

# **A Comprehensive Data Analytics Framework to Support Research Data Management in Distributed Systems**

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der  
RWTH Aachen University zur Erlangung des akademischen Grades  
eines Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

**M. Amin Yazdi, M.Sc.**

aus Tehran, Iran

Berichter: Universitätsprofessor Dr. rer. nat. Matthias S. Müller  
Universitätsprofessor Dr. rer. pol. Stefan Decker

Tag der mündlichen Prüfung: 12. Dezember 2023

Diese Dissertation ist auf den Internetseiten der Universitätsbibliothek online verfügbar.









*“The scholar’s path, a journey of the soul,  
Through books and scrolls, they strive to enroll,  
In the secrets of the universe unveiled,  
To decipher life’s mysteries, their ultimate goal.”*

— Hafez [Persian Poet] (1325 - 1390)



# Contents

<b>Acknowledgments</b>	<b>v</b>
<b>Abstract</b>	<b>vii</b>
<b>Kurzfassung</b>	<b>ix</b>
<b>List of Figures</b>	<b>xi</b>
<b>List of Tables</b>	<b>xiii</b>
<b>Nomenclature</b>	<b>xv</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivations . . . . .	2
1.2 Related Works . . . . .	4
1.2.1 Interdisciplinary Context . . . . .	4
1.2.2 Open Science Underlying Activities . . . . .	10
1.2.3 Process-Aware Software Analysis in Distributed Services . .	14
1.2.4 Data Abstraction for Business Process Discovery . . . . .	17
1.3 Research Outline . . . . .	21
1.3.1 Research Questions . . . . .	21
1.3.2 Course of Research . . . . .	24
1.3.3 Outline of the Thesis . . . . .	25
<b>2 Landscape of RDM Services</b>	<b>27</b>
2.1 Comparison of RDM Service Providers . . . . .	27
2.1.1 Comparison of Services Based on RDM Lifecycle . . . . .	30
2.1.2 Comparison of Services Based on Technical Aspects . . . . .	31
2.2 Reference RDM Architecture . . . . .	34
2.2.1 Conceptual RDM Architecture . . . . .	35
2.3 Systems Under Study . . . . .	37
2.3.1 SimpleArchive . . . . .	37
2.3.2 Metadata Manager . . . . .	38
2.3.3 Coscine . . . . .	39

<b>3</b>	<b>Data Acquisition Techniques from RDM Systems</b>	<b>43</b>
3.1	Motivations and Challenges . . . . .	44
3.2	Dataset Characteristics and Quality Measures . . . . .	46
3.3	Strategies . . . . .	47
3.3.1	Centralized . . . . .	48
3.3.2	Client-Side . . . . .	52
3.3.3	Server-Side . . . . .	55
3.4	Proposed Key Technique . . . . .	60
3.4.1	Methodology . . . . .	62
3.4.2	Preliminaries . . . . .	67
3.4.3	Discussion . . . . .	70
<b>4</b>	<b>Data Analytics Framework</b>	<b>73</b>
4.1	Semi-Supervised Data Abstraction . . . . .	73
4.1.1	Conceptual Foundations . . . . .	75
4.1.2	Proposed Methodology . . . . .	76
4.1.3	Discussion . . . . .	79
4.2	Modeling Process-Aware Activities . . . . .	80
4.2.1	Characteristics . . . . .	81
4.2.2	Functionalities . . . . .	82
4.2.3	Software Design and Architecture . . . . .	85
4.2.4	Discussion . . . . .	95
4.3	Discovering RDM Lifecycle . . . . .	97
4.3.1	Conceptual Foundations . . . . .	97
4.3.2	Proposed Methodology . . . . .	98
4.3.3	Discussion . . . . .	102
4.4	Data Collections Reusability Enhancement . . . . .	103
4.4.1	Content-Based Similarity Measurement . . . . .	103
4.4.2	User-Interaction Based Collaborative Filtering . . . . .	108
4.4.3	Discussion . . . . .	115
<b>5</b>	<b>Case Studies</b>	<b>119</b>
5.1	Data Extraction . . . . .	120
5.1.1	RWTH Distributed Services . . . . .	120
5.1.2	Coscine . . . . .	125
5.1.3	Summary . . . . .	128
5.2	Event Data Abstraction . . . . .	130
5.2.1	Research Data Archiving Tool (SimpleArchive) . . . . .	130
5.2.2	Metadata Management Tool (MetadataTool) . . . . .	133
5.2.3	Summary . . . . .	137
5.3	Knowledge Discovery . . . . .	139
5.3.1	Descriptive Research Processes . . . . .	139

---

5.3.2	Control-Flow Requirement Analysis . . . . .	143
5.3.3	University Organizational Mining . . . . .	146
5.3.4	CRC1394 RDM Lifecycle . . . . .	148
5.3.5	Summary . . . . .	152
5.4	Research Data Recommender System . . . . .	154
5.4.1	Content-Based Recommender for CRC1394 . . . . .	154
5.4.2	Item Based Recommender for Cosine Resources . . . . .	159
5.4.3	Summary . . . . .	163
<b>6</b>	<b>Conclusion</b>	<b>167</b>
6.1	Answering Research Questions . . . . .	167
6.2	Contributions . . . . .	173
6.3	Outlook . . . . .	174
	<b>Appendix</b>	<b>177</b>
A.1	Screenshots of Case Studies . . . . .	177
A.2	Python Packages Developed . . . . .	185
	<b>List of Publications</b>	<b>191</b>
	<b>Statement of Originality</b>	<b>193</b>
	<b>References</b>	<b>195</b>



# Acknowledgments

First and foremost, I would like to express my sincere appreciation to my wife, Mahshid, who has supported me throughout this endeavor. Her unwavering love, understanding, and encouragement enabled me to pursue my aspirations. Mahshid, you have been my guiding light and have shown immense patience with me. I am eternally grateful for your presence beside me every step of the way, particularly during the most challenging periods of our lives. Moreover, to our son Daniel, who would read this in the future, your arrival brought immeasurable joy and delight to our lives; I appreciate the motivation you have given me to be productive and to understand the value of time being together. Additionally, to my family, who have supported me during my academic journey. Their unceasing love, advice, and encouragement have played a crucial role in molding my character and helping me reach my objectives.

I am deeply grateful to Prof. Matthias Müller, my principal supervisor, who recognized my potential and granted me the opportunity to undertake this research journey. His mentorship, wisdom, and support have been indispensable in guiding my research and facilitating my development as a scholar. I would also like to thank my second supervisor, Prof. Stefan Decker, for his insightful input and constructive critiques of my research. His expertise and commitment to my success have greatly influenced this dissertation.

I sincerely thank Dr. Marius Politze for his invaluable iterative feedback throughout my research. His intellectual support and guidance have substantially contributed to the evolution and fine-tuning of my work.

I am grateful to my dear friends, particularly Majid and Mahsa, for their technical insights, friendship, and support. Their fellowship has made this academic goal achievable. Also, I wish to express my appreciation to my colleagues at the IT Center for their expertise, collaboration, and support. Lastly, I would like to thank the examination committee for their dedication and effort in evaluating my dissertation. Their feedback and suggestions have been invaluable in polishing my work and ensuring its high quality.





# Abstract

Effective Research Data Management (RDM) practices are essential for fostering research collaboration, increasing discoverability and repurposing research data, and advancing scientific progress in higher education. In recent years, adopting Open Science Platforms (OSPs) and the Findable, Accessible, Interoperable, and Reusable (FAIR) data principles has highlighted the need for improved RDM methodologies and tools for flourishing higher education achievements. However, existing literature has provided limited guidance on monitoring RDM processes, their adoption, and their use. This dissertation addresses this gap by investigating how to enable discovering and enhancing process-aware RDM activities via modeling the underlying researcher's actual practices.

This dissertation presents a series of methodologies as a framework combining data acquisition, abstraction, knowledge discovery, and operation enhancement techniques. Furthermore, the case studies highlight the challenges associated with RDM-related activities by assessing the proposed methodologies' validity in real-world environments. Initially, this work presents a universal reference software architecture for RDM services; then, it proposes four approaches for data acquisition, including a novel Hybrid logger technique for acquiring datasets from information systems that operate on distributed settings, providing a comprehensive view of user activities by evaluating corresponding software component executions. This approach enables a projection of user behavior and facilitates the development of further machine-learning studies.

Furthermore, this work introduces a semi-supervised learning approach for abstracting datasets by accommodating non-sequential events in distributed systems while balancing data granularity and model fitness. The methodology for discovering process-aware activities incorporates a modular and layered architecture, providing insights into RDM compliance, identifying deviations, and optimizing user experience. Additionally, it outlines a method for determining and visualizing the user and system interactions and discovers the RDM phases of research projects, providing a practical understanding of the progression and activities of different research groups.

Finally, this thesis proposes and evaluates two recommender systems, demonstrating the potential of Content-Based and Collaborative Filtering recommender systems in enabling the reusability of research data repositories and fostering cooperation among researchers. The findings contribute significantly to the expanding body of

literature on RDM and provide valuable insights into the potential of the presented methodologies for enhancing RDM practices in OSPs.

In conclusion, this dissertation offers holistic strategies for addressing the difficulties related to facilitating RDM in OSPs, providing guidelines for implementing necessary architecture and demonstrating the applicability of the proposed methods to other RDM services that adhere to the reference software architecture of RDM systems.

# Kurzfassung

Effektive Praktiken im Bereich des Forschungsdatenmanagements (FDM) sind essenziell, um Forschungskollaborationen zu fördern, die Auffindbarkeit und Wiederverwendung von Forschungsdaten zu erhöhen und den wissenschaftlichen Fortschritt in der Hochschulbildung voranzutreiben. In den letzten Jahren hat die Einführung von Open-Science-Plattformen (OSPs) und die FAIR-Prinzipien (findable, accessible, interoperable, reusable) die Notwendigkeit verbesserter FDM-Methoden und -Werkzeuge für den Erfolg von Hochschulen unterstrichen. Die vorhandene Literatur bietet bisher jedoch nur wenig Anleitung zum Monitoring von FDM-Prozessen, deren Einführung und Nutzung. Diese Dissertation schließt diese Lücke, indem sie untersucht, wie man die Entdeckung und Verbesserung von prozessbewussten RDM-Aktivitäten durch Modellierung der tatsächlichen Praktiken der Forschenden ermöglichen kann.

Diese Dissertation präsentiert eine Reihe von Methoden als Baukasten, der Datenakquise, Abstraktion, Wissensentdeckung und Prozessverbesserungen kombiniert. Die Fallstudien heben die Herausforderungen hervor, die mit FDM-bezogenen Aktivitäten verbunden sind, indem sie die Gültigkeit der vorgeschlagenen Methoden in realen Umgebungen bewerten. Zunächst präsentiert diese Arbeit eine universelle Referenzsoftwarearchitektur für FDM-Dienste; dann schlägt sie vier Ansätze zur Datenakquise vor, einschließlich einer neuartigen Hybrid-Logger-Technik zur Akquise von Datensätzen aus verteilten Informationssystemen und bietet einen umfassenden Überblick über Benutzeraktivitäten durch Auswertung entsprechender Softwarekomponentenausführungen. Dieser Ansatz ermöglicht eine Projektion des Benutzerverhaltens und erleichtert die Entwicklung weiterer Studien im Bereich maschinelles lernen.

Darüber hinaus führt diese Arbeit einen semi-überwachten Lernansatz zur Abstraktion von Datensätzen ein, indem sie nichtsequenzielle Ereignisse in verteilten Systemen unter Beibehaltung des Gleichgewichts zwischen Datenkörnung und Modellpassung berücksichtigt. Die Methodik zur Entdeckung prozessbewusster Aktivitäten beinhaltet eine modulare und geschichtete Architektur, die Einblicke in die Einhaltung von FDM, die Identifizierung von Abweichungen und die Optimierung des Benutzererlebnisses bietet. Darüber hinaus skizziert sie eine Methode zur Bestimmung und Visualisierung der Interaktionen zwischen Benutzer und System und

entdeckt die FDM-Phasen von Forschungsprojekten, was ein praktisches Verständnis des Fortschritts und der Aktivitäten verschiedener Forschungsgruppen bietet.

Schließlich schlägt diese Arbeit zwei Empfehlungssysteme vor und bewertet sie, um das Potenzial von inhaltsbasierten und kollaborativen Filterempfehlungssystemen zur Förderung der Wiederverwendbarkeit von Forschungsdatenbanken und zur Förderung der Zusammenarbeit zwischen Forschern aufzuzeigen. Die Ergebnisse tragen erheblich zur Erweiterung der Literatur über FDM bei und bieten wertvolle Einblicke in das Potenzial der vorgestellten Methoden zur Verbesserung von FDM-Praktiken in OSPs.

Die Dissertation entwirft ganzheitliche Strategien zur Bewältigung der Schwierigkeiten, die mit der Erleichterung von FDM in OSPs verbunden sind, gibt Richtlinien für die Implementierung der notwendigen Architektur vor und demonstriert die Anwendbarkeit der vorgeschlagenen Methoden auf andere FDM-Dienste, die sich an die Referenzsoftwarearchitektur von FDM-Systemen halten.

# List of Figures

1.1	A depiction of the interdisciplinary study of RDM and research gaps. . .	5
1.2	Research life cycle . . . . .	11
1.3	Phases of research data management. . . . .	13
1.4	Unexplored research gaps and associated research questions. . . . .	22
1.5	A representation of the course of research. . . . .	24
2.1	A universal reference architecture for RDM services. . . . .	36
2.2	pSTAIX model for archiving files on SimpleArchive. . . . .	37
2.3	pSTAIX model for uploading metadata on Metadata Manager. . . . .	38
2.4	pSTAIX model for creating projects and resources on Coscine. . . . .	39
2.5	pSTAIX model for uploading files and metadata on Coscine. . . . .	40
3.1	OAuth workflow in distributed services. . . . .	49
3.2	The Client-Side logger workflow. . . . .	53
3.3	The Server-Side logger workflow . . . . .	56
3.4	The schematic representation of the Hybrid-logger. . . . .	63
3.5	Event Log Generator algorithm and dataset transformations . . . . .	65
4.1	Event log abstraction methodology. . . . .	77
4.2	DA4RDM: Representation of its framework and modules. . . . .	82
4.3	DA4RDM: Input data configuration page. . . . .	83
4.4	DA4RDM: Data pre-processing page. . . . .	84
4.5	DA4RDM: Process discovery page. . . . .	85
4.6	DA4RDM: Conformance checking page. . . . .	86
4.7	DA4RDM: RDM visualization page. . . . .	86
4.8	DA4RDM: Schematic representation of software components. . . . .	88
4.9	DA4RDM: file creation and uploading flowchart. . . . .	90
4.10	DA4RDM: Data sources relational model. . . . .	91
4.11	DA4RDM: User session relational model. . . . .	92
4.12	RDM phases and their corresponding user interaction models. . . . .	99
4.13	Pairwise fitness scores of user interaction models in RDM. . . . .	100
4.14	Content-Based recommender system pipeline. . . . .	105
4.15	Item-Item collaborative filtering recommender system pipeline. . . . .	112
5.1	A schematic representation of RWTH distributed services. . . . .	120
5.2	IT Center software components interconnection and performance evaluation. . . . .	123

5.3	Service level software interconnections and process model. . . . .	124
5.4	Coscine research data management page. . . . .	125
5.5	Coscine: Frequency analysis of activities over resources. . . . .	127
5.6	Coscine: Performance analysis at the research projects level. . . . .	128
5.7	Data Archiving Tool: Abstracted user interaction models. . . . .	132
5.8	Data Archiving Tool: Correlation of abstraction to classifiers' accuracy. .	133
5.9	Metadata Manager Tool: Abstracted user interaction models. . . . .	135
5.10	Metadata Manager Tool: Correlation of abstraction to classifiers' accuracy.	136
5.11	DA4RDM: User interaction model at Coscine resource level. . . . .	140
5.12	Hierarchical user interaction models on Coscine. . . . .	141
5.13	DA4RDM: Data-Driven requirement engineering on Coscine. . . . .	144
5.14	DA4RDM: Interoperability assessment of Coscine. . . . .	145
5.15	Role mining using machine learning classifiers. . . . .	147
5.16	Two-years activity pattern evolution for CRC1394. . . . .	149
5.17	Radar chart of RDM lifecycle for CRC1394 and its three sub-groups. . .	150
5.18	Files distribution across repositories. . . . .	155
5.19	Data sparsity of sample dataset for CB Recommender System. . . . .	156
5.20	Pairwise repositories similarity ratings using CB recommender. . . . .	158
5.21	Class-specific and overall AUC performance using CB recommender. . . .	158
5.22	Statistical analysis of sample dataset used in IICF. . . . .	161
5.23	Pairwise repositories similarity ratings using CF recommender. . . . .	162
6.1	Research questions meta-model. . . . .	168
A.1	DA4RDM Screenshot: Dataset source definition and parameters. . . . .	178
A.2	DA4RDM Screenshot: Data source and pre-processing pipeline selection. .	178
A.3	DA4RDM Screenshot: Process discovery and filter options. . . . .	179
A.4	DA4RDM Screenshot: Conformance checking and requirement engineering. .	179
A.5	DA4RDM Screenshot: RDM life cycle visualization for a project. . . . .	180
A.6	Coscine Screenshot: Login page . . . . .	180
A.7	Coscine Screenshot: Dashboard. . . . .	181
A.8	Coscine Screenshot: Project view. . . . .	181
A.9	Coscine Screenshot: Resource and metadata management. . . . .	182
A.10	SimpleArchive Screenshot: Archiving research data. . . . .	182
A.11	SimpleArchive Screenshot: List of archived nodes. . . . .	183
A.12	Metadata Manager Screenshot: Specialized application profile. . . . .	184
A.13	Metadata Manager Screenshot: Metadata entries. . . . .	184
A.14	Metadata Manager Screenshot: Metadata search user interface . . . . .	185

# List of Tables

1.1	Approaches for analysis of distributed systems. . . . .	15
1.2	Classification of several event log abstraction techniques. . . . .	18
2.1	Service comparison in relation to RDM phases. . . . .	31
2.2	A technical analysis of services facilitating RDM. . . . .	41
3.1	An overview of logging methods in terms of data quality metrics. . . . .	48
3.2	Example events captured by the Centralized logger. . . . .	51
3.3	Example events captured by the Client-Side logger. . . . .	55
3.4	Example events captured by the Server-Side logger. . . . .	59
4.1	Positioning the proposed method in relation to data abstraction standards. . . . .	75
4.2	Example events derived by Server-Side logger for RDM analysis. . . . .	97
4.3	Sample dataset aligned with CRC1394 team’s metadata profile. . . . .	105
5.1	An example dataset from the University’s distributed services. . . . .	122
5.2	Data objects prepared for DA4RDM data analysis. . . . .	127
5.3	Hybrid logger performance assessment on Data Archiving Tool . . . . .	130
5.4	Abstraction of sample dataset for Data Archiving Tool . . . . .	131
5.5	Hybrid logger performance assessment on Metadata Manager Tool . . . . .	134
5.6	Abstraction of sample dataset for Metadata Manager Tool . . . . .	134
5.7	Descriptive matrix from expanded and altered dataset via Centralized logger. . . . .	146
5.8	Class assessment of CB recommender using Cosine and Euclidean metrics. . . . .	157
5.9	Overall CB recommender performance using Cosine and Euclidean metrics. . . . .	157
5.10	Class assessment of CF recommender using Cosine and PCC metrics. . . . .	163
5.11	Overall CF recommender performance using Cosine and PCC metrics. . . . .	163





# Nomenclature

**API** Application Programming Interface.

**AUC** Area Under the Curve.

**BERT** Bidirectional Encoder Representations from Transformers.

**BPI** Business Process Intelligence.

**CB** Content-Based.

**CF** Collaborative Filtering.

**Coscine** Collaborative Scientific Integration Environment.

**CRC** Collaborative Research Centre.

**CSV** Comma-Separated Values.

**CVC** Centralized Version Control.

**DA4RDM** Data Analysis for Research Data Management.

**DCAT** Data Catalog Vocabulary.

**DFG** Directly Follow Graph.

**DMP** Data Management Plan.

**DOI** Digital Object Identifier.

**DT** Decision Tree.

**DVC** Distributed Version Control.

**ECS** Elastic Container Service.

**ELG** Event Log Generator.

**ELN** Electronic Lab Notebook.

**EOSC** European Open Science Cloud.

**ePIC** European Persistent Identifier Consortium.

**FAIR** Findable, Accessible, Interoperable, and Reusable.

**FDM** Forschungsdatenmanagements.

**FDO** FAIR Digital Objects.

**FTP** File Transfer Protocol.

**GDPR** General Data Protection Regulation.

**HCI** Human Computer Interaction.

**IICF** Item-Item Collaborative Filtering.

**IM** Inductive Miner.

**IODE** International Oceanographic Data and Information Exchange.

**JSON** JavaScript Object Notation.

**JWT** JSON Web Token.

**KNN** K-Nearest Neighbors.

**KPI** Key Performance Indicator.

**LDAP** Lightweight Directory Access Protocol.

**MAE** Mean Absolute Error.

**MD** Metadata.

**NFDI** Nationale Forschungsdaten Infrastruktur.

**NLP** Natural Language Processing.

**OAI-PMH** Open Archives Initiative Protocol for Metadata Harvesting.

**OAuth** Open Authorization.

**OSF** Open Science Framework.

**OSP** Open Science Platform.

**PCA** Principal Component Analysis.

---

**PCC** Pearson Coefficient Correlation.

**PID** Persistent Identifier.

**PM4PY** Process Mining for Python.

**pSTAIX** Process-Oriented Software Tiers for Application Interfaces and eXtension.

**RCV** Resource Content View.

**RDA** Research Data Alliance.

**RDF** Resource Description Framework.

**RDLC** Research Data Life Cycle.

**RDM** Research Data Management.

**RDMO** Research Data Management Organizer.

**REST** Representational State Transfer.

**RF** Random Forest.

**RLC** Research Life Cycle.

**RMSE** Root Mean Squared Error.

**ROC** Receiver Operating Characteristic.

**RQ** Research Question.

**SAML** Security Assertion Markup Language.

**SHACL** Shapes Constraint Language.

**SMOTE** Synthetic Minority Oversampling Technique.

**SQL** Structured Query Language.

**SSO** Single Sign On.

**STS** Secure Token Service.

**SUS** System Under Study.

**SVD** Singular Value Decomposition.

**TF-IDF** Term Frequency - Inverse Document Frequency.

**UCD** User-Centered Design.

**UICF** User-Item Collaborative Filtering.

**URL** Uniform Resource Locator.

**UX** User Experience.

**WORM** Write-Once-Read-Many.

**XES** eXtensible Event Stream.

**XML** eXtensible Markup Language.





# 1 Introduction

Every research project attempts to accelerate the world toward discovering new knowledge in order to make it a better place to live. In traditional research environments for data-driven studies, researchers tend to rely on data accessible in close physical proximity, resulting in substantial struggles in reproducing their peers' results and reusing preliminary research data. Nevertheless, "*in order to make progress in science, we need to be open and share*" [1]. Today, in the era of digitalization and the emergence of the need for support for scientific research operations, different terms such as eScience, Open Research, eResearch, Open Science, or Research Software Engineering are used to express and emphasize aspects of good scientific practices for research data collection, organization, management, sharing and reusing [2].

This dissertation defines Open Science Platforms (OSPs) as digital environments, infrastructures, or services that facilitate and promotes Research Data Management (RDM) practices by providing researchers, institutions, and other stakeholders with tools, resources, and collaboration opportunities. These platforms are not merely standalone utilities; their benefit is derived from their systematic structure on top of distributed systems facilities. This underpinning enhances the robustness, accessibility, and scalability of these platforms, supporting scientific cooperation. Consequently, OSPs aim to make scientific research more transparent, accessible, and reproducible by offering features that support storing and sharing research data and study results. These platforms often adhere to the FAIR (Findable, Accessible, Interoperable, and Reusable) principles, which guide the proper management and sharing of research data. In addition, OSPs can include functionalities such as data repositories, publication archives, collaboration tools, data management systems, and analytics services. By leveraging OSPs, researchers can more easily discover, access, and reuse research data and findings, fostering collaboration, knowledge dissemination, and scientific progress.

There are many stand-alone services specialized in specific RDM activities, but OSPs such as Collaborative Scientific Integration Environment (Coscine) [3], Open Science Framework (OSF) [4], or Zenodo [5] attempt to integrate distributed and heterogeneous RDM services into a centralized gateway. Lefebvre et al. [6] and Mosconi et al. [2] call for an investigation and obtain more insights on RDM processes in research organizations due to current limited knowledge about the actual deployment of RDM

aspects to optimize research data management tools competently. By enabling the study of RDM activities, this thesis aims to provide insights into compliance of the OSPs according to RDM requirements.

The European Open Science Cloud (EOSC) under the Horizon Europe program has initiated an infrastructure to promote Open Science practices and envisioned its policy to:

Enable trusted access to services, systems and the reuse of shared scientific data across disciplinary, social and geographical borders.

— EOSC, 2016 [7]

In order to facilitate the Open Science envisioned guidelines for handling research data to openly promote sharing and collaborating on research projects, various heterogeneous, distributed IT services are available to researchers. The resulting continuously evolving research data management plans require constant monitoring and maintaining Open Science standards within an IT infrastructure. However, this requires specialized tools to manage research data, and consequently, it adds layers of complexity concerning services providing research data management facilities.

So far, there has been little interest in investigating the gap between Open Science principles and researchers' actual practices, but the consequent efforts for the digitalization of research data management principles create opportunities to study underlying descriptive Open Science processes. Analyzing researchers' practices within the research data management context requires accurate and precise data to enable exploring business processes and gaining valuable insights into actual RDM operations. One needs to be familiar with Distributed Systems, RDM, and Data Science domains to investigate underlying research processes in research data management systems effectively.

In this chapter, Section 1.1 heightens the necessity and impact of the analysis of underlying Research Data Life Cycle (RDLC) processes within RDM; Section 1.2 emphasizes the interdisciplinary context and discusses an in-depth look into the state of the art and current research in the fields. Finally, Section 1.3 elaborates on research questions driven by identified research gaps.

## 1.1 Motivations

The symbiotic relationship between industry and academia is an essential driver of every nation's economic development. On the one hand, academia produces excellent scientists and experts who continuously push for innovative ideas and knowledge to shift state of the art. On the other hand, industries look for the applicability, scalability, and reproducibility of those scientific findings in real-world environments



to generate economic wealth eventually. Unfortunately, although the academic and industry objectives intend to go hand in hand, scholars are mostly encouraged to focus on publishing their results in scientific journals. As a result, they are less attentive to ensuring the publishing of verifiable discoveries that meet the industry's needs via disseminating their research data.

Thanks to recent attention to digitalization, there is a radical change in enabling comprehensive acceleration of the knowledge turnover [8]. Increasingly, the data sources are shifting from analog to digital resources to respond to the demands of articulating the Open Science; however, “*The data generated in this deluge requires active management to meet basic needs of access and reuse*” [9]. Additionally, “*ethnographic research requires more than just data. [...] we need a good sense of what the context in question might be from the point of view of the researcher*” [2]. Subsequently, as part of the EU's research and innovation funding program (Horizon 2020) [10], the European Commission has adopted and released a set of measurable principles for *Good Scientific Practices* in order to ensure Findable, Accessible, Interoperable, and Reusable (FAIR)ness of research data [11].

As a central pillar of FAIR principles to enable reusability and verifiability of findings, universities have made significant efforts to encourage researchers to share their research data, metadata, and findings in the public domain by providing appropriate toolsets and data stewardship consultants. Furthermore, to align with the Open Science agenda, an increasing number of funding agencies mandate providing research Data Management Plan (DMP) as an indispensable and essential measure for receiving research funding. A DMP describes the strategies for efficiently structuring and handling research data before and after a research project [12]. Nevertheless, applying the FAIR codes to researchers' practices, although necessary, poses unprecedented challenges to the actual conduct of research, curating data, and data stewardship [13]. Moreover, the apparent discrepancies between the discipline-specific requirements and conditions of funding institutions add additional layers of complexity for satisfying both the heterogeneous nature of research practices and DMPs [14]. Therefore, the *top-down* enforcement of FAIR doctrines results in a gap between the Open Science vision and the researcher's actual practices [2], [15], which demands a further study of the research process within RDM software tools that facilitate the data collection, analysis, storage, and publishing.

Experts in the field of RDM argue that the significant effort required by researchers for structuring and documentation of data and the uncertainty over data accessibility leads to allocating limited time for RDM, hence, viewing it as a low-priority task for researchers' activities [2], [16]–[18]. However, researchers utilize various software tools to assist them with the tasks of data collection, analysis, storage, and sharing to satisfy the DMP requirements. Accordingly, scientists usually employ data management tools that can easily fit into researchers' daily activities while adhering to the FAIR guidelines and providing control over data and research data access [19],

[20]. However, these tools are often decentralized and heterogeneous. Furthermore, the diversity of data and distributed RDM services make it inquiring to understand the context and meaning of the data. This is due to the specific manners scientists conduct research and process data, resulting in challenging knowledge transfer and data reusability [17], [19].

Politze et al. [21] call for a more profound understanding of research practices to better understand the underlying activities of data stewardship practices by investigating the RDM activities. However, to manifest the process-oriented research environment, a careful bottom-up approach is necessary to investigate how the research data is prepared and used within the decentralized software architecture of RDM systems. Enabling the RDM system landscape to collect, process, and analyze actual research practices is a fundamental challenge to provide added value, insights, and services for researchers.

## 1.2 Related Works

One needs to begin with the related works and studies conducted in the field to address the lack of sufficient command for investigating Open Science activities and provide insights into actual researcher’s practices. Therefore, this Section begins by throwing light on the interdisciplinary nature of this thesis and describes the necessary information required for enabling the investigation of Open Science and RDM. Moreover, Section 1.2.3 provides an overview of the extraction of process-aware data from distributed environments, and Section 1.2.4 describes state of the art on data abstraction to discover underlying processes in RDM systems successfully.

The work in this Section is partially based on self previously published research and supervised technical reports. Most notably, the Open Science underlying activities have been previously discussed in [21]–[23], the process-aware software analysis of distributed services is presented by Yazdi et al. [24]–[26], and the related study on data abstraction for the task of business process discovery is also presented by Yazdi et al. [27].

### 1.2.1 Interdisciplinary Context

Despite the Data Modeling task being broadly regarded as a Data Science related discipline, one needs to consider several other domains involved to address the existing challenges in the RDM real-world environment. This section discusses the relationship of various research fields to Open Science and their state-of-the-art to uncover the interdisciplinary nature of this research project. The challenge in this

thesis is *to provide a set of novel methods that combine scientific solutions from various disciplines and harmonize approaches toward a common goal in order to satisfy the research objectives.*

Accordingly, this thesis contributes to three main disciplines that significantly influence the course of study: Research Data Management, Distributed Systems, and Data and Process Science. Additionally, the advances in Human Computer Interaction (HCI), Privacy, and Requirement Engineering domains navigate my suggested approaches to ensure the applicability of the proposed methodology in real-world settings. Figure 1.1 illustrates an overview of research streams and highlights the corresponding research gaps.

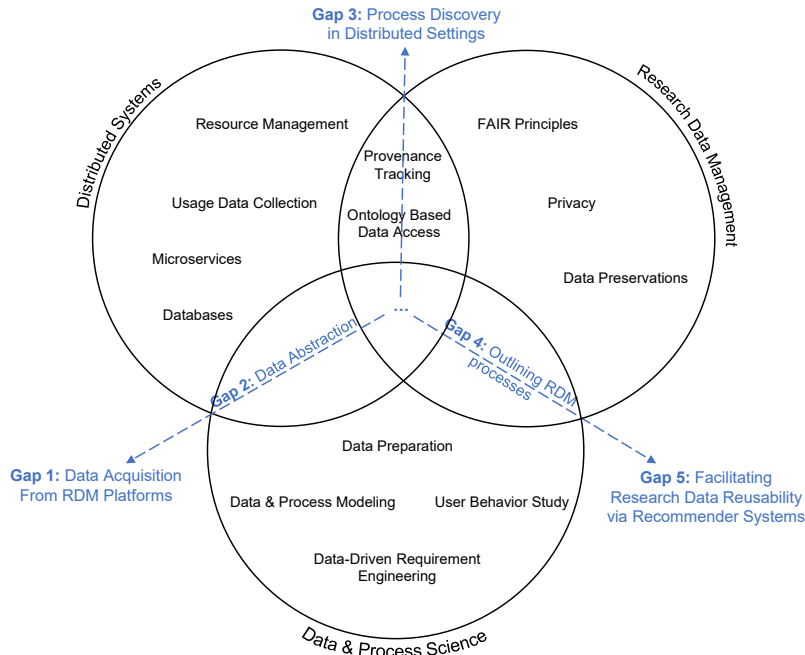


Figure 1.1: A visual depiction of the interdisciplinary study of RDM and existing research gaps.

## Research Data Management

In general, the term Research Data Management refers to all policies and activities involved in handling research data before, during, and after starting a research project within an Open Science setting [2], [28]. Tenopir et al. and Cox claimed that university libraries are naturally suitable centers for discussing RDM activities and services [29], [30]. However, although librarians are an ideal source of information for RDM operations and policy development, they are often not equipped with enough resources and competencies to provide the necessary IT infrastructure [2], [31]. Furthermore,

university libraries often see RDM as the extension of traditional consultation services to advise researchers on available internal or external tools [32].

In order to deliver high-quality research results and encourage data sharing, Hua et al. debate the need for open, collaborative scientific platforms where researchers can benefit from long-term data preservation solutions and enable cross-disciplinary research projects [33]. Authors discovered the cloud-based tools like Sciebo [34], Dropbox [35], Google Drive [36], Gitlab [37] to be the most famous and common tools among scholars to facilitate storage and managing data within a community despite lack of support for implementing or enforcing RDM policies.

Ashiq et al. [31] call for a necessity of generic IT services that incorporate RDM policies as a cornerstone to long-term data storage. Hitherto, there are specialized Open Science forums such as TR CRC 32 for geographical data [38], chemical sample management [39], or medical study data [40] are focused on discipline-specific workflows, and their study shows its approval by domain users. Nevertheless, these often satisfy the needs for narrow research communities with limited, targeted RDM objectives and do not support scalability to conduct more extensive interdisciplinary studies. To deliver a universal OSP, IT service providers of universities need to close the gap between these ad-hoc RDM solutions and cater to more generic RDM workflows to address a wider community of researchers [21]. A more detailed study of related work on RDM and Open Science processes is discussed in Section 1.2.2.

### Distributed Systems

It refers to a collection of independent computers sharing resources such as software components, databases, services, or computational power within a network to achieve a common goal [41]. Likewise, decentralized systems contribute to RDM workflows by typically employing distributed storage services that span across physical boundaries [42]. Yet, Yazdi et al. [26] discuss the challenges of the RDM task for data managers to keep track of all resources in cross-institutional and distributed settings and a need for a single point of user interaction despite utilizing multiple services in an RDM workflow.

Additionally, as part of RDM goals, an OSP that benefits from the advantages of distributed systems should incorporate data provenance requirements. Data provenance is described as “*any information describing the production process of an end product*” [43]. For instance, Mufti et al. and Hu et al. have investigated the challenges and benefits of data provenance in the Internet of Things (IoT) as an example of a distributed environment [44], [45]. The authors concluded that although upholding data provenance within distributed systems can optimize processing performance, increase trust in the data reliability or facilitate recovery, it raises concerns about data security, indexing, accessibility, or unforeseeable data migrations. Scientific

collaboration platforms such as OSF[46], and Coscine [21] seek to resolve these issues by integrating a python web application such as WaterButler [47] for interacting with dispersed file storage services across domains in a single user interface, using Representational State Transfer (REST)ful Application Programming Interface (API). In addition, although Persistent Identifier (PID) are used to reference research data uniquely [48], the tracking of the path a research data takes between resources is still a challenging task due to the heterogeneous and distributed nature of the system.

In order to deal with the issue of tracking research data in a dispersed setting, Smith et al. and Stephan et al. [49], [50] have suggested toolkits to log the data provenance but fail to demonstrate high-level to low-level provenance related activities, lack the possibility for scalability with the growth of an OSP and require source code modifications to collect logs. Thus, Section 1.2.3 provides an in-depth look into the criteria for investigating the distributed systems to collect process-aware details in distributed services. Furthermore, Chapter 3 represents data mining methods to capture related information with minimal changes while interpreting user actions on research data within a distributed and heterogeneous environment.

## Data and Process Science

As are explained by several scholars [51]–[53], Data Science includes but not limited to data extraction, preparation, transformation, presentation, predictions, and visualization of a massive amount of structured or unstructured data that are static or streaming. It aims to turn data into real value by analyzing an extensive data set and obtaining insights from the mass of data. A comprehensive description of data science is given by Van der Aalst [54]: “*Data science is an interdisciplinary field, ..., it includes data extraction, data preparation, data exploration, data transformation, storage and retrieval, computing infrastructures, various types of mining and learning, presentation of explanations and predictions, and the exploitation of results taking into account ethical, social, legal, and business aspects.*”.

As a sub-discipline of data science, Van der Aalst has introduced process mining and suggested three types of phases [54], [55]. Namely, *Model Discovery* algorithms such as Fuzzy Miner and Inductive Miner (IM) [56]–[58] are usually employed as a starting point for extracting a business process model from raw event logs. Each event contains at least a Case, an Activity, and a Timestamp. *Conformance Checking* techniques compare an existing process model with an event log of the same process to check if the extracted model complies with that of reality and vice versa. This gives opportunities to diagnose deviations and determine the effects of non-compliance events and bottlenecks [24]. Finally, *Enhancement* aims to improve the actual process performance by extending the former model [59].

Data and Process Science disciplines equip scientists with techniques to uncover hidden insights from data sets and explain complex behaviors and trends within a system. Developing process-aware RDM workflow requires tracking and understanding how an Open Science system is being used [2]. Thus, Business Process Intelligence (BPI) supports modeling user interaction processes with research data and how software components are interconnected to fulfill user requests. Accordingly, many potentials for improving a system can be identified by modeling and mapping an actual user journey to an expected process model [22].

Moreover, researchers have also emphasized the limitation of data analysis for large, unstructured, and complex processes [60], [61]. Therefore, to extract insightful information from extensive data sets, approaches such as *combination of abstraction* and *clustering* are proposed to simplify the disorderly processes and prepare data for process intelligence analysis [62]. Section 1.2.4 highlights a detailed view of current data abstraction techniques for process discovery. Furthermore, Section 4.1 proposes a novel scientific methodology for data abstraction to allow for the extraction of insightful knowledge.

### Human Computer Interaction

The field of HCI have long been engaged in studying collaborative research practices and the necessary infrastructure to support them. Relevant research in these areas has examined the practices within extensive, enduring, and geographically dispersed research endeavors, exploring the sociotechnical infrastructure required to facilitate common resource sharing, dataset access, and specialized tools for data storage and processing [63], [64] conducted an ethnographic investigation of the practices employed in a trailblazing effort in research data management and sharing as part of a long-term ecological research program [65]. Their study offered valuable insights into the complexities associated with local data stewardship by observing and giving voice to scientists and data managers collaborating.

Such insights are increasingly crucial in light of the Open Science movement, which emphasizes the need for institutionalization and standardization across all research disciplines. Researchers such as Yazdi et al. [26] and Calero Valdez et al. [23], [66] have detailed the influence of User-Centered Design (UCD) methods in devising solutions to promote interdisciplinary collaboration among researchers. Using a visualization tool, these researchers effectively facilitated research collaborator recommendations and information exchange among researchers based on their publication records. Their investigations employed participatory design techniques and iterative evaluations, resulting in a design that enabled researchers to identify potential scientific collaborators, address scientific challenges, locate relevant literature, discover field experts, and locate research facilities or resources. The authors posited that

UCD methods could pinpoint users' primary challenges and subsequently direct the development process toward suitable tools for supporting research processes.

## User and Data Privacy

Data privacy and protection are two main challenges preventing any tool from gaining users' attention and influencing users' participation [22], [67]. Perrier et al. [68] at the University of Toronto have conducted a mixed-method user study with 28 researchers to identify requirements for RDM tools and services. The authors have concluded that participants have taken the users' data protection very seriously. Hence, Open Science services should provide clarity and guidance on anonymizing data to satisfy ethical prerequisites. Kokolakis [69] and Adler et al. [70] have conveyed a phenomenon of *privacy paradox* and users' behavioral inconsistencies toward sharing information. They found that, despite significant privacy concerns in social, collaborative platforms, people are willing to share information when perceived benefits surpass observed risks. Authors [71] suggest that data-driven analysis can reliably echo actual user perception regarding privacy in a social network platform rather than only self-reported behaviors.

The EU has introduced legal laws for privacy protection, known as the General Data Protection Regulation (GDPR) [72]. Subsequently, Labastida [73] discusses legal requirements for RDM systems to promote research data sharing while adhering to GDPR and FAIR principles. He discusses the common practice of aggregating data or anonymizing before publicly reusing research data. Mosconi et al. [2] debate that FAIR data guidelines can be applied only to a certain extent due to ethical codes imposed on researchers to ensure the anonymity of research subjects. Thus, extensive data anonymization causes the loss of contextual information necessary for research results' reproducibility and data reusability.

Thus, to comply with GDPR rules and respect user and data privacy concerns, OSPs have to collect and analyze anonymized data and not contain any sensitive domain. Accordingly, the suggested data analytics framework in this thesis focuses on studying collective user activities and discovering process models and patterns without needing explicit and individual user information.

## Data-Driven Requirement Engineering

Sedelmaier and Landes define requirement engineering as “*methods and techniques for eliciting, analyzing, documenting and validating requirements*” [74]. The traditional requirement engineering task involves gathering the essential needs of stakeholders and establishing relations to currently available features of a software solution. However, as the number of features and software pieces increases, so does the complexity

of these systems; thus, the interfaces between components within a distributed service quickly get overlooked and result in implicit interdependencies and loss of sight over non-functional or hidden requirements [75].

On the other hand, data-driven requirement engineering promotes an evidence-based analysis of requirements and facilitates the process of decision-making by employing implicit usage data [76]. For example, Hemmati [77] and Moukhi et al. [78] demonstrate the benefits of a data modeling pipeline on top of usage data collected by distributed storage systems. Using a data-driven approach, authors could successfully investigate the system usage and identify bugs without user engagement, potential user needs, and new opportunities to enhance existing features. Additionally, Mohammadi and Heisel [79] highlight the importance and relation of data-driven requirement evaluation to users' perception of system trustworthiness. This becomes important when users expect to trust and rely on OSPs with their sensitive research data. Finally, Dabrowski et al. [80], and Filipowska et al. [81] elaborate on the advantages of studying user experience and goal-oriented analysis by utilizing process modeling techniques.

Likewise, my proposed methodology enables OSPs to explore and discover non-functional and implicit user requirements, such as a study of fall-out activities, discovery of bottlenecks, workflow auditing, and software components' performance analysis while respecting users' privacy concerns.

### 1.2.2 Open Science Underlying Activities

The current trend to emphasize the aspects of *Good Scientific Practice* [82] has continuously related to various commonly mentioned terminologies, but the practical distinction or their relationship to one another is overshadowed in the literature. Hence, this section focuses on the brief definition of each term and how these terms are co-related.

#### Data Management Plan

In recent years, public funding agencies have required DMP as a prerequisite for granting research funding. DMPs demand researchers supply a detailed strategy for data governance, duration, and methods for preserving data and clarifying data anonymization and sharing intentions [2].

#### Research Life Cycle

Research Life Cycle (RLC) is referred to the researchers' stages of study, starting from its research ideation to its completion [83]. Figure 1.2 visualizes the five phases



of RLC, which are inspired by [84]. However, as illustrated, the RLC is not necessarily a cyclic process, and researchers may need to repolish their research idea at any stage of RLC.

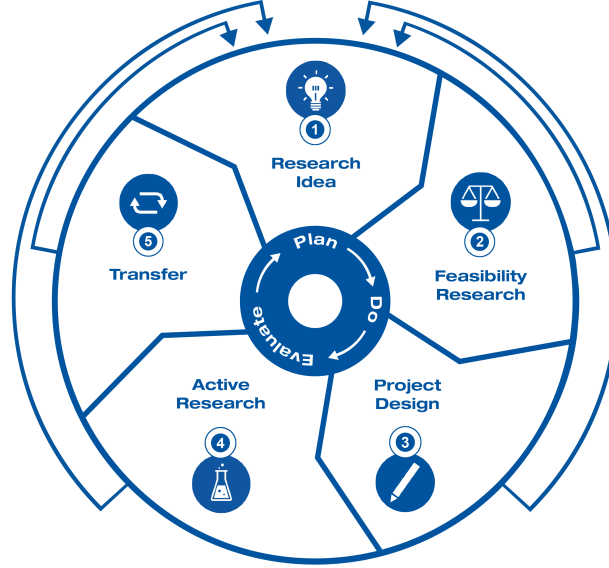


Figure 1.2: Research life cycle inspired by Humphrey [84].

1. **Research Idea:** The initial process of exploring research gaps and developing the hypothesis according to available knowledge in the field.
2. **Feasibility Research:** The activities to evaluate the possibility and advantages of investing resources for a research project. This may include searching for relevant funding, data availability, proposal writing, and potential collaborators.
3. **Project Design:** Specification of project collaborators and the respective work packages. Specifying and preparing the tools and acquaintance necessary to execute research methods and prepare DMPs.
4. **Active Research:** All the crucial exercises for executing a research task. This includes data generation or reusing existing preliminary data to analyze and achieve new knowledge in a field.
5. **Transfer:** The process of documentation and publishing of findings to reach out to scientific peers and get an outside perspective.

Researchers within their RLC are commonly evaluated against several Key Performance Indicators (KPIs) such as the novelty of a study, number of published papers, the reputation of the publisher, conference rankings, H-index, and citations [85].

### Data FAIRness

Data FAIRness refers to high-level 15 fine-granular measurable units for encouraging researchers to increase research data Findability, Accessibility, Interoperability, and Reusability [86]. The FAIR guidelines for scientific data management and stewardship advocate for human and machine-readable data. Various organizations and initiatives such as International Oceanographic Data and Information Exchange (IODE) [87], EOSC [88], and Research Data Alliance (RDA) [89], Nationale Forschungsdaten Infrastruktur (NFDI) [90] contribute toward the definition, awareness, and implementation of FAIR objectives. According to Higman et al., [91], FAIR is not a binary indicator for scientific projects but rather a measurable spectrum for varying degrees of *FAIRness*.

In abstract terms defined by Tanhua et al. [92], *Findability* is associated with pinpointing each data using a unique PID enriched by domain-relevant standardized metadata. *Accessibility* refers to the possibility of retrieving research data by humans and machines using PIDs via standardized communication protocols for authentication and authorizations. The *Interoperability* translates into applying domain-specific standard metadata in the form of vocabularies and ontologies to describe research data to increase the capacity to exchange data between research groups. Lastly, the *Reusability* indicates the integrability and repurposing of research data into other research projects while enough metadata sufficiently describes them.

To develop mechanisms to efficiently provide data FAIRness for scalable cross-disciplinary research projects with vast data volumes, proposals have been made for FAIR Digital Objects (FDO) [93]. These FDO-encapsulated bit sequences are wrapped by relevant metadata to represent informational and operational units. FDOs are then linked and referenced using PIDs [94].

### Research Data Management

Corrall [95] and Cox et al. [96] have defined RDM as all activities in RLC involving the data from its production to reusing, such as planning, production, analyzing, Storage, maintaining the data accessibility and preparing for reuse. The RDLC is often used with RDM to suggest to researchers the course of underlying interconnected processes that produce new research data and highlight the evolution of science [97]. Mainly, RDLC is referred to as a representation of stages and phases of RDM. The RDM data governance emphasizes research data annotations with metadata and contextual information (data provenance) required to share with others to improve the reproducibility and reliability of scientific experiments [6]. Therefore, data provenance is not only a final research data but also capturing and recording the process of data creation [98], [99]. Overall, RDM greatly enriches the knowledge transfer process for scientific collaborations and enables further secondary data reuse.

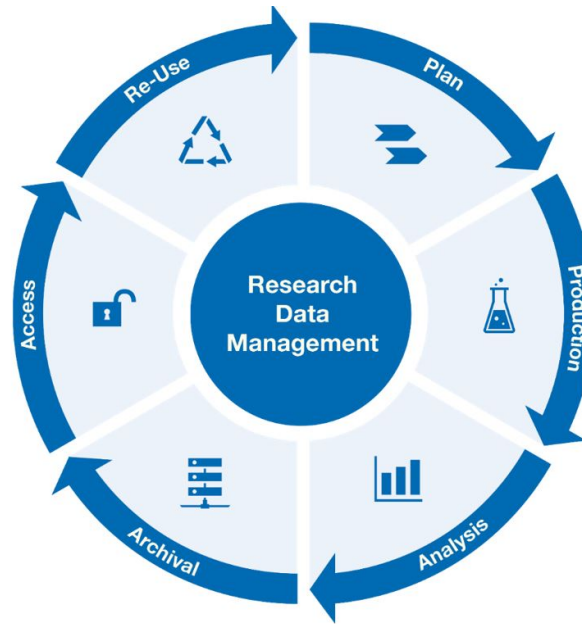


Figure 1.3: Phases of research data management.

Figure 1.3 illustrate the RDM phases that a research project goes through within an OSP:

- **Plan:** It determines the organizational structure of a research project within an OSP alongside a mutual agreement on data standards that all project collaborators need to comply with.
- **Production:** This phase deals with metadata management and ensuring that produced and collected research data are accompanied by corresponding specialized metadata.
- **Analysis:** The Analysis phase deals with identifying patterns among existing research data and supports modification of research data to achieve desired results.
- **Archival:** This phase ensures the preservation of research data for the long term while preventing further modification of research data and metadata.
- **Access:** Besides providing access to the data or metadata when permitted, this phase allows verifying the eligibility of researchers who would like to access research data when needed.
- **Reuse:** Principally, data reuse is evidence of sufficient contextual information on research data to guarantee data re-discovery and preparation for future reuse.

