The Journal of Systems and Software 207 (2024) 111855

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

Data pipeline quality: Influencing factors, root causes of data-related issues, and processing problem areas for developers☆

Harald Foidl [a,*], Valentina Golendukhina [a], Rudolf Ramler [b], Michael Felderer [c,a,d]

[a] University of Innsbruck, Austria
[b] Software Competence Center Hagenberg GmbH, Austria
[c] Institute for Software Technology, German Aerospace Center (DLR), Germany
[d] University of Cologne, Germany

ARTICLE INFO

Keywords:
Data pipeline
Data quality
Influencing factors
GitHub
Stack Overflow
Taxonomy

ABSTRACT

Data pipelines are an integral part of various modern data-driven systems. However, despite their importance, they are often unreliable and deliver poor-quality data. A critical step toward improving this situation is a solid understanding of the aspects contributing to the quality of data pipelines. Therefore, this article first introduces a taxonomy of 41 factors that influence the ability of data pipelines to provide quality data. The taxonomy is based on a multivocal literature review and validated by eight interviews with experts from the data engineering domain. Data, infrastructure, life cycle management, development & deployment, and processing were found to be the main influencing themes. Second, we investigate the root causes of data-related issues, their location in data pipelines, and the main topics of data pipeline processing issues for developers by mining GitHub projects and Stack Overflow posts. We found data-related issues to be primarily caused by incorrect data types (33%), mainly occurring in the data cleaning stage of pipelines (35%). Data integration and ingestion tasks were found to be the most asked topics of developers, accounting for nearly half (47%) of all questions. Compatibility issues were found to be a separate problem area in addition to issues corresponding to the usual data pipeline processing areas (i.e., data loading, ingestion, integration, cleaning, and transformation). These findings suggest that future research efforts should focus on analyzing compatibility and data type issues in more depth and assisting developers in data integration and ingestion tasks. The proposed taxonomy is valuable to practitioners in the context of quality assurance activities and fosters future research into data pipeline quality.

1. Introduction

Data pipelines play a crucial role in today's data-centric age and have become fundamental components in enterprise IT infrastructures. By collecting, processing, and transferring data, they make it possible to use the ever-growing amounts of information to gain valuable insights and make improved decisions. In addition, they have also become integral parts of data-driven systems, e.g., recommender systems, speech, and image recognition systems. Within these systems, they are primarily responsible for putting the data into a suitable form so that the built-in machine learning (ML) algorithms can automatically make intelligent decisions.

Since the quality of the data plays an important role in making reliable and accurate decisions, research on data cleaning and validation has gained significant interest in the last decade (Breck et al., 2019; Biessmann et al., 2021). Most of the work dealt with cleaning and validating data at the early stages of a pipeline, assuming that low data quality was already caused before the pipeline (Foidl and Felderer, 2019). The quality of data pipelines themselves, that is, processing data correctly and without errors, has not been treated in much detail. However, the importance of reliable data pipelines was recently underpinned by a global survey of 1,200 organizations. This survey found that 74% of companies that invest in high-quality and reliable data pipelines have increased their profits by an average of 17% (IDC InfoBrief, 2020).

Nevertheless, evidence suggests that data pipelines tend to be error-prone, hard to debug, and require a lot of maintenance and management (Bomanson, 2019; Romero et al., 2020; Munappy et al., 2020b). A recent survey even identified debugging and maintaining data pipelines as the most pressing issues for data engineers (Data.world and DataKitchen, 2021). Moreover, recent studies (Islam et al., 2019b; Wang

☆ Editor: Aldeida Aleti.

* Corresponding author.
E-mail address: harald.foidl@uibk.ac.at (H. Foidl).

https://doi.org/10.1016/j.jss.2023.111855
Received 7 April 2023; Received in revised form 4 August 2023; Accepted 13 September 2023
Available online 16 September 2023

et al., 2022) show that developers face huge difficulties implementing data processing logic. These difficulties were also observed by Yang et al. (2021). In their study, they found data handling code to be often repetitive, dense, and error-prone. This bad state of data pipelines contributes to the fact that today's data-driven systems suffer heavily from data-induced bugs and data-related technical debt (Bogner et al., 2021; Foidl et al.).

Research has recently started to address these issues in a variety of ways. First, there are research efforts aiming to improve the debugging of data pipelines (Zwick, 2019; Rezig et al., 2020; Lourenço et al., 2020). Second, there are several attempts to automate and guide the creation of data processing components (Bilalli et al., 2018; Giovanelli et al., 2022; Konstantinou and Paton, 2020). Third, research appears to be actively shifting its focus toward an end-to-end analysis of data pipelines especially considering the entire inner data handling process (Schäfer et al., 2020; Munappy et al., 2020a).

However, this recent research stream is still in its early stages. A deeper understanding of the underlying aspects contributing to successful data pipelines is required to ensure dependable pipelines consistently deliver high-quality data. Such an enhanced insight is instrumental in enabling further research endeavors advancing the quality of data pipelines.

This paper aims to address this need as follows. *First*, we aim to identify influencing factors (IFs) that may affect a data pipeline's ability to provide high-quality data. We define an IF or factor of influence as any human, technical, or organizational aspect that may affect the ability of a data pipeline to deliver quality data. Those aspects uncover the core drivers of data pipeline quality and serve as fundamental building blocks in the improvement of data pipeline success. *Second*, we adopt a more technological perspective and seek to understand the nature of data pipeline quality better. In particular, we first look at the root causes and stages of data-related issues in data pipelines by studying GitHub projects. Moreover, we further examine whether there are problem areas for developers that do not correspond to the typical processing stages of pipelines by analyzing Stack Overflow questions. These insights are essential to support debugging activities and to define possible strategies to mitigate data-related issues and can further help uncover specific training needs, focus quality assurance efforts, or discover future research opportunities. The major contributions of the paper are:

- A *taxonomy* of 41 data pipeline IFs grouped into 14 IF categories covering five main themes: data, development & deployment, infrastructure, life cycle management, and processing. The taxonomy was validated by eight structured expert interviews.
- An *empirical study* about (1) the root causes of data-related issues and their location in data pipelines by examining a sample of 600 issues from 11 GitHub projects, and (2) the main topics developers ask about data pipeline processing by analyzing a sample of 400 Stack Overflow posts.

The remaining paper is structured as follows. First, Section 2 provides relevant background information on data pipelines and discusses related work. Section 3 describes the applied research procedure in this paper. Section 4 presents the developed taxonomy of IFs and its evaluation. Section 5 elaborates on the root causes of data-related issues and the topics developers face in processing data in the context of data pipelines. Afterward, Section 6 discusses the findings and limitations of the study. Finally, the paper is concluded in Section 7.

2. Background and related work

In this section, we first cover background information on data pipelines (Section 2.1) and subsequently provide an overview of earlier work related to this paper in Section 2.2.

2.1. Data pipeline

This section first presents the concept and architecture of a data pipeline. Afterward, common pipeline types and the underlying software stack of data pipelines are outlined.

2.1.1. Data pipeline concept

The concept of a 'data pipeline' is described differently in the literature, depending on the perspective taken (Agostinelli et al., 2021). Following, we describe a data pipeline first from a theoretical and then from a practical perspective.

Theoretically, a data pipeline refers to a *directed acyclic graph* (DAG) composed of a sequence of nodes (Drocco et al., 2017). These nodes process (e.g., merge, filter) data while the output of one node will be the input of the next node. At least one source node produces the data at the beginning of the DAG, and at least one sink node finally receives the processed data. Considering pipelines from this perspective is often done in mathematical settings to formally describe and analyze data flows (e.g., node dependencies).

From a practical point of view, data pipelines typically constitute a piece of software that automates the manipulation of data and moves them from diverse source systems to defined destinations (Munappy et al., 2020b). Thus, data pipelines represent digitized data processing workflows based on a set of programmed scripts or simple software tools. Given the increasing importance and complexity of data processing, data pipelines are nowadays even treated as *complete software systems with their own ecosystem* comprising several technologies and software tools (Koivisto, 2019). The primary purpose of this set of complex and interrelated software bundles is to enable efficient data processing, transfer, and storage, control all data operations, and orchestrate the entire data flow from source to destination. We will adopt this practical point of view in the remaining paper.

2.1.2. Data pipeline architecture

We present a generic architecture of a data pipeline shown in Fig. 1. As not otherwise stated, the remaining description in this section is based on Munappy et al. (2020b), Hapke and Nelson (2020), Munappy et al. (2020c), García et al. (2016), Chapman et al. (2020), Hlupić and Puniš (2021), Malley et al. (2016).

Data pipeline components. While the internal structure of a data pipeline can vary significantly, its main components are typically the same: *data sources*, *data processing*, *data storage*, and *data sinks*. Following, we describe these components while focusing on the processing component, as it contains the core tasks of a data pipeline.

Data sources. The starting point of every data pipeline is its data sources. They can be of different types, e.g., databases, sensors, or text files. The data produced by data sources are typically either in a *structured* (e.g., relational database), *semi-structured* (e.g., e-mails, web pages), or *unstructured* (e.g., videos, images) form.

Data processing. The second and central component of a data pipeline is its processing component. Within this component, the core tasks for manipulating and handling data (i.e., data ingestion, data preprocessing, and data loading) are provided. We will refer to these core tasks as stages in the remaining paper.

> *Data ingestion.* Typically, the extraction of the data from the data sources and their import in the pipeline starts the data flow. Depending on the size and format of the raw data, different ingestion methods are applied. The data can be ingested into the pipeline with different load frequencies, i.e., the data can be ingested continuously, intermittently, or in batches.

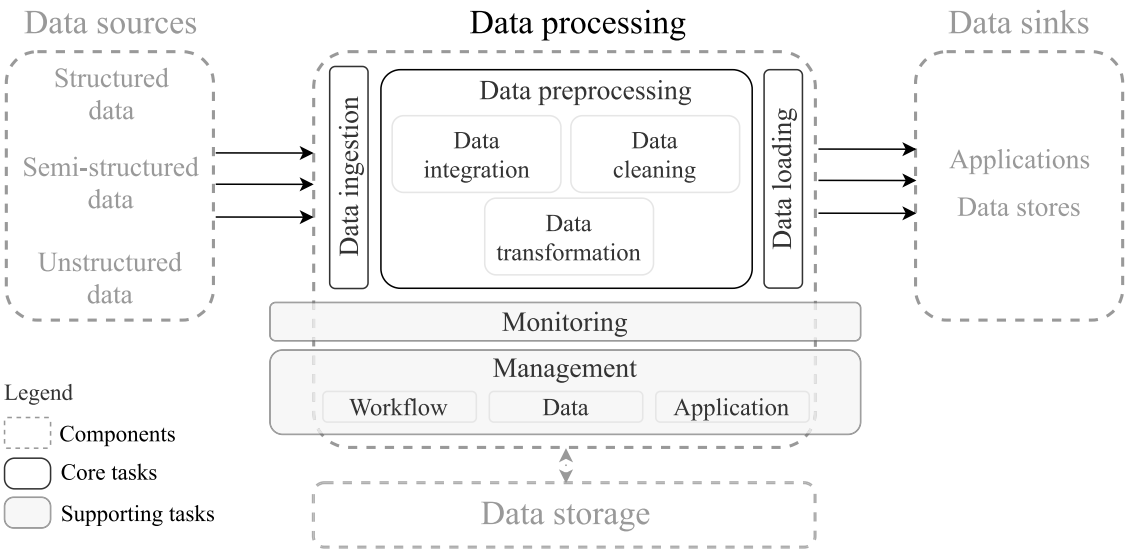

Fig. 1. High-level data pipeline architecture.

Data preprocessing. After the raw data are available in the pipeline, data preprocessing aims to bring the data into an appropriate form for further usage. Typical preprocessing techniques include data integration, cleaning, and transformation. Note that not all preprocessing techniques are applicable in every use case and can be orchestrated differently. *Data integration* aims to merge and integrate the ingested data through, for example, schema mapping or redundancy removal. *Data cleaning* removes data errors and inconsistencies by applying data preparation techniques grouped into missing value treatment (e.g., imputing or discarding) and noise treatment (e.g., smoothing or polishing). *Data transformation* seeks to bring the data in a form suitable for further processing or usage (e.g., statistical analysis or ML models). Typical data preparation techniques for this are binarization, normalization, discretization, dimensionality reduction, numerosity reduction, oversampling, or instance generation.

Data loading. This task loads the ingested or preprocessed data into internal storage systems or external destinations.

Data storage. The data storage component represents the internal storage of the pipeline. It stores raw, ingested, and processed data, depending on the configuration of the pipeline. A common distinction is made between temporary or long-term storage of the data.

Data sinks. The fourth component of a data pipeline describes the destinations where the data are finally provided. These can be other pipelines, applications of any type, or external data storage systems (e.g., data warehouses, databases).

Data pipeline supporting tasks. The data processing tasks of a pipeline are usually supported by several monitoring and management tasks.

Monitoring. This set of tasks refers to overseeing the data quality, all data operations, and controlling the performance of the pipeline. By using logging mechanisms and creating alerts, monitoring prevents data quality issues and eases debugging and recovering from failures.

Management. The main management tasks in the context of data pipelines are related to the data, the workflow, and the overall application. Workflow management describes all activities regarding the orchestration and dependencies between the different data processing tasks (i.e., the entire data flow). Data management includes tasks such as creating and maintaining metadata, data catalogs, and data versioning. Application management encompasses the configuration and maintenance (e.g., upgrades, version control) of all software tools a pipeline is built upon.

2.1.3. Data pipeline types

Pipelines can be classified based on several characteristics. The three most common types of classification are described in the following.

Data ingestion strategy. Data pipelines can be classified based on their data ingestion frequency. Data can be ingested either in batches or continuously. A data pipeline operating in *batch mode* ingests data only in fixed intervals (e.g., daily, weekly) or when a trigger (e.g., manual execution, size of available data) occurs. In contrast, a pipeline continuously ingesting data is operating in *streaming mode*. In this mode, data are consumed in real-time as they become available. There are also scenarios where both ingestion modes are used in parallel (e.g., lambda architecture).

Data processing method. Another way to classify pipelines is based on the application and order of data processing tasks. Commonly used concepts in this regard are *Extract Transform Load* (ETL) and *Extract Load Transform* (ELT). Originally, ETL was used to describe the data transfer from diverse data sources into data warehouses. Data pipelines applying the ETL concept (i.e., *ETL pipelines*) ingest the data (i.e., Extract), preprocess them (i.e., Transform), and then load (i.e., Load) them to the data sinks. During this usually batch-oriented process, the data are typically stored temporarily in the data storage component of the pipeline. On the other hand, *ELT pipelines* ingest (i.e., Extract) the data and then directly load (i.e., Load) them to the data sinks (e.g., data lakes) without preprocessing them. The preprocessing of the data (i.e., Transform) occurs in the data sinks or by applications consuming these data. As a newer concept compared to ETL, ELT is often applied in cloud-based settings, providing fast data without the need for intermediate storage in the data pipeline.

Use case. Data pipelines can be used for a variety of different purposes. They are typically used for data movement and preparation for, or as part of, other applications (e.g., visualization, analysis tools, ML and deep learning applications, or data mining). A common scenario is the usage of data pipelines to gather data from a variety of sources and to move them to a central place for further usage. In this realm, data pipelines are often referred to as *data collection pipelines* and are applied in nearly every business domain (e.g., manufacturing, medicine, finance). In the context of data science (Biswas et al., 2022), AI or ML applications, data pipelines are usually used for preparing the data so that they are in a suitable form when fed to algorithmic models. Data pipelines used as part of AI-based systems, ML pipelines, or in data science projects are thus usually referred to as *data preprocessing pipelines*.

2.1.4. Data pipeline software stack

Various technologies and software applications are used for running data pipelines in production. Data processing components can be implemented with different programming languages (e.g., Python, Java), tools or frameworks (e.g., ETL tools such as Apache Mahout or Pig, message brokers such as RabbitMQ or Apache Kafka, or stream processors such as Apache Spark or Flink). Distributed filesystems (e.g., Hadoop Distributed File System) or databases (relational database management systems such as PostgreSQL, or NoSQL such as Apache Cassandra) are usually used to store the data. To coordinate all data processing tasks in a pipeline, workflow orchestration tools are typically used (e.g., Apache Airflow, Apache Luigi). A further group of several tools is used for monitoring the infrastructure, the data lineage, and data quality (e.g., OpenLineage, MobyDQ). From the plethora of tools, one can choose between open-source or proprietary solutions. Moreover, running the pipeline in the cloud is common to address use cases with high demands on scalability.

2.2. Related work

To the best of our knowledge, no previous study has specifically examined factors that may affect the quality of data in the realm of data pipelines. We thus classify related work into three categories: (1) publications on factors and causes influencing data quality in the field of information and communication technology, (2) contributions on data processing topic issues in related areas, and (3) literature in the context of data pipelines that examine aspects intimately related to the quality of data provided by pipelines.

2.2.1. Related work on factors and causes that influence data quality

There is a considerable amount of literature that studies factors influencing data quality. These studies can roughly be classified by the perspective taken and the domain of data being examined. The perspective describes the type of factors (e.g., managerial, technical factors) and the level of abstraction (e.g., very detailed or general factors) considered. The domain of the data describes whether the data represent a specific area (e.g., health, accounting data) or application (e.g., Internet of Things, data warehouse). While many publications treat IFs from a neutral point of view, some treat them from either a positive (e.g., facilitators, drivers of data quality) or a negative (e.g., barriers, impediments of data quality) perspective.

Several studies (Xu et al., 2002; Tee et al., 2007; Xiao et al., 2009; Xu, 2013) investigated factors that influence the quality of data in organizations' information systems. Collectively, these studies highlight the importance of management support and communication as crucial factors that influence organizations' general data quality. Other studies took a more narrow approach by focusing, for example, on IFs on master data (Haug et al., 2013; Ibrahim et al., 2021) or accounting data (Nord et al., 2005; Zoto and Tole, 2014; Hongjiang, 2015; Knauer et al., 2020). For accounting and master data, literature agrees (Ibrahim et al., 2021; Hongjiang, 2015) that the characteristics of information systems (e.g., ease of use, system stability, and quality) are among the three most influential factors.

In addition to business-related data, there are studies on factors influencing health data quality. Recently, Carvalho et al. (2021) identified 105 potential root causes of data quality problems in hospital administrative databases. The authors associated more than a quarter of these causes with underlying personnel factors (e.g., people's knowledge, preferences, education, and culture), thus being the most critical factor for the quality of health data. A different perspective was taken by Cheburet & Odhiambo-Otieno (Cheburet and Odhiambo-Otieno, 2016). In their study, the authors tried to identify process-related factors influencing the data quality of a health management information system in Kenya. Further, Ancker et al. (2011) investigated issues of project management data in the context of electronic health records to uncover where those issues arose from (e.g., software flexibility that allowed a single task to be documented in multiple ways).

With the widespread adoption of the Internet of Things (IoT), there also has been emerging interest in investigating factors that affect IoT data. In 2016, Karkouch et al. (2016) proposed several factors that may affect the data quality within the IoT (e.g., environment, resource constraints). A recent literature review of Cho et al. (2021) identified device- and technical-related factors, user-related, and data governance-related factors that affect the data quality of person-generated wearable device data.

The most relevant research regarding our work has investigated factors that influence the quality of data in data warehouses. In 2010, Singh and Singh (2010) presented a list of 117 possible causes of data quality issues for each stage of data warehouses (i.e., data sources, data integration & profiling, ETL, schema modeling). In contrast to Singh & Singh, a more high-level perspective was taken by Zellal and Zaouia (2017). In their work, they examined general factors that influence the quality of data in data warehouses. Therefore, the authors proposed several factors that may affect the data quality loosely based on literature (Zellal and Zaouia, 2016a). They further developed a measurement model (Zellal and Zaouia, 2016b) to enable the measurement of these factors. Based on this preliminary work, they conducted an empirical study and found that technology factors (i.e., features of ETL and data quality tools, infrastructure performance, and type of load strategy) are the most critical factors that influence data quality in data warehouses. We will compare these contributions with our work in Section 6.1.

2.2.2. Related work on data processing topic issues

Bagherzadeh and Khatchadourian (2019) investigated difficulties for big data developers by mining Stack Overflow questions. They found connection management to be the most difficult topic. In contrast to our work, where we zoom into data processing, they focused on the rather broad and more abstract area of big data. Islam et al. (2019b) and Alshangiti et al. (2019) analyzed the most difficult stages in ML pipelines for developers. Both studies identified data preparation as the second most difficult stage in ML pipelines. Wang et al. (2022) analyzed 1,000 Stack Overflow questions related to the data processing framework Apache Spark. They reported that questions regarding data processing were among the most prevalent issues developers have, with 43% of all analyzed questions asked. We will compare these contributions with our work in Section 6.2.

2.2.3. Related work on data pipelines

The quality of the data delivered by data pipelines is closely coupled with the quality of the pipelines themselves. Well-engineered pipelines are likely to provide data of high quality. Following, we give a brief overview of research directions in the context of pipelines that contribute to increasing their quality.

Experiences & guidelines. There is a large volume of literature providing frameworks (Badidi et al., 2018; Oleghe and Salonitis, 2020), guidelines (Ismail et al., 2019; Tardio et al., 2020), or architectures (Ronkainen and Iivari, 2015; Helu et al., 2020) to foster the development and use of data pipelines. While many contributions focused on specific application domains, e.g., manufacturing (O'Donovan et al., 2015; Frye and Schmitt, 2020), others took a more generic approach (Von Landesberger et al., 2017; Munappy et al., 2020a). Further, there are a number of studies that share experiences (e.g., lessons learned, challenges) about engineering data pipelines (Goodhope et al., 2012; Tiezzi et al., 2020; Munappy et al., 2020b).

Quality aspects. Research providing a comprehensive overview of the quality aspects of data pipelines is rare. The studies available mostly focus on specific quality characteristics of pipelines, for example, performance and scalability (Bhandarkar, 2017; Van Dongen and Van Den Poel, 2021) or robustness (Munappy et al., 2021). However, extensive investigations of quality characteristics were conducted on ETL processes (Alves et al., 2014; Theodorou et al., 2014; Akkaoui et al., 2019). As a sub-concept of data pipelines, ETL processes, and thus their quality characteristics, are highly relevant to data pipelines.

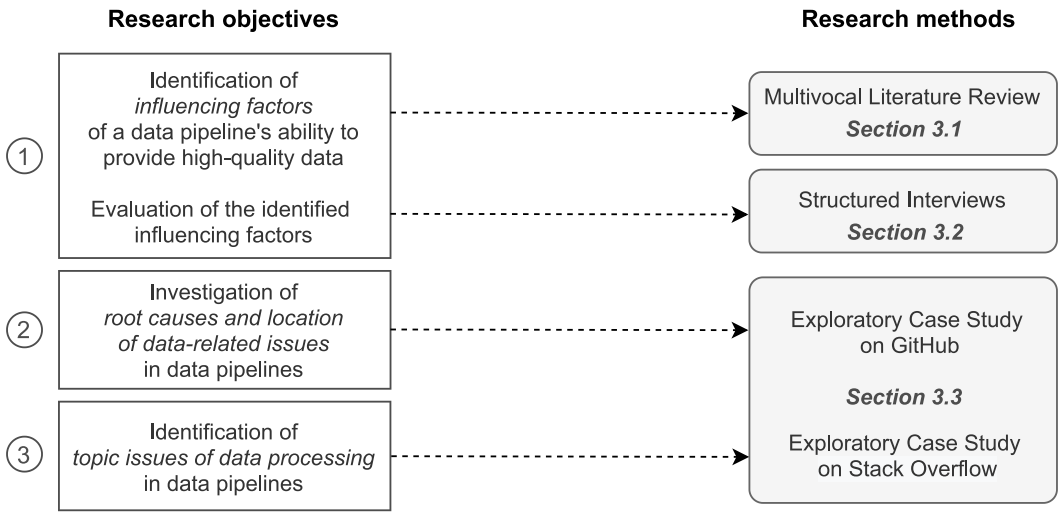

Fig. 2. Research procedure.

Development and maintenance support. Recently, several research works have focused on supporting data and software engineers in developing and maintaining data pipelines. There are plenty of works that presented tools for debugging pipelines (Zwick, 2019; Rezig et al., 2020; Lourenço et al., 2020; Kuchnik et al., 2021). Further, several publications aimed to improve the reliability of pipelines by addressing provenance aspects of pipelines (Wang et al., 2020; Rupprecht et al., 2020; Grafberger et al., 2021).

Data preprocessing. Besides work on pipelines in general, there is also plenty of research that focuses on the preprocessing of data within pipelines. Studies in this area specifically aim to assist in choosing appropriate data operators and defining their optimal combinations (Bilalli et al., 2019; Yan and He, 2020; Desai and Dinesha, 2020) or even automate the entire creation of preprocessing chains (Giovanelli et al., 2022).

3. Research procedure

In this paper, we seek to improve the understanding of data pipeline quality. In fact, this research attempts to address the following objectives.

1. To identify and evaluate *influencing factors* of a data pipeline's ability to provide data of high quality.
2. To analyze *data-related issues* and elaborate on their *root causes* and *location* in data pipelines.
3. To identify *data pipeline processing topic issues* for developers and analyze whether they correspond to the typical processing stages of pipelines.

An overview of all applied research methods and the corresponding research objectives is depicted in Fig. 2. The remaining section describes the research methods used to reach these objectives.

3.1. Multivocal literature review

To identify general factors that influence data pipelines regarding their provided data quality, we conducted a Multivocal Literature Review (MLR). The goal of the literature review was to synthesize the available literature related to aspects that may affect the ability of pipelines to deliver high data quality. We gathered problem-related and quality-related aspects of pipelines to derive corresponding IFs.

Table 1
Search strings.

Google Scholar
['data pipeline' AND 'software quality'] OR ['data pipeline' AND 'quality requirements'] OR ['data pipeline' AND 'code quality'] OR ['data pipeline' AND '*functional requirements'] OR ['data pipeline' AND 'quality model'] OR ['data science pipeline'] OR ['data engineering pipeline'] OR ['data pipeline'] OR ['machine learning pipeline'] OR ['data flow pipeline'] OR ['data *processing pipeline'] OR ['data processing pipeline']
Google Search Engine
'data pipeline' AND ('software quality' OR 'quality requirements' OR 'code quality' OR '*functional requirements' OR 'quality model' OR 'pitfall' OR 'error' OR 'anti pattern' OR 'problem' OR 'challenges' OR 'issues')

3.1.1. Search process and source selection

As a first step, we conducted a trial search on the Google and Google Scholar search engines to identify relevant keywords and elaborate the search strategy. While the Google search engine returned a large number of results for the term 'data pipeline', Google Scholar returned instead few in comparison. One possible reason we encountered in our trial search could be that modified versions of the term data pipeline (e.g., data processing pipeline) are often used in the scientific literature, whereas the term data pipeline is commonly used in practice. Another reason could be that research in the developing field of data engineering and its concepts (e.g., data pipeline) is still relatively new.

Because of these differences in the search results, we decided to use different sets of keywords for Google Scholar and the regular Google search engine. The detailed search strings can be found in Table 1. For comprehensibility, the search string used for the regular Google search engine was rewritten with distributive law.

We performed initial searches with these search strings on both search engines. By scanning each hit's title, abstract, and conclusion on Google Scholar, we selected the first set of potentially relevant scientific sources. Potential relevant grey literature was selected based on the title and introduction of each hit on Google's regular search engine.

Based on their generic nature, some keywords (e.g., quality model, error, issue) caused a very large number of hits. To reduce the number of hits to a manageable size, we restricted the search space by utilizing the relevance ranking algorithms of both databases. That is, we assumed that the most relevant sources usually appear on the first few result pages. Thus, we only checked the first three pages of each search string's result (i.e., 30 hits) and only continued if a relevant source was found on the last page.

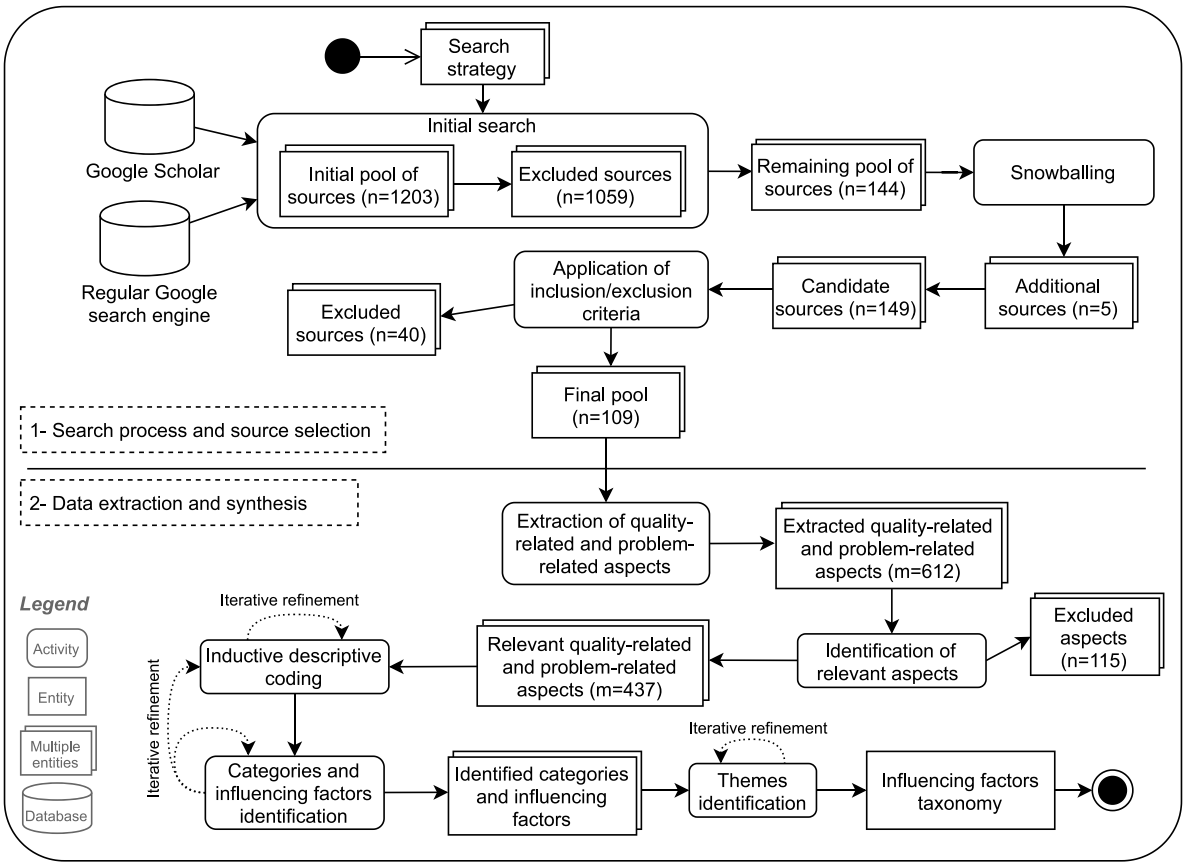

Fig. 3. Multivocal literature review process.

Table 2
Inclusion and exclusion criteria.

Inclusion criteria	Exclusion criteria
Accessible in full-text	Non-english articles
Published between 2000 and 2021	Only address machine learning aspects
Addressing quality characteristics, best practices, lessons learned, requirements *or* problems, issues, challenges related to data pipelines	

In total, we excluded 1,059 sources from an initial pool of 1,203 scanned sources resulting in a remaining pool of 144 potential relevant sources. Of these, 111 were found with Google Scholar and 33 with Google's regular search engine (i.e., grey literature).

To ensure finding all relevant sources, we additionally applied forward and backward snowballing (Wohlin, 2014) to the 111 scientific sources. By examining the references of a paper (backward snowballing) and citations to a paper (forward snowballing), we identified five additional papers. Thus, we got a final set of 149 candidate sources.

As a next step, we reviewed each of the 149 sources in detail based on the defined inclusion and exclusion criteria shown in Table 2. To ensure the validity of the results, two researchers independently voted on whether to include or exclude each source. In case of disagreements, the corresponding sources were discussed again, and the final choice was made. Finally, we excluded 40 sources and got a remaining pool of 109 sources (26 grey literature, 83 scientific literature) for further consideration. The complete process of the literature review is depicted in Fig. 3.

3.1.2. Data extraction and synthesis

We used a Google Sheets spreadsheet to extract all quality-related (i.e., best practices, quality characteristics) and problem-related (i.e., issues, challenges) aspects of data pipelines from the final pool of sources. In detail, we extracted text fragments describing either quality- or problem-related aspects that may influence the data quality of data pipelines and entered them in separate columns in the spreadsheet. To ensure clarity, we additionally entered context information as notes for some extracted text fragments. Note that whereas some sources contained both quality- and problem-related aspects, others contained information on only one aspect.

After all text fragments (612) were extracted, we reviewed them based on their potential influence on a data pipeline's ability to deliver high data quality. We excluded text fragments (175) describing either too abstract quality concepts (e.g., the flexibility of pipelines), too general (e.g., data transformation error), or not impacting the quality of data provided by pipelines (e.g., cosmetic bugs of graphical user interfaces). To be able to assess the influence on the data quality, we relied on the ISO/IEC's (ISO/IEC, 2008) inherent data quality characteristics, i.e., accuracy, completeness, consistency, credibility, and currentness.

The remaining data (437 text fragments) were then synthesized using the thematic synthesis approach. Thematic synthesis is a qualitative data analysis technique that aims to identify and report recurrent themes in the analyzed data (Cruzes and Dybå, 2011a,b). This technique was chosen because it is frequently used by studies for similar purposes (Badampudi et al., 2016; Fontão et al., 2017)

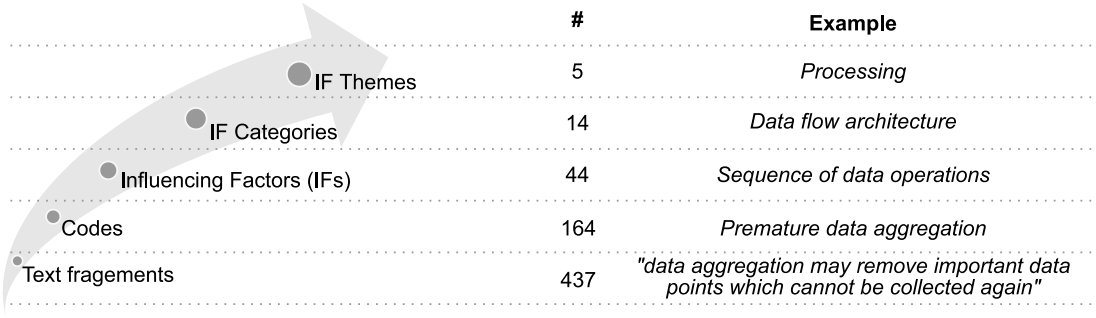

Fig. 4. Thematic synthesis procedure.

and fits well with different types of evidence (i.e., scientific and grey literature) (Cruzes and Runeson, 2015).

We started the synthesis by applying descriptive labels (i.e., codes) to each data fragment following an inductive approach. Thus, we generated the codes purely based on the concept the text fragment described. In detail, we derived neutral code names that described the underlying influencing aspect of the extracted quality- and problem-related text fragments. After the fragments were coded, two researchers created a visual map of the codes and reviewed them together. Thereby, it was recognized that the level of abstraction varied widely between the codes. Thus, we defined codes that represent higher-level concepts as IF categories (e.g., monitoring) or directly as IFs (i.e., data lineage). In addition, we identified IFs (e.g., code quality) based on the similarity of the codes (e.g., glue code, dead code, duplicated code) and derived overarching IF categories (e.g., software code). During this process, we constantly relabeled and subsumed codes and refined the emerging IFs and categories. In fact, the procedure was carried out in an interactive and iterative manner by two researchers.

As the last step, the identified IF categories were reviewed together, and a set of concise higher-order IF themes was created to succinctly summarize and encapsulate the categories. The themes were developed based on the experience of the researchers in previous studies (Golendukhina et al., 2022; Foidl et al.) and refined by reviewing relevant current literature (Lenarduzzi et al., 2021; Munappy et al., 2020a; Martínez-fernández et al., 2022). Each researcher then independently assigned all categories to a theme. In the case of different assignments, the corresponding category, as well as themes, were discussed again with a third researcher and refined to reach a consensus. Finally, a terminology control (Usman et al., 2017) of all IFs, categories, and themes was executed to ensure a consistent and accurate nomenclature. Fig. 4 shows the procedure of the thematic synthesis illustrated with an example. In total, we identified five IF themes comprising 14 IF categories and a further 41 IF based on 164 codes assigned to 437 text fragments.

3.2. Expert evaluation

To evaluate our findings and strengthen the trustworthiness of the identified IFs, we conducted structured interviews following the guidelines of Hove & Anda (Hove and Anda, 2005) with eight experts to collect empirical evidence about our identified IFs.

The purpose of the interviews was to validate the identified factors. In total, we collected opinions from eight experts with a minimum of three years of practical experience in the field of data engineering or a similar field where data pipelines are used.

The interview consisted of two parts: the profiling of the respondents with questions about their experience and background; and the main part with questions about the IFs. To not overwhelm the experts and receive quality feedback, we decided to validate the 14 IF categories and provide the IFs as examples for each category.

The opinions of the experts regarding the influence of a certain IF category on data quality were measured with a 4-point Likert scale including high, medium, low, or no influence. We also allowed no answer in case participants did not have sufficient experience with a category or were not certain about their answers.

3.3. Empirical study

To get a better understanding of the main problems a data pipeline encounters in its main task, i.e., processing data, we conducted two exploratory case studies. The first study aimed at identifying the root causes of data-related issues and their location in a data pipeline. To achieve this, we analyzed open-source GitHub projects that have a data pipeline as one of its main components. In the second study, we analyzed Stack Overflow posts to identify the main topics developers ask about processing data and examine whether the found problem areas correspond to the typical processing stages of pipelines. The process of mining GitHub and Stack Overflow is presented in Fig. 5 and described in the following sections.

3.3.1. Exploratory case study on GitHub

For mining GitHub, we followed the procedure used by a related study of Ray et al. (2016). First, we analyzed projects on GitHub and identified those that were suitable for our analysis. The main inclusion criterion was that a project either has a data pipeline as one of the main components or includes several data processing steps before further data application. Following these criteria, we identified 11 projects. The list of the projects, their description, and an overall number of open and closed issues are presented in Table 3.

Next, we extracted 12,345 open and closed issues reported by users or developers from these projects. Since it was unfeasible to analyze over 12,000 posts manually, a random sample of the population was taken considering the confidence level of 95%. For our population, the minimum sample size is equal to 373. To cover the minimum properly, we chose a sample size of 400.

In the next step, one researcher manually labeled each issue by assigning three groups: data-related, not data-related, and ambiguous. The last category was added for the issues, including project-specific names or issues that cannot be classified without deeper knowledge of the project. These issues were not included in the further analysis since they do not represent common but rather project-specific issues. As a result, 42 issues were classified as data-related and used in further analysis.

Because only approximately 10% of all the sample issues were classified as data-related, we decided to apply an additional strategy to increase the number of data-related issues further. The main idea of this strategy was to reduce the total number of extracted issues (12,345) based on a set of keywords that are related to the non-data-related issues. These keywords were extracted from the non-data-related issues from the initial sample. We validated them by checking that they were not present in the data-related issues. The list of the non-data-related keywords is presented in Table 4.

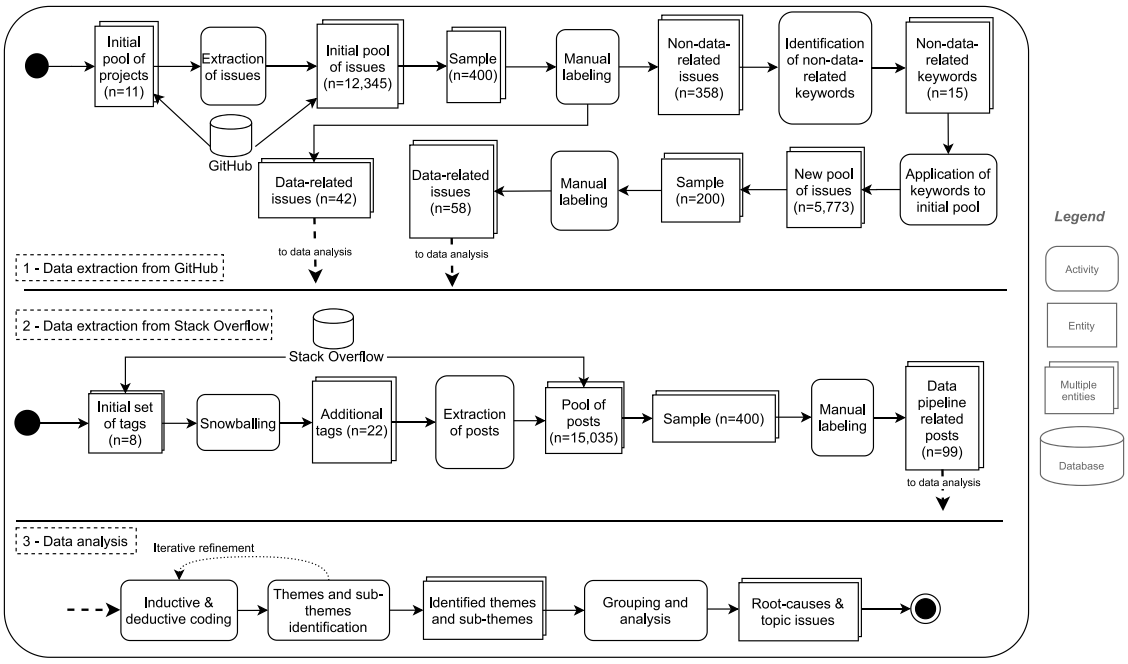

Fig. 5. GitHub and Stack Overflow mining procedure.

Table 3
Analyzed GitHub projects.

№	Project name	Project description	# of Issues
1	ckan	Data management system	2908
2	covid-19-data	Data collector of COVID-19 cases, deaths, hospitalizations, tests	888
3	DataGristle	Tools for data analysis, transformation, validation, and movement.	69
4	dataprep	A tool for data preparation	376
5	doit	Task management and automation tool	269
6	flyte	Kubernetes-native workflow automation platform for complex, mission-critical data and ML processes at scale	1442
7	networkx	Network Analysis in Python	2712
8	opendata.cern.ch	Source code for the CERN Open Data portal	1603
9	pandas-profiling	HTML profiling reports from pandas DataFrame objects	564
10	pybossa	A framework to analyze or enrich data that cannot be processed by machines alone.	999
11	rubrix	Data annotation and monitoring for enterprise NLP	515

Table 4
Non-data-related keywords.

UI, API, support, version, tutorial, guide, instruction, license, picture, GitHub, typo, logo, documentation, readme, graphics

Afterward, we applied the set of non-data-related keywords to the total pool of issues and identified 6,172 issues (52%) as not data-related. In the next step, we took a sample of 200 issues from the remaining 5,773 issues while considering the distribution of all issues in every chosen project, thereby identifying 58 additional data-related issues. As a result, the final set consisted of 100 data-related issues extracted from 11 different projects.

To assign issues to the root causes and data pipeline's stages, we applied descriptive labels following an inductive (root causes) and deductive (stages) approach. If the root cause or stage were not identifiable from the issue description, the source code and solution of the issue, if available, were examined. To maintain the objectivity of the results, the labels were then investigated by two other researchers until an agreement was reached.

3.3.2. Exploratory case study on stack overflow

We followed the procedure described by Ajam et al. (2020) for mining Stack Overflow posts. The process is shown in Fig. 5.

One of the features of the platform is the ability to add tags to every question according to the topic the given question covers. A tag is a word or a combination of words that expresses the main topics of the question and groups posts into categories and branches.

If there is more than one topic to which a question belongs, several tags can be applied. Each post can have up to five different tags. To identify the questions relevant to our study, we defined a starting collection of data pipeline processing-related tags. It includes the main tasks of data pipelines defined in the earlier sections such as: 'data pipeline', 'data cleaning', 'data integration', 'data ingestion', 'data transformation', and 'data loading'. In addition, we added two tags, 'Pandas' and 'Scikit-learn', that describe packages frequently used in processing data in pipelines.

To identify and extract posts containing these tags, we used Stack Overflow API, which provides various options for interaction and parsing of the website through commands. API facilitates the process of post-collection and allows the specification of the required tags. Since every post can have up to five tags simultaneously, API helps to extract more related tags based on the initially added tags. Such functionality

Table 5
Data pipeline processing-related tags on Stack Overflow.

arrays, classification, data-augmentation, data-cleaning, dataframe, data-ingestion, data-integration, data-loading, data-pipeline, data-preprocessing, data-science, data-transformation, data-wrangling, datetime, deep-learning, discretization, etl, feature-selection, image-processing, keras, kettle, machine-learning, numpy, opencv, pandas, pdi, pentaho-data-integration , python, scikit-learn, tensorflow

allows the application of the snowball method where new tags are discovered based on the original ones (Alshangiti et al., 2019). Based on the starting tags, API showed up to eight related tags. We repeated the procedure for several iterations until no new tags were identified. As a result, we got 30 tags shown in Table 5 and extracted all posts containing these tags. After handling duplicates, we got a final pool of 15,035 posts from 11,290 different Stack Overflow forum branches.

Since it was unfeasible to analyze over 15,000 posts manually, we randomly selected a sample of 400 posts using a 95% confidence level. From these posts, we excluded 301 posts asking general questions about the usage of certain libraries or specific data analytics questions (e.g., about image transformation, feature extraction or selection, dimensionality reduction, algorithmic optimization, visualization, or noise reduction).

Afterward, one researcher investigated all 99 remaining sample posts and first assigned deductive labels describing the usual data processing stages of a pipeline. In several iterations, the labels were grouped, refined, and new labels were created until the main data pipeline processing topic issues were identified. The whole labeling process was constantly verified by a second researcher.

4. Data pipeline influencing factors

This section deals with influencing factors (IFs) affecting a pipeline's ability to deliver quality data. First, Section 4.1 presents the IFs in the form of a taxonomy. Afterward, Section 4.2 outlines the evaluation of the taxonomy in the form of structured expert interviews.

4.1. Taxonomy of influencing factors

The taxonomy is depicted in Fig. 6. In total, we identified 41 IFs grouped into 14 IF categories and five IF themes. Following, we describe each identified theme, namely *Data*, *Development & deployment*, *Infrastructure*, *Life cycle management*, and *Processing*, and provide a description of all identified IFs.

4.1.1. Data

This theme covers aspects of the data processed by pipelines that may affect a pipeline's ability to process and deliver these data correctly. The aspects are represented by the following three categories: *Data characteristics*, *Data management*, and *Data sources*. We identified four factors of influence related to the category Data characteristics: 'Data dependencies', 'Data representation', 'Data variety', and 'Data volume'. The category data management comprises the IFs 'Data governance', 'Data security', and 'Metadata'. Note that the IFs data governance and security only relate to the pipeline and not to upstream processes (e.g., data producers). Finally, the 'Complexity' and 'Reliability' of sources providing data are IFs summarized under the category Data sources. A description of each identified IF is given in Table 6.

4.1.2. Development & deployment

The development and deployment processes of pipelines were identified as significant aspects that may influence a pipeline's data quality. This theme is structured into four categories: *Communication & information sharing*, *Personnel*, *Quality assurance*, and *Training-serving skew*. Within the category of Communication & information sharing, we identified the 'Awareness' to perceive a pipeline as an overall construct that comprises different actors and 'Requirements specifications' as IFs. Regarding the category Personnel, 'Expertise' and 'Domain knowledge' were found as important IFs. 'Testing scope' and applying 'Best practices' manifest the IFs of the category quality assurance. Concerning the category Training-serving skew, 'Code equality' between development and deployment and 'Data drift' constitute factors influential to the data quality provided by pipelines. Table 7 provides a description of all identified IFs of this theme.

4.1.3. Infrastructure

This theme reflects important aspects of the infrastructure data pipelines are based on that may affect their ability to provide data of high quality. The identified IFs of this theme are grouped into two categories: *Serving environment* and *Tools & technology*. The category Serving environment encompasses infrastructural factors related to pipelines that are running in production, that is, 'Hardware', 'Performance', and 'Scalability'. Under the IF category Tools and technology, we identified the factors 'Appropriateness', 'Compatibility', 'Debugging capabilities', 'Functionality', 'Heterogeneity', 'Reliability', and 'Usability'. This category covers tools and technologies used during development and in the operational state of pipelines. In Table 8 a description of each identified IF of this theme is given.

4.1.4. Life cycle management

The life cycle management of data pipelines has been found to be influential on the quality of the data delivered by pipelines. Within this theme, two IF categories were identified: *Application management* and *Monitoring*. Important IFs of the category Application management are: 'Configuration management', 'Continuous integration and deployment', and 'Workflow and orchestration management'. The category Monitoring comprises the factors 'Application performance monitoring' and 'Data lineage'. Table 9 describes all IFs of both categories in more detail.

4.1.5. Processing

This theme includes factors that are related to the processing of the data in pipelines. In total, we identified eleven processing factors that may impact the quality of the processed data. We grouped them into three categories: *Data flow architecture*, *Functionality*, and *Software code*. Regarding the category Data flow architecture, following factors were found to be influential: 'Complexity', 'Dependencies', 'Modularization', 'Processing mode', 'Reproducibility', and 'Sequence of data operations'. The category Functionality summarizes the IFs 'Automation', 'Configuration', and 'Data cleaning'. Finally, 'Code quality' and 'Data type handling' comprise the IFs of the category Software code. Table 10 provides a description of all eleven identified processing factors.

4.2. Expert evaluation

To assess the validity of the identified influencing factors, we conducted eight structured interviews with experts from different business areas. A prerequisite for the candidates was at least three years of experience in data engineering. Half of the respondents had three to five years of experience, and the others had more than five years of experience, including two experts with more than ten years of experience. Seven experts assessed all 14 IF categories, while one expert only assessed 11 of them. Additionally, we asked about the format of data being processed in their data pipelines. The main types of data mentioned were tabular data and text.

The results of the experts' evaluation of the taxonomy are presented in Fig. 7. The Y-axis represents the average influence of all 14 assessed IF categories assessed by the experts. Each category is represented by a bubble while the color of the bubble highlights the corresponding IF theme. The size (i.e., area) of the bubbles represents the experts' level of agreement on every IF category and is calculated as the standard deviation. High standard deviation coefficients evidence low agreement

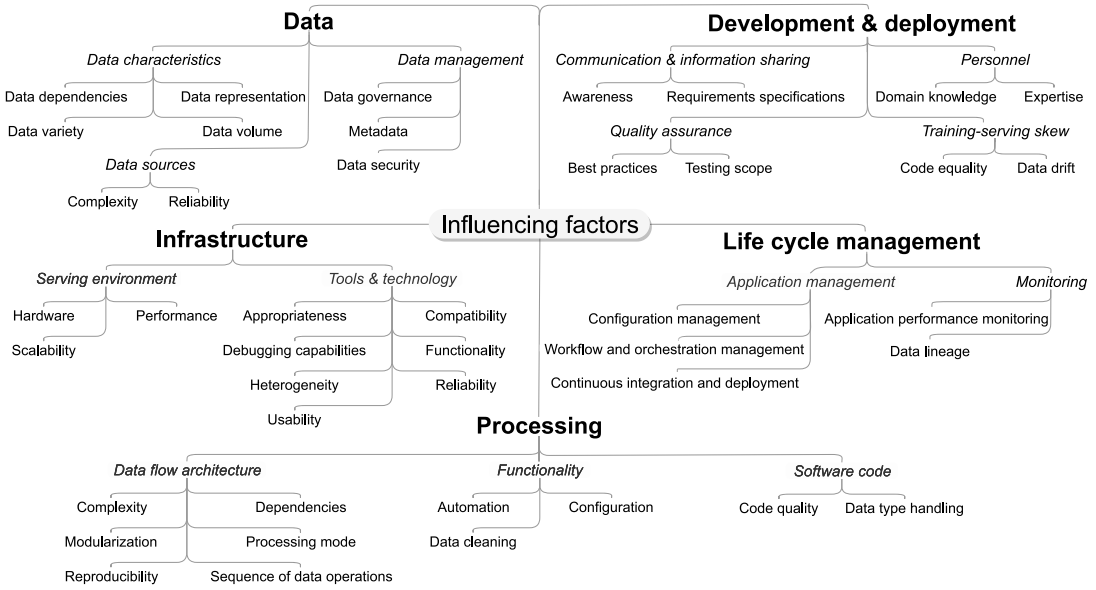

Fig. 6. Taxonomy of data pipeline influencing factors (IFs).

Table 6
Influencing factors (IFs) of the theme Data.

IF Category & IF	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 data sources a pipeline has to deal with, including the degree of simplicity of merging the corresponding data (e.g., joinability).
Reliability	Degree to which data sources are available, accessible, and provide data of high quality.

Table 7
Influencing factors (IFs) of the theme Development & deployment.

IF Category & IF	Description
Communication & information sharing	
Awareness	The perception of the pipeline as a coherent overall construct and the awareness and ability of knowledge exchange.
Requirements specifications	Completeness and level of detail regarding the specification and documentation of the pipeline (e.g., data transformation rules, processing requirements).
Personnel	
Domain knowledge	Entities' knowledge and understanding of the domain underlying the data.
Expertise	Background, experiences, technical knowledge, and quality awareness of entities.
Quality assurance	
Best practices	Code and configuration reviews, refactoring, canary processes, and further best practices (e.g., use of control variables).
Testing scope	Testing depth and space (e.g., test coverage, test cases).
Training-serving skew	
Code equality	Equality of software code between development and production (e.g., ported code, code paths, bugs).
Data drift	Differences in the data between development and production (e.g., encodings, distribution).

Table 8
Influencing factors (IFs) of the theme Infrastructure.

IF Category & IF	Description
Serving environment	
Hardware	Type and reliability of the hardware in production.
Performance	Availability and manageability of the resources in production.
Scalability	The capacity to change resources in size or scale.
Tools & technology	
Appropriateness	Types (e.g., code/GUI-first, own solutions, propriety), maturity, flexibility, and up-to-dateness of tools and technology.
Compatibility	Ability of tools and technology (e.g., frameworks, platforms, libraries) to work together.
Debugging capabilities	Availability and extent (i.e., level of detail) of opportunities (e.g., tools) to find and correct errors.
Functionality	The scope of functions (e.g., advanced data operations, support of data management, engineering, and validation) provided.
Heterogeneity	Number of different technologies and tools used.
Reliability	Degree to which tools and technologies are working correct (e.g., software quality, complexity).
Usability	Availability, documentation, and ease of use of APIs, libraries, and tools.

Table 9
Influencing factors (IFs) of the theme Life cycle management.

IF Category & IF	Description
Application management	
Configuration management	Version control and dependency management for code, libraries, and configurations.
Continuous integration and deployment	Automation of providing new releases and changes.
Workflow and orchestration management	Tools for managing the entire workflow of pipelines.
Monitoring	
Application performance monitoring	Observing and logging the operational runtime of all software components.
Data lineage	Observability of the data during all transformation steps, including data versioning.

Table 10
Influencing factors (IFs) of the theme Processing.

IF Category & IF	Description
Data flow architecture	
Complexity	Computational effort and pipeline complexity (e.g., transformation complexity).
Dependencies	Dependencies between architectural components or processing steps.
Modularization	Modularization of data pipeline components (e.g., codes, APIs).
Processing mode	Type of processing (e.g., batch, stream, distributed or parallel processing).
Reproducibility	Degree to which processing is reproducible, including aspects such as caching and idempotency.
Sequence of data operations	Application order of data preparation and processing techniques.
Functionality	
Automation	Automation of data validation, handling, and providing corresponding guidance.
Configuration	Configuration (e.g., parameters, settings) for processing data.
Data cleaning	Treatment of data issues (e.g., modification or deletion).
Software code	
Code quality	Quality and complexity of the code for processing data (e.g., dead code, duplicated code, language heterogeneity).
Data type handling	Handling of data types (e.g., type casting, delimiter handling, encoding).

among the experts, which is expressed by smaller sizes of the bubbles. As a reference, the two gray shaded bubbles on the lower right-hand side of the figure visualize the range between no and full agreement.

The majority of categories, eight out of 14, were assessed similarly by experts, i.e., within one point (influence level) difference. The largest disagreement appeared in five categories: data management, data sources, personnel, serving environment, and tools & technology. Although the average assessed influence of serving environment-related IFs received the lowest score, it also showed the lowest agreement among experts from no influence to medium influence. Notably, this category received a "no influence" assessment from the expert who did not assess all categories. Besides the three no answers, there were 51 high, 49 medium, eight low, and one no influence assessments.

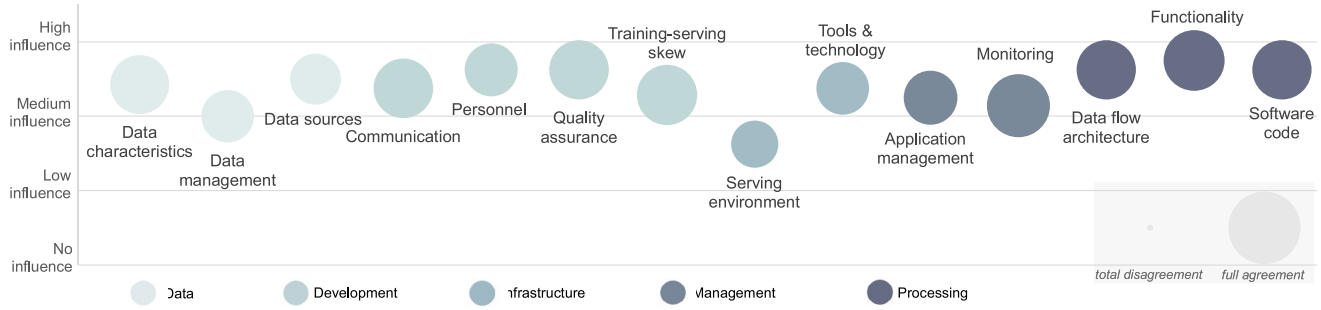

Fig. 7. Experts evaluation of IF categories and the levels of their agreement.

Despite different levels of agreement, 13 out of 14 categories were identified by the majority of experts to have medium to high influence. According to the experts' assessment, the five most influential categories are functionality, personnel, quality assurance, data flow architecture, and software code. They fall under the development and deployment, and processing categories. Overall, processing-related IFs were rated highly influential, with a high level of agreement among the experts. In conclusion, the expert interviews have confirmed that the categories have the potential to affect the data quality of a data pipeline.

5. Empirical study on data-related and data processing issues

In this section, we first outline the root causes of data-related issues and their location in data pipelines based on analyzing GitHub projects (Section 5.1). Afterward, Section 5.2 presents the main data pipeline processing problem areas for developers identified by mining Stack Overflow posts and compares them to the typical processing stages of pipelines.

5.1. Root causes and stages of data-related issues in data pipelines

After manually analyzing 100 data-related issues from the chosen GitHub projects, we identified seven root cause categories of these issues. The categories are listed on the left-hand side of Fig. 8.

The most frequent root cause identified in 33% of the analyzed problems is related to *data types*. These issues occur at almost every stage of the data pipeline; incorrectly defined data types make it difficult to clean, ingest, integrate, process, and load the data. It can lead to more obvious issues when the data cannot be processed due to unknown or incorrect data types, but also it can cause loss of information if data cannot be read and no error is raised. Data type issues are not limited to single data items but also concern how data types are handled in data frames. In almost 90% of all cases, data type issues arise in the cleaning or integration stages.

Issues caused by the misplacement of *symbols and characters* account for 17% of all investigated issues. Special characters, such as diacritics, letters of different alphabets, or symbols not supported by used encoding standards, cannot be processed and cause errors in the data pipeline.

The next identified category of root causes is related to *raw data*. This category describes all issues rooted in the raw data. For example, duplicated data, missing values, and corrupted data. Mostly, the issues appear at the ingestion and integration stages.

Functionality issues account for 13% of the issues and describe misbehavior of functions or lack of necessary functions. About half of the issues occurred during the cleaning process and characterize wrong outputs of the cleaning functions, i.e., correctly processed data are recognized as incorrect after the cleaning stage of a pipeline. Further issues describe functions delivering inconsistent results or not processing the data as the function intended it.

Data frame-related issues were identified in 11% of all issues. They cover all data frame-related activities, such as data frame creation, merging, purging, and other changes. Additionally, some of the issues are connected to the access of different groups of users to the data.

The last two categories were related to processing large data sets and logical errors, with seven and two percent, respectively. The *input data set size* can cause problems at different pipeline stages. The issues discovered during the analysis included failure to read, upload, and load large data sets. Therefore, the ability of the data pipeline to scale according to the amount of data must be considered already in the early stages of data pipeline development. Issues in the category *logical errors* describe, for example, calls to the non-existing attributes of objects or non-existing methods.

An overview of the pipeline stages where the analyzed data-related issues occurred is shown on the right-hand side of Fig. 8. Most of all issues manifested during the cleaning and ingestion stages of the pipelines, with 35% and 34%, respectively. Further, 21% of all issues were detected at the integration stage. The stages with the fewest issues seen are loading and transformation.

5.2. Main topic issues of data processing for developers in data pipelines

After manually labeling all 99 data pipeline processing-related posts, we identified six main data pipeline processing topic issues, five of which represent typical data pipeline stages: integration, ingestion, loading, cleaning, and transformation. The sixth topic identified was compatibility. Fig. 9 shows the number of posts related to different data pipeline stages and their distribution. Since one post may include more than one label, there are more than 99 labels represented in the figure.

The largest topic was related to *integration* issues. Here users mostly asked questions about the transformation of databases, operations with tables or data frames, and platform- or language-specific questions.

Posts related to ingestion and loading were the second and third most asked group of posts. Posts related to *ingestion* typically discussed the upload of certain data types and the connection of different databases with data processing platforms. Posts regarding *loading* mostly cover several issues: efficiency, process, and correctness. Efficiency relates to the speed of loading and memory usage. Posts about processes ask platform-related questions on how to connect different services. Correctness covers questions about incorrect or inconsistent results of data output.

Cleaning and transformation posts account for 13% of all posts. A large portion of all posts regarding *cleaning* include questions about the handling of missing values. Posts that were labeled as *transformation* cover such issues as the replacement of characters and symbols, data types handling, and others.

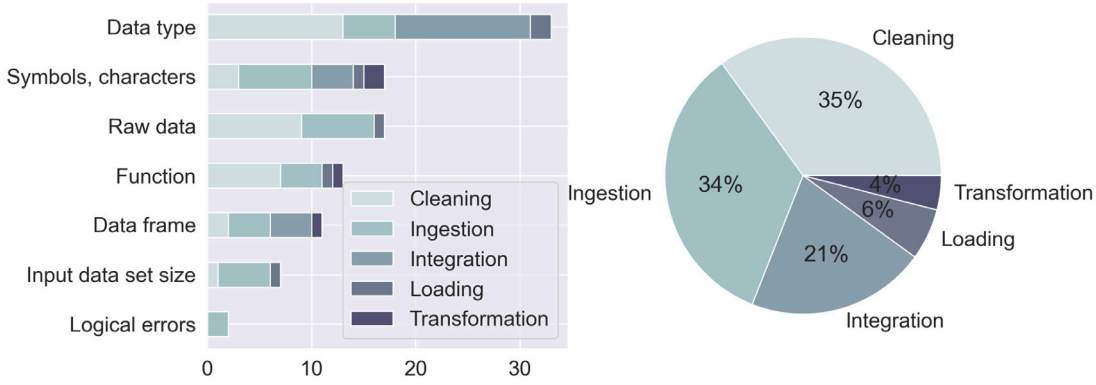

Fig. 8. Data-related issues' root causes and pipeline stage in GitHub projects.

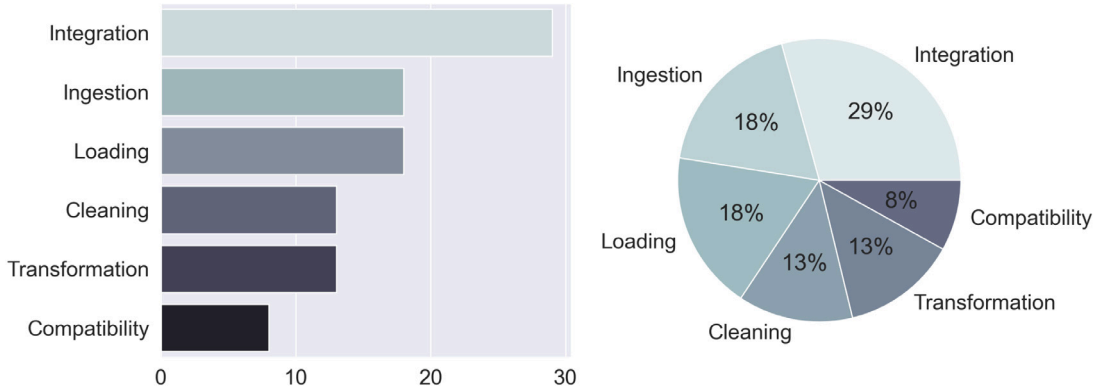

Fig. 9. Data pipeline processing topics asked in Stack Overflow posts.

The last topic identified is *compatibility*. This topic is not specific to any data pipeline stage but includes a block of questions regarding software or hardware compatibility for data pipeline-related processes. Examples are code running inconsistently in different operational systems, e.g., Ubuntu and macOS, programming language differences, software installation in different environments, and others. Compatibility issues are critical but hard to foresee. We found 8% of posts on Stack Overflow with a focus on compatibility issues in various stages of data pipelines. In all cases, the issues affected the core functionality of pipelines.

6. Discussion

This section discusses the findings of the research in connection with the related literature and describes the limitations of the study. Section 6.1 focuses on the taxonomy and its evaluation. Afterward, Section 6.2 highlights the core findings and limitations of the exploratory case studies on GitHub and Stack Overflow and connects them to related works. Finally, Section 6.3 provides an overview of the findings and their relations.

6.1. Data pipeline influencing factors taxonomy

Our developed taxonomy summarizes the factors influencing a data pipeline's ability to deliver high-quality data into five main pillars: data, development and deployment, infrastructure, life cycle management, and processing.

Interpretation. The identification of 41 IFs grouped into 14 categories underlines the complex and wide range of aspects that may affect data pipelines. This necessitates an urgent need to further study the influencing effects of each factor. Notwithstanding the relatively limited size of our evaluation, the following conclusions can be drawn. Processing-related IFs were identified as highly influential by the majority of the experts in the structured interviews. In contrast, IFs of the theme infrastructure were estimated to have low to medium influence with, however, low agreement rates between respondents. This leads to several conclusions. First, the impact of different IFs on the final data quality is not equal and homogeneous, and some IFs have a higher effect than others. Consequently, they should be considered in the first place. Second, the lower levels of experts' agreement regarding data sources, personnel, and the serving environment might indicate that the influence of different factors depends on the domain of the data pipeline application and the form of data (e.g., tabular, text, images). However, since the main purpose of the evaluation was to confirm the influencing ability of the factors, the results in terms of the relative influencing ranking must be interpreted with caution.

Comparison with related work. We briefly compare our taxonomy to two contributions closest to our work regarding the domain of data and the perspective taken. The first article was published by Singh and Singh (2010) and presents possible causes of data quality issues in data warehouses. In contrast to our work, the authors only consider the causes of poor data quality at a very detailed level and do not form overarching causes (i.e., IFs categories and themes). Moreover, they did not provide empirical validation of the proposed factors. The second article (Zellal and Zaouia, 2017) took a more high-level perspective and investigated general factors that influence the quality of data in data warehouses. In their work, Zellal & Zaouia found that technological factors (e.g., ETL and data quality tools), data source quality, teamwork, and the adoption of data quality management practices are the most critical factors that influence data quality in data warehouses. As these factors are also represented in our taxonomy (i.e., tools & technology, data sources, communication & information sharing, and best practices), our findings confirm their influential characteristics. Unlike Zellal & Zaouia's work,

however, we base our research on a comprehensive set of systematically gathered grey and scientific literature. In conclusion, our taxonomy differs from the presented related work in the following ways. First, we provide factors of influence on several abstraction levels (i.e., themes, categories, and factors). Second, we do not focus on any domain, thus ensuring the taxonomy is valuable to researchers and practitioners of all fields.

Limitations. Although the taxonomy was constructed in a way to mitigate threats of validity (e.g., several researchers worked on the literature review and data synthesis), the final taxonomy has several limitations. First, we took a software and data engineering perspective, thus not considering managerial or business-related factors. Second, although different types of data were considered when analyzing the literature, there is more evidence of challenges and data quality for tabular data than for other data types. This aspect can lead to a biased representation of IFs for different data types. Third, it is important to bear in mind possible IF relationships. In a study on factors influencing software development productivity, Trendowicz (2009) described dependencies between these factors. Similarly, data quality IFs may also be in a casual relationship which determines the final change of the quality of the data delivered by a pipeline. Nevertheless, more research is needed to investigate these relationships between different factors and examine their interdependence.

6.2. Data-related and data pipeline processing issues

By mining GitHub and Stack Overflow, we got more profound insights into the root causes of data-related issues, their location in data pipelines, and the main topics of data pipeline processing issues for developers.

Interpretation. The majority of the analyzed issues on GitHub were caused by incorrect data types. A possible explanation for this may be that many data handling and ML libraries use custom data types which often cause interoperability issues (i.e., type mismatch) within pipelines (Islam et al., 2019b,a; Zhang et al., 2020). This matches with the results of analyzing the main data pipeline processing topics developers ask on Stack Overflow. In fact, compatibility issues were found to represent a separate topic besides the typical data processing stages in pipelines. Regarding these stages, we found data integration and ingestion to be the most asked topics. This is in good agreement with the work of Kandel et al. (2012), who found integrating and ingesting data to be the most difficult tasks for data analysts. A further aspect worth mentioning is that data-related issues rarely occur in the data transformation stage of a pipeline. A possible explanation for this is that incorrect transformations do not cause errors that are immediately recognized and thus can stay undetected. A further interesting finding was that data-related issues mainly occurred in the data cleaning stage of a pipeline. In contrast, developers ask the most about integrating, ingesting, and loading data but not about cleaning data. Although we cannot reason that the amount of asked questions of a topic reflects its difficulty, this can partly be explained by the fact that most issues in the data cleaning stage can be attributed to the raw data and data type characteristics and, thus, not directly to developers' skills.

Comparison to related work. We first compare our case study on GitHub with the work of Rahman and Farhana (2021). In their work, the authors investigated the bug occurrence in Covid-19 software projects on GitHub. They found data-related bugs to be their own category and identified storage, mining, location, and time series as corresponding sub-categories. However, their classification of data-related issues is strongly based on the inherent characteristics of Covid-19 software. For example, location data bugs describe issues where geographical location information in the data is incorrect. In contrast, our work describes general root causes applicable to a broader range of domains. A further study that dealt with data bugs was published by Islam et al. (2019a). In their paper, the authors identified data-related issues as the most occurring bug type in deep learning software. However, the authors do not provide further empirical evidence on concrete root causes of these issues.

Regarding mining Stack Overflow, several other contributions used this platform to investigate topics and challenges for developers in related fields. Prior work (Alshangiti et al., 2019; Islam et al., 2019b) analyzed difficulties for developers in the closely related area of ML pipelines. Both studies found data preprocessing to be the second most challenging stage. However, these studies focused in particular on preparing data for ML models, e.g., data labeling and feature extraction. A further study (Wang et al., 2022) on issues encountered by Apache Spark developers identified data processing as the most prevalent issue, accounting for 43% of all questions asked. However, this study did not detail the main topics of data processing and maps them to the usual processing stage of a pipeline.

Limitations. To maintain the internal validity of the empirical studies, two researchers worked on the labeling independently. Inconsistencies were discussed and resolved together. However, the conducted exploratory case studies have several limitations. First, we solely used GitHub and Stack Overflow for our study. Thus, the generalizability of our results must be treated with caution. Second, the findings are limited by the tags used in both studies. To reduce these threats, we applied the snowballing method in defining the Stack Overflow tags and, besides using data-related tags in mining GitHub, non-data-related tags to ensure excluding only non-relevant issues. The findings of the studies are further limited by the fact that we relied on sampling during the analysis. Thus, we cannot guarantee the completeness of the identified results, although we chose a statistically significant sample of posts and issues for the detailed analysis. Moreover, similarly to the taxonomy limitations, most Stack Overflow posts are related to tabular data; thus the results can be biased towards other data types.

6.3. Overview of results and their relation

Recapitulating our main research objectives, we recognize the following relations shown in Fig. 10. Influencing factors contribute to the occurrence of the root causes of data-related issues. For example, the influencing factors 'data type handling' and 'data representation' explain the most frequent root cause 'data type'. This, in turn, suggests an interdependence between the influencing factors. Further, the typical data pipeline stages where data-related issues occur mainly reflect what developers ask about data pipeline processing. However, we found compatibility issues as a separate category of questions developers ask. It is possible that this category reflects the most identified root cause data type, especially as previous research links these aspects (Islam et al., 2019b,a; Zhang et al., 2020).

7. Conclusions

This article contributes to enhancing the understanding of data pipeline quality. First, we descriptively summarized relevant data pipeline influencing factors in the form of a taxonomy. The developed taxonomy of influencing factors emphasizes today's complexity of data pipelines by describing 41 influencing factors. The influencing ability of the proposed factors was confirmed by expert interviews. Second, we explored the quality of data pipelines from a technological perspective. Therefore, we conducted an empirical study based on GitHub issues and Stack Overflow posts. Mining GitHub issues revealed that most data-related issues occur in the data cleaning (35%) stage and are typically caused by data type problems. Studying Stack Overflow questions showed that data integration and ingestion are the most frequently asked topic issues of developers (47%). In addition, we further found that compatibility issues are a separate problem area developers face alongside the usual issues in the phases of data pipeline

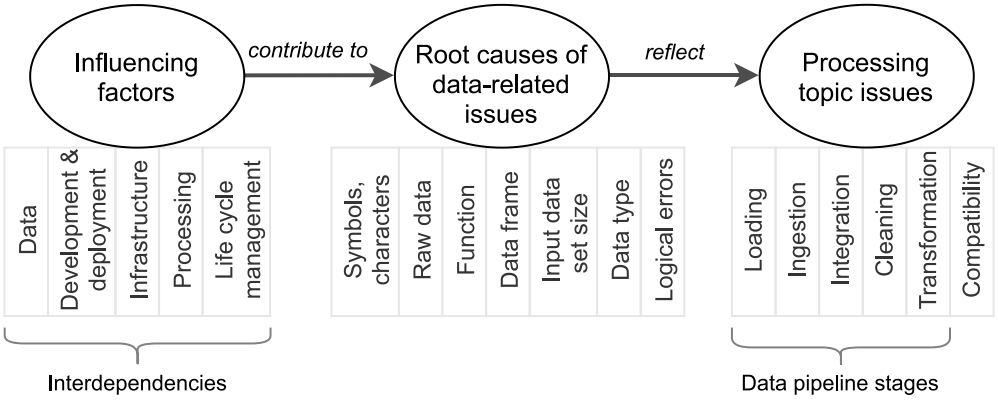

Fig. 10. Overview of influencing factors' contribution to data-related issues and processing topic issues in data pipelines.

processing (i.e., data loading, ingestion, integration, cleaning, and transformation).

The results of our research have several practical implications. First, practitioners can use the taxonomy to identify aspects that may negatively affect their data pipeline products. Thus, the taxonomy can serve as a clear framework to assess, analyze, and improve the quality of their data pipelines. Second, the root causes of data-related issues and their specific locations within data pipelines can help industry practitioners prioritize their efforts in addressing the most critical points of failure and enhancing the overall reliability of their data products. Third, the main data processing topics of concern for developers enable companies to focus on common challenges faced by developers, particularly in data integration and ingestion tasks, which can lead to more effective support and assistance for developers in tackling those issues.

Moreover, this study lays the groundwork for future research into data pipeline quality. First, further research should be carried out to explore the identified compatibility and data type issues developers face during engineering pipelines in more detail. Second, future studies need to determine a ranking of the IFs based on the analysis of the individual influence levels of each factor. Third, further work may extend our study aiming to infer the difficulty of each data processing stage by additionally considering difficulty metrics of Stack Overflow posts, e.g., the average time to receive an accepted answer or the average number of answers and views.

We intend to focus our future research on studying the dependencies between the proposed influencing factors. For this purpose, we plan to apply interpretive structural modeling. There are already promising applications of this modeling technique to uncover interrelationships between factors (Samantra et al., 2016).

CRediT authorship contribution statement

Harald Foidl: Conceptualization, Methodology, Investigation, Writing – original draft, Writing – review & editing, Visualization, Validation, Data curation, Project administration. **Valentina Golendukhina:** Methodology, Validation, Writing – original draft, Writing – review & editing, Visualization, Data curation. **Rudolf Ramler:** Funding acquisition, Writing – review & editing. **Michael Felderer:** Supervision, Funding acquisition, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the Austrian ministries BMK & BMAW and the State of Upper Austria in the frame of the SCCH competence center INTEGRATE [892418] part of the FFG COMET Competence Centers for Excellent Technologies Programme, and by the Austrian Research Promotion Agency (FFG) in the frame of the projects Green Door to Door Business Travel [FO999892583] and ConTest [888127]. We also thank Ilona Chochyshvili and Matthias Hauser for their contribution in the conduction of the exploratory case studies.

References

Agostinelli, S., Benvenuti, D., De Luzi, F., Marrella, A., 2021. Big data pipeline discovery through process mining: Challenges and research directions. CEUR Workshop Proc. 2952 (101016835), 50–55.

Ajam, G., Rodr, C., Sydney, U., 2020. API topics issues in stack overflow q&as posts: An empirical study.

Akkaoui, Z.E., Vaisman, A., Zim, E., 2019. A quality-based ETL design evaluation framework. ICEIS 1, 249–257.

Alshangiti, M., Sapkota, H., Murukannaiah, P.K., Liu, X., Yu, Q., 2019. Why is developing machine learning applications challenging? A study on stack overflow posts. In: International Symposium on Empirical Software Engineering and Measurement.

Alves, T.L., Silva, P., Dias, M.S., 2014. Applying ISO / IEC 25010 standard to prioritize and solve quality issues of automatic ETL processes. In: IEEE International Conference on Software Maintenance and Evolution. IEEE, pp. 573–576.

Ancker, J.S., Shih, S., Singh, M.P., Snyder, A., Edwards, A., Kaushal, R., investigators, H., 2011. Root causes underlying challenges to secondary use of data. In: AMIA Annual Symposium Proceedings. pp. 57–62.

Badampudi, D., Wohlin, C., Petersen, K., 2016. Software component decision-making: In-house, OSS, COTS or outsourcing - A systematic literature review. J. Syst. Softw. 121, 105–124.

Badidi, E., El Neyadi, N., Al Saeedi, M., Al Kaabi, F., Maheswaran, M., 2018. Building a data pipeline for the management and processing of urban data streams. In: Handbook of Smart Cities: Software Services and Cyber Infrastructure. pp. 379–395.

Bagherzadeh, M., Khatchadourian, R., 2019. Going big: A large-scale study on what big data developers ask. In: ESEC/FSE 2019 - Proceedings of the 2019 27th ACM Joint Meeting European Software Engineering Conference and Symposium on the Foundations of Software Engineering. pp. 432–442.

Bhandarkar, M., 2017. AdBench: A Complete Benchmark for Modern Data Pipelines. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 10080 LNCS, pp. 107–120.

Biessmann, F., Golebiowski, J., Rukat, T., Lange, D., Schmidt, P., 2021. Automated data validation in machine learning systems. Bull. IEEE Comput. Soc. Tech. Comm. Data Eng. 44 (1), 51–65.

Bilalli, B., Abelló, A., Aluja-banet, T., Wrembel, R., 2018. Intelligent assistance for data pre-processing. Comput. Stand. Interfaces 57, 101–109.

Bilalli, B., Abelló, A., Aluja-Banet, T., Wrembel, R., 2019. Presistant: Learning based assistant for data pre-processing. Data Knowl. Eng. 123 (August), 101727.

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. In: 44th International Conference on Software Engineering (ICSE '22), May 21-29, 2022, Pittsburgh, PA, USA. Association for Computing Machinery, pp. 2091–2103.

Bogner, J., Verdecchia, R., Gerostathopoulos, I., 2021. Characterizing technical debt and antipatterns in AI-based systems: A systematic mapping study. In: IEEE/ACM International Conference on Technical Debt. (TechDebt), pp. 64–73.

Bomanson, J., 2019. Diagnosing data pipeline failures using action languages. In: International Conference on Logic Programming and Nonmonotonic Reasoning. Springer, pp. 181–194.

Breck, E., Polyzotis, N., Roy, S., Whang, S.E., Zinkevich, M., 2019. Data validation for machine learning. In: Proceedings of Machine Learning and Systems (MLSys). pp. 334–347.

Carvalho, R., Lobo, M., Oliveira, M., Oliveira, A.R., Lopes, F., Souza, J., Ramalho, A., Viana, J., Alonso, V., Caballero, I., Santos, J.V., Freitas, A., 2021. Analysis of root causes of problems affecting the quality of hospital administrative data: A systematic review and ishikawa diagram. Int. J. Med. Inform. 156 (September), 104584.

Chapman, A., Missier, P., Simonelli, G., Torlone, R., 2020. Capturing and querying fine-grained provenance of preprocessing pipelines in data science. Proc. VLDB Endow. 14 (4), 507–520.

Cheburet, S.K., Odhiambo-Otieno, G.W., 2016. Process factors influencing data quality of routine health management information system: Case of Uasin Gishu county referral hospital , Kenya. Int. Res. J. Public Environ. Health 3 (6), 132–139.

Cho, S., Ensari, I., Weng, C., Kahn, M.G., Natarajan, K., 2021. Factors affecting the quality of person-generated wearable device data and associated challenges: Rapid systematic review. JMIR mHealth uHealth 9 (3), 1–12.

Cruzes, D.S., Dybå, T., 2011a. Recommended steps for thematic synthesis in software engineering. In: International Symposium on Empirical Software Engineering and Measurement. (7491), pp. 275–284.

Cruzes, D.S., Dybå, T., 2011b. Research synthesis in software engineering: A tertiary study. Inf. Softw. Technol. 53 (5), 440–455.

Cruzes, D., Runeson, P., 2015. Case studies synthesis: A thematic , cross-case , and narrative synthesis worked example. Empir. Softw. Eng. 20 (6), 1634–1665.

Data.world, DataKitchen, 2021. 2021 Data Engineering Survey Burned-Out Data Engineers Call for DataOps. Tech. rep..

Desai, V., Dinesha, H.A., 2020. A hybrid approach to data pre-processing methods. In: IEEE International Conference for Innovation in Technology. INOCON 2020, pp. 1–4.

Drocco, M., Misale, C., Tremblay, G., Aldinucci, M., 2017. A Formal Semantics for Data Analytics Pipelines. Tech. rep., pp. 1–24, arXiv:1705.01629.

Foidl, H., Felderer, M., 2019. Risk-based data validation in machine learning-based software systems. In: MaLTeSQuE 2019 - Proceedings of the 3rd ACM SIGSOFT International Workshop on Machine Learning Techniques for Software Quality Evaluation, co-located with ESEC/FSE 2019. pp. 13–18.

Foidl, H., Felderer, M., Ramler, R., Data smells: Categories , causes and consequences , and detection of suspicious data in AI-based systems. In: IEEE/ACM 1st International Conference on AI Engineering – Software Engineering for AI. (CAIN), pp. 229–239.

Fontão, A., Dias-Neto, A., Viana, D., 2017. Investigating factors that influence developers' experience in mobile software ecosystems. In: Proceedings - 2017 IEEE/ACM Joint 5th International Workshop on Software Engineering for Systems-of-Systems and 11th Workshop on Distributed Software Development, Software Ecosystems and Systems-of-Systems. JSOS 2017, (2), pp. 55–58.

Frye, M., Schmitt, R.H., 2020. Structured data preparation pipeline for machine learning-applications in production. In: 17th IMEKO TC 10 and EUROLAB Virtual Conference "Global Trends in Testing, Diagnostics & Inspection for 2030". pp. 241–246.

García, S., Ramírez-gallego, S., Luengo, J., Benítez, J.M., Herrera, F., 2016. Big data preprocessing: methods and prospects. Big Data Anal. 1 (1), 1–22.

Giovanelli, J., Bilalli, B., Abelló, A., 2022. Data pre-processing pipeline generation for AutoETL. Inf. Syst. (108), 101957.

Golendukhina, V., Lenarduzzi, V., Felderer, M., 2022. What is software quality for AI engineers? Towards a thinning of the fog. In: IEEE/ACM 1st International Conference on AI Engineering – Software Engineering for AI (CAIN), Vol. 1. (1), pp. 1–9.

Goodhope, K., Koshy, J., Kreps, J., 2012. Building linkedin's real-time activity data pipeline.. IEEE Data Eng. Bull. 35 (2), 1–13.

Grafberger, S., Munich, T.U., Stoyanovich, J., Schelter, S., 2021. Lightweight inspection of data preprocessing in native machine learning pipelines. In: Conference on Innovative Data Systems Research. (CIDR).

Hapke, H., Nelson, C., 2020. Building Machine Learning Pipelines, first ed. O'Reilly, Sebastopol, CA.

Haug, A., Arlbjø rn, J.S., Zachariassen, F., Schlichter, J., 2013. Master data quality barriers: An empirical investigation. Ind. Manag. Data Syst. 113 (2), 234–249.

Helu, M., Sprock, T., Hartenstine, D., Venketesh, R., Sobel, W., 2020. Scalable data pipeline architecture to support the industrial internet of things. CIRP Ann. 69 (1), 385–388.

Hlupić, T., Puniš, J., 2021. An overview of current trends in data ingestion and integration. In: 44th International Convention on Information, Communication and Electronic Technology. (MIPRO), pp. 1265–1270.

Hongjiang, 2015. What are the most important factors for accounting information quality and their impact on AIS data quality outcomes? J. Data Inf. Qual. 5 (4), 1–22.

Hove, S.E., Anda, B., 2005. Experiences from conducting semi-structured interviews in empirical software engineering research. In: Proceedings - International Software Metrics Symposium, Vol. 2005. (Metrics), pp. 10–23.

Ibrahim, A., Mohamed, I., Satar, N.S.M., 2021. Factors influencing master data quality: A systematic review. Int. J. Adv. Comput. Sci. Appl. 12 (2), 181–192.

IDC InfoBrief, 2020. Data as the new water: The importance of investing in data and analytics pipelines.

Islam, M.J., Nguyen, G., Pan, R., Rajan, H., 2019a. A comprehensive study on deep learning bug characteristics. In: Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. pp. 510–520.

Islam, M.J., Nguyen, H.A., Pan, R., Rajan, H., 2019b. What do developers ask about ML libraries? A large-scale study using stack overflow. arXiv:1906.11940.

Ismail, A., Truong, H.L., Kastner, W., 2019. Manufacturing process data analysis pipelines: a requirements analysis and survey. J. Big Data 6 (1), 1–26.

ISO/IEC, 2008. Iso/iec 25012:2008 software engineering - software product quality requirements and evaluation (square) - data quality model.

Kandel, S., Paepcke, A., Hellerstein, J.M., Heer, J., 2012. Enterprise data analysis and visualization: An interview study. IEEE Trans. Vis. Comput. Graphics 18 (12), 2917–2926.

Karkouch, A., Mousannif, H., Al Moatassime, H., Noel, T., 2016. Data quality in internet of things: A state-of-the-art survey. J. Netw. Comput. Appl. 73, 57–81.

Knauer, T., Nikiforow, N., Wagener, S., 2020. Determinants of information system quality and data quality in management accounting. J. Manag. Control 31 (1–2), 97–121.

Koivisto, T., 2019. Efficient data analysis pipeline. Data Sci. Natural Sci. Sem. 2–5.

Konstantinou, N., Paton, N.W., 2020. Feedback driven improvement of data preparation pipelines. Inf. Syst. 92, 101480.

Kuchnik, M., Klimovic, A., Simsa, J., Smith, V., Amvrosiadis, G., 2021. Plumber: Diagnosing and removing performance bottlenecks in machine learning data pipelines. In: Proceedings of Machine Learning and Systems. pp. 33–51.

Lenarduzzi, V., Lomio, F., Moreschini, S., Taibi, D., Tamburri, D.A., 2021. Software quality for AI: Where we are now? Lect. Notes Bus. Inf. Process. 404 (August), 43–53.

Lourenço, R., Freire, J., Shasha, D., 2020. BugDoc: A system for debugging computational pipelines. In: Proceedings of the ACM SIGMOD International Conference on Management of Data. pp. 2733–2736.

Malley, B., Ramazzotti, D., Wu, J.T.-y., 2016. Data pre-processing. In: Secondary Analysis of Electronic Health Records. Springer, pp. 115–141.

Martínez-fernández, S., Bogner, J., Franch, X., Oriol, M., Siebert, J., Trendowicz, A., Vollmer, A.M., Wagner, S., 2022. Software engineering for AI-based systems: A survey. ACM Trans. Softw. Eng. Methodol. (TOSEM) 31 (2), 1–59.

Munappy, A.R., Bosch, J., Holmstr, H., Wang, T.J., 2020a. Modelling data pipelines. In: 46th Euromicro Conference on Software Engineering and Advanced Applications. (SEAA), pp. 13–20.

Munappy, A.R., Bosch, J., Olsson, H.H., 2020b. Data pipeline management in practice: Challenges and opportunities. In: Product-Focused Software Process Improvement. PROFES 2020, In: Lecture Notes in Computer Science, vol. 12562, pp. 168–184.

Munappy, A.R., Bosch, J., Olsson, H.H., 2021. On the trade-off between robustness and complexity in data pipelines. In: International Conference on the Quality of Information and Communications Technology. pp. 401–415.

Munappy, A.R., Bosch, J., Olsson, H.H., Wang, T.J., 2020c. Towards automated detection of data pipeline faults. In: 27th Asia-Pacific Software Engineering Conference. (APSEC), pp. 346–355.

Nord, G.D., Nord, J.N., Xu, H., 2005. An investigation of the impact of organization size on data quality issues. J. Database Manag. 16 (3), 58–71.

O'Donovan, P., Leahy, K., Bruton, K., O'Sullivan, D.T., 2015. An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities. J. Big Data 2 (1), 1–26.

Oleghe, O., Salonitis, K., 2020. A framework for designing data pipelines for manufacturing systems. Procedia CIRP 93, 724–729.

Rahman, A., Farhana, E., 2021. An empirical study of bugs in COVID-19 software projects. J. Softw. Eng. Res. Dev. 9 (3).

Ray, B., Hellendoorn, V., Godhane, S., Tu, Z., Bacchelli, A., Devanbu, P., 2016. On the naturalness of buggy code. In: Proceedings - International Conference on Software Engineering, Vol. 14-22-May-. ACM, pp. 428–439.

Rezig, E.K., Brahmaroutu, A., Tatbul, N., Ouzzani, M., Tang, N., Mattson, T., Madden, S., Stonebraker, M., 2020. Debugging large-scale data science pipelines using dagger. Proc. VLDB Endow. 13 (12), 2993–2996.

Romero, O., Wrembel, R., Song, I.Y., 2020. An alternative view on data processing pipelines from the DOLAP 2019 perspective. Inf. Syst. 92, 101489.

Ronkainen, J., Iivari, A., 2015. Designing a data management pipeline for pervasive sensor communication systems. Procedia Comput. Sci. 56 (1), 183–188.

Rupprecht, L., Davis, J.C., Arnold, C., Gur, Y., Bhagwat, D., 2020. Improving reproducibility of data science pipelines through transparent provenance capture. Proc. VLDB Endow. 13 (12), 3354–3368.

Samantra, C., Datta, S., Mahapatra, S.S., 2016. Interpretive structural modelling of critical risk factors in software engineering project. Benchmarking: Int. J. 23 (1), 2–24.

Schäfer, D., Palm, B., Schmidt, L., Lünenschloß, P., Bumberger, J., 2020. From source to sink-sustainable and reproducible data pipelines with SaQC. In: EGU General Assembly Conference Abstracts. p. 19648.
Singh, R., Singh, K., 2010. A descriptive classification of causes of data quality problems in data warehousing. IJCSI Int. J. Comput. Sci. Issues 7 (2), 41.
Tardio, R., Mate, A., Trujillo, J., 2020. An iterative methodology for defining big data analytics architectures. IEEE Access 8, 210597–210616.
Tee, S.W., Bowen, P.L., Doyle, P., Rohde, F.H., 2007. Factors influencing organizations to improve data quality in their information systems. Account. Finance 47 (2), 335–355.
Theodorou, V., Abell, A., Lehner, W., 2014. Quality measures for ETL processes. In: International Conference on Data Warehousing and Knowledge Discovery. Springer, pp. 9–22.
Tiezzi, J., Tyler, R., Sharma, S., 2020. Lessons learned: A case study in creating a data pipeline using Twitter's API. In: Systems and Information Engineering Design Symposium, SIEDS 2020. IEEE, pp. 1–6.
Trendowicz, A., 2009. Factors influencing software development productivity-state-of-the-art and industrial experiences related papers. In: Advances in Computers, Vol. 77. pp. 185–241.
Usman, M., Britto, R., Börstler, J., Mendes, E., 2017. Taxonomies in software engineering: A systematic mapping study and a revised taxonomy development method. Inf. Softw. Technol. 85, 43–59.
Van Dongen, G., Van Den Poel, D., 2021. Influencing factors in the scalability of distributed stream processing jobs. IEEE Access 9, 109413–109431.
Von Landesberger, T., Fellner, D.W., Ruddle, R.A., 2017. Visualization system requirements for data processing pipeline design and optimization. IEEE Trans. Vis. Comput. Graphics 23 (8), 2028–2041.
Wang, Z., Chen, T.H.P., Zhang, H., Wang, S., 2022. An empirical study on the challenges that developers encounter when developing apache spark applications. J. Syst. Softw. 194, 111488.
Wang, R., Sun, D., Li, G., Wong, R., Chen, S., 2020. Pipeline provenance for cloud-based big data analytics. Softw. - Pract. Exp. 50 (5), 658–674.
Wohlin, C., 2014. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: Shepperd, M., Hall, T., Myrtveit, I. (Eds.), Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering - EASE '14. ACM Press, New York, New York, USA, pp. 1–10.
Xiao, J.H., Xie, K., Wan, X.W., 2009. Factors influencing enterprise to improve data quality in information systems application - an empirical research on 185 enterprises through field study. In: International Conference on Management Science and Engineering - 16th Annual Conference Proceedings, ICMSE. (1996), IEEE, pp. 23–33.
Xu, H., 2013. Factor analysis of critical success factors for data quality. In: 19th Americas Conference on Information Systems, AMCIS 2013 - Hyperconnected World: Anything, Anywhere, Anytime, Vol. 3. (August 2013), pp. 1679–1684.
Xu, H., Nord, J.H., Brown, N., Nord, G.D., 2002. Data quality issues in implementing an ERP. Ind. Manag. Data Syst. 102 (1), 47–58.
Yan, C., He, Y., 2020. Auto-suggest: Learning-to-recommend data preparation steps using data science notebooks. In: Proceedings of the ACM SIGMOD International Conference on Management of Data. pp. 1539–1554.
Yang, C., Zhou, S., Guo, J.L.C., Kästner, C., 2021. Subtle bugs everywhere: Generating documentation for data wrangling code. In: 36th IEEE/ACM International Conference on Automated Software Engineering. (ASE), pp. 304–316.
Zellal, N., Zaouia, A., 2016a. An exploratory investigation of factors influencing data quality in data warehouse. In: Proceedings of 2015 IEEE World Conference on Complex Systems. WCCS 2015, IEEE.
Zellal, N., Zaouia, A., 2016b. A measurement model for factors influencing data quality in data warehouse. Colloquium Inf. Sci. Technol., CIST 46–51.
Zellal, N., Zaouia, A., 2017. An examination of factors influencing the quality of data in a data warehouse. IJCSNS Int. J. Comput. Sci. Netw. Secur. 17 (8), 161–169.
Zhang, R., Xiao, W., Zhang, H., Liu, Y., Lin, H., Yang, M., 2020. An empirical study on program failures of deep learning jobs. In: Proceedings - International Conference on Software Engineering. pp. 1159–1170.
Zoto, E., Tole, D., 2014. The main factors that influence data quality in accounting information systems. Int. J. Sci., Innov. New Technol. 1 (1), 1–8.
Zwick, M., 2019. ML-PipeDebugger: A debugging tool. In: International Conference on Database and Expert Systems Applications. Springer International Publishing, pp. 263–272.

Harald Foidl is a Ph.D. student in computer science at the University of Innsbruck. He received a M.Sc. in information systems in 2015 from the University of Innsbruck. Besides his studies, Harald worked as a programmer and test engineer for an Electronics Manufacturing Services company. His research interests encompass the design, development, security, and testing of software and data-intensive systems, with a particular focus on guaranteeing their quality. In addition to his research activities, Harald is currently head of the Surface-Mount Technology Department of the company Kontron Austria.

Valentina Golendukhina is a Ph.D. student in computer science at the University of Innsbruck. She received a M.Sc. in information systems in 2021 from the University of Innsbruck. Valentina's interests include data engineering, data analytics, and empirical software engineering. In the course of her research, she closely cooperates with industrial partners in the realm of data-driven projects.

Rudolf Ramler is a research manager at Software Competence Center Hagenberg (SCCH), Austria. Rudolf holds a M.Sc. in Business Informatics from Johannes Kepler University Linz. He has more than 20 years of experience in applied research in the fields of software engineering, software quality assurance and testing, software analytics, and application lifecycle management. He is author of over 100 reviewed publications, co-organizer and chair of international conferences and workshops, an ISTQB certified tester, and an IEEE and ACM member. His mission and passion are to support industry in turning research results into practically successful solutions.

Michael Felderer is the Director of the Institute for Software Technology at the German Aerospace Center (DLR), a full professor at the University of Cologne, Germany and an associate professor at the University of Innsbruck, Austria. He holds a Ph.D. and a habilitation degree in computer science. His research interests include software quality, software architectures as well as software engineering and system engineering for AI, digital twins and quantum computing. His research is performed in close collaboration with industry. Prof. Felderer is an internationally well-recognized researcher in software and systems engineering and has published more than 200 papers.