnenberg & Vom Brocke, 2012). The evaluation step should be conducted with pre-determined metrics on the applicability of the solution (Peffers et al., 2007). Common metrics could be reliability, system activity metrics, or syntactic accuracy metrics. According to Sonnenberg & Vom Brocke (2012), the final evaluation is the ultimate show of applicability and practicality. Possible evaluation methods could be analytics, case studies, experiments, simulations (Hevner et al., 2004), surveys, expert interviews, and focus groups (Sonnenberg & Vom Brocke, 2012). Venable et al. (2016) provide DSR evaluation strategies depending on the nature of the artifact and ranging from Quick & Simple to Purely technical.

The communication step aims to elaborate and reassure the problem's relevance and the artifact's novel properties (Peffers et al., 2007). They also indicate that the DSR process can be used to structure the research paper to build the knowledge base and communicate the research process and contributions. Vom Brocke et al (2020) note that there is not one set of formal communication methods, but the goal is to communicate 1) "all aspects of the problem" and 2) "designed artifact" to relevant stakeholders. Communication should be formatted clearly for both academics and practitioners and still indicate all relevant topics with correct terms (Aken, 2004).

3.3 Conducted DSR Process

Following the DSR methodology, the final monitoring solution was created because of many iterations that were initiated mainly by feedback and observations. Feedback provided by the key stakeholders and new observations made from solutions data sources were the main drivers for new iterations during the DSR research process. Before interviews main research avenues for the possible practical solution were based on estimation. Two prototypes were created that were positioned similarly to the solution options one and two explained in more detail in Chapter 5.1. The first practical solution options had problems filling the requirements since purely estimating the end user data would only show us disruptions in the data pipeline after the problem was fixed. These options were eventually relinquished in an early development phase since methods used in those solution concepts were prone to errors and other influencing factors like supplier delays or reduced purchases.

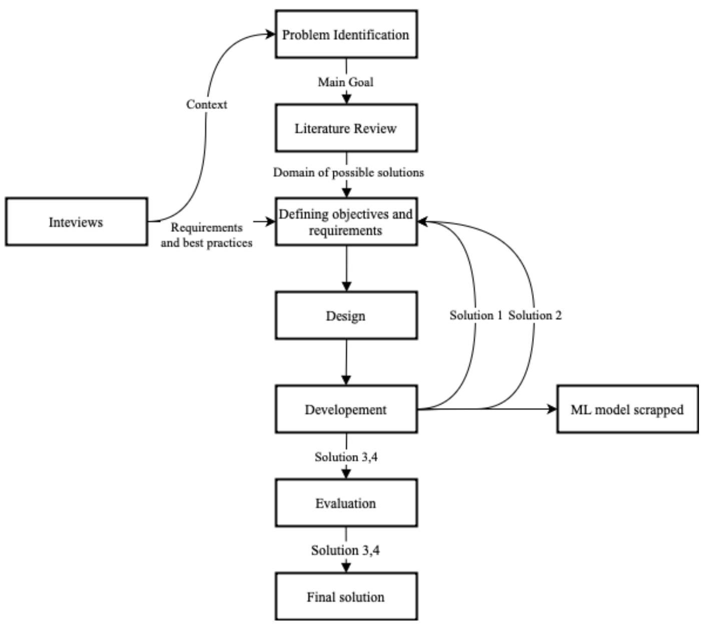

Figure 11: Followed DSR Process

Where this thesis documentation follows a linear path, we did purposefully not include the less successful attempts in the description of creating the solution. Figure 11 shows the general process flow of the process to demonstrate the realization of the final solution. Notably iterating and generating solutions options and redefining objectives during the process. During the DSR process, a total of 5 different solution options were prototyped and the combination of the final two was accepted as the final solution design and instantiation that is proven to be both practical and effective.

A simple machine-learning model prototype was also developed during the DSR study. Machine learning mode was supposed to label time-series data to anomalous and non-anomalous days based on both qualitative and quantitative factors. Machine Learning model prototype had a few different problems so ultimately the decision to diverge from this solution option was made with stakeholders. The main problems that the machine learning model had were the following: lack of providing useful new information for the final solution and lack of possible continuous support and maintenance. While the final solution approach was designed to require a minimal amount of maintenance and created with tools that the company had competence in the machine learning model would have required special expertise to maintain. While factoring in the lack of useful results, the cost of running these large models and additional organization costs it is not hard to see why this solution approach was abandoned

4 Case description of the company and data quality picture in the company

Chapter 4 outlines the case company attributes and the practical environment for which the artifact is developed and designed. The background information subchapter creates an understanding of the process where the problem domain is located and an overview of the significance of this thesis research. The background chapter maps to the identification of the problem step of DSR. Data for this chapter is gathered from workshops held with company representatives, internal documentation, and empirical observations during the DSR project.

4.1 Background information of the case company

The case company is a pioneering technological leader in its manufacturing sector and serves customers across utilities, industry, transportation, and infrastructure globally. The company provides a wide range of goods and services in electronic device manufacturing among other fields. With operations in multiple countries, the company employs a significant amount of people worldwide and is well-established in key global economies. The company focuses on innovation, research, and development, with key research areas in digitalization, AI, advanced robotics, and sustainability. It is recognized as an industry leader and has been acknowledged for its innovation, sustainability efforts, and business excellence.

This thesis investigates a singular division that manufactures and develops electronic devices designed to work in various applications, such as pumps, fans, compressors, conveyors, and other machinery. Common sector applications include manufacturing, mining, oil and gas, and water and wastewater treatment, though they can be applied in other use cases as well. Quality, reliability, and innovation are key selling points for this primarily business-to-business division. Consequently, the traceability of components used in manufacturing these electronic devices is crucial for maintaining customer relations and addressing any issues that may arise.

Traceability throughout the whole core business process from suppliers to customers is ensured by communicating the appropriate certification and reports from one entity to another. This thesis is focused mainly on a procure-to-pay part of the core business process. Procure-to-pay (P2P) in this case consists of Purchase Order Creation, Order Confirmation and Fulfillment, and communication of the testing documentation from different suppliers. Suppliers in this case do not change often and the trade between the supplier and case company is regular.

Testing documents and other quality-related documents are important since electronic devices can contain multiple different components that are included in the device. Data about suppliers' manufacturing, testing and materials used can be used by the company to ensure and observe the quality aspect of their components and possibly visualize possible problems. Completeness of quality data, in contrast, can ease the pending implementations of predictive maintenance to a component level. Predictive maintenance would improve customer relations and working efficiency. This case has an interesting undertone since our case company has provided the suppliers with testing machines and software to ensure components' performance to the electrical industry and quality standards. Every single component produced by the supplier is tested in the supplier's factory with case companies testing machines and software.

Suppliers' testing of the components is therefore standardized between suppliers and generates an abundance of measurements in a single data file. This is the primary source of the data pipeline under the investigation in this thesis. There are functionally two types of tester machines New and Old, and each one has its intricacies. The newer version of this testing machine automatically generates one XML file. XML file is then sent with data load client to a specific quality data server that processes and analyzes data. A newer version of ingestion is considered Type One for the rest of the thesis. From the quality server data is loaded to a general data storage which contains data from multiple different systems. These systems include, for example, manufacturing execution systems (MES), Enterprise resource planning (ERP), and customer relationship management (CRM). In this general data storage, different data sources can be combined easily.

Older testing environments differ slightly in that the testing results are generated as CSV files and then sent to case companies' inboxes. Then custom scripts are run on a network drive that extracts the CSV file and converts it to an XML file for processing. These generated XML files are then transferred to the Internal Data Load Client that pushes the test results to the quality server once a day in batches for processing purposes. Older Batch Ingested files are named as Type Two for the rest of the thesis. The quality server combines new and old results with a combined data table that includes one test result in one row. Data is then combined in the data lake with other sources and can be used for multiple different applications.

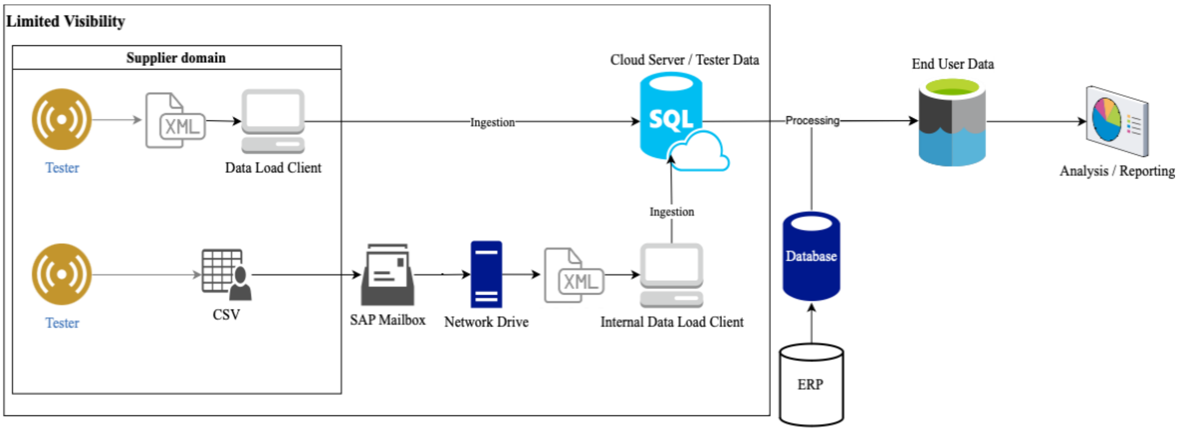

Figure 12: Illustration of Data Pipeline

Figure 12 above is an illustration of the architecture used by both new and old testing machines. The data pipeline in this architecture needs to handle batch ingestion and continuous data flow. The case company is slowly moving away from the old tester machines, but they are expected to stay for a while. Quality control and by extension traceability are important to the company from regulatory and customer support points of view, any data quality issues, or missing data should be addressed. Since the data's nature and purpose is to be always available if needed, the company has acted to ensure the performance of this data pipeline. A similar type of research could be conducted on different data pipelines, but this was a case of high priority and possible improvements. Limited visibility indication explains the parts of the architecture that are not accessible or modifiable at the current point in time. This means that no major modifications in that domain can be made, and this factor limits the possible solution domain.

4.2 Defining objectives for the monitoring solution

This chapter provides objectives and initial requirements to design and develop a data pipeline monitoring artifact and initial metrics to help evaluate the artifact. Inputs for the formal objectives and artifact requirements are composed of the data pipeline, data management, anomaly detection literature, workshops with key stakeholders, and evaluated and reworked with experts' interviews and end-user needs.

Challenges, theories, and best practices from the surrounding topics can be summarized as requirements for the artifact. Literature shows that many of the topics under data management are somewhat intertwined it would be naïve to only limit the artifact objective and initial requirement gathering to one singular topic without considering the potential implications of different topics. Therefore, combining noteworthy and relevant characteristics and challenges of data quality monitoring, data governance, data pipeline management, and data quality assessment, is necessary to create requirements and refine objectives for the artifact. The requirements gathered can be found in Table 9 and Table 10.

The problem that we are trying to solve with this new artifact is that there is no knowledge as of right now whether the company has a complete collection of the tester results or not. The case company is aware that some of the test results don't end up in a data lake where the data is usable for necessary people. The case company has identified the problem as most likely in the pipeline's earlier phases. To clarify this does not mean that there would be rows that have missing values, but rather completely missing data rows or tester results on specific component serial numbers. The most likely cause for the data loss is the tester's failure to generate the XML or CSV file. If the tester result generation script fails or performs unexpectedly, the tester result file will be corrupted and cannot be ingested to the data pipeline causing missing testing and traceability records for a longer period. Another potential causes of "Data leakages" are data load clients sending failures or exceptions in data processing during pipelines.

The first main objective is to analyze and inform the user of test results missing from the end user reporting. Literature refers to alerting as a possible fault mitigation strategy. Therefore, artifacts should include a feature to create customizable alerts for the stakeholders. While this may seem an easily solvable objective the real-life process makes this in practice complex. In the next chapter where we analyze the different approaches. Artifact is going to be implemented on top of an already in-use system to solve a specific problem and therefore can be seen as instantiation.

What about the quality of the non-corrupted tester data? The quality of data that is ingested properly into the pipeline is processed accordingly. The company has robust data quality control and monitoring throughout this data pipeline past the ingestion according to experts. Experts responsible for data engineering for this data pipeline are aware of data quality assessment frameworks and data quality dimensions. A data quality assessment framework was used to assess context-relevant data sources. Assessing the quality of all data companywide would be an extensive and tedious task and would not be possible during this thesis, therefore, we limit the assessment to a few select sources.

The second main objective is to create real-time monitoring of the status of the data pipeline for pipeline owners. This objective is presented both in the literature and the main needs of the company. The company can monitor in real-time currently some of the supplier data pipelines, but others are currently left with limited visibility. Real-time monitoring will help the company's responsible personnel to address potential faults, anomalies, or problems proactively rather than needing to solve escalated situations. Worst worst-case scenario could involve frustrated customers and the overloading of data professionals

We aim to create an automatized data pipeline monitoring tool and then demonstrate the applicability of this automatized tool to monitor one data pipeline. The designed approach will then be used for the other similar data pipelines and scaled upwards to the global multipipeline setting. The design and development process step will review similarly executed practical tools, solutions, applications, and frameworks. Based on the review, we will create a customized solution based on feedback from semi-structured interviews and the best practices identified from different sources. For metrics of the solution, error rates and data volume are chosen. The artifact's purpose is to improve using new and context-specific methods for an already known problem.

During workshops, it was determined that the metrics best suited to evaluating the artifact are alert counts, the correctness of real-time status and historical analysis, system load time, and extensibility to possible new sources. During testing, the scalability of the system is tested by overloading it with an extensive amount of data. Evaluating the artifact performance will also indicate if alerting thresholds should be changed or made dynamic.

4.3 Interview results

Regarding the development of artifacts and solutions as well as gathering knowledge of data quality, we held 5 interviews in total. These interviews were conducted after initial requirements, the literature review was conducted, and the development of the artifact was started. The interview guide can be found in Appendix 1. Interviews were conducted in different parts of the DSR project on purpose as a continuous source of feedback. Another continuous form of feedback implemented in this research was workshops with key stakeholders. Interview results give insight into the current data maturity context, organizational drivers, and new possible solution ideas for the main problems.

Throughout the interviews, it became clear that similar problems illustrated in the literature are common in this organization. All the interviewees valued high-quality data and were excited about this project. Common data quality issues in the organization included challenges like outdated data, data completeness, inaccuracies, missing data, human-induced errors, and latency of the data, which are summarized in Table 6. Complicating data quality issues many of the interviewees mentioned the large scale of the company. As a Global Company with a very complicated organizational structure standardization of data and data management practices would require huge investments and large workloads. Some of the more general level data quality problems like discrepancies in grouping and hierarchies between business, production, and Information systems make analyzing and creating new analyses much harder.

Table 6: Interview Results, Common Data Quality Problems

Data Quality Issue	**Possible Cause**	**Data Quality Dimension**
Outdated data	Disruptions in data flows, Manual updating of some sources	Timeliness
Differences between sources	Lack of data governance	Consistency
Missing data	Lack of adequate monitoring	Completeness
Errors made by humans	System design, Text Fields without imputation checking	Accuracy

More practical data quality problems introduced in the interviews were examples of logical errors during processing and rounding problems. Verification of the data related to some of the systems was also not as common but still appearing. Constant re-naming and reconstructing of organizational structure, annotations, or terminology was identified by interviewees as a factor that lowers data quality and causes misinterpretations. Even though there is still a lot of work that needs to be done in the organization regarding data quality, most of the common data was perceived as high-quality.

The next topic that was discussed in the interviews was about the future development of data quality. Answers from interviewees did not have as much variation as the common problem data quality question. Interviewed experts indicated a few key directions for future development. These directions proposed included: harmonization of data, unified data model, standardization of data processes, data life cycle approach, data quality by design, and cloud-based solutions. "*Upholding data quality cannot solely rely on the data specialists to fix, but rather every employee must do their part*."- Interviewee. Many of these hoped and needed development directions center around standardization and strongly center around data governance. Other organizational factors mentioned during the interviews were awareness and training needs for all employees and not just the ones responsible for data activities.

Data Governance was one of the themes investigated in the interviews. Comparing interviewees' answers to for example DAMA data governance framework can be concluded that the organization is still in the early stages or missing. The data governance topic was not very well understood or defined in this organization. Most practices under data governance are performed, but a lack of policies and standards regarding data is apparent. For example, initiates around metadata and metadata management are common. The practices are divided in organization in a way that does not complement each other. Some attempts to create standards and manage data have been made, but overarching data governance is still missing. Interviewees analyzed that the absence of holistic data governance might be due to the large size of the organization and the lack of higher-level support. The sheer complexity of organizations' information systems that are acquired from multiple vendors and some internally built would require those attempting to generalize data governance vast understanding, technical prowess, and large time investments.

Data pipelines were a well-understood topic by the interviewees. Around 90% of the errors and issues encountered are user error-induced errors, but still actual pipeline errors occur weekly. Interviewees noted that investigating issues and root causes in data pipelines is still a "very manual" process. This can take data engineers hours to find the problem if it happens before the cleaning and transformation. Data pipelines are managed by data orchestration tools to manage and configure the different parts. "Some data quality checks are integrated into the cleaning and transforming nodes and we have a good understanding of the available data"- Interviewee X. Each data pipeline in the organization is unique and built for a specific purpose, but according to organization experts, they all have similar components to the general model.

Interviews provided a few previously unconsidered approaches to the problem. One of the interviewees indicated that it would be possible to get unutilized logging analytics data from an external source that was previously thought to be inaccessible. This will provide us with an opportunity to analyze and utilize data that could be key to solving the main problem. Another proposed approach could be using checksums to verify that the data has reached the endpoint. Interviewees proposed using Power BI to create simple reporting structures that can be maintained by their owners and can be easily modified. The original artifact design idea of creating actual software and code with Python was not seen as optimal since the long-time maintainability might be inefficient.

After explaining the current monitoring objectives and requirements and possible solutions, interviewees provided additional requirements. These requirements are added to the list of initial requirements planned for the artifact. Requirements gathered from the interviewees were: Intuitive user interface, ease of use, visualization of failures and failure points solution, rigorously tested and validated, uses star schema, customized from already made solution, and must be maintainable by current resources.

5 Designing, Developing, and Evaluating data pipeline ingestion monitoring solution

When designing and implementing data pipeline monitoring tools and solutions, it's essential to start by clearly defining the key objectives and requirements of the monitoring system. Using the requirements and established objectives from the last chapter different solutions and alternatives are considered. DSR being an iterative process requirements and objectives may change during the design and development phase if new alternatives emerge. Once the objectives are established, the next step is to identify the key components of the monitoring solution and methods that complement them. This can include data collection mechanisms, processing and analysis tools, visualization, and reporting interfaces, and alerting or machine learning implementations. After outlining the components, it's crucial to define the data pipeline monitoring architecture, including the infrastructure and technology stack required to support the monitoring solution. This involves considering topics like data storage, processing engines, and integration with existing data pipeline systems.

5.1 Designing the data pipeline ingestion monitoring solution

This chapter will illustrate the design process of the solutions and considerations made between possible solutions. Initially, we had three approaches, but the expert interviews and discussion provided new alternatives to creating a data pipeline monitoring solution. Data pipeline monitoring solutions and strategies are a heterogeneous group of different practices from data engineering adjacent practice fields like data analytics, data science, and computer science. Examples of possible data sources for solutions are represented in the appendix. First, in this chapter, the possible data pipeline parts where data can be extracted are illustrated.

The solution can gather data from three different servers. These servers are the end-user server and two different processing servers. The end user server is essentially combined from multiple different pipelines and contains data from all parts of the company’s core operations. The end-user server contains the processed results of the tester generated among other factors. Other useful data that can be accessed from this server are manufacturing and purchasing data. The syntactic data quality of this end-user server is relatively good although some databases are not as standardized. End-user servers' data has been processed and is fit for use, although the granularity of data depends on the context, and groupings or aggregations must be considered when using this server as indicated by interviews.

The first of the two processing servers contains data about the processing and cleaning of the tester data, and unaggregated test results. The system is not internal consequently, access and visibility are limited even though all the data is owned by the case company. External management of this system does affect the possibility of developing this server without additional costs. This system already has data processing and ingestion monitoring in place, but not on a level that would satisfy the solution requirements. Batch ingested data, for example, is not status monitored although the stream ingested data is. The possible reason for the lack of monitoring on batch ingestion is the aggregation of all supplier data. From an external systems standpoint, all the batch ingested data is loaded from the same source. This system has logging data for processing and ingesting which can be used for further analysis. An example of the logging data generated by the logging mechanism can be found in Appendix 2.

Already developed solutions can provide and satisfy the requirements set for the solution. Since most of the data pipeline in question is already covered by data quality checks and validation and the problem lies in the data ingestion phase the decision was made to not proceed with possible external options. The decision was based on the resources, maintainability, and integration of the existing pipeline. Based on this also was decided that the final solution would be a monitoring dashboard where KPI, performance analytics, and results of anomaly detection algorithm would be integrated. Workshops provided plenty of approaches to ingestion monitoring. Possible custom solutions rely on getting quality metadata, logging data, or summarizing large datasets. Table 8 below summarizes the 5 solution concept designs.

Table 7: Solution Concept Designs

Name	Description	Data sources
Comparing Purchasing orders and Tester datasets	The concept relies on combining purchasing orders and end-user database tester data to seek out discrepancies.	ERP data about purchasing orders and stock movements, test result data for individual test results
Estimation algorithm	Creating an estimation algorithm using moving averages and integrating them for predicting possible missing loads. It would be based on the quantity of each supplier.	Test result data
Logging data analysis	Interviews highlighted the possibility of using logging data about file transfers in the ingestion phase of the data pipeline. This logging data can be analyzed to provide real-time status and analysis of different supplier	Logging data from down-stream
Comparing logging data and end-user data.	The concept relies on comparing and analyzing serial numbers from two different phases to measure completeness	Logging data, End-user datasets.
Using Machine-Learning	Using machine learning methods that are specialized to anomaly detection to gain visibility	Logging data, End-user datasets.

The first solution option considered was comparing purchasing orders sent to the supplier and tester data from the end reporting database. This solution would require modeling the data in a way that the tester data is aggregated to a serial number level. When the tester data is queried unedited it will be on a test level. The data table can contain multiple test results for one serial number. Even higher granularity of purchasing orders needs to be taken into consideration as well. Essentially tester data needs to be aggregated twice to be able to compare it with purchases. Supplier data quality was analyzed with a profiling tool, and the analysis showed that there were inconsistencies in data caused by human errors. Other challenges arise when creating a prototype and modeling the data model. The purchasing orders could not be reliably linked to a specific week or even a month which eliminates the requirements like real-time monitoring and scalability. A gap in production and goods receipt could be modeled, but it would only reduce the monitoring to a weekly level. The design was made, and this could be used for monitoring for longer time frames like months or quartiles.

The second solution option considered is using a time series estimation algorithm to estimate and predict the data loads. This solution would be based on only the tester outcome data. Time series prediction algorithms like autoregressive integrated moving averages (ARIMA) could implement and predict the amount of test results supposed to be ingested into the database. A fine-tuned time series algorithm could effectively predict if all data is ingested. Like all models that are trained on historical data, they are affected by changes in the surrounding conditions and context. ARIMA being a statistical method can help predict the expected values and identify significant deviations from these predictions. Consideration will be made since ARIMA can only capture linearity and more complexity cannot be modelled. One of the downsides of the ARIMA model is that it requires a lot of computing power so implementation would have to run inside a cloud computing service. The method would utilize automated alerts to the responsible team of any detected anomalies or breakages in the data pipeline. Additionally, it would incorporate data quality checks using Python libraries to ensure that all required data is successfully captured and processed

The third solution option emerged from the interviews held with company experts. Logging analytics were initially crossed off since there was no reliable data source and implementing the ingestion mechanism would be time-consuming. Interviews noted that the company has logging data of file transformations and transfers in a specific, restricted, and previously unavailable database. This database is in a limited visibility zone and is responsible for integrating and processing data from multiple suppliers. Logging mechanisms of the database display each processed supplier file and the status of the file among other features. In Appendix 2 an illustration of logging data can be found of the data that has been simulated to reduce the risk of breaking any security policies. Preliminary examination and analysis showed that this database is configured to retry storing when errors occur. Other accessible data from this database are run times and summarizations of data loads from specific suppliers. This data is currently not being used by the company therefore validation and data quality assessment must be done during development if this data source is used for the final solution.

The fourth solution option is an extension of the third option that emerged after a preliminary review of the data was conducted. Logging data contains many descriptive attributes and has low latency from processing to data being available. Those attributes enable us to achieve real-time monitoring for data engineers to monitor the status. During the preliminary review filename containing the serial number and creation date of the file was noticed. This file name stored in the database followed a standardized and system-created path for example ***C:\X\XXXXXXXXXX\FI\SERIALNUMBER_MM_DD_YYYY***. The filename was then parsed to possibly compare the serial numbers from the logging table to the end-user dataset. If the end-user data contains all the serial numbers of both failed and successful file processes from the logging data, then the completeness of the data can be validated. Data Quality raises its ugly head also with this solution option. After making queries and parsing the filenames some of the file events had filenames that could not be parsed. Unparsed filenames did not contain any serial number or creation date information and were examples of corrupted data files. This could not be seen with the standard profiling technique to observe the data quality since the column was not empty or would contain a different number of characters. These events will go under further review and analyzed based on the processing date and tester machine.

The fifth option would be using machine learning to predict or label the anomalies. Using historical data of either end user or file event data anomalies could be predicted. Machine Learning algorithm could be a support vector machine that is effective for classifying network anomalies. The most effective machine learning algorithms for anomaly detection vary depending on the specific application and data type. Supervised learning algorithms like SVM and deep neural networks are highly effective for network anomalies. Unsupervised methods, including VAEs, CNNs, and autoencoders, excel in time series and network anomaly detection. Hybrid approaches and ensemble methods, such as combining KPIs with VAEs or pattern mining with GNNs, offer enhanced performance by leveraging multiple sources of information. Anomaly detection applications are context-specific, but a time series or network anomaly detection algorithm could be modified to fit our needs.

The design of the artifact is the combination of the possible solutions. It was decided during a meeting with key stakeholders to not continue with solution options one and two. The first option was not feasible since the accuracy and timeliness of the data would not have allowed us to reach the requirements of near real-time monitoring. Monthly level analysis would not be useful in practice since at that point the potential leakage should have been noticed by users. The second option was made obsolete by the emergence of logging data. Although time-series analysis could be performed successfully in this context it would be time and resource-consuming without a realistic maintainability factor. Proceeding on with the work it was decided to move forward and combine options three, four, and five. Rather than fully implementing the machine learning algorithm it was decided during the meeting that this part of the solution would be a demonstration rather than something that is used continuously. The third and fourth options would be combined to create a monitoring system that utilizes and analyses both near real-time and historical data to provide a comprehensive view of this pipeline.

Next readers are presented with design specifications for the monitoring system and given as much information as possible by the companies' publicity policy. Detailed descriptions of the data sources, architecture, or other factors that could harm the company will be intentionally left out or abstracted. Abstractions are made to a point where the core functionality or concept can be understood without knowing the finer details. This design specification will include key users, user needs, requirements, example data sources and modeling of data, analysis methods, KPI and other metrics, alerting and alerting thresholds, visualization types, access control, artifact performance indicators, and maintenance and support factors.

Stakeholders and Users

Stakeholders in this monitoring report/dashboard are the process owner and designated solution owner. Users for this solution are companies' data engineers, data architects, data analysts, process owners, and managers.

User needs

Users need a dashboard to monitor the ingestion phase of the data pipeline. They will need to see the historical analysis of data load failures and causes. They need to know if any tester result is yet to be processed through the data pipeline and analysis of the disruption. Users are also interested to see if the disruption occurs on the serial number level, tester level, supplier level, or pipeline level. Users need to know if the data pipeline automated retry mechanisms perform as intended or if there are discrepancies in this factor.

Requirements

Requirements were gathered from literature, workshops, and interviews. Requirements were designed to illustrate the solutions' functional and non-functional features. If the final designed dashboard follows the requirements, it could be considered satisfactory and successful instantiation. Requirements are divided into functional and non-functional requirements as is common practice in software engineering.

Table 8: Functional Requirements

Requirement	**Explanation**
Real-time Monitoring	Continuous monitoring of data pipeline performance in real-time.
Customizable Alerts	Configuration of alerts based on specific performance thresholds or anomalies.
Informatic Metrics	Tracking of various metrics including throughput, latency, error rates, and data quality.
Historical Data Analysis	Analysis of historical performance data for trend analysis and forecasting.
Integration with Existing Tools	Seamless integration with existing data pipeline tools, monitoring systems, and analytics platforms.
Data Refreshing	Data should refresh within a specific timeframe.
User-Friendly Interface	Intuitive interface accessible to technical and non-technical users for configuring alerts and analyzing data.
Predictive Analytics	Utilization of machine learning and predictive analytics for proactive issue detection and optimization suggestions.
Extensibility	Ability to add new data sources, metrics, and monitoring capabilities as the data pipeline changes.

Functional requirements are making sure that the monitoring solution has everything planned out for development and all the different features that the company has requested. These requirements have been modified during this thesis whenever new issues or information has been gathered. Implementable features in the final solution include real-time monitoring, alerts, informative metrics, historical data analysis, integration with existing tools, automatic data refreshing, a user-friendly and pleasing interface, some predictive analysis, and the possibility of extensions in the way of new data sources, metrics, etc.

Table 9: Non-Functional Requirements

Requirement	**Explanation**
Scalability	Ability to scale with growing data volume.
Security and Compliance	Adherence to security best practices and regulatory compliance standards.
Resource Optimization	Insights into resource usage patterns for optimizing data pipeline efficiency.
Documentation and Support	Comprehensive documentation, training resources, and technical support for implementation and troubleshooting.

Table 10 shows the summarization of non-functional requirements defined for the solution. These requirements included scalability, security and compliance, resource optimization, and documentation. Compliance includes the need to satisfy data governance factors.

Data Sources

The monitoring dashboard could use both on-premises and cloud-based databases. This causes some technical limitations on the analytical platform for aggregating the data and the need for gateways in place for on-premises databases. These gateways are created by companies' experts to make access to the on-premises database available. Different data possible data sources are gathered below in Table 11.

Table 10: Possible Data Sources

Source System	**Tables**	**Cloud / On-premises**
Source 1	Logging data, Supplier information	Cloud
Source 2	Tester data, Supplier information	On-premises
Source 3	Machine Learning algorithm results	Cloud

Queries to Source 1 include aggregating data from multiple tables and multiple queries for different purposes. For near real-time monitoring Structured Query Language (SQL) queries using indexing will fetch the latest file-events for each supplier and this is to be implemented in the most efficient way to reduce the waiting time of the user. Near real-time monitoring cannot be implemented with scheduled dashboard refreshes for example: every hour as the rest of the queries but will be run every time the user opens the dashboard, refreshes, or filters the data. In use cases other than near real-time monitoring second query is created that loads the data into the model. The second query results will be illustrated in the appendix but will include approximately 1000-10000 times more rows than the near real-time query and do necessary parsing and calculations.

Source system 2 uses loading queries like the query of source system 1. Loaded queries are then refreshed automatically on specific time intervals by the cloud platform that the monitoring dashboard is created on. The platform can also optimize the incoming queries to refresh only when changes have been made, which improves the overall performance, and timeliness of data and helps lower the refreshing costs. The Machine Learning model, if implemented, is on the same platform as the monitoring dashboard, so the connection between these two parts of the solution will be seamless.

Analysis methods

Analysis methods used in this monitoring dashboard are customized to the needs of the company. Near real-time monitoring, analysis is performed as traffic lights where traffic light color indicates the latest status of a specific tester machine. Initial color coding is red for errors and green for successful file events. Visuals will also show other visual indications than color to ensure accessibility. Historical analysis of logging data will be conducted with metrics and KPIs. These indicators for logging analysis are event failure rate, retry count, number of unique files processed, and file processing time.

For qualitative features, we can analyze the error codes to better indicate the problems each tester machine is facing using Pareto analysis. The failure rate is calculated by dividing the errors by all files processed in a specific time frame. Logging data from source 2 is aggregated to a test result granularity and compared to data source 1 test result with cross-checking and discrepancies are reported. Incomplete records from the logging table are reported on a specific page and as much identifying information as possible about them is gathered.

Visualizations

Visualizations are reviewed with key stakeholders during the review of prototypes, and the creation of custom visuals is also considered at that time. Custom visuals are unlikely to be implemented since they would require additional maintenance resources. Visualizations used follow the company's best practice guidelines. These guidelines ensure that data is visualized in a way that best represents the data, and is clear, easily digestible, and informative. Each graph and calculation should represent reality and strive for accuracy. The principle of "less is more" is applicable in this context

Security And Access

Data governance and data management outline the security and access to this tool. The platform's standard procedures are integrated. Visibility of the results is limited to individuals who A) already have access to the data, B) have a real need and significantly benefit from the tool, or C) are accepted by the tool's owner to view the results. Most importantly, the tool is limited to internal use only, and results may be shared with suppliers only concerning their tester machines. Sharing incorrect supplier results with another supplier can result in serious repercussions. Security is enforced by row-level security, ensuring users with limited visibility see only the data applicable to them. Access management is handled by platform-wide security groups, which do not need to be reinstated. This access grouping reduces the administrative and support work required when, for example, new employees start, or organizational changes occur.

Performance and Scalability

The performance of the underlying data model will be measured by data model refresh time and visual refresh time. The platform has already implemented metrics to measure data model performance, which is accessible to specifically named personnel. Refresh times depend on the data model size, optimization of SQL queries, other concurrent refreshes, and capacity. A refresh time ranging from 1-5 minutes, considering all the queries this data model requires, can be considered acceptable. Performance optimization will be conducted during the development and testing phases. The scope of the prototype will be limited to a singular supplier with multiple tester machines. During the evaluation phase, the prototype will be tested with all available suppliers to measure scalability. Due to the standardization of formatting in both sources, adding new suppliers will not be an issue.

Support and Maintenance

Supporting and Maintenance of the tool will be transferred to the commissioning organization when the stakeholders have deemed the solution to meet the requirements. Transferring will provide detailed documentation of the architecture, logic, and reasoning created in the dashboard and data model. All calculations, aggregations, queries, and data sources are documented in this final documentation and classified as internal. Therefore, this documentation will not be publicly available or included in this thesis.

5.2 A general framework for implementing data pipeline monitoring

Here practitioners are provided with a general framework for implementing monitoring into an existing data pipeline. Framework doesn't recommend any specific already made tool or solution since this decision is context-specific and depends on the current technology stack in use. Rather the framework outlines topics to consider when implementing a monitoring solution and divides a multifaceted problem into more digestible parts. The framework seen in Figure 13 is based on the indications from literature, expert interviews, and empirical observations.

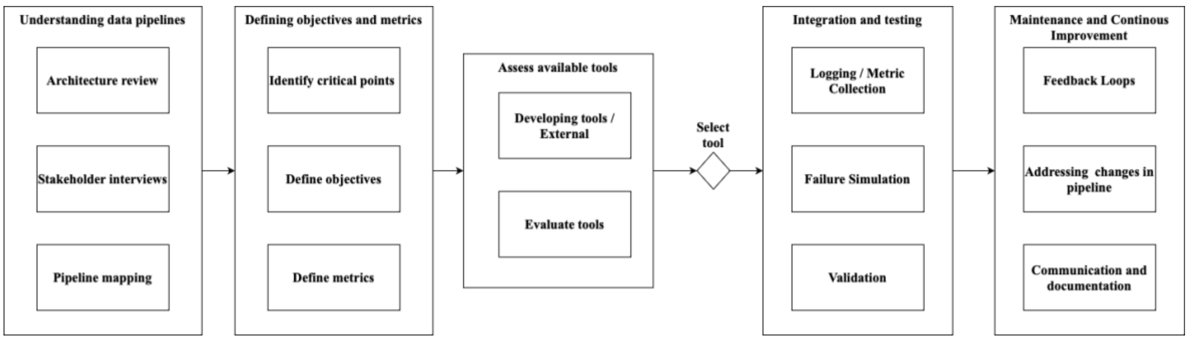

Figure 13: General Data Pipeline Monitoring Implementation Framework

Understand the Existing Pipeline

As usual, creating or implementing new solutions to an existing pipeline should start me examining the current state and understanding topics affecting it. For data pipeline solutions the current data pipeline structure should be mapped, and other architectural designs should be made clear. Understanding the pipeline structure and behavior of the underlying business process is crucial since it will limit the possibilities regarding gathering data or using a specific technology stack. Architecture review centers around gathering and reviewing documentation and information of the current data pipeline architecture Interviewing stakeholders should provide an understanding of key problem points, critical data points, and existing monitoring mechanisms. Pipeline mapping focuses on a holistic understanding of the pipeline and identifying data sources, processing stages, data storage, and data sinks.

Define Monitoring Objectives and Metrics

Defining monitoring objectives and metrics step should include formally defining the objectives of the solution, but first critical points where the monitoring is implemented should be defined. Critical points are the parts of the pipeline that need the monitoring most. Critical points can be assessed using many decision-support tools, but generally, they should be the parts that affect performance, reliability, or data quality the most. Therefore, might be useful to include stakeholders and experts to determine what are the most essential and high-priority data pipeline parts that need monitoring. Re-structuring or changing the used technology stack might be needed if pipeline monitoring is implemented throughout the pipeline, but if the objective is to gain visibility to a specific part of the pipeline it might not be necessary. After identifying critical points in the pipeline monitoring objectives should be made in a way that they are realistic and aligned with the general goal of the project. Objectives define the metrics that can reflect different attributes like performance, reliability, or data quality. This step might include also prioritizing monitoring objectives if there are multiple. Examples of metrics can be data latency, error rates, throughput, data record completeness, and accuracy, but specific needs might need different metrics.

Assess available tools

As previously mentioned in this chapter there are many already constructed tools whether commercial or open source. Tool implemented or developed should be selected based on cost, features, and fittingness to use. The cost of monitoring tools is a complicated factor, which has not been at the center of this thesis. It is important to note that even though using commercial solutions might seem like a more expensive option, the time used for developing internally can ultimately cost more. External solutions can provide wide coverage, but all the features might not be useful, and seldom are plug-and-play options that still need some resources from experts. IS project management literature provides plenty of usable options to assess the cost-effectiveness of the project. Other evaluation criteria that should be used are fittingness of use and features. The solution option should consider the key capabilities to measure the metrics. Note that most modern platforms offer analytical solutions or logging mechanisms which might not be utilized yet.

Integration and Testing

Integrating the monitoring tool is not a straightforward process and is highly dependent on the selected tool. There are countless different options for implementing the new tool depending on the tool ranging from querying logging data from logging mechanisms to a report to re-building the entire pipeline on a new platform to satisfy the objectives. Regardless the defined metrics should be collected and analyzed. Some platforms can even support implementing custom metrics written in specific programming languages. As demonstrated by the literature, anomaly detection mechanisms and logging mechanisms can provide information beyond the basic statistical methods by utilizing machine learning. Testing should be done as any other system implementation to ensure monitoring's effectiveness. Testing can be carried out with simulations of errors or in a less time-sensitive timeline by intentionally creating errors to test the monitoring performance. Depending on the use case analyzing and validating the monitoring effectiveness can utilize different methods but they should assure the accuracy of metrics and alerts.

Maintenance and Continuous Improvement

The final step of the framework consists of a feedback loop, addressing changes in the pipeline and communicating the solution. A feedback loop should be established to continuously improve the monitoring solution. Dedicating time to ensure metrics or machine learning models still provide accurate and intended results as time moves forward is essential to a high-performing solution. Feedback from key stakeholders can also provide new metrics that could be used for measuring the pipeline more accurately. Changes in the data pipeline like changes in sources or processing are likely to occur and therefore regular reviews should be held to adopt monitoring to the environment. Finally, appropriate documentation and communication should be established to ensure effective changes and maintenance. Accurate documentation should be written in a way that does not require an engineering degree to be understood and will help new stakeholders understand the data pipeline monitoring picture and decisions made.

5.3 Developing and implementing the data monitoring solution

Developing and implementing the data monitoring solution to existing architecture is demonstrated in this chapter. The solution was developed using SQL and data analysis/visualization tools. First, this sub-chapter illustrates the architecture overview and key components and their roles. After that file event logging analysis module is explained thoroughly. Finally, comparisons between data sources and the overall outcome are explained.

The monitoring system was designed to be integrated into a separate already in-use cloud platform where companies' other analytical solutions are already located. Since the data pipeline monitoring solution was not integrated as part of the data pipeline but rather queries the relevant performance data it used less processing capacity than integrated solutions would. This solution design where the solution is located outside of the actual data pipeline was necessary to satisfy the maintenance and support requirements. The solution consists of a pipeline monitoring report which has a combination of historical and near real-time data analysis as well as of alerting system shown in Figure 14. The cloud platform enabled built-in notifications based on analyzed and processed values like error rates.

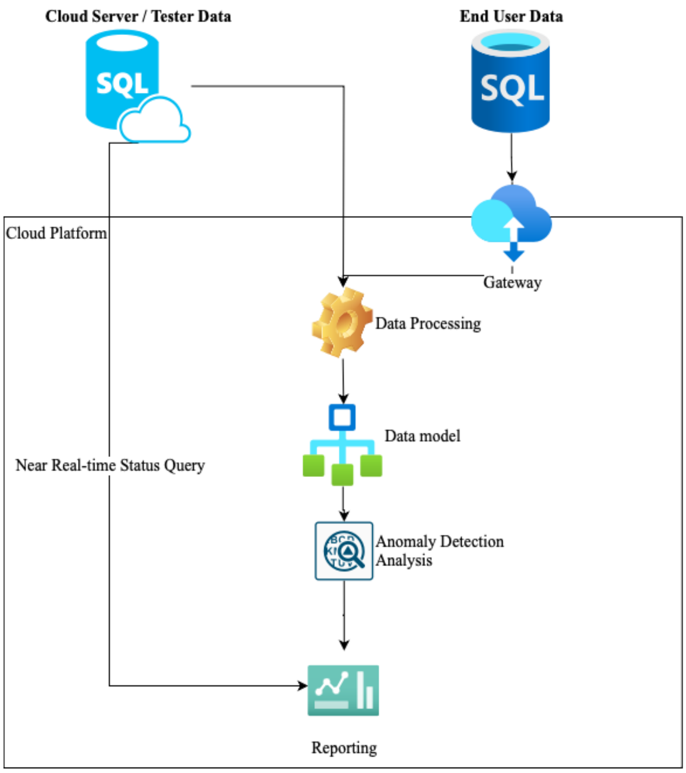

Figure 14: Solution Components

A solution was created with different databases where the data was queried. Data was then parsed and cleaned for analysis. No rows were omitted since end-user data was consistent and cloud servers logging data had to be analyzed and assessed thoroughly since indicators of missing data files or anomalies could be in columns or rows that would normally be cleaned. Examples of the usually processed values could missing or non-unique values. After that the data was modeled and a picture of the final iterated data model can be found below in Figure 15.

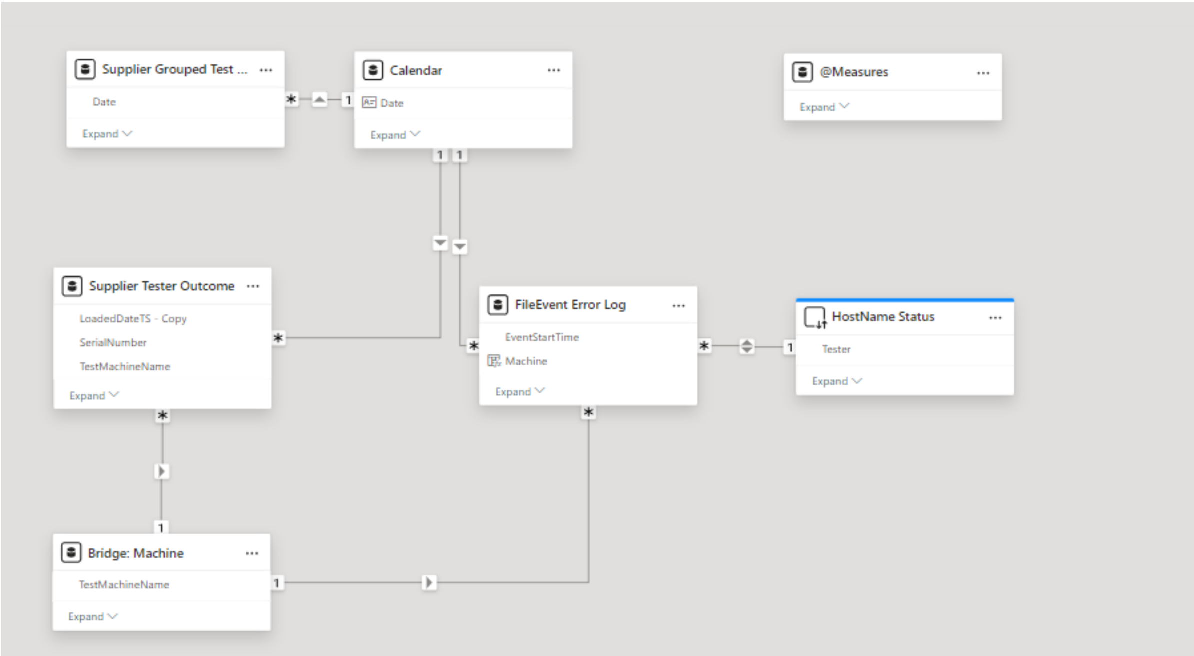

Figure 15: Data Model

The data model needed some usual standard dimension tables like a calendar table that connects all other models' tables' dates to it and the need for a bridge table was apparent. Analyzing between two tables with many to many connections is frowned upon in the industry and does not follow the company's best practices, therefore a bridge table called "Bridge: Machine" in Figure 15 was created using a simplified and optimized query. After creating the data model with industry-standard star schema and only having one to many connections, we were ready to proceed with analyzing the data. @Measure table contains all the calculations used in statistical analysis like moving averages.

Dashboard

The created dashboard contains key performance indicators of the data ingestion performance as well as the numerical indicators for potentially missing files. Key performance indicators were composed of specific time frames suited for each indicator. The average error percentage was represented on three different levels: past week, yesterday, and today. In addition, these KPIs also contained calculated differences to the whole population's average error rate shown as a change in percentage points. Later after the initial version of the dashboard seen in Figure 16, more KPIs were added: latency and throughput of the pipeline. Latency was calculated for each serial number by comparing timestamps of file events and load times of the end-user database. Latency was after that averaged between individual files relating to the same serial number since serial numbers could contain between 1 and 16 different files. Latency was then summarized to similar time frames as the error to give insight into the possible changes. The throughput of files ingested was simply calculated by counting all successful files processed.

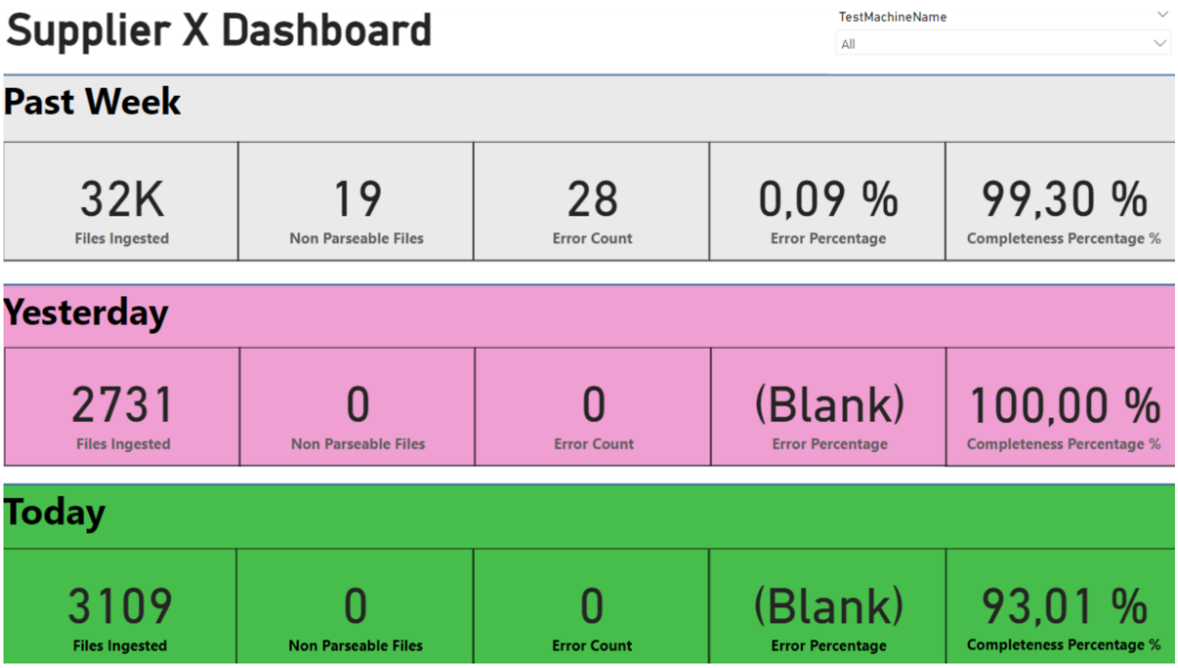

Figure 16: Supplier X Dashboard

Other indicators that the final dashboard contained were the percentage of serial numbers present in the ingestion phase but missing in the end-user database seen in Figure 16 as the completeness percentage. Due to some of the data being ingested in batches missing data file count was designed to look to yesterday rather than the day when the data refresh was conducted. In Figure 16 the 93,01% completeness percentage is an indication that batch files are not ingested yet. Batch ingested data is not updated to the end user database continuously limiting the real-time analytics for this indicator. A more detailed comparison of the completeness of records and the analysis is demonstrated in the comparison of records section. The missing data file count was 0 for the analyzed supplier on those files available for analysis.

File Event Logging Data Analysis

Source 1 included the logging data from the test ingestion was useful in trying to work out anomalous or faulty processing of data. The analysis and reporting platform was directly connected to the data source 1 using already-built connectors. This connector enables querying the logging data table with SQL and can be used to both querying data directly when refreshing is needed to a visual without storing it to cache and with importing. Importing means loading the data to the model with refreshes and does not need to load again every time filtering or other modifications are made. Choosing between importing and just querying data on demand depends on many factors. To achieve both near-real-time monitoring and demanding historical analysis both methods were used. In the data model picture live query “table” is indicated in blue. The database used for gathering file event data had other related tables that could be connected to the file event fact table, but funnily enough on those rows that included errors key column data was also missing. Therefore, getting more information from this database was unfruitful.

Observation of the filename in new style type one case containing serial numbers and unique file names meant that useful data could be gathered by parsing. Parsing was performed by using regular expressions (regex) and integrated into SQL query. From old-style files that are CSV-based even more parsing is necessary since the name of the source is the same for all the testers that still send old-style files. Tester machine data was parsed also using regex alongside the serial number and the actual filename. An example of the final SQL query to better illustrate can be found in the appendix. Parsed data was loaded to the data model and was not aggregated. Parsed data was then profiled using built-in techniques to understand the data better before visualizing. Profiling techniques revealed that some of the type one data had missing serial numbers on specific dates. On average file events that had errors were located at both ends of processing time distribution. In other words, error events either ran only a few milliseconds or took significantly longer than successful events.

First analysis is needed to visualize the error percentage as a function of time. This was performed to better understand the nature of errors and whether they were evenly distributed timewise. Expectations of company experts were that some errors occur consistently, but higher error rate spikes could indicate disturbances in normal data pipeline performance. The first analysis was narrowed to the last six months and presented on a daily level. The analysis confirmed both expectations. Data pipeline ingestion errors consistently occurred with error rates ranging from 0-1% during six months. Outlier days of the analysis divided significantly from the mass ranging from 5-38% percentage. During a three-day frame in the middle of

February nearly half of the file ingestions attempted failed. Other outliers did not happen near February. These outliers will be ingested in more detail later to determine if these problems were examples of performance errors that should create an alert to personnel.

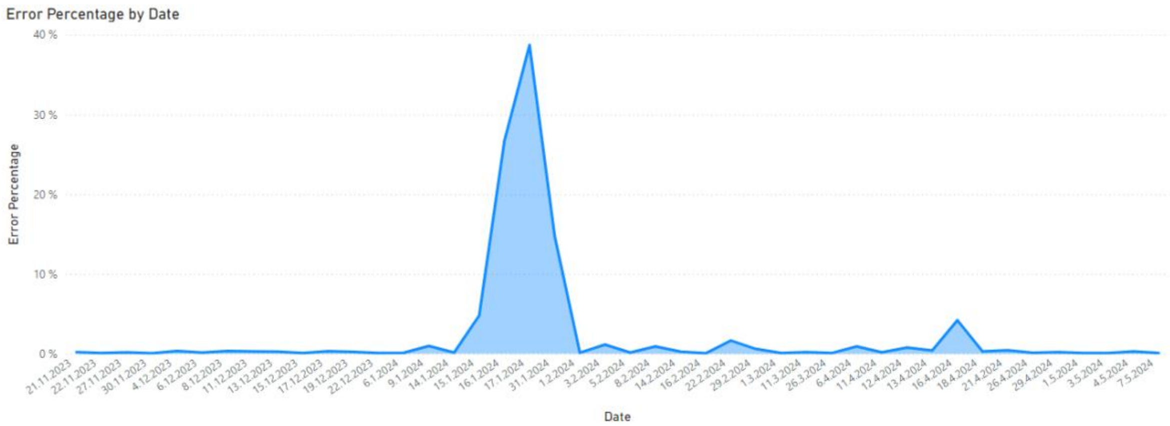

Figure 17: Error Percentage by Date

During the time frame, 0,25 % of overall ingestion attempts fail where daily percentages can rise to nearly 40 % as seen in Figure 17 above. The daily mean error percentages were 0,19 %. When a longer historical period was observed results indicated similar trends.

The second analysis included in the final solution was the frequency of ingestion retries on failed files. A measure used for this pipeline performance analysis was retry count. Retry mechanisms trigger when file parsing or ingestion otherwise fails. Overall, during the time frame retry mechanisms have triggered 2196 times compared to 660 individual errors. This means that during the time series on average file failing to be ingested is tried around 3,33 times before successfully being ingested or left unprocessed. This frequency analysis was conducted with the same data as the error percentage analysis

Comparison between databases

During the next analysis, previously mentioned outliers were filtered out of the data, since these error outliers did not contain any serial number. Rows with missing serial numbers did not have any other identifying factors, therefore rows could not be compared to the end-user database. In the last six months, 48 of these events were distributed along different type one testing machines. Outliers indicate that something went wrong during the generation of these files and the generation should be assessed. The completeness of companies' records with these outliers mentioned could be verified by asking suppliers for their production amounts in a specific time frame.

Comparison analysis between records was done by summarizing the data on a serial number level counting unique files for each serial number and comparing the records. The file event table had unique identifiers in the file path column as shown in Appendix 1. These unique identifiers were parsed, and necessary data was used for the summarization. The summarized data table from the file event table showed the count of unique files tried, unique files successfully ingested, the sum of the tries for the serial number, and the sum of errors for the serial numbers. End user data was similarly summarized to serial number levels and unique test results were counted. After summarizing data from the sources on the same level tables were connected via serial numbers to be presented in the same visualization. Their percentage was calculated at a weekly level representing the completeness of records.

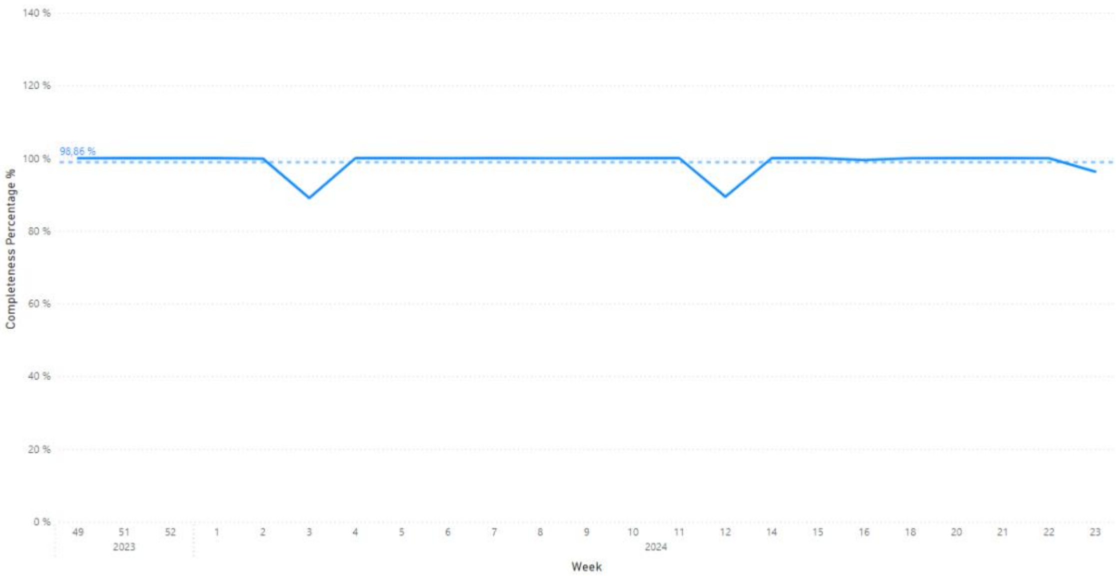

Figure 18: Weekly Completeness Percentage

The actual weekly completeness analysis comparison in Figure 18 shows the percentage test result file quantities between the sources. The difference between the sources is constant 0 which would indicate that all rows that have reached the ingestion phase have been successfully loaded and there is no actual data loss during the data pipeline processing during the last six months. Figure 18 shows that on a weekly level, some discrepancies occur but are fixed in some amount of time. The average completeness of records on a weekly level is

98,6%. This is where the outliers become much more interesting since they could be the actual usable indicator of dysfunction in the data pipeline. Also, another factor to observe more closely is the retry mechanisms implemented. Closer observations should be made by analyzing the time between individual file error logs and successful logs. Many factors could impact the completeness of the end-user database and as time constantly moves forward the earliest errors or missing files are more likely to be added.

Some other interesting observations that were made when comparing the two datasets were that from a file event standpoint, it could be seen that some files were ingested multiple times successfully as well. Multiple successful indications would result in duplicates in the end user database if duplicates were not processed correctly. The end user database did not contain any duplicate rows, which proved that the data quality mechanisms in place worked. Multiple successful ingestions could also suggest that ingestion errors in this data pipeline were fixed by intentionally re-ingesting all data files from a specific timeframe. Multiple successful ingestions were 0,05% of the population ranging between two and five successful ingestions.

Real-time monitoring

Real-time monitoring was implemented with live query to the database querying the last file ingestion attempt. In more simple terms each time the user opens the monitoring report the latest data is queried from the database. After the last ingestion attempt status and possible error code were queried from the logging dataset for each tester, and query results were visualized as traffic lights. Conditional formatting of traffic lights visualization is the following green indicates a successful ingestion and red indicates that errors happened in ingestion. For more information on the specific tester machine a drill-down option was created to satisfy the observation needs on a daily level and to express the context better.

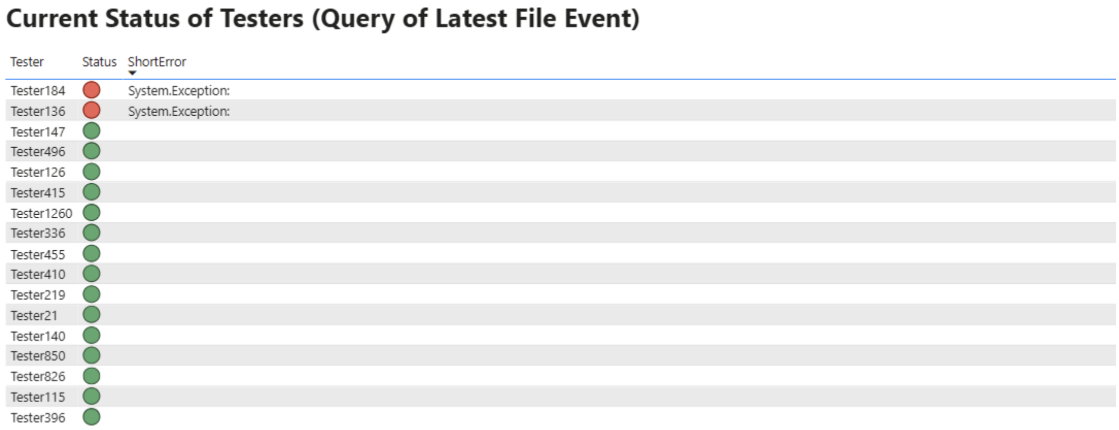

Figure 19: Traffic lights (Real time monitoring)

An example of the visualization of real-time monitoring is represented above in Figure 19. The visualization shows users clearly which testers have currently failed their last ingestion with red and what error code was logged. Logged error code determines the error type further helping with debugging. The most difficult part of implementing this part of the solution was to optimize the live query. Long waiting times on the query may cause frustrations in users resulting in a less user-friendly solution. Query was optimized by using indexing and sub queries ultimately reducing the query times.

While a similar visualization was previously in use this solution added the type 2 or CSV-generating tester to the picture. Previously only those testers that generated XML files were included due to technical limitations. This part of the solution is a clear improvement on the previous tools and creates a more holistic picture for real-time monitoring. A holistic view was otherwise created by including all the suppliers in this part of the solution deviating from the scope for the rest of the solution. The reasoning for this was that the previous tool in use had all suppliers as well.

Alerting mechanisms

Analyzing and developing the solution logic for alerting was much more apparent. The alerting mechanism was designed to follow specific metrics or key performance indicators. After analyzing the data further was decided that the final solution should contain multiple alerts. First alerting mechanisms in simple terms followed datasets for files that were completely corrupted and file information could not be parsed. The second alerting mechanism was designed to trigger based on the daily error rate. The third alert followed the completeness of the end-user records and triggered if the day before had a deviation of percentage or more between records.

The first alerting mechanism in detail is rule-based. Rules or conditions for trigger were simply the following if during refreshing the logging mechanism dataset one or more of the corrupted files occur alert is triggered, and a notification is sent to the process owner. Corrupted files simply indicate that something out of the ordinary occurred during the generation of XML or CSV files and the data was lost. Alert will send the remaining information of the file to a process owner who will need to validate whether to act straight away or open the monitoring solution and find more detailed information. The file will most likely need to be generated again and the supplier will be notified by the process owner of potential data loss and disturbance. Of course, the alert could be sent straight to a supplier, but it was decided to have a manual step of validation in between to reduce potential false positives.

The second alerting mechanism followed the daily error rate to determine the performance of the streaming sources performance. The error rate was calculated by dividing the processing errors by the total processing attempts made. Alerting was triggered when the percentage reached a threshold of 5% but was later changed to a dynamic threshold. Where this alert monitor's error rate percentage it was determined that during normal operations some errors are likely to occur during ingestion and processing. Thus, the dynamic threshold was selected to better reflect the actual reality and account for changes in the pipeline. The dynamic threshold is calculated as the weighted moving average of the error percentage from the past six months. The weight of the monthly error percentage is based on the difference between dates favoring more recent dates.

The third alerting mechanism is used to monitor the completeness of the end-user dataset based on the file event dataset. End-user dataset should have all the rows that were processed each day. Because part of the pipeline uses batch ingestion this alerting was deemed accurate enough to observe the completeness on a daily level. Batch ingestion happens usually in the middle of the night this mechanism is run on a schedule each day at 8 AM to assess yesterday's completeness. The daily level is in this case enough since the failed ingestion and processing are often successfully re-tried. To eliminate pre-mature alerting the daily alerting scope is better than hourly. The first alert already covers the completely corrupted files this alert is more tailored towards potential ingestion errors and corrupted files are omitted. The alert is triggered if the count of unique processed files from file events doesn't match the amount of test results in the end-user database. By default, this alert will trigger with even one inconsistency between sources but can be later changed with easy-to-use parameters to suit user needs better.

5.4 Testing and evaluation of the data pipeline monitoring solution

The data pipeline monitoring solution was tested with simulated intentional breakage of pipeline and error creation, which provides us information on whether the implemented alerting mechanisms work as intended and whether analysis reports showed correct results. Since the data pipeline has a critical role from the business standpoint this breakage was simulated by generating data corresponding to a data pipeline ingestion failure. Simulated breakage provided the following results. The alerting mechanisms worked as intended as the solution was refreshed automatically and the real-time monitoring showed the right error codes and tester machines. Alert triggering inputs were designed for both the first and second mechanism and the third trigger was triggered the next morning since those files were not ingested to the end-user dataset. Error rate percentage showed a six percent spike in error rate correctly. Testing prompted a design change since every half an hour as refresh happened alerts were retriggered. A mechanism was created to check if an alert has already been sent with similar specifications to the necessary personnel.

Testing the final solution included similar topics to the evaluation while the evaluation had higher levels of perspective. The final solution form emerged as a report from a testing point of view practical things to be tested are data refreshes, visual unity, representability, and performance. Data refresh latency was tested with automated refreshing in the cloud platform although the datasets loaded to the model were relatively large. SQL queries were optimized in a way that the whole refresh took an average of 2.5 minutes with refreshes ranging from 1-10 minutes. The reason for the variation with refreshes stems from the difference in the hourly platform load. No errors in data refreshes were encountered during a two-week observation period. During testing scalability of the system was tested by overloading it with an extensive amount of data and performance dropped significantly when approaching approximately 100 million rows. The normal amount of data loaded due to limiting the queries to a specific 12-month timeframe was around 20 million rows that performed efficiently. Testing the alerting mechanism indicated that the alerting thresholds initially set were too high and they were brought down from 2% to 0.5%.

Visual unity on the other hand refers to unity between report pages and in the visualization. The final solution used a company template to match existing reporting and time was spent on making the report easy to use and accessible to colorblind users. Overall key stakeholders were happy with the visual unity and representability of the solution. The performance of the solution was preluded with the data refreshes but using performance metrics like initial visualization load time and filtering load time gave us concrete indicators. Visualization load time on average was 500 milliseconds and a major outlier was the real-time monitoring page that live queries the cloud server. Real-time monitoring took on average 5 seconds to refresh which was considered acceptable and could be optimized further in the future.

As usual, the formal evaluation of the artifact is based on the key requirements for the solution and mainly on how well it meets key objectives. This evaluation also summarizes the key functionalities and effects of the created practical artifact. Two main objectives were to have data pipeline status monitoring and assess the record completeness. Both objectives were complemented with alerting mechanisms.

The first objective was to include real-time status monitoring which had a basis in both literature and practical needs. Real-time monitoring was implemented with live query to the database querying the last file ingestion attempt. The purpose of this feature was to generate live visibility of ingestions happening and this feature was accomplished by the traffic lights. For example, if a data engineer or process owner was alerted to a possible breakage, they could see the live status and errors of each tester machine. Status and errors help personnel to better assess the situation and help with resource allocation to fix to problem.

Customizable alerts were not accomplished per se since the alerting in the final solution is not yet customizable but rather based on logic explained in the previous chapter. Alerting mechanisms were implemented in the same cloud service to allow the creation of custom alerts for knowledgeable users. Three current alerting mechanisms provide key stakeholders with alerts to assess and act on specific errors and provide coverage of main objectives. Although these alerts are not yet customizable, they are effective for the solution extending the real-time monitoring and completeness analysis.

The dashboard covers the main metrics used and determined objectives for the pipeline status monitoring and record completeness. Users can quickly assess the situation with the results shown by the dashboard. Therefore, the dashboard is a valuable addition to the report and gives a general picture of different analysis results. The dashboard is informative and fills the requirements. User feedback provided stated that the dashboard is usually the first or second thing they use after receiving an alert. The dashboard provides informative KPIs of the data pipeline's performance in specific time frames. The time frames can be adjusted by users filtering and saving to filters to adjust to the users' needs. Calculations and logic used are annotated on a separate report page to increase user friendliness and ease of re-use.

Monitoring the completeness of records was one of the key objectives to accomplish. Completeness was monitored by comparing the data of two different points in the data pipeline and indicating discrepancies. While the solution was designed to be simple this was revolutionary for the company since the data form logging mechanism was not already in use. While this part of the solution design is not anything exciting from a theoretical standpoint it is effective and does not require making predictions.

To summarize the testing and evaluation the accomplished artifact fulfilled the objectives and most of the requirements. While the predictive analysis was one of the requirements this feature was not realized since the key stakeholders saw no gained benefits for using it. Developing the solution on a pre-made reporting platform satisfied many of the requirements of stakeholders regarding further development and scalability. Using a pre-made reporting platform as the basis for the solution entailed extending the same solutions to other pipelines or suppliers. Of course, some modifications are needed when copying the solution to other pipelines, but logic and a general model can be used to satisfactory results. The assumption is that these pipelines have logging mechanisms already implemented. This concludes the evaluation of the solution, and this chapter is extended in the next chapter where the results are discussed.

6 Results of data quality and data pipeline monitoring in manufacturing company

Nearly all companies have data pipelines or constructs that resemble those in use. Data Pipelines are already widely used in organizations and the literature on usages in different applications is increasing. The current literature does not provide many examples of anomaly or error detection in data pipelines, although these constructs are essential in providing many organizations with high-quality data. Recent literature shows interest in data quality and monitoring of pipelines, but only a few examples of practical or case studies exist. Literature examples specifically from ETL are plenty but the broader picture of the data pipeline is not as much researched. The data pipeline monitoring solution created provides much-needed visibility for the company and literature with a case study with easy to implement and effective for other similar limited visibility contexts. Logging mechanisms and analyzing the logging data allowed effective performance monitoring. For instance, the dashboards can provide companies near real-time status, historical analysis, and information on potential error causes reducing the time consumed by the investigation.

The general framework included in Chapter 5 combines topics and guidelines from the related literature and observations made during this study. The framework can be used as a guide to help when companies or organizations need to implement monitoring of an existing data pipeline. Framework was iterated through the development of the artifact and used basic DSR and data quality assessment processes as a baseline. The processes found in the related literature were adapted to better fit the data pipeline monitoring context and further developed due to observations of practice. This framework extends the available literature by providing an effective framework for implementing data pipeline monitoring to an existing pipeline. This framework also illustrates the notable factors that need to be considered when data pipeline monitoring is implemented. The design of the solution followed this framework and the DSR methodology.

6.1 Results of data quality assessment and overall data quality picture in the company

The data quality of the sources used for the solution was assessed. The quality assessment framework introduced in the literature review was deemed working. A broader data quality picture of the company was researched with semi-structured interviews which provided insight beyond basic measuring. The data quality problems mentioned during interviews could be connected to the most researched and important data quality dimensions: accuracy, completeness, timeliness, and consistency. Companies' experts have experienced frustrations regarding these exact data quality dimensions. This further validates the specific four data quality dimensions selected by Ehrlinger & Wöß (2022). Although the possible problems demonstrated by the interviews were not necessarily the same as the ones in the literature, similar dimensions indicate that practice supports general data quality literature.

Data quality was assessed for the two sources used for the final solution. File event dataset and tester outcome dataset. The assessment did not provide any unique or distinct data quality problems that would differ from the interviews. Assessed data and maintained good data quality throughout the continuous assessment. A practical assessment was conducted using Python pandas to assess the data quality better since the datasets were massive around 300 million rows in total. A sample of a million rows was chosen. Examples of Python scripts used can be found in Appendix 3. Note that these scripts are illustrations of the actual scripts used and modified to fit the example data in Appendix 2. The results of the data quality assessment can be found below in Table 12.

Table 11: Data Quality Assessment Results

Data Quality Dimensions	**Metric**	**File Event Dataset**	**End User Dataset**
Accuracy	Data Validation Errors	0.00359 %	0.0001%
Completeness	Missing value rate	0.0009%	0%
Consistency	Duplicate Records Rate	0%	0%
Timeliness	Latency	~15 seconds to 1 day	~ 1 minute to 1 day

Accuracy was assessed with a data validation error. Data validation error was calculated by dividing the number of invalid rows by the total number of records. Data validation rules used were data type and range checks. Ranges for numerical values were set based on expected values and for each column, they were assessed individually. Duplicate rows did not exist, and each row contained at least one unique identifier. Completeness, in this case, refers to a specific value missing rather than a whole row. There were 0.009% missing values in the file event dataset and 0% in the end-user dataset. Consistency was assessed by calculating duplicate records. In the previous chapter are examples of more detailed consistency checks between sources. Latency was calculated as the average time delay between data generation and its availability to use.

Data quality assessment overall indicated that data is high quality in specific sources, but the bigger problem that the company has is the less clear holistic picture. Where sources used for this solution were properly maintained and data correctly processed this is most likely not the case for other sources. Both sources were on an acceptable level based on accuracy, completeness, consistency, and timeliness and were deemed fit to use. Organizational factors are most likely the cause of most data quality issues. Reasoning for the previous is lack of data governance, lack of general data quality knowledge, and observed concentration of data quality efforts to company's top experts. These experts have many other responsibilities and limited hours to dedicate to improving the data quality. Since developing data competencies is not necessarily included in core business processes hence a limited allocation of additional resources to improving data capability. Three possible scenarios imaginable are reduced competitiveness due to lacking data capabilities, training personnel to create data quality awareness, or allocating more resources to increase the capabilities.

6.2 Results of data pipeline monitoring solution and general framework

The artifact created should consider finding the right context for the application of the solution. The problem of the context was clear from the beginning as this problem DSR process is context initiated. Therefore, the actual context is already detailed in chapter 4 and the demonstration is integrated as a part of the following solution evaluation. Although visibility to the data ingestion phase of the data pipeline and completeness of tester records was improved significantly there are still some file events logged that cannot be analyzed

with certainty due to errors in file generation. This can be seen as one of the non-intended results and gives the company new avenues for further research of what is the cause of errors in file generation.
Designed monitoring solution employed techniques like logging and alerting mechanisms and analyzed the results. Logging and alerting mechanisms proved to be crucial for the success of the final solution. While Aishwarya et al (2020b), Munappy et al. (2020), and Foidl et al. (2024) suggest the use of similar practical techniques some of the literature like Redyuk et al., (2021) and Somasundaram (2023) advocate for more complex approaches. There exists the problem of rigor versus relevancy which is essential in design science research (Gregor & Hevner, 2013). Our solution was initiated by the context and therefore was focused on practicality rather than contributing heavily to the literature. During the thesis, more complex methods like machine learning methods and anomaly detection algorithms were trialed, but they did not produce anything that could not be accomplished by the simple final solution. While we did not include complex methods in the final solution there are strong implications and expectations for those methods to perform monitoring to a satisfactory degree in a different context. Also, the design choice to not include the alerting mechanisms and the rest of the solution to the data pipeline but rather include them in a separate cloud platform was practical and fruitful. The importance of logging mechanisms and the data generated cannot be understated as they played an instrumental part in the final solution. The actual logic behind the ingestion logging mechanism was not unfortunately demonstrated in detail since it was considered confidential.

Although the solution satisfied the objectives it could have been designed differently. The policy of not changing or modifying the data pipeline was set from the beginning while the solution was not designed to be part of the actual pipeline, without the policy we could have used a multitude of external tools or data orchestrators. Munappy et al. (2021) argued between robustness and complexity in data pipelines. Our simple solution satisfies both conditions since no additional steps were added, and the robustness of the pipelines was improved. When this data pipeline is eventually re-developed or designed the considerations of the technology stack should include built-in monitoring. Even though a practical and efficient solution could be designed without changing the data pipeline new technologies could provide the needed monitoring built-in as demonstrated by Redyuk et al., (2021) and (Aiswarya et al., 2020b).

A general framework for implementing monitoring to an existing pipeline was developed based on the rigor and practical knowledge generated during the DSR process. The framework cannot be considered conclusive or to work all possible pipelines, since the purpose of the framework was to attempt to create guidelines or instructions for different topics that need to be considered. The framework consists of relevant theoretical topics and steps that should be taken or at least considered when implementing monitoring. Based on the rigor demonstrated in this thesis this is an attempt at generalization and future research will show whether selected topics are relevant in practice. Characteristics of monitoring implementation to an existing pipeline are the review of potentially complex problem domains and the definition of metrics. The framework can be seen as an extension and synthesis of the work of Hevner et al. (2004), Foidl et al. (2024), Munappy et al. (2020), and Cai & Zhu (2015). The framework contains similarities to data quality assessment, data pipeline, and DSR frameworks that are induced with practical knowledge created.

Implications of the practice and theory that the designed and developed solution provides are the effectiveness of analyzing logging mechanisms data and using simple metrics and analysis. In practice often laziness or resources are the driving mechanisms to drive simple yet novel solutions whereas literature focuses on making complex methods to possibly marginally better analyze or predict the outcome. For literature, the designed artifact provides a case example of the successful implementation of data pipeline monitoring using a combination of relevant methods. The framework provides further research as at least a starting point for monitoring implementation.

7 Conclusions

The state of data quality in global manufacturing companies has reached uneven maturity, considering various aspects of data integrity and consistency across diverse data architectures. This research examined the general data quality within a major global manufacturing company, identifying key factors influencing data quality and developing a customized data pipeline ingestion monitoring solution to increase visibility. Both major research objectives were investigated in detail and a general framework was developed. The following chapter summarizes the main findings and answers the research questions. The thesis followed DSR methodology and guidelines to design practical solutions to well-defined problems. Each of the research questions is answered in a structured format: summary of objectives and key findings, comparison of the previous research and perceived impacts, and finally describing the limitations of future research directions.

The first research question investigated data quality and causes within the company and was structured as:

What are the main data quality issues identified in the company and possible causes for the issues?

The objectives of this research question were to identify common data quality-related challenges faced in organizations and determine the possible causes for these issues. Interview results determined that common data quality challenges were outdated data, discrepancies between sources or systems, missing data, and human-induced errors. Another challenge interviewees faced was the abundance of sources and practices in use while hardly any standards or best practices were present in the company. The data quality of two data sources was assessed and the findings were slightly reduced data quality in accuracy and completeness of the data. The most critical finding in the overall picture was the lack of data governance in practice. Data governance should be the guiding arm alongside data management in all data practices. Other possible causes for the data quality challenges identified were a lack of adequate monitoring, overburdened data professionals, and poor data entry practices. This further outlines the importance of data governance and the need for data quality liability throughout the company. Findings provide a good starting point for improving data quality in the company as both common challenges and possible causes were identified thus fulfilling the research objective.

Regarding the previous data quality research, the findings did not contradict the widely adopted approaches. The assessment framework used provided insights into the practical assessment of datasets. The objective was not to construct any extensions in literature but rather to use scientific rigor in practice. Since the literature outlined the context as the main factor, methods were used to evaluate the data in the company context. The common data quality problems that were discovered aligned with the four major data quality dimensions. This further validates the previous research on the importance of accuracy, completeness, timeliness, and consistency. Thus, data quality literature has strong ties to relevance and practicality. Outcomes for the company and other manufacturing companies are the call to implement data governance and data management, considering data as an asset, and providing examples of previously less investigated data quality problems.

Although scientific rigor was demonstrated in assessing, analyzing, and concluding the research material, research limitations were expected to occur. Limitations of the data quality investigation faced were the small sample size of interviews, the scope of data analyzed, and possibly biased selection of the assessed datasets. The interview sample size was limited by the resources and time allocated, still, the experts interviewed represented different divisions and were chosen by their different points of view. Experts interviewed provided a wide range of perspectives on different elements of data quality. Assessed datasets do not present a wide range of systems and data sources, but from the start, it would not be realistic to assess the quality of every dataset in the scope of the master thesis. Therefore, the datasets were limited to the ones used in the development of the artifact. Future research regarding data quality will likely compare different industries and investigate wider emerging trends like the application of advanced data quality validation and correction. Improving and focusing on data quality will most likely improve decision-making, enhance operational effectiveness, and help move toward actual data-driven decision-making. If employees don’t trust the data, what are they basing their decisions on?

Progressing on from data quality to practical solution solving data quality problem. The second research question investigated and followed the DSR methodology to construct a practical monitoring solution in a specific context. The phrasing of the research question is following:

How can an automated monitoring solution be designed to fit into practice within the existing data pipeline?

The objective of this research question was to design and develop practical and usable monitoring solutions for existing and limited visibility data pipelines. Designing and developing the practical solution required an understanding of a wide array of elements and latent factors. This array of elements is further demonstrated with the next research question. Five different solution options were seriously considered to accurately monitor the completeness of end-user records. Records consisted of supplier-performed testing of electronic components and completeness and awareness of corrupted or missing files was deemed necessary. The final developed solution used a combination of common data pipeline monitoring techniques to achieve satisfactory monitoring results. Methods used for detecting missing files and pipeline breakages were logging mechanisms, customized alerting mechanisms, statistical analysis, and comparing datasets. Developed solution provided the company with much-needed visibility to both the completeness of records and analysis of data pipeline performance.

Since the existing data pipeline could not be modified the solution relied on querying data from different processing points demonstrating a simple approach to a complicated problem. Drivers for a simple solution approach were reduced complexity, lower maintenance costs, proven techniques, scalability, and low latency. Developing more complex approaches might have been most likely less beneficial especially considering the maintenance and continuous development required. Using simple analysis and alerting techniques also provided the possibility of developing the solution with technology that has been widely used in the company. The final solution included an assessment of end-user record completeness, a dashboard with key performance indicators, an analysis of data pipeline error logs, and alerting mechanisms. These features exceeded the objectives defined for the solution.

Recent literature showed some case studies of data pipeline monitoring in practice and more theoretical approaches to data pipeline monitoring. The research gap of data pipeline monitoring usage to an existing pipeline without changing the pipeline technology was missing. The main approaches used for the designed artifact were already heavily present in literature among more complex methods. Of course, simple, and straightforward solutions are not as theoretically interesting, but they are still relevant. The thesis still demonstrated the power of logging mechanisms and alerting validating previous research. While the theoretical implications are limited from this instantiation it provided a novel solution and a case study to the existing collection of data engineering literature.

Focusing mostly on the relevance this design and development process had mostly practical and legal limitations. Since some parts of the case pipeline had limited accessibility due to data generation happening in the supplier's domain, design options were limited. The design could have been also integrated into the pipeline if the permission to modify the pipeline was granted. Modifying the existing pipeline could have enabled the use and testing of pre-made solutions. Still, the artifact created was satisfactory for the case company. Other limitations of legal nature were that most of the used technology could not be named in this thesis due to policies of the case company and non-disclosure agreements. A similar solution still could be replicated by other companies with similar data architecture and logging mechanisms. In the future, the focus of the data pipeline theoretical and practical research will shift towards data pipelines with increasing amounts of data and how to monitor the enormous and unstructured data flowing through data pipelines. Data pipeline and data engineering research will diverge in the future in two directions to efficiently monitor and validate extraordinary amounts of data and to highly specialized and complex applications of machine learning.

DSR methodology being a two-sided coin of theoretical and practical knowledge the third research question attempts generalization of the data pipeline monitoring solution. The third research question was formulated as:

What is the general framework for implementing monitoring solutions to an existing pipeline?

The research question's objective was to create a generalized framework based on the practical and theoretical knowledge gathered during the design and development of the solution. The general framework illustrates the process flow to achieve a successfully implemented monitoring. The proposed framework illustrates the key topics to consider and guides the development process. The attempt made to create a generalization based on rigor and relevance demonstrated during this work finalizes the framework and meets the research objectives. Key components of the framework highlighted in data pipeline monitoring are understanding the data pipeline, defining monitoring objectives and metrics, assessing available solutions or tools, selecting tools based on multiple criteria like practicality or cost, integration and testing, and finally maintenance and continuous improvement. Following these framework steps provides practitioners with a comprehensive approach to monitoring data pipelines. Framework steps are complimented by both the designed solution and descriptions of each step with examples. Overall, the general framework will be an applicable approach to similar context situations in the industry.

The framework has roots in data quality, data pipeline, and data management literature as well as practical knowledge of continuous improvement and Lean Six Sigma to create a comprehensive and usable generalization. Previous data pipeline literature has focused on both the overall picture of data pipelines and demonstrations of specific monitoring techniques. While descriptions of data pipeline monitoring exist in the literature, they are either vague or highly context-dependent. The general framework illustrated in this thesis falls in between demonstrating both rigor and practical relevance with an approach that would be usable and understandable for practitioners. Thus, the framework essentially bridges a knowledge gap. While the design of the practical solution followed the DSR methodology and process similar framework to what is represented here would have made the design process more efficient and practical.

While balancing between practicality and contributions to the literature some limitations had to be made. The framework although applicable in the current context is not comprehensive yet along different contexts with distinctive characteristics. While the framework illustrates both the previous literature and knowledge gained due to the scope and limitations of the master thesis it cannot be stated as the truth. The context of this thesis could have affected the framework structure and therefore inspiration from similar more established fields was taken. Framework constitutes a new approach to applying data pipeline monitoring in the future and can help practitioners with examples and logical process flow. While applicable today, technology changes may affect the usefulness of this framework. More case studies and competing frameworks are needed to either prove or disprove this framework.

REFERENCES:

Abedjan, Z., Golab, L. & Naumann, F. 2015. Profiling relational data: a survey. *The VLDB Journal*. Vol. 24, no. 4, p. 557-581.

Abraham, R., Schneider, J. & vom Brocke, J. 2019. Data governance: A conceptual framework, structured review, and research agenda. *International Journal of Information Management*. Vol. 49, p. 424-438.

Ahmed, M., Mahmood, A. N. & Hu, J. 2016. A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*. Vol. 60, p. 19-31.

Aiswarya, R., Bosch, J. & Olsson, H. H. 2021. On the Trade-Off between Robustness and Complexity in Data Pipelines. *Quality of Information and Communications Technology: 14th International Conference, QUATIC 2021*. Algarve, Portugal. 8–11.9, 2021. Proceedings 14: Springer. p. 401-415.

Aiswarya, R., Bosch, J., Olsson, H. H. & Wang, T. J. 2020a. Modelling Data Pipelines. *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, Portoroz, Slovenia. 26-28.8, 2020. IEEE. p. 13-20.

Aiswarya, R., Bosch, J., Olsson, H. H. & Wang, T. J. 2020b. Towards Automated Detection of Data Pipeline Faults. *2020 27th Asia-Pacific Software Engineering Conference (APSEC)*. Singapore, Singapore. 1-4.12, 2020. IEEE. p. 346-355.

Aken, J. E. v. 2004. Management research based on the paradigm of the design sciences: the quest for field-tested and grounded technological rules. *Journal of Management Studies*. Vol. 41, no. 2, p. 219-246.

Albrecht, R., Overbeek, S. & van de Weerd, I. 2023. Designing a Data Quality Management Framework for CRM Platform Delivery and Consultancy. *SN Computer Science*. Vol. 4, no. 6, 742

Al-Ruithe, M., Benkhelifa, E. & Hameed, K. 2019. A systematic literature review of data governance and cloud data governance. *Personal and Ubiquitous Computing*. Vol. 23, p. 839-859.

Barriball, K. L. and While, A. 1994. Collecting data using a semi-structured interview: a discussion paper. *Journal of Advanced Nursing-Institutional Subscription*. Vol. 19, no. 2, p. 328-335.

Baskerville, R., Baiyere, A., Gregor, S., Hevner, A. & Rossi, M. 2018. Design science research contributions: Finding a balance between artifact and theory. *Journal of the Association for Information Systems*. Vol. 19, no. 5, p. 3.

Batini, C., Cappiello, C., Francalanci, C. & Maurino, A. 2009. Methodologies for data quality assessment and improvement. *ACM Computing Surveys*. Vol. 41, no. 3, p. 1-52.

Biswas, S., Wardat, M. & Rajan, H. 2022. The Art and Practice of Data Science Pipelines: A Comprehensive Study of Data Science Pipelines in Theory, in-the-Small, and in-the-Large. *Proceedings of the 44th International Conference on Software Engineering*. Pittsburgh, Pennsylvania, United States. 21 - 29.5, 2022. New York, NY, United States: Association for Computing Machinery. p. 2091-2103.

Blázquez-García, A., Conde, A., Mori, U. & Lozano, J. A. 2021. A review on outlier/anomaly detection in time series data. *ACM Computing Surveys (CSUR*). Vol. 54, no. 3, p. 1-33.

Cai, L. and Zhu, Y. 2015. The Challenges of Data Quality and Data Quality Assessment in the Big Data Era. *Data Science Journal*. Vol. 14, p. 2.

Carretero, A. G., Gualo, F., Caballero, I. & Piattini, M. 2017. MAMD 2.0: Environment for data quality processes implantation based on ISO 8000-6X and ISO/IEC 33000. *Computer Standards and Interfaces*. Vol. 54, p. 139-151.

Chandola, V., Banerjee, A. & Kumar, V. 2009. Anomaly detection: A survey. *ACM Computing Surveys (CSUR*). Vol. 41, no. 3, p. 1-58.

Cichy, C. and Rass, S. 2019. An Overview of Data Quality Frameworks. *IEEE Access*. Vol. 7, p. 24634-24648.

DAMA International 2017. DAMA-DMBOK: Data Management Body of Knowledge (2nd Edition), Technics Publications, LLC, Denville, NJ, USA, p. 71

Dehury, C. K., Jakovits, P., Srirama, S. N., Giotis, G. & Garg, G. 2022. TOSCAdata: Modeling data pipeline applications in TOSCA. *The Journal of Systems and Software*. Vol. 186, p. 111164.

Dreves, M., Huang, G., Peng, Z., Polyzotis, N., Rosen, E. & GC, P. S. 2021. Validating Data and Models in Continuous ML Pipelines. *IEEE Data Eng.Bull*. Vol. 44, no. 1, p. 42-50.

Ehrlinger, L., Werth, B. & Wöß, W. 2023. Automating Data Quality Monitoring with Reference Data Profiles. *International Conference on Data Management Technologies and Applications, DATA 2022*, Lisbon, Portugal. Cham:Springer. p. 24-44.

Ehrlinger, L. and Wöß, W. 2022. A Survey of Data Quality Measurement and Monitoring Tools. *Frontiers in Big Data*. Vol. 5, p. 850611.

Fan, W. 2015. Data quality: From theory to practice. *SIGMOD Record*. Vol. 44, no. 3, p. 7-18.

Fan, W. and Geerts, F. 2022. Foundations of Data Quality Management. Springer Nature Switzerland AG. p. 5-11.

Farooqui, A., Bengtsson, K., Falkman, P. & Fabian, M. 2020. Towards data-driven approaches in manufacturing: an architecture to collect sequences of operations. *International Journal of Production Research*. Vol. 58, no. 16, p. 4947-4963.

Firmani, D., Leotta, F., Mandreoli, F. & Mecella, M. 2021. Big data management in industry 4.0. *Frontiers in Big Data*. Vol. 4, p. 788491.

Foidl, H., Golendukhina, V., Ramler, R. & Felderer, M. 2024. Data pipeline quality: Influencing factors, root causes of data-related issues, and processing problem areas for developers. *The Journal of Systems and Software*. Vol. 207, p. 111855.

Glowalla, P. and Sunyaev, A. 2014. Process-driven data quality management: A critical review on the application of process modeling languages. *ACM Journal of Data and Information Quality*. Vol. 5, no. 1-2, p. 1-30.

Gregor, S. and Hevner, A. R. 2013. Positioning and Presenting Design Science Research for Maximum Impact. *MIS Quarterly*. Vol. 37, no. 2, p. 337-355.

Gupta, M. and George, J. F. 2016. Toward the development of a big data analytics capability. *Information & Management*. Vol. 53, no. 8, p. 1049-1064.

Hevner, A. R., March, S. T., Park, J. & Ram, S. 2004. Design Science in Information Systems Research. *MIS Quarterly*. Vol. 28, no. 1, p. 75-105.

Hinrichs, H. 2003. Data Quality Management in Data Warehouse Systems. PhD, Universität Oldenburg

Ismail, A., Truong, H. & Kastner, W. 2019. Manufacturing process data analysis pipelines: a requirements analysis and survey. *Journal of Big Data*. Vol. 6, no. 1, p. 1-26.

Järvinen, P. 2007. Action research is similar to design science. *Quality & Quantity*. Vol. 41, p. 37-54.

Ji, C., Shao, Q., Sun, J., Liu, S., Pan, L., Wu, L. & Yang, C. 2016. Device data ingestion for industrial big data platforms with a case study. *Sensors (Switzerland).* Vol. 16, no. 3, p. 34-42.

Kallio, H., Pietilä, A., Johnson, M. & Kangasniemi, M. 2016. Systematic methodological review: developing a framework for a qualitative semi-structured interview guide. *Journal of Advanced Nursing*. Vol. 72, no. 12, p. 2954-2965.

Klievink, B., van Stijn, E., Hesketh, D., Tan, Y., Aldewereld, H., Overbeek, S. & Heijmann, F. 2012. Enhancing Visibility in International Supply Chains: The Data Pipeline Concept. *International Journal of Electronic Government Research*. Vol. 8, no. 4, p. 14-33.

Koltay, T. 2016. Data governance, data literacy and the management of data quality. *IFLA Journal*. Vol. 42, no. 4, p. 303-312.

Kuechler, W. and Vaishnavi, V. 2012. A Framework for Theory Development in Design Science Research: Multiple Perspectives. *Journal of the Association for Information Systems*. Vol. 13, no. 6, p. 395-423.

Kwon, O., Lee, N. & Shin, B. 2014. Data quality management, data usage experience and acquisition intention of big data analytics. *International Journal of Information Management*. Vol. 34, no. 3, p. 387-394.

Madnick, S. E., Wang, R. Y., Lee, Y. W. & Zhu, H. 2009. Overview and framework for Data and information quality research. *Journal of Data and Information Quality*. Vol. 1, no. 1.

Mayernik, M. S. 2020. Metadata. *Knowledge Organization: KO*. Vol. 47, no. 8,

Miksa, T., Oblasser, S. & Rauber, A. 2022. Automating Research Data Management Using Machine-Actionable Data Management Plans. *ACM Transactions on Management Information Systems*. Vol. 13, no. 2, p. 1-22.

Munappy, R., Bosch, J. & Olsson, H. 2020. Data Pipeline Management in Practice: Challenges and Opportunities. *Product-Focused Software Process Improvement: 21st International Conference, PROFES 2020*. Turin, Italy. 25–27.11, 2020. Proceedings 21: Springer. p. 168-184.

Narayanan, S., Maheswari, S. & Zephan, P. 2024. Real-Time Monitoring of Data Pipelines: Exploring and Experimentally Proving that the Continuous Monitoring in Data Pipelines Reduces Cost and Elevates Quality. *EAI Endorsed Transactions on Scalable Information Systems.* Vol. 11, no. 4.

Nasution, M. K. 2021. Industry 4.0: Data Science Perspective. *Iop conference series: Materials science and engineering*. Vol. 1122. no. 1.

O'Donovan, P., Leahy, K., Bruton, K. & O'Sullivan, D. T. J. 2015. An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities. *Journal of Big Data*. Vol. 2, no. 1, p. 1-26.

Oleghe, O. and Salonitis, K. 2020. A framework for designing data pipelines for manufacturing systems. *Procedia CIRP*. Vol. 93, p. 724-729.

Omri, N., Al Masry, Z., Mairot, N., Giampiccolo, S. & Zerhouni, N. 2021. Towards an adapted PHM approach: Data quality requirements methodology for fault detection applications. *Computers in Industry*. Vol. 127, p. 103414.

Ottonicar, S. L. C., Valentim, M. L. P. & Mosconi, E. 2019. A competitive intelligence model based on information literacy: organizational competitiveness in the context of the 4th Industrial Revolution. *Journal of Intelligence Studies in Business*. Vol. 8, no. 3.

Peffers, K., Tuunanen, T., Rothenberger, M. A. & Chatterjee, S. 2007. A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*. Vol. 24, no. 3, p. 45-77.

Perlin, N. 2006. Introduction to Metadata. *2006 IEEE International Professional Communication Conference*. Saragota Springs, NY, USA. 23-25.10, 2006. IEEE International Professional Communication: IEEE. p.153-155.

Pipino, L. L., Lee, Y. W. & Wang, R. Y. 2002. Data Quality Assessment. *Communications of the ACM*. Vol. 45, no. 4, p. 211-218.

Plale, B. and Kouper, I., 2017. The centrality of data: data lifecycle and data pipelines. In: Mashrur Chowdhury, Amy Apon, and Kakan Dey, eds. Data analytics for intelligent transportation systems. Amsterdam, Netherlands: Elsevier, p. 91–111.

Rahul, K. and Banyal, R. K. 2020. Data life cycle management in big data analytics. Procedia Computer Science. Vol. 173, p. 364-371.

Raptis, T. P., Passarella, A. & Conti, M. 2019. Data Management in Industry 4.0: State of the Art and Open Challenges. *IEEE Access: Practical Innovations, Open Solutions.* Vol. 7, p. 97052-97093.

Redyuk, S., Kaoudi, Z., Markl, V. & Schelter, S. 2021. Automating Data Quality Validation for Dynamic Data Ingestion. *EDBT*. p. 61-72.

Sajid, A., Nayyar, A. & Mohsin, A. 2010. Modern Trends Towards Requirement Elicitation. *Proceedings of the 2010 National Software Engineering Conference (NSEC '10)*. p. 1–10.

Schelter, S., Lange, D., Schmidt, P., Celikel, M., Biessmann, F. & Grafberger, A. 2018. Automating large-scale data quality verification. *Proceedings of the VLDB Endowment*. Vol. 11, no. 12, p. 1781-1794.

Schröer, T., Janßen, J., Schuh, G. & Stich, V. 2023. Systematisation Approach: Handling Insufficient Data Quality. *Proceedings of the Conference on Production Systems and Logistics: CPSL 2023*. p. 469-478.

Shah, A., Alasow, M. A., Sajjad, F. & Baig, J. J. A. 2017. An Evaluation of Software Requirements Tools. *2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS)*. Cairo, Egypt. 5-7-12, 2017. IEEE. p. 278-283.

Somasundaram, P. 2023. Improving Real-Time Job Monitoring for Data Pipelines in The Context of Cloud Analytics. *International Journal of Computer Engineering and Technology (IJCET)*. Vol. 14, no. 3, p. 34-42.

Sonnenberg, C. & Vom Brocke, J. 2012. Evaluation Patterns for Design Science Research Artefacts. *Practical Aspects of Design Science: European Design Science Symposium, EDSS 2011*. Leixlip, Ireland. 14-14.10, 2011. Revised Selected Papers 2: Springer. p. 71-83.

Tambare, P., Meshram, C., Lee, C., Ramteke, R. J. & Imoize, A. L. 2021. Performance measurement system and quality management in data-driven Industry 4.0: A review. *Sensors*. Vol. 22, no. 1, p. 224.

Torres, R. and Sidorova, A. 2019. Reconceptualizing information quality as effective use in the context of business intelligence and analytics. *International Journal of Information Management*. Vol. 49, p. 316-329.

Venable, J., Pries-Heje, J. & Baskerville, R. 2016. FEDS: a framework for evaluation in design science research. European Journal of Information Systems. Vol. 25, p. 77-89.

Vom Brocke, J., Hevner, A. & Maedche, A., 2020. Introduction to design science research. In: Vom Brocke J, Hevner A. & Maedche A. (eds) Design Science Research.Cases. Cham: Springer, p.1-13.

Wang, J., Li, Y., Song, W. & Li, A. 2018. Research on the Theory and Method of Grid Data Asset Management. *Procedia Computer Science.* Vol. 139, p. 440-447.

Wang, R. Y. and Strong, D. M. 1996. Beyond Accuracy: What Data Quality Means to Data Consumers. *Journal of Management Information Systems*. Vol. 12, no. 4, p. 5-33.

Weber, K., Otto, B. & Österle, H. 2009. One Size Does Not Fit All---A Contingency Approach to Data Governance. *ACM Journal of Data and Information Quality*. Vol. 1, no. 1, p. 1-27.

Weigand, H., Johannesson, P. & Andersson, B. 2021. An artifact ontology for design science research. *Data & Knowledge Engineering*. Vol. 133, p. 101878.

Zhang, L., Howard, S., Montpool, T., Moore, J., Mahajan, K. & Miranskyy, A. 2023. Automated data validation: An industrial experience report. *The Journal of Systems and Software*. Vol. 197, p. 111573.

Zorrilla, M. and Yebenes, J. 2022. A reference framework for the implementation of data governance systems for industry 4.0. *Computer Standards and Interfaces*. Vol. 81, p. 103595.

APPENDIX 1. Interview Guide

Topic	**Question**	**Goal of the question**
Metadata	Could you briefly elaborate your role and involvement?	To categorize the answers
Data quality	How do you view data quality in organization?	Overall Maturity and Context for the question
	Could you describe common data quality related problems.	To understand better the challenges in data quality
	What are the most important data quality improvements in the future?	Future and direction. Priority
Data Quality Management	From your perspective, what role does data governance play in ensuring data quality?	Data Governance
	Can you share any success stories or best practices related to improving data quality management within the organization?	Overall Maturity and Context for the question
Data pipeline Monitoring framework	What processes or tools do you find useful for identifying anomalies in data from pipelines?	Already in use tools
	What would your feedback on the current Data Pipeline Monitoring framework?	Re-evaluating and reflecting on the model
	Do you have any perspectives to add to the monitoring framework?	Additional dimensions or topics to consider when implementing the tool.
Reflection	Do you have anything else to add to the data quality topic?	Additional related Information.

APPENDIX 2. File Event Table example (Generated)

TestResultKey	Status	Supplier	Tester Machine	File Path	Runtime (ms)	Error Code
001	1	Supplier 1	Tester Machine 1	C:\TesterData\0123456789	1200	-
002	0	Supplier 1	Tester Machine 2	C:\TesterData\9876543210	2500	Integration Error
003	1	Supplier 1	Tester Machine 3	C:\TesterData\1234567890	1300	-
004	0	Supplier 1	Tester Machine 4	C:\TesterData\0987654321	300	Parsing Error
005	1	Supplier 1	Tester Machine 5	C:\TesterData\2345678901	1400	-
006	0	Supplier 1	Tester Machine 6	C:\TesterData\3456789012	2700	Multiple Errors
007	1	Supplier 1	Tester Machine 7	C:\TesterData\4567890123	1100	-
008	0	Supplier 1	Tester Machine 8	C:\TesterData\5678901234	150	Parsing Error
009	1	Supplier 1	Tester Machine 1	C:\TesterData\6789012345	1350	-
010	0	Supplier 1	Tester Machine 2	C:\TesterData\7890123456	2900	Integration Error

APPENDIX 3. Data Quality Assessment Pandas Scripts

Metric	Python Script Example

Data Validation Errors	# Calculate Data Validation Errors
	validation_errors =df[df['Status'].isin([0, 1])].shape[0]
	data_validation_errors_rate = (validation_errors / df.shape[0]) * 100
	data_validation_errors_rate

Outlier Detection Rate	# Calculate Outlier Detection Rate using IQR
	Q1 = df['Runtime (ms)'].quantile(0.25)
	Q3 = df['Runtime (ms)'].quantile(0.75)
	IQR = Q3 - Q1
	outliers = df[(df['Runtime (ms)'] < (Q1 - 1.5 * IQR)) | (df['Runtime (ms)'] > (Q3 + 1.5 * IQR))]
	outlier_detection_rate = (outliers.shape[0] / df.shape[0]) * 100
	outlier_detection_rate

Missing Values Rate	# Calculate Missing Values Rate
	missing_values = df.isnull().sum().sum()
	total_values = df.shape[0] * df.shape[1]
	missing_values_rate = (missing_values / total_values) * 100
	missing_values_rate

Duplicate Records Rate	# Calculate Duplicate Records Rate
	duplicate_records = df.duplicated(subset='TestResultKey').sum()
	duplicate_records_rate = (duplicate_records / df.shape[0]) * 100
	duplicate_records_rate