### 1.2.3 Process-Aware Software Analysis in Distributed Services

In order to satisfy the requirements of DMPs, researchers often use heterogeneous tools and services [100]. Moreover, university services are decentralized to increase computing power, scalability, maintainability, security, rapid response, and diversification [101]. However, collaborative research projects entail challenges for data management and create opportunities for OSPs to address these challenges according to RDM principles [2]. For instance, Coscine, as an OSP, has several distributed services integrated to fulfill the technical requirements for achieving RDM goals. Another challenge in a dispersed environment is producing and collecting data that can be utilized for further studies and modeling of process-aware RDM activities [100]. Unfortunately, most techniques do not support the issue of correlation in distributed systems in a scalable fashion, and they focus on discovering a control-flow model rather than the interactivity of software components, resulting in research gaps. Therefore, this section is a deep dive into the criteria for analyzing distributed systems to assist the research objectives and introduce a methodology for appropriate extraction and collection of data for later modeling tasks. The work in this section is partially based on previously published research by Yazdi et al. [24], [25].

#### Analysis of Distributed Systems:

To understand a distributed system's behavior, looking into source code and observing the intercommunication between different components and dynamic bindings within a system is incomprehensible. Hence, Table 1.1 compares and provides a bird's-eye view of different approaches used to analyze distributed systems. This work introduces multiple criteria such as *Communication Type*, *Correlation of Distributed Events*, *Application Layer*, *Environment*, *Information Source*, *Granularity*, *Performance Analysis*, and *Target Model* to facilitate the comparison of different proposed approaches and thus position the contributions within state of the art accordingly.

**Information Source:** *It is the strategy used to retrieve dynamic data.* Similar to the approach discussed in Chapter 3, most of the suggested methods [102]–[104] use some sort of instrumentation or alternation of existing code to generate the traces of information required for analysis. For example, the method introduced by Beschastnikh et al. [105] suggests re-purposing and adopting previously generated logs to acquire the necessary data. Ackermann et al. [106] and Moe et al. [107] have considered monitoring and intercepting low-level network packets between clients and servers as a source of information.

**Application Layer:** *The constraint of the proposed approach is to be able to reverse engineer a distributed system in a particular programming language.* Most of the related studies focus on the instrumentation of the Java programming language [102],

[103], [105], [108]. Furthermore, Ackermann et al. methodology [106] relies on the TCP/IP application layer, and Moe et al. [107] targets the COBRA translation layer and acts as middleware.

Table 1.1: Bird’s-eye view on different approaches for analyzing distributed systems.

<i>Author</i>	<i>Info. Source</i>	<i>App. Layer</i>	<i>Corr. of Events</i>	<i>Sequence Order</i>
Ackermann et al. [106]	Network Packets	TCP/IP	Network Packets	Dependent
Beschastnikh et al. [105]	Instrumentation	Java	Comm. Channels.	Dependent
Briand et al. [102]	Instrumentation	Java	Comm. Channels	Dependent
Van Hoorn et al. [103]	Instrumentation	Java	No Correlation	Independent
Leemans et al. [104]	Instrumentation	Independent	Comm. Channels	Dependent
Moe et al. [107]	Network Interceptor	COBRA	No Correlation	Dependent
Salah et al. [108]	Provided Log	Java	Comm. Channels	Dependent

<i>Author</i>	<i>Comm. Type</i>	<i>Per. Ind.</i>	<i>Granularity</i>	<i>Enviro.</i>	<i>Target Model</i>
Ackermann et al. [106]	Single Thread	No	Components	Real	UML
Beschastnikh et al. [105]	Multi Thread	No	Components	Test	CFSM
Briand et al. [102]	Single Thread	No	Control-Flow	Test	UML
Van Hoorn et al. [103]	Multi Thread	No	Components	Real	Monitor Log
Leemans et al. [104]	Single Thread	Yes	External Interface	Real	Process Model
Moe et al. [107]	Multi Thread	Yes	Varied	Real	LQN
Salah et al. [108]	Multi Thread	No	Components	Test	UML

**Correlation of Distributed Events:** *The methodology to correlate executed events in distributed systems.* The authors of [102], [104], [105], [108], are using extra communication channels to inspect and indicate the correlation of events. Instead, Ackermann et al. [106] focuses on network packets transmitted between the sender and receiver.

**Sequence Order:** *The order of execution of events within a process.* For the analysis of traces, except the approach suggested by Van Hoorn et al. [103], authors have suggested techniques to reverse engineer the run-time software execution with respect to the order of events occurring during a process. Therefore, a trace variant directly influences the target model.

**Communication Type:** *Communication type defines the execution of tasks of a process in either single thread or multi-threaded fashion.* It refers to the approach’s capacity to interpret multiple software execution processes simultaneously. In [102], [104], [106], the proposed techniques can analyze the single-threaded processes, while others [103], [105], [107], [108] facilitate the evaluating of multi-threaded software execution.

**Performance Indicator:** *Focuses on the possibility of using a technique for analyzing the software run-time performance in a distributed environment.* Only a few existing techniques [104], [107] facilitating the analysis of software execution performance and hence, help to find bottlenecks in a system.

**Granularity:** *It is the level of detail of information acquired by an approach.* In [105]–[108], authors have focused on the behavior captured at the software components level. On the other hand, authors for [102] focus on deep, low-level software control-flow information. In [104], authors have attempted to capture the user requests and trace the software execution cycle to respond to those user requests. The suggested approach in this thesis, on the other hand, can adopt its data granularity based on the goal of a project. It can support acquiring low-level (software components) to high-level (intercommunication of services) data from a distributed environment.

**Environment:** *The environment in which a method is built for.* In order to be able to replicate a reverse engineering approach for a distributed system, it is essential to replicate that method in real-life settings. However, the authors in [102], [105], [108] have only examined their proposed solutions in a lab/test environment. This is especially critical in an expanding distributed systems, as the scalability of a solution depends on being able to handle real-life software execution processes without further readjustments.

**Target Model:** *The model that an approach can produce.* The strategies introduced in [102], [106], [108] are aiming for a generation of UML sequence diagrams. Beschastnikh et al. [105] use Communicating Finite State machines as their targeting model. Moe et al. [107] use the Layered Queuing Networks model to study tradeoffs in software architecture to predict the effects of changes to architecture before the actual implementation. Similar to the work done by Leemans et al. [104], this thesis also focuses on data and process models to precisely express the mapping of traces and the expected software behaviors.

## Summary

Despite the various studies in the area of analysis of distributed systems, many of them are focused on a lab setting and the generation of UML sequence diagrams. However, in a real-life scenario, the suggested approach for gathering data should have a minimal impact on a system’s performance. Additionally, the proposed approaches generate a non-dynamic data granularity for analysis purposes, some are too detailed, and some are very high level. In an Open Science distributed system environment, it is essential to propose a solution that can acquire the right level of information with respect to the requirements at hand. Hence, the current approaches lack the possibility of scalability and adapting the level of data. Moreover, every microsystem in a distributed environment may have a different programming language, so one should focus on

language-independent solutions. As OSPs are continuously implemented and operating, the suggested solutions should be non-intrusive to pre-existing business logic.

To summarize, the main challenge for analyzing distributed systems is to introduce a method for generating data to enable discovering the interactivity aspect of a distributed systems and the *intercommunication* between different *system components*.

### 1.2.4 Data Abstraction for Business Process Discovery

In order to study user dynamics and track the state of operational RDM processes within a distributed system, process mining methods are utilized, allowing for the extraction of process models [22]. However, handling the immense volume of data generated by distributed systems is akin to drinking from a fire hose, necessitating the use of an appropriate data abstraction technique to facilitate the identification of structured RDM operations. As previously discussed, the primary tasks in process mining are *discovery* and *conformance* checking [54]. The goal of process *discovery* is to uncover a process model that accurately reflects real-world behavior based on actual data. Numerous process discovery algorithms exist, including IM, Alpha Miner, Fuzzy Miner, and Heuristic Miner. *Conformance* checking involves comparing a process model with a data log to determine similarities and differences, allowing for the assessment of the discovered model's fitness [109]. Several software tools, including ProM [110], Disco [111], and Rapidminer [112], can aid in process analysis tasks.

Complex infrastructure characterizes distributed systems, where a user request may necessitate the involvement of numerous software components and microservices for task execution. This complexity often leads to a  $n:m$  relationship between Server-Side events and Client-Side events that are excessively fine-grained. Applying process discovery to these low-level data logs frequently yields an unstructured process model known as “Spaghetti” [54]. As a result, several data log abstraction techniques have been proposed to convert fine-grained (low-level) data logs into more abstract (high-level) representations. However, the primary challenge lies in determining the optimal abstraction level that is comprehensible to domain experts and accurately portrays actual business activities.

Recent research has focused on data abstraction techniques for process discovery purposes [113]. This section classifies the various criteria for developing any abstraction method and discusses the relevant literature.

#### Criteria for Data Abstraction Techniques

**Grouping Strategy:** *The grouping strategy pertains to the learning techniques employed to transform low-level data logs into a higher level of granularity. Two*

primary categories of abstraction operations exist: *aggregation* and *elimination* [114]. The data *elimination* operation adopts a naive approach by excluding inconsequential events, which can result in underfitting models that lack information about the eliminated events. The *aggregation* operation creates meaningful groups of events containing a mix of relatively unimportant events. There are two subcategories of the *aggregation* operation: *unsupervised* and *supervised* learning methods.

The *Unsupervised* abstraction methods utilize fully automated techniques for event grouping and often yield less accurate high-level models. These methods generate labels by concatenating activities or making assumptions without incorporating domain knowledge. For example, Mannhardt and Tax [115] employ Local Process Model discovery to automatically identify frequent activity patterns in the process model, while De Leoni and Dündar [116] use clustering methods to group activities based on occurrence frequency and relabel corresponding groups using cluster centroids. Unsupervised learning methods are advantageous when domain knowledge is unavailable or obtaining additional information is impractical.

In contrast, the *supervised* learning methods are the most prevalent and widely used techniques for simplifying process models. These methods depend on some form of external or supplementary information to group and relabel data logs. Numerous related works [117]–[120] employ *supervised* learning techniques, either by capturing additional information or by creating a training dataset to annotate fine-grained data logs.

Table 1.2: The work of literature and classification based on abstraction techniques.

Author	Grouping		Input Data		Event Perspective	Mapping Relationship	Internal Abstraction	
	Su.	Un.	Co.	Di.			Pr.	De.
Begicheva and Lomazova [117]	✓		✓		Sequential	n:1	✓	
Mannhardt et al. [118]	✓		✓		Sequential	n:1		✓
Tax et al. [119]	✓		✓		Sequential	n:m	✓	
Liu et al. [120]	✓		✓		Sequential	n:1		✓
De Leoni and Dündar [116]		✓	✓	✓	Non-Sequential	n:1		✓
Mannhardt and Tax [115]		✓	✓		Sequential	n:1		✓

Author	Validity	Quality Indicator	Target Domain
Begicheva and Lomazova [117]	Formal	Fitness	Acyclic Cases
Mannhardt et al. [118]	Real-Life	Fitness & Matching Error	Information Systems
Tax et al. [119]	Synthetic	Levenshtein Distance	Parallel Activities
Liu et al. [120]	Synthetic	Fitness & Precision & Generalization	Information Systems
De Leoni and Dündar [116]	Real-Life	Fitness	Information Systems
Mannhardt and Tax [115]	Real-Life	Fitness	Automatic Pattern Discovery



**Input Data:** *Input Data refers to the type of data, whether discrete or continuous events.* Discrete events consist of an infinite set of values (e.g., categorical or real values), while continuous events are a data set that can assume any value (e.g., activity names). Data logs may be discrete event sequences or continuous due to activity execution. Most techniques reviewed in the literature utilize some form of continuous activity names for analysis. However, De Leoni and Dündar [116] vectorize continuous data sequences and convert the data log into a discrete data format. This discrete data facilitates the implementation of data mining algorithms, such as analyzing the frequency of activity occurrences within a data log.

**Event Perspective:** *Event Perspective indicates the impact of the data log order on the abstraction technique.* Most procedures in the literature only consider the order of the event sequence as it appears in low-level events. In contrast, the method proposed by De Leoni and Dündar [116] employs the non-sequential order of events with unsupervised abstraction methods.

**Mapping Relationship:** *The mapping relationship between instances of low-level and higher-level activities.* There are  $1:1$ ,  $n:1$ , and  $n:m$  mapping relationships. In the case of  $1:1$  mapping, no abstraction can be performed, as the granularity level remains unchanged, meaning that each event corresponds to only one activity. Most literature focuses on  $n:1$  mapping, also known as shared functionality, where multiple higher-level activities may correspond to each low-level data log. In software systems, user activities often do not directly correspond to a recognizable server-side event, resulting in a  $n:m$  mapping relationship.

**Internal Abstraction Modeling:** *Internal Abstraction Modeling emphasizes using probabilistic or deterministic factors to determine event sequence distributions.* Most literature concentrates on deterministic approaches. However, Begicheva and Lomazova [117] also apply the theory of regions to categorize related high-level activities, and Tax et al. [119] utilize Conditional Random Fields (CRF) as a probabilistic factor for event sequence relabeling.

**Validity:** *Validity refers to the evaluation procedure employed to demonstrate the robustness of a proposed technique.* While Begicheva and Lomazova [117] rely solely on formal mathematical representation, all other works examine their approaches using real-life or synthetic event logs. However, only Mannhardt et al. [118] evaluate their technique by implementing it in existing infrastructure. This is crucial because real-life event logs may not encompass actual field challenges, such as noise, outliers, or missing data in event logs.

**Quality Indicator:** *These indicators contribute to the quality dimensions of the discovered abstracted model.* Typically, discovered models are evaluated based on metrics like fitness, precision, and generalization, which provide further insights into the alterations made by the event log abstraction process [54]. According to Buijs et al., [58], fitness measures how well the discovered model can accurately reproduce the

cases recorded in the log. Precision calculates the proportion of behavior the model allows that is not observed in the event log. Finally, generalization estimates the extent to which the resulting model can reproduce the future behavior of the process.

Based on my literature review, only Liu et al. [120] successfully report their approach's quality using fitness, precision, and generalization. The remaining researchers either report model fitness as the sole quality factor or support their work with additional measures such as Matching Error as a threshold for excluding unreliable matches between fine-grained event logs and high-level activities [118]. However, Tax et al. [119] exclusively rely on the Levenshtein similarity distance as a degree to express the closeness of two traces from different granularity levels.

**Target Domain:** *The primary application domain a technique focuses on.* Regrettably, much of the literature is not generic enough and targets a specific application domain. For example, Begicheva and Lomazova's suggested solution [117] only aims at non-recursive activities (acyclic cases). The technique provided by Tax et al. [119] is intended to abstract only parallel activities with any execution order. The automatic pattern discovery and clustering of activities are the focus of the method discussed by Mannhardt and Tax [115]. Other literature under study seems to concentrate on the analysis of information systems such as digital whiteboard software [118], user behavior analysis on software systems [120], and analysis of website visit frequency [116].

### Summary

While Begicheva and Lomazova's work [117] focuses on acyclic cases, they only employ a formal representation to illustrate handling data duplication, overlooking stuttering sub-sequences of activities. Furthermore, the methods suggested by Mannhardt et al. [118] and Liu et al. [120] depend on manually labeling low-level events to high-level activities using interviews and observations and exclude traces with missing or unknown events. Tax et al.'s method [119] can handle parallel activities but requires extensive domain knowledge. The approaches of De Leoni and Dündar [116] and Mannhardt and Tax [115] leverage unsupervised learning techniques for automatically relabeling abstract traces. While they generate new labels for higher-level activities using cluster centroids or label concatenations, domain experts' acceptance of this technology remains unexplored. Mannhardt and Tax [115] use Local Process Models and automatic pattern discovery techniques to detect activity start and end points but are constrained by a fixed number of candidate process models based on event frequency ranking.

Most authors have generally employed the IM algorithm for discovering process models. The IM algorithm addresses infrequent behaviors while maintaining the soundness of the discovered model [121]. In addition, various Petri Nets notations

have been used to represent the discovered models. Consequently, Section 4.1 suggests an approach suitable for abstracting data logs obtained from the distributed infrastructure of an OSP.

## 1.3 Research Outline

According to discussed motivation and the related research as a foundation, this work discusses the fundamentals to prepare the existing decentralized services involved in OSPs to meet the rising demand of researchers for supporting data management processes toward discovering new insights. Apart from discussing RDM's typical system architecture landscape, this research provides a theoretical basis for several data analytics strategies covering a comprehensive spectrum of data extraction, preparation, knowledge discovery, and recommender systems. Moreover, I assess each building block using real-world case studies to back the external validity and practicality of the suggested methods.

### 1.3.1 Research Questions

As mentioned earlier in Section 1.2.1, the investigation of RDM processes is a study of the overlap of multiple research disciplines, where *Data and Process Science* provides the means for the discovery of new information from *Distributed Systems*, supporting the *RDM* related activities. Thus, this thesis focuses on answering the main Research Question (RQ):

**Main RQ:** How to enable discovering and enhancing RDM practices via modeling the underlying activities?

With respect to the literature review, it is clear that FAIR principles play a crucial role in achieving RDM goals and advancing the overall research performance, but a few studies have been published on the actual monitoring of RDM processes and how they are adopted and used. Thus, for an effective RDM, authors in [122]–[124] call for investigation of institutional data management frameworks that couple with the process of integration of data sources with their metadata.

This thesis split the main RQ into five data-driven sub-research questions listed in Figure 1.4 in conjunction with existing research gaps illustrated in Figure 1.1.

**RQ 1:** How to overcome the challenges of acquiring datasets suitable for process and data analytics from continuously evolving RDM services while maintaining data quality standards?

RDM services largely benefit from the advantages of distributed systems; however, dynamic data capturing from a growing distributed environment is challenging due to

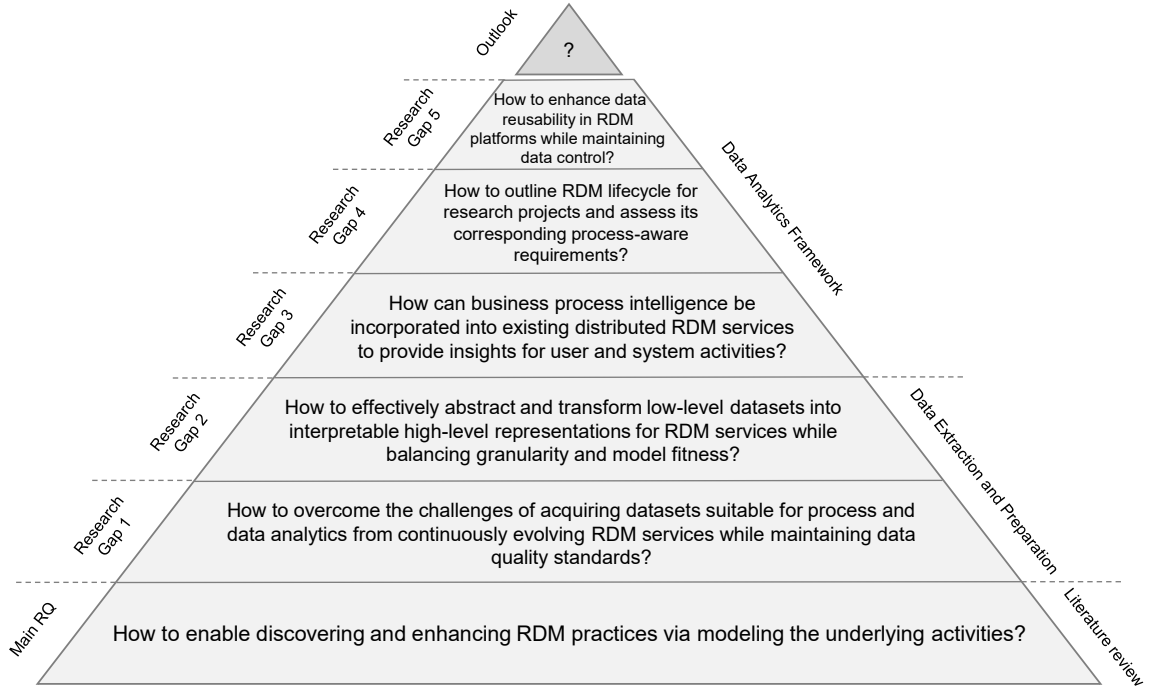


Figure 1.4: Unexplored research gaps and associated research questions.

its complex execution of heterogeneous components across a network and continuous feature improvements. In order to capture events for the purpose of analyzing underlying process-aware RDM activities and enable machine learning studies, one needs to introduce an appropriate approach for acquiring data constructed from cross-applications and independent of programming language with minimum implementation efforts. Answering this research question will lay the foundation for scalable and maintainable suitable datasets and event data while ensuring quality standards such as completeness, timeliness, validity, uniqueness, consistency, and accuracy. The resulting infrastructure should also comply with European and German data protection laws to safeguard user privacy.

**RQ 2:** How to effectively abstract and transform low-level datasets into interpretable high-level representations for RDM services while balancing granularity and model fitness?

The institutional OSPs provide distributed software solutions to support RDM. However, this results in a heterogeneous and distributed IT infrastructure with a complex and unstructured landscape of user processes. Furthermore, the data gathered from these disparate services include descriptive process variations that are too detailed and complex for domain experts to understand. Thus, the answer to this research question should propose a methodology that deals with unordered event data while

balancing the data granularity with respect to process models' fitness to ensure the discovery of structured and analyzable process models.

**RQ 3:** How can business process intelligence be incorporated into existing distributed RDM services to provide insights for user and system activities?

Despite the advantages of distributed environment for sharing tasks and resources, an essential overarching overview of the execution of microservices involved in RDM operations is quickly overshadowed due to its architectural complexity. Answering this research question aims to investigate incorporating BPI algorithms across available services to deliver insights into hierarchical control-flow modeling of microservices and software components supporting the Open Science processes. The resulting findings should efficiently enable reverse-engineering the non-trivial process models in a decentralized IT system landscape and provide a means to generate RDM process models based on user activities.

**RQ 4:** How to outline the RDM lifecycle for research projects and assess its corresponding process-aware requirements in collaborative scientific platforms?

As mentioned in Section 1.2.2, the RDM principles and activities are defined by stakeholders from national and international initiatives, and the study of actual practices of researchers applying RDM principles is a research gap. This research question aims to utilize the gathered event data to bridge the knowledge gap regarding the actual practices of researchers within OSPs and RDM guidelines. Thus, the resulting finding should enable visualizing descriptive RDM stages for research projects while identifying RDM-related requirements using process-aware activities. Furthermore, outlining the RDM operations will encourage troubleshooting the potential barriers for each RDM phase and provide self-awareness regarding the current collective RDM activities within a research project.

**RQ 5:** How can recommender systems enhance research data discoverability and reusability in RDM Platforms while maintaining user control over data artifacts?

The ultimate goal for RDM principles and services is to facilitate and encourage researchers to reuse data artifacts and consequently save time in reproducing peers' research results. However, researchers are often reluctant to publish data openly due to uncertainty about legal consequences or possible misuse; hence the potential loss of control over data outweighs the gain, resulting in oversight of research data reusability. Answering this research question should bring forward a comprehensive method for enabling the reusability of research repositories within an OSP without compromising sensitive information by employing available datasets such as user activities or knowledge graphs.

### 1.3.2 Course of Research

In order to answer the research questions, the course of research is inspired by CRISP-DM (CROSS-Industry Standard Process for Data Mining) [125] and L\*lifecycle [54] research methodologies.

Considering that applying data analytics projects in practice is not a trivial task, it has been necessary to run each stage iteratively to steer to an optimal conclusion. Figure 1.5 presents the research methodology and its stages. There are five steps (rows) repeated within every stage (column).

	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5
	Planning & Business Understanding	Data Analysis	Data Preparation	Modeling	Evaluation
<b>Awareness of the Problems</b>	Acquiring an Overview of RDM Services	Possible Unsuitability of Data for Process Modeling	Fine-Grained Unstructured & Complex Event Data	Identification of Suitable Research Projects in an OSP	Best Practices to Outline RDM & Maturity of an OSP
<b>Justify Methodology</b>	Hybrid Data Collection using OAuth & Client-Side Logger as a Training Set	Hierarchical Process Discovery Using Directed Flow Graphs	Iterative Semi-Supervised Event Log Abstraction Using Fitness & PCC	Using Token Replay Over Petri-net to Discover BPI	Content-Based Filtering using Pearson/Cosine Similarities & Trace Alignment Checking
<b>Development</b>	Implementation of Event listeners	Running Suitable P.M. Algorithms on the Event Logs	Implementation of Pre-Processing Pipelines	Implementation of DA4RDM Web-based Framework for Post-Processing	Visualization of RDM in DA4RDM Python Package
<b>Evaluation</b>	Assessment of Data Quality Standards	Distributed IT Services of the University	Data Refinement for SimpleArchive & Metadata Manager	Cosine Resource Object Analysis	Discovery of RDM for Research Projects & Control-flow Requirement Analysis
<b>Conclusion</b>	Preparing Data for Further Analysis	Outlining the Interdependencies of RDM Services with Other Microservices within a Distributed Environment	Abstracting Event Logs to Desired Level while Keeping the Fitness	Recommending Research Data Repositories using Recommender Systems	Outlining RDM Activities for Research Projects & Accessing RDM process-Aware Requirements

Figure 1.5: A representation of the course of research.

The findings from every stage are used as input for the next stage. The stages within this research methodology were planned to gradually shape the research project and answer the research problems as they mature and evolve. A description of each stage is as follows:

*Stage 1) Planning and business understanding:*

Dedicated to comprehending the domain knowledge, available services, and identification of research gaps. The results of this stage are a set of research questions and an awareness of software architecture limitations in place.

*Stage 2) Data Analysis:*

This stage investigates the nature of available data for further data-driven studies and acknowledges the potential challenges of utilizing the data. Thus, this stage locates and enables the exploration of data.

*Stage 3) Data preparation:*

Before running any data analysis task, structured event data must be obtained in this stage to enable further data analysis techniques using pre-processing pipelines.

*Stage 4) Modeling:*

The findings from the previous step within a control flow model have to get enriched and integrated with additional perspectives to help better understand the as-is processes. In addition, this stage supports us in verifying prescriptive models with actual data via user journey modeling.

*Stage 5) Evaluation:*

It is the stage when an Open Science system can detect deviations in process-oriented operations and determine stages of RDM for research projects using real-world data.

### 1.3.3 Outline of the Thesis

Chapter 2 presents a generic landscape of Open Science related services and discusses a reference architecture for distributed services supporting the process-aware RDM activities. Furthermore, this Chapter describes RDM systems under study utilized to run case studies and evaluate the external validity of proposed methodologies.

Chapter 3 introduces the methodologies required for extracting reliable and suitable data from OSPs to answer **RQ 1**. It discusses the quality measures and the standards essential for modeling the RDM processes. This Chapter introduces several data extraction techniques that were applied and elaborates on the advantages and disadvantages of each technique. Moreover, concerning the discoveries and challenges of acquiring data from OSPs, this thesis proposes a novel pivotal method for gathering reliable data.

Chapter 4 elaborates on a set of methodologies for modeling RDM operations toward addressing **RQ 2-5**. This Chapter introduces a web-based data modeling framework that commences for analyzing and modeling tasks. Besides introducing the fundamental technologies used to implement the modeling framework, this Chapter justifies the overall methodology by an exemplary set of interfaces (data abstraction, user journey mapping, outlining of RDM phases, and recommending data objects) that are separately introduced and evaluated. Finally, the overarching methodology should be applied to OSPs based on the theoretical foundations and lessons learned to model process-aware RDM operations.

Chapter 5 discusses the findings and evaluates suggested approaches by applying case studies featuring RDM processes using real-world data and scenarios. The assessment of data modeling implementations is then examined using qualitative

and quantitative metrics to determine the reliability and scalability of approaches discussed in Chapter 4.

Lastly, Chapter 6 will conclude the thesis by answering the research questions and summarizing the methodologies supporting the discovery of underlying process-aware RDM activities. Moreover, this Chapter presents the thesis' contributions to the body of Open Science and positions itself in an applied interdisciplinary research context. It also elaborates on the potential future works and suggests an outlook by utilizing the results and lessons learned.



## 2 Landscape of RDM Services

The rise of data-driven collaborative research projects has underlined the importance of Open Science objectives and led to increasing demand for suitable software infrastructures to fulfill FAIR data principles' requirements and boost the research data discoverability, reproducibility, and reusability [126]. According to Hofeditz et al., [127], despite the growing number of available technical solutions, scientists have been reluctant to employ them due to the complexity of integrating RDM practices into everyday scientific workflows and a lack of knowledge on implementing IT services that support researchers in every phase of RDM lifecycle.

This Chapter focuses on an overview of services that can be adopted in the long tail of RDM operations. Firstly, a pragmatic comparative study of fifteen RDM services for the goal of adopting the RDM requirements is discussed. Then, Section 2.2 conceptualizes a reference RDM software architecture model for OSPs and elaborates on how they profit from distributed services environment to satisfy various RDM activities.

The last Section introduces three RDM services as a set of System Under Study (SUS). It demonstrates the alignment of each subjected SUS with respect to the reference RDM software architecture model and the applicability of suggested methods in this thesis to OSPs.

### 2.1 Comparison of RDM Service Providers

The starting point for studying underlying RDM processes within OSPs is to analyze the existing IT landscape and generate a reference software architecture model that can later be utilized and enhanced. Typically, the IT systems employed by researchers to facilitate RDM activities are situated in heterogeneous and independent infrastructural software services. For example, a survey study on researchers' actual RDM practices by Mosconi et al. [2] discovered researchers' extensive usage of file servers or file-sharing systems such as Google Drive, Dropbox, and Sciebo, but they lack support for RDM workflows or enforcing policies. In contrast, despite an expressed demand for Metadata management, the abovementioned systems do not support any structured metadata during a research process. Moreover, Mosconi found a limited usage of several decentralized and detached discipline-specific services to satisfy RDM

requirements due to the complexity of maintaining IT systems, despite a growing number of competent IT services that adequately incorporate the RDM lifecycle.

The following Section compares 15 web services that are either specialized or utilized in/as RDM-related activities. It briefly introduces each service and then discusses each tool's applicability and technical feasibility for RDM-related processes.

**Research Data Management Organizer (RDMO)** [128]: A web-based tool funded by DFG to assist principal investigators and stakeholders in creating DMPs according to funding agencies' requirements. Through RDMO, users can define research projects and partners, utilize grant proposal templates to fill into various funders' required questionnaires, and export DMPs as ready text documents.

**SimpleArchive** [129]: A tool developed by RWTH Aachen University to facilitate long-tail preservation of research data using tape archives and associating PIDs to each entry. It allows researchers to specify retention periods of a minimum of 10 years to maintain archived files and link research data to publications via PIDs.

**Metadata Manager** [130]: A tool developed by RWTH Aachen University to document metadata for research data by defining graph-based discipline-specific metadata schemas and receiving PIDs. It can also record references to data artifacts regardless of data storage location.

**Coscine** [131]: A collaborative scientific integration environment developed by RWTH Aachen University providing access to research data and the corresponding structured metadata. Users of this platform can mirror their organizational structure by creating projects and sub-project and prescribing customized, discipline-oriented metadata schemata. In addition, Coscine supplies PIDs for research data regardless of storage locations and ensures metadata collections while adding new resource entries.

**Zenodo** [132]: It is an open repository created by the European OpenAIRE program to facilitate storing digital artifacts and linking research papers to data sets. Furthermore, researchers can cite their data by minting Digital Object Identifier (DOI) for each submission and assigning suitable licenses.

**OSF** [133]: Open Science Framework is a collaboration tool developed by a non-profit organization that connects publications to research data located in remote repositories such as Dropbox and Google Drive in project dashboards. In addition, it allows for creating structured projects with controlled access rights and connections to other tools such as Google Scholar, ORCID, and Github.

**Sciebo-RDS** [134]: is a middleware implemented by the University of Münster, designed as a layer between existing research data management services such as Zenodo or RDMO to cloud-based repositories like Sciebo or OwnCloud. The goal of Sciebo-RDS is to allow the indexing of research data with respect to their metadata

by connecting repositories to data management services that support taxonomies and capturing of metadata.

**eLabFTW (ELN)** [135]: Electronic Lab Notebook (ELN), originally developed by a researcher as a logbook for lab experimentations, enables researchers to capture and track laboratory experiments and processes to be able to reproduce test results. The eLabFTW is an ELN to create logbooks for test samples. eLabFTW allows researchers to design their experiments and define custom-made metadata schema.

**CKAN** [136]: CKAN is an open-source package tool maintained by the Open Knowledge Foundation for data-heavy systems, enabling access and sharing of data collections. The scientific community adapts CKAN for RDM purposes by implementing extensions to handle graph-based metadata management and producing DOIs as persistent identifiers for research data.

**Nextcloud** [137]: Developed by a German private company, brings together file server services and social network collaborative group work functionalities. It benefits from modular architecture, and its open-source community can publish its new extensions via the manufacturer platform. Furthermore, the facilities such as data versioning and indexing using Elasticsearch make it desirable for some research groups where metadata schema can not be planned in advance, but there is a need for a version control system integrated with tools such as video calls and task manager.

**Dataverse** [138]: It is a web application created by Institute for Quantitative Social Science at Harvard University to share, cite and explore research data. Dataverse allows researchers to log in using their Shibboleth identity provider or other OAuth-enabled platforms. In addition, researchers can connect their publications with datasets using DOIs, and its database provides data versioning to facilitate exploring data collections.

**EUDAT CDI** [139]: It is a Collaborative Digital Infrastructure sustained by 20 European research organizations, including various data management services. EUDAT has split its services into seven specialized group works and packages. B2FIND is a service for data and metadata discovery. B2SAFE provides data management and preservation, B2SHARE is responsible for data storage and publishing research data, B2DROP supports researchers with synchronization and sharing of data with local repositories, B2NOTE enables users to create annotations on research data, B2ACCESS empowers the identity management and authorization service, and B2HANDLE registers research data with PIDs.

**DSpace** [140]: It is a repository platform that facilitates easy ingestion of research data and metadata corresponding to the Dublin Core standard. DSpace issues DOI as trustworthy persistent identifiers for data but only provides limited prefixed metadata sets to describe research datasets.

**Figshare** [141]: Founded by a private British company is a repository platform promoting sharing of research data under a Creative Commons license. Although DOIs are issued for research data, Figshare does not use this platform to track data provenance. Furthermore, metadata schemas can only be configured via system administration per system instance, resulting in a technical barrier for researchers to define their own custom-made and specialized metadata schemas.

**GitLab** [142]: It is a distributed version control repository manager similar to GitHub. GitLab provides a toolset for task managers, activity trackers, and defining continuous integration and deployment pipelines ( CI/CD ) to build, automatically test, and deploy applications.

### 2.1.1 Comparison of Services Based on RDM Lifecycle

This Section presents a qualitative comparison of RDM tools mentioned earlier concerning the RDM phases defined in Section 1.2.2 and, with the help of Table 2.1, examines the tools that can satisfy corresponding RDM-specific requirements.

**Planning:** Coscine, OSF, eLabFTW (ELN), and RDMO are the only services providing the infrastructure to define project structures according to mutual agreements of project members. Even though RDMO focuses on DMPs, it still helps specify the organizational structure of research projects. By utilizing these services, scientists can configure research projects and create resources with the necessary metadata schema and the necessary resource quota.

**Production:** Except for RDMO, SimpleArchive, Metadata Tool, and Sciebo-RDS, all other services support researchers collecting research data with different levels of metadata granularity. For example, SimpleArchive encourages researchers to provide metadata for their archived research data by redirecting them to the Metadata Tool, where metadata can be captured for research data. Thus, as individual entities, neither SimpleArchive nor Metadata Tool can be considered contributing services to the Production phase.

**Analysis:** Most services under investigation provide means to modify research data and metadata, but only Coscine, Sciebo-RDS, and GitLab support analyzing research data via their metainformation. Researchers can benefit from GitLab Runner pipelines to analyze their existing data for patterns. Coscine and Sciebo-RDS actively prepare and develop their services to analyze research data by digesting implicit and explicit metadata to build recommender systems. Indeed, it is clear that RDM services lack additional services enabling researchers to find new knowledge without maintaining a local copy of research data and employing stand-alone tools for further analysis.

**Archival:** Despite DMPs' demand for a data-archival plan, besides SimpleArchive and Coscine, no other services ensure researchers with long-term data preservation

Table 2.1: Service comparison in relation to RDM phases.

Service Name	Planning	Production	Analysis	Archival	Access	Reuse
RDMO	✓					
SimpleArchive				✓	✓	
Metadata Manager					✓	
Coscine	✓	✓	✓	✓	✓	✓
Zenodo		✓			✓	
OSF	✓	✓			✓	
Sciebo - RDS			✓		✓	
eLabFTW (ELN)	✓	✓			✓	
CKAN		✓			✓	
Nextcloud		✓			✓	
Dataverse		✓			✓	
EUDAT		✓			✓	
Dspace		✓			✓	
Figshare		✓			✓	
GitLab		✓	✓		✓	

and prevent further data and metadata modifications after archiving a resource. Although services like GitLab offers processes for repository archival, the archived research data are only accessible for the duration of the service lifecycle.

**Access:** To provide access to data or metadata, except RDMO, all services support research data openness and check the eligibility of user attempts to access research data. Thus researchers can define restrictive access boundaries for a spectrum of users who potentially could access research data.

**Reuse:** According to the definition of reusability of research data, currently, only Coscine has the infrastructure that genuinely supports the reusability of research data by issuing PIDs to ensure data findability and capturing extensive discipline-oriented contextual information on data artifacts. All other services in the field either do not issue persistent identifiers or do not collect any provenance information.

### 2.1.2 Comparison of Services Based on Technical Aspects

Besides evaluating the RDM services for their applicability against the RDM lifecycle, this Section investigates the technical aspects of RDM services under investigation. Furthermore, it highlights their commonalities and contributes to creating a universal reference software architectural model that conforms to all RDM services.

This study evaluates eleven technical and technological aspects of the RDM lifecycle. These criteria are Data Storage Location, Version Control, Persistent Identifier, Metadata (MD) Schema, Information Model, Search Engine, Archival, Authenticator, Data Sharing, Provenance Tracking, and Harvesting API. In addition, Table 2.2 demonstrates a birds-eye view of the RDM tools concerning their technical feasibility.

**Data Storage Location:** The location where RDM services use for file storage. As mentioned by Zuiderwijk et al.[143], research data are often scattered and accessible in local or/and remote resources throughout a research project. Zenodo, Sciebo-RDS, EUDAT CDI, and Figshare allow researchers to utilize their remote data resources by connecting their existing file servers to their data management service. Coscine, OSF, CKAN, Nextcloud, Dataverse, and DSpace empower researchers to include resources that are either only locally accessible or connect their projects to cloud-based repositories; thus, these systems are more suitable to fulfill the natural requirement of distributed file resources.

**Version Control:** The process of tracking a history of changes to data over time. Version control systems use the file name as a unique identifying indicator and keep previous versions only if a file with the same name already exists in a repository. Respectively, GitLab offers a Distributed Version Control (DVC) system [144] that allows copying a mirror of the repository in collaborating local machines. Thus, the entire version history of all files is available to all users and can be reconstructed in case a remote repository fails. On the other hand, all other software solutions that support version control utilize a Centralized Version Control (CVC) system [145] that keeps data history in a single remote server despite its vulnerability to missing all backups and versions in case of database corruption.

**Persistent Identifier:** A standardized and unique identifier for digital objects. The FAIR data management principles explicitly require DOI for research data to make data citable [146]. Parland-von et al.[147] describe a two-tier architecture for PID systems. The first tier is for assigning unique identifiers, and the second tier is for resolving the Identifier to its location. SimpleArchive, Metadata Tool, Coscine, and EUDAT CDI support users with PIDs for their research data. PIDs are a variant of DOIs created by European Persistent Identifier Consortium (ePIC) services compatible with DOI systems. Thus, PIDs can also be resolved and interpreted using the DOI server [148].

**Metadata Schema:** It highlights the extent of adaptability and customizability of MD schema by researchers. The development of FAIR metadata schemata requires close collaboration with domain experts to determine a set of vocabulary and ontologies that best describe their data [149]. Services like Metadata Tool, Coscine, Sciebo-RDS, eLabFTW, and EUDAT CDI allow users to define discipline-specific metainformation and create and adapt MD schema to fit a new series of experiments or datasets. However, despite the possibility for DSpace, Dataverse, and CKAN to

create MD schema, users are restricted to a pre-defined schema per system instance; thus, these systems are not flexible enough to fit multiple schemata concerning new datasets or disciplines.

**Information Model:** Machine-readable data formats facilitate annotating datasets with Metadata. Researchers widely use the Resource Description Framework (RDF) to build knowledge graphs for the task of metadata annotation [150]. Metadata Tool, Coscine, and CKAN benefit from RDF Data Catalog Vocabulary (DCAT) to link and reference between groups of data catalogs. DSpace employs Dublin Core eXtensible Markup Language (XML) schemas to structure documents and Metadata. The remaining services utilize either JavaScript Object Notation (JSON) or traditional Structured Query Language (SQL) schemata for their information model.

**Search Engine:** Software systems to index data and respond to on-demand search queries. Among all RDM software under study, only eLabFTW employs the Meilisearch engine [151] due to its suitability for keyword and filter searching; however, it is not suitable for complex queries over big datasets. OSF, Dataverse, and DSpace utilize Apache Solr [152] as a powerful full-text search engine that provides faceted search and near real-time indexing. Elasticsearch [153] is widely used by almost all other RDM software to empower the indexing and processing of big datasets while enabling semantic search capabilities.

**Archival:** The technology used for the long-term preservation of research data. Despite the RDM requirement to ensure data preservation regardless of the expiration of research projects, only Coscine and SimpleArchive provide their users with technologies that guarantee data archival. SimpleArchive uses a library of tape hard drives to store data, and Coscine uses Dell Elastic Container Service (ECS) object storage with support for Write-Once-Read-Many (WORM) [154] compatibility; in both techniques, users can not alter data after resource archival, and researchers are ensured from long term storage of data. Additionally, GitLab provides an archive feature that sets a database flag preventing users from further data modification. However, due to its reversibility, GitLab archival may not be satisfactory for some research projects.

**Authenticator:** The access protocol RDM software uses for authentication or authorization of users. There are various access control protocols used in RDM software, namely, Open Authorization (OAuth) [155], Lightweight Directory Access Protocol (LDAP) [156], Security Assertion Markup Language (SAML) [157], and API authentication Keys. Almost all RDM software benefits from the OAuth protocol to approve users' authorization with every user's attempts to access resources. Besides OAuth, Coscine, eLabFTW, Nextcloud, EUDAT CDI, and GitLab use SAML to allow users to log in with pre-existing Single Sign On (SSO) accounts such as university accounts. eLabFTW, Nextcloud, and DSpace provide LDAP to authenticate users to find directories. Additionally, eLabFTW and CKAN supply users with API keys for direct interaction with RESTful API.

**Data Sharing:** The possibility for controlling and restricting access to data in RDM platforms. There are three levels of data sharing, Private, Internal and Public. Research data set to Private are only accessible to the sole owner of a resource; the Internal level would allow anyone who is part of the research team to access research data; this can be either identified via users' affiliations or explicit allocation of a research project. Finally, the Public status would allow anonymous or platform users to access data for revision purposes. Most RDM solutions reflect the researcher's demand to control access rights by providing either Internal or Private resources [158]. However, Zenodo is found to be the only platform that pushes its users toward open research with no possibility to restrict access to research data, making it unsuitable for data with sensitive information. Moreover, CKAN and Sciebo-RDS do not provide data-sharing configuration; instead, they rely on data management platforms that utilize these services.

**Provenance Tracking:** The process of collecting contextual information to ensure the reusability and reproducibility of research results to increase the trustworthiness of digital objects [159]. Coscine and GitLab are found to be the only platforms capturing information about activities over data, allowing users to form assessments over historical records of data origins. Furthermore, Coscine builds a data provenance model by employing the PROV ontology standard [160], which can be serialized to RDF format to describe entities, activities, and agents.

**Metadata Harvesting:** It refers to a standardized interface for provisioning access to data and metadata repositories [161]. Almost all RDM services provide RESTful APIs and API keys, enabling a machine-operatable interface to access data repositories. In addition, Zenodo, Dataverse, EUDAT CDI, DSpace, and Figshare provide a dedicated interoperability framework via the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) interface. OAI-PMH is a low barrier protocol based on XML metadata standard in Dublin Core format, facilitating efficient dissemination of Metadata from repositories to ingest Metadata for indexing at generic data infrastructures.

## 2.2 Reference RDM Architecture

The qualitative analysis of the investigated RDM services reveals the challenges of fulfilling RDM requirements and its integration into existing research practices [162]. According to the findings, due to the technical challenges of seamless integration of RDM processes and maintaining IT infrastructure, RDM solutions often focus on discipline-oriented solutions or only satisfy a few RDM operations.



### 2.2.1 Conceptual RDM Architecture

To present a conceptual model for RDM process-aware services, this study adapts the reference software architecture from the n-tier Process-Oriented Software Tiers for Application Interfaces and eXtension (pSTAIX) model [163] as also shown in Figure 2.1. To construct a service-oriented architecture, pSTAIX abstracts decentralized systems and uses mediator tier pattern representation to allow for separation of concern for business processes and related technologies [164]. Furthermore, the tier interfaces highlight the process-aware services and the composition of multi-layer independent services from other tiers such that backend systems from lower tiers can be utilized independently or in combination to fulfill new process-oriented responsibilities. The following is a brief introduction to pSTAIX tiers adapted from Politze et al.[163], [165]:

- **Tier 0:** Authorization and Security refers to the crucial need for services to identify users and ensure data security and privacy.
- **Tier 1:** General Services offer fundamental and process-independent functionalities to support different operations across boundaries.
- **Tier 2:** Technology Dependent Services support specialized core process-aware functionalities.
- **Tier 3:** Standardized Access to Backend Systems via APIs to enable communication of services and interfaces.
- **Tier 4:** Process-Aware Services combines services from previous tiers into an abstracted organizational level.
- **Tier 5:** RDM Operations provides client applications and processes to fulfill RDM activities.

The conceptual architecture presents a generic software architecture model for RDM services inspired by the pSTAIX model. The data analytics framework suggested in this thesis applies to systems that fit into the universal architecture for RDM software illustrated in Figure 2.1. Furthermore, the reference architecture demonstrates the decentralized nature of services and their crucial interconnections between each tier.

With respect to the analysis of RDM tools, besides LDAP and SAML, all RDM tools employ OAuth services in *Tier 0* to check the authorization of users to access resources.

*Tier 1* is responsible for generic RDM-related services such as Organization Structure to supply user identity information, Resource Storage Management to locate research data, Metadata management to ensure collecting contextual information over research data, Indexing in order to list data objects into digital libraries, Unique Identifier for pinpointing individual datasets, Archival service supplies services for long term

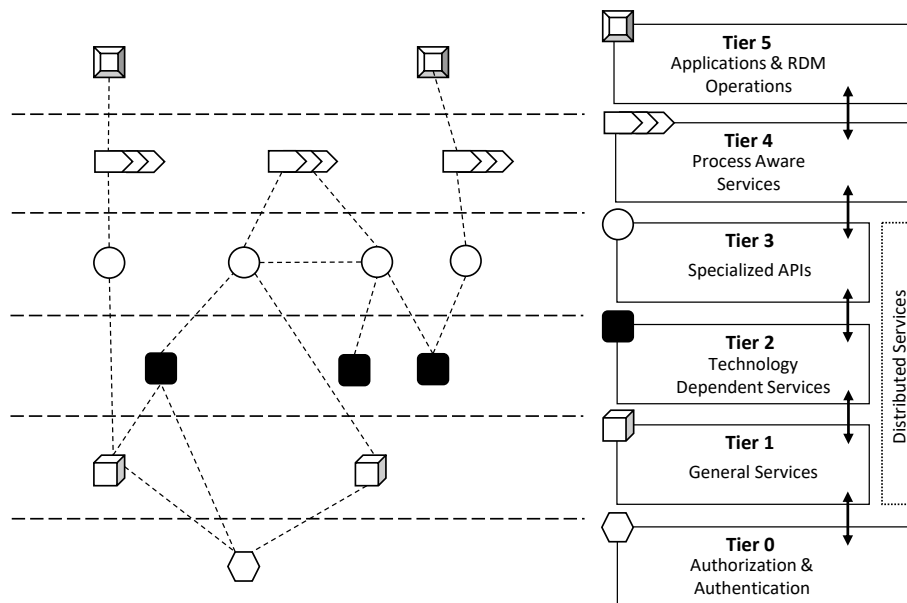


Figure 2.1: A universal reference architecture for RDM services derived from the pSTAIX model.

data preservation, and Notification service provides a channel for informing users of prespecified actions.

*Tier 2* offers general-purpose but technology-specific services that work in users' contexts: Elasticsearch, SSO, Object Store, Linked Data, Temporary File Storage, Tape Library, DOI, EPIC, and SMTP Services.

*Tier 3* includes process-independent backend APIs designed and implemented by experts with a technical understanding of relations to other adjacent tiers. Some of the found APIs that respond to demands of *Tier 3* are: Resource API provides methods for interacting with resources, Secure Token Service (STS) API for securely connecting to SSO service, Quota Handling API to deals with provisioning resource quotas, Blob API handles the interaction with large binary objects, PID API issues persistent identifiers using either DOI or ePIC services, Versioning API monitor and track changes in data, Tree API support storing or retrieving metadata, Project API handles the creation of projects and sub-project structure and restricting access to project members. User Directory API captures users' affiliations and information, Archive API take responsibility for preserving resource from further modification, Search API utilizes search engine services to discover data, and Notification API informs users of necessary actions by sending messages.

*Tier 4* is a process-aware service providing actual business values for end users by integrating interfaces from previous tiers.

*Tier 5* corresponds to stages of the RDM lifecycle and its related client applications. Additionally, external distributed services such as high-performance computing, publication libraries, and encryption can fit well into the conceptual RDM software architecture.

## 2.3 Systems Under Study

In the remainder of this thesis, SimpleArchive, Metadata Tool, and Coscine are employed as candidate RDM services due to their comprehensive IT infrastructure accessibility and the possibility of acquiring reliable data for additional studies. This Section discloses the supported processes using each service and how they benefit from the distributed systems and fit the reference RDM software architecture model described in Section 2.2. Furthermore, in Chapter 5, through several case studies, the validity of the proposed methodologies for modeling RDM processes on top of the services mentioned earlier is investigated.

### 2.3.1 SimpleArchive

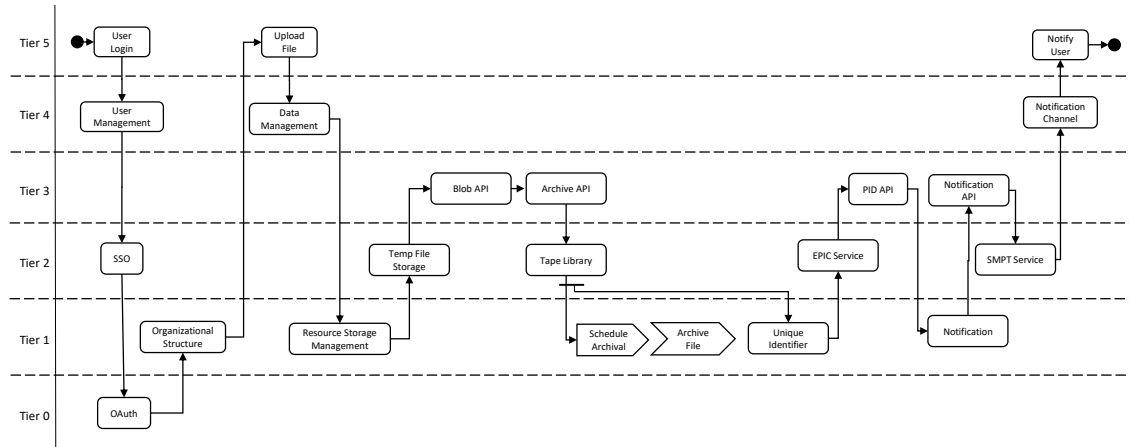


Figure 2.2: pSTAIX tier architecture model for archiving files on SimpleArchive (Archival Phase).

The goal of SimpleArchive is to enable the long-term preservation of research data at RWTH Aachen University using tape archive technology. Users of SimpleArchive can archive their files for 10 to 30 years via a web interface where users can upload files and select desired retention period. Furthermore, the system allocates PID for each file, and researchers can link scientific papers to the corresponding data with PIDs' help.

Figure 2.2 shows the archival process for SimpleArchive according to pSTAIX RDM reference software architecture. The process starts with the user attempting to log in to the system; after authorization and authentication verification, the researcher uploads the data artifact. The data is initially stored on temporary storage and queue for the tape archival process to execute. In the archival process, a PID is issued for each data artifact, and since the process is asynchronous, users are notified of the completion of the archival process via email. Furthermore, researchers can use PID to identify their data and request restoring artifacts, which generates a temporary copy of archived data that is only accessible for two weeks.

### 2.3.2 Metadata Manager

In order to support the rediscovery and reusability of data artifacts, one needs to explain data with the help of metainformation. Accordingly, Metadata Manager facilitates metadata registration for research data to help further researchers document essential contextual information. Although this tool allows collecting metadata independent of data location, researchers can link their data from SimpleArchive to metadata objects. Moreover, researchers can predefine standardized discipline-specific metadata schemas according to their needs while utilizing their own IT infrastructure to store data.

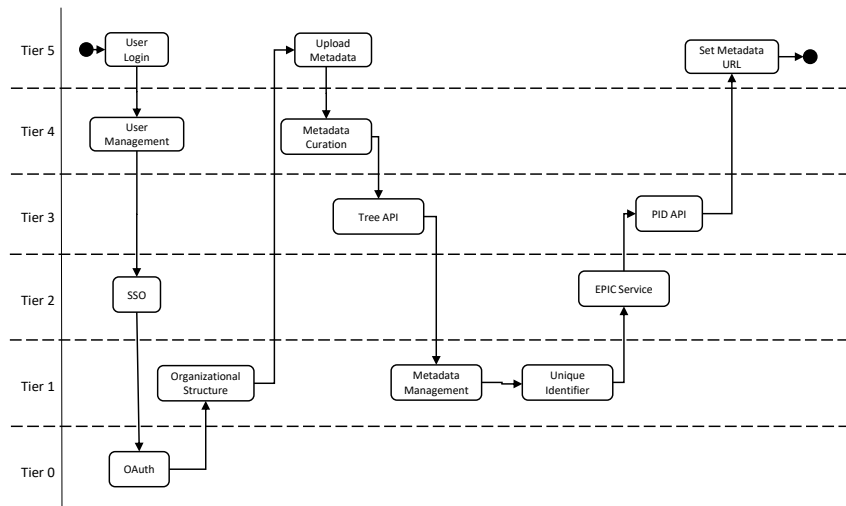


Figure 2.3: pSTAIX tier architecture model for uploading metadata on Metadata Manager (Production Phase).

Figure 2.3 demonstrates the supported process workflow according to the RDM reference architecture. After the user attempts to log in to Metadata Manager, authorization and authentication verification steps occur; researchers can then use

the web interface to upload RDF formatted metadata files generated by Metadata Manager. Next, metadata Curation at Tier 4 receives the request for registering metadata, and Tree API works as a service to validate conformity to predefined metadata schema and save metadata in a centralized database. Furthermore, the Metadata Manager creates a request to the Unique Identifier service to issue a PID and generate a metadata Uniform Resource Locator (URL) to identify a dataset and link metadata set to a corresponding data artifact.

### 2.3.3 Coscine

Coscine is a collaborative scientific integration platform focusing on metadata management according to FAIR principles. An intermediate description of supporting technical and RDM processes by Coscine is previously described and adapted by Politze et al. [21], [166].

Coscine allows researchers to manage, combine and link data resources while accompanying research data with metadata down to individual files level, eventually empowering researchers with data reusability and letting peer researchers regain a meaningful interpretation of data.

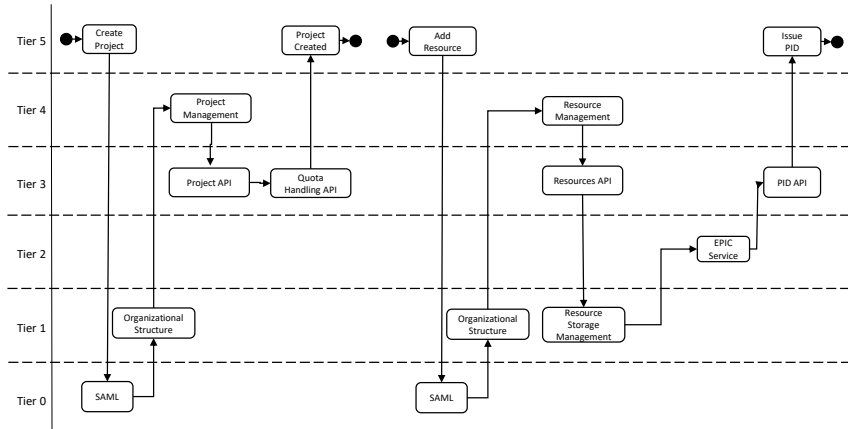


Figure 2.4: pSTAIX tier architecture model for creating projects and resources on Coscine (the Planning phase).

Coscine acts as an information hub by providing access and managing interlinked resources in a centralized environment without interfering with data-intensive workflows. Accordingly, researchers with specific affiliations can automatically get storage capacity for certain resource types without human intervention or evaluation. Furthermore, the ePIC PID system and its sub-identifiers allow uniquely locating data objects and deep linking to resource contents by using the fragmentation of the PID relative file path, thus identifying a particular file version without requiring

issuing PIDs for each file. Additionally, Coscine utilizes the Virtuoso RDF database for storage of knowledge graphs and retrieval in RDF triples (subject-predicate-object) format, where the subject is the file-specific PID, the predicate expresses the relationship, and the object is the metadata value.

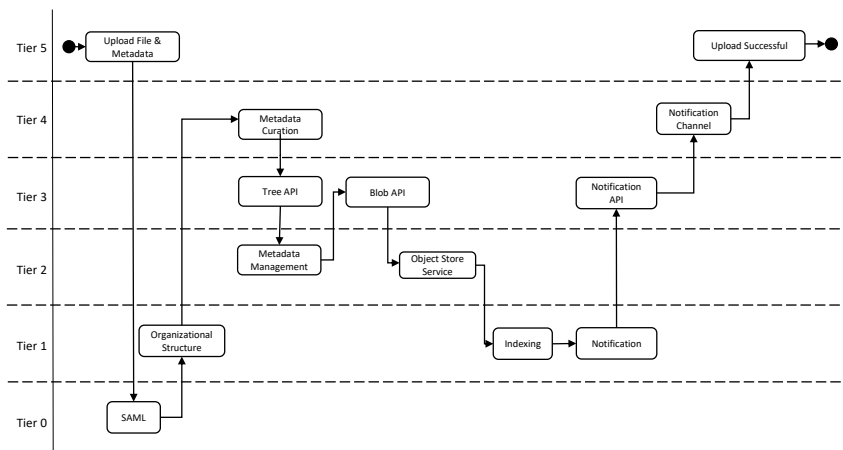


Figure 2.5: pSTAIX tier architecture model for uploading files and metadata on Coscine (the Production phase).

A central integrative platform like Coscine provides the opportunity to mimic the organizational structure existing in scientific projects by creating sub-projects and allowing for managing project memberships and defining rules for accessing research data. This OSP pushes users to provide additional metadata while uploading data; thereby, the data interaction workflows and system design guarantee the existence of meta information for each uploading object. To minimize the required metadata documentation efforts, the predefined metadata schemas can mark metadata as optional or mandatory with suggested prefilled or fixed values, and the metadata can be uploaded or harvested via RESTful APIs.

According to RDM reference architecture, Figure 2.4 and Figure 2.5 demonstrate a typical researcher's workflow process in Coscine for Planning and Production phases of RDM Lifecycles. Figure 2.4 shows the events triggering for creating projects and resources according to DMPs, which implicitly contributes to the Planning phase of a resource project. The software architecture highlights the execution of Quota Handling API as part of the project creation process and the minting of PIDs for each resource. Furthermore, Figure 2.5 highlights the services and APIs involved in uploading files. The model indicates that files can only be uploaded to the database after successfully capturing their metadata.

Overall, all systems under investigation satisfy the generic software architecture described in Section 2.2. With the help of this model, we can determine ideal methods to capture reliable data that enable us to study RDM underlying processes.

Service	Data Storage Location	Version Control	Persistent Identifier	Metadata Schema	Information Model	Search Engine	Data Archival	Authentication Protocol	Data Sharing	Provenance Tracking	Metadata Harvesting
RDMO	Local										
SimpleArchive	Local		PID		SQL		Tape Library	OAuth	Internal / Internal / Internal / Private / Internal		
Metadata Manager	Local		PID	Dynamic	RDF (DCAT)	Elasticsearch		OAuth	Internal / Internal / Internal / Private / Public		
Coscare	Local / Remote	CVC	PID	Dynamic	RDF (DCAT)	Elasticsearch	ECS WORM / DB Flag	OAuth / SAML	Internal / Private / Public	Own Prov Model (Process Mining) / Prov Ontology	
Zenodo	Remote		DOI	Fixed	SQL	Elasticsearch		OAuth	Public		OAI-PMH
OSF	Local / Remote	CVC	DOI	Fixed	SQL	Apache Solr		OAuth	Private / Public		
SciELO - RDS	Remote			Dynamic				OAuth	Internal / Private / Public		
elabFTW (ELN)	Local	CVC		Dynamic	JSON	Meilisearch	API Key / LDAP / SAML		Internal / Private / Public		
CKAN	Local / Remote	CVC	DOI	Fixed	RDF (DCAT)	Elasticsearch	API Key		Internal / Private / Public		
Nextcloud	Local / Remote	CVC			SQL	Elasticsearch	LDAP / SAML / OAuth		Internal / Private / Public		
Dataverse	Local / Remote	CVC	DOI	Fixed	JSON	Apache Solr		OAuth	Private / Public		OAI-PMH
EUDAT CDI	Remote	CVC	PID	Dynamic	JSON	Elasticsearch		OAuth / SAML	Internal / Public		OAI-PMH
Dspace	Local / Remote		DOI	Fixed	XML (QDC)	Apache Solr	OAuth / LDAP		Internal / Private / Public		OAI-PMH
Figshare	Remote		DOI	Fixed	JSON	Elasticsearch		OAuth	Internal / Private / Public		OAI-PMH
GitLab	Local	DVC			SQL	Elasticsearch		OAuth / SAML	Internal / Private / Public	Own Prov. Model (Actions History)	

Table 2.2: A Technical Comparison of Services Supporting RDM





# 3 Data Acquisition

## Techniques from RDM Systems

In a distributed environment, many machines and services often respond to executing tasks, resulting in a non-trivial assignment for resource monitoring and forecasting of the state of a system. Moreover, oversight of architectural system dependencies and resource utilization is unavoidable as a distributed system matures and expands its services. Thus, it is progressively essential to enable logging services merely focus on collecting traces of resource usage and discovering supported processes.

Acquiring appropriate datasets from real-world environments is the most challenging task for data scientists and often requires many iterations to establish a suitable data acquisition approach. Moreover, there are already several studies to reconstruct tables and databases to form datasets necessary for data modeling analysis; nevertheless, since data and its attributes in databases often get replaced or altered without any trace, the extraction of datasets primarily comprises the current status of a system. Thus, this Chapter discusses several data acquisition techniques to assist with capturing data on-the-fly and ensure the reliability of discoveries using premortem datasets.

In this approach, the process began by asking stakeholders and software architects to gain a mutual understanding of constraints and expected data structure within available services. Then, with every iteration of the methodology evaluation, the findings were discussed with domain experts to revise the method development path and its validity. The following are descriptions of final iterations for establishing various procedures to acquire datasets to answer the research questions.

In order to begin studying the underlying RDM activities and discover data models within OSPs, one needs to initially identify all services in use and explore methods for generating logs. As mentioned earlier in Section 2.2, the system architecture of RDM services benefits from the scalability and modularity of distributed environment and incorporates remote resources for responding to user requirements. Hence, a suitable logging technique should be able to outline user activities across interconnected resources.

Besides discussing three distinct approaches for collecting datasets and their advantages and drawbacks, this chapter proposes a scalable mechanism for capturing

and storing reliable data to support uncovering resource utilization in distributed settings. The results and suggested data acquisitions techniques in this Chapter were previously partially published in [22], [24], [27], [167], [168].

## 3.1 Motivations and Challenges

In recent years, the volume of data generated by distributed systems has grown exponentially. This increase in data volume has led to new challenges in obtaining data logs from these systems. Event logs are critically important for understanding what has happened in a system and why. In particular, researchers can execute a wide range of activities within an OSP service to fulfill their RDM-related needs; furthermore, OSPs typically benefit from distributed systems and decentralized resource management, consequently invoking several services to respond to user requests. Also, data modeling studies often undermine the complexity and challenges of data acquisition and preparation in a distributed environment [169]. Thus, the benefits of logging user interaction on RDM services are twofold; on the one hand, *it enables scientists to gain insights into research data origins and highlight user interaction traces over research data artifacts*. On the other hand, *it supports system admins in accurately monitoring resource usage and tracing data flow between resources*.

As part of Good Scientific Practice, researchers are encouraged to share research files, but there is a lack of study on tracing research data flow between resources due to the absence of capturing its footprints [170]. Additionally, these footprints are crucial for researchers to evaluate file reusability, authenticity, and origin [171]. Hence, logging file roots, tracing, and illustrating user interactions over files are essential factors for users in decision-making a research file usefulness [172]. As mentioned earlier, although logging user interactions in a distributed environment is not a trivial task, a suitable and efficient logging system allows for proactive data modeling analysis, uncovering errors in a system before user reports, and discovering requirements using implicit usage data.

The challenges of acquiring sample data from the distributed system for data analysis and modeling have already been discussed in many pieces of scientific literature. Subsequently, there are challenges with extracting and collecting reliable data from RDM tools that benefit from distributed settings. The primary challenge is that data collected from RDM tools in distributed settings can be inaccurate or incomplete, making it difficult to draw valid conclusions. In addition, the process of collecting data from decentralized data sources can be time-consuming and expensive. Another challenge is that distributed systems often have different types of nodes, each with its unique logging requirements and programming language. For example, a web server may generate access logs that need to be parsed and analyzed, while a database server may

generate transaction logs that need to be processed differently and need to ensure that all messages are delivered in the order in which they were originally generated.

Additionally, there is a potential for inconsistency in the data collected from different sources and leading to errors in the analysis and interpretation of the data [173]. A robust logging solution for a distributed system must tolerate failures of individual nodes without affecting the availability of the logging service as a whole. The heterogeneity of data derived from distributed systems is a significant challenge for data integration. The differences in conceptual models, data models, and semantics make it difficult to integrate data from different sources [174]. The difficulty in querying heterogeneous data sources is due to underlying different data models and interfaces. This results in difficulty in identifying an appropriate data source for a particular query and translating queries between different data sources.

In order to enable the study of underlying RDM processes, one needs to collect event logs in a format appropriate for the process and data mining tasks. As discussed in Section 1.2.1, an event refers to a *Case*, an *Activity*, and *Timestamp*. An event log is a collection of cases that occur in a sequence of events. There are many benefits to using event data logging. For example, event data can be used to detect patterns of behavior, trends over time, and relationships between different events. Moreover, an event log contains suitable attributes to be converted to a matrix of labels and features for machine learning algorithms. Also, event data logging can improve efficiency and decision-making by automating the collection and analysis of data. However, despite the advantages of event data logging, some challenges associated with employing event logs, such as collecting event data, can be resource-intensive, requiring significant storage capacity and processing power. In addition, extracting meaning from event data can be difficult, particularly if the data is unstructured or poorly organized. Finally, privacy concerns may arise if event data is used to identify individuals or track their behavior.

Thus, it is essential to carefully plan and design the data collection process for any scientific study. The process must be designed to accurately and efficiently collect the desired data. Several considerations have to be evaluated when designing a data collection process, including the type of data being collected, the methods used to collect the data, and the study's specific objectives. Additionally, as a system grows, more nodes get added to the overall body of a distributed setting, and the amount of data generated will increase. Hence, the logging solution should handle this increase in data heterogeneity without affecting performance by using standardization, mediation, and mapping techniques.

## 3.2 Dataset Characteristics and Quality Measures

There are several potential problems with relying on database tables as the sole source of data for later analysis. Database tables may not carry crucial and meaningful information, such as the Timestamp, which may not exist, resulting in the impossibility of reconstructing the sequence of events or exploring the actual lifecycle duration for a process. Moreover, database table modifications such as insertion or deletion are performed on table entries, leading to inconsistencies in the extracted dataset. Consequently, sample data may be incomplete or even unattainable. For these reasons, data logs should be captured on the fly and stored separately to maintain a long-tail record of events per Case. In addition, it allows for easy retrieval and analysis of event data, even in possibilities where the database is no longer available.

The phrase “garbage in, garbage out” is often used to describe the importance of data quality concerning the accuracy of results. The same is valid for data and process mining tasks, which rely on event logs to provide accurate and reliable insights into business processes. If the event log data is of poor quality, it results in inaccurate discoveries. Therefore, ensuring that event log data is clean and complete before undertaking any process mining analysis is essential. Likewise, datasets acquired from RDM services should enable data scientists to study the start and end of user interactions within the RDM lifecycle to reflect how resources are accessed or utilized, support identifying failures or errors in a system, and discover causalities. Provide means for pinpointing processes that lead to encouraging positive behavior toward RDM doctrines. Foster frequency analysis to disclose suitable KPIs and assist with isolating activities, user stories, or processes from the rest of the activity traces and tracing activities processed within the connected systems.

In consonance with Bose et al.[175], Suriadi et al. [176], Van der Aalst [55], and the IEEE Task Force on Process Mining [177], data quality are identified by six dimensions:

***Accuracy :*** *How well does the information reflect reality according to the level of data granularity.* The level of data granularity is vital to consider when determining how well the information reflects reality. For example, if the data is too coarse, it may not accurately reflect what is happening at a more complex level. Conversely, if the data is too fine, it may be challenging to reveal the overall trends.

***Timing:*** *How data is being collected to deal with out-of-order arrival of events.* There are a few practices to collect data to deal with the out-of-order arrival of events, such as using a reliable timestamp, a sequence number, or a buffer, which can temporarily store data until it can be processed in the correct order.

**Completeness:** *The data’s comprehensiveness and if a dataset is complete or may include missing crucial information.* The dataset completeness may have implications on missing key data points, yielding challenges for data analyses. For example, a dataset missing information on a crucial variable, such as case Id or user activity, would not be considered complete.

**Uniqueness:** *The notion of cases to identify a subset of records using correlation identifiers to specify a trace of events.* A unique case is one in which there may be multiple records to complete a user story, while correlation identifiers should help determine and isolate a trace unique to its user story or Case.

**Consistency:** *How well the collected data is stored in multiple locations without mismatch to represent interconnections.* It can be challenging to maintain consistency if data is spread across different data sources.

**Validity:** *A data acquisition method should produce a standardized and unique data format for logging relevant information.* If data is not formatted correctly, it can be difficult or impossible to converge it with other data sets.

Though most studies focus on cleaning and refinement, not enough emphasis is given to a systematic and generalizable approach for data extraction according to the criterion mentioned earlier. As a result, there is a significant challenge in applying process mining when it comes to the quality of event data in distributed systems [175], [177]–[179].

According to van Cruchten [180], there are several requirements that a data extraction technique must meet in order to be effective. First, the technique should be systematic and extract logs from data sources consistently without requiring manual interventions. Second, the data acquired should apply to various data analysis tasks to ensure the usefulness of extracted event logs for later decision-making regardless of a specific data analytics project. Third, data extraction should be automated as much as possible to avoid human error, which is crucial when dealing with large data sets or when the data is susceptible. Finally, the extraction technique should be transparent, allowing domain experts to understand how the event logs were generated and verify the data’s accuracy. Therefore, while conforming to the discussed data quality measures, four data acquisition techniques suitable for the general software architecture of RDM services are discussed in this Chapter, contributing to answering **RQ 1**.

### 3.3 Strategies

The following section investigates approaches exerted during this thesis to acquire suitable datasets for data and process modeling tasks. Namely, there are Centralized,

Client-Side, Server-Side, and Hybrid logging techniques corresponding to RDM software architecture *Tiers 0, 5, 3*, and *0&5*, respectively.

Table 3.1 provides a comparative view of each logging technique concerning data quality metrics. According to findings via several case studies discussed in Chapter 5, the suggested techniques serve for RDM services that fit the RDM reference architecture discussed in Chapter 2.

Table 3.1: An overview of logging methods in terms of data quality metrics.

Logging	Accuracy	Timing	Completeness	Uniquess	Consistancy	Validity
Client-Side ( <i>Tier 5</i> )	✓	×	✓	✓	×	!
Server-Side ( <i>Tier 3</i> )	!	✓	!	✓	✓	✓
Centralized ( <i>Tier 0</i> )	!	✓	!	✓	✓	✓
Key Hybrid Technique ( <i>Tiers 0 &amp; 5</i> )	✓	✓	!	✓	✓	✓

### 3.3.1 Centralized

According to Chapter 2, RDM web services require independent microservices that work together to form a system landscape. Distributed IT architecture provides software components’ scalability, maintainability, and reusability. Consequently, responding to a user request can be processed using multiple web services. For example, the software components for archiving a file can be reused by other applications in the system to initiate an archiving process. However, due to the scattered data flow within such a distributed but interconnected workflow, analyzing the dynamic behavior of components during usage and user activities becomes challenging and nearly impossible, especially when dealing with legacy systems. A centralized logging service can provide homogeneous event logs, making tracking and managing events easier. This approach is more straightforward and can be tailored to the specific needs of each service.

This Section introduces the instrumentation of OAuth workflow to capture suitable logs for analyzing underlying RDM activities. According to findings discussed in Chapter 2, an authorization workflow is utilized by nearly all RDM tools for authorization and security. For instance, the OAuth workflow corresponds to *Tier 0* of the general architecture of RDM services. The OAuth protocol is a secure way to allow applications to access resources on behalf of a user. It handles the user’s authorization without needing their credentials to be supplied to applications or REST APIs. The OAuth workflow has various extension points to integrate with existing identity federations, but the core protocol remains consistent [181], [182].

The verification of user tokens during each RDM operation makes it an ideal spot to capture logs. By authenticating users before executing requested operations and triggering related APIs, thus, it can be ensured that only authorized users can access sensitive information. This makes OAuth an optimal place to acquire logs while protecting sensitive data.

### Approach

As a result of employing the OAuth token service to collect and aggregate the information about processed requests, we can monitor the resources and services responsible during software and users' workflow. This helps to identify a process for an instance of executed resources uniquely [183].

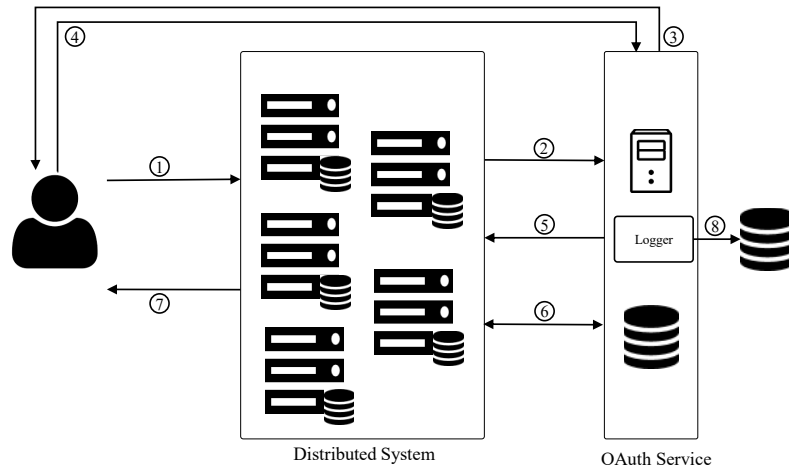


Figure 3.1: The schematic representation of the OAuth workflow in distributed services. Adapted from Yazdi and Politze [24].

Figure 3.1 shows the phases for the authorization process using the OAuth service and capturing logs, and the following is an explanation of each step:

1. The workflow initializes with a user establishing a connection to an application, which needs to access several microservices in the context of that user.
2. The application diverts the user to the web interface of the token service for submission of username and password.
3. The user authorizes the application to access the microservice with his credentials.
4. The token service allocates an access token to the application to represent the authorization.

5. On each later request to the microservices, the application uses the token to obtain the authorization.
6. The microservice checks the token's validity against the OAuth service for each subsequent request.
7. The application and the user obtain a response with requested data from the microservice.
8. The logs are generated by intercepting all authorization calls to microservices, thus facilitating the instrumentation of logs according to the data analysis project.

Service providers may add contextual information to their requests when processing the token verification via the authorization server. In addition, the authorization server can log n-tuples with an arbitrary length to incorporate supplementary information according to the agreement on reported data and their semantics with services in a federation. Listing 1 is an example of a token validation request in JSON notation that is extended with additional information to enable further data analysis studies.

#### **accessToken**

It is a compact and encoded JSON Web Token (JWT) to transmit signing information during and after the authorization process securely. The token remains active, corresponding to users' continuous activity within the application.

#### **userHashId**

Encrypted user ids with SHA-256 algorithm to uniquely identify users while preventing the database user IDs from being revealed in the verification process.

#### **microservice**

Indicates the microservice required by an application to respond to a user request.

```
1      {
2          "accessToken": "2pdMihRsq10Z3eqAGibdSFndRnCbt2w...",
3          "userHashId": "BaU564AJHPSCFKJFi6wKUOWY...",
4          "microservice": "microservice.rwth",
5          "methodCall": "https://resources.example.com/method()",
6          "operation": "GET"
7      }
```

Listing 1: An extended token validation request in JSON notation.



**methodCall**

It is defined by a URL that indicates the API method that demands the token for execution.

**operation**

Indicates the kind of RESTful HTTP method performed on a resource.

## Preliminaries

The following is a formal definition for event logs generated from the OAuth logger. An example of an event log is shown in Table 3.2. The **operation** is excluded from the event log as it does not carry valuable data for further analysis.

Table 3.2: An example of dataset  $e_o$  captured from distributed services using the Centralized logger.

timestamp ( $t_o$ )	caseId ( $c_o$ )	userHashId ( $u_o$ )	activity ( $a_o$ )	microservice ( $m_o$ )
...	...	...	...	...
2019-12-16 08:59:56	Izo9VZ9vz...	dhcwb4MIL...	GetNotifications()	simpleArchive
2019-12-16 08:59:56	Izo9VZ9vz...	dhcwb4MIL...	GetUser()	simpleArchive
...	...	...	...	...

**Definition 3.1. (OAuth Event, Trace, Event Log).** An event driven from OAuth workflow is denoted by  $e_o$  and is a n-tuple  $e_o = (t_o, c_o, u_o, a_o, m_o)$ , and  $e_o \in L_o$ . Where  $t_o \in T$  is the event timestamp for the OAuth server corresponds to an activity,  $c_o \in C$  is the case id that corresponds to **accessToken**,  $u_o \in U$  is the **userHashId**,  $a_o \in A$  is the activity that corresponds to **methodCall**.  $m_o \in M$  is the microservice triggered by an application.  $\nabla_n(e_o)$  is the value of attribute  $n$  for event  $e_o$ . If  $e_o$  does not have an attribute named  $n$ , then  $\nabla_n(e_o) = \perp$ , where  $\perp$  is a designated null value.

The event log  $L_o$  contains traces, and each trace contains events such that each case has an attribute trace denoted with  $\nabla_{trace}(c) \in \xi_o^*$ . Given a set  $A$ , a trace  $\sigma = \langle e_1, e_2, \dots, e_n \rangle \in A^*$  is the set of all finite sequences over the  $A$  that each event appears only once.

I denote  $\xi_o = T \times C \times U \times A \times M$  as the universe of all events for the OAuth event log, s.t.  $L_o \subseteq \xi_o$ . In the event log, each event can appear only once. Hence, events are uniquely identifiable by their attributes.

## Discussion

Table 3.2 shows a sample of the aggregated data collected by the OAuth service. The *timestamp* determines when a request has been executed and processed. The

*caseId* is the unique token utilized during the authorization process. The *userHashId* is a hashed and anonymized user's unique identifier. The *activity* column collects the triggered software components. The *microservice* is a unique identifier for every single microservice involved in the process.

Traces are created by logging the execution threads between different microservices  $m_o$ . These traces represent the lifecycle of various processes, allowing us to discover a system's behavior. In a distributed system, a user request creates a sequence of event logs that may scatter to multiple nodes depending on the nature of the request, each is collected as traces individually, and each trace corresponds to an execution process instance to generate an overall model. These traces can be service requests by external users or internal communication of different service components. Thus, analyzing or interpreting event logs generated by centralized logging is challenging.

According to data quality dimensions discussed in Section 3.2, despite the nature of distributed systems to execute events in a dispersed and out-of-order manner, using server timestamps allows for sorting event logs according to events' timestamps and consequently discovering valid real-life processes. However, there is uncertainty concerning the accuracy and completeness of event logs yielded by OAuth logging. A centralized logging solution should handle different types of logs generated by the various nodes in a distributed system and scale horizontally as a system grows. Additionally, these system events may still miss crucial information, as OAuth only logs the software components that demand authorization to continue processing the execution of an operation, thus lacking events that do not require OAuth service. A case study of discovering underlying process models from distributed services contributing to SimpleArchive and Metadata Manager as two RDM services using centralized logging is analyzed in Section 5.1.1.

#### 3.3.2 Client-Side

Client-Side logging is the process of tracking and recording user interactions with a web browser. This information can be used for various purposes, including usability studies, marketing research, and service optimization. According to the RDM general architecture, the Client-Side logging corresponds to *Tier 5* or the presentation layer to capture user RDM activities on an application.

##### Approach

There are two main approaches for collecting Client-Side logs. One is to use browser extensions or plugins, which the user or server admins can install. Such plugins are

Google Analytics, Piwik, Adobe Analytics, and many more. However, due to uncertainty of the compliance of these browser extensions to GDPR and user privacy policies, usage of such plugins is discouraged for RDM services. Thus, a self-implemented, embedded JavaScript code was used to track and record user interactions and send the data to the database for analysis without placing cookies into browsers.

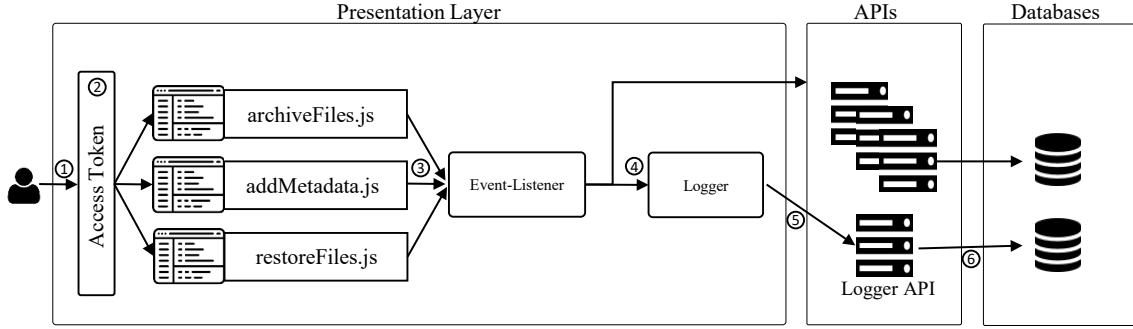


Figure 3.2: The schematic depiction of the Client-Side logger workflow.

Figure 3.2 demonstrates the approach for capturing event logs from the Client-Side.

- 1- A user attempts to log in to an RDM application.
- 2- Web socket provides an access token identifier to identify a user uniquely.
- 3- By executing any Client-Side action, Event-Listener is executed to make an asynchronous call to the back end.
- 4- Event-Listener executes two simultaneous calls to the Logger and, additionally, to respective APIs.
- 5- The logger collects and compiles the necessary event information and passes data to the logger API.
- 6- Logger API handles checking the format of calls and storing events in a centralized logger database.

## Preliminaries

The following is a fundamental concept and a formal model used for the Client-Side logger. An example of an event log derived from the Client-Side logger is shown in Table 3.3.

**Definition 3.2. (Client-Side Event, Trace, Event Log).** An event derived from client-side logger is denoted by  $e_c$  and is a n-tuple  $e_c = (t_c, c_c, a_c, r_c, w_c)$ , and

```

1 let logger = function(){
2   socketUtil.GetToken(function(accessToken){
3     let requestData = [];
4     targets.forEach(function(element){
5       let target = element.target;
6       requestData.push({
7         "timestamp":Date.now(),
8         "caseId":accessToken,
9         "activity":getSelector(target.xpath),
10        "resource":getApp(),
11        "webPage":getWebPage(target)
12      });
13    });
14    jQuery.ajax({
15      type:"POST",
16      contentType:"application/json",
17      url:requestUrl,
18      data:JSON.stringify("attributes":requestData)
19    });
20  });
21 };

```

Listing 2: An example JavaScript Client-Side logger.

$e_c \in L_c$ . Where  $t_c \in T$  is the event timestamp for client activity,  $c_c \in C$  is the case id that corresponds to `accessToken`,  $a_c \in A$  is the user activity that corresponds to JavaScript method call,  $r_c \in R$  is the application, and  $w_c \in W$  is the web page.

The event log  $L_c$  contains traces, and each trace contains events such that each case has an attribute trace denoted with  $\Omega_{trace}(c) \in \xi_c^*$ . Given a set  $A$ , a trace  $\sigma = \langle e_1, e_2, \dots, e_n \rangle \in A^*$  is the set of all finite sequences over the  $A$  that each event appears only once.

I denote  $\xi_c = T \times C \times A \times R \times W$  as the universe of all events for the Client-Side event log, s.t.  $L_c \subseteq \xi_c$ . In the event log, each event can appear only once. Hence, events are uniquely identifiable by their attributes.

## Discussion

As indicated in Table 3.1, several challenges and advantages are associated with Client-Side logging concerning data quality dimensions. Client-Side logging may increase network bandwidth overhead due to posting a logger with every call. There are also significant concerns related to events Timing, Consistency, and Validity. For example, mismatches in timestamps and out-of-order arrival events were identified,

Table 3.3: An example of event  $e_c$  captured from the presentation layer of SimpleArchive using embedded Client-Side logger.

timestamp ( $t_c$ )	caseId ( $c_c$ )	activity ( $a_c$ )	resource ( $r_c$ )	webPage ( $w_c$ )
...	...	...	...	...
2012-01-06 02:24:01	Izo9VZ9vz...	selectFile	SimpleArchive	manage
2012-01-06 02:24:01	Izo9VZ9vz...	restoreFile	SimpleArchive	admin
...	...	...	...	...

which were caused by browsers and device dependencies, resulting in uncaught errors. Moreover, there are threats to the validity of logs generated by the Client-Side logger. Such as the loggers on the presentation layer have to post-process events to clean the dataset due to the execution of many unknown events as the result of existing legacy and third-party JavaScript codes. Also, the log may include events that users on the front end manually manipulate, and there is no possibility to record processing an API call, thus, causing threats to the consistency of the log.

Despite the challenges mentioned above, there are advantages to using a Client-Side logger. First, the log represents a high accuracy of actual events carried out; thus, there would be no need to extend the logger as a system grows or new features are implemented. Secondly, this approach does not require `UserId` and can entirely depend on `accessToken` to uniquely identify cases for the duration of validity of a token. Finally, the Client-Side logger can easily get extended with additional contextual information to enrich the dataset further. For instance, it can help detect unexpected user behaviors that cause an error in the system, such as clicking a button multiple times [184]. Hence this approach can provide valuable insights into how users interact with RDM web interfaces.

### 3.3.3 Server-Side

Server-Side logging is being used as a method for capturing events before being processed by back-end APIs and Libraries. The Server-Side logging corresponds to *Tier 3* of the RDM general software architecture.

The obtained information is a set of user requests received by the API endpoints to receive requested resources. In this approach, event data is generated and captured in a serialized JSON object format due to its ease of extendibility to incorporate supplementary attributes and entities without extending database tables to incorporate new event types. The event logs include detailed insights into the sequence of actions and respective RDM-relevant entities.

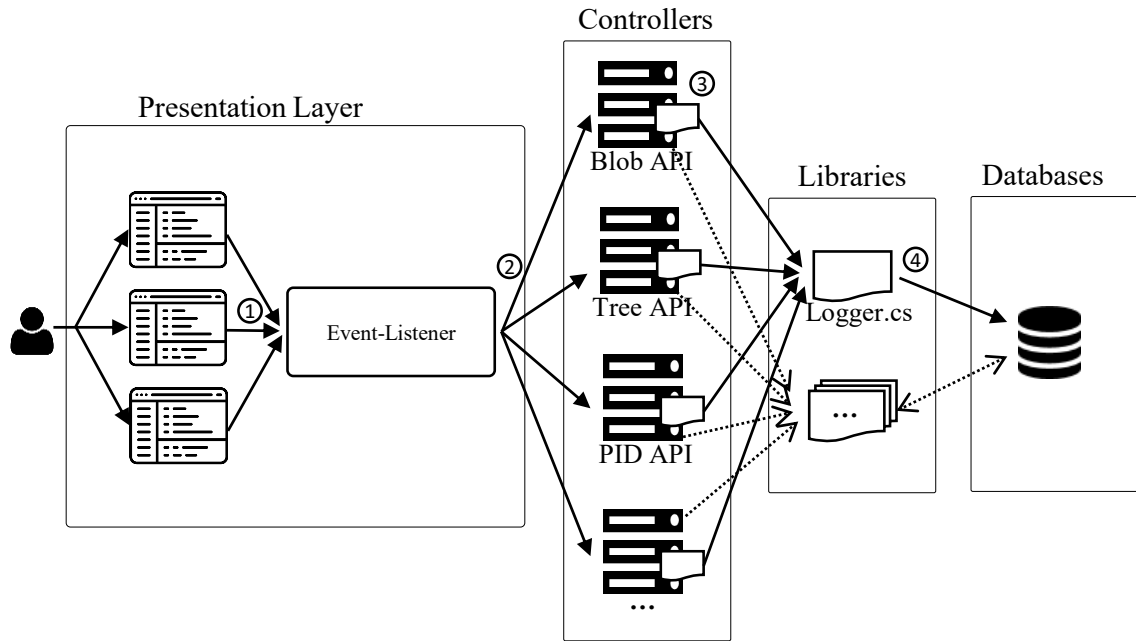


Figure 3.3: The diagrammatic illustration of the workflow of the Server-Side logger.

### Approach

The Server-Side logger comprises two main components: a Logger library and API endpoints. The Logger library defines log objects and their types, configurations, and level layouts. Listing 3 provides an example of logger object definition with three main attributes Timestamp, Activity, and Case. Note that a Case is an object with additional attributes that may or may not be present depending on an Activity.

The following is a stepwise description of acquiring events from the Server-Side logger represented in Figure 3.3:

- 1- User interaction with a web application executes a request.
- 2- The Event-Listener is responsible for receiving requests and triggers asynchronous calls to responsible APIs.
- 3- The logger object is constructed from existing attributes only after processing the user request, ensuring complete processing of the user request.
- 4- The Logger.cs operate the configuration and object definitions to store the event in its respective table as a serialized JSON object.

As mentioned earlier, a Case is an object defined in Listing 3, and according to its definition, a Case object can not be null despite the possibility of missing specific attributes due to non-existing values at the time of execution. For example, Listing 4

```

1
2 public AnalyticsLogObject(string activity, DateTime timestamp, string
   ↪  userId, string roleId, string projectId, string resourceId, string
   ↪  fileId, string applicationsProfile)
3 {
4     Timestamp = string.Format("{0:yyyy-MM-dd HH:mm:ss.fff}",
   ↪  timestamp.UtcNow);
5     Activity = activity;
6     this.Case = new Case();
7 }
8 public Case(string userId, string roleId, string projectId, string
   ↪  resourceId, string fileId, string applicationsProfile)
9 {
10     UserId = userId;
11     RoleId = roleId;
12     ProjectId = projectId;
13     ResourceId = resourceId;
14     FileId = fileId;
15     ApplicationsProfile = applicationsProfile;
16 }
17

```

Listing 3: Server-Side logger library, instantiating an event object.

is an endpoint POST method call for adding a resource, and subsequently, several attributes such as ResourceId and FileId are unavailable while creating a resource. Table 3.4 demonstrates an example of an event log created by a few activities and their corresponding accessible attributes. Note that the timestamp is constructed by the AnalyticsLogObject library (Listing 3) to avoid mismatch.

## Preliminaries

The following is a formal model used for the Server-Side logger. An example of an event log derived from the Server-Side logger is shown in Table 3.4.

**Definition 3.3. (Server-Side Event, Trace, Event Log).** Let  $\xi$  be the universe of all possible event identifiers, and  $\xi^*$  is the set of all finite sequences over  $\xi$ . For every Server-Side event, there are standard attributes such as *Timestamp*, *Activity*, and *Case*.

Every event is characterized by various attributes corresponding to an *Activity* and *Timestamp*. Let  $\Lambda_{e_s}$  be the set of all possible event attributes. For any  $e_s \in \xi$  and attribute  $att \in \Lambda_{e_s}$ ,  $F_{att}(e_s)$  is the value of attribute  $att$  for event  $e_s$ . If an event  $e_s$

```
1
2 [HttpPost("[controller]/project/{projectId}")]
3 public async Task<IActionResult> addResourceToProject(string projectId,
4     ↪ ResourceMeta resourceMeta)
5 {
6     ....
7     await AddResource(projectId, resourceMeta)
8     ...
9     LogAnalytics(user, projectId, resourceMeta);
10    return Json(resourceReturnObject);
11 }
12 private void LogAnalytics(User user, Guid projectId, ResourceMeta
13     ↪ resourceMeta)
14 {
15     _coscineLogger.AnalyticsLog(
16         new AnalyticsLogObject
17         {
18             Activity = "Add Resource",
19             CaseId.UserId = user.Id.ToString(),
20             CaseId.ProjectId = projectId.ToString(),
21             CaseId.RoleId = user.Role,
22             CaseId.ApplicationsProfile =
23                 ↪ resourceMeta.ApplicationsProfile,
24         });
25 }
```

Listing 4: An example of API endpoint for *Add Resource* operation.



Table 3.4: An example of events ( $e_s$ ) generated from a Server-Side logger.

Timestamp ( $t_s$ )	Activity ( $a_s$ )	C.UserId ( $c_s^1$ )	C.RoleId ( $c_s^2$ )	C.ProjectId ( $c_s^3$ )	C.ResourceId ( $c_s^4$ )	C.FileId ( $c_s^5$ )	C.AppProfile ( $c_s^6$ )
...	...	...	...	...	...	...	...
1579108918	Add Member	29613-d8...	be29c-4e...	4b15f...	NULL	NULL	NULL
1579109840	Resource Create	29613-d8...	be29c-4e...	4b15f...	NULL	NULL	EngMeta
1579129397	Upload File	29613-d8...	be29c-4e...	4b15f...	4e9f-97...	PID/FileName.csv	EngMeta
1579835452	Archive Resource	8G7s6-c2...	be29c-4e...	4b15f...	4e9f-97...	NULL	Radar
...	...	...	...	...	...	...	...

does not have an attribute, then  $F_{att}(e_s) = null$ . For example,  $\Upsilon_{t_s}$  is denoted as the universe of timestamps, and subsequently,  $F_{t_s}(e_s) \in \Upsilon_{t_s}$  is the timestamp of event  $e_s$ .

Case Id is an object with accessible features in the corresponding API Controller. It contains required (e.g., **userId**) and optional (e.g., **FileId**) elements. Let  $C$  be the universe of all cases, and  $\mathcal{N}_{case}$  be the set of case attribute names. For any case  $c^i \in C$  and name  $n \in \mathcal{N}_{case}$ :  $\#(c^i)$  is the value of attribute  $n$  for case  $c^i$ , and is an index for each caseId. For a case  $c$ , if  $c$  does not have an attribute named  $n$ , then  $\#_n(c) = null$ . Each case object has a mandatory attribute name **userId** such that  $\#_{userId} \in \xi^*$ .

A trace  $\sigma$  is a finite sequence of events such that each event appears only once and  $1 \leq i < j \leq |\sigma| : \sigma(i) \neq \sigma(j)$ . An event derived from Server-Side logger is denoted by  $e_s$  and is a n-tuple  $e_s = (t_s, a_s, obj_s)$ , and  $e_s \in L_s$ . Where  $t_s \in T$  is the event timestamp for a Server-Side activity,  $a_s \in A$  is the activity corresponding to an API endpoint call within a controller, and  $obj_s \in C$  is an object with a list of possible case Ids.

## Discussion

The methodology and the acquired event logs for the Server-Side logger have several strengths and weaknesses that are also reflected in Table 3.1. The major strength of this method is the potential to facilitate monitoring a wide range of servers at once and produce comprehensive logs. However, another issue with this method is that it can be time-consuming and error-prone to set up and configure.

The event logs derived from Server-Side loggers can be complex, diverse, and occasionally inaccurate, making the data cleansing task challenging and complex. In addition, if not programmed carefully and the logger is not triggered at the right time within the API controller, loggers can also capture events that may not have been processed, thus becoming a threat to the validity and accuracy of logs. Therefore, to use event logs effectively, it is essential to be aware of these issues and take steps to encounter them.

Ensuring the completeness of an RDM system's logging is essential, as missing important events can lead to significant problems while analyzing logs. Unfortunately, a significant manual cognitive effort is necessary for developers to continue extending

API endpoints, and as a system grows, this can become more challenging and quickly become outdated. As a result, the acquired dataset may include missing events or attributes without one's knowledge. Significantly, this can become more visible if extending or refactoring an API endpoint does not contain necessary attributes or simply if the logger is executed before successfully processing user requests, thus resulting in misleading event records. Unlike the Client-Side logger, relying on the backend library to instantiate the timestamps eliminates the need for a Client-Side logger and the potential for capturing sensitive information.

The uniqueness of events in this logging method is an integral aspect of event-driven systems that allows for adapting the notion of a case according to a question-driven study. This flexibility supports adapting the notion of a case via post-processing the logs according to a question in hand and provides multiple detailed views on a stream of events. Additionally, including multiple cases per event allows for divergence and convergence of data from multiple data repositories crucial for a distributed system by using various case attributes as relational identifiers, thus maintaining the consistency required by an event log. It enables data scientists to consider different case notations using the same event log without losing the order of activities. Finally, despite the significant challenges for the scalability of capturing different activities, my approach opens opportunities for various object-centric data mining projects and the discovery and elicitation of system operational KPIs.

## 3.4 Proposed Key Technique

In order to discover the descriptive user activities and their corresponding underlying infrastructure operations, a key technique is proposed for acquiring data that, according to the studies, is the most efficient, reliable, and least expensive. In this methodology, besides acquiring sufficient data to study the dynamic system behaviors, the aim is to choose an approach that has the most negligible impact on the efficiency of the SUS. This is achieved by creating a Hybrid logger that carries benefits from logging of *Tier 0* and *5* of the RDM architecture through minimal instrumentation of the centralized (OAuth workflow) and Client-Side logging, respectively.

This approach allows for overcoming the shortcomings of each logging technique. On the one hand, relying only on the OAuth workflow as a centralized logger yields uncertainty over the accuracy and completeness of event logs. On the other hand, a Client-Side logger represents a high accuracy of actual events being executed and ensures completeness by capturing actual user activities. Additionally, unlike the Client-Side logger, data acquired by the centralized logger can handle the out-of-order arrival of events due to the reliability of timestamps generated by the server, consequently discovering a valid lifecycle for the execution of events.

The proposed methodology and findings were previously presented in Yazdi et al. [25].

#### 3.4.1 Methodology

This Section introduces a novel approach for reannotating centralized events with their corresponding Client-Side activities. This uses Client-Side logs as a training set, OAuth service as a centralized logging service to build a correlation between responding resources, and machine learning to learn to model and classify user activities by only acquiring OAuth logs from the distributed environment.

According to the proposed technique, one can create a training set in a controlled environment, establish a training model, and use a real-world dataset from services in production to reannotate and map server activities to their corresponding user activity with high accuracy.

Initially, the description begins with the approach to capture logs from web services and then further describes the algorithm used for the Event Log Generator (ELG) to support the tasks of data cleansing, merging event data, and predicting user activities.

#### Approach to Capture Logs

Figure 3.4 provides a schematic overview of the process for capturing event logs using both Client and OAuth logging services. As discussed in Section 3.3.1, the OAuth logging services provide information to help trace errors and access resources within the distributed environment. Initially, extending the current implementation of the token service is necessary to include information about the requested resources, the methods used, and a *correlation identifier*. Furthermore, the OAuth logging service provides an *Append Log* endpoint that allows appending various attributes such as a *correlation identifier* to the collected logs to pinpoint user requests and their corresponding accessed resources within the connected systems [165]. Therefore, the *Append Log* endpoint is adopted and the `message` attribute is extended with a counter as the *correlation identifier*. This approach helps to track and identify events that belong to a unique process instance.

The following is a detailed description of the execution of processes within the Hybrid-logger demonstrated in Figure 3.4:

1. User attempts to utilize an RDM service, and the OAuth interface acquires the user credentials.
2. OAuth service checks the credentials and allocates an `accessToken` after evaluating the user's authorization.
3. With every user activity on the front end, the user triggers Client-Side activities that utilize an `accessToken` as an authentication parameter and passes it to the Event-Listener for executing asynchronous API calls.

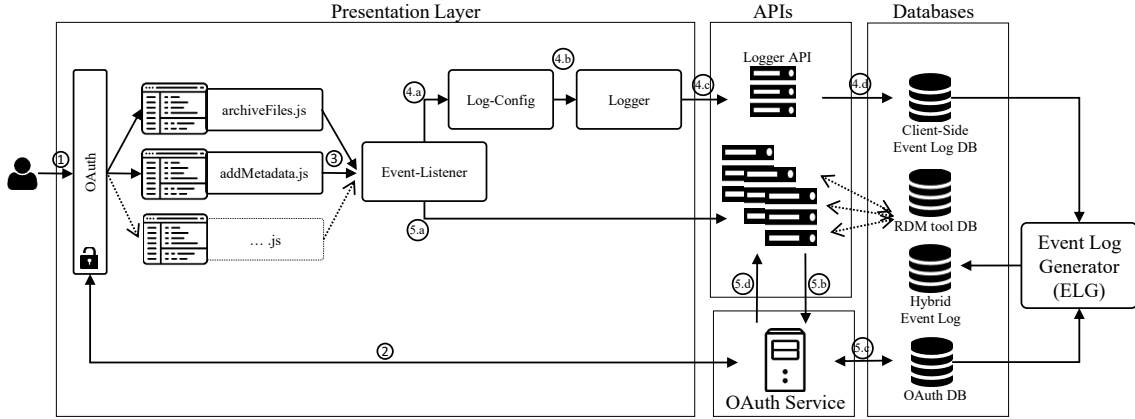


Figure 3.4: The schematic representation of the Hybrid-logger.

#### 4. Client-Side Logger

- a) The *Log-Config* serves as a configuration flag to determine enabling/disabling Client-Side logger.
- b) The logger creates a global javascript counter  $\geq 1$  within the logging library, which increments each time a user triggers an event.
- c) The logger adds a query parameter `_rid={counter}` while calling the Logger API.
- d) The Logger API is then responsible for storing the event within its respective database table or instance.

#### 5. OAuth Logger

- a) The Event-Listener forwards the user request to its corresponding API endpoint.
- b) APIs verify the user's authorization with the OAuth service before processing a request.
- c) The OAuth logging service stores every executed event that requested authentication with a counter value  $\geq 0$  in its database table or instance.
- d) After verification of the access token, the requested resources and methods in the APIs execute.

6. ELG fetches, cleans, and merges logs from Client-Side and OAuth databases and stores the newly generated logs in the Hybrid event log database after reannotating user activities.

One needs to note that, in this approach, the Client-Side and OAuth loggers remain two asynchronous and independent logging systems that execute parallelly. In this method, the counters are instantiated as *correlation identifier* with a different starting value. Such that the Client-Side counter starts with 1, and the OAuth counter starts with 0. The distinction between the instantiation of the two counters is due to the execution of events upon web page first loadings that correspond to `counter=0` on OAuth logs. On every further user activity on the Client-Side, the counter value will match its corresponding OAuth counter, which, alongside the user access token, can be used later as a *correlation identifier* to uniquely identify a process instance. The counters reset with every page refresh.

## Event Log Generator

Event Log Generator (ELG) is a method for predicting users' activities based on previous training sets of corresponding OAuth executions; it is responsible for running an algorithm for converging event logs from Client-Side and OAuth log databases. The ELG includes seven steps, including data cleansing, dataset merging, mapping, training a model, and applying that model to provide us with a predicted user activity based on corresponding method calls captured by the OAuth workflow. Figure 3.5 provides an overview of example logs and how they transform within the ELG method. In addition, formal descriptions of the following steps are provided in Section 3.4.2.

### Step 1) Client-Log Refinement (CLR)

Client-logs refinement provides data cleansing, filtering, and merging of attributes such as activity and webpage to further distinguish similar named activities with their corresponding webpage where an activity occurs.

### Step 2) Merging and Mapping (MM)

Merging and mapping is a function to map events using `caseIds` and `identifiers` to merge client activity and corresponding Server-Side activities. For example, in case of the OAuth identifier resets to 0, the mapping function replaces the missing client activity with *page loading* as it does not correspond to any particular user activity.

### Step 3) Merged Log to Matrix Conversion ( $L_m \mapsto M_m$ )

In order to enable machine learning to train models and predict client activities, the event log ( $L-m$ ) is converted to a matrix ( $M_m$ ). The first column in the matrix ( $M_m$ ) includes Client-activity( $a_c$ ) and indicates a target/label with a categorical type; every other feature/attribute represent OAuth-Side methods in the binominal type where execution(s) of every software component is denoted with either 0 or 1.

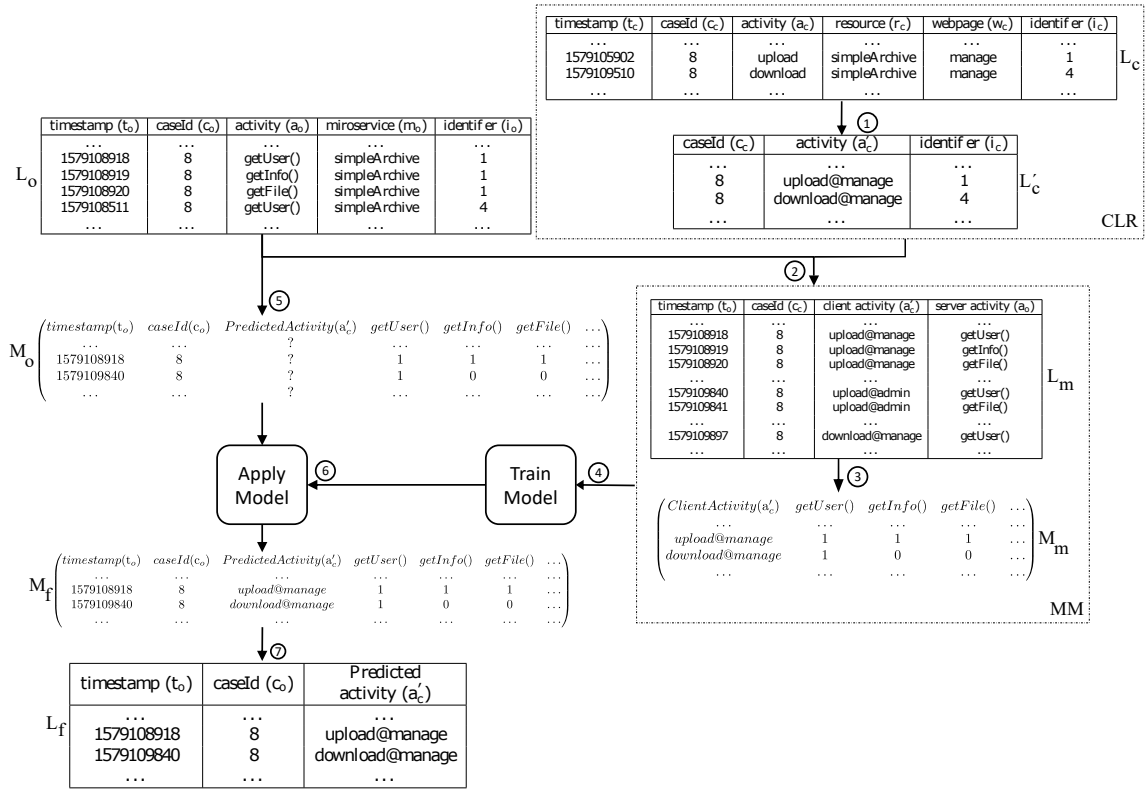


Figure 3.5: A schematic representation of the Event Log Generator and transformation of an example datasets. Adapted from [25].

*Step 4) Train Model ( $\models$ )*

A matrix ( $M_m$ ) is used to create a predictive model as a training set. Various predictive modeling algorithms such as Random Forest, Decision Tree, Deep Learning, etc. can be utilized.

*Step 5) OAuth Log to Matrix Conversion ( $L'_o \mapsto M_o$ )*

Similar to step 4, the OAuth event log ( $L'_o$ ) is converted to the matrix ( $M_o$ ) to prepare it for predicting labels according to features. The matrix ( $M_o$ ) contains features/attributes corresponding to Server-Side methods. The `caseId` ( $c_o$ ) and `identifiers` ( $i_o$ ) help uniquely determine a set of activities executed and thus create a corresponding row in the matrix  $M_o$ . Furthermore, the matrix withholds the additional attributes for timestamp( $t_o$ ) and `caseId` ( $c_o$ ) to facilitate step 8 in the approach.

*Step 6) Apply Model ( $M_o \models M_f$ )*

The approach uses the trained model from step 5 and the matrix ( $M_o$ ) to predict activities for every row based on occurrences of Server-Side software calls and annotate every row with its predicted activity ( $a_c$ ). Storing it as a separate Matrix ( $M_f$ ) with predicted values.

*Step 7) Matrix to Event Log Conversion ( $M_f \mapsto L_f$ )*

The additional columns like `timestamp` and `caseId` from  $M_f$  provide the essential ingredients to reconstruct the event log with necessary attributes and enable the discovery of processes using the final event log ( $L_f$ ). This step's expected output event log is  $e_f = (t_o, c_o, a'_c)$ . Therefore, we denote  $\xi_f = T_o \times C_c \times A'_c$  as the universe of all final event logs.

By employing the methodology mentioned above and my case studies findings in Section 5.2.3, acquiring the training set can be performed in a controlled environment, eliminating the need for the Client-Side log from a production server. Thus, after training the model, steps 5, 6, and 7 can be executed as a pipeline to continuously generate event logs with high accuracies while solely relying on the OAuth event log  $L_o$ .



### 3.4.2 Preliminaries

This Section provides a basic concept and formal models for the ELG approach. It begins with a definition of a set, event, event log, and trace, then provides a formal explanation of Log Refinement, Merging and Mapping, and Dataset Conversion.

**Definition 3.4. (Set)** Given a set  $A$ ,  $\sigma = \langle a_1, a_2, \dots, a_n \rangle \in A^*$  is the set of all finite sequence over the  $A$ .  $\sigma(i) = a_i$  denotes the  $i^{th}$  element of the sequence.  $\beta(A)$  is the set of all multi-set over set  $A$ .  $P_{NEA}$  is the set of all non-empty subsets over the set  $A$ .  $|\sigma| = n$  is the length of  $\sigma$ . For  $\sigma_1, \sigma_2 \in A^*$ ,  $\sigma_1 \subseteq \sigma_2$  if  $\sigma_1$  is a sub-sequence of  $\sigma_2$ . e.g.,  $\langle g, b, o \rangle \subseteq \langle z, g, b, b, g, b, q, o \rangle$ . For  $\sigma \in A^*$ ,  $\{a \in \sigma\}$  is a set of elements in  $\sigma$ . For  $\sigma \in A^*$ ,  $[a \in \sigma]$  is a multi-set of elements in  $\sigma$ , e.g.,  $[a \in \langle g, b, z, g, b \rangle] = [g^2, b^2, z]$ . For multi-set  $X \in \beta(A)$ ,  $X(a)$  denotes number of times the element  $a \in A$  appears in  $X$ .

**Definition 3.5. (Event, Event Log)** A Client-Side  $e_c$  and OAuth-Side  $e_o$  events are both tuples defined as  $e_c = (t_c, c_c, a_c, r_c, w_c, i_c)$  and  $e_o = (t_o, c_o, a_o, m_o, i_o)$  respectively. Where  $t_c, t_o \in T$  are the client-side and OAuth-Side event timestamps respectively, but  $t_c \not\leq t_o$  as the timestamps in Client-Side may be different from its respective event's timestamp on the OAuth-Side,  $c_c, c_o \in C$  are the Client-Side and OAuth-Side case ids respectively which are nothing but the **accessTokens** generated by OAuth workflow and uniquely identifiable  $C_c \subseteq C_o$ .  $a_c, a_o \in A$  are the Client-Side and OAuth-Side activities respectively, such that  $A_c \cap A_o = \emptyset$ .  $r_c, m_o \in R, M$  is the resource, or the microservice names shared between Client-Side and OAuth-Side event logs, i.e.,  $R_c \subseteq M_o$ .  $w_c \in W$  is the web page an activity occurs.  $i_c, i_o \in I$  are the correlation identifiers that map Client-Side activities to corresponding OAuth-Side activities. Between OAuth-Side and Client-Side events, this identifier is instantiated such that  $i_o = 0$  and  $i_c = 1$ , and the values get incremented with every user action executed over the web page.  $\xi_c = T_c \times C \times A_c \times R \times W \times I$  denotes as the universe of all Client-Side events and  $\xi_o = T_o \times C \times A_o \times M \times I$  as the universe of all OAuth-Side events. A Client-Side event log is  $L_c \subseteq \xi_c$ , and an OAuth-Side event log is  $L_o \subseteq \xi_o$ . In event logs, each event can appear only once. Hence, events are uniquely identifiable by their attributes, and no special start and end activities exist.

**Definition 3.6. (Trace)** Let  $x \in \{c, o\}$ ,  $\sigma_x = \langle e_{x_1}, e_{x_2}, \dots, e_{x_n} \rangle$  is defined as a trace in client or OAuth-Side event logs, s.t., for each  $e_{x_i}, e_{x_j} \in \sigma_x$ ,  $1 \leq i < j \leq n: \pi_C(e_{x_i}) = \pi_C(e_{x_j})$  and  $\pi_T(e_{x_i}) \leq \pi_T(e_{x_j})$ .

### Log Refinement

To initialize the process, first, it begins with refinement of event logs and using a subset of the dataset from Client-Side and OAuth event logs.

*Client-Log Refinement (CLR)*: A refined Client-Side event is a tuple  $e'_c = (c'_c, a'_c, i'_c)$ . we denote  $\xi'_c = C \times A_c \times I$  as the universe of all refined Client-Side events. Due to the unreliability of Client-Side timestamps, we dismiss  $t_c$  from the rest of the procedure. In the system under study, we discovered that similar activity names from different web pages could result in different software execution behavior. As an example,  $(a_c = (\text{"upload"}) \in w_c = (\text{"manage"})) \neq (a_c = (\text{"Upload"}) \in w_c = (\text{"admin"}))$ , Therefore for every OAuth-Side activity,  $w_c$  is concatenating with  $a_c$  to form a unique name per web page.  $A_c \rightarrow W$  is an injective function that maps a webpage to a set of Client-Side activities. (Ex.  $a'_c = \text{"Upload@manage"}$ ), then  $w_c$  is removed from  $L'_c$ . Hence,  $CLR^r_c \in \xi_c \rightarrow \xi'_c$  is defined as the Client-Side event log refinement function and  $r \in R$ . For  $CLR^r_c(L_c) = L'_c$ , s.t.,  $L'_c = \{e'_c \in \xi'_c \mid \exists e_c \in L_c \pi_R(e_c) = r \wedge \pi_C(e'_c) = \pi_C(e_c) \wedge \pi_{A'_c}(e'_c) = (\pi_{A_c}(e_c), \pi_W(e_c)) \wedge \pi_I(e'_c) = \pi_I(e_c)\}$ .

### Merging and Mapping

The log labeling at this stage is a one-time process to create a merging between Client-Side activities and OAuth-Side activities. The output of event logs refinements ( $L_o$  and  $L'_c$ ) are used as inputs to this step and marked as  $L_m$  in Figure 3.5.

*Mapping Logs*  $(L_o, L'_c) \mapsto L_m$ : The refined event log outputs ( $L_o$  and  $L'_c$ ) are utilized as inputs for this step, labeled as  $L_m$  in Figure 3.5, and denoted as  $(L_o, L'_c) \mapsto L_m$ . The expected output event log from this step is  $e_m = (t_o, c_c, a'_c, a_o)$ . We denote  $\xi_m = T_o \times C_c \times A'_c \times A_o$  as the universe of all merged and mapped OAuth-Side and Client-Side events.  $mm \in 2^{\xi_o} \times 2^{\xi'_c} \mapsto 2^{\xi_m}$  is defined as the merging and mapping function. The log merging is instantiated per Client-Side case  $c_c \in L'_c$ . For each Client-Side caseId  $c_c$ , we check the respective  $i_o$  and  $i_c$  to build the relationship between the two logs.  $ma: A_c \cup \{\perp\} \mapsto 2^{A_o}$  is a function that maps a Client-Side activity to a set of OAuth-Side activities. If the value of  $i_o > 0$ , we check for the respective Client-Side activity  $a'_c$ , i.e.,  $\pi_{A'_c}(e'_c) = ma(\pi_{A_o}(e_o))$ , thus, extending every OAuth-Side event with a respective Client-Side activity  $a'_c$ . Note that  $A'_c \cup \{\perp\}$ , and  $ma(\perp) = \emptyset$ , so, if  $i_o = 0$ , we consider the  $a'_c$  label as "PageLoading". The output of  $(L_o, L'_c) \mapsto L_m$  is an event log  $L_m \subseteq \xi_m$  where  $\xi_m \subseteq (\xi'_c \cup \xi_o)$ . Hence  $mm(L_o, L'_c) \mapsto L_m$ , s.t.,  $L_m = \{e_m \in \xi_m \mid \exists e_o \in L_o, \exists e'_c \in L'_c, \pi_C(e_o) = \pi_C(e'_c) \wedge \pi_T(e_m) = \pi_T(e_o) \wedge \pi_C(e_m) = \pi_C(e'_c) \wedge \pi_{A'_c}(e_m) = \pi_{A'_c}(e'_c) \wedge \pi_{A_o}(e_m) = \pi_{A_o}(e_o)\}$ .

### Dataset Conversion

$L_m \mapsto M_m$ : We know that,  $\sigma_m = \langle e_{t_1}, e_{t_2}, \dots, e_{t_n} \rangle$  is a trace within  $L_m$ , such that each  $e_{t_i}, e_{t_j} \in \sigma_m$ , thus  $L_m$  is sorted by  $T$  s.t.,  $t_i \leq t_j \leq t_n$ . Given a set of features  $\Upsilon$  in  $L_m$ ,  $v = \langle a_{o_1}, a_{o_2}, \dots, a_{o_n} \rangle$  and  $v(i) = a_{o_i}$  denotes the  $i^{th}$  element of sequence

$v$ .  $|v| = n$  is the length of  $v$ , thus determining the number of features for the matrix  $M_m$ . we define a function mapping  $row_m : a'_c \wedge \pi_{a_o}(\sum_{a_{o_0}}^{|v|} \forall(a'_c \exists a_o))$ , so that for every caseId  $c_c$  while  $a'_c$  is unchanged, the algorithm creates a new row in  $M_m$  and indicate the existence of attributes  $a_o$  for each label  $a'_c$ . For the matrix  $M_m$ , we expect to get one column as a categorical label and features equivalent to the number of unique OAuth-Side activities  $|v|$ . Hence,  $L_m \mapsto M_m = \{e_m \in \xi_m | \exists row_m(L_m)\}$ .

$L_o \mapsto M_o$ : the OAuth event log ( $L_o$ ) is converted to the matrix ( $M_o$ ) to prepare it for label prediction based on attributes. The matrix ( $M_o$ ) contains attributes related to software components. The caseId ( $c_o$ ) and identifiers ( $i_o$ ) aid in uniquely identifying a set of executed activities, thereby creating a corresponding row in the matrix  $M_o$ . Moreover, the matrix retains additional attributes for timestamp( $t_o$ ) and caseId ( $c_o$ ) to facilitate step 7 of the approach. Accordingly, with respect to definition Definition 3.6,  $\sigma_o = \langle e_{t_1}, e_{t_2}, \dots, e_{t_n} \rangle$  is a trace within  $L_o$ , such that each  $e_{t_i}, e_{t_j} \in \sigma_o$ , thus  $L_o$  is sorted by  $T$  s.t.,  $t_i \leq t_j \leq t_n$ . Given a set of features  $\Upsilon$  in  $L_m$ ,  $v = \langle a_{o_1}, a_{o_2}, \dots, a_{o_n} \rangle$  and  $v(i) = a_{o_i}$  denotes the  $i^{th}$  element of sequence  $v$ .  $|v| = n$  is the length of  $v$ , thus determining the number of features for matrix  $M_o$ . We define a function mapping  $row_o : t_o \wedge c_o \wedge \pi_{a_o}(\sum_{a_{o_0}}^{|v|} \forall(i_o \exists a_o))$ , so that for every caseId  $c_o$  while  $i_o$  is unchanged, the algorithm creates a new row in  $M_o$ . For the matrix  $M_o$ , we expect to maintain attributes for timestamp  $t_o$ , caseId  $c_o$  and features equivalent to the number of unique OAuth-Side activities  $|v|$ . Hence,  $L_o \mapsto M_o = \{e_o \in \xi_o | \exists row_o(L_o)\}$ .

### 3.4.3 Discussion

The proposed technique is positioned for each criterion according to the criteria for analyzing a distributed system using dynamic information retrieval strategies discussed in Section 3.2. Note that the findings have been previously partially published by Yazdi and Politze [25].

*Information Source:* The token service's implementation of the OAuth workflow provides a secure way to authorize web services and manage users' access without providing their credentials [185] or to rely on users' unique identifiers.

*Application Layer:* The token service provides a language-independent way to inter-communicate between software components via different interfaces, making it easy to use, maintain, and scale across the SUS.

*Correlation of Distributed Events:* The findings demonstrate that tracing sequences of events occurring between different software components can be promising. In order to achieve this, a non-intrusive approach is taken to collect enough data from all services without interfering with legacy code or modifying source code manually. Furthermore, it is argued that instrumenting the access token identifier generated by the token service, the access token, and the correlation identifier are reliable candidates for the correlation of events in SUS despite the involvement of common components across all services.

*Sequence Order:* It is crucial to capture the ordering of events in which user requests occur, as executing several of them simultaneously is possible. This method also allows us to run conformance checking by capturing the sequence of events executed to fulfill user requests and comparing them to the standard sequence order.

*Communication Type:* Multiple threads of activity involving several microservices may be executed responding to a single resource request. These activities that require authorization and token validation are nevertheless captured and stored. All instances of a token used by multiple services can be captured and analyzed, regardless of the number of microservices associated with a single user request.

*Performance:* It is an essential indicator for effectively tracking a system's performance and discovering process bottlenecks at different levels of granularity. In addition, the token service generates timestamps for each log entry, allowing us to analyze the performance of the SUS in responding to user requests and identifying possible bottlenecks.

*Data Granularity:* The logging system captures high-level data, such as the microservice in demand, and low-level data, such as the method calls (software components) involved in responding to a user request. The merging and mapping of an event log from the Client-Side assist in gaining an appropriate level of granularity corresponding to actual user activities. Moreover, the OAuth logger can capture additional

information to enable other contextual analyses, such as the Role of a user, allowing for organizational mining.

*Target model:* This methodology enables studying process and data models with precise and unambiguous semantics using process mining or machine learning algorithms.

## Summary

In this Section, an essential technique is proposed for acquiring datasets according to the reference architecture for RDM services. The instrumentation of OAuth workflow to produce datasets alongside mapping and merging of token service (OAuth) logger with Client-Side logger as a training set for building a predictive machine learning model provides an opportunity to discover and enhance user and system behavior within a distributed system. Furthermore, this approach fulfills the requirement for extracting logs that comply with GDPR by introducing ELG methodology to interpret logs and provides meaning for software component execution lifecycles without a need for placing cookies or tracking every user action explicitly.

However, there are several benefits and shortcomings to the suggested approach. For instance, the OAuth workflow only captures traces of events that require authorization; as a result, the acquired dataset lacks events that do not require OAuth service. In addition, Client-Side logging naturally increases network bandwidth overhead due to posting events with user activities; using the ELG method, on the other hand, eliminates the issue by relying on an up-to-date training set created in a controlled environment and disabling the Client logger in production servers. Thus, the training set must be updated once new functionalities are added to an OSP front end or a logic to process a user request changes. Moreover, one needs to assess the performance of the ELG algorithm with a higher number of Client-Side activities, as it may highly influence the accuracy of predicted user activities. Also, such a technique is found appropriate for systems that have already achieved a level of maturity and are not rapidly adjusting software components.

In these case studies, it was discovered that the ELG algorithm is sensitive toward a set of client activities that trigger similar function calls, which may result in predicting inaccurate client activities. For example, software components responding to *Page Loading* are very similar to *Page Redirecting*, but this issue is resolved using the suggested approach for data abstraction in Section 5.2. Moreover, noise was identified in the OAuth logs that included a few incorrectly logged events on the OAuth workflow due to the dispersed and heterogeneous nature of a distributed system, and this issue could be overcome by mapping OAuth events to their corresponding Client-Side events.

Employing the Hybrid logger and the ELG algorithm makes it possible to discover and exclude activities executed by bots or internal nightly operations within a dis-

tributed system. Additionally, the OAuth protocol logs authorization operations from all resources connected to a system. Hence one can create a training set and analyze processes in other resources retrospectively in a distributed setting.

## 4 Data Analytics Framework

The Data Analytics Framework aims to provide a comprehensive approach to extracting valuable insights and understanding complex relationships within datasets, which are crucial for informed decision-making and system optimization. This framework encompasses methodologies for data abstraction, modeling process-aware activities, discovering the RDM lifecycle, and enhancing data collection reusability. By integrating these methodologies, the framework facilitates the development of robust and interpretable models that capture the underlying structure and dynamics of various data-driven processes, ultimately leading to more effective strategies for managing and improving software applications and services.

Data abstraction is a critical framework component, as it simplifies and consolidates raw data into more manageable and meaningful representations. Furthermore, this process aids in feature engineering, paving the way for more accurate and interpretable analysis and data modeling. On the other hand, modeling process-aware activities focuses on understanding and presenting user behavior on OSPs based on observed interactions within a system. Likewise, discovering the RDM lifecycle entails uncovering the intricate relationships and dependencies between different stages of RDM, allowing for better coordination and alignment of resources and efforts. Lastly, the reusability enhancement methodology aims to improve the overall utility and accessibility of datasets by identifying opportunities for reuse, repurposing, and integration, ultimately promoting a more efficient and collaborative research ecosystem through appropriate recommender systems.

### 4.1 Semi-Supervised Data Abstraction

The demand for digitalization has increased the number of novel software solutions designed to address various needs. This constant evolution has led to the creation of intricate software systems with multifaceted internal processes. For example, institutions of higher learning utilize decentralized software systems to facilitate RDM, ensuring that data is easily discoverable, accessible, and reusable [21], [26]. This results in a diverse and dispersed IT infrastructure with a labyrinth of user processes. To address this challenge, it is necessary to model and investigate researchers'

processes to enable domain experts to comprehend and oversee user behaviors and unearth underlying operational procedures. Such process-centric modeling facilitates identifying obstacles researchers encounter throughout their research lifecycle [23]. To scrutinize user dynamics and oversee operational procedures, process mining techniques are employed to uncover process models [22].

Software systems often operate within complex infrastructures, where a user request may necessitate the involvement of multiple software components and microservices to execute a task. This complexity often leads to a  $n:m$  relationship between Server-Side and Client-Side events that are highly detailed. Applying process discovery on low-level event logs typically yields unstructured process models, known as “Spaghetti” models, which are unsuitable for process analysis [54]. Consequently, various event log abstraction techniques have been developed to convert fine-grained (low-level) event logs into more abstract (high-level) representations. However, the primary challenge lies in identifying an abstraction level that is both interpretable by domain experts and accurately reflects the reality of business activities.

Number of methods have been proposed to address the challenge of abstracting low-level events (and are discussed in Section 1.2.4). Some existing methods rely on unsupervised learning techniques to cluster groups of events into higher-level events. However, these unsupervised learning methods struggle to achieve an appropriate level of abstraction, often resulting in unclear relabeling strategies for the abstracted event clusters. Furthermore, these learning techniques either necessitate manual labeling by domain experts or automatic concatenation of labels, leading to lengthy and unintelligible activity names. In contrast, several supervised learning methods have been developed to surmount these challenges. These methods either utilize training sets to guide the abstraction process or depend on domain experts’ input to relabel activity names. Thus, relabeling abstracted activity names remains a complex task. The challenges mentioned above underscore the need for an approach that effectively abstracts low-level event logs to a suitable higher level of granularity.

In this section, an iterative semi-supervised abstraction method for Client-Server applications is presented in which Client-Side event logs serve as the training set for characterizing Server-Side event logs. At the same time, the Pearson Coefficient Correlation (PCC) is utilized to assess pairwise event similarities to abstract activities. The approach outlined in this study builds upon the method proposed in [24] for obtaining Server-Side event logs. As a result, employing the OAuth workflow to capture Server-Side event logs. *The Client-Side event logs function as descriptive user activities that are comprehensible to domain experts, thereby facilitating the mapping of low-level Server-Side event logs to high-level activities.*

The following Table 4.1 offers a synopsis of how the research aligns with the literature review explored in the Section 1.2.4. Each criterion is comprehensively described in



Section 1.2.4. The insights and outcomes discussed in this Section have been partially documented and shared in an earlier publication, as referenced in [27], [186].

Table 4.1: An overview of the proposed methodology with respect to data abstraction criteria discussed in Section 1.2.4.

	<i>Grouping</i>		<i>Input Data</i>		<i>Event Perspective</i>	<i>Mapping Relationship</i>	<i>Internal Abstraction</i>	
	<i>Su.</i>	<i>Un.</i>	<i>Co.</i>	<i>Di.</i>			<i>Pr.</i>	<i>De.</i>
Proposed Approach	✓		✓	✓	Non-Sequential	n:m		✓

	<i>Validity</i>	<i>Quality Indicator</i>	<i>Target Domain</i>
Proposed Approach	Real-Life	Fitness, PCC	Client-Server Applications

Most authors have employed the IM algorithm to discover process models. The IM algorithm can address infrequent behaviors while maintaining the soundness of the discovered process model [121]. Furthermore, different versions of Petri Nets notations have been utilized to represent the discovered model. However, a review of Table 1.2 reveals that none of the existing approaches are suitable for implementation within current infrastructures, as they rely on internal probabilistic abstraction techniques or demand extensive domain knowledge to assist in relabeling abstracted activities.

### 4.1.1 Conceptual Foundations

This section presents the fundamental concepts and formal models employed throughout the research.

Given a set  $A$ ,  $\sigma = \langle a_1, a_2, \dots, a_n \rangle \in A^*$  is the set of all finite sequence over the  $A$ .  $\sigma(i) = a_i$  denotes the  $i^{th}$  element of the sequence.  $\beta(A)$  is the set of all multi-sets over set  $A$ .  $P_{NE}(A)$  is the set of all non-empty subsets over the set  $A$ .  $|\sigma| = n$  is the length of  $\sigma$ . For  $\sigma_1, \sigma_2 \in A^*$ ,  $\sigma_1 \subseteq \sigma_2$  if  $\sigma_1$  is a sub-sequence of  $\sigma_2$ . e.g.,  $\langle g, b, o \rangle \subseteq \langle z, g, b, b, g, b, o, q \rangle$ . For  $\sigma \in A^*$ ,  $\{a \in \sigma\}$  is a set of elements in  $\sigma$ . For  $\sigma \in A^*$ ,  $[a \in \sigma]$  is a multi-set of elements in  $\sigma$ , e.g.,  $[a \in \langle g, b, z, g, b \rangle] = [g^2, b^2, z]$ .

**Definition 4.1** (Event, Event Log). An event is a n-tuple  $e = (t, c, a_c, a_s)$  and  $e \in L$ .  $t \in T$  is the event timestamp,  $c \in C$  is the case id, and  $a_c \in A_c$  is the Client-Side activity.  $a_s \in A_s$  is the Server-Side activity, such that there is a set of corresponding Server-Side activities in response to a user request for each Client-Side activity.  $\xi = T \times C \times A_c \times A_s$  denotes as the universe of all events for the original event log, s.t.  $L \subseteq \xi$ . In the event log, each event can appear only once. Hence, events are uniquely identifiable by their

attributes. Also denote  $\xi_f = T_{start} \times T_{complete} \times C \times A_c$  as the universe of abstracted events. The abstracted event log is represented as  $L_f \subseteq \xi_f$ , and each event  $e_f \in L_f$  is an n-tuple  $e_f = (t_{start}, t_{complete}, c_c, a'_c)$  where  $\{t_{start}, t_{complete}\} \in T$  is the start and completion timestamp of an event respectively, representing a full software execution lifecycle,  $c_c \in C$  is the case id and  $a'_c \in A_c$  is the abstracted activity name.

**Definition 4.2** (Trace). Let  $x$  be a set of activities. Then  $x \in e$ ,  $\sigma_x = \langle e_{x_1}, e_{x_2}, \dots, e_{x_n} \rangle$  is defined as a trace in the event log, s.t., for each  $e_{x_i}, e_{x_j} \in \sigma_x$ ,  $1 \leq i < j \leq n: \pi_C(e_{x_i}) = \pi_C(e_{x_j})$  and  $\pi_T(e_{x_i}) \leq \pi_T(e_{x_j})$ . Trace  $\sigma \in L$  is a sequence of activities. Note that there are no special start and end activities.

**Definition 4.3** (Labeled Petri net). A Petri net is an n-tuple  $N = (P, T, F)$  where  $P$  is the set of places,  $T$  the set of transitions,  $P \cap T = \emptyset$ ,  $F \subseteq (P \times T) \cup (T \times P)$  represents the flow relation. A labeled Petri net  $N = (P, T, F, l, A)$  extends the Petri nets with a labeling function  $l \in T \rightarrow A$ , which maps a transition to an activity from  $A$ . For example, if  $l(t) = a$ , then  $a$  is an observed activity when transition  $t$  is executed.

### 4.1.2 Proposed Methodology

Figure 4.1 illustrates an overview of the method for abstracting event logs in Client-Server applications, along with examples of event log transformations throughout the process. It is important to note that, in the sample logs of Figure 4.14, Client activity names are denoted as “[activityName]@[webPage]”, signifying the web page on which a Client activity takes place. In the subsequent section, we explain each step involved in the recursive log abstraction process. Moreover, Algorithm 4.1 showcases a recursive log abstraction procedure utilizing the PCC [187]. The PCC allows us to determine the pairwise similarity between various Client-Side activities by evaluating the occurrence of corresponding Server-Side activities.

As suggested in the literature [58], we employ model fitness as an indicator of the appropriateness of the discovered model. The method utilizes the IM algorithm on the event log for the process model discovery task (step 1 of Figure 4.1). First, the original event log  $L$  is replicated into a temporary event log  $L'$  for further abstraction and reuse in the recursive process (Step 2 of Figure 4.1). The abstracted event log is then replayed over the discovered model derived from the original log to assess model fitness (step 3 of Figure 4.1). During this procedure, the fitness of the discovered model is computed as  $fitness = \frac{True\ Positive}{True\ Positive + False\ Negative}$ . The *True Positive* represents the number of traces that align with the discovered model, while the *False Negative* refers to the number of traces that do not align with the discovered model [188]. If the discovered model can fully replay the traces from  $L'$ , the model’s fitness is 1. Consequently, as log abstraction increases, fitness decreases.

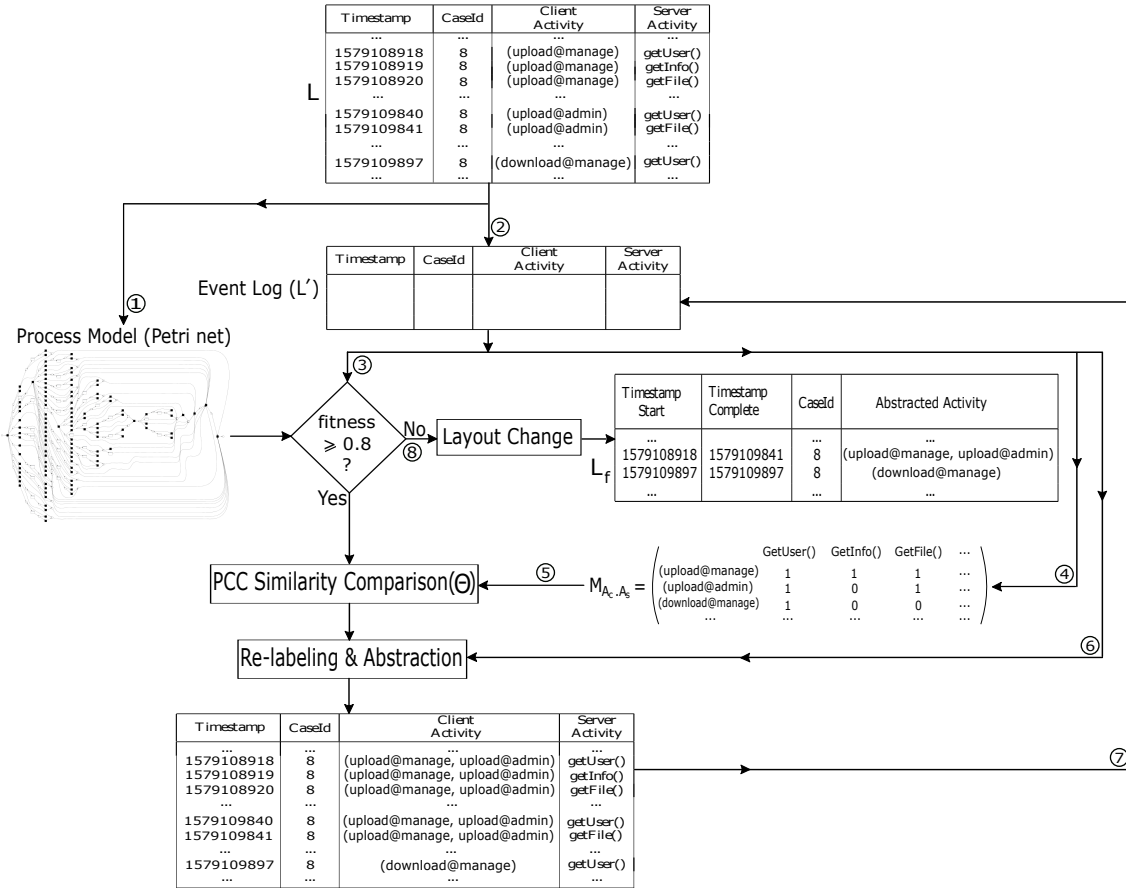


Figure 4.1: An overarching summary of the event log abstraction methodology.

**Algorithm 4.1** Recursive event log abstraction algorithm.

---

**Input:** Fine-granular event log  $L$   
**Output:** Abstracted event log  $L_f$

```

1 set PCC threshold ( $\Theta$ ) to 1
2 discover Petri net model for  $L$  using the Inductive Miner
3 clone  $L$  to  $L'$  as a temporary event log
4 while  $fitness \geq 0.8$  do
5   foreach  $L'$  do
6     convert  $L'$  to a vectorized weighted matrix  $M_{A_c \times A_s}$ 
7     foreach  $e$  do
8       calculate the vector similarity( $\Delta$ ) between every  $a_c$ 
9       if  $\Delta \geq \Theta$  then
10        relabel the activity  $a_c$  by the concatenation of the two similar activity names  $a'_c$ 
11        replace the corresponding event in  $L'$  with  $e' = (t, c, a'_c, a_s)$ 
12      end
13    end
14  end
15  Lower PCC threshold  $\Theta$  by  $\Upsilon$ 
16 end
17 foreach  $c$  in  $L'$  do
18   while  $a'_{c_i} = a'_{c_j}$  do
19     capture the first ( $t_{start}$ ) and last ( $t_{complete}$ ) occurring event for the activity  $a'_c$ 
20     create event  $e_f = (t_{start}, t_{complete}, c, a)$ , and append to  $L_f$ 
21   end
22 end
23 return  $L_f$ 

```

---

In line 6 of Algorithm 4.1, the event log  $e = (t, c, a_c, a_s) \in L$  is converted into a vectorized descriptive matrix  $M_{A_c.A_s}$ , where the first column of the matrix comprises the group of Client-Side activities ( $a_c$ ), and all other columns represent Server-Side activities ( $a_s$ ). Each row contains the number of Server-Side activity executions corresponding to a specific Client-Side activity. It is crucial to note that we increment the number of  $a_s$  occurrences for each  $a_c$  (step 4 of Figure 4.1). The weighted matrix  $M$  allows for more accurate vector similarity analyses.

In the section between lines 7 to 13 of Algorithm 4.1, we assess the similarity between every two pairs of Client-Side activities (step 5 of Figure 4.1) and relabel logs by concatenating Client-Side activity names (separated by a comma) that fall within the PCC threshold (step 6 of Figure 4.1). The PCC evaluates the strength of a linear association between two vector variables, thereby quantifying the similarity between two events. As the PCC description is beyond the scope of this report, we direct readers to [187] for the formula and a comprehensive explanation. The abstraction and relabeling results are then fed into the method (step 7 of Figure 4.1).

In the proposed approach, the abstraction process is initiated with a PCC threshold  $\Theta = 1$  and the threshold  $\Theta$  is incrementally decreased by  $\Upsilon$ . As depicted in Figure 4.1, the abstraction process continues until a fitness quality of 0.8 is achieved. A fitness of 0.8 is suggested as the stopping criterion for the iterative abstraction process in the algorithm, as a fitness value below 0.8 cannot fully replay event logs with that model [54], [58], thus losing its essential quality dimension of a process model.

The section between lines 17 to 22 alters the event log format, reducing the number of events with the same activity for each case (step 8 of Figure 4.1). At this point, every event is accompanied by the event's start and completion time, representing the time taken for the Server-Side components to process and respond to a Client-Side request. In instances of a single event appearance, the start and completion timestamps are identical, indicating an instantaneous event occurrence. The resulting abstracted event log is  $L_f \subseteq \xi_f$ , where  $e_f \in L_f$  and  $e_f = (t_{start}, t_{complete}, c, a'_c)$ .

### 4.1.3 Discussion

As the *supervised* learning technique is deemed more reliable for real-life scenarios and allows for additional domain expert interventions, the methodology also employs a *supervised* learning approach. Furthermore, the method utilizes continuous and discrete data to discover process models and measure the similarity between two activities. Due to the nature of the distributed systems in the SUS, the order of event execution is not sequential; thus, the approach must accommodate non-sequential events. The work is positioned as an  $n:m$  mapping relationship, as there are both  $1:1$  and  $n:1$  mappings between event logs at each level. To demonstrate the effectiveness of the approach, the method has been validated with real-life event logs by directly collecting and utilizing event logs from an existing software system.

In this study, event logs have been successfully abstracted reliably and descriptively for domain experts. Furthermore, the approach has effectively transformed low-level event logs into high-level representations, ensuring that the resulting abstracted event logs are comprehensible to domain experts. This achievement has facilitated a deeper understanding of user processes and operational procedures, enabling more informed decision-making and improving system efficiency and effectiveness.

Furthermore, an abstracted event log that balances granularity and the fitness of process models is successfully introduced. By identifying the optimal level of abstraction, it is ensured that the resulting process models accurately represent real-world behaviors while maintaining sufficient detail to be useful for analysis and interpretation. This balance has proven crucial in providing meaningful insights into the underlying processes and identifying areas for improvement, ultimately contributing to the overall optimization of the system.

## 4.2 Modeling Process-Aware Activities

Online services like Celonis<sup>1</sup> and Disco<sup>2</sup> can be deployed to cloud servers. However, they are not at a point where they have a sufficiently developed API or import and export functionality to allow integration into an existing ecosystem [6]. For example, while Celonis provides a flexible way of customizing the usage of their application by giving the user a way to upload Python scripts which in turn can utilize the Celonis API to fit the process discovery into a custom workflow, this cannot still have the process discovery itself ready as a service and to access process models on-demand [189].

Different approaches to data preprocessing can be categorized into specialized tools, comprehensive tools, and low-level language features. Specialized tools are good at executing specific data cleaning and preprocessing tasks, but combining them with other specialized tools can be difficult due to proprietary languages, APIs, and restrictive licenses. On the other hand, comprehensive tools like Googles OpenRefine<sup>3</sup> provide users with a graphical user interface, allowing them to manipulate their data sets efficiently, but the reusability of the generated sets of operations is limited [190].

Researchers have acknowledged the need for a tool to analyze research data management processes in recent years. Local applications like ProM [54], Disco, and RapidMiner<sup>4</sup> provide users with a graphical user interface that guides them through the entire process from importing data to generating a process model. However, these applications have certain limitations when integrating them into a larger organizational ecosystem with distributed services [6]. For example, they require that the input comes in a single file with a specific format, with no option to access a database directly. Moreover, they do not offer a suitable API to enable communication with running services, limiting their potential benefits of developing a sound process model that could be used in real-time detection for deviations from the model [189]. The following section discusses a web-based tool for analyzing research data management processes.

The Data Analysis for Research Data Management (DA4RDM) framework provides a web-based solution for data analysis in research data management. It consists of four modules: data source configuration, data cleansing, data transformation, and presentation of findings. The implementation of the DA4RDM web application is based on the Flask web framework and is publicly available on GitLab<sup>5</sup>. The four modules of DA4RDM allow for a streamlined data analysis process for research

---

<sup>1</sup><https://celonis.com/>

<sup>2</sup><https://fluxicon.com/disco/>

<sup>3</sup><https://openrefine.org>

<sup>4</sup><https://rapidminer.com/>

<sup>5</sup><https://git.rwth-aachen.de/AminYazdi/da4rdm>

projects. The data source configuration module enables users to connect to various data sources, while the data cleansing module provides tools for cleaning and filtering data. Furthermore, the data transformation module allows for data transformation to fit the scope of specific research projects. Finally, the presentation of findings module provides customizable post-processing and data modeling tools to present the analysis results. Overall, DA4RDM is a valuable addition to the existing tools for data analysis in research data management. Its modular design and customizable tools provide users with a streamlined and comprehensive solution for analyzing their data. In addition, the availability of the implementation on GitLab makes it easily accessible for researchers and data scientists to use and modify to fit their specific needs.

It is worth noting that the description of DA4RDM has been previously presented in my supervised technical reports and publications in [167], [191]–[193] and has undergone rigorous reviews. The framework has also been discussed and scrutinized by fellow scholars, leading to valuable feedback.

### 4.2.1 Characteristics

DA4RDM is designed with the following characteristics:

**Scalability:** DA4RDM offers web-based solutions for linking data repositories, tailoring and implementing project-oriented pre-processing workflows, and supports integrating Python libraries for process and data analysis initiatives. The present version of DA4RDM has incorporated the Process Mining for Python (PM4PY) Python library [Berti2019Process] within the current service framework for process discovery purposes.

**Extendability:** As depicted in Figure 4.2, DA4RDM’s modular design enables the incorporation or expansion of external Python libraries to meet the requirements of a data analysis investigation. For example, diverse data modeling endeavors can be established and executed by enhancing suitable modules for data simulation, anomaly detection, or data equalization and standardization.

**Accessibility:** The accessibility of DA4RDM is divided into two aspects. First, a data scientist must set up the pipeline, which involves connecting to a data source, determining an appropriate data query, and arranging the pre-processing pipeline for a project. After defining a pre-processing pipeline, its steps are saved for future utilization as pre-established projects. Second, non-technical users can repurpose a preconfigured project and apply extra web-based filters, attributes, and algorithms to the data, facilitating data model post-processing without coding expertise. Consequently, the DA4RDM client application empowers principal investigators to examine the RDM system without requiring technical knowledge.

Following this, DA4RDM enables using a generated dataset by an RDM system through the user interface depicted in Figure 4.4. This interface permits the selection of a pre-processing pipeline to assess and convert data samples into a new format appropriate for data modeling algorithms. As a result, all JSON objects are gathered and stored in a relational database, primed for import into DA4RDM.

### 4.2.2 Functionalities

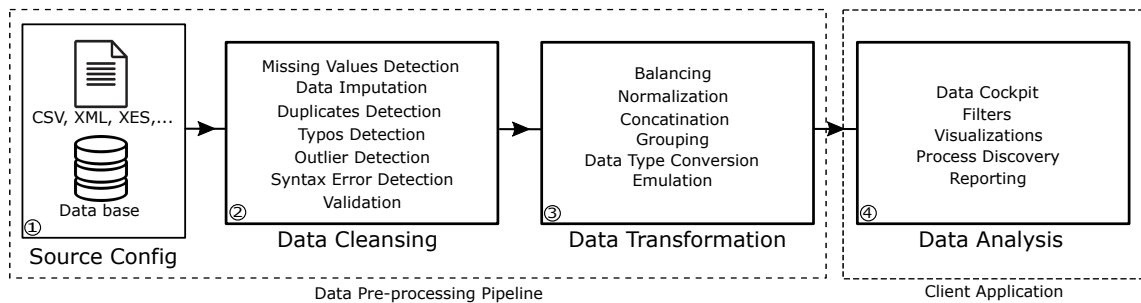


Figure 4.2: The comprehensive web-based framework of DA4RDM and its modules, adapted from [167].

In the subsequent section, the features of the DA4RDM web application are discussed. The functionality of DA4RDM has been categorized into three primary sections: data source configuration, data pre-processing (cleansing and transformation), and data analysis.

**Data Source Configuration:** The data source module enables the provision of raw data to DA4RDM. As illustrated in step 1 of Figure 4.2, the data input module supports uploading local files in Comma-Separated Values (CSV) and eXtensible Event Stream (XES) formats or directly connecting to a relational database table. In the data source user interface (refer to Figure 4.3), users can input additional parameters for each data source further to define a file configuration or a database query command. Although local data storage is available, each new execution of the pre-processing pipeline retrieves the most recent version of a specified data source. Additionally, this module is in charge of serializing the data structure into a Panda Dataframe, facilitating data alterations necessary for later stages of the process. It is important to note that XES file formats are standardized XML data types utilized by Process Mining algorithms; therefore, DA4RDM manages to convert input data into XES format and prepare for process mining algorithms.

**Data Pre-Processing:** As displayed in Figure 4.4, the data pre-processing interface handles data cleaning and transformation. The user interface enables the selection



Figure 4.3: A screenshot of the input data configuration page of DA4RDM.

of a pre-defined data source and a pre-processing pipeline to initiate a data-driven study. While the pre-processing pipeline largely depends on the input data properties and the goals of a data analysis project, this tool allows for reusing a pipeline with operational data. As a result, a data scientist specifies a data source and then develops or adjusts a pre-processing pipeline based on their needs, using the methods outlined in steps 2 and 3 of Figure 4.2 to conform to a given data structure or use case. Furthermore, DA4RDM generates a user session that preserves the chosen data source and pre-processing pipeline results, ensuring a seamless workflow throughout the client application.

**Data Analysis:** The data analysis module lets users post-process the data for further examinations and studies. This stage contains features such as process discovery, conformance checking, and RDM lifecycle visualization.

The process analysis, as displayed in Figure 4.5, offers principal investigators the necessary toolset and user interface, allowing them to further examine data without requiring technical expertise. In this phase, users can employ several process mining algorithms provided by the PM4PY library to uncover process models and investigate specific user journey scenarios based on frequency or performance analysis. DA4RDM users can easily define the primary event attributes (Timestamp, Case Id, and Activity) required for process discovery based on existing data features.

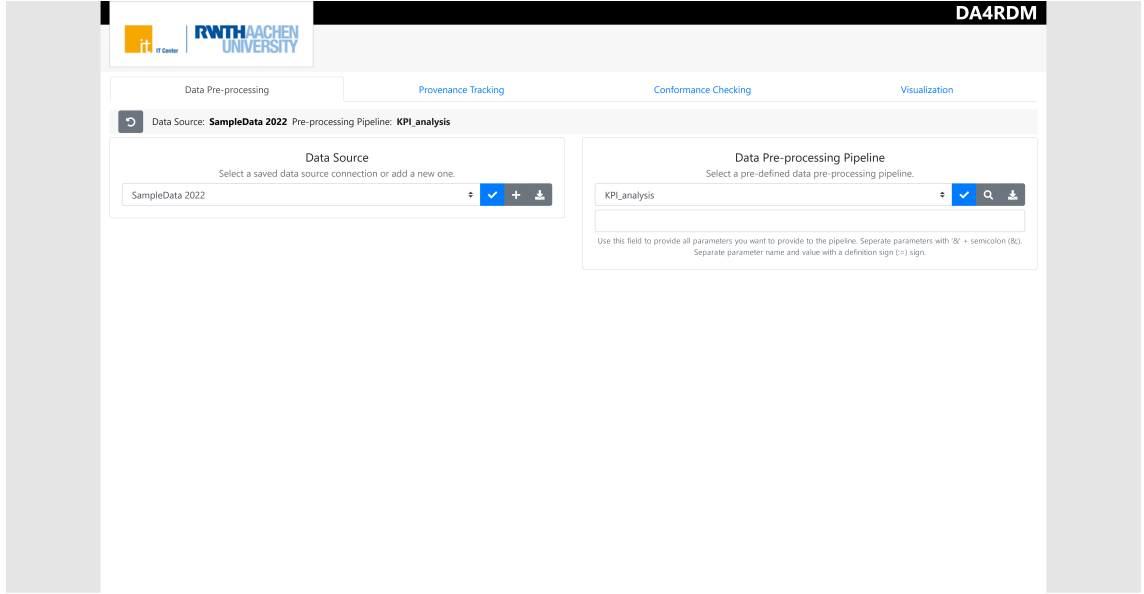


Figure 4.4: A screenshot of the data pre-processing page of DA4RDM.

Furthermore, the filter module enables more focused analysis on a specified dataset. In addition, the information module delivers vital statistics of the chosen dataset for specified criteria and filters. The results section presents a visualization of the discovered model according to selected algorithms such as Alpha Miner, Heuristic Miner, IM, or Directly Follow Graph (DFG) [54].

For conformance checking shown in Figure 4.6, we examine traces of user interactions against the defined variance. One can define a sequence of actions and performance criteria to be checked against a descriptive dataset. The following is a mathematical foundation for running conformance checking on DA4RDM.

Let  $\mathcal{E}$  be an event log represented as an n-tuple  $(u, p, a, t, r, x, f, s)$  where  $u \in U$ ,  $p \in P$ ,  $a \in A$ ,  $t \in T$ ,  $r \in R$ ,  $x \in X \cup \perp$ ,  $f \in F \cup \perp$ ,  $s \in S$  denotes user Id, project Id, action, timestamp, role Id, resource Id, file Id, session Id respectively,  $\perp$  represents a null value. The  $s \in S$  session id helps to identify the start and end of a sequence of actions per user within a project and are used as identifiers for cases. As illustrated in Figure 4.6, the users of DA4RDM can define two sequences of actions, denoted as  $\alpha_1 = \langle a_1^n, a_2^n, \dots, a_m^n \rangle$  and  $\alpha_2 = \langle a_{m+1}^n, a_{m+2}^n, \dots, a_k^n \rangle$ , where  $a_i \in A$ ,  $1 \leq m < k$  and  $a_i^n$  represents action  $a_i$  reoccurrences. Within every case trace, the second sequence of actions  $\alpha_2$  should be followed by  $\alpha_1$ . If an action other than those defined in  $\alpha_2$  occurs within a trace, it is recorded as a non-conforming case. Users can enable the

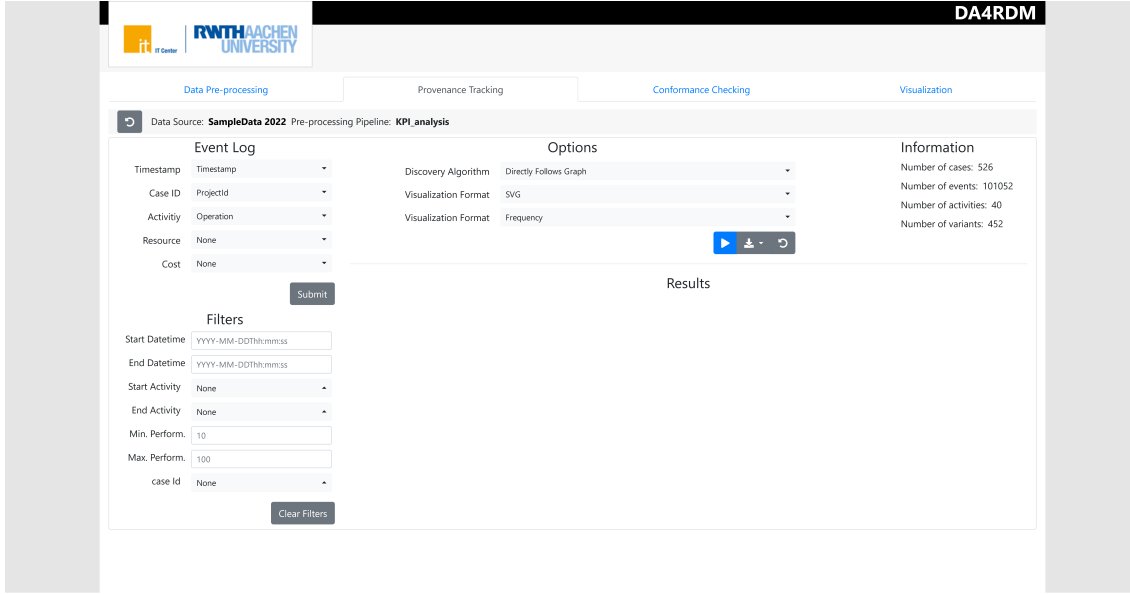


Figure 4.5: A screenshot of the process discovery page of DA4RDM.

*Eventually Followed By* checkbox so that if other activities are observed between the defined sequences, we can still ignore the actions in between and consider them a set of conforming cases. Additionally, we can evaluate a trace with respect to performance. The performance is the time span between every two defined sequences of actions, denoted as  $\Delta$ . If two defined sequences of actions occur but still fall short of the defined performance, it is also counted as a non-conforming case. The output is a set of non-conforming cases  $\mathcal{N} = \{e_i \in \mathcal{E}' \mid \alpha_1 \rightarrow \alpha_2 > \Delta\}$ . Every event in the non-conforming cases contains the 8-tuple  $e_i = (u, p, a, t, r, x, f)$ .

The detailed explanation of the RDM data visualization, depicted in Figure 4.7, is provided in Section 4.3.

### 4.2.3 Software Design and Architecture

To better describe DA4RDM system design and architecture, it is splitted into three sub-sections. Namely, Client-Side, Server-Side and API Architecture.

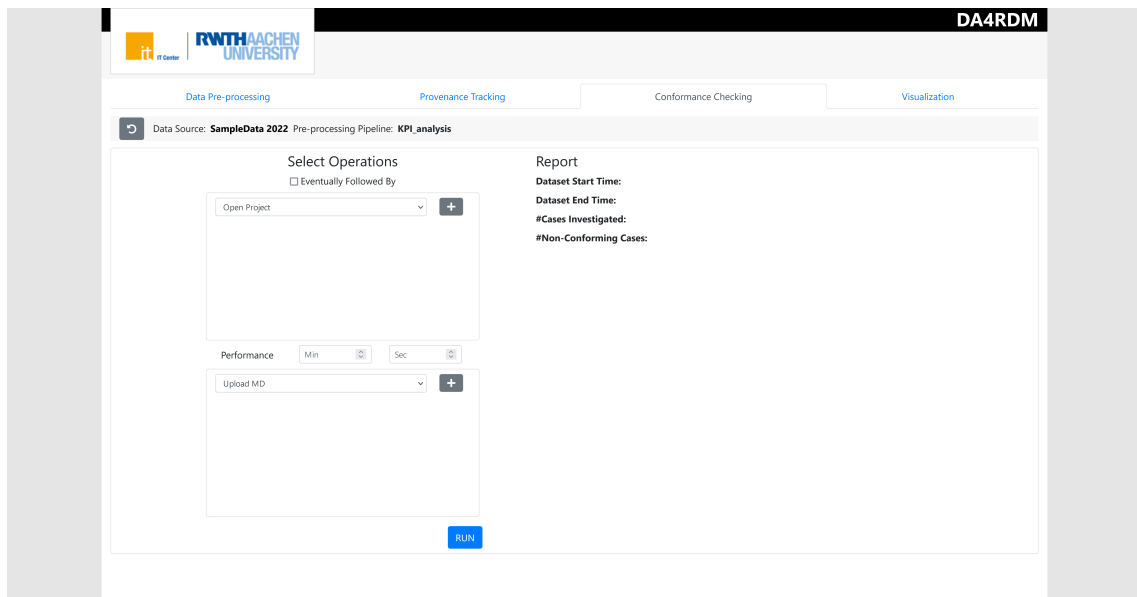


Figure 4.6: A screenshot of the conformance checking page of DA4RDM.

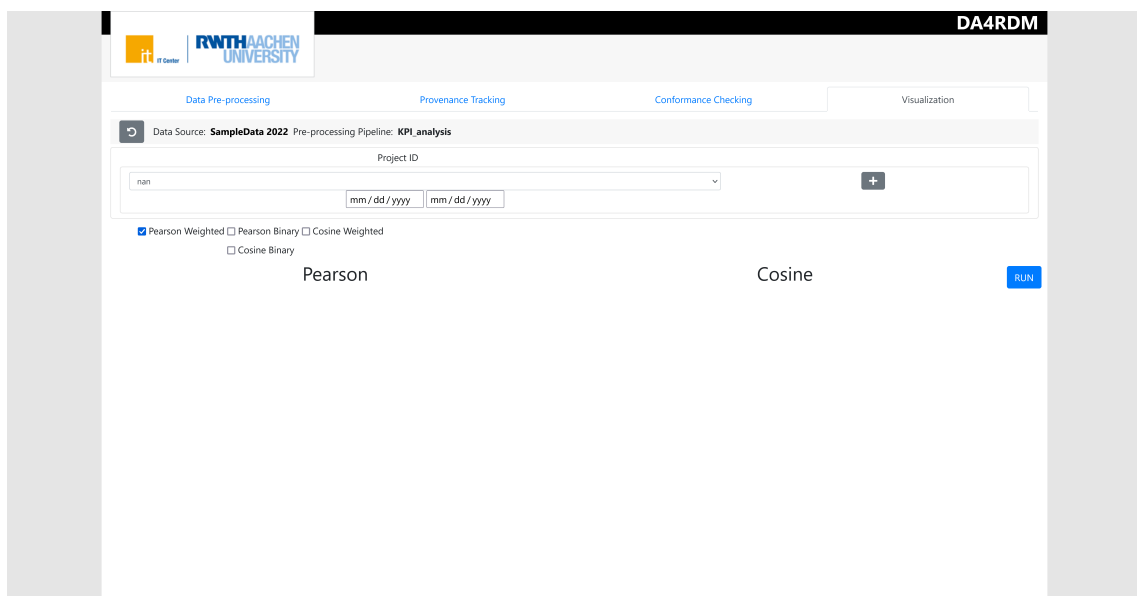


Figure 4.7: A screenshot of the data visualization page of DA4RDM.

## DA4RDM Client-Side

**Architecture:** The Client-Side architecture of the web-based tool, DA4RDM, is structured into a frame and content pages. The frame comprises a shared header, user-session information, and navigation elements for all content pages. Extracting these shared components into a template is sensible since they remain unchanged. Dynamic parts of the dataframe, such as the navigation indicator, session information, and page title, can be partially adapted by the Server before sending the pages to the Client and updated by a global script. The template for the frame also includes global scripts and style sheets relevant to each content page. This incorporates external libraries, a custom style sheet, a user session handler, and a file request handler. The user session handler requests a new session for interacting with DA4RDM, and if no user session is detected upon entering the site. It also appends the session id to POST requests of forms for transmitting data that informs the Server-Side session state.

The file request handler appends request URLs for file downloads with an arbitrary query parameter. This is necessary due to file association with user sessions. Without the query parameter, the browser uses cached files for the requested data if it recently accessed a resource URL path, saving internet traffic. However, this undesired side effect prevents users from accessing newly generated files on the Server after downloading files within a session and changing filters, event log attributes, or file formats.

The content pages include a pre-processing data page and a post-processing pages. The pre-processing page is a static web page displaying all available data sources and user projects accessible from the current user session. It has two sub-pages for adding a new data source and inspecting pre-processing pipeline evaluation details. The process discovery page is more dynamic, frequently updating its contents based on user interactions, such as event log attributes, filters, dataset information, and downloadable resources. It also reserves space for the graphical representation of the process model.

**Technologies:** Technologies used in the DA4RDM (Client-Side) web application include:

- **HTML and CSS:** These two frameworks are the standard for implementing and styling web forms.
- **JavaScript:** A programming language used for the Client-Side logic of the application.
- **Bootstrap:** A web framework utilizing HTML, CSS, and JavaScript for dynamic styling of the web Client application.
- **Axios:** A lightweight JavaScript library for sending and handling HTTP requests, offering a comfortable and efficient alternative to jQuery.

- **FontAwesome:** A collection of icons used to improve usability and reduce usage barriers.
- **Socket.IO:** A framework available in different programming languages for establishing and maintaining enduring network connections, emitting and listening to events over the connection. This is useful for exchanging messages between Server and Client, even when the Server is busy with tasks like running a pre-processing data pipeline or generating a process model.

### DA4RDM Server-Side

**Architecture:** The high-level Server component structure of DA4RDM is illustrated in Figure 4.8. At the heart of the Server lies the main Flask application, which manages all incoming and outgoing requests, maintains Client connections as needed, and delegates tasks to the responsible modules for processing requests. The main configuration is utilized across the application for centrally managed options, such as Server connection timeout settings and file paths for local data storage. The Flask application also establishes a connection to the local database and offers access through the data model and an object-relational map via Flask SQLAlchemy.

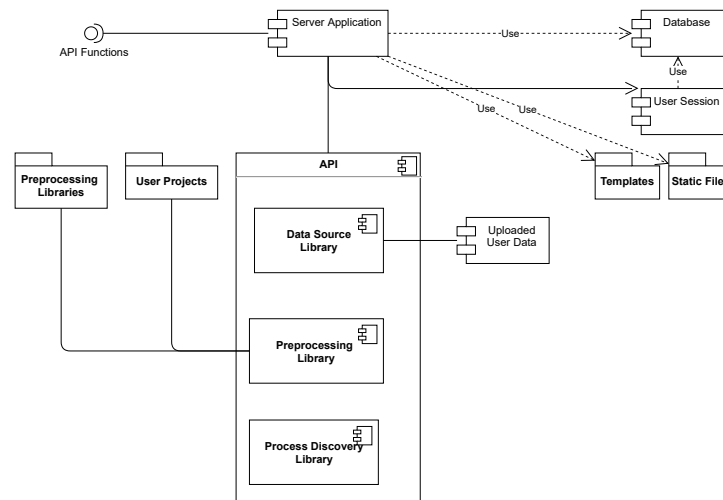


Figure 4.8: Schematic representation of the DA4RDM components.

The database is employed to keep track of user sessions and data sources. Although the data processed by the Server should adhere to a tabular format, the data itself is not stored in the database. This is due to the Server's use of a relational database and data format. In addition, the content can vary significantly across users or use cases, making providing a one-size-fits-all database schema challenging. Therefore, utilizing a basic schema solely for dumping all user data into a single column is not

a practical approach, as it undermines the advantages of storing data in a database, such as memory and query efficiency.

The inclusion of the modules for uploaded user data and temporary work results is justified on demand to store user data after the distinct steps of preparing it for process discovery. Both modules designate storage space for the data on persistent memory, and they are kept separate due to differences in purpose and form.

The module for uploaded data stores the data exactly as the user uploaded it, maintaining the original file format and making no changes to the content. This data is not associated with any specific user session but with the specified data source reference created when the file was uploaded. Consequently, the same uploaded data can be used across multiple projects without altering the data itself. This also allows for user session-related data to be cleaned up after the session is over without affecting the functionality of any data source.

To reduce the dependency between different project stages, the results of reading input data from the data source are stored separately from the results of applying a pre-processing pipeline. Additionally, CSV and XES files must be stored during process discovery. These files are all locally kept in a folder associated with the user session during which they were constructed. As shown in Figure 4.9, selecting a data source via the application always creates a file for use in the pipeline and a file for post-processing analysis.

The pipeline intends to use the source to fetch the sample dataset, store it in the specified file, and overwrites it for discovery only if it runs successfully. This approach makes running a pipeline optional in terms of the application structure. The database stores the associations between the raw data and the available data source and between the temporary results and the user session.

The templates module contains HTML pages, while the static directory accommodates non-HTML files accessible to the frontend application. In addition, the static directory includes style sheets, script files, and images (e.g., the graphical representation of process models).

**Client and Server Communication:** The communication between Client and Server, except for basic page requests and file up- and downloads, is socket-based. When a page is called, the connection between the Client and the Server in the API namespace relevant to the current flask project is established and maintained by regularly sending keep-alive requests and responses between them. The specifics regarding the timeout timer and frequency of these messages can be configured in the global Server configuration.

Well-formed requests that the Server can process follow this format: Besides being a well-defined socket-based HTTP request (handled by the `socket.IO` library on both

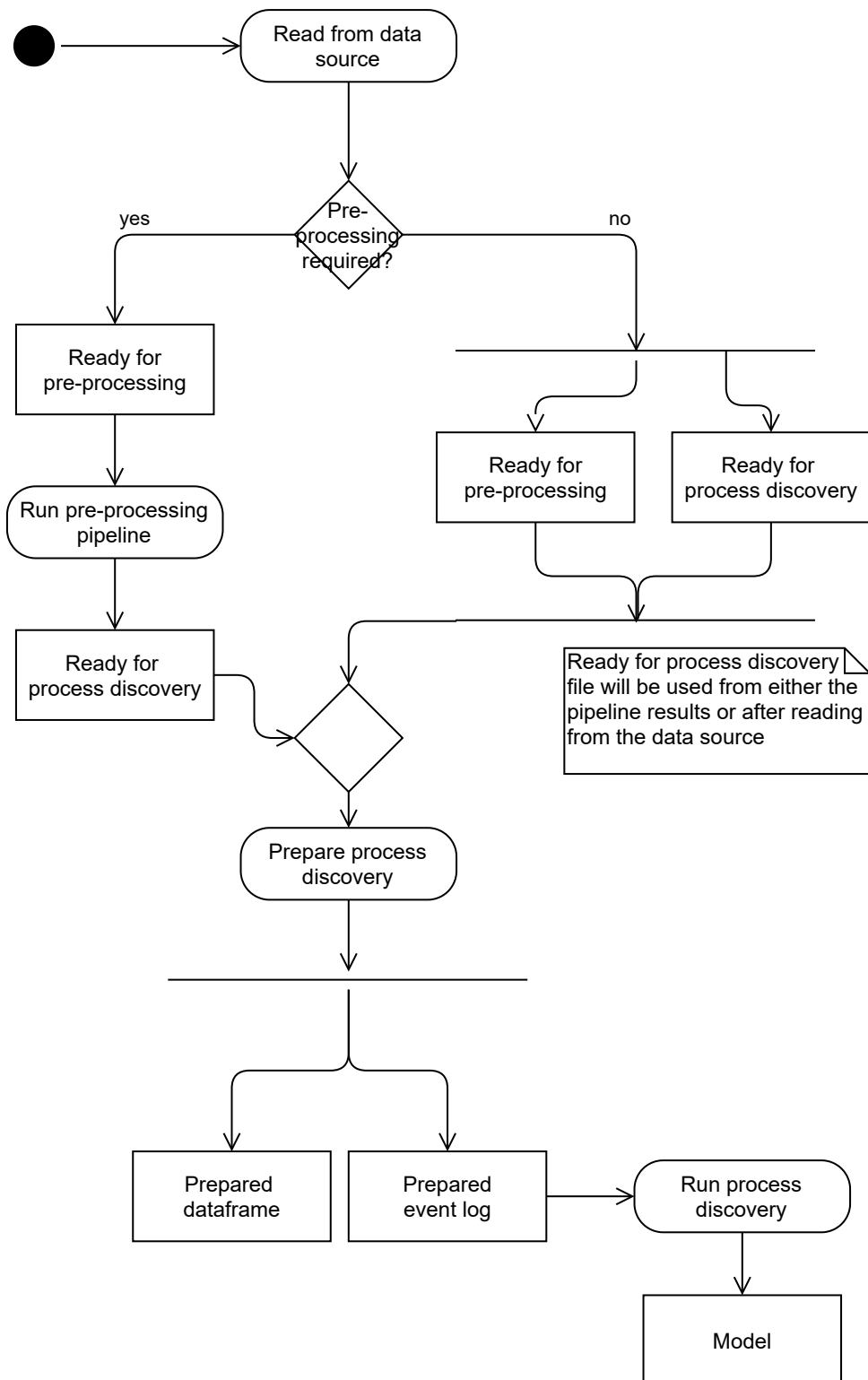


Figure 4.9: Files Created on uploading data and for handling user sessions.



Client and Server), they start with an event name, usually followed by the session Id and then the rest of the data. An example of the Client request is listed below:

```
let socket = io.connect('http://' + document.domain + ':' +
+ location.port + '/api/run_process_discovery');
socket.emit("requestDiscoveryPreparation",
session_id, selectedXesAttributeColumns);
```

The first instruction establishes the connection. Evidently, in this case, the connection is directly established in the namespace,

`/api/run_process_discovery`

which contains all the event listeners listening to incoming events corresponding to requests that can be made on the process discovery page. As mentioned, this will have already happened upon entering the page. The second line is an example of a specific request. While the socket connection is open, both ends listen to incoming requests and use the event name to identify the kind of event and trigger the corresponding functionality. The rest of the data is gathered in a serialized collection and sent as a separate parameter. `socket.IO` also manages the serialization for many of JavaScript's and Python's default data structures, although more complex objects sometimes require additional preparatory work. One case where this happens is when the process discovery page serializes the data containing `DateTime` objects, which have no default serialization that `socket.IO` can utilize.

**Database Model:** The application utilizes a relational database, specifically an *SQLite* database, due to its ease of setup, maintenance, and adequate performance for a small number of concurrent users. If scalability becomes necessary in the future, such as accommodating more concurrent users, the database can be replaced easily, thanks to the abstraction provided by SQLAlchemy for the data model, connection, and migrations within the application. As it stands, read and write operations are efficient and infrequent, so scaling does not pose an issue until a significantly higher number of concurrent users is reached. This observation is further supported when examining the database schema.

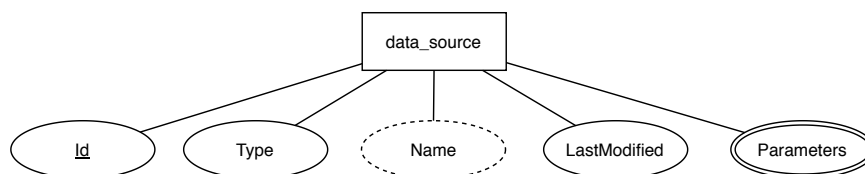


Figure 4.10: Relational model of data sources in DA4RDM.

The database consists of only two separate tables: one for data source information and the other for user-session information. The data sources table includes attributes for the data source connection name, the data source type, and a multi-value attribute for parameters (refer to figure Figure 4.10). The name serves to help users identify the data source when selecting it after saving. The type is employed internally to guide the Server's control flow for various data connection types, which include `csv`, `xes`, and `database`. The parameters attribute contains user-provided information for specifying the connection. For example, a CSV file may require a separator to denote the characters used for column delimitation, while a database connection might need a connection string and an initial query. The latter should adhere to the format used by SQLAlchemy for establishing a database connection engine.

Supported parameters like database connection strings or queries can freely use white spaces, equal signs, and other single-character symbols. Similarly, common column separators for CSV-like files often use characters like semicolons, commas, or tabs, which could also be employed to separate different parameters or their keys from their values. Consequently, the application uses `&` as a separator for different parameters and `:=` to separate parameter keys from their values. While these unconventional signs might make the input less intuitive, their abnormality also reduces the likelihood of errors related to parsing verbatim input for the parameters.

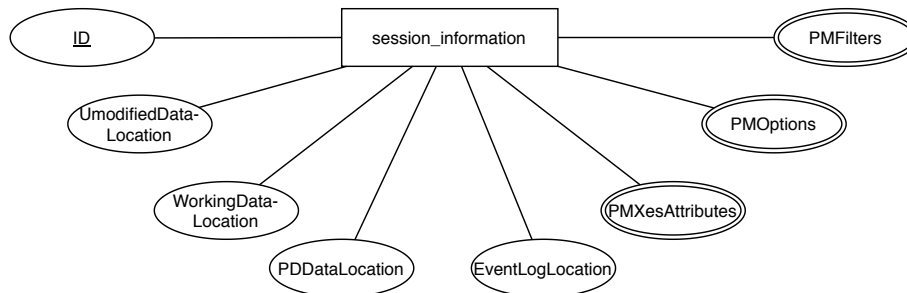


Figure 4.11: Relational model of a user session in DA4RDM.

The session information includes fields for further data analytics and process discovery filters, event log attributes, and paths to various data locations, which indicate where the files are located on the Server shown in Figure 4.11. The filter option and event log attributes are multi-value attributes that store sequences of key-value pairs for the related selections. In terms of functionality, how filters and options are implemented in PM4PY implies that any new parameter will require special API handling. This should occur in the `apply_all_filters` function of the process discovery module.

The path attributes store single-value string attributes generated according to a predefined pattern when a new user session object is created. For example, the temporary results and static files modules allocate designated spaces for the created

files on the Server. In the current implementation, the path attributes obtain values that consist of the path to these modules and the session Ids, used as either directory or file names to ensure organization and individual identification.

### **Technologies:**

- **Flask:** Flask is a lightweight, flexible Python web framework for building web applications, offering a modular approach to adding necessary components. It simplifies tasks such as routing, handling web requests, and rendering templates while providing extensibility for custom features.
- **PM4PY:** It is a Python process mining package focused on analyzing and optimizing business processes using event log data. PM4PY[194] provides a comprehensive set of algorithms for discovering, analyzing, and visualizing process models from event logs. PM4PY is built on top of popular Python libraries like Pandas, NumPy, and Graphviz, making integrating with other data analysis and visualization tools in the Python ecosystem easy.
- **sqlalchemy and Alembic:** It is a versatile Python library that facilitates connecting to various popular database engines and offers object-relational mapping, simplifying data manipulation through data models. Alembic, working in tandem with sqlalchemy, manages database migrations by creating and executing migration scripts based on data models, ensuring reliable database schema updates.
- **Socket.IO:** It is a JavaScript library that enables real-time, bidirectional, and event-based communication between Clients and Servers. It abstracts the complexities of real-time communication protocols, such as WebSockets and long polling, allowing developers to quickly build real-time applications like chat, notifications, or live updates without worrying about the underlying technical details.

### **DA4RDM API Architecture**

The API comprises several sub-modules: data source, pre-processing pipeline, and process analysis for the core functionality, along with session and input parser modules for additional logic.

The API has a layered structure, where the top level handles incoming requests, input arguments, and session information. It passes relevant information to more specialized layers and manages the results concerning user sessions, which dictate the Server's overall state. The user session and input parser modules operate simultaneously, supporting parsing various input types and managing user session database objects.

The next level consists of the core modules for handling data sources, pre-processing pipelines, and process discovery. Finally, a handler sub-module, called by the API, delegates specific tasks to specialized functions as needed.

The lowest level includes specialized functions and external libraries or their wrappers, responsible for specific computations such as reading from a specific data source like a CSV file, renaming a data frame column, or executing a particular process discovery algorithm.

**Data Source:** The data source module uses a strategy pattern to provide a simple interface that makes it easy for the API to read from different kinds of data sources. Depending on which data source type is specified in the data source object in the database, the corresponding reader gets loaded, and its `read` function is applied. Additionally, depending on its kind, the module processes parameters that may be given to the reader. This includes the delimiters for CSV-like files or the connection and query strings for database connections. To that end, as well as for registering new data sources, this module extends the object-relational map of the data model by wrapping the communication with the database table via `SqlAlchemy` with some easy-to-use functions.

**Pre-Processing:** The pre-processing module dynamically reads and presents all custom pre-processing pipelines in the configured directory to the Client, eventually loading the selected one via runtime import. The pipelines are saved as individual Python modules in a designated directory (modifiable in the global application configuration). These modules must follow an interface that mandates the implementation of at least an `init` and a `run` function.

`init(data_source, pipeline_parameters)` is essential for setting up the pipeline. Data source refers to the unaltered CSV-serialized data frame saved after being read from a data source. Pipeline parameters can be any arbitrary parameters the pipeline developer wishes to utilize. For example, these may include alternative data sources or values that should be applied in every pipeline execution but may vary based on the data source. These parameters should be combined into a configuration item, which can be further extended by any configuration choices the pipeline developer desires. This configuration should then be returned, as it will be employed to run the pipeline.

`run(project_config)` executes the pipeline, optionally utilizing any items set in the configuration. The API and the pre-processing handler will use the returned dataframe to store the results for the post-processing tasks. In addition, a library of wrapper functions for commonly used pre-processing operations is also provided to facilitate the development of custom pre-processing pipelines. Currently, this primarily simplifies the use of some more complex operations, but this approach offers two additional potential benefits:

1. Although currently only pandas dataframes are supported, these wrapper functions make it easier to replace the pandas library with another one (such as Dask, which wraps pandas to improve performance with larger data sources). Furthermore, if the wrapper functions are used, they only need to be adapted to the other library, eliminating the need to change the pipelines.
2. Creating these pipelines is a complex task and requires a developer to understand how this application and pandas data frames work. While manually crafting the pipelines offers flexibility, further improvements in the application's usability can be achieved by providing options for adding data-cleaning operations to the Client.

**Process Analysis:** This module incorporates the PM4PY library and offers wrapper functions for the utilized process discovery and event log filter algorithms. Similar to the various file readers in the data source module, different process discovery algorithms and visualizers can be selected, imported, and employed at runtime using a strategy pattern. Additionally, there is a helper module for formatting a data frame into a valid event log format based on PM4PY's minimal requirements. The process discovery module is currently the only one that stores its results in the output directory (within the static module), allowing the Client to access the results and display them directly. Available files include the filtered data frame, event log, and the graphical representation once created.

**Server States and User Sessions:** To maintain data persistence between different requests, a user session system is implemented alongside temporary local file storage for the results of various steps. When a user first accesses the Client application or refreshes their session, a session is created and stored on the backend, and the session Id is provided to the Client, which saves it in local storage. Most subsequent requests from the Client require this session Id to identify the corresponding data on the Server. The Server utilizes the session to store data locations saved during each application step and additional information like pipeline columns and filters selected during the discovery process.

#### 4.2.4 Discussion

This Section covered various aspects of the data-driven application for modeling process-aware activities, including its design, implementation, and the technologies used.

The application employs a modular and layered architecture, with API modules for data source handling, pre-processing, process discovery, and user session management. This design choice allows for easy extensibility and maintainability. Additionally,

using strategy patterns for data sources and process discovery algorithms provides flexibility and adaptability to future requirements.

The DA4RDM application relies on well-established technologies and libraries like Flask, PM4PY, SQLAlchemy, and SocketIO. Python is used as the primary programming language to ensure access to a rich ecosystem of libraries and frameworks, which facilitates the development of process mining and data science applications. Flask, as a lightweight and extensible web framework, enables the rapid extension of my tool while providing the necessary tools for various tasks, such as database connection and handling web requests. Finally, PM4PY serves as the core process mining library, offering various algorithms and functionalities.

The application utilizes a user session system to manage data persistence between requests. This approach enables the server to store the locations of data saved during each step of the application and also allows for tracking additional information, such as pipeline modules and filter selection, during the discovery process.

The chosen design and technologies provide several benefits, including flexibility, extensibility, and a rich ecosystem of libraries. However, there are also limitations and areas for improvement. For instance, the current implementation of pre-processing pipelines may require a developer with a good understanding of pandas' dataframes and the application itself. Future enhancements could focus on providing more user-friendly interfaces for creating custom pre-processing pipelines or incorporating additional data-cleaning operations. Using wrapper functions could facilitate the adoption of alternative libraries, such as Dask, for improved performance with larger datasets.

Beyond these technical aspects, the application offers valuable insights for RDM, such as:

- The application can be used to check if research projects comply with the RDM process models and identify areas where deviations frequently occur. This helps ensure that projects follow the desired path and adhere to the guidelines set by the RDM platform.
- Analyzing user behavior patterns within the RDM platform can determine if there are certain activities or stages where users tend to struggle or need more assistance. This information can be used to improve the user experience and provide targeted support.

In conclusion, the data-driven application for process mining exhibits a robust and flexible design, employing well-established technologies and libraries to address various data science and process mining requirements. Despite potential constraints and areas requiring augmentation, the app's modular structure, and expandable framework establish a solid foundation for future upgrades, modifications, and pragmatic applications. This includes the evaluation of user interactions within OSPs during the execution of Research Data Management (RDM) practices.

## 4.3 Discovering RDM Lifecycle

The following Section presents the proposed methodology for discovering and outlining the various RDM phases of research projects. This novel approach aims to provide a comprehensive understanding of the RDM lifecycle and facilitate effective management of research projects regarding data handling and organization. The methodology is designed to uncover insights and patterns in the RDM process by leveraging advanced process mining techniques and conformance checking, ultimately guiding researchers in their RDM efforts. Furthermore, the proposed methodology allows stakeholders to monitor the progress of their research project and ensure alignment with RDM best practices.

The study and findings discussed in this Section have been previously partly presented in [191], [195], [196].

### 4.3.1 Conceptual Foundations

For the task of discovering the RDM Lifecycle, the conceptual foundations of *set*, *Event Log*, *Trace*, and *Labeled Petri net* are described in this Section.

Table 4.2: An example of event log collected by Server-Side logger of the Coscine platform.

Timestamp ( $t$ )	Activity ( $a$ )	CaseId ( $c$ )	ProjectId ( $p$ )
...	...	...	...
1579108918	Add Member	29613-d8...	4b15f...
1579109840	Add Resource	29613-d8...	4b15f...
1579129397	Upload File	29613-d8...	4b15f...
1579835452	Archive Resource	8G7s6-c2...	4b15f...
...	...	...	...

**Definition 4.4** (Set). Given a set  $A$ ,  $\sigma = \langle a_1, a_2, \dots, a_n \rangle \in A^*$  is the set of all finite sequence over the  $A$ .  $\sigma(i) = a_i$  denotes the  $i^{th}$  element of the sequence.  $\beta(A)$  is the set of all multi-sets over set  $A$ .  $P_{NE}(A)$  is the set of all non-empty subsets over the set  $A$ .  $|\sigma| = n$  is the length of  $\sigma$ . For  $\sigma_1, \sigma_2 \in A^*$ ,  $\sigma_1 \subseteq \sigma_2$  if  $\sigma_1$  is a sub-sequence of  $\sigma_2$ . e.g.,  $\langle g, b, o \rangle \subseteq \langle z, g, b, b, g, b, o, q \rangle$ . For  $\sigma \in A^*$ ,  $\{a \in \sigma\}$  is a set of elements in  $\sigma$ . For  $\sigma \in A^*$ ,  $[a \in \sigma]$  is a multi-set of elements in  $\sigma$ , e.g.,  $[a \in \langle g, b, z, g, b \rangle] = [g^2, b^2, z]$ .

**Definition 4.5** (Event & Event Log). An event, represented as  $e = (t, a, c, p)$ , belongs to an event log  $L$ . Table Table 4.2 presents an example event log. Here,  $t$  is the event timestamp,  $a$  is the user activity intercepted by an API endpoint, and  $c$  is the

user session determining each case. The sessions are generated upon login and expire after an hour of user inactivity or a logout attempt. Furthermore,  $p$  refers to Coscine research project identifier. We denote  $\xi = T \times A \times C \times P$  as the universe of all events for the original event log, s.t.  $L \subseteq \xi$ .

**Definition 4.6** (Trace). A Trace  $x$  is a set of activities where  $x \in e$ , and  $\sigma_x = \langle e_{x_1}, e_{x_2}, \dots, e_{x_n} \rangle$  is defined as a trace in the event log. For every  $e_{x_i}, e_{x_j} \in \sigma_x$ , if  $1 \leq i < j \leq n$ , then  $\pi_C(e_{x_i}) = \pi_C(e_{x_j})$  and  $\pi_T(e_{x_i}) \leq \pi_T(e_{x_j})$ . A trace  $\sigma \in L$  is a sequence of activities, with login and logout or session timeout as start and end activities respectively.

**Definition 4.7** (Labeled Petri net). A Petri net is denoted as  $N = (D, S, F)$  where  $D$  is the set of places,  $S$  is the set of transitions,  $D \cap S = \emptyset$ , and  $F$  belongs to  $(D \times S) \cup (S \times D)$ , representing the flow relation. Extending from Petri nets, a labeled Petri net is denoted as  $N = (D, S, F, l, A)$ , encompassing a labeling function  $l \in S \rightarrow A$ . This function maps a transition to an activity from  $A$ . For instance, if  $l(s) = a$ , then  $a$  becomes the observed activity when the transition  $s$  is executed.

### 4.3.2 Proposed Methodology

The methodology evaluates the conformance of research projects to the prescribed RDM phases. The prescriptive models for each RDM phase are based on the definition of each RDM phase discussed earlier in Section 1.2.2 and are derived from a series of user interaction scenarios within Coscine in a controlled environment. The first step is to acquire event logs corresponding to each RDM phase by considering user interaction scenarios specific to each phase. Using the IM algorithm, these event logs are then used to discover process models per RDM phase. The discovered process models using IM and Petrinet representation corresponding to each RDM phase are illustrated in Figure 4.12.

The example user interaction scenarios to generate the event log from Coscine as a real RDM system for each phase are as follows:

*Planning:* Creation of research projects, resources, application profiles, adding project members, and adopting resource quotas.

*Production:* Recording data collections and artifacts and their specialized meta-information by a project member.

*Analysis:* Updating previously documented data with new insights or restoring research repositories for exploring and extending meta information.

*Archival:* Preserving data and metadata for the long term or leaving a project after archival.



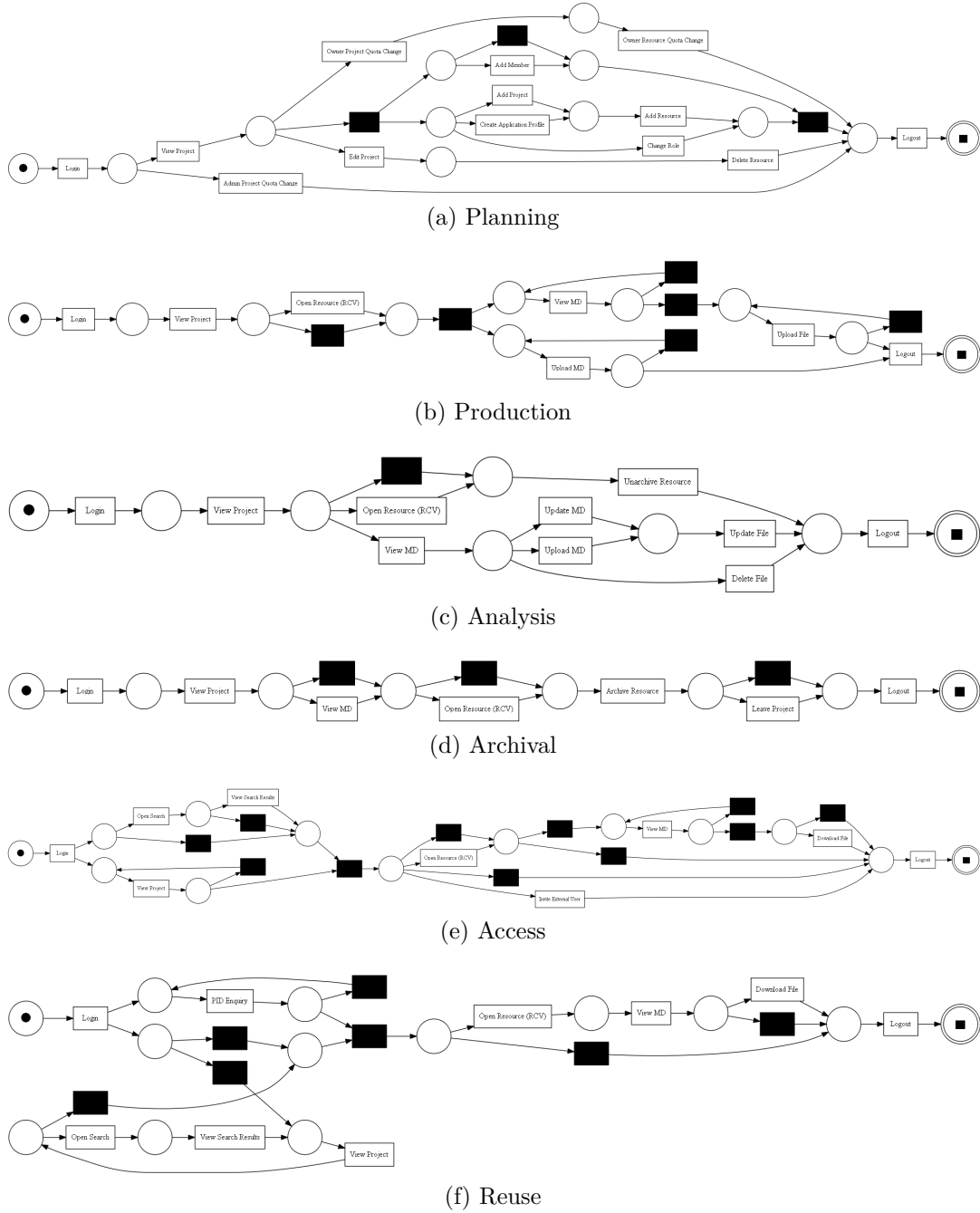


Figure 4.12: User interaction models corresponding to each RDM phase.

*Access:* User attempts to share or discover research data or metadata with other project members or externals.

*Reuse:* Internal or external user attempts to find and access data artifacts.

To ensure the distinguishability between RDM phases, pairwise fitness is computed between every two RDM phase models and presented in a confusion matrix, as shown in Figure 4.13. However, it is suggested that the pairwise fitness of models should not exceed 0.30, as it would compromise the ability to differentiate between two phases.



Figure 4.13: Pairwise fitness scores for every two prescribed user interaction models in RDM.

By leveraging process mining techniques, an effective assessment of the conformance of research projects to the prescribed RDM phases can be made, providing valuable insights into an OSP's usage patterns.

Algorithm 4.2 presents a structured methodology to compute and measure the degree of adherence of the observed research project activities to the prescribed RDM lifecycle phases. The alignment evaluation is numerically assessed through the fitness measure comparison between the observed (descriptive) and expected (prescriptive) event logs, utilizing the technique elaborated in [197]. The algorithm works on a descriptive event log  $L_{desc}$ , an assortment of prescriptive event logs  $L_{pres}$  and produces a fitness dictionary for a chosen project  $P$ .

The Inductive Miner (IM) and Alignment Analysis represent two robust, cost-effective strategies for conformance validation. With a demonstrated record of success, these methodologies facilitate the analysis and interpretation of a system's behavior alignment with its envisaged model [197].

The algorithm receives a four-element descriptive event log  $L_{desc} = (t, a, c, p)$ , each element corresponding to *Timestamp*, *Activity*, *Case Id*, and *ProjectId*, respectively. In

---

**Algorithm 4.2** Discovering RDM phases for a research project.

---

**Input:**  $L_{desc} = (t, a, c, p)$

**Input:**  $L_{pres} \in \{L_{ph_1}, L_{ph_2}, L_{ph_3}, L_{ph_4}, L_{ph_5}, L_{ph_6}\}$  where  $L_{ph_i} = (t'_i, a'_i, c'_i)$

**Output:** Fitness dictionary  $fitness_{dict}$  for a selected project  $P$

Initialize empty event log  $L_w$

$L_w \leftarrow \{\varphi(e) | e \in L_{desc}\}$

Initialize empty dictionary:  $fitness_{dict}$

**foreach**  $L_{ph_i}$  **in**  $L_{pres}$  **do**

$(net, initial_{marking}, final_{marking}) \leftarrow \text{INDUCTIVEMINER}(L_{ph_i})$

$\mathbb{N} = (net, initial_{marking}, final_{marking})$

$fitness \leftarrow \text{ALIGNMENTCHECKER}(L_w, \mathbb{N})$

$fitness_{dict} \leftarrow \text{APPEND}(fitness_{dict}, fitness, (ph_i).label)$

**return**  $fitness_{dict}$

---

addition, it accepts six distinct prescriptive event logs  $L_{pres} = (t', a', c')$ , with each corresponding to *Timestamp*, *Activity*, and *Case Id*, as portrayed in Figure Figure 4.12.

Initially, an empty event log  $L_w$  is established. Subsequently, the function  $\varphi(e)$  is invoked for every event  $e$  in the descriptive event log  $L_{desc}$  to isolate the events related to the selected project within a specific timeframe, with the findings compiled in  $L_w$ .

$$\varphi(e) = \begin{cases} e, & \text{if } (T_{start} \leq T(e) \leq T_{end}) \& (P(e) = P_{sel}) \\ \emptyset, & \text{otherwise} \end{cases}$$

An empty dictionary, labeled as  $fitness_{dict}$ , is then initialized, intended to store the fitness values for each RDM phase later.

The algorithm progresses by sequentially processing each prescriptive event log  $L_{ph_i}$  from  $L_{pres}$ . For every  $L_{ph_i}$ , the *InductiveMiner* method is executed to discover the corresponding process model, expressed as a Petri net, along with the initial and final markings, denoted as  $(net, initial_{marking}, final_{marking})$  and collectively referred to as  $\mathbb{N}$ .

Subsequently, the algorithm calculates the fitness measure between the curated descriptive event log  $L_w$  and the process model  $\mathbb{N}$  using the *AlignmentChecker* function. This fitness value signifies the conformance level between the observed project activities in  $L_w$  and the prescribed RDM phase model  $L_{ph_i}$ .

The fitness value and the label of the corresponding RDM phase are added to the  $fitness_{dict}$  dictionary. This procedure is reiterated for all the prescriptive event logs in  $L_{pres}$ .

Upon its conclusion, the algorithm outputs the  $fitness_{dict}$ , encapsulating the fitness measures between the project's observed activities and each RDM phase. This

dictionary acts as a quantitative indicator of the adherence of the descriptive event log to the stipulated RDM lifecycle phases, thereby facilitating an evaluation of a research project's alignment with the RDM's expected dimensions. It equips researchers with the capacity to examine the consistency of a research project's real-world activities with the anticipated RDM phases, thereby highlighting potential areas for improvement, optimization, and dedication to best practices in RDM.

### 4.3.3 Discussion

This study presents a methodology for discovering and outlining RDM phases and determining the current RDM phase of a research project using IM and Alignment Analysis as the two most reliable and least expensive methods for conducting conformance checking[197]. Furthermore, this problem can be framed as a machine learning task, where each operation is a feature column, and the label to be predicted is the RDM phase. In this case, a training set is necessary to assist with the prediction task.

To achieve outlining RDM phases, the granularity of collected information is crucial. If the granularity is too fine, event data abstraction may be necessary to manage the complexity of the dataset. The discovery of RDM phases is subjective to the set of predefined or prescriptive process models for each RDM phase. Creating these prescribed process models may require alterations concerning the range of activities an RDM platform could collect. Domain knowledge is essential to create these prescriptive process models to represent the RDM phases accurately.

For future work, several studies can be explored to extend the capabilities of the proposed methodology. The first expected initiative involves deploying machine learning models to predict the timeframe for the completion of distinct RDM stages, leveraging historical data. The objective is to aid researchers in proficiently planning their projects and allocating resources. Renowned Python libraries such as Scikit-learn or TensorFlow may prove beneficial in accomplishing this. Secondly, grouping research projects based on their RDM attributes is proposed to discern prevalent patterns, challenges, or best practices. To facilitate this analysis, unsupervised learning techniques such as k-means or hierarchical clustering can be employed. Lastly, an examination of role-based activity patterns in the dataset is advised. Such an investigation would shed light on the responsibilities and behaviors of various roles within the RDM process. This understanding could then be used to modify the RDM platform to better align with user needs, thereby enhancing the efficiency of the RDM process.

## 4.4 Data Collections Reusability Enhancement

The research community can accelerate scientific progress and promote effective collaboration by facilitating the reusing and repurposing of existing datasets. In the present investigation, the focus is directed toward examining two distinct types of recommender systems: Content-Based (CB) and Collaborative Filtering (CF) approaches. These analyses aim to shed light on these systems' varying methodologies to comprehensively understand their strengths and limitations for OSPs use cases to foster RDM practices.

The research findings and breakthroughs addressed within this Section have already been introduced partially in [162], [198]–[200].

### 4.4.1 Content-Based Similarity Measurement

RDM has become increasingly essential for researchers to store, manage, and share data efficiently. As digital data continues to grow exponentially, managing research data becomes more complex, and the significance of RDM has been increasingly recognized. Consequently, RDM has attracted the attention of researchers and institutions globally.

Recent studies indicate a steady increase in researchers employing RDM for their data artifacts. A 2019 survey revealed that 73% of researchers acknowledged data management's importance in their research [201]. For instance, Coscine<sup>6</sup> is an RDM platform that utilizes Shapes Constraint Language (SHACL)<sup>7</sup> and RDF<sup>8</sup> to construct knowledge graphs that can be validated according to predefined metadata profiles (or application profiles) to preserve meta information and controlled vocabularies for each data item. These metadata profiles aid in implementing good scientific practices, enhancing FAIRness [162]. Domain experts and scholars within a scientific discipline define the Application Profiles to ensure accurate metadata descriptions for each data artifact. However, facilitating the reusability of relevant research data can be challenging, particularly with the ongoing digital data explosion [24]. A recent study found that over 90% of researchers encountered difficulties locating and accessing relevant data [202], attributable to insufficient metadata or lack of standardization in metadata descriptions. Consequently, many researchers devote considerable time and resources to reproducing findings or results, leading to unnecessary duplication of efforts [22].

To tackle this issue, a CB recommendation system is suggested to enhance the discoverability and reuse of research data. These systems employ machine learning

---

<sup>6</sup>[www.coscine.de](http://www.coscine.de)

<sup>7</sup><https://www.w3.org/TR/shacl/>

<sup>8</sup><https://www.w3.org/TR/rdf11-concepts/>

algorithms to analyze research data content and provide recommendations based on content similarities. Multiple studies have investigated the application of CB recommendation systems within the context of RDM. For instance, Färber et al. [203] introduced a system for recommending data artifacts based on content and metadata, utilizing machine learning algorithms to identify similarities and suggest them to researchers. Likewise, Nair et al. [204] proposed a CB recommendation system for research papers, focusing on semantic relationships between papers. However, implementing CB recommendation systems in research data management requires a structured metadata profile that accurately portrays data properties and crucial contextual information. RDM platforms like Coscine can play a vital role in enabling CB recommendations by offering a standardized metadata profile using SHACL and RDF graph data models, fostering the accurate description of research data, and facilitating the effective operation of CB recommendation systems.

In conclusion, while the need for RDM continues to expand, the discoverability and reuse of research data remain significant challenges for researchers. CB recommendation systems present a promising solution by capitalizing on the relationships between research data items and their metadata. However, applying these systems in RDM requires a structured metadata profile, such as that provided by RDM platforms like Coscine.

### Preliminaries

RDM practices play a vital role in scientific research by enabling long-term data storage, preservation, and sharing. To optimize RDM efficiency and effectiveness, domain-specific metadata profiles have been devised to characterize data within a scientific context. The data management platform Coscine utilizes these specialized metadata profiles, permitting domain experts to define their metadata and vocabulary for describing data in their particular context. However, even with search engines available for data discovery, locating relevant data artifacts can be difficult without prior awareness of their existence [205].

Coscine’s metadata profiles are founded on the SHACL model, which stores information in *Subject*, *Predicate*, and *Object* formats [206]. This approach allows for more comprehensive and adaptable data descriptions, where the *Subject* points to the data, the *Predicate* establishes the relationship with its *Object*, and the *Object* contains meta information elucidating the *Subject*. For example, Table 4.3 displays a segment of the collected sample dataset, in which *Repository Name* and *File Name* together represent the *Subject*, while *MillingParameters*, *LaserPulseEnergy*, *LiftOutRegion*, and *PulseFrequency* serve as the *Predicate*; all values beneath the *Predicate* are *Objects*. In addition, *Object* typically consist of unstructured texts manually entered by users in a field [205].

Table 4.3: An example dataset compiled following the metadata profile by the CRC1394 research group.

Repository Name	File Name	MillingParameters	LaserPulseEnergy	LiftOutRegion	PulseFrequency	...
...	...	...	...	...	...	...
Repository-1	F1.csv	pA2.0/0.1 um 40 pA	60	Pt protection layer 14x4x4 m Trench milling (2x)	125	...
Repository-1	F2.jpg	0.43 nA / 1.5 m0.43	100	fracture cross-section 10x1.5x0.5 m	125	...
Repository-2	F3.csv	0.43 nA / 0.75 m0.23	100	Pt protection layer 14x4x4 m Trench	250	...
Repository-2	F4.csv	pA2.0/0.1 um 40 pA	100	cross-section 3x1.5x0.5 m	250	...
...	...	...	...	...	...	...

## Methodology

This section discusses the recommender system pipeline, focusing on six essential steps:

**1) Sparsity Check and Imputation:** The imputation step is vital in a recommender system pipeline, as it ensures the availability of a complete dataset with minimal missing values for further analysis and evaluation. Prior studies have indicated that imputing missing values can enhance the performance of recommender systems [207], [208]. Consequently, the approach to addressing missing values is informed by best practices from previous researches.

In this step, let  $D$  represent the input dataset. First, the input dataset  $D$  is assessed for data sparsity before imputing missing values for each feature and row. Generally, based on the benchmark study by Bennett et al., imputing missing data is considered statistically reasonable when the volume of missing data is less than 10% of the total data [209]. However, due to the cold start problem and limited sample data accessibility, the threshold is raised for data imputation to 30%, beyond which features and rows will be excluded from further evaluation or imputation.

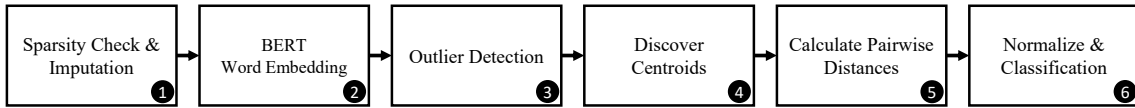


Figure 4.14: The general pipeline overview of the Content-Based recommender system.

The imputation module employs Panda dataframe data types to ascertain features' data types. For numerical data types, the K-Nearest Neighbors (KNN) imputer is utilized to predict missing values in the dataset by identifying  $k$  neighbors, which are the

most similar to a selected feature using the Euclidean distance metric. Let  $d_{i,j}$  represent the value of the  $j^{th}$  feature of the  $i^{th}$  instance in  $D$ , and let NaN denote a missing value. Let  $M$  be the binary mask matrix, where  $M_{i,j}=1$  if  $d_{i,j}$  is observed (not missing) and  $M_{i,j}=0$  if  $d_{i,j}$  is missing. Let  $\mathcal{N}_i$  be the set of  $K=3$  nearest neighbors of the  $i^{th}$  data point (excluding itself) based on the Euclidean distance between the observed features. Then, the imputed value  $\hat{X}_{i,j}$  for the missing value  $X_{i,j}$  is given by:

$$\hat{d}_{i,j} = \frac{1}{K} \sum_{k \in \mathcal{N}_i} d_{k,j} \cdot M_{k,j} \quad (4.1)$$

The imputed value is the average of the observed values for the same feature among the  $K=3$  nearest neighbors of the data point with the missing value. The binary mask matrix ensures only observed values are factored into the average. A simple imputer for the object data type is employed, filling missing values with the *most frequent* value for each column. For a given feature  $j$  of object type, let  $V_j$  represent all unique non-missing values for feature  $j$ . Then, any missing values are filled for feature  $j$  as  $d_{i,j} = \text{mode}(V_j)$ , where  $\text{mode}(V_j)$  denotes the *most frequent* value in  $V_j$ .

**2) BERT Word Embedding:** In this stage, the aim is to transform textual data into vector representations using Natural Language Processing (NLP) techniques. the Bidirectional Encoder Representations from Transformers (BERT) pre-trained model is used with TensorFlow to generate contextualized word and sentence embeddings. In contrast to traditional methods such as Term Frequency - Inverse Document Frequency (TF-IDF), BERT employs a deep learning approach and is trained on extensive text data to create rich representations of words and sentences that capture their context and meaning. A multilingual version of BERT called `bert_multi_cased`[210] is employed to support cross-lingual sentence comprehension. This model is engineered to pre-train deep bidirectional representations from an unlabeled text corpus by joint conditioning on both left and proper contexts, maintaining the distinction between lower and upper case.

Each textual data column is embedded into a 768-dimensional vector using the `bert_multi_cased` model. The size of the resulting data is proportional to the number of columns with textual data and the vector length, in addition to every column with numerical data. The data is processed in chunks to prevent overloading the processing power [211]. Thus, let  $D$  be a dataset with  $m$  samples and  $n$  features, where each column  $j$  has a data type  $dt_j \in \text{numeric}, \text{string}$ , and let  $D_s \in \mathbb{R}^{m \times k}$  represent the sub-dataset comprising the string columns and their corresponding BERT embeddings, where  $k$  is the embedding dimension (i.e.,  $k=768$  for the `bert_multi_cased` model). The final dimension of the processed data is then given by:

$$D_f \in \mathbb{R}^{m \times (k \cdot \forall dt_j = \text{string} + \forall dt_j = \text{numeric})} \quad (4.2)$$



Overall, this stage produces comprehensive vector representations of the textual data, which can subsequently be employed for downstream tasks.

**3) Outlier Detection:** This step focuses on pinpointing outlier data points or groups that display considerable dissimilarity from other data points or fail to adhere to most points' anticipated behavior. Distance-based techniques are the most prevalent methods for outlier detection [212], [213]. These algorithms calculate the distance between observations, considering a point an outlier if it is significantly distant from its neighbors. We first compute the repositories' average pairwise distance in the methodology, then estimate the corresponding **z-score** for each group. A threshold factor of 3 is employed to identify and filter out repositories that should be excluded from the recommendation task.

**4) Discover Centroids:** In this step, the aim is to identify the centroids of the clusters. K-means clustering is a widely adopted unsupervised learning method for clustering data points into K clusters based on their similarity. In this study, the K-means clustering algorithm is employed to create one cluster per repository, where each point in the cluster represents a file with a fixed number of feature dimensions derived from the BERT word embeddings mentioned earlier.

The K-means clustering algorithm begins with a random initial assignment of data points to clusters and computes the initial cluster centroids. It then iteratively assigns each data point to the nearest centroid and updates the centroids until convergence is achieved. The algorithm terminates when cluster assignments no longer change or when the maximum number of iterations is reached.

The Scikit-learn implementation of K-means clustering with default settings is utilized. After clustering, the k clusters are obtained with their respective centroids corresponding to the number of repositories after outlier detection. These centroids serve as the foundation for the final step in the recommendation task.

**5) Calculate Pairwise Distances:** Pairwise distances between centroids are computed using Euclidean or Cosine metrics. This enables the construction of a confusion matrix that captures the distances between every pair of repository centroids by employing either of these metrics. This facilitates the evaluation of similarity between items, ultimately leading to the generation of recommendations for end-users.

The Euclidean distance is defined as follows:

$$d(x,y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (4.3)$$

where  $x$  and  $y$  are two centroids and  $n$  is the number of features in the data.

The Cosine distance is defined as:

$$d(x,y) = 1 - \frac{\sum_{i=1}^n x_i y_i}{\sqrt{\sum_{i=1}^n x_i^2} \sqrt{\sum_{i=1}^n y_i^2}} \quad (4.4)$$

where  $x$  and  $y$  are two centroids and  $n$  is the number of features in the data.

**6) Normalization and Classification:** It describes the approach used to scale and categorize similarity results in the analysis. The final step of the analysis entails normalizing the confusion matrix so that it falls within a range of 0 to 1, with 0 signifying the shortest distance and 1 indicating the most significant distance between any two clusters. This normalization process is accomplished by dividing the raw values of the confusion matrix by the matrix's highest value.

Mathematically, the normalized confusion matrix is represented as follows:

$$C_{i,j}^{norm} = \frac{C_{i,j}}{\max_{i,j} C_{i,j}} \quad (4.5)$$

where  $C_{i,j}$  is the raw value of the confusion matrix and  $C_{i,j}^{norm}$  is the corresponding normalized value.

The resulting values are further transformed upon normalization to ease the evaluation process. Specifically, the normalized values are mapped onto an equal-length multi-class scale of five classes. Consequently, the range of 0 to 1 is divided into five equal intervals, with the lowest interval denoting the least similarity and the highest interval signifying the most significant similarity between cluster pairs. This casting method is chosen due to its intuitive mapping of star-like ratings, where five stars indicate the most liked, and one star conveys dislikes [214]. The resulting classification scheme is represented as follows:

$$C_{i,j}^{class} = \begin{cases} 1 & 0.8 \leq C_{i,j}^{norm} \leq 1.0 \\ 2 & 0.6 \leq C_{i,j}^{norm} < 0.8 \\ 3 & 0.4 \leq C_{i,j}^{norm} < 0.6 \\ 4 & 0.2 \leq C_{i,j}^{norm} < 0.4 \\ 5 & 0.0 \leq C_{i,j}^{norm} < 0.2 \end{cases} \quad (4.6)$$

where  $C_{i,j}^{class}$  denotes the resulting class of the normalized confusion matrix. It is crucial to emphasize that the classification scheme serves as a tool for assessing the similarity between cluster pairs. As a result, a cluster pair assigned to class 1 is regarded as the least similar, while a pair allocated to class 5 is viewed as the most similar.

#### 4.4.2 User-Interaction Based Collaborative Filtering

The emergence of big data in various scientific fields has led to the creation of numerous data-collections, and repositories of raw research data that other scholars in the field could potentially use. These data-collections are valuable resources for

researchers, as they enable the reuse and repurposing existing datasets, fostering collaboration and reducing redundant efforts in data generation [215].

CF is one of the most popular and widely used approaches in recommender systems, which can be categorized into two main techniques: User-Item Collaborative Filtering (UICF) and Item-Item Collaborative Filtering (IICF)[216]. UICF predicts a user's preferences for items by analyzing the preferences of other users with similar tastes. This technique assumes that users who have exhibited similar behavior will continue to have similar preferences in the future. To generate recommendations, UICF computes similarity measures between users and finds the most similar users (i.e., nearest neighbors) to target users. These nearest neighbors' preferences are then used to make personalized recommendations for the target user.

On the other hand, IICF predicts a user's preferences by analyzing the relationships between items. The system first computes item similarity based on user preferences in this approach. Then, for a given user, the system identifies the most similar items to those the user has previously interacted with and recommends these similar items to the user [217].

IICF recommender systems can significantly improve the findability and reusability of relevant data-collections, thus having more significant potential to contribute to more efficient RDM. IICF can identify and recommend similar data-collections based on researchers' preferences and usage patterns within a particular domain. As researchers interact with various data-collections, the IICF recommender system can establish connections between these repositories based on their similarities [218]. This approach can help researchers discover relevant data-collections they may not have been aware of, promoting the reuse of existing datasets and preventing redundant data creation. IICF has several advantages over UICF. First, IICF tends to be more stable and accurate in its recommendations. Since the number of items is often smaller than the number of users, and item characteristics change less frequently than user preferences, item-item similarity measures are more stable and less prone to fluctuations. This improves recommendation quality and consistency over time [219].

In addition, IICF recommender systems can facilitate interdisciplinary research by uncovering related data-collections across different research areas. By analyzing the usage patterns of researchers from various domains, the system can identify and recommend data-collections that are of potential interest to researchers from other disciplines [220]. This cross-disciplinary recommendation can promote interdisciplinary collaboration and enable researchers to leverage insights from related fields, ultimately leading to novel findings and scientific advancements.

Furthermore, IICF recommender systems can improve RDM by assisting in curating and organizing data-collections. As the system continuously updates its similarity measures based on user interactions, it can help data curators identify data-collections that share common characteristics, thereby allowing them to be

grouped and organized more effectively [221]. This improved organization can enhance data-collections' findability and provide researchers with a more structured and navigable data landscape.

In summary, IICF recommender systems have significant potential for enhancing data-collections' findability and reusability, leading to more efficient RDM. By identifying and recommending similar data-collections based on user preferences and usage patterns, these systems can facilitate data reuse, promote interdisciplinary research, and support better curation and organization of research data. Recent advances in recommender systems and their application to RDM demonstrate the growing importance of leveraging CF techniques to manage the ever-increasing volume of scientific data.

### Preliminaries

This section provides an overview of the input sample data format and describes the different attributes present in the event log. The event log captures user interactions with the system and consists of the following attributes: **Timestamp**, **Activity**, **UserId**, **ProjectId**, **ResourceId**, and **FileId**.

- **Timestamp** ( $t$ ): It represents the time a specific event occurs, corresponding to a user's interaction with the system. It is denoted as  $t \in T$ , where  $T$  is the set of all timestamps.
- **Activity** ( $a$ ): It is a tuple that contains activity names such as *File Upload*, *Update Metadata*, and others. It is represented as  $a \in A$ , where  $A$  is the set of all activity names. My study focuses on a subset of activities involving interactions with actual data files, denoted as  $A' \subseteq A$ .
- **UserId** ( $u$ ): It is a unique identifier assigned to each user in the system. It is denoted as  $u \in U$ , where  $U$  is the set of all unique user identifiers.
- **ProjectId** ( $p$ ): It is a unique identifier for each project within the Coscine platform. It is denoted as  $p \in P$ , where  $P$  is the set of all unique project identifiers.
- **ResourceId** ( $r$ ): It is a unique identifier for data-collections (also known as Resources). It is denoted as  $r \in R$ , where  $R$  is the set of all unique resource identifiers.
- **FileId** ( $f$ ): It is a unique identifier generated by combining the **ResourceId** and file name. In Coscine, each data collection cannot have duplicate file names. The **FileId** is represented as  $f \in F$ , where  $F$  is the set of all unique file identifiers.

At the time of the study, the event log contained 18 activities. However, to gain a perspective on user interest in research data, filtering the **Activity** list based on

the occurrence of events on `FileId`. After applying this filter, a smaller set of 7 activities is obtained, denoted as  $A'$ , which are the focus of the analysis.

Let us denote the filtering function as  $\psi$ . This function takes the event log  $E$  and returns a filtered event log  $E'$  containing only those events with activities that involve interactions with actual data files. The event log  $E$  is a set of tuples  $(t, a, u, p, r, f)$ , where  $t \in T$ ,  $a \in A$ ,  $u \in U$ ,  $p \in P$ ,  $r \in R$ , and  $f \in F$ . The filtered event log  $E'$  is a set of tuples  $(t, a', u, p, r, f)$ , where  $a' \in A'$  and  $A' \subseteq A$ .

The filtering function  $\psi$  can be defined as follows:

$$\psi(E) = \{(t, a', u, p, r, f) \mid (t, a, u, p, r, f) \in E \wedge a' \in A' \wedge a = a'\}$$

The function  $\psi$  iterates through all tuples in the event log  $E$  and selects only those tuples where the **Activity**  $a$  is an element of the reduced set of activities  $A'$ . The resulting filtered event log  $E'$  contains activities involving interactions with actual data files.

## Methodology

The following sections explore the methodology of the IICF pipeline, consisting of four main parts. These components encompass the essential steps in the pipeline, working together to provide an effective and efficient recommendation system. By breaking down the methodology into these parts, a clear and comprehensive understanding of the IICF process for recommending data collection is intended to be provided. A visual representation of the pipeline can be found in Figure 4.15, which provides an extended guide to the methodology and its stages.

**1) Calculate Implicit Rating:** To deduce an implicit rating or interest of a user in an item (Resource) based on the given criteria. In the sample data,  $a' \in A'$  represents activities, and  $A' = \{a'_1, a'_2, \dots, a'_i\}$  is a finite set of activities, where  $i$  is the total number of activities in the set  $A'$ . The frequency of activities  $a'$  is calculated by the function  $f_{a'_i}(u, r)$ , where a user  $u$  on a Resource  $r$  has executed.

A variation of the weighted-sum formula as  $W = w_{a'_1}, w_{a'_2}, \dots, w_{a'_i}$  is used. The  $w_{a'_i} \in W$  represents the weights/importance of each action to represent a user's interest in a resource. In this study, the weight of each action is pre-set as follows:

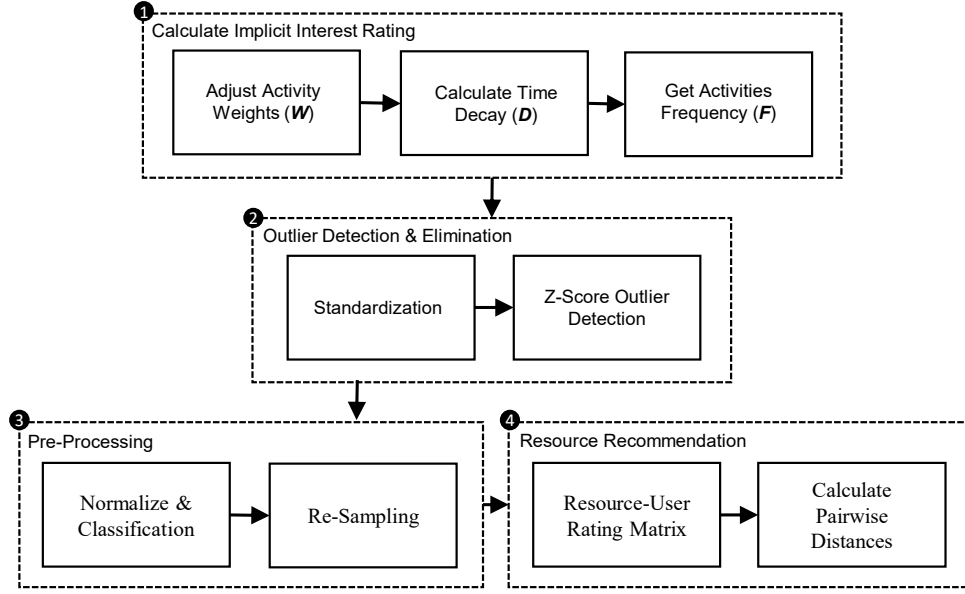


Figure 4.15: A visual representation of the IICF recommender pipeline.

$$W_{a'_i} = \begin{cases} 1.0 & w_{a'_1} = \text{Download File} \\ 0.9 & w_{a'_2} = \text{Update MD} \\ 0.9 & w_{a'_3} = \text{Update File} \\ 0.5 & w_{a'_4} = \text{Upload MD} \\ 0.5 & w_{a'_5} = \text{Upload File} \\ 0.2 & w_{a'_6} = \text{View MD} \\ 0.1 & w_{a'_7} = \text{Delete File} \end{cases}$$

Time decay  $D$  also needs to be considered, where the value of activities decreases over time. Users' interest in items can depreciate over time, which is estimated by time decay. Scholars' research interests commonly change over time; thus, recent user interest in an item is incorporated by time decay. An exponential decay function with a decay factor  $\lambda$  is used to consider this. Let  $t_{(u,r)}$  be the time elapsed since the last action of any type by the user  $u$  on the Resource  $r$ . The time decay is denoted as:

$$D_{(u,r)} = e^{-\lambda t_{(u,r)}}$$

The decay factor  $-\lambda$  controls how quickly the value of an action depreciates over time. A larger value of  $\lambda$  will result in a faster decay.

Putting everything together, the implicit user  $u$  ratings on a resource  $r$  are calculated with the following formula:

$$Rating_{(u,r)} = \sum_1^{a'_i} w_{a'_i} \cdot D_{(u,r)} \cdot f_{a'_i}(u,r)$$

This formula considers the frequency of actions, the importance of each action, and time decay. It can generate recommendations by ranking items based on their predicted implicit ratings for a given user.

**2) Outlier Detection and Elimination:** To identify and remove outliers from the  $Rating_{(u,r)}$  column, a standardization technique known as **z-score** is utilized. The **z-score** of observation is calculated by subtracting the mean ( $\mu$ ) of the dataset and dividing the result by the standard deviation ( $\sigma$ ):

$$z_i = \frac{x_i - \mu}{\sigma},$$

where  $x_i$  is an individual  $(u,r)$  rating, and  $z_i$  is the corresponding **z-score**. By transforming the  $Rating_{(u,r)}$  column into **z-scores**, the relative positions of each rating within the dataset's distribution can be identified. In this case, a threshold value of 3 is defined to identify and eliminate outliers. Observations with a **z-score** greater than the threshold value (in terms of absolute value) are considered outliers:

$$\text{Outliers} = x_i \mid |z_i| > 3.$$

By applying this outlier detection and elimination method, extreme values that may adversely impact the performance of the IICF pipeline can be filtered out.

**3) Pre-Processing:** The rating values are scaled into a consistent range, thereby improving the performance of the recommender system. The aim is to transform the ratings into a standardized range between 0 and 1. To normalize the ratings, the min-max scaling technique is employed, which scales the values of the  $Rating_{(u,r)}$  column based on their minimum ( $min$ ) and maximum ( $max$ ) values. The normalized rating ( $Rating'$ ) for each original rating ( $Rating$ ) can be calculated using the following formula:

$$Rating' = \frac{Rating - min}{max - min},$$

Where  $Rating'$  is the normalized rating, and  $min$  and  $max$  are the minimum and maximum values of the ratings, respectively.

Following the normalization process, the obtained values undergo an additional transformation to enhance the evaluation procedure. In particular, the normalized

values are allocated to a multi-class scale comprising five equally sized classes. Consequently, the range from 0 to 1 is partitioned into five uniform intervals, where the smallest interval signifies minimal similarity and the largest interval indicates maximal similarity between pairs of clusters. The subsequent classification scheme can be described as follows:

$$Rating'_{u,r}{}^{class} = \begin{cases} 1 & 0.8 \leq R'_{u,r} \leq 1.0 \\ 2 & 0.6 \leq R'_{u,r} < 0.8 \\ 3 & 0.4 \leq R'_{u,r} < 0.6 \\ 4 & 0.2 \leq R'_{u,r} < 0.4 \\ 5 & 0.0 \leq R'_{u,r} < 0.2 \end{cases}$$

where  $Rating'_{u,r}{}^{class}$  represents the resulting class of the normalized rating.

Additionally, a balanced dataset is essential to ensure the performance and accuracy of machine learning algorithms. To balance the dataset, the mean number of instances for each of the five classes is calculated. Let  $C_i$  represent the  $i^{th}$  class and  $|C_i|$  denote the number of instances in that class, where  $i \in 1, 2, 3, 4, 5$ . The mean number of instances for all classes can be calculated as follows:

$$\bar{m} = \frac{1}{5} \sum_{i=1}^5 |C_i|.$$

Whether to oversample or undersample each class is determined based on the mean  $\bar{m}$ . For a class  $C_i$ , if  $|C_i| < \bar{m}$ , oversampling is performed by generating additional instances until the number of instances in the class reaches the mean. Conversely, if  $|C_i| > \bar{m}$ , undersampling is performed by randomly removing instances until the number of instances in the class equals the mean. The re-sampling process can be defined as follows:

$$C'_i = \begin{cases} \text{oversample}(C_i, \bar{m}) & \text{if } |C_i| < \bar{m}, \\ \text{undersample}(C_i, \bar{m}) & \text{if } |C_i| > \bar{m}, \\ C_i & \text{otherwise.} \end{cases}$$

During the oversampling process, the algorithm generates new **UserId**  $u'$  for every other occurrence of **ResourceId**  $r$  to ensure no duplicate pairs of **UserId**  $u$  and **ResourceId**  $r$ . This can be achieved by creating a unique identifier for each newly generated instance in the oversampled class, guaranteeing the absence of duplicates.

Applying this re-sampling technique ensures that each class has an approximately equal number of instances, contributing to a balanced dataset. In addition, this



process enhances the robustness and accuracy of the IICF pipeline by mitigating potential biases arising from imbalanced class distributions.

**4) Resource Recommendation:** Given the pair of  $Rating_{(u,r)}$ , which represents the rating of every user  $u$  for a resource  $r$ , a user-resource matrix  $M$  is created. Each row of matrix  $M$  corresponds to a resource and contains a vector of user ratings for that Resource:

$$M = \begin{bmatrix} Rating_{(1,r_1)} & Rating_{(2,r_1)} & \dots & Rating_{(n,r_1)} \\ Rating_{(1,r_2)} & Rating_{(2,r_2)} & \dots & Rating_{(n,r_2)} \\ \vdots & \vdots & \ddots & \vdots \\ Rating_{(1,r_m)} & Rating_{(2,r_m)} & \dots & Rating_{(n,r_m)} \end{bmatrix},$$

Where  $n$  is the total number of users, and  $m$  is the total number of resources.

To measure the similarity between resources, pairwise distances are calculated between every pair of resources using both Cosine and Pearson distance metrics. Let  $R_i$  and  $R_j$  denote the vectors representing resources  $i$  and  $j$  in the user-resource matrix  $M$ . The Cosine distance  $d_C(R_i, R_j)$  and Pearson distance  $d_P(R_i, R_j)$  can be calculated as follows:

$$d_C(R_i, R_j) = 1 - \frac{R_i \cdot R_j}{|R_i| |R_j|}$$

$$d_P(R_i, R_j) = \frac{\sum_{i=1}^n (R_i - \bar{R})(R_j - \bar{R})}{\sqrt{\sum_{i=1}^n (R_i - \bar{R})^2} \sqrt{\sum_{j=1}^n (R_j - \bar{R})^2}}$$

By computing the pairwise distances, we can identify the similarities between resources. This information is then used to generate resource recommendations for each user based on their preferences and the similarities among resources. The resulting recommendation system enables users to discover and engage with resources most relevant to their interests and needs. For example, the pairwise similarity of resources is stored in a separate confusion matrix  $M_C$  using Cosine (Figure 5.23a), and  $M_P$  using Pearson (Figure 5.23b).

### 4.4.3 Discussion

The literature review results highlight the various advantages associated with Content-Based recommender systems, such as the ability to offer personalized suggestions, alleviate information overload, and enhance the quality of retrieved resources, particularly when addressing cold start issues. Nevertheless, these systems face inherent

limitations due to their dependence on metadata and sample data quality. For the IICF recommender system, the literature review also emphasizes the numerous benefits of this recommender system, such as their ability to discover and utilize underlying patterns in user preferences and lower computational complexity compared to UICF and resilience to user profile changes [222]. However, these systems are not without limitations, as they may suffer from data sparsity and might not perform well in situations with insufficient user interactions.

Similar to Content-Based recommender systems, Euclidean, Cosine, and Pearson distances are frequently employed in IICF systems to measure the similarity between items. These distance metrics result in a diagonally symmetric similarity matrix, reflecting the natural comparison process between items.

For future research, the aim is to enhance the accuracy and applicability of the Content-Based recommender system across a more comprehensive array of disciplines. One crucial aspect involves assessing the sensitivity of the BERT pre-trained model to multilingual datasets, considering that the sample data includes manually entered values in German and English. While the BERT model possesses cross-lingual sentence comprehension capabilities, additional evaluation is necessary to determine its performance with multilingual datasets, as advised by Devlin et al.[210]. Furthermore, our research indicates that sustaining uniformity in data attributes, datatype information, and semantic definitions allows for examining intersecting profiles. Consequently, this enables the recommender system to accommodate a more comprehensive array of datasets.

In the context of IICF, the investigation of the impact of incorporating additional similarity metrics could be fruitful, such as Jaccard similarity, on the performance of the IICF recommender system. Furthermore, as the quality of recommendations in IICF systems depends heavily on user interactions, investigating methods to handle cold start issues, such as incorporating Content-Based information or user demographic data, can enhance the system's overall performance and utility [223].

Moreover, it is suggested to analyze the accuracy of the recommenders after including data augmentation techniques such as contextual word augmentations offered by the *nlpaug*<sup>9</sup> Python library or utilizing Principal Component Analysis (PCA) to reduce dataset dimensionality following vectorization and word embeddings. Lastly, a naive approach of replacing missing values with the most common value within each column is adopted for non-numerical columns. Nevertheless, RDF knowledge graphs supply accurate information on the data type for every feature. Therefore, this type-property could be integrated into the sample dataset to facilitate the selection of suitable methods for data imputation or vectorization according to a feature data type.

In conclusion, Content-Based recommender and IICF systems offer unique advantages and face distinct limitations. Future work should investigate methods to overcome

---

<sup>9</sup><https://nlpaug.readthedocs.io>

these limitations and explore hybrid approaches that leverage the strengths of both techniques, ultimately leading to more accurate, robust, and personalized resource recommendations.



## 5 Case Studies

This chapter assesses and demonstrates the validity of the methodologies on the data analytics framework for OSPs. By conducting a series of case studies, the goal is to evaluate the effectiveness of the proposed approaches in various contexts and applications. These case studies will provide valuable insights into the practical implications of the methodologies and showcase their potential benefits for researchers, data managers, and platform administrators.

The case studies evaluate the discussed data extraction techniques, event data abstraction, knowledge discovery, and applying suitable recommender systems on top of the acquired datasets. A rigorous assessment of these methods in real-world scenarios aims to demonstrate their applicability and relevance to Open Science processes. Furthermore, any potential limitations or challenges in their implementation and applicability are sought to be identified, allowing for the refinement of the methodologies and providing actionable guidance for their usage in practice.

The first case study investigates the effectiveness of the data extraction techniques in capturing relevant data from diverse sources and formats while preserving the essential information required for subsequent analysis. By examining the efficiency and accuracy of these methods, it can be ensured that the data collected is comprehensive and representative of the actual RDM activities.

The second case study evaluates the event data abstraction technique, which involves transforming raw data into a structured format suitable for further analysis. The quality of the resulting event logs and their ability to capture the complexities of RDM processes are assessed. This evaluation will provide insights into the abstraction methods' robustness and ability to facilitate meaningful analysis.

The third case study explores the potential of knowledge discovery techniques to uncover valuable insights and patterns within the event data. By applying various algorithms and techniques, the utility of this approach in revealing hidden relationships, trends, and user behaviors within the Open Science ecosystem is demonstrated.

Finally, the fourth case study examines the implementation of recommender systems on top of the acquired datasets. The performance of these systems in providing personalized and context-aware recommendations to users, thereby enhancing the overall user experience and promoting research data reusability, is evaluated.

Through rigorous evaluation and refinement, a contribution to the development and optimization of RDM practices is made, fostering greater collaboration, efficiency, and transparency within the research community.

## 5.1 Data Extraction

This Section presents two case studies exploring Centralized and Server-Side data extraction techniques employed to analyze and understand complex systems and their underlying processes. These case studies aim to demonstrate the applicability and effectiveness of different extraction methods in real-world scenarios, highlighting their respective strengths and limitations. Furthermore, by examining the outcomes of these techniques in diverse contexts, one can gain valuable insights into their practical implications and provide a solid foundation for developing new and improved data extraction and analysis approaches.

### 5.1.1 RWTH Distributed Services

This case study aims to investigate and discover the process model of interconnected microservices and apply process mining on top of aggregated data driven by the approach. It focuses on microservices that utilize OAuth as their authorization interface before executing software components in response to user requests to access functionalities or resources. The study examines how distributed services of RWTH University work together to process and execute users' requests and evaluates the microservices' end-to-end processes. Figure 5.1 illustrates a presumed interconnection of RWTH distributed services in a tree representation.

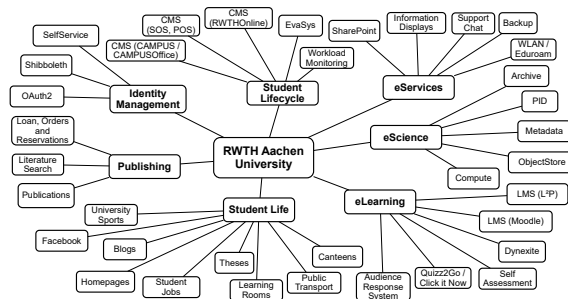


Figure 5.1: The assumed representation of RWTH distributed services, adapted from [165].

Process mining is applied on top of the aggregated data to visualize and analyze the microservices' overall process. The hypothesis is that the centralized logging

approach can produce suitable datasets that support process mining techniques to produce and extract business process models from distributed systems in order to reverse engineer and draw insights into the dynamic behavior of interconnected services at RWTH Aachen University. By doing so, it is expected to improve the overall user experience by drawing the attention of development teams for further improvements by locating bottlenecks and uncovering deficiencies in the overarching distributed system architecture.

## Dataset

The complexity and challenges of extracting high-quality data from raw datasets are often underestimated or inadequately addressed, particularly in interconnected and heterogeneous environments. The data preparation process is time-consuming, error-prone, and iterative, but it is crucial to ensure the data is accurate and valuable [224]. Furthermore, factors such as noise, outliers, undetected variations, redundancies, and missing values can influence the data's suitability, making it vital to prepare the data before conducting a data analysis study. Consequently, a standard procedure for offline data preprocessing has been developed to guarantee optimal data quality. Initially, the data and its features are filtered and anonymized. Subsequently, noise and outliers are identified through a **Z-Score** analysis [225] and eliminated from the sample data. Finally, the reliability of the obtained data is assessed based on whether the sample data meets the standards of completeness, integrity, accuracy, and consistency mentioned in Section 3.2.

As described earlier, logs should contain **Timestamp**, **Case Id**, and **Activity** to enable process investigations. The centralized logging approach assists us in aggregating datasets that meet this necessary standard. Every user interaction with the Client-Side initiates a series of activities and calls upon various resources and microservices to respond appropriately. The sample dataset, spanning one week, comprises 102,981 cases, 776,370 events, and 13 activities. An illustration of the collected dataset can be seen in Table 5.1.

Disco [111] is employed for the process mining tasks and model analysis, as it offers excellent usability, reliability, and performance. Disco uses the Fuzzy Miner [54] to generate simplified DFG models, emphasizing the most frequent activities. This is particularly beneficial for identifying crucial task paths within a process. Each Case represents a complete process variance, including one or multiple services in the execution lifecycle. In the model, nodes represent activities, and edges represent execution paths. The numbers displayed on the nodes indicate activity frequencies, while the numbers on the arcs represent the frequencies of the corresponding execution paths.

Table 5.1: A sample dataset  $L_o$  captured from the distributed services of RWTH Aachen University.

timestamp ( $t_o$ )	caseId ( $c_o$ )	userHashId ( $u_o$ )	activity ( $a_o$ )	microservice ( $m_o$ )
...	...	...	...	...
2019-12-16 08:59:55	Izo9VZ9vz...	dhcwb4MII...	GetInfo	userinfo.rwth
2019-12-16 08:59:56	Izo9VZ9vz...	dhcwb4MII...	GetNotifications	editpns.rwth
...	...	...	...	...
2019-12-16 09:00:23	fW6O7i85R...	LOrv8dNur...	GetWeeks	campus.rwth
...	...	...	...	...
2019-12-16 09:00:32	CF12boZC3...	nfW6O7i85...	GetReport	campus.rwth
2019-12-16 09:00:32	CF12boZC3...	nfW6O7i85...	IsAuthorized	editpns.rwth
2019-12-16 09:01:07	nCkCF12bo...	nfW6O7i85...	GetAllFiles	simplearchive.rwth
2019-12-16 09:03:15	eoebGbLfl...	nfW6O7i85...	GetSchema	metadataadmin.rwth
...	...	...	...	...
2019-12-16 09:15:18	mcnneXNOe...	6XE8m5KaE...	GetReport	campus.rwth
2019-12-16 09:15:19	mcnneXNOe...	6XE8m5KaE...	GetNotifications	editpns.rwth
...	...	...	...	...
2019-12-16 09:34:46	skPXZjMwR...	9PtIBdGFS...	GetPicturesForUser	picturemanagement.rwth
...	...	...	...	...

## Results

Figure 5.3 illustrates the software interaction model obtained by applying the Fuzzy Miner to the sample data shown in Table 5.1. By evaluating the discovered process model, insights into the system’s actual behavior can be gained, making it possible to identify which services operate as groups and which function independently. Furthermore, it becomes apparent that *editpns* (the notification service) and *userinfo* (authorization service) serve as middleware microservices; if either of them fails, it leads to significant disruptions in the entire system. Additionally, from Figure 5.3b, it can be observed that these services are the most frequently active, which can help system architects direct the development team’s attention to the services in the highest demand.

In addition to providing deep and insightful perspectives on the overall software architecture, modeling processes within distributed services allows for assessing the impact of newly developed software components and microservices on the other related services. Moreover, concentrating on a particular use case scenario makes it possible to identify traces’ most common starting and ending points within the entire distributed environment.

Furthermore, Figure 5.3b highlights the bottlenecks in the SUS. The *editpns* and *userinfo* services substantially influence application performance across the SUS as they handle many requests. Using the collected dataset, it becomes possible to examine the system at various levels of granularity by focusing on the methods



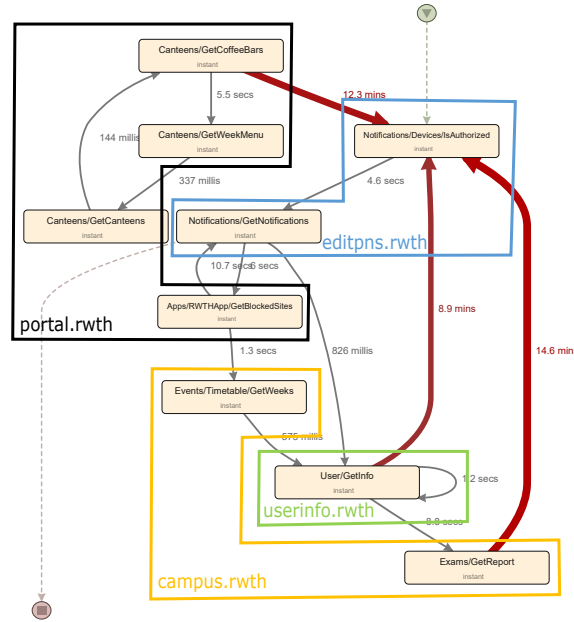
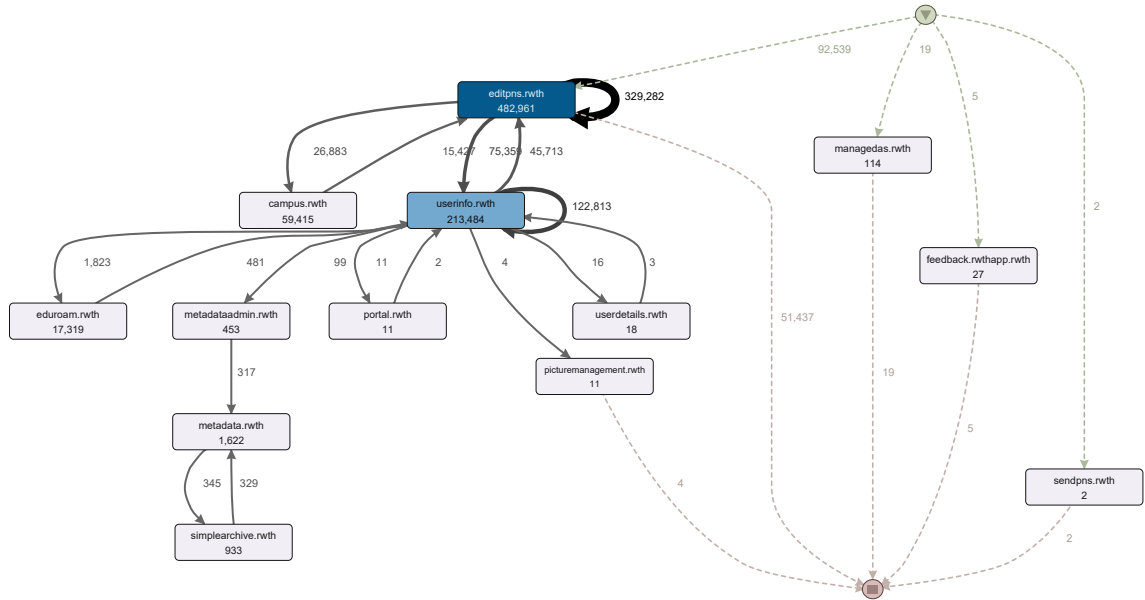


Figure 5.2: RWTH IT Center software components interconnection and performance evaluation. Each colored group signifies a service and displays a hierarchical representation.

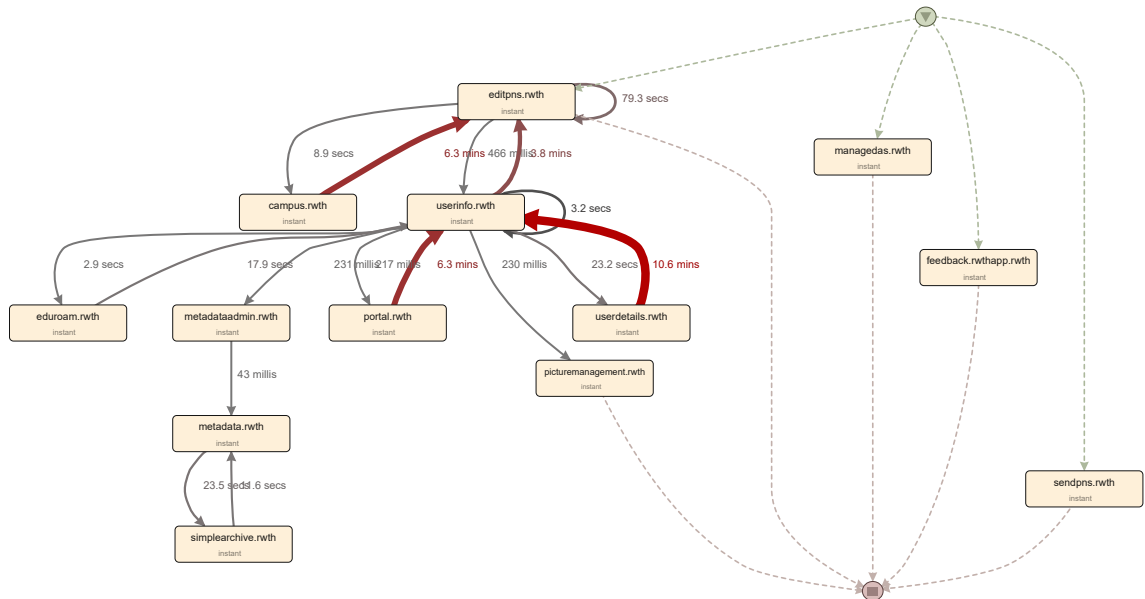
activated within each service. This approach allows for identifying the primary sources of delays in user responses at the component level.

For instance, using the constructed hierarchical model displayed in Figure 5.2, a deeper investigation into the *editpns* service was conducted to identify potential causes of inefficiency. With the help of the discovered model, it can be directly pinpointed which additional software components from the other three services contribute to the bottleneck issue in the *editpns*. For example, as shown in Figure 5.2, the method calls *Canteens/GetCoffeeBars* from the *portal*, *User/GetInfo* from *userinfo*, and *Exams/GetReport* from the *campus* are candidates for further investigation, as they appear to be contributing to inefficiencies in processing requests for the *Notifications/Devices/IsAuthorized* method in the *editpns* service.

Employing the Centralized data acquisition method makes it possible to uncover previously unknown architectural issues across distributed services and inefficiencies within a system. For example, unnecessary method calls, methods or interservice loops, unneeded authorization checks, callbacks, and code inefficiencies within software components were found. The results demonstrated that the Centralized logger could enable the collection of datasets necessary to extract descriptive business process models from distributed services accurately.



(a) Absolute frequency of service usage. Darker node or edge color represent higher case frequency.



(b) Performance analysis and bottleneck discovery. The thickness and color of edges represent higher delays in a process.

Figure 5.3: Process model discovered using Fuzzy Miner at service level on the SUS.

### 5.1.2 Coscine

As mentioned, Coscine is an example OSP that assists researchers with data management tasks and supports research projects to achieve the FAIR maturity paradigm. Researchers can use Coscine to store their research data, provide specialized meta-data, and collaborate on research projects. Figure 5.4 presents a screenshot of the Resource Content View (RCV) web page on Coscine. While Coscine is still under active development, it benefits from the microservices software architecture model, accompanied by the Server-Side logger technique.

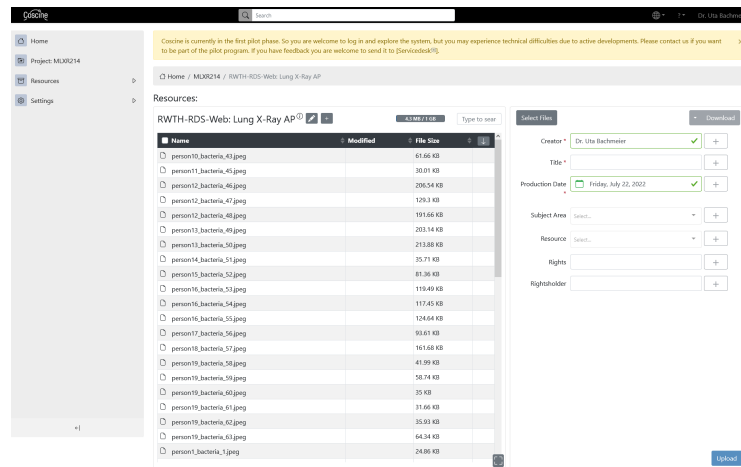


Figure 5.4: Coscine’s web page for managing research data and metadata.

According to the Server-Side logging technique, the sample dataset collected for this case study contains various APIs, with each software module interacting with the system, making each API distinguishable by its name. For example, the same API processes all requests handling file modifications. Thus, analyzing the log files obtained from the Server-Side logging technique makes it possible to pinpoint the root cause of bugs. Similarly, Coscine system administrators use Server-Side logging techniques to detect abnormalities and trace the root cause of system issues. Doing so helps prevent future occurrences of these problems and enhances system performance.

### Dataset

Coscine’s Server-Side logger enables the collection of user-based RDM activities while formalizing appropriate data objects and relationships between attributes. The acquired information consists of user requests from the Server-Side and is processed according to specific application domains. Coscine generates and captures this data in a serialized JSON object format, allowing for easy scalability to incorporate

additional attributes and entities without extending database tables. The data fields provide detailed insights into the sequence of actions and their respective RDM-relevant entities. For instance, Listing 5 represents a JSON object triggered by a user attempting to update research data entry metadata, demonstrating the Server-Side logger’s extensibility with additional attributes. From a single user action, meta information such as its PID, selected metadata schema, percentage of metadata provided, discipline, and other associated properties can be deduced. The sample dataset for this case study contained five months’ worth of user activities on Coscine, accumulating 4332 events.

In line with responsible data mining principles, it is crucial to maintain privacy within user-based datasets [67]. Therefore, Coscine, as the system under study, generates and contains no human-readable names. Instead, GUIDs are used as unique identifiers to study user actions and behavior without compromising the user’s identity.

Subsequently, the DA4RDM web interface introduced in Section 4.2 provides the means to utilize the dataset produced by Coscine or any other RDM system via a user interface. Moreover, it allows the selection of a pre-processing pipeline to evaluate and transform data samples into a new data format suitable for data analysis algorithms. Accordingly, all JSON objects are accumulated and stored in a relational database, ready to import to DA4RDM. For example, Table 5.2 exemplifies a dataset converted into columns of features and labels, ready for in-depth post-processing and modeling.

```
1      {
2          "Activity": "Update Metadata",
3          "Timestamp": "1579109897",
4          "UserId": "29613-d8...",
5          "RoleId": "be29c-4e...",
6          "SessionId": "4b15f...",
7          "ProjectId": "4e9f-97...",
8          "ResourceId": "ef9175...",
9          "AppProfile": "EngMeta",
10         "MetadataCompleteness": "70%",
11         "License": "MIT",
12         "Discipline": ["Mechanical Engineering"],
13         "Organizations": ["ETH", "Darmstadt"]
14     }
```

Listing 5: Sample JSON object from Coscine’s Server-Side logger capturing a user action.

Table 5.2: Data objects prepared and transformed for data analysis using DA4RDM.

Activity ( $a_s$ )	Timestamp ( $t_s$ )	UserId ( $c_s^1$ )	RoleId ( $c_s^2$ )	SessionId ( $c_s^3$ )	ProjectId ( $c_s^4$ )	ResourceId ( $c_s^5$ )	AppProfile ( $c_s^6$ )	MetadataComp. ( $c_s^7$ )	License ( $c_s^8$ )	Discipline ( $c_s^9$ )	Organizations ( $c_s^{10}$ )
...	...	...	...	...	...	...	...	...	...	...	...
View Project	1579108918	29613-d8...	be29c-4e...	4b15f...	4e9f-97...	NULL	NULL	NULL	MIT	Mechanical Eng.	ETH, Darmstadt
View Resource	1579109840	29613-d8...	be29c-4e...	4b15f...	4e9f-97...	e9175...	EngMeta	NULL	MIT	Mechanical Eng.	ETH, Darmstadt
Update Metadata	1579109897	29613-d8...	be29c-4e...	4b15f...	4e9f-97...	Xn3on4...	EngMeta	73%	MIT	Mechanical Eng.	ETH, Darmstadt
...	...	...	...	...	...	...	...	...	...	...	...

## Results

The subsequent section emphasizes qualitative and quantitative outcomes from two case studies utilizing DA4RDM. First, user activities related to resources as data collections were examined, followed by exploring the overall system performance for research projects on Coscine. This was done by uncovering process models for complex user interaction paths and pinpointing non-functional requirements within the system.

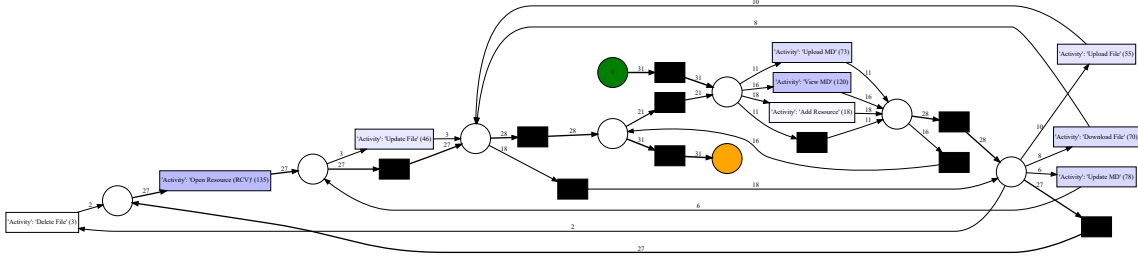


Figure 5.5: Activity frequency analysis on resources with the Petri-net model.

In the first case study, a frequency analysis was carried out using the IM on resource PIDs, which led to the discovery of a Petri-net model [226] representing the most common user journey paths while interacting with various resources and research data. Utilizing DA4RDM, as depicted in Figure 5.5, it was found that the number of executions for the *Open Resource(RCV)* activity was unusually high compared to the number of incoming and outgoing transitions. This suggests the presence of loops in the code, highlighting the need to refactor the corresponding code to avoid unnecessary Server-Side requests.

In the second case study, the overall performance of the RDM system was examined as displayed in Figure 5.6 using the DFG process model. With the assistance of a domain expert, the findings (indicated by red circles) were evaluated against key performance indicators. Despite the process model's anticipated complexity and unstructured nature due to users' freedom to interact within the system, DA4RDM effectively helped us identify previously unknown bottlenecks. For example, the transition from *Open User Management* to *View Users* should only take a few seconds, yet the process of creating new projects (transition from *Add Project* to *Open Project*) on Coscine was found to take more than 70 seconds. Consequently, uncovering

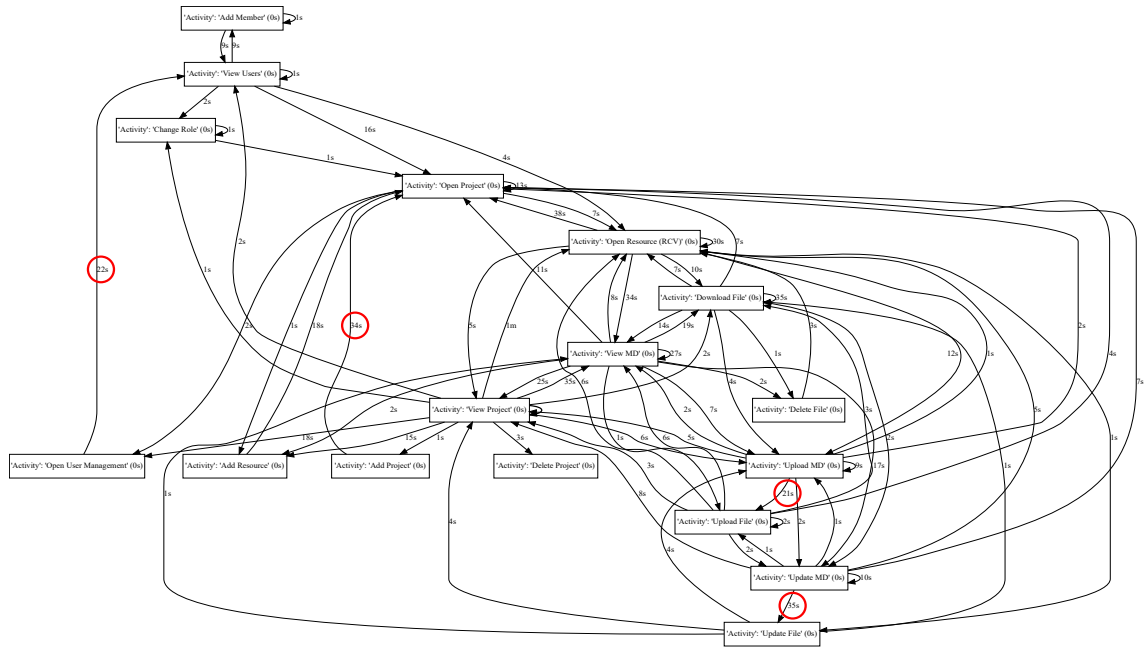


Figure 5.6: Performance analysis using data flow graph at the research projects level, with red circles highlighting violated KPIs.

evidence of software execution bottlenecks led to the developers' bringing attention to investigating and addressing the causes of performance issues.

The empirical analysis of a real dataset using the Server-Side logger suggests that the cyclical order of operations within the RDM lifecycle, as depicted in Figure 1.3, might not accurately represent the actual research data lifecycle. Moreover, despite a limited data sample size, both use cases demonstrate DA4RDM's capabilities and how the web application can be expanded to accommodate non-technical users. This allows them to examine their research projects against specific criteria by extending data modeling techniques or visualization for a targeted use case.

Even though maintaining a Server-Side logger poses challenges, the findings showcase the advantages of the Server-Side logging technique in an OSP. This technique allows data scientists to obtain fine-grained, expandable, and adaptable datasets tailored to data analysis projects, such as user behavior studies, data analytics, or uncovering non-functional requirements.

### 5.1.3 Summary

In conclusion, the two case studies examine methods for extracting datasets from OSPs using a Centralized logger and a Server-Side logger. The Centralized logger

case study investigates the process model of interconnected microservices of RWTH Aachen University and applies process mining to the aggregated data. It successfully uncovers architectural issues across distributed services and inefficiencies within a system. The Server-Side logger case study focuses on Coscine as an example OSP, demonstrating its ability to identify previously unknown bottlenecks and providing valuable insights into user behavior and system performance. Both methods have their merits, but they also come with certain limitations.

Merits of both methods can be seen in various aspects. Firstly, both approaches provide valuable insights into the system's actual behavior and facilitate the discovery of process models. Secondly, they enable the identification of bottlenecks in the system, sources of delays in user responses, and inefficiencies within the distributed system. Thirdly, they offer fine-grained, expandable, and adaptable datasets tailored to specific data analysis projects while providing a holistic view of interconnected microservices. Lastly, both methods can be extended to enable service providers to examine their underlying complex processes against specific use cases using fine-granular datasets rendered by these logging approaches.

Nevertheless, the Centralized logger and the Server-Side logger methods have certain limitations. One significant challenge is that both approaches require considerable effort to integrate and maintain logging and may impose additional overhead on the system. Furthermore, analyzing the aggregated data from either method may be challenging due to the volume and variety of information collected, especially when dealing with large-scale distributed systems. Additionally, the datasets collected may not always provide a comprehensive view of the entire system, as they focus primarily on individual services or specific aspects of the system tailored to the logger specification.

The validity of the suggested Hybrid logger is evaluated in the next Section by understanding the merits and limitations of both the Centralized logger and the Client-Side logger methods. It combines the strengths of each approach while addressing their respective challenges. The Hybrid logger aims to offer comprehensive insights into user activities and provide adaptable datasets for various data analysis projects while minimizing the overhead and complexity associated with maintaining and analyzing the logs.

## 5.2 Event Data Abstraction

To demonstrate the quality and reliability of the proposed method for the abstraction of event logs generated by the Hybrid logger, it is essential to test the system with at least two experiments, as recommended by Dean et al.[227]. As a result, the approach’s effectiveness, robustness, and generalizability in obtaining readable and usable abstracted event logs is validated by conducting two rounds of empirical evaluations in real-life settings. The event logs are gathered using the Hybrid event logger discussed in Section 3.4, sourced from information systems provided by RWTH-Aachen University, which manage metadata and archive research data. Furthermore, adherence to the guidelines outlined by [228] is maintained to ensure that the collected event logs comply with EU GDPR privacy regulations. As previously mentioned, an  $n:m$  relationship exists between Client and Server activities for the system under investigation. Consequently, each user activity initiates a series of Server-Side activities to process and execute the user’s request for a Client-Side operation. Client-Server event logs from two web applications, specifically the Metadata Manager and Data Archiving tool, are gathered and analyzed for the case studies and abstracted according to the technique discussed in Section 4.1. The validity and scalability of the approach are evaluated with the help of these two case studies.

### 5.2.1 Research Data Archiving Tool (SimpleArchive)

The Data Archiving tool is a web application designed for researchers at RWTH-Aachen University to archive research data and restore files on demand. Users of this service can set a storage expiration date to comply with the requirements of funding organizations for research projects, ensuring that research data remains accessible even after a project’s conclusion. The most common user activities include uploading/downloading data, searching for a research record, and restoring research data. This case study gathers event logs over six months, comprising 413 case Ids and 10,487 events. The Data Archiving tool generates 19 unique Client-Side activities and 28 Server-Side activities responsible for processing users’ Client-Side requests.

Table 5.3: Hybrid logger performance assessment on Data Archiving Tool using Decision Tree and Random Forest Classifiers.

	RMSE	MAE	Precision	Recall	F1	Accuracy
Decision Tree	0.31	0.19	0.89	0.87	0.87	0.85
Random Forest	0.54	0.51	0.87	0.85	0.85	0.83

Table 5.3 provides performance statistics of the Hybrid logger discussed in Section 3.4 to acquire the dataset. As shown in Table 5.3, the Decision Tree (DT) classifier consistently outpaces the Random Forest (RF) model across all metrics. The former yields



superior results regarding both Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), with respective scores of 0.31 and 0.19 compared to the latter's 0.54 and 0.51. This suggests the DT model boasts lower average errors when predicting user activities. The model also performs excellently in Precision, Recall, and F1 scores. Precision and Recall scores of 0.89 and 0.87, respectively, outstrip the RF's 0.87 and 0.85, signifying a higher proportion of correctly predicted positive activities and better identification of true positives. The F1 score of the DT model, standing at 0.87 against the RF's 0.85, attests to a superior balance between Precision and Recall. Additionally, the DT model manifests higher overall accuracy at 0.85 compared to the RF's 0.83, demonstrating its enhanced capacity to predict user activities across the entire dataset accurately. The Figure 5.7a depicts the Client-Side process model discovered from the Data Archiving Tool with the help of the Hybrid logger applying the DT classifier.

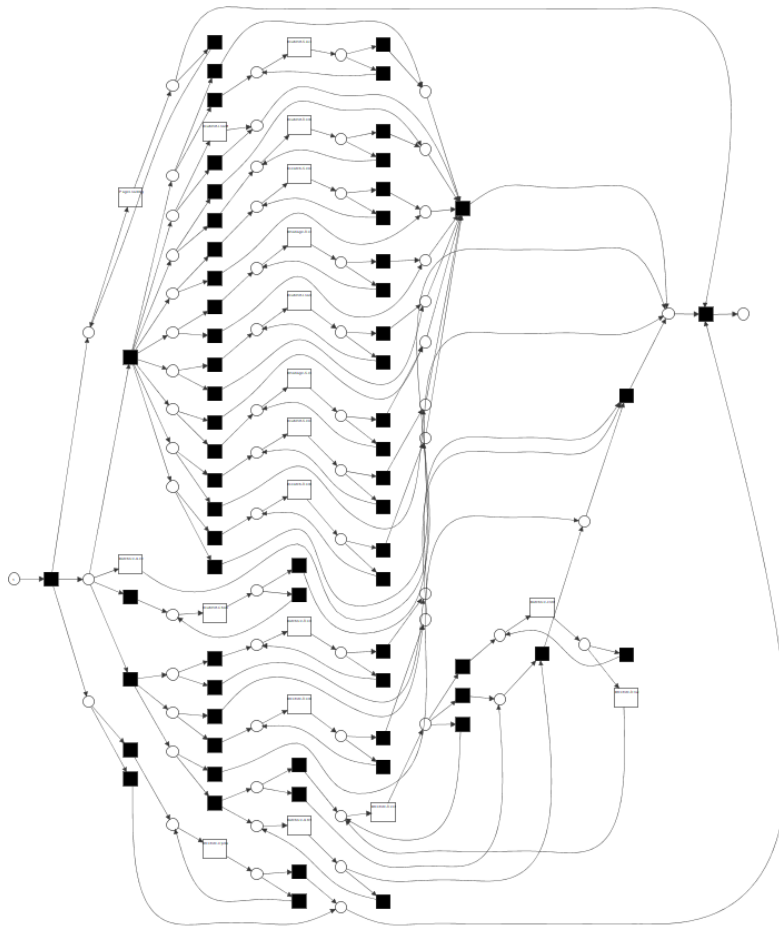
Table 5.4: Abstraction of sample dataset for Data Archiving Tool, and its effect on the number of Client and Server-Side activities.

PCC Similarity Threshold	1	97-1	91-1	88-1	<b>85-1</b>	82-1	55-1	28-1
Fitness	1	0.98	0.88	0.83	<b>0.81</b>	0.76	0.70	0.59
# Client Activities ( $a_c$ )	19	18	17	15	<b>14</b>	12	10	4
# Events	10487	9973	9481	8297	<b>7920</b>	6553	5481	2203

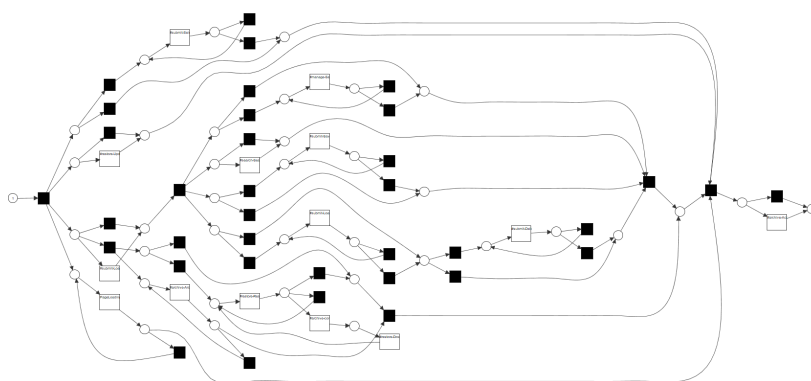
**Results:** Figure 5.7a and Figure 5.7b respectively depict the discovered process models before and after implementing the abstraction technique. By examining the abstraction results at the suggested fitness criteria, an event log with 7,920 events and 14 Client-Side activities is obtained, maintaining a fitness accuracy of 0.81. The bold column in the Table 5.4 represents the abstraction threshold introduced in the methodology. Nevertheless, for demonstration purposes, Table 5.4 presents each abstraction iteration's results until no further activities can be compared. Thus, a suitable PCC threshold range of 85-1 is identified for achieving the desired level of abstraction. It is also observed that no further abstraction is possible below the fitness of 0.59, as no two events are similar beneath the PCC threshold of 28-1. As a result, the system achieves a 26.32% reduction in the number of activities and a 24.47% decrease in the total number of events for the SUS at the threshold specified in the methodology.

Furthermore, DT and RF are utilized as two supervised learning techniques to assess the accuracy of predicting Client-Side activity,  $a_c$ , based solely on Server-Side event logs,  $L_s$ . This approach uses the ELG framework discussed in Section 3.4.1 to explore user Client-Side activities by exclusively accessing Server-Side event logs. It is crucial to note that this study's choice of data mining algorithms (DT and RF) serves as an illustration and warrants further exploration.

To prepare the data for modeling, the following preprocessing steps are performed:



(a) Data Archiving Tool Spaghetti Model.



(b) Data Archiving Tool Abstracted Model.

Figure 5.7: Demonstration of discovered models from the abstracted event log.

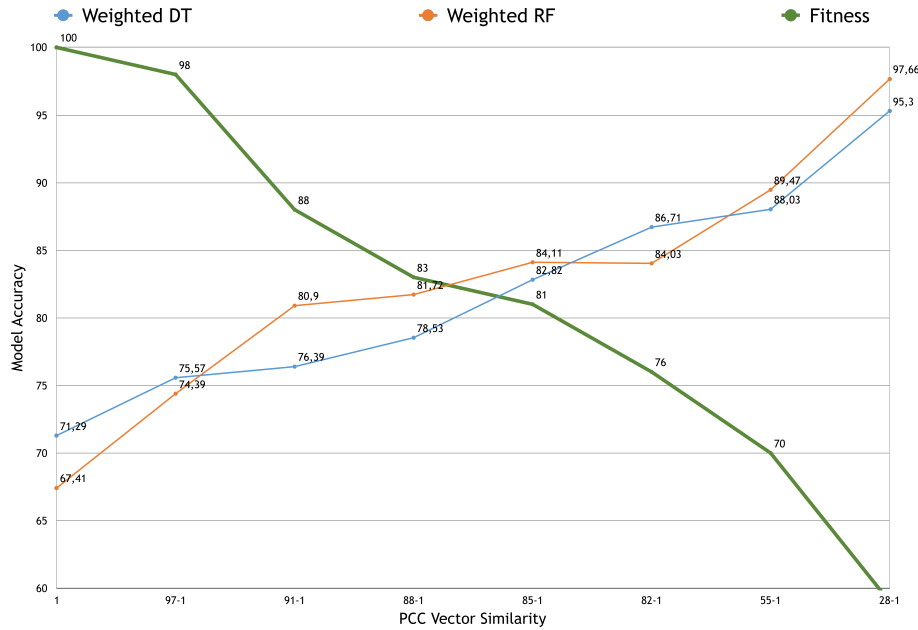


Figure 5.8: The negative correlation of the model *fitness* to DT and RF models accuracy on the Data Archiving Tool.

1. Missing Value Imputation: Filled in missing values in the dataset with zeros.
2. Resampling: A hybrid resampling technique is employed on the target label. The Synthetic Minority Oversampling Technique (SMOTE) technique for over-sampling and Tomek Links for undersampling are combined to balance the class distribution effectively.
3. Feature Scaling: The numerical features are standardized with a mean of 0 and a standard deviation of 1. This ensures that all features have equal importance and are on the same scale, particularly for distance-based algorithms and regularization techniques.

User activities can be efficiently analyzed and indicated by utilizing vectorized events where only Server-Side events are accessible and a complete training set is available. The DT and RF algorithms are applied to attain 82% and 84% accuracy, respectively. The inverse relationship between model fitness and model accuracy for activity prediction is further demonstrated in Figure 5.8, underlining the significance of maintaining a balance between abstraction and interpretability within the methodology.

### 5.2.2 Metadata Management Tool (MetadataTool)

The Metadata Manager, as its name suggests, is a web application tool designed for researchers at the university to create customizable metadata schemas, facilitating

the classification and exploration of research data based on their metadata. The most common user interactions include providing metadata schemas, searching metadata, and adding or editing metadata. This study collected event logs over six months, encompassing 573 case Ids and 13,794 events. The Metadata Manager generates 21 unique Client-Side activities and 32 Server-Side activities responsible for processing user requests.

Table 5.5: Hybrid logger performance assessment on Metadata Manager Tool using Decision Tree and Random Forest Classifiers.

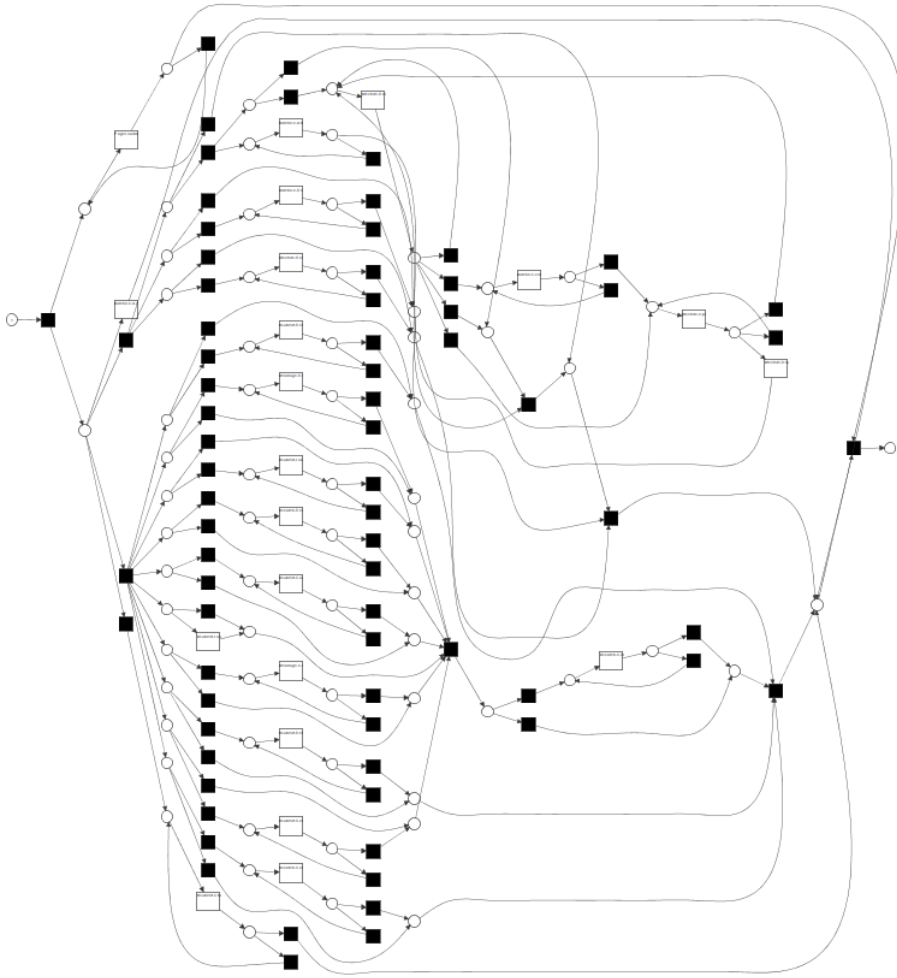
	RMSE	MAE	Precision	Recall	F1	Accuracy
DT	0.30	0.17	0.90	0.89	0.89	0.88
Random Forest	0.45	0.42	0.88	0.86	0.87	0.86

Table 5.3 provides performance statistics of the Hybrid logger discussed in Section 3.4 to acquire the dataset. Table 5.5 shows the outcomes of the evaluated performance indicators, revealing that the DT classifier outperforms the RF algorithm in all evaluative metrics to a minor yet noticeable degree. The DT classifier reports RMSE and MAE values of 0.30 and 0.17, respectively, denoting fewer prediction errors compared to the RMSE and MAE values of the RF algorithm, which are 0.45 and 0.42, respectively. In addition, the DT classifier yields Precision, Recall, and F1 scores of 0.90, 0.89, and 0.89, respectively, implying a more precise and equitable performance in recognizing true positives and negatives. In contrast, the RF algorithm produces marginally lesser values, with Precision, Recall, and F1 scores at 0.88, 0.86, and 0.87, respectively. Ultimately, the overall accuracy of the DT classifier is observed to be 0.88, a figure superior to the 0.86 accuracies presented by the RF algorithm. The Figure 5.9a depicts the Client-Side process model discovered from the Metadata Manager Tool with the help of the Hybrid logger applying the DT classifier.

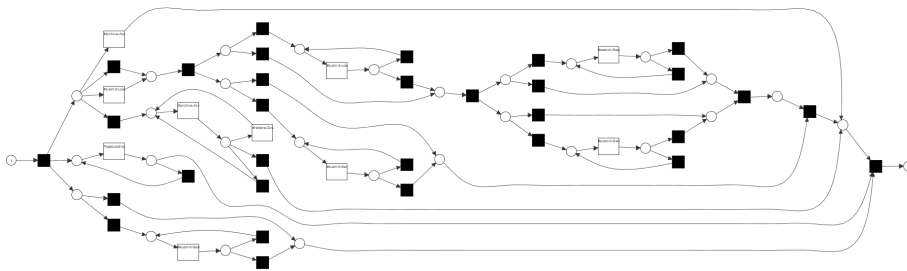
Table 5.6: Abstraction of sample dataset for Metadata Manager Tool, and its effect on the number of Client and Server-Side activities.

PCC Similarity Threshold	1	97-1	94-1	91-1	82-1	<b>79-1</b>	52-1	40-1	22-1
Fitness	1	0.91	0.89	0.88	0.88	<b>0.84</b>	0.79	0.77	0.73
# Client Activities ( $a_c$ )	21	16	15	14	11	<b>10</b>	8	7	6
# Events	13794	10302	9824	9268	7261	<b>6414</b>	5378	4735	4019

**Results:** By implementing the proposed approach, an abstracted process model is obtained that effectively illustrates the anticipated user interaction lifecycle. Figure 5.9a displays the intricate Petri net discovered before the event log abstraction, while Figure 5.9b presents the abstracted model obtained using the recommended method.



(a) Metadata Manager Tool Spaghetti Model.



(b) Metadata Manager Tool Abstracted Model.

Figure 5.9: Demonstration of discovered models from the abstracted event log.

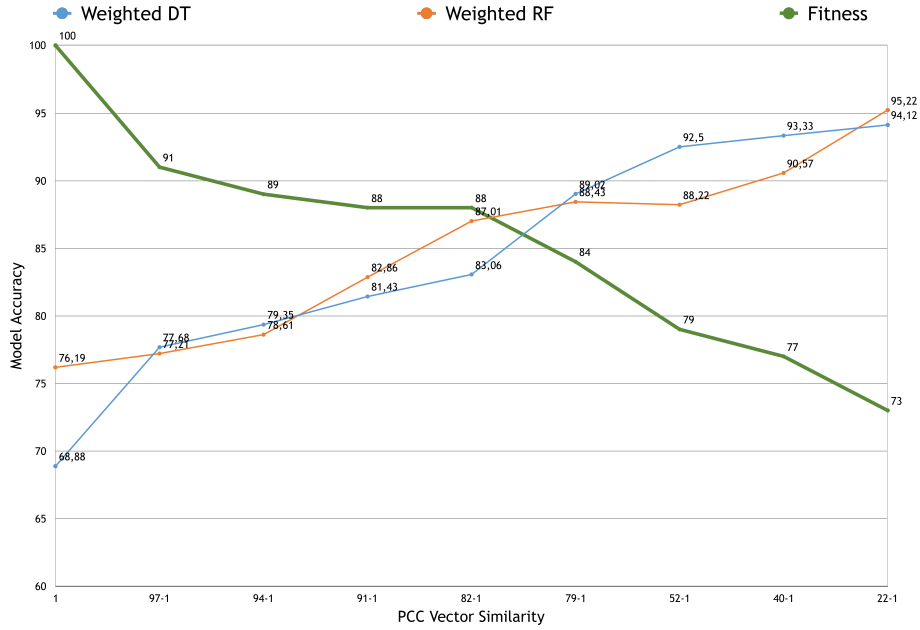


Figure 5.10: The correlation of the model *fitness* to DT and RF models accuracy on the Metadata tool.

Table 5.6 showcases the outcomes of the iterative process abstraction until no further abstraction can be achieved. The process is performed beyond the suggested fitness threshold as a proof of concept. Taking the recommended fitness criteria as the stopping point for the abstraction process, an event log is derived containing 10 activities, 6,414 events, and 573 cases, with an overall fitness accuracy of 0.84.

Moreover, by examining the results in Table 5.6, an appropriate PCC threshold for classifying similar events and merging events without jeopardizing fitness in the Metadata Manager tool is observed to be 79-1. Notably, no further abstraction is feasible below a fitness of 0.73, as no two events share similarities beneath a PCC threshold of 22-1. Consequently, a 52.39% reduction in the number of activities and a 53.51% decrease in the overall number of events for the system under investigation is attained.

This study used DT and RF as two supervised learning methods to evaluate the accuracy of predicting Client-Side activity,  $a_c$ , based solely on Server-Side event logs,  $L_s$ . This method employs the ELG procedure discussed in Section 3.4.1 to assess user behavior by only accessing Server-Side event logs. It is important to note that this study's choice of data mining algorithms (DT and RF) serves as a demonstration and requires further investigation.

The following preprocessing steps are performed to prepare the data for modeling:

1. Missing Value Imputation: Missing values in the dataset are filled in with zeros.

2. Resampling: A hybrid resampling technique is employed on the target label. The SMOTE technique for oversampling and Tomek Links for undersampling are combined with balancing the class distribution effectively.
3. Feature Scaling: The numerical features are standardized with a mean of 0 and a standard deviation of 1. This ensures that all features have equal importance and are on the same scale, particularly for distance-based algorithms and regularization techniques.

By using vectorized events in situations where only Server-Side events are accessible and a comprehensive training set is available, effective analysis and prediction of user behavior can be achieved. The DT and RF algorithms are applied, achieving 89% and 88% accuracy, respectively. The inverse relationship between model fitness and model accuracy for activity prediction is further illustrated in Figure 5.10, emphasizing the importance of balancing abstraction and interpretability in the context of the approach. Moreover, the most suitable PCC threshold for detecting similar events and merging them without reducing the fitness score on the Metadata Manager web application is determined to be 79-1.

### 5.2.3 Summary

The analysis reveals an inverse correlation between the model fitness and the accuracy of the DT and RF classifier for predicting Client-Side activities based on Server-Side traces of software execution on the Data Archiving and Metadata Management Tools. This relationship indicates that as the model fitness decreases, the accuracy of the DT and RF classifiers are also affected. The choice of a 0.8 threshold to conclude the algorithm for further abstraction is based on the observation that beyond this point, the abstraction becomes too generalized, causing the findings to lose their interpretability and effectiveness. Consequently, maintaining a balance between abstraction and meaningful insights is crucial for the approach's effectiveness.

The DFG model is employed to represent the abstracted models, as it facilitates comprehension for domain experts. Obtained qualitative feedback, such as: “[...] *I can now understand user interactions and assess user behaviors without being overwhelmed by numerous activities that do not represent the actual processes. [...] I’m wondering why we have such poor performance in restoring files; I need to investigate this.*” referring to a previously unknown KPI violation for restoring files. Therefore, the proposed method successfully identified non-functional requirements and directed developers’ attention to the appropriate software components for further technical investigations.

Moreover, the Hybrid logger was utilized to gather the dataset for the task of event log abstraction. The Hybrid logger approach combines the advantages of Centralized and

Client-Side logging techniques to provide a comprehensive view of Client-Side user activities and corresponding back-end software executions. By integrating these two perspectives, the Hybrid logger enables more accurate predictions of user behavior, allowing for a deeper understanding of how users interact with the system. Additionally, the findings confirms a fitness threshold (0.8) for the abstraction algorithm, which aligns with high accuracy in predicting user activities at the proposed threshold. The right balance between abstraction and preserving the necessary information for accurate predictions is the key. According to the findings, predictions' accuracy is attributed to the granularity of data captured and each iteration of data abstraction, making it easier for the prediction model to capture the underlying patterns and relationships. As a result, this method facilitates the development of more effective machine learning models and prediction algorithms, ultimately leading to improved system performance, better user experience, and more informed decision-making.

Despite the efforts to develop a generalizable event log abstraction technique, the proposed method only applies to Client-Server applications. One of the most labor-intensive and manual aspects of this technique, acknowledged here, is evaluating the input log for noise, outliers, and anomalies. Regrettably, the case studies' report does not examine Precision and Generalization as additional quality indicators due to the considerable computational power required to calculate these factors via ProM implementation. As a result, these calculations were never completed. Activity name concatenation is employed for the relabeling task; however, this may create readability issues for some domain experts if the number of original activities increases significantly. Lastly, the method relies on the ELG technique that relies on the OAuth workflow (authorization service), which may overlook essential Server-Side activities not recorded by the OAuth service which is also a disadvantage of the Centralized logger. Additional research is needed to validate the approach while incorporating all executing software components.



## 5.3 Knowledge Discovery

The Knowledge Discovery case studies aim to explore various aspects of researchers' interactions with Open Science processes. Through a series of targeted case studies, insights were gained into researchers' process models, conformance to established standards was assessed, and predictions were made regarding user organizational roles based on their activities. The RDM lifecycle for research projects was also uncovered. These case studies provided valuable insights into the dynamics of Open Science processes and shed light on potential areas for improvement, optimization, and enhancement of the overall research experience.

### 5.3.1 Descriptive Research Processes

This case study aims to discover the descriptive research processes within an RDM platform called Coscine. Coscine allows users to set up multiple Resources (data collections or repositories) per project, with each Resource acting as a file repository to store data along with specialized metadata. The DA4RDM web-based tool enables users without technical knowledge to interact quickly with datasets and assess workflows.

The sample dataset used in this study was acquired from the Coscine platform from April 2021 to June 2022. Figure 5.11 presents a screenshot from the DA4RDM web page, showcasing the interface for discovering processes based on selected criteria. Upon analysis, Figure 5.12a reveals a collective process models view, uncovering a Spaghetti model for user interactions with all Resources within Coscine. From this dataset, 551 Resources, 16 recorded activities, and 32 variations of user interaction are discovered, resulting in 66,028 related events.

DA4RDM allows for the selection of individual Resources for further investigation and study. For example, Figure 5.12b illustrates the DFG process model of a single Resource, eventually uncovering the complete lifecycle of the Resource. 144 events and 10 activities are observed for the selected case and variance. The process model begins with *Add Resource* and concludes with *Delete Resource*. Upon further investigation, it was determined that a research group utilized this Resource to assess the functionality and reliability of Coscine as their data repository.

Figure 5.12c presents the process model for user interaction on a single data artifact within the selected Resource. This process model, which contains 13 events and 4 activities, enables us to study user behavior. In this example, a user uploaded a data artifact with metadata, and other users later discovered the file entry, assessed its metadata, and in 5 cases, attempted to update pre-existing meta information.

Based on the findings and the discussed definition of RDM phases, the user activities discovered on the selected file can be correlated to the *Analysis* phase. In this phase, research data requires further curation with each research iteration. This case study demonstrates the potential of the DA4RDM tool to discover and analyze research processes within the Coscine RDM platform, providing valuable insights into user behavior and data management practices.

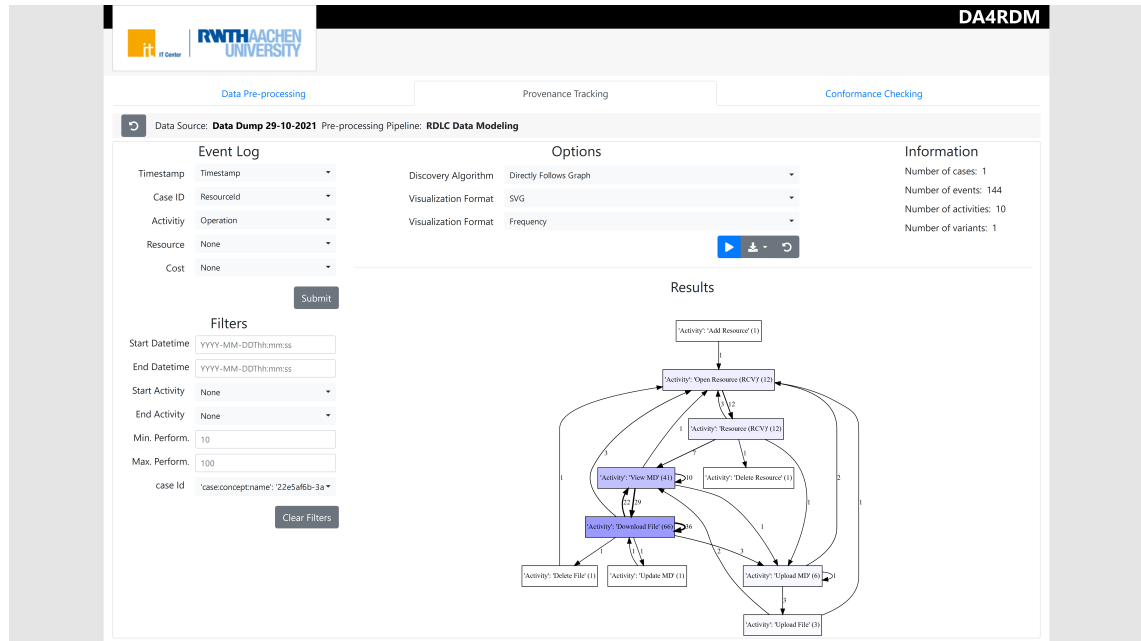


Figure 5.11: Screenshot of DA4RDM for discovering user interaction model over a selected Resource on Coscine.

The ability to discover and analyze research processes within RDM platforms like Coscine offers numerous potential applications and use cases, benefiting various stakeholders in the research ecosystem. Some of these applications and use cases include:

*Research Institutions and Universities:* Academic institutions can utilize DA4RDM to understand better how their researchers interact with data, enabling them to optimize data management practices, facilitate collaboration, and improve research efficiency. In addition, by analyzing these interactions, institutions can identify areas of improvement and implement targeted training programs to enhance researchers' data management skills.

*Data Managers and Librarians:* Data managers and librarians can use the insights generated by DA4RDM to develop and implement data management policies

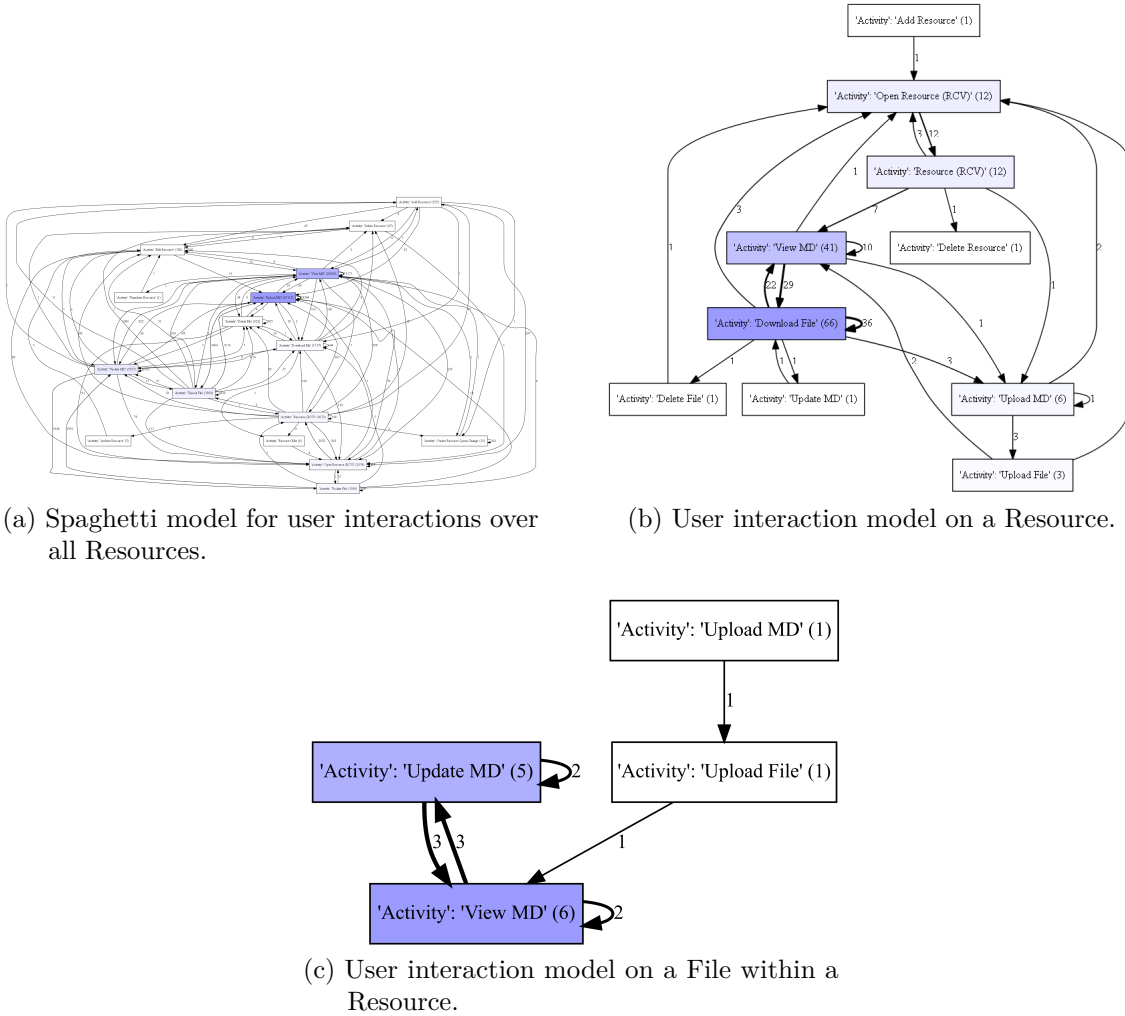


Figure 5.12: Hierarchical descriptive research process models.

and best practices tailored to the unique needs of their organization. They can also use this information to create targeted data curation strategies and metadata guidelines, ensuring consistency and discoverability across various research projects.

*Research Funding Agencies:* Funding agencies can employ DA4RDM to evaluate the data management practices of grant applicants and awardees. By understanding how researchers interact with data, these agencies can ensure that funded projects adhere to data management standards, promote transparency, and facilitate data sharing and reusability.

*Research Collaborations and Consortia:* In large, multi-institutional research projects, the DA4RDM tool can assess and harmonize data management processes across different research groups. The tool can help develop shared data management strategies and foster collaboration among researchers by identifying commonalities and discrepancies in data workflows.

*Commercial RDM Platform Providers:* Providers of RDM platforms like Coscine can use the insights from DA4RDM to enhance their platform's usability and features. By understanding user interactions and needs, they can develop more user-friendly interfaces, improve metadata management capabilities, and integrate additional functionalities to support different research workflows.

*Researchers:* Individual researchers can benefit by gaining a deeper understanding of their data management practices and identifying areas for improvement. This knowledge can help them optimize their workflows, save time, and enhance the quality and reproducibility of their research.

*Hierarchical Analysis of Data:* DA4RDM can be used to assess collective users' behavior at different levels of granularity, allowing stakeholders to narrow down their analysis. For example, a university may analyze data interactions at the department, research group, or individual researcher level, leading to tailored recommendations for each group to improve data management practices.

*User Experience (UX) Evaluation and Finding Bottlenecks:* By studying user interactions with RDM platforms, DA4RDM can help identify bottlenecks and areas where users struggle, informing UX improvements. For instance, the tool might reveal that users find it challenging to locate specific metadata fields or navigate between Resources, prompting platform developers to streamline these processes.

*Supporting Data Provenance:* DA4RDM can contribute to data provenance efforts by collecting and modeling information on *who did what, when an action was executed, what has happened*, and the order of events. This information can help researchers and data managers trace the origin and lineage of data, ensuring its accuracy, validity, and reliability.

*Studying User Journey Mapping:* The insights provided by DA4RDM can enable user journey mapping, allowing stakeholders to visualize how users interact with RDM platforms throughout their research processes. Platform developers can create more intuitive and user-friendly interfaces by understanding the sequence of user actions and pain points.

*Service Level KPI Analysis:* DA4RDM can assess service level KPIs for RDM platforms, such as user satisfaction, data management efficiency, and collaboration rates. Organizations can monitor their RDM platforms' performance by analyzing these KPIs, identifying improvement areas, and making data-driven decisions to optimize their research data management strategies.

In summary, the DA4RDM tool holds significant potential for various stakeholders in the research ecosystem. Providing insights into user interactions and data management practices can lead to more efficient, collaborative, and transparent research processes, ultimately driving scientific innovation and discovery.

### 5.3.2 Control-Flow Requirement Analysis

Another case study aims to discover non-conforming process variances (sequence of actions) to find requirements within an RDM platform called Coscine using the DA4RDM tool. Coscine allows users to set up multiple resources (data collections) per project, with each Resource acting as a file repository to store data along with specialized metadata.

The DA4RDM web-based tool is utilized to run conformance checking for two use cases to assess the applicability of DA4RDM in this context. Figure 5.13 demonstrates a screenshot from the DA4RDM for discovering non-functional requirements.

For this case study, system process domain knowledge is leveraged. With respect to Coscine development, the user's attempts to create Resources should be followed by being redirected to that Resource view. Accordingly, the first sequence is defined as *View Project* and *Add Resource* and the follow-up expected sequence as *Open Resource (RCV)*.

Additionally, according to our service level agreement, creating resources on Coscine should not take longer than 30 seconds. Therefore, 30 seconds is defined as the **Performance** criteria. The conformance checking for the given conditions runs over 297 cases, of which 65 cases do not conform to the specified expected user path. For example, in Figure 5.13, it has violated the time span expected to create a Resource despite observing the correct expected activity sequence.

When precisely this has happened in the case information field, for which **User**, **Role**, **Project**, and **Resource** can be found. Then, further evaluation can be conducted to determine and uncover the source of the problem.

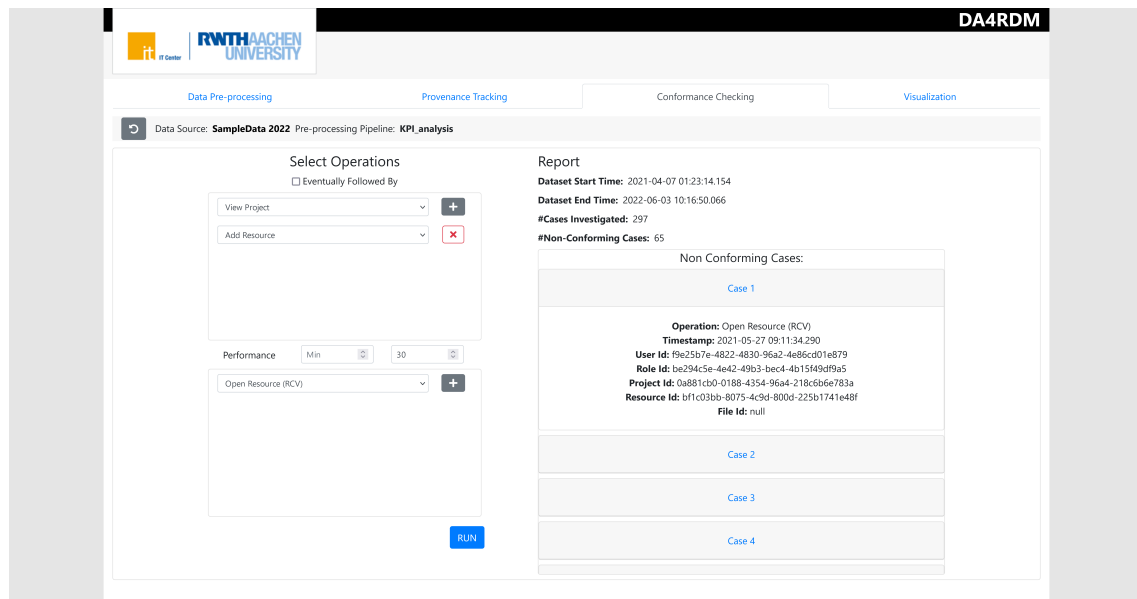


Figure 5.13: Screenshot of DA4RDM for identifying requirements on Coscine.

Continuing with the second use case, Figure 5.14 demonstrates the assessment of FAIRness on Coscine. For instance, as part of the interoperability of the FAIR principle, all data artifacts should be accompanied by specialized metadata. Accordingly, Coscine provides an environment that supports the FAIR principle by design, requiring users to upload files with metadata.

According to this promise, operations *Open Project*, *Open Resource (RCV)*, and *Upload File* are expected to be seen as the first sequence of action followed by *Upload MD*. Upon analyzing the conformance, out of 3919 cases, 144 non-conforming cases have been discovered. The **File Id**, provided in the example, enables system administrators to investigate the issue further.

After the discovery of the system issue, the development team changed the software execution process such that no files could be uploaded without ensuring that metadata was stored in advance. This modification helps ensure that Coscine complies with the FAIR principles, fostering a more interoperable and reusable data ecosystem.

The DA4RDM tool offers various potential applications for stakeholders working with RDM platforms, such as:

*Enhancing User Experience:* By identifying non-conforming process variances, RDM platform developers can pinpoint issues in the user experience, leading to targeted improvements in the platform's interface and functionality.

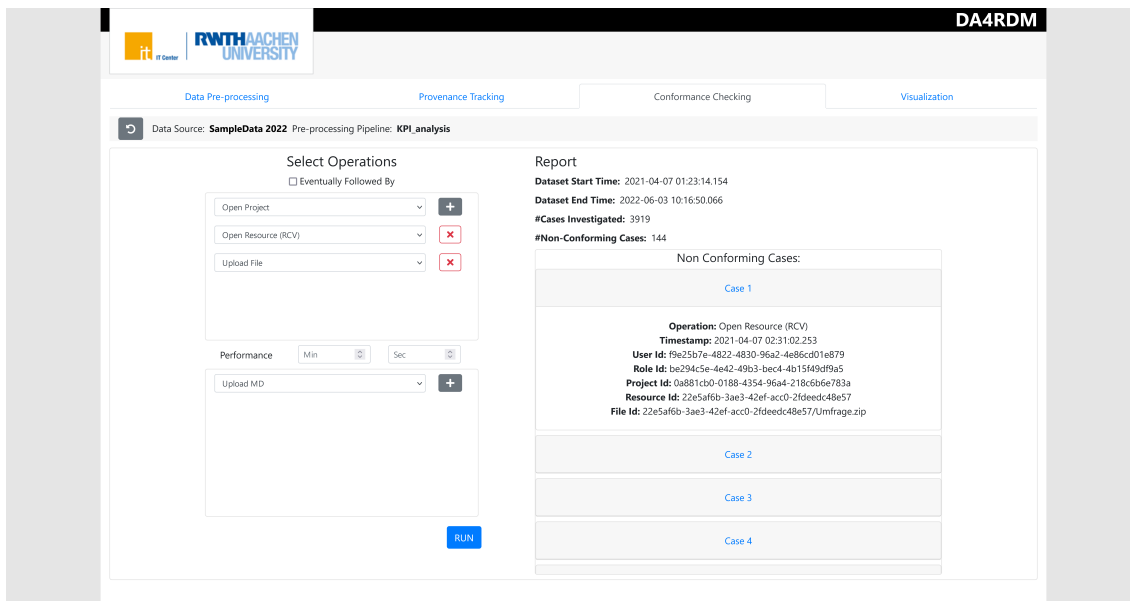


Figure 5.14: Interoperability assessment of Coscine as an RDM system.

*Ensuring Compliance with Data Standards:* By identifying non-conforming cases that violate established data standards, organizations can take corrective actions to ensure that their platforms and data repositories remain compliant with relevant guidelines and best practices.

*Identifying Areas for System Improvement:* The insights uncovered by the DA4RDM tool can help development teams identify areas where the system's design or execution may be suboptimal, leading to targeted improvements that enhance the overall platform's performance and functionality.

*Monitoring Adherence to FAIR Principles:* By analyzing the conformance of user behavior with the FAIR principles, stakeholders can assess the effectiveness of their data management practices and make necessary adjustments to promote the FAIRness of their data resources.

*Detecting Anomalies:* The DA4RDM tool can help identify anomalous user behavior, such as unauthorized access to resources, which may indicate potential security vulnerabilities or misuse of the platform.

*Optimizing Platform Performance:* By analyzing non-conforming process variances, stakeholders can identify performance bottlenecks and optimize the platform's infrastructure and architecture to ensure timely and efficient resource creation and management.

*Training and Support:* The insights generated by conformance checking can help institutions develop targeted training programs and support resources to address common user errors and improve data management practices.

*Quality Assurance:* Regularly monitoring and analyzing non-conforming process variances can be a quality assurance measure, ensuring that the RDM platform adheres to established performance standards and user expectations.

The findings yield promising results for enabling the discovery of data-driven requirements. As such, these non-conforming cases are typically not reported by the user otherwise. This approach empowers the monitoring of processes and workflows within an RDM platform. Future work can include periodic execution of pre-defined conformance analysis and automatic countermeasures or automatic reporting before user service-desk reports.

In summary, the DA4RDM tool can be instrumental in uncovering non-conforming process variances within RDM platforms, enabling stakeholders to monitor compliance with data standards, optimize platform performance, and ensure adherence to the FAIR principles.

### 5.3.3 University Organizational Mining

To enhance the reliability and scalability of the approach, organizational elements (user roles) have been incorporated into the sample dataset. The goal of organizational mining is *to determine and forecast users' roles based on their executed activities using data mining techniques*. This analysis aims to examine the feasibility and precision of data mining for extracting users' roles within university services by assessing the dataset obtained through the method. At present, the system tracks three role types, specifically *student*, *employee*, and *student-employee*. The *student-employee* role signifies individuals who hold both *student* and *employee* affiliations.

Table 5.7: Descriptive matrix produced by augmenting and modifying the sample dataset derived from the Centralized logger.

UserId	Role	GetInfo	GetNotifications	GetWeeks	GetReport	IsAuthorized	GetAllFiles	GetSchema	...
dhcwb4MIL...	student	1	1	0	0	0	0	0	...
LOrv8dNur...	student	0	0	1	0	0	0	0	...
nfW6O7i85...	student-employee	0	0	0	1	1	1	1	...
6XE8m5KaE...	student	0	1	0	1	0	0	0	...
2E1c56wFG...	employee	0	0	0	0	1	0	0	...
...	...	...	...	...	...	...	...	...	...
cE5G9LoF5...	student-employee	2	0	0	1	3	1	0	...
...	...	...	...	...	...	...	...	...	...

**Data Selection:** As mentioned in Section 3.3.1, the approach can be extended to incorporate additional attributes, such as user roles. For example, the sample



data displayed in Table 5.1 encompasses user interactions and services utilizing the token service. Through heuristic observation, it is hypothesized that there is a direct connection between a user's role and the set of activities they perform.

**Data Preprocessing:** The selected data posed a challenge due to the uneven distribution of groups, which could result in biased machine-learning outcomes. The sample dataset consists of approximately 82.3% entries associated with the *student* role, about 16.1% linked to *employees*, and 1.6% connected to *student-employees*. However, this distribution mirrors the actual user group distribution at the University. Therefore, it is crucial to apply balancing techniques to address the imbalanced dataset issue. For example, to explore minority and less frequent classes, one could employ the SMOTE method [229] to over-balance a dataset. Conversely, under-balancing can maintain an even class distribution by reducing the frequency of majority class occurrences.

**Data Conversion:** As depicted in Table 5.7, the data is transformed into a vectorized descriptive matrix format, where each row represents a unique user and each column corresponds to an executed software component. In addition, this matrix format comprises aggregated data, where repeated interactions merely increase the counter for an activity associated with a specific user, thus adding weight to the descriptive vector.

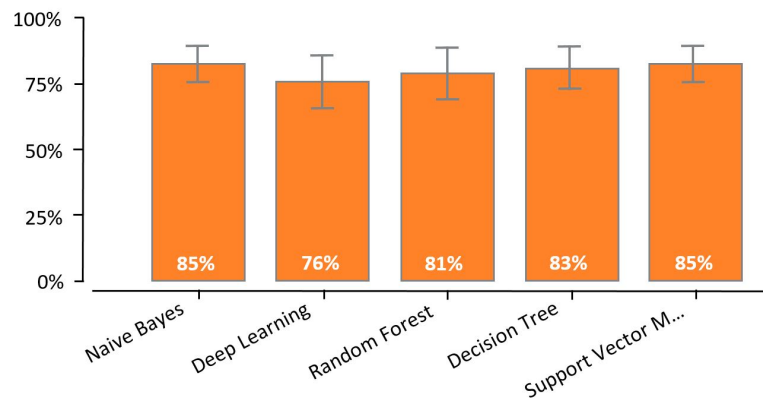


Figure 5.15: Classifiers model accuracy for role mining with the help of RapidMiner studio.

**Data Mining:** To determine the best-performing classification algorithm for the objectives, various machine learning techniques are applied. A 3-fold cross-validation methodology [230] is employed for the analysis. The sample dataset was divided into two sets: a training set consisting of 2/3 of the data and a validation set comprising 1/3 of the data. The analysis was carried out using RapidMiner Studio [231].

**Analysis Result:** The data is analyzed using various algorithms, including Random Forest, Support Vector Machine, Decision Tree, Deep Learning, and Naive Bayes. The

obtained accuracy reflects the presence of noise and the need for further data preprocessing. However, as depicted in Figure 5.15, the role-mining results show an average model accuracy of 85% for Naive Bayes and Support Vector Machine algorithms after oversampling using the SMOTE technique. Furthermore, the overlapping error bars suggest that the datasets do not differ significantly. These findings validate the data collection approach employed in the method and suggest that more satisfactory results can be achieved by conducting additional outlier and noise analysis.

### 5.3.4 CRC1394 RDM Lifecycle

This case study investigates the event data extracted from the Coscine platform to outline RDM phases for research projects using radar visualization and process mining techniques. The analyzed sample dataset spans from April 2021 to May 2023, encompassing 1,119 projects and 32 unique activities recorded. However, since Coscine has been still in its beta stage at the time of recording this study, most projects and user interactions may not accurately reflect real user interaction. This is primarily due to the experimental nature of the platform during its beta stage, leading to incomplete or inconsistent data that may not represent actual research data management practices.

Nonetheless, a group of researchers, Collaborative Research Centre (CRC)1394, has been using Coscine for RDM practices, and the case study is focused on this group. Therefore, two years' worth of data from May 2021 to May 2023 for CRC1394, which consists of 45 members and 28 sub-projects, is analyzed. In particular, three sub-projects, namely A03, B06, and C02, each with six scholars maintaining RDM within their sub research groups, are concentrated on.

To observe changes over time, the dataset is divided into four equal periods, each for six months. Also, process models for CRC1394 for each six-month user interaction period are discovered for demonstration purposes. Figure 5.16 illustrates the discovered process models for CRC1394, while Figure 5.17 displays the discovered RDM phases for all named projects within the specified timeframes.

As shown in the Figure 5.17, in the time interval between May and November 2021, an apparent inclination toward the *Planning* phase at the beginning of the projects can be seen. Sub-group B06 quickly gains momentum and starts the *Production* phase within the first six months of their research project. Between December 2021 and May 2022, while CRC1394 is still mainly in the *Planning* phase, the C02 group quickly transitions to the *Production* and *Analysis* of research data. Interestingly, group B06 analyzes previously documented records, while group A03 appears to be just starting with their research *Planning*.

The final year between June 2022 and May 2023 shows similar trends. The central group (CRC1394) continues leaning toward *Production* and updating their files.

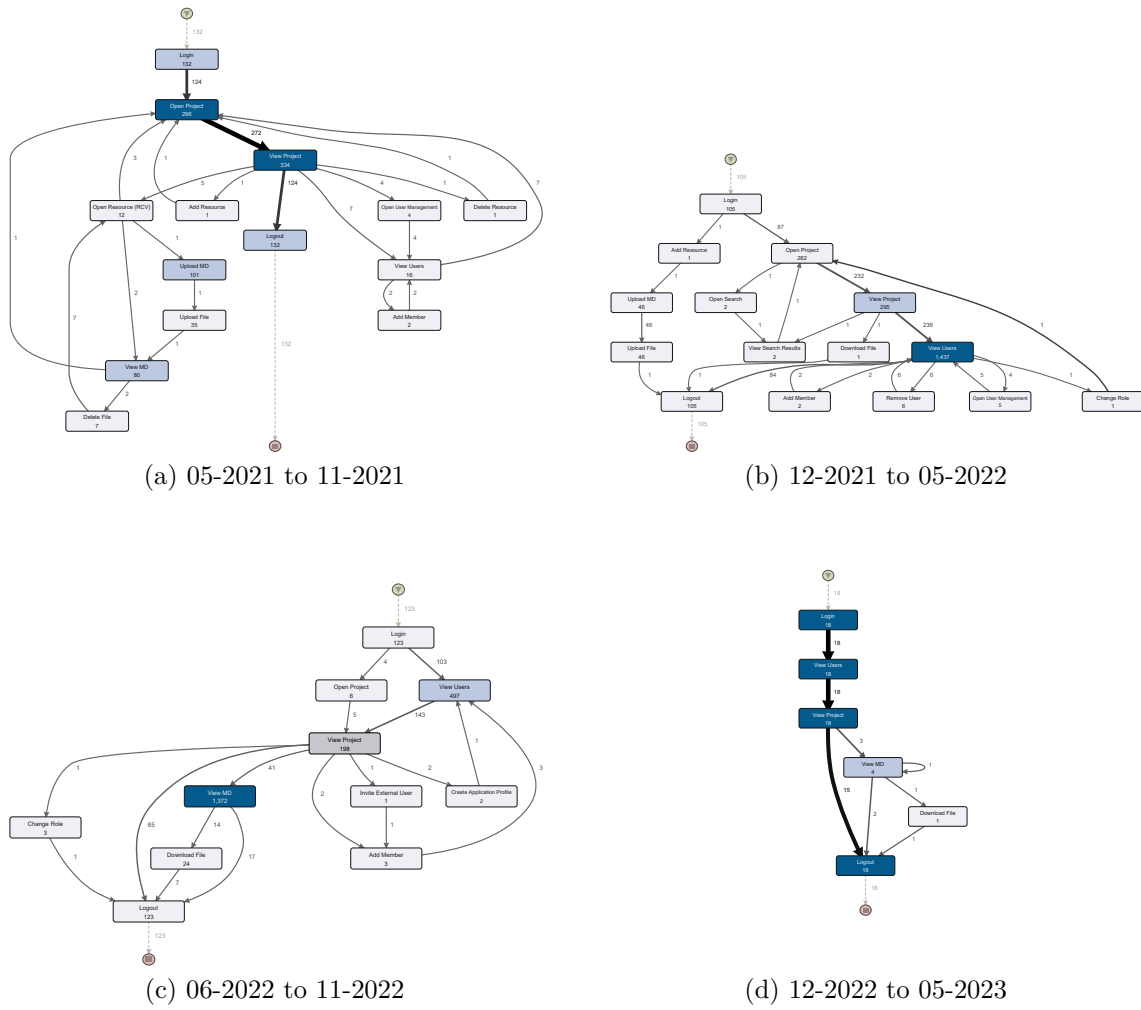


Figure 5.16: The evolution of user interaction patterns for CRC1394 in two years of activity.

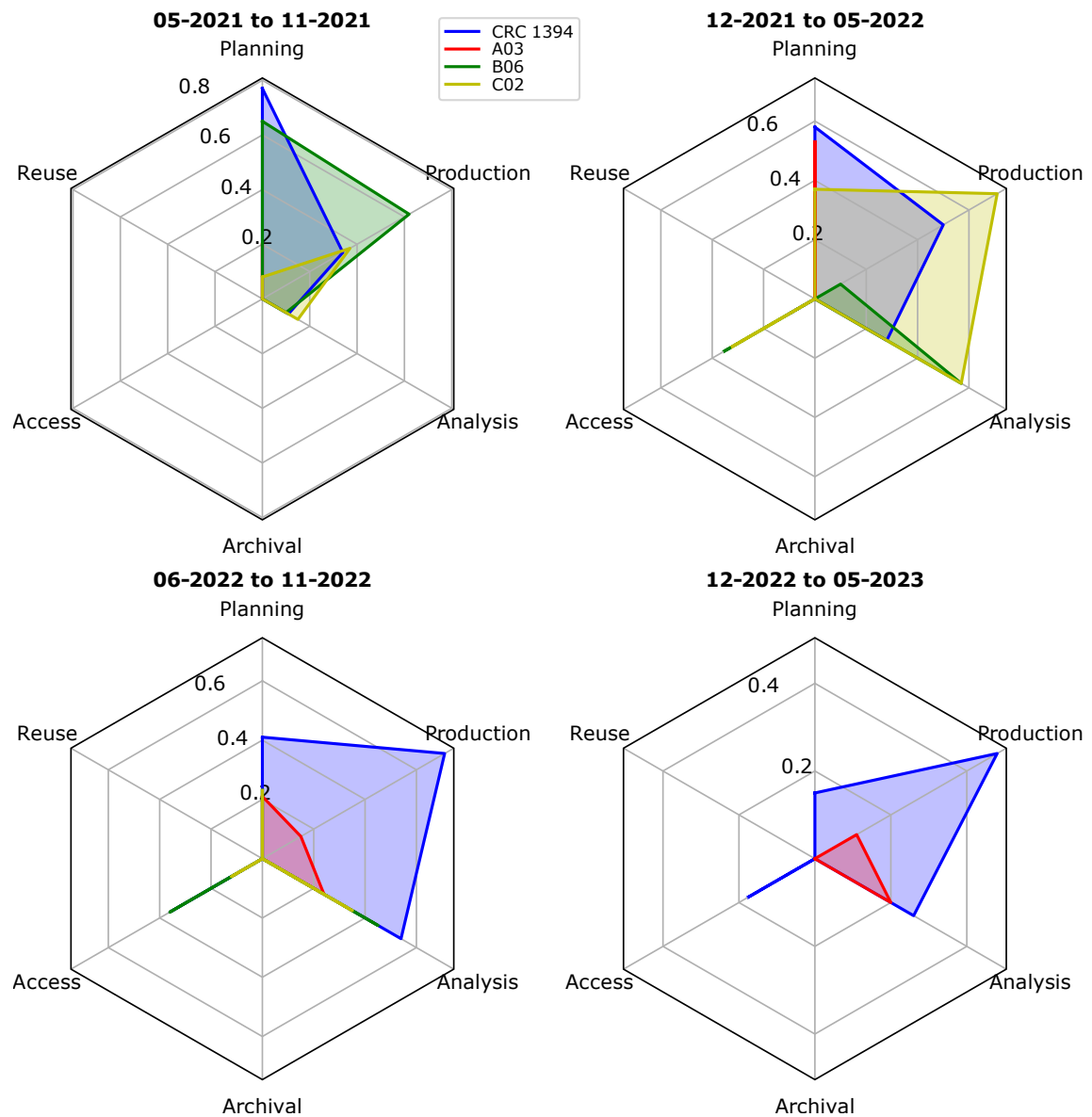


Figure 5.17: Radar visualization of RDM lifecycle for CRC1394 and three sub-groups in two years of activity.

However, the group A03 appears less actively involved in RDM and only provides a limited quantity of research data on the RDM platform under study. Conversely, the B06 group seems to be engaged in locating and analyzing their research data. In the final stage of the record, not much activity in these research groups during the past six months is seen. This finding has prompted the RDM consulting team to reach out and investigate the reasons behind the inactivity and decreased participation of researchers in RDM practices. Since the CRC1394 project is still in progress, no evidence of research data *Archival* or *Reusing* data between groups, as correctly represented in Figure 5.17, is found.

The case study presented, demonstrates the potential of the proposed methodology in outlining RDM phases and determining the current RDM phase of research projects. In addition, the discovered process models and the radar visualization of RDM phases provide valuable insights into the progression and activities of different research groups.

The granularity of the collected information plays a crucial role in the analysis. If the data is too fine-grained, event data abstraction may be necessary to avoid overwhelming complexity. Furthermore, the discovery of RDM phases is subjective to the predefined process models for each RDM phase, which may need alterations to accommodate the spectrum of activities that an RDM platform could track or record. Domain knowledge is also crucial for creating the prescribed process models.

In the case of the CRC1394 group, it is observed that some sub-projects had over 20 users. However, after investigating the members of CRC1394 and the metadata provided for sub-projects, it is concluded that these sub-groups were created for coordination purposes and to maintain a central data center for all sub-groups rather than using Coscine for RDM or focusing on specific research areas. To name a few examples of such sub-groups: General Assemblies, Workspaces, Schools, and Tutorials. Since Coscine does not have an inheritance rule for sub-projects, members, files, and access privileges are not propagated across groups. Consequently, the discovered RDM visualization for CRC1394 does not include user activities within its sub-groups. By extending the event logger, the association of groups/projects can be recorded to discover the overall RDM lifecycle without needing overlays for individual sub-groups in the visualization.

Determining a minimum dataset or criteria for the validity of the methodology to outline the RDM lifecycle for research projects is challenging, as it strongly depends on scholars' culture of conducting research and their commitment to maintaining RDM practices. Nevertheless, the visualization provided by the methodology serves as a tool to help researchers, team leaders, and principal investigators assess the current phase of a research team and plan countermeasures if a research group is falling behind its schedule or requires assistance in a specific RDM phase without pinpointing individual researcher and maintaining user privacy.

To facilitate the application of the methodology, a Python package named **DA4RDM-Vis-ProcessBased** is published, which can be utilized to discover the RDM lifecycle for a given project. This package offers researchers and data managers an accessible way to analyze, monitor, and improve research data management practices.

Several research directions can be explored for future work, such as predictive, clustering, role-based, compliance, and user behavior analysis. Predictive analysis can be employed to train machine learning models for predicting the completion time of different RDM stages based on historical data, enabling researchers to plan their projects better and allocate resources effectively. Clustering analysis can group research projects based on their RDM characteristics to identify common patterns, challenges, or best practices. Role-based analysis can investigate the activity patterns of different roles within the dataset to better tailor the RDM platform to suit user needs. Compliance analysis can be used to check if research projects comply with the RDM process models and identify areas where deviations frequently occur. Lastly, user behavior analysis can be performed to determine if there are specific activities or stages where users tend to struggle or need more assistance, thereby improving the user experience and providing targeted support.

According to the findings, the proposed methodology offers a valuable tool for researchers, data managers, and platform administrators to analyze, monitor, and improve research data management practices. Furthermore, continuously refining the methodology and incorporating domain knowledge can contribute significantly to understanding and optimizing the RDM process within the research community.

### 5.3.5 Summary

In conclusion, these studies have demonstrated the potential of the DA4RDM tool as an effective means to analyze, monitor, and improve research data management practices. By exploring various research directions, such as predictive analysis, role-based analysis, compliance analysis, and user behavior analysis, there is significant potential for further advanced applications in future work. Furthermore, the approach can significantly contribute to understanding and optimizing the RDM process within the research community by incorporating domain knowledge and continuously refining the methodology.

While the DA4RDM tool provides valuable insights into user behavior and data management practices, it is crucial to acknowledge its limitations. The evaluation of user behavior in an RDM platform is inherently subjective, depending on the data collection method, operations tracked, and their level of granularity. This subjectivity is highly influenced by the initial project idea and the research questions driving the study.

For example, the data collected may not capture every user interaction or specific data management practices. Additionally, the level of granularity in the data may not be sufficient to provide detailed insights into certain aspects of user behavior. Therefore, the data granularity has to be adapted to the SUS. This limitation can impact the conclusions drawn from the analysis and the subsequent recommendations for improving data management practices and platform usability.

To address these limitations, researchers should consider the research questions and objectives driving their study when designing data collection strategies. By ensuring that the data collected is comprehensive and captures the desired level of granularity, researchers can enhance the reliability and validity of their findings, leading to more accurate conclusions and actionable recommendations.

In summary, while the DA4RDM tool offers a powerful means to discover and analyze research processes within RDM platforms, it is crucial to recognize and account for its limitations. By doing so, stakeholders can obtain more accurate insights into user behavior and data management practices, ultimately driving improvements in research efficiency, collaboration, and transparency.

## 5.4 Research Data Recommender System

This Section focuses on facilitating data reusability by employing two-pronged approaches: CB and IICF recommender systems. The primary objective of this case study is to investigate and demonstrate how these two techniques can be effectively applied to enhance the discoverability and reusability of research data, thereby fostering a more efficient and collaborative research environment. The exploration begins with the CB recommender system, highlighting its underlying principles, advantages, and potential drawbacks. Subsequently, the findings derived from implementing IICF are discussed, discussing its synergistic potential when combined with CB techniques. These case studies aims to provide valuable insights for researchers and institutions seeking to optimize data reusability and facilitate collaboration within the scientific community.

### 5.4.1 Content-Based Recommender for CRC1394

In this study, metadata from files uploaded and provided by experts in Material Science and Chemistry (CRC1394 <sup>1</sup>) over one year were obtained. Manually entering metadata for every file is a challenging task for researchers and is often neglected or postponed, making these relatively small datasets valuable and expensive to acquire [232]. Figure 5.18 presents the distribution of files across their respective data repositories (data collections), with some containing about 55 files and others having as few as 1 or 2 files. In total, 543 entries from 25 repositories were acquired.

Figure 5.19 depicts the data sparsity for each *Predicate* or column. Although the RDM system under study does not permit file uploads without providing mandatory meta information, employing DA4RDM for analysis of the user interaction models discussed in Section 4.2 reveals that some users circumvent this requirement by uploading their research data to their data repositories using File Transfer Protocol (FTP) Clients [167]. Thus, addressing data imputation for missing values is critical in the recommender system preprocessing pipeline. The proposed recommender system suggests relevant repositories by building profiles from files' metadata. Additionally, the recommender system suggests repositories' PIDs as links, which users can be used to contact the data owner and request access to data while complying with data protection laws. The proposed methodology is implemented using the `scikit-learn`<sup>2</sup> machine learning library for the Python programming language.

This Section discusses the empirical studies using the proposed approach to offer repository recommendations and evaluate its performance. The dataset employed for

---

<sup>1</sup><https://www.sfb1394.rwth-aachen.de/>

<sup>2</sup>[www.scikit-learn.org](http://www.scikit-learn.org)



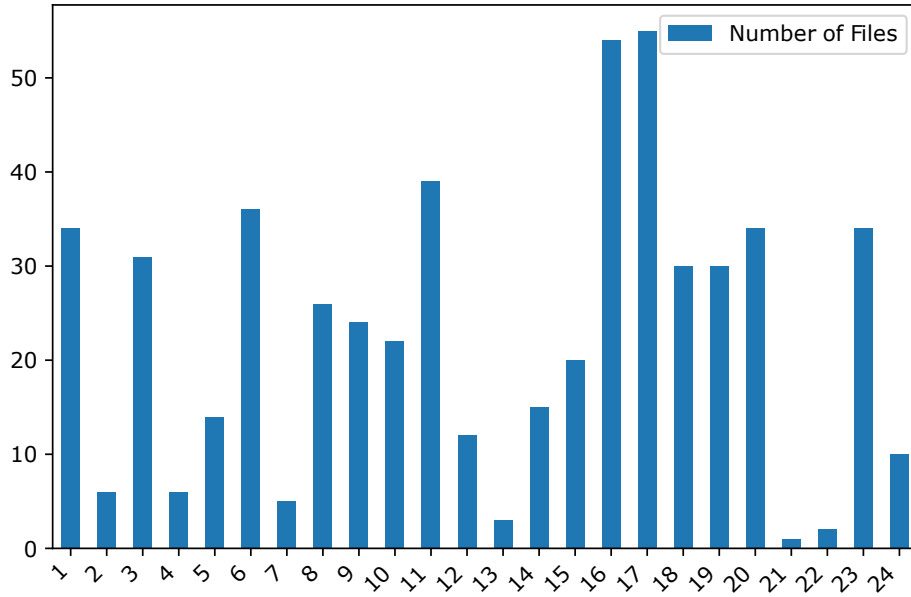


Figure 5.18: Files distribution across repositories. The x-axis denotes the Repository Id, while the y-axis signifies the file count.

this evaluation is detailed in Section 4.4.1, and an example dataset is listed in Table 4.3, which is sourced from the Coscine platform. A domain expert was consulted to identify an appropriate set of twelve features (*Predicates*) that accurately represent artifacts within repositories. The columns in Figure 5.19 (excluding File column) display the respective independent variables. According to the expert, the meta-data for the chosen set of features can help domain scientists differentiate between repositories without necessitating the individual examination of data artifacts.

The evaluation process was carried out twice, once for each distance metric, to ensure the reliability and consistency of the results. Heatmaps depicting the pairwise similarity of repositories are shown in Figure 5.20. To evaluate the efficacy of the pipeline, a ground truth matrix was created by asking the domain expert to subjectively evaluate and rank the similarity of pairwise repositories, providing a similarity score between 1 and 5 based on the classification scheme discussed earlier. This evaluation approach has been widely adopted in previous studies to assess the performance of distance metrics [233].

Table 5.8 displays each class's Precision, Recall, and F1 score separately for Cosine and Euclidean distance metrics. The F1 score performs best for class 1 regardless of the distance metric. The imbalanced distribution of the dataset across classes, indicated by the support values, may explain this observation.

Table 5.9 presents an overview of the overall performance of the recommender system. The Silhouette score is used to evaluate the quality of the predicted ratings, which

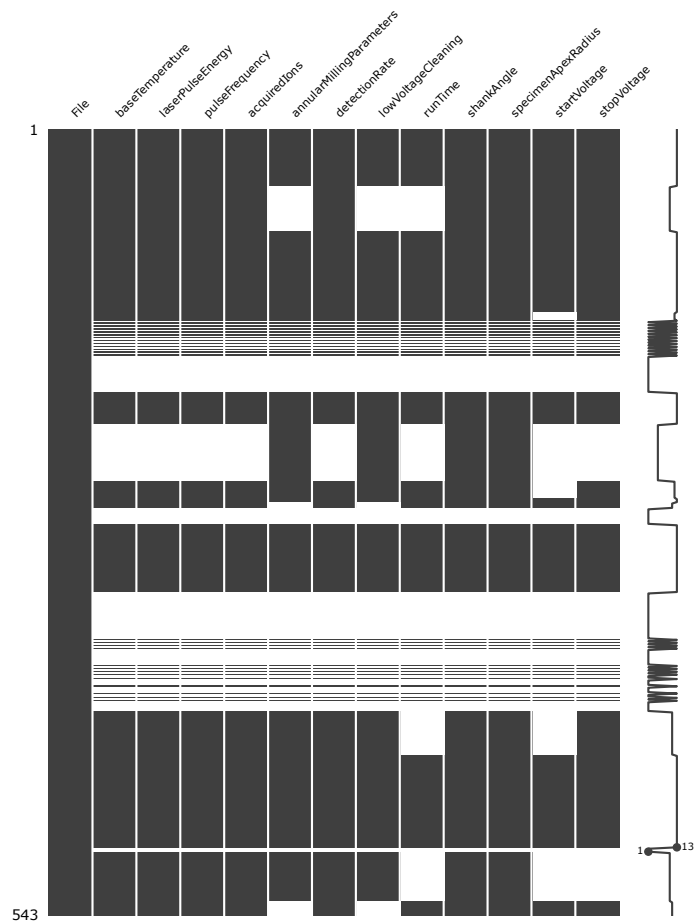


Figure 5.19: Sparsity of data for each feature. Columns for files and repositories work together as *Subjects*, feature headers serve as *Predicates*, and white spaces indicate missing *Objects*.

Table 5.8: Evaluating CB recommender system results for each class using Cosine and Euclidean distance metrics for contrast.

	Precision		Recall		F1		Support
	Cos.	Euc.	Cos.	Euc.	Cos.	Euc.	
Class 1	0.95	0.87	0.70	0.93	0.81	0.90	177
Class 2	0.47	0.70	0.29	0.88	0.36	0.78	153
Class 3	0.42	0.76	0.49	0.56	0.45	0.64	111
Class 4	0.26	0.76	0.32	0.68	0.29	0.72	82
Class 5	0.44	0.97	1.00	0.70	0.61	0.81	53
Macro Avg.	0.51	0.81	0.56	0.75	0.50	0.77	576
Weighted Avg.	0.58	0.80	0.52	0.79	0.53	0.78	576

measures how well the data points fit into their assigned clusters and the degree of separation between them. The results show that the Silhouette scores for both metrics indicate reasonable distances between the clusters. RMSE and MAE are also used to measure the difference between predicted and actual ratings. The results show high error rates for both RMSE and MAE when using the Cosine distance metric, indicating that the Euclidean distance metric performs better for this dataset.

The Area Under the Curve (AUC) score of 0.68 shown in Figure 5.21 suggests that the recommender system is reasonably effective in distinguishing between true positive and false negative items, as reported in Table 5.9. This is further supported by the Receiver Operating Characteristic (ROC) curves for each class shown in Figure 5.21. The Euclidean distance metric performs well with a recommender accuracy of 0.79, while the Cosine metric has a lower accuracy of 0.52. The evaluation metrics include RMSE and MAE, which show high error rates for the Cosine metric but lower for the Euclidean metric. These results demonstrate the importance of selecting an appropriate distance metric and the effectiveness of the proposed methodology in providing recommendations for data repositories in the domain of Material Science and Chemistry.

In addition to quantitative evaluation, a qualitative evaluation of the recommender system was conducted. The code base was converted into a publicly available python

Table 5.9: Overall effectiveness of Cosine and Euclidean metrics in recommending accurate Resources.

	Silhouette	RMSE	MAE	AUC	Accuracy
Cosine	0.68	0.83	0.54	0.68	0.52
Euclidean	0.74	0.54	0.24	0.68	0.79

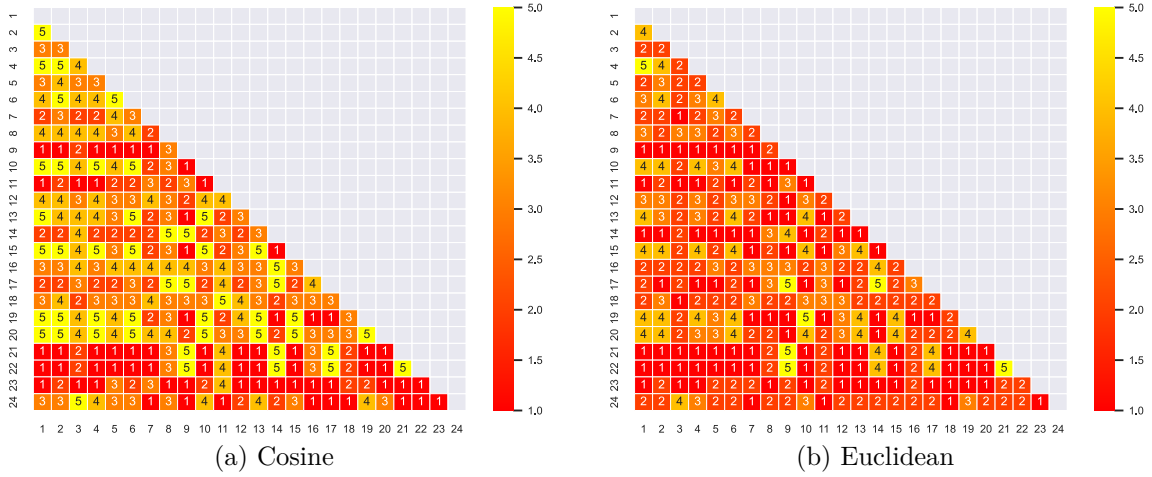


Figure 5.20: Pairwise similarity ratings for 24 repositories, where the values range from 1 (dissimilarity) to 5 (high similarity).

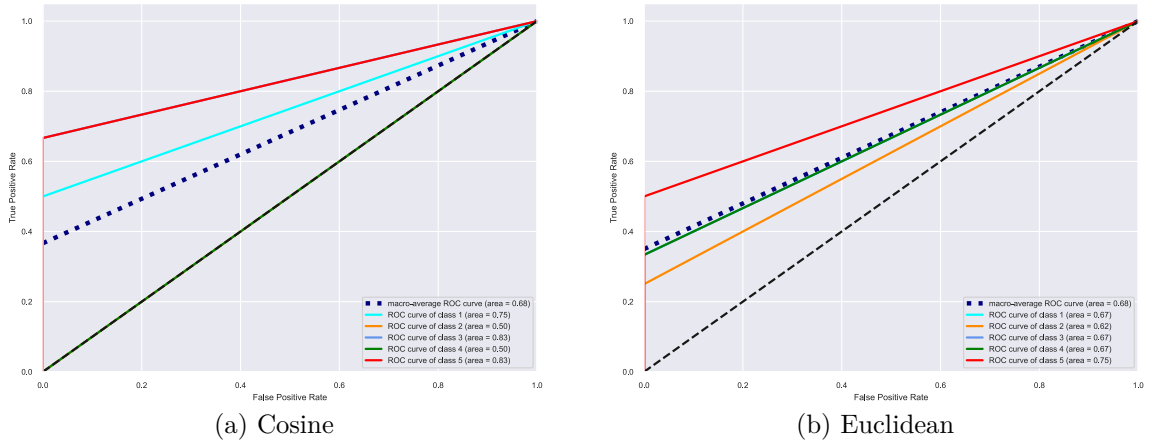


Figure 5.21: AUC for the overall and class-specific performance of the system to distinguish between true-positive to false-positive ratio.

package<sup>3</sup> where a repository Id (pivot Id) and a set of pre-selected features are given as package input, and it returns the top five similar items. A domain expert evaluated the system by picking the independent variables as parameters. The study was conducted twice using the Euclidean distance factor. During the 30 minutes long assessment, the domain expert reviewed files and their metadata for the recommended set and heuristically judged the similarity and ranking of findings. The evaluation involved a detailed review of each recommended item. The user agreed with the recommendations by stating, “*these two repositories seem to be really alike, maybe they should be merged.*” suggesting that project members have unknowingly created separate repositories for similar studies. Additionally, the user was able to mentally construct a mind map of the relations between the recommended items and understand the reasoning behind the recommendations without the need for explicit justifications by mentioning “*I think I can see why this one is not the first [recommended Repository in the list of top five].*”

### 5.4.2 Item Based Recommender for Coscine Resources

Unlike the CB recommender system, the study utilized the PCC as the similarity metric instead of the Euclidean distance. The presence of significant magnitude differences in the input dataset drove this decision. As the dataset contains items with varying ratings scales and degrees of user engagement, the Euclidean distance can be sensitive to these discrepancies, potentially leading to inaccurate similarity assessments between items.

On the contrary, the PCC and Cosine are robust to magnitude differences, primarily focusing on the linear relationship between two variables. Furthermore, by centering the data around the mean, Pearson correlation emphasizes the pattern of user preferences, making it more suitable for capturing the underlying structure of the dataset.

This study analyzes a two-year sample event log from the Coscine platform. Given that the platform was in its pilot phase and under continuous development during this period, users primarily engaged with Coscine cautiously and out of curiosity, as opposed to genuine RDM usage. Consequently, considerable effort and resources were dedicated to data preprocessing and identifying the most relevant sample data to ensure the final dataset accurately represents serious RDM users.

The refined dataset contains records from July 2021 to February 2023, encompassing 26 resources and 69 unique user Ids. On average, each Resource exhibits 100 unique user interactions. The Figure 5.22 provide a comprehensive overview of the sample dataset employed for this case study.

Figure 5.22a illustrates the number of unique users interacting with each Resource, focusing on data collections (Resources) with a minimum of at least four unique

---

<sup>3</sup><https://pypi.org/project/DA4RDM-RecSys-ContentBased/>

users per Resource. Figure 5.22b demonstrates users' average time on each Resource. Figure 5.22c highlights the distribution of activities within the selected dataset, emphasizing operations occurring on files. Meanwhile, Figure 5.22d presents the density of calculated implicit ratings.

To avoid bias in the recommender system, the dataset was balanced so that the ratings were evenly distributed across all classes (i.e., 1-5) using the SMOTE technique. Then, following pairwise Resource-to-Resource similarity evaluation using Cosine and PCC measures, the output was normalized between 0 and 1, and the results were classified into five classes, as described earlier. Here, a rating of 1 represents the least similarity, while a rating of 5 indicates the highest similarity between resource pairs.

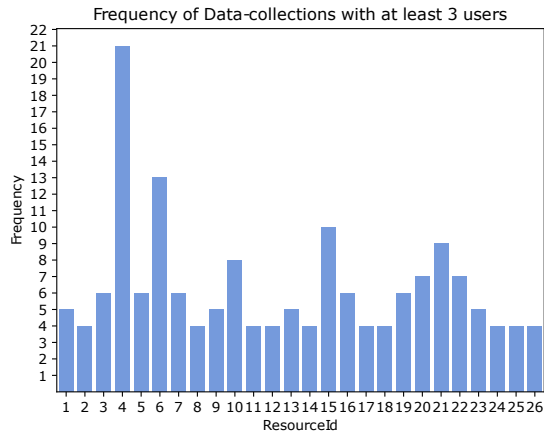
The Hold-Out method, a widely adopted evaluation technique in machine learning, was employed to assess the recommender system's effectiveness. The Hold-Out method involves splitting the dataset into training and test sets. In this case, 70% of the data was allocated to the training set, which was used to train the model, while the remaining 30% formed the test set, utilized for evaluating the model's performance. This approach helps simulate real-world scenarios by assessing the model's ability to generalize and perform well on unseen data.

The Singular Value Decomposition (SVD) algorithm was employed for the recommender system. SVD is a matrix factorization technique that has gained popularity in recommender systems due to its ability to handle sparse data and scalability compared to KNN algorithms [234]. While KNN-based methods can suffer from computational inefficiencies when dealing with large datasets, SVD can efficiently process high-dimensional data, making it a more suitable choice for this study [235].

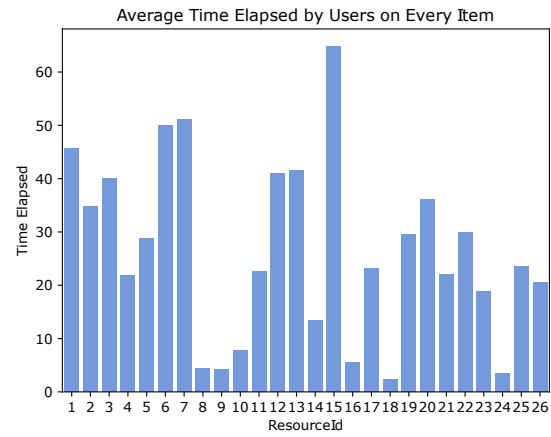
To further evaluate the recommender system's performance, a 5-fold cross-validation approach was used to calculate the RMSE and MAE metrics. Cross-validation is a robust method for model assessment that reduces the risk of overfitting and provides a better estimation of the model's performance on unseen data. In 5-fold cross-validation, the dataset is divided into five equal parts, and the model is trained and tested five times, with each fold serving as the test set exactly once. The RMSE and MAE metrics, which measure the average difference between the predicted and actual ratings, are then averaged across the five iterations to obtain the final performance scores. By evaluating the model using these metrics, a better understanding of its predictive accuracy and identifying areas for potential improvement can be achieved.

Figure 5.23 demonstrates the pairwise similarity of resources, with similarities classified on a scale of 1 to 5. Table 5.10 provides a comparative performance analysis of the two metric systems, showing each class's Precision, Recall, and F1 scores.

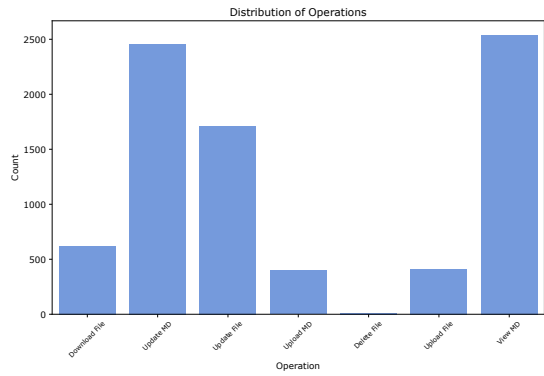
Interestingly, the performance for classes 4 and 5 do not yield any results for both Cosine and Pearson for Precision, Recall, and F1. This could be attributed to users' interaction patterns with each Resource being very distinct and thus not fitting other user interaction models of other Resources.



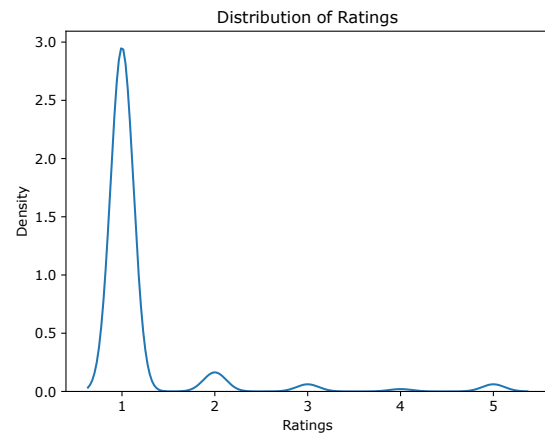
(a) Frequency of data collections with at least 3 user interactions.



(b) Average Time Elapsed.



(c) Distribution of Operations.



(d) Distribution of Ratings.

Figure 5.22: Statistical analysis of sample dataset used in IICF acquired from Coscine platform.

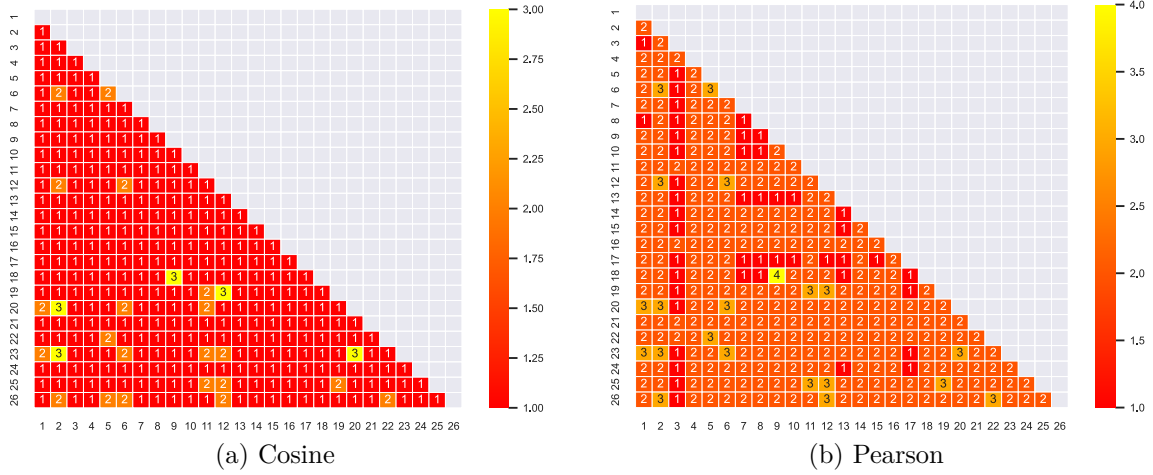


Figure 5.23: Predicted pairwise ratings for 26 repositories, where the values range from 1 (dissimilarity) to 5 (high similarity).

The Support shows that most classifications fall into class 1 or 2. For example, the Macro Average F1 score for Cosine and Pearson is 0.27 and 0.23, respectively. The Macro Average F1 score measures the quality of a classification model by averaging the F1 scores of each class. The F1 score is the harmonic mean of precision and recall. These scores indicate the overall performance of the recommender system for both similarity metrics and provide a basis for comparing them. However, the relatively low Macro Average F1 scores suggest that the recommender system does not perform optimally for all classes.

Unfortunