OULUN AMMATTIKORKEAKOULU

Ernesta Petraitytė

Exploring Efficient Workflow Frameworks for Data Management

Optimising Data Management for Sponsoring Consortium for Open Access Publishing in Particle Physics

Exploring Efficient Workflow Frameworks for Data Management

Optimising Data Management for Sponsoring Consortium for Open Access Publishing in Particle Physics

Ernesta Petraitytė
Final projects
Spring 2024
Modern Software and Computing Solutions
Oulu University of Applied Sciences

ABSTRACT

Oulu University of Applied Sciences
MBA in Modern Software and Computing Solutions

Author(s): Ernesta Petraitytė
Title of the thesis: Exploring Efficient Workflow Frameworks for Data Management
Thesis examiner(s): Teemu Leppänen
Term and year of thesis completion: Spring 2024 Pages: e.g. 47 + 1 appendice

---

This thesis, titled "Exploring Efficient Workflow Frameworks for Data Management," focuses on improving data management strategies for the Sponsoring Consortium for Open Access Publishing in Particle Physics (SCOAP³). The objective is to evaluate how workflow management systems, particularly Apache Airflow, can enhance SCOAP³'s ability to manage a large volume of data effectively.

Structured in two main sections, the study first reviews relevant literature to set the theoretical groundwork for workflow management. It then conducts a comparative analysis of workflow management tools, with a detailed case study on Apache Airflow's application within the SCOAP³ project. The research methodology combines qualitative methods to assess the impact of these systems on data handling.

Findings from the case study indicate that integrating Apache Airflow leads to notable improvements in data workflow management, including enhanced follow-up on the state of data processing. These results suggest that workflow management systems play a critical role in streamlining data operations, especially in contexts dealing with extensive datasets.

The thesis concludes by linking these findings to existing literature and theories on workflow management. It confirms that while Apache Airflow enhances data management processes, its integration and optimization come with challenges. Future research is recommended to explore more adaptable workflow management solutions to further improve data management practices. This work provides a solid foundation for entities like SCOAP³ to leverage workflow management tools for more efficient data management.

Keywords: Workflow Management Systems, Apache Airflow, Data Management Optimization, Academic Publishing, SCOAP³ Project, Comparative Analysis

CONTENTS

1 INTRODUCTION 6

1.1 Importance of Data Collection, Processing, and Integration....6

1.2 Significance of Workflow Frameworks....7

1.3 Describing the Challenge and Setting Our Research Goals....9

1.4 Thesis Structure....9

2 LITERATURE REVIEW 12

2.1 Introduction to SCOAP³....12

2.2 Data Collection....12

2.3 Processing....17

2.4 Integration....18

2.5 Challenges Faced by Organizations....19

2.6 Reusable Workflow Frameworks....21

2.7 Identifying Knowledge Gaps....23

3 FRAMEWORKS COMPARISON AND AIRFLOW IMPLEMENTATION 23

3.1 Comparative Analysis of Features and Evaluation of Framework....23

3.1.1 Functional Capabilities Assessment....23

3.1.2 Feature and Performance Comparison....26

3.1.3 Contextual Suitability Analysis....27

3.2 Implementation Analysis....28

3.2.1 Airflow 2.2.4: Navigating Complexity in Workflow Management....29

3.2.2 Dagster 0.13.0: Streamlining Data Engineering for Beginners....29

3.2.2 Conclusion and Framework Selection for the Thesis Project....30

3.3 Implementation of Airflow in SCOAP³....30

3.3.1 Implementation Strategy....30

3.3.2 Initial Setup and Development Environment....31

3.3.3 Publishers' Analysis and Data Harvesting....32

3.3.4 Data Parsing and Transformation....33

3.3.5 Data Enhancement and Integration with External Sources..... 34
3.3.6 Validation and Compliance with Established Schema..... 34
3.3.7 Conclusion..... 35

4 RESULTS AND ANALYSIS 36
4.1 Data Quality Measures..... 36
4.2 Data Standardisation..... 37
4.3 Issues Identified and Resolved..... 38

5 CONCLUSION 41
5.1 Workflow Management Frameworks: Analysis and Selection Insights..... 40
5.2 Recommendations..... 41
5.3 Research Limitations..... 42

REFERENCES..... 44
APPENDICES..... 46

1 INTRODUCTION

1.1 Importance of Data Collection, Processing, and Integration

The digital revolution has profoundly transformed the landscape of research investigation, introducing advanced methodologies for the collection, processing, and integration of data. These methodologies are particularly vital in the context of the Sponsoring Consortium for Open Access Publishing in Particle Physics (SCOAP³) repository project, a pioneering initiative in the field of particle physics. Launched by CERN, the European Organization for Nuclear Research, it aims to convert the high-quality peer-reviewed journals in the field of high-energy physics to open access at no cost for authors. SCOAP³ is a unique partnership of libraries, funding agencies, and research centres in more than 40 countries. This section, "Importance of Data Collection, Processing, and Integration", delves into the critical role these processes play in underpinning the research integrity and innovation that SCOAP³ stands for.

Data collection serves as the cornerstone of the SCOAP³ project, ensuring that the vast array of scholarly articles and research findings within the repository are accurately represented and accessible. This step is fundamental, as the accuracy and comprehensiveness of the data directly influence the project's ability to facilitate open access to particle physics literature, thereby advancing the global scientific discourse.

Upon collection, the data undergoes meticulous processing to ensure it is not only analytically viable but also aligned with the project's standards for quality and accessibility. This involves refining the raw data into a structured format, making it possible to conduct meaningful analyses that can inform future directions for the SCOAP³ initiative and the broader scientific community.

Integration is pivotal in creating a cohesive repository that combines diverse data sources into a unified platform. This process allows for a more holistic view of the research landscape in particle physics, enhancing the project's capacity to support interdisciplinary research and collaboration. By synthesising data from various contributors, SCOAP³ fosters a more inclusive and comprehensive repository, bridging gaps in knowledge and promoting a synergistic approach to scientific exploration.

Together, these processes of data collection, processing, and integration are indispensable to the success and impact of the SCOAP³ repository project. They ensure that the repository not only serves as a testament to the collective efforts of the particle physics community but also as a beacon for open access and collaborative research. In the ensuing discussion, we will explore each of these processes in greater depth, elucidating their significance in the context of SCOAP³ and their contribution to the advancement of particle physics and beyond.

1.2 Significance of Workflow frameworks

Before discussing the significance of workflow frameworks, it's essential to grasp the concept of a workflow framework. A workflow framework is a critical tool for controlling, planning, and tracking the flow of tasks. Workflow modelling, a central aspect of this, encompasses several crucial elements: defining and selecting suitable tasks (which may be sourced from a task library), arranging these tasks in a sequence that satisfies data and logical dependencies, assigning the resources required for task execution, allocating agents to perform these tasks, scheduling tasks while considering concurrency, and, finally, confirming and validating the model (Madhusudan, Zhao, & Marshall, 2004, p. 2). Manual workflow modelling is typically supported by graphical interfaces, where the workflow model is represented as a graph.

Workflow modelling entails the implicit exploration of a design space that includes numerous alternative process models. The goal is to select the most optimal process model that effectively addresses the given problem. As the adoption of workflow management systems rises and flexible process integration technologies like web services become more prevalent, there is an urgent need to develop tools and methods that can assist in the design and modelling of workflows.

Workflow frameworks hold a crucial role in the field of data engineering, delivering notable advantages to professionals working with data. These frameworks are of significant importance as they help data engineers orchestrate and streamline the complex processes involved in data management, thereby contributing to the efficiency and reliability of data-related tasks.

One of the key values of workflow frameworks in data engineering is process efficiency. Data engineers deal with vast quantities of data that require collection, storage, processing, and transformation. Workflow frameworks provide a structured approach to execute these tasks, minimising manual interventions and reducing the chances of errors. As a result, data-related operations become more time-efficient and cost-effective.

Another noteworthy aspect is process standardisation. Workflow frameworks ensure that data engineering tasks are carried out consistently and according to predefined guidelines. This standardisation is essential for maintaining data quality, adhering to best practices, and ensuring data reliability.

Additionally, workflow frameworks offer enhanced visibility and transparency into data engineering processes. They allow data engineers, managers, and stakeholders to monitor the progress of data pipelines and identify any bottlenecks or areas in need of improvement. This transparency empowers data engineers to make informed decisions and optimise data workflows.

Moreover, workflow frameworks enable process monitoring and control. Many of these frameworks include monitoring and reporting capabilities, and ensure compliance with data management policies. This proactive approach to process management minimises risks and enhances data governance.

In a rapidly evolving field like data engineering, adaptability is crucial. Workflow frameworks offer flexibility, allowing data engineers to modify processes to meet changing business requirements, adapt to evolving technology, and address emerging data challenges.

In conclusion, the significance of workflow frameworks in data engineering is undeniable. They play a pivotal role in enhancing process efficiency, reducing errors, improving communication, ensuring transparency, and providing adaptability in the face of evolving data needs. These frameworks are invaluable tools for data engineers, enabling them to structure and manage data-related tasks in a manner that aligns with organisational goals and industry standards.

1.3 Describing the Challenge and Setting Our Research Goals

The primary challenge lies in selecting an appropriate workflow framework. The desired features encompass the framework's capacity to restart the pipeline run at any stage, ensure seamless traceability of each record within the workflow, provide clear and informative logs for an enhanced understanding of the processes, offer a transparent and easily navigable user interface, and enable the initiation of the workflow programmatically through an API. Additionally, considerations must be given to the framework's long-term maintenance ease and its compatibility with the prevailing technology stack.

Limited understanding of data collection, processing, and transformation also poses a significant challenge. Complicating matters, the data lacks uniformity; various sources offer diverse formats and structural arrangements. Nevertheless, the ultimate objective of our workflow is to generate a uniform and consistent output. This outcome will facilitate a clearer comprehension of the received data and the identification of any missing information.

This study has two main objectives. Firstly, to address the challenge of selecting an appropriate workflow framework, which can effectively meet our specific requirements. These requirements include the ability to restart the pipeline run at any point or from the beginning, ensure traceability of every record within the workflow, provide clear and informative logs for a better understanding of the processes, offer a user-friendly interface, and enable the programmable initiation of the workflow through an API. Additionally, we aim to implement and assess the chosen framework's performance in real production environments. Secondly, our research also seeks to investigate the long-term maintainability of the selected framework and its alignment with our existing technology stack, primarily centred around Python. By achieving these goals, we aim to contribute valuable insights and solutions to the field of data processing and workflow management while improving operational efficiency and sustainability.

1.4 Thesis Structure

The structure of this thesis is carefully designed to navigate the complexities of data collection, processing, and integration within the realm of workflow management frameworks, specifically focusing on their application in the SCOAP³ project. The thesis is divided into distinct chapters,

each serving a unique purpose in the exploration and analysis of the subject matter. Following is a brief overview of each chapter's focus and contributions:

Chapter 1: Introduction

Sets the stage for our research, underscoring the importance of data collection, processing, and integration. It presents the significance of workflow frameworks, articulating the challenges faced and the research goals set forth. This opening chapter lays a foundational understanding of the thesis's scope and objectives, culminating in an overview of the thesis structure itself, ensuring a clear guide for the reader.

Chapter 2: Literature Review

This chapter delves into a comprehensive review of the existing literature on data collection, processing, and integration techniques, alongside the exploration of reusable workflow frameworks. It highlights the challenges commonly faced by organisations in data workflow management and identifies existing knowledge gaps. The review sets a foundation for the research by situating it within the broader academic and practical discourse on data management.

Chapter 3: Frameworks Comparison and Airflow Implementation

Dedicated to a comparative analysis of Apache Airflow and Dagster, this chapter evaluates the frameworks based on their features, usability, and applicability to the SCOAP³ project's needs. Following the comparative analysis, the chapter details the implementation process of Apache Airflow within the SCOAP³ project, including the strategic considerations, challenges encountered, and the solutions devised to overcome these obstacles.

Chapter 4: Results and Analysis

Focusing on the outcomes of the research, this chapter presents the results related to data quality measures, standardisation efforts, and the resolution of specific issues identified during the project's execution. It evaluates the impact of the chosen workflow management framework on the project's data processing and standardisation practices, providing critical insights into the effectiveness of the methodologies employed.

Chapter 5: Conclusion

Concluding the thesis, this chapter synthesises the key insights and findings from the research. It offers a set of recommendations for future work, drawing on the lessons learned from the SCOAP³ project's experience. Additionally, it discusses the limitations of the research, highlighting areas for further exploration. This final chapter aims to contribute to the ongoing dialogue in data engineering and workflow management, offering perspectives that may inform future research and practice in the field.

2 LITERATURE REVIEW

2.1 Introduction to SCOAP³

SCOAP³ (Sponsoring Consortium for Open Access Publishing in Particle Physics) represents a transformative collaborative initiative aimed at fostering open access publishing within the domain of high-energy physics (HEP). Established in 2012, this consortium is a partnership comprising libraries, funding agencies, and research institutions with the shared mission of providing free, unrestricted access to HEP research articles globally. By transitioning the traditional subscription-based publishing model to a cooperative open access framework, SCOAP³ ensures that authors maintain copyright over their scholarly works, coupled with the extensive right of reuse. Importantly, the financial aspects of publishing under this model are collectively borne by the member institutions, which effectively democratises access to scientific knowledge in this specialised field.

In the context of this initiative, the author of this thesis is presently engaged in the developmental and migration aspects of the SCOAP³ Repository. This repository serves as a pivotal archive, hosting all the peer-reviewed articles funded through the SCOAP³ initiative. It presents a comprehensive, searchable collection of publications accessible to both the scientific community and the general public. Encompassing a diverse array of articles from participating journals, the repository is a testament to the wide-ranging research interests within the field of high-energy physics. The contribution to the development and migration of the SCOAP³ Repository is instrumental in ensuring that it continues to serve as an indispensable resource. By facilitating immediate access to pioneering research, the repository aids researchers, students, and enthusiasts, thus propelling the continual advancement of knowledge in particle physics. This effort aligns with the overarching objective of SCOAP³, which is to remove access barriers to scientific literature and herald a new era of open scholarship.

2.2 Data Collection

In the context of SCOAP³, data collection is essentially a process of transferring information between the source (provider) and the destination (SCOAP³). SCOAP³ manages data acquisition

from three distinct types of data resources: FTP, SFTP, and API. While the reasons behind selecting FTP, SFTP, and API as data sources remain unexplored, a comprehensive exploration of each source's concepts is essential for effective communication.

Presently, SCOAP³ exclusively harvests data from a single source utilising FTP services. The File Transfer Protocol (FTP) was originally developed by Abhay Bhushan and introduced in RFC 114 on April 16, 1971. It underwent a significant revision to accommodate TCP/IP protocols in June 1980 with RFC 765 and received further updates in October 1985 with RFC 959, which remains the standard in use today (Arsan, Günay, & Kaya, 2014, p. 34). RFC 959 has been supplemented by additional standards, notably RFC 2228 (Linn, 1997), which provides security enhancements, and RFC 2428, which adds support for IPv6 and introduces a new passive mode (Arsan, Günay, & Kaya, 2014, p. 34).

FTP, a standard network protocol, facilitates the transfer of computer files between hosts over a TCP-based network. Formulated to allocate files and improve remote computer usage, FTP operates on a client-server architecture, managing various control and data connections between the client and server.

There are two modes for establishing a data connection: active and passive, and FTP employs both modes. In active mode, the client initiates a TCP control connection from a non-specific port N to the FTP server's command port 21. During active modes, the client receives incoming data connections on port N+1 from the server. Port N+1 serves to notify the server when an FTP command is transmitted by the client. In passive mode, a scenario reveals where a client is situated behind a firewall, rendering it unavailable for incoming TCP connections. In this mode, the client dispatches a *PASV* command to the server through the control connection. Subsequently, the server communicates its IP address and port number to the client. This allows clients to establish a data connection from a random client port to the server's IP address and port number (Arsan, Günay, & Kaya, 2014, p. 35).

SCOAP³ uses passive mode due to its numerous advantages when compared to active mode. Active mode poses challenges when the client is situated behind a firewall that restricts incoming connections, requiring firewall configuration to permit the incoming data connection. Conversely, passive mode is commonly more firewall-friendly since the client establishes a connection to the

server's open port, mitigating problems associated with blocked incoming connections (Bellovin, 1994). Additionally, passive mode is generally perceived as more secure, as it allows the server to dictate the data connection port.

The server communicates using three-digit status codes in ASCII during the control connection. These codes are categorised into series such as 100 Series, 200 Series, 300 Series, 400 Series, 500 Series, etc. For instance, the code "200 OK" indicates the success of the last command. The numeric code signifies the response category, while optional text provides an explanation for human interpretation. It is worth noting that the ongoing transfer may experience interruptions during data transfer (Arsan, Günay, & Kaya, 2014, p. 36).

In the context of data representation, there are four modes available: ASCII Mode, EBCDIC Mode, Local Mode, and Image Mode, also known as Binary mode. We specifically apply Binary mode, where the sender transmits the file byte by byte, and the receiver stores it as a byte stream (Arsan, Günay, & Kaya, 2014 p. 36). Binary mode guarantees the file's transfer without character set conversions or line-ending modifications, preserving the exact byte-for-byte representation. This mode is crucial when handling archive files, as these files typically contain binary data, including a mixture of characters, control codes, and binary data, rather than plain text. Opting for ASCII mode with archives may lead to corrupted files due to character set conversions.

FTP operates within a client-server architecture, where clients transfer files to and receive files from a server. The FTP process involves two connections: one for transmitting standard FTP commands and responses, and the other for transferring the actual data (Arsan, Günay, & Kaya, 2014, p. 37).

Initiating the process is the Client Control Process, which establishes the Control Connection. Through this connection, FTP commands and responses are communicated, initiating the data connection as required. It is crucial to maintain the open status of the Control Connection throughout the data transfer period. In the event of a collision during data transfer, the FTP Data connection is closed, resulting in a failed session. Parameters such as transfer mode, file structure, and data representations are communicated via FTP commands. When operations and parameters are transmitted, the client refers to a predefined TCP port and server (Arsan, Günay, & Kaya, 2014, p. 37).

In summary, describing FTP passive mode involves a few straightforward steps: The client connects to the FTP server using command port number 21. The FTP server dispatches a message and a username query to the user, who provides the necessary connection information. The server verifies the sender's information and responds to the user. If the information is accurate, the FTP client awaits an additional port from the server through the *PASV* command. Subsequently, the FTP client establishes a connection to this port, initiating data transfer, and sends a confirmation message.

Currently, SCOAP³ collects information from three separate SFTP servers. SFTP, an acronym for Secure File Transfer Protocol. It is a method used to secure FTP, similar to alternatives such as FTPS. SFTP functions through a single, secure channel known as the Secure Shell (SSH) protocol, ensuring the secure transmission of files. Unlike traditional FTP, SFTP does not have active or passive modes. The SSH protocol encrypts both commands and data, adding an extra layer of security to prevent unauthorised access to sensitive information and passwords. It's important to note that SFTP operates independently from traditional FTP software and cannot operate concurrently with FTP. In contrast to FTPS, SFTP does not require separate port configurations, as it is seamlessly integrated into the SSH protocol (Arsan, Günay, & Kaya, 2014. p. 39).

Establishing a secure SFTP connection involves a series of steps within the Secure Shell (SSH) protocol, ensuring the protected transfer of files. The process begins with the client providing necessary details, like the server's address. Authentication follows, requiring user credentials or more advanced methods such as SSH keys. The subsequent encryption of commands and data adds a layer of security during transmission. Users can then use SFTP commands to navigate files and securely transfer data. The encrypted data connection is initiated by the SFTP client, safeguarding the confidentiality of transferred files. Throughout the process, the server communicates status messages, providing insights into command success. After completing file operations, the SFTP client gracefully terminates the connection, ensuring proper resource release. Particularly, SFTP operates as part of SSH, usually utilising port 22. These steps offer a comprehensive overview for secure and efficient data management in the context of a master's thesis.

Moreover, SCOAP³ gathers data from two APIs sources. The integration of Web APIs has become increasingly central to software development, as evidenced by the growing documentation and utilisation reported by key platforms like ProgrammableWeb and RapidAPI. This trend reflects the expanding API economy and underscores APIs' vital role in refining the efficiency and productivity of development processes. APIs have proven indispensable across a vast range of applications, from web and mobile platforms to desktop environments, facilitating data manipulation, fostering new business models, and enhancing product systems (Nordic APIs, 2023; RapidAPI, 2023).

APIs markedly enhance software reusability, thereby diminishing the time and resources required for development. By providing ready-to-use functionalities, they allow developers to pivot towards more innovative endeavours. The adoption of microservices architecture, leveraging APIs for communication among application components, underscores the benefits of scalable and flexible development. This model not only promotes rapid development but also ensures the resilience of the system by isolating failures to individual components (Nulab, n.d).

Furthermore, APIs are crucial for facilitating interoperability among diverse software systems, establishing a foundational platform for seamless information exchange. This capability is essential for promoting service exchange, enhancing organisational productivity, and ensuring data security across networks. The strategic use of reusable APIs extends these advantages, reducing the necessity for repetitive development work and thus streamlining project timelines and minimising security risks associated with API management (PlektonLabs, 2021).

A practical demonstration of APIs' effectiveness can be seen in the Academic Publication System (APS), which exemplifies how APIs facilitate efficient and secure data access, enhancing communication and collaboration. Looking ahead, SCOAP³'s initiatives to advocate for a shift from traditional file transfer protocols like FTP and SFTP to API-based data provision highlight the manifold benefits of this transition. These include improved security, through mechanisms like API keys or OAuth tokens, increased flexibility for customizable queries, and reduced manual intervention, thanks to the automation capabilities inherent in APIs. Well-documented APIs also expedite development, enabling rapid system integration and deployment.

2.3 Processing

Data processing transpires during the collection and transformation of raw data into actionable information. Typically carried out by a data scientist or a team of data scientists, precision in data processing is crucial to avoid adverse impacts on the final product or data output.

Initiating with raw data, the data processing journey involves converting it into a comprehensible format, such as graphs or documents. This transformation imparts the data with the structure and context essential for computer interpretation and subsequent utilisation by employees across an organisation (Talend, n.d.).

Looking from the perspective of SCOAP³, in the final stages of our data processing, our aim is to establish a standardised and intelligible data format for easy comprehension, analysis, and storage. As an output format, JSON was already chosen to be used in legacy SCOAP³, since it has numerous advantages. It has emerged as a favoured data format among developers due to its human-readable text, lightweight structure, minimised coding demands, and swift processing without the need for deserialization. Given the diversity of data formats we receive—such as XML, marcXMLa, and JSON—during processing, the adoption of a uniform format is essential. Primarily, the conversion to and interpretation of JSON is streamlined, particularly beneficial for our project's requirements, as Python is a fundamental component of our project's stack.
However, dealing with different input formats adds complexity to the process. Each format requires a specific approach, using different parsers for handling. Even when sources send data in the same format, like XML, the values in the fields can vary. The way to extract the same information differs, but it remains consistent within each source. Adding to the challenge, some XMLs with article metadata may have been edited by humans, introducing the possibility of errors.

Occasionally, essential fields are missing. It's during the last step of data processing, validation, that we realise the output does not meet the required standards. Consequently, we have to reach out to the data providers, asking them to update the information. This leads to the need for reprocessing the data.

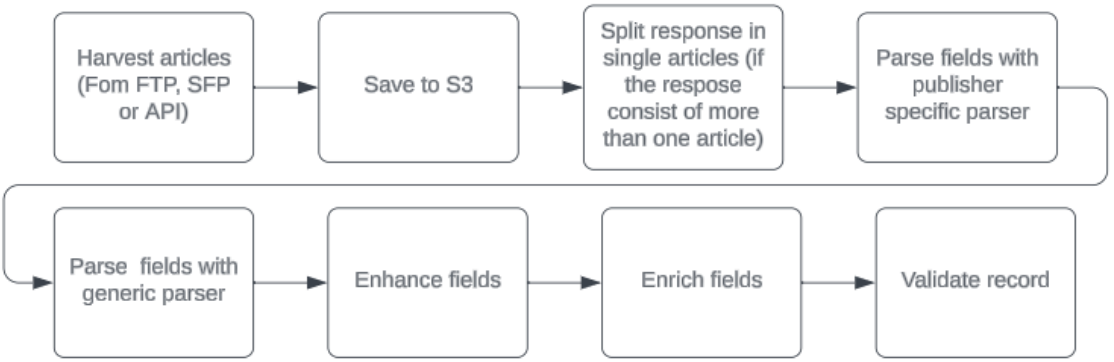

FIGURE 1. Data flow from data collection to validation.

A critical point is ensuring that if the metadata for one article does not pass validation, it should not affect the processing of other articles harvested at the same time. The goal is to allow those unaffected articles to smoothly proceed to the next stages in the workflow.

The processing steps encompass more than mere data format conversion; they involve the creation of new fields, the organisation of information into groups, and the enhancement of data from external sources. As the processing culminates, we arrive at a final output containing meticulously constructed information. This output provides us with an environment conducive to tracking the volume of received articles and delving into deeper insights, such as identifying contributing countries and discerning the publication locations of the articles, among other valuable aspects. In essence, the final output serves as a gateway to a wealth of useful information, facilitating comprehensive exploration and understanding.

2.4 Integration

To optimise our workflow system's efficiency and functionality, seamless integration with continuous integration and continuous deployment (CI/CD) tools is imperative. The use of CI/CD tools, such as GitHub CI/CD, is particularly advantageous due to the incorporation of automated testing, ensuring software quality and security. This automated testing significantly contributes to the continuous delivery process, enhancing the robustness and security of the software while improving the profitability of the code in production (Synopsys, n.d.).

This integration is especially beneficial considering SCOAP³'s backend is built on Django. Ensuring seamless integration with this framework creates a cohesive and interconnected environment, allowing for the smooth exchange of information and processes between the workflow system and the Django backend, thereby optimising overall functionality.

Additionally, the storage of both raw and processed data in AWS S3 buckets introduces another layer of integration requirements. Integrating with AWS services ensures secure and efficient data storage, aligning with best practices for cloud-based data management. This integration enhances the scalability and reliability of the overall system, facilitating accessibility and retrieval as needed throughout the workflow (Amazon Web Services, n.d.).

Furthermore, the database where the output of the workflow is saved is Postgres. This choice of database complements the system's requirements for robust data management. Postgres offers reliability, scalability, and advanced features that align with the demands of our workflow, ensuring the efficient storage and retrieval of processed data.

The integration with CI/CD tools not only guarantees software quality and security but also significantly reduces the time to market for new product features. These CI/CD pipelines foster customer satisfaction, alleviate strain on development teams, and contribute to an overall more responsive development cycle (Synopsys, n.d.). The accelerated delivery of updates and features translates to happier customers and a strategic edge for the organisation in the competitive landscape. The seamless integration of CI/CD tools, automated testing, and efficient deployment workflows, combined with Postgres as the database, creates a robust and agile environment, elevating the project's effectiveness and ensuring its relevance in a dynamic operational context.

2.5 Challenges Faced by Organizations

In today's digital age, effective data workflow management stands as a cornerstone of operational excellence within IT organisations. The ability to accurately process, integrate, and manage data not only fuels informed decision-making but also catalyses innovation and efficiency. However, this domain is fraught with challenges ranging from data quality and integration issues to managing unforeseen costs. This section provides an in-depth research of these challenges, drawing upon recent literature to underscore their impact and propose actionable solutions.

1. **Data Extraction and Integrity Challenges**: Organisations face significant challenges in extracting data from diverse sources with varying formats, structures, and types. Ensuring data integrity becomes a complex task due to poor data quality, which can affect the entire integration cycle and lead to faulty analytics. A solution involves creating a data quality management plan and selecting integration tools that support structured, unstructured, and semi-structured sources to streamline the extraction process (Surani, 2020).
2. **Data Security Concerns**: With the increasing volume of data collected, organisations encounter heightened risks of security breaches and cyber-attacks. The involvement of multiple stakeholders further introduces insider threats to data security and privacy. To mitigate these risks, it is crucial to implement real-time monitoring, advanced data masking techniques, and ensure compliance with data protection regulations (Perumal, 2023).
3. **Handling Large Data Volumes**: As companies create and gather more data, processing and integrating large volumes become complex, leading to longer processing times and increased risk of errors. Modern data management platforms and incremental data loading strategies are recommended to manage large data volumes effectively, ensuring efficient data integration without data loss or corruption (Richman, 2023).
4. **Infrastructure Management for Data Integration**: Ensuring a strong and dependable infrastructure for data integration is essential for smooth data extraction, transformation, and loading. Organisations need to plan for system outages, network disruptions, or hardware failures that can disrupt integration. Comprehensive ecosystem verification and selecting robust data integration solutions are critical for effective infrastructure management (Richman, 2023).
5. **Managing Unforeseen Costs:** the complexity of data integration can lead to unexpected situations, raising data integration costs. These unforeseen costs can burden the budget and resources. Implementing contingency planning and regular monitoring of systems can help manage unexpected costs effectively, ensuring that data integration remains cost-effective (Richman, 2023).

Organisations face numerous challenges in managing data, including extracting data from varied sources and ensuring its integrity, which is complicated by poor data quality. To address this,

developing a data quality management plan and using integration tools that handle different data types is essential. Additionally, the increasing volume of data raises security risks and the complexity of processing large datasets, necessitating advanced security measures, real-time monitoring, and efficient data management platforms. Infrastructure reliability is also crucial for seamless data integration, requiring planning against system disruptions and robust solutions. Lastly, unforeseen costs of data integration can strain resources, highlighting the need for contingency planning and regular system monitoring to keep data integration efforts cost-effective.

2.6 Reusable Workflow Frameworks

Reusable workflow frameworks in data engineering are an essential aspect of modern data management and processing. They offer a structured, efficient, and scalable way to automate and monitor the flow of data between different systems and processes. At their core, these frameworks help manage task dependencies, scheduling, execution, and error handling, allowing data engineers to focus on insights and analytics rather than the intricacies of the underlying infrastructure.

One of the primary benefits of reusable workflow frameworks is their ability to standardise and streamline the development of data pipelines. By adopting these frameworks, organisations enforce coding standards, reduce the likelihood of errors, and decrease the development and deployment time of new pipelines. This standardisation is crucial not only for maintaining consistency but also for facilitating easier updates and maintenance. The modular nature of these systems allows individual components to be updated without affecting the entire pipeline, leading to more robust and adaptable data infrastructure.

Building on this foundation, the practice of reusing workflows significantly avoids duplication of effort and resources. Organisations can rapidly create and deploy new pipelines by building upon the existing, tested, and proven workflows, thereby accelerating development time and promoting best practices. This reuse of workflows encourages the establishment of a central library of standardised components, which can be centrally maintained and updated. Such a library becomes a valuable asset, ensuring all workflows are consistent, up-to-date, and adhere to the best standards and practices. This approach is supported and facilitated by platforms like GitHub,

which provide extensive resources for sharing, reusing, and managing workflows effectively (GitHub, n.d.).

The proliferation of these frameworks can be attributed to the diverse needs and environments in data engineering. For instance, Kubeflow and Dagster are more oriented towards machine learning operations, providing tools and integrations specifically designed for developing, deploying, and managing machine learning workflows. Kubeflow utilises Kubernetes to enable scalable and distributed training of machine learning models, while Dagster focuses on the complexity of data dependencies in ML workflows. It is an orchestrator built for the creation and upkeep of data assets, including tables, datasets, machine learning models, and reports (Dagster, n.d.). On the other hand, frameworks like Apache Airflow and Luigi are often used for general-purpose data orchestration, suitable for a wide range of data tasks from simple batch jobs to complex data pipelines involving multiple data sources and transformations.

The choice of a particular workflow framework often depends on the specific needs of the project, such as the complexity of the workflows, the scale of data processing, the integration with other tools and systems, and the specific domain, such as batch processing, stream processing, or machine learning.

In conclusion, the variety of reusable workflow frameworks in data engineering reflects the field's complexity and the diverse requirements of different data processing scenarios. These frameworks bring numerous benefits, including efficiency, standardisation, and a focus on higher-level tasks. As data continues to grow in volume and importance, the role of these workflow systems is becoming increasingly critical in enabling scalable, reliable, and efficient data operations.

2.7 Identifying Knowledge Gaps

Prior to the commencement of this project, the thesis author’s familiarity with workflow management tools was nonexistent. A comprehensive investigation into workflow management and orchestration tools was imperative, necessitating a deep dive into their various types, optimal utilisation strategies, and the specific project contexts they best serve. Concurrently, the understanding of the data requisites—particularly the aspects related to its harvesting and

processing—was markedly limited. This necessitated a thorough exploration of the diverse data sources, the methodologies for effective data processing, and the description of the expected results from these efforts.

Understanding the operational and strategic aspects of how articles are uploaded, how timing varies among different publishers, and addressing common issues associated with each publisher were part of the learning curve. Moreover, comprehending the reports and analytics was critical. For instance, writing code to generate reports on country shares—detailing the contributions of authors from different countries for SCOAP³—demanded not just technical know-how but a nuanced understanding of the underlying business logic. This included recognizing the significance of these metrics in the broader context of SCOAP³'s mission and objectives. It was essential to understand the implications of these reports to spot anomalies or errors that might arise from coding mishaps or during the data parsing process.

This extensive learning process illuminated the need for a comprehensive strategy to handle the nuanced requirements of SCOAP³'s operations. It was an opportunity to refine and optimise the data gathering, taking into consideration the specific needs and challenges associated with managing such a diverse and dynamic collection of academic content. As I navigated through these complexities, the project evolved from a simple technology shift to an elaborate endeavour enhancing the efficiency, scalability, and reliability of the SCOAP³ repository system. This foundational inquiry sets the stage for a nuanced understanding and application of workflow management tools in effectively handling and analysing data, thereby propelling the project toward a successful transition and implementation. Understanding the intricacies of both the technological and business aspects of the system was crucial for a holistic approach to the project, ensuring not only a successful technical implementation but also alignment with the strategic vision and operational efficiency of the SCOAP³ initiative.

3 FRAMEWORKS COMPARISON AND AIRFLOW IMPLEMENTATION

3.1 Comparative Analysis of Features and Evaluation of Framework

In the rapidly evolving field of data engineering, effective management and orchestration of workflows are important for operational efficiency and success. Among the variety of tools available, workflow frameworks such as Dagster and Airflow have turned into solutions, each offering unique capabilities and features. Apache Airflow and Dagster are open-source platforms built for managing and scheduling data workflows. These platforms enable data engineers to create intricate pipelines, monitor their progress, and handle task dependencies. This section aims to provide a comparative analysis of two significant frameworks: Dagster 0.13.0 and Airflow 2.2.4. The objectives of this comparison are multifaceted and are outlined as follows:

1. Functional Capabilities Assessment.
2. Feature and Performance Comparison.
3. Contextual Suitability Analysis.

3.1.1 Functional Capabilities Assessment

The functional capabilities of both frameworks are undeniable: both frameworks are able to take data and process it, convert it to needed format.

Apache Airflow, employing Directed Acyclic Graphs (DAGs) and tasks, represents a sophisticated approach to workflow orchestration. Within its framework, DAGs serve as structured models that map out collections of tasks, where each task forms an integral part of the overall workflow. The strategic arrangement of tasks within a DAG is critical, as it outlines their relationships and dependencies, all managed through Python code. This meticulous arrangement enables precise control over the sequencing and scheduling of tasks, highlighting Airflow's emphasis on managing workflows rather than executing direct data processing. For example, in a workflow comprising tasks A, B, and C, Airflow allows for the specification that task C depends on the completion of task B, which in turn requires task A to finish first (Apache Airflow, n.d.). This

capability ensures that tasks are executed in a defined order and at the appropriate time, illustrating the platform's strength in complex pipeline development.

This orchestration-centric philosophy of Airflow is augmented by several key features that enhance its functionality and user experience (Solanki, 2023):

1. **Task-based Workflow Definition**: Airflow enables data engineers to define workflows as tasks within a DAG, offering granular control over the execution sequence and dependencies. This task-based approach allows for the modular design of workflows, making it easier to develop, understand, and maintain complex pipelines.
2. **Dynamic Task Generation**: Through the use of Python code, Airflow supports the dynamic generation of tasks. This flexibility allows workflows to adapt to varying data or operational requirements dynamically, enhancing the framework's applicability to a wide range of scenarios.
3. **Built-in Operators for Common Tasks**: Airflow comes equipped with a variety of operators, such as the PythonOperator and BashOperator, facilitating the execution of common task types without the need for custom code. These operators simplify the process of integrating different tasks into workflows, from running scripts to performing data transformations.
4. **Large Community and Ecosystem of Plugins and Integrations**: The extensive community around Airflow has contributed to a rich ecosystem of plugins and integrations, extending its functionality and enabling seamless integration with various data sources, tools, and services. This ecosystem not only enhances Airflow's versatility but also supports users in customising and extending their workflows to meet specific project needs.
5. **Web-based User Interface for Monitoring and Managing Workflows**: Airflow's web UI provides a comprehensive platform for monitoring and managing workflows. This interface allows users to visually track the progress of tasks, inspect logs, and manage the operational aspects of their pipelines, contributing to Airflow's appeal for managing complex data workflows.

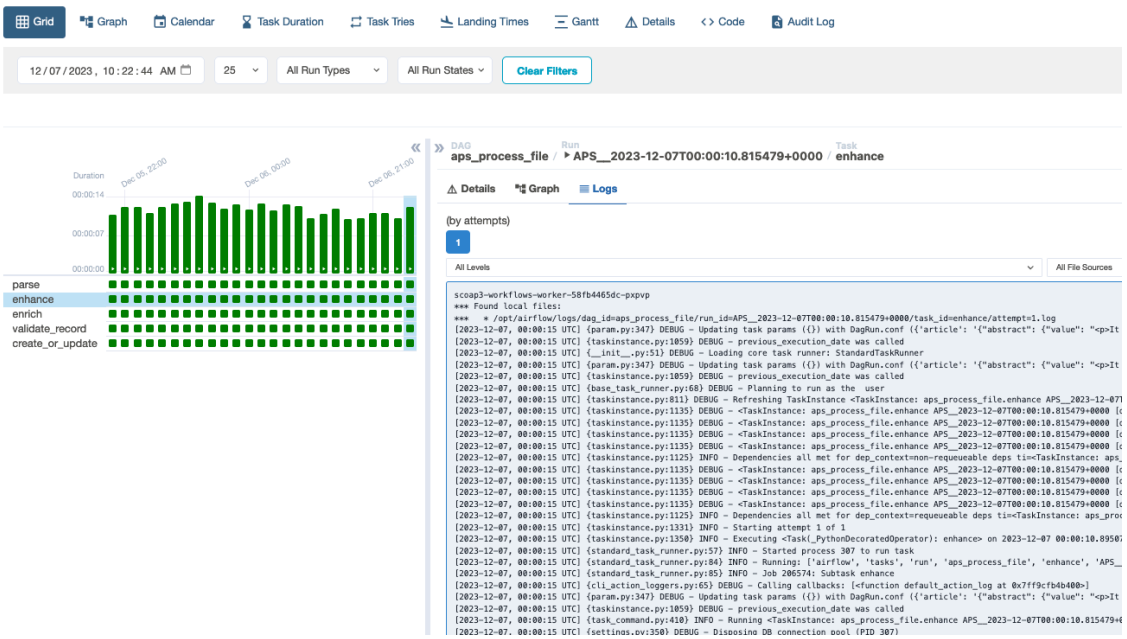

FIGURE 2. Apache Airflow user interface. Logs of APS file processing DAG.

Despite its strengths in task orchestration and workflow management, Airflow's design is not inherently geared towards handling intensive data processing within tasks. As Ansam Yousry (2023) insightfully points out in her article, a common misconception about Airflow is its capability as a data processing tool. While it is capable of executing Python code and performing data-related tasks, Airflow is not designed to handle substantial data processing tasks within its DAGs. Its primary role is to ensure that tasks are executed in the correct sequence and at designated times, rather than performing heavy computational tasks. For projects that require significant data manipulation or processing, this characteristic may necessitate the integration of Airflow with other data processing tools or systems.

In essence, Airflow stands out as a powerful tool for projects that prioritise a structured, reliable flow of tasks, particularly where the complexity lies in task orchestration rather than in the computational intensity of data processing. Its architectural design, complemented by features such as dynamic task generation, built-in operators, and a user-friendly web interface, positions Airflow as a preferred choice for workflows requiring meticulous task management and scheduling.

Shifting from Airflow, we turn to Dagster, a newer entry in data engineering and machine learning, known for its distinct approach to workflow orchestration. Dagster focuses on the fine-grained aspects of data processing within tasks (Restack, n.d.), organising workflows into pipelines with multiple interdependent steps. It uses 'solids' as the basic computational units, each encapsulating a piece of business logic. This design promotes the reusability and maintenance of data pipelines.

Solanki, Jatini (2023) in his article analysed several features of Dagster that enhance its data processing capabilities:

1. **Type-checked, Composable Pipeline Definitions**: Dagster ensures data integrity through type-checked pipeline definitions, which verify the type of data passed between tasks. This feature aids in composing reliable pipelines by enforcing data type consistency across 'solids'.
2. **Automatic Tracking of Dependencies Between Tasks**: The framework automatically manages task dependencies, simplifying workflow orchestration. This feature ensures tasks are executed in the correct sequence based on their dependencies.
3. **Built-in Data Validation and Error Handling:** Dagster includes mechanisms for runtime data validation and error handling to enhance the reliability of data workflows. This approach ensures pipelines are more fault-tolerant by catching and managing errors as they occur.
4. **Integration with ML Frameworks**: Dagster supports integration with machine learning frameworks such as TensorFlow and PyTorch, facilitating the development and management of ML pipelines. This makes it suitable for projects that involve complex data science tasks.
5. **Emphasis on Testing and Reproducibility**: The framework emphasises the creation of testable and reproducible pipelines, offering tools that help ensure consistent execution across different environments. This is crucial for maintaining the quality of data processing.

3.1.2 Feature and Performance Comparison

In assessing the performance and features of Apache Airflow and Dagster, according Solanki, Jatini (2023) ,several key differences merit consideration:

1. **Task-based vs. Pipeline-based Design**: Airflow adopts a task-based approach, requiring the definition of individual tasks and their dependencies. Dagster, conversely, utilises a pipeline-based model, treating the pipeline as a cohesive unit with tasks embedded within. This distinction can simplify dependency management in complex workflows, offering a streamlined approach in Dagster as opposed to the more granular control in Airflow.
2. **Dynamic Task Generation:** Airflow supports the dynamic creation of tasks based on data or external factors, enhancing flexibility in workflow design. Dagster lacks this capability, which might restrict its application in scenarios that demand such dynamism.
3. **Error Handling and Data Validation**: Dagster excels with its built-in mechanisms for data validation and error handling, features particularly beneficial in data-heavy workflows. While Airflow does incorporate error handling for tasks, it lacks native support for data validation, potentially necessitating additional tools or custom implementations for comprehensive data integrity checks.
4. **Integration with Machine Learning Frameworks**: Dagster offers direct integration with machine learning frameworks such as TensorFlow and PyTorch, facilitating the development of ML pipelines. Although Airflow doesn't provide built-in ML framework integration, it does offer connectivity with a variety of data processing tools like Spark and Hadoop, underscoring its versatility.

In summary, both platforms present unique advantages and limitations. Airflow, with its maturity, broad community, and extensive ecosystem, stands out for general workflow orchestration. Dagster, on the other hand, introduces innovative features tailored for data-intensive workflows, making it an attractive option for projects prioritising data validation, error handling, and machine learning integration. The selection between Airflow and Dagster should thus be informed by the specific demands of the project, including the complexity of workflows, the need for dynamic task generation, and the emphasis on data integrity and machine learning capabilities.

3.1.3 Contextual Suitability Analysis

This section aims to methodically compare two prominent frameworks in this data engineering field: Dagster 0.13.0 and Airflow 2.2.4. Our focus is directed towards an in-depth evaluation of requirements capabilities, particularly their proficiency in dynamically managing tasks, their

resilience in task restarts, and the intuitiveness of their user interfaces. These functionalities are not just technical features; they are foundational to the robustness and reliability of data workflow management, significantly influencing the risk of data loss and operational transparency.

1. **Dynamic Branching at Runtime**
    a. **Dagster**: Offers dynamic task branching adaptable to real-time data, beneficial in varied data scenarios. Its intuitive developer experience and interactive Dagit UI contribute to a more user-friendly approach to pipeline construction and monitoring.
    b. **Airflow**: Facilitates dynamic branching but requires predefined workflow structures. Its strength in dynamic workflow creation and extensive visualisation tools are advantageous for managing complex, conditional workflows.
2. **Restarting Tasks**
    a. **Dagster**: Supports task restarts but necessitates manual data management. Dagster's emphasis on functional modularity and pipeline versioning assists in maintaining data integrity and managing changes in workflows.
    b. **Airflow**: Excels with its automatic task restart mechanism, aided by XCom for efficient data sharing between tasks. This feature, combined with its robust task management, ensures continuity and reduces the risk of data loss.
3. **Understandable User Interface**
    a. **Dagster**: The Dagit UI is highly interactive, offering real-time insights into pipeline execution. Its design focuses on simplifying the user experience, especially for developers.
    b. **Airflow**: Boasts an enhanced, user-friendly interface in its 2.x versions, facilitating workflow visualisation and operational monitoring, essential for large-scale data workflows.

3.2 Implementation Analysis

This section provides a theoretical analysis of implementing Dagster 0.13.0 and Airflow 2.2.4. The focus is on examining the theoretical aspects of each framework, including setup, scalability, and maintenance, and understanding their operational dynamics in hypothetical scenarios.

3.2.1 Airflow 2.2.4: Navigating Complexity in Workflow Management

Airflow, renowned for its robustness in managing complex data workflows, presents a detailed architecture that demands a thorough understanding. The framework's setup, while comprehensive, can be challenging, especially for beginners. It requires a solid grasp of Python programming (AltexSoft, 2022) and an in-depth knowledge of Airflow's unique components, such as its executors, schedulers, and workers. These elements are pivotal for orchestrating and managing workflows but introduce a significant learning curve.

Scalability is one of Airflow's key strengths. Its design supports scaling up to handle large volumes of tasks and data. However, the maintenance of such scalable systems can be complex, necessitating continuous monitoring and optimization of the various components. This complexity, while manageable for experienced engineers, can be a barrier for new users, requiring a substantial investment in time and resources for mastery.

3.2.2 Dagster 0.13.0: Streamlining Data Engineering for Beginners

Dagster 0.13.0, in contrast, offers a more beginner-friendly approach to workflow management. The framework emphasises simplifying the data engineering process, focusing on features like software-defined assets and an enhanced local development experience. These aspects make Dagster particularly appealing for those new to the field, as it lowers entry barriers and facilitates a smoother learning curve.

From a theoretical standpoint, Dagster's setup is more straightforward compared to Airflow, focusing on ease of use without compromising on functionality. In terms of scalability, Dagster provides adequate capabilities to manage growing data workflows, though it may not match the extensive scalability of Airflow in larger and more complex environments. Maintenance with Dagster is more accessible due to its intuitive design and clearer abstraction layers, making it a suitable choice for teams with varying skill levels.

In comparing these two frameworks, it becomes evident that each has its distinct advantages. Airflow 2.2.4 offers a robust framework for managing complex data workflows, emphasising the need for a solid understanding of Python and familiarity with its components like executors and

schedulers. Despite its scalability advantages, Airflow's complexity and maintenance requirements present challenges, especially for beginners. In contrast, Dagster 0.13.0 targets a more beginner-friendly experience in workflow management, simplifying data engineering with features that lower entry barriers and offer a smoother learning curve. While Dagster is easier to maintain and suitable for various skill levels, it may not reach the scalability heights of Airflow in complex scenarios.

3.2.3 Conclusion and Framework Selection for the Thesis Project

The choice between Dagster and Airflow is influenced by the specific requirements of the project. Airflow's capabilities in managing scalable and dynamic workflows, coupled with its robust community support and feature set, make it the more suitable choice for the project's needs. Despite the initial setup complexity, its task management proficiency, and the flexibility offered by XCom for data sharing align well with the project's emphasis on operational transparency and ease of task follow-up. Therefore, Airflow is selected for its comprehensive functionality and proven effectiveness in diverse operational settings, particularly fitting for the project's requirements.

3.3 Implementation of Airflow in SCOAP³

In the SCOAP³ repository project, Airflow will primarily be tasked with overseeing the critical processes of data harvesting and processing. The fundamental purpose of the repository is to maintain records and track articles published under open access by SCOAP³ members. Airflow will play a pivotal role in gathering articles from renowned publishers like Springer, the Institute of Physics (IOP), Hindawi, Oxford University Press (OUP), American Physical Society (APS), and Elsevier.

3.3.1 Implementation Strategy

Following the introduction to the role of Airflow in the SCOAP³ repository project, this section outlines the strategic plan for its implementation. This strategy is vital for ensuring the efficient

management and processing of data, specifically in transforming diverse file formats into a standardised JSON output.

3.3.2 Initial Setup and Development Environment

The environment is pivotal in ensuring a smooth transition into the core tasks of data processing and workflow management. To achieve this, the official Airflow Docker image will be utilised, offering a pre-configured and efficient start to the development process. The use of this Docker image is a deliberate choice, aimed at standardising the development environment to mirror the production setup closely. This alignment is crucial to minimise discrepancies and potential issues between development and production environments.

However, recognizing the limitations and challenges posed by a full Docker-based setup, especially considering the hardware specifications of the MacBook Pro (Apple M1, 2020, 16 GB, MacOS 13.1), a more hybrid approach is adopted for local development. This approach combines the use of a Makefile with Docker Compose. The Makefile is strategically employed to streamline and automate routine tasks, thus enhancing the overall efficiency of the development process. Concurrently, Docker Compose is utilised to orchestrate Airflow processes and other essential services for development and testing such as Redis, Postgres, Minio, FTP and SFTP. This modular approach to service management enables a more controlled and manageable environment.

The decision to adopt this mixed approach stems from the need to address the resource-intensive nature of Docker when running the complete setup. By partitioning tasks between the Makefile and Docker Compose, the setup is optimised for better performance and reduced load on the system. This optimization is crucial in ensuring smoother and quicker development cycles, enhancing the local development experience. Special attention is also devoted to optimising Docker configurations and resource allocations. These optimizations are targeted at mitigating slow performance and high resource consumption, key considerations given the hardware constraints.

3.3.3 Publishers' Analysis and Data Harvesting

In our SCOAP³ repository project, we will approach each publisher – including APS, Springer, IOP, Hindawi, OUP, and Elsevier – as a unique entity. This means we'll carefully study and understand the specific data formats they use, which vary from XML and MarcXML to JSON. It is essential to grasp the details of these formats because they directly influence how they will be processed.

In the harvesting step, which is the first step of our workflow for the SCOAP³ project, the focus is on gathering data from different publishers, each with their unique way of providing information.

1. **Hindawi**: the content is accessed through an API, receiving data in the MarcXML format. This method allows directly connect with Hindawi's system and download the latest publications.
2. **APS (American Physical Society):** For APS, an API is also used but the data comes in JSON format, which is known for its user-friendly structure, making it easier to handle.
3. **OUP (Oxford University Press)**: With OUP, the data is downloaded via FTP, and it is in XML format, usually in zip files. This means these zipped files have to be unpacked to get to the individual documents.
4. **IOP (Institute of Physics):** IOP's data is similar to OUP's, provided in XML format via SFTP and packed in zip files. The SFTP ensures a secure transfer of these files.
5. **Elsevier**: the data is gathered from Elsevier through SFTP as well, in XML format. Their files come as zip and tar files, requiring an extraction process for these different formats.
6. **Springer**: data is also retrieved via SFTP, presented in XML format. The files are delivered as zip, necessitating adjustments to an extraction process to accommodate these varied formats.

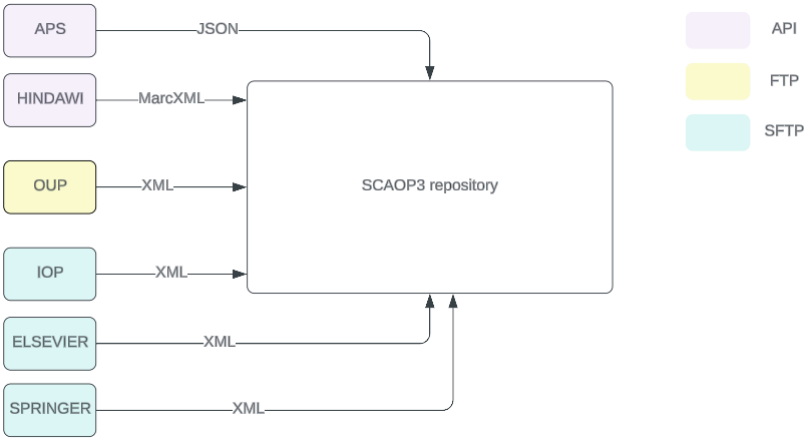

Figure 3. Publishers' data exchange methods and data formats.

After the download of these files is done from each publisher, they store them in an Amazon S3 bucket for safekeeping. Next, we extract the contents from any zip, tar files and JSON responses, and split the data into individual articles if needed. The last step in this phase is to start processing each article, preparing it for the next stages of our workflow.

3.3.4 Data Parsing and Transformation

The custom approach involves the development of specialised parsers for each publisher, such as APS, Springer, IOP, Hindawi, OUP, and Elsevier. As it was mentioned before,these parsers are not just generic tools; they are unique characteristics and formats of data provided by each publisher. This approach is important to accurately extract the necessary information from the diverse data sources we will be handling.

The development of these parsers requires a deep understanding of each publisher's data structure. For instance, a parser designed for Springer's XML format might be substantially different from one handling APS's JSON data. This specificity in design ensures that the extracted data is not only accurate but also comprehensive, capturing all the essential elements required for our repository.

Once the data is extracted, it enters a crucial phase of cleaning and formatting. This stage is designed to address any inconsistencies, errors, or irrelevant information present in the initial data. The aim here is to refine and polish the data, preparing it for integration into our repository.

To enhance efficiency and maintain consistency in this phase, we plan to develop a generic parser. This parser will be engineered to handle the outputs from each specific publisher. While the initial extraction parsers are tailored to individual publishers, this generic parser focuses on standardising the cleaning and formatting process. It is a strategic move to ensure that, despite the initial diversity in data formats, the final output maintains a uniform structure and quality.

The implementation of this generic parser is a delicate balance between flexibility and standardisation. It needs to be flexible enough to adapt to the outputs from various

publisher-specific parsers but also standardised enough to maintain consistency in the final data JSON format. This uniformity is crucial for the next stages of the project, where data from different publishers will be aggregated and analysed collectively.

3.3.5 Data Enhancement and Integration with External Sources

In the SCOAP³ project, following the initial stage of data processing, the progress moves to the data enhancement phase. This crucial step involves enriching our dataset by creating new, comprehensive data fields from existing information. A practical example of this is the consolidation of basic copyright details, which are initially extracted into separate fields, into a singular, coherent location. This phase transcends mere technical data handling; it entails a deeper understanding and analysis of the data to determine what additional information will be most valuable and informative. This strategic enhancement significantly increases the utility and coherence of the dataset, adding layers of clarity and usefulness to the information process.

Additionally, integrating data from external sources such as arxiv.org is a significant part of the approach. This happens in the later stages of the workflow and is essential for adding more depth to our dataset. Incorporating this external data effectively enhances the context and richness of the records. It is important that this integration is done carefully to ensure that the external data complements our existing data without causing duplication or inaccuracies.

In summary, the data enhancement and external data integration stages are about adding more depth and context to our dataset. We are expanding a dataset and also ensuring it is more informative and useful. This approach is key in transforming our collection of records into a comprehensive, well-rounded resource.

3.3.6 Validation and Compliance with Established Schema

In the SCOAP³ project, our data processing workflow concludes with a thorough validation step, ensuring that all records comply with the standards of the SCOAP³ Repository Schema. This stage is crucial for preserving the integrity and accuracy of the data in our repository.

During validation, special attention is given to the enhanced data, which includes both new fields created during processing and data integrated from external sources. The objective is to ensure that this data is not only accurate but also well-structured. It is accomplished through detailed checks, carefully verifying each piece of data against the predefined standards and formats.

Once a record passes the validation checks, signifying that it has been parsed correctly, then this output is stored in an Amazon S3 bucket. This final step of validation and storage is essential in maintaining the high quality and reliability of the dataset, reinforcing the SCOAP³ repository as a valuable resource for academic research and publishing, ensuring its availability and integrity.

3.3.7 Conclusion

This implementation strategy is designed to optimise the data processing workflow within the SCOAP³ repository, leveraging Airflow's capabilities for efficient data management. By systematically addressing each stage - from individual publisher analysis to final data validation - the aim is to create a robust, scalable, and effective data processing pipeline that aligns with the overarching goals of the SCOAP³ initiative.

4 RESULTS AND ANALYSIS

4.1 Data Quality Measures

For the SCOAP³ project, the approach to data quality assessment is underpinned by a focus on the completeness of the dataset, particularly the presence of required fields in each record. This focus is crucial as it directly influences the integrity and utility of the data for both analytical and archival purposes. Given that the data harvested for SCOAP³ is anticipated to be both valid and reliable, sourced from reputable scientific publishers, our quality checks are dedicated to ensuring no critical information is missing from the processed records.

The cornerstone of our quality metrics involves a rigorous verification process to ascertain that every record is complete, containing all required fields. These fields are fundamental for the precise categorization, retrieval, and subsequent analysis of data within the digital repository. Critical fields typically encompass article titles, author names, publication dates, and unique identifiers like DOI numbers. The absence of any of these fields could significantly compromise the dataset's value, potentially introducing gaps in the repository's coverage or metadata inaccuracies.

To enhance the robustness of this verification process, we employ the jsonschema Python library, utilising a schema we have meticulously crafted to match our data structure requirements. This schema serves as a blueprint for the data, specifying the expected format and presence of required fields, thereby automating the validation of each record against these predetermined standards. This validation ensures not only the presence of essential information but also that the data adheres to the expected structure and format, further reinforcing data integrity.

This targeted approach to data quality assessment—focusing on the completeness of required fields and validating record structure and format—optimises our validation process, making it both efficient and thorough. It enables us to quickly identify and rectify datasets that are incomplete or necessitate further communication with publishers. By confirming that all harvested data satisfies these criteria, we ensure the maintenance of a high standard of data integrity, which is vital for the success of the SCOAP³ project.

Through diligent validation of data completeness and structural integrity, using the jsonschema Python library and a custom schema, we affirm the project's dedication to offering a comprehensive and dependable digital repository. This commitment facilitates the global scientific community's access to a wealth of high-quality scholarly articles, supporting research and advancement across multiple scientific disciplines.

4.2 Data Standardisation

Data standardisation is a crucial step in the process to ensure uniformity and accessibility of the data within the SCOAP³ repository. This multifaceted process involves several key actions to prepare and integrate data from various publishers into a coherent, standardised format. Initially, the parsed values undergo a thorough cleaning process, which includes the removal of extraneous spaces, unnecessary strings, and other non-essential elements. This cleansing ensures that the data is not only accurate but also formatted consistently across different records.

Following the cleaning phase, it is proceeded to structure these cleaned values into well-defined fields. This step is vital for organising the data systematically, facilitating its retrieval and analysis. Forming specific fields from the parsed values creates a structured framework that easily accommodates the addition of data from various sources.

Incorporating data from third-party sources, such as arXiv, is another critical aspect of the standardisation process. This involves enriching our dataset with external metadata, categories, or identifiers that enhance the value and utility of the records. The inclusion of arXiv data, for instance, provides deeper insights into the pre-publication history of articles and their subject categorizations, offering a richer context for researchers accessing the repository.

Finally, all cleansed and enriched data is compiled into a final JSON format. A key feature of the standardisation process is that every JSON file—regardless of the originating publisher—adheres to the same structural template. This uniformity ensures that each JSON, representing a distinct article from a separate publisher, maintains a consistent structure. Such consistency across records from different publishers significantly enhances the interoperability and usability of the

data, making it easier for users to navigate, search, and extract information from the SCOAP³ repository.

4.3 Issues Identified and Resolved

During the enhancement of the dataset for the SCOAP³ repository, several challenges were encountered related to human errors and technical issues, which were effectively addressed as follows:

1. **OUP File Extension Errors**: it was discovered that articles intended for the project were uploaded to the OUP FTP server with incorrect file extensions, causing our automated harvesting script to miss these files. This error necessitated immediate corrective action to ensure the collection of relevant data.
2. **Misplaced Records in Hindawi Dataset**: in the Hindawi dataset, some records were inaccurately categorised, rendering us unable to harvest specific articles through their API, as they were not recognized as belonging to the intended dataset.
3. **ArXiv API Downtime and Delayed Category Updates**: A significant challenge involved the arXiv API's frequent downtime. The arXiv platform is crucial for retrieving categories of articles, serving as a repository for preprints across various scientific disciplines. Each article's category is essential for organising and accessing content within our repository. Additionally, the instances were encountered where articles on arXiv were either not categorised immediately or had their categories updated after the initial data retrieval attempt.

To address these issues, proactive engagement with the publishers was essential. For the OUP file extension errors, the publisher was contacted to correct the extensions, enabling the script to process the data successfully. Similarly, Hindawi was informed of the misclassified records, prompting them to reclassify them into the appropriate dataset.

To combat the challenges posed by the arXiv API's unreliability and the need for updating categories, we utilised the Python backoff library to implement a retry mechanism with exponential backoff. This strategy allowed automatically retry data fetching, enhancing the

chances of successful data retrieval amidst API downtimes and capturing category information for articles.

This approach of employing a retry mechanism with exponential backoff, facilitated by the backoff Python library, was instrumental in maintaining the integrity and completeness of the dataset, despite external challenges. It exemplified the commitment to data quality and demonstrated the importance of adaptability and robust communication with data providers in overcoming unforeseen obstacles. These measures significantly minimise the impact on the data collection process, ensuring that the SCOAP³ repository remains reliable and comprehensive. The insights gained have led to improvements in the data quality assessment protocols, incorporating additional checks and contingency strategies to efficiently address similar challenges in the future.

5 CONCLUSION

5.1 Workflow Management Frameworks: Analysis and Selection Insights

This thesis has delved into the roles of Apache Airflow 2.2.4 and Dagster 0.13.0 within the SCOAP³ project, examining how each framework aligns with the project's specific needs for managing and processing a vast array of scientific data. A detailed comparative analysis has unpacked the user-friendliness and overall fit of these frameworks, leading to several important conclusions and insights.

Apache Airflow has proven to be highly effective in orchestrating complex workflows, thanks to its use of Directed Acyclic Graphs (DAGs) for outlining task dependencies and execution paths. Its scalability and extensive plugin ecosystem make it ideal for handling detailed data processing tasks. However, Airflow's complexity and steep learning curve could potentially slow down the onboarding process for new users.

On the other hand, Dagster offers a more accessible approach to workflow management, prized for its simplicity and intuitive pipeline development. Its emphasis on type-checked pipelines and built-in data validation is particularly beneficial for ensuring data quality. Yet, Dagster may not be as scalable as Airflow for projects with very large datasets.

Choosing Apache Airflow for SCOAP³ was a strategic decision based on the project's requirements to efficiently manage and standardise extensive data from various scientific publishers. Airflow's dynamic workflow orchestration capabilities, combined with its seamless integration with diverse data sources and processing tools, directly align with the project's goals. Additionally, the active community and wealth of documentation for Airflow provide crucial support for troubleshooting and optimising workflows.

The experience of integrating Airflow highlights the crucial role of selecting an appropriate data workflow management framework in facilitating the efficient processing and dissemination of data. The lessons learned from this project offer valuable guidance for future data engineering

initiatives, underscoring the importance of a dynamic, solution-oriented approach to managing scientific data.

In conclusion, the adoption of Apache Airflow has set a strong foundation for the SCOAP³ project, aligning with its ambitious goals for data processing and management. This thesis not only contributes valuable insights for the SCOAP³ project but also offers actionable strategies for the broader field of data engineering. Moving forward, the methodologies and insights gained from this work contribute to the ongoing discussion on scientific data management. They offer a basis for further exploration in the field, which could inform future research and the evolution of workflow management practices within the scientific community.

5.2 Recommendations

Building on the work done with the SCOAP³ project, this section outlines practical recommendations for future projects in data management. These suggestions are drawn from our experiences with workflow management tools, data quality checks, and collaboration strategies. The aim is to provide clear, actionable advice that can help improve the efficiency and reliability of similar projects moving forward.

1. **Deep Dive into Workflow Management Tools**: Future efforts should concentrate on thoroughly exploring the capabilities of the chosen workflow management framework, such as Apache Airflow. Specifically, making full use of Airflow's extensive operators can drastically streamline workflow development, minimising the need for custom code and enhancing overall efficiency.
2. **Expand Data Quality Assurance Practices**: Considering the SCOAP³ project's reliance on pre-validated data, future projects without this advantage should broaden their data quality assessment methods. Developing a framework for more complex validation techniques, including semantic analysis and automated anomaly detection, will be crucial for ensuring data integrity across various datasets.
3. **Proactive Collaboration with Data Providers**: Direct engagement with data providers to address and rectify common issues, such as incorrect file extensions or misplaced dataset records, proved essential in maintaining the integrity of the SCOAP³ project's

data pipeline. Continued and enhanced collaboration will be vital, especially in projects where data quality and format consistency are critical from the outset.

4. **Adaptability in Data Collection Strategies**: The implementation of a retry mechanism with exponential backoff, as utilised for managing arXiv API downtime, highlights the importance of adaptable data collection strategies. Future projects should incorporate flexible, resilient approaches to data harvesting to navigate the unreliability of external data sources effectively.

These targeted recommendations aim to underscore the lessons learned from the SCOAP³ project, emphasising the importance of framework familiarity, advanced data quality metrics, collaborative problem-solving, and adaptability in data collection. By adhering to these guidelines, future data management initiatives in scientific research can achieve greater efficiency, reliability, and integrity, building upon the foundational work of the SCOAP³ project. This thesis not only contributes valuable insights into the practical application of data workflow management tools but also sets the stage for further advancements in the field of scientific data processing.

5.3 Research Limitations

This thesis has aimed to provide an in-depth analysis of workflow management frameworks, specifically Apache Airflow 2.2.4 and Dagster 0.13.0, within the context of the SCOAP³ project. While the study has yielded significant insights into the selection and implementation of these frameworks, it is important to acknowledge certain limitations that might influence the scope and applicability of the findings.

Limited Framework Comparison: The comparative analysis was confined to Apache Airflow and Dagster, possibly omitting other workflow management tools that might be equally or more suitable for certain project requirements. The choice of frameworks was guided by the specific needs of the SCOAP³ project, which may not be representative of all data management scenarios.

Project-Specific Findings: The conclusions drawn from this study are deeply rooted in the operational context and specific challenges faced by the SCOAP³ project. As such, the

applicability of these insights to other projects, especially those with differing data workflows or objectives, may be limited.

Assumption of Data Pre-validation: A key assumption underpinning this research was the pre-validation of data by reputable sources. This assumption simplifies the complexity of data quality challenges but does not address the potential intricacies of managing raw or unvalidated data, which could present significant hurdles in other contexts.

Practical Implementation Challenges: The focus of this thesis on theoretical analysis means that practical implementation challenges, such as system integration, data security, and cost considerations, were not exhaustively explored. These factors are critical for the real-world application of workflow management frameworks and could significantly impact their effectiveness and scalability.

Technological Advancements: The rapid evolution of data engineering technologies poses a challenge to the longevity of the study's findings. New developments in workflow management solutions could render some conclusions outdated, emphasising the need for continuous review and adaptation of the chosen frameworks.

User Experience Considerations: Although the complexity and user-friendliness of Apache Airflow and Dagster were discussed, a comprehensive examination of user experience factors was beyond the study's scope. These aspects, including the ease of onboarding, community support, and documentation, are vital for the successful deployment and utilisation of any technology solution.

Considering these constraints, future research should aim to broaden the comparison to include a wider range of workflow management tools, delve into the practicalities of implementing these frameworks in diverse environments, and continuously update the analysis to reflect the latest technological advancements. Such efforts will enhance the robustness of workflow management strategies and support the ever-evolving needs of scientific data projects.

REFERENCES

Allman, M., & Metz, C. (1998, September). RFC 2428: Extensions for IPv6, NAT, and Extended Passive Mode. Search date: 28.02.2024 https://tools.ietf.org/html/rfc2428

AltexSoft. (2022, November 7). The Good and the Bad of Apache Airflow Pipeline Orchestration. Search data: 07.02.2024 https://www.altexsoft.com/blog/apache-airflow-pros-cons/

Amazon Web Services. (n.d.). Benefits of Application Hosting on AWS. Search data: 07.04.2024 https://aws.amazon.com/application-hosting/benefits/

Apache Airflow. (n.d.). Core Concepts: DAGs. Search data: 07.02.2024 https://airflow.apache.org/docs/apache-airflow/stable/core-concepts/dags.html

Arsan, T., Günay, F., & Kaya, E. (2014, August). Implementation of Application for Huge Data File Transfers. International Journal of Wireless & Mobile Networks (IJWMN), 6(4). Search date: 28.02.2024
https://www.researchgate.net/publication/334145268_Application_Programming_Interface_API_Research_A_Review_of_the_Past_to_Inform_the_Future

Bellovin, S.M. (1994, February). RFC 1579: Firewall-Friendly FTP. Search date: 28.02.2024 https://tools.ietf.org/html/rfc1579

Dagster. (n.d.). Getting Started with Dagster. Search data: 23.01.2024 https://docs.dagster.io/getting-started

GitHub. (n.d.). Overview of reusing workflows. Search data: 07.04.2024 https://docs.github.com/en/actions/using-workflows/reusing-workflows#overview

Madhusudan, T., Zhao, J. L., & Marshall, B. (2004, July). A case-based reasoning framework for workflow model management. Data & Knowledge Engineering, 50(1), 87-115. Search date: 28.02.2024 https://doi.org/10.1016/j.datak.2004.01.005

Nordic APIs. (2023). Search data 07.04.2024 https://nordicapis.com/20-impressive-api-economy-statistics/

Nulab. (n.d.). Search data: 07.04.2024 https://nulab.com/blog/tech/microservices-and-apis/

Perumal, I. (2023, December 7). Top 7 Data Integration Challenges and Solutions. DCKAP. Search data: 28.02.2024 https://www.dckap.com/blog/data-integration-challenges/

PlektonLabs. (2021, December 28). Search data: 07.04.2024 https://www.plektonlabs.com/api-reuse-everything-you-need-to-know-do/

RapidAPI. (2023). Search data: 07.04.2024 https://rapidapi.com/blog/state-of-apis-growth-and-more-growth-on-tap-for-2023/

Restack. (n.d.). Dagster History. Search data: 06.02.2024 https://www.restack.io/docs/dagster-knowledge-dagster-history#clpa073cp0f8uyp0ujn2fsi6v

Restack. (n.d.). Dagster Use Cases. Search data: 06.02.2024 https://www.restack.io/docs/dagster-knowledge-dagster-use-cases

Richman, J. (2023, July 31). 11+ Most Common Data Integration Challenges & Solutions. Estuary. Search data: 07.04.2024 https://estuary.dev/data-integration-challenges/

Solanki, J. (2023, March 27). Dagster vs Apache Airflow — side by side comparison. CodeX. Search data: 07.02.2024 https://medium.com/codex/dagster-vs-apache-airflow-side-by-side-comparison-965149997cee

Surani, I. (2020, February 17). Challenges of Integrating Heterogeneous Data Sources. Dataversity. Search data: 28.02.2024 https://www.dataversity.net/challenges-of-integrating-heterogeneous-data-sources/

Synopsys. (n.d.). What are the benefits of CI/CD? Search data: 01.03.2024 https://www.synopsys.com/glossary/what-is-cicd.html

Talend. (n.d.). What is Data Processing? Search data: 07.04.2024 https://www.talend.com/resources/what-is-data-processing/

Yousry, A. (2023, May 25). Don’t Use Apache Airflow in That Way. ILLUMINATION. Search data: 06.02.2024 https://medium.com/illumination/what-apache-airflow-is-not-e9dc9722500b

APPENDICES

APPENDIX 1: SOURCE CODE OF WORKFLOW IMPLEMENTATION

SOURCE CODE OF WORKFLOW IMPLEMENTATION

APPENDIX 1

This appendix provides the URL to the GitHub repository containing all the source code developed and utilised in the thesis. The repository includes scripts, code snippets, and configurations essential for implementing the workflow optimization processes discussed, particularly focusing on the application of Apache Airflow within the SCOAP³ project.

GitHub Repository URL

The complete source code and related materials can be found at the following GitHub repository:
GitHub Repository: cern-sis/workflows

Repository Content Description

The repository is structured to facilitate easy navigation and understanding of the codebase used in the thesis project. It includes:

- Code base for APS, Hindawi, IOP, OUP, Elsevier and Springer DAGs.
- Extensive documentation
- Tests
- Github actions file

Usage Guidelines

Readers wishing to replicate the study's findings, adapt the workflow for their purposes, or explore the code further are encouraged to review the repository's README for detailed setup and execution instructions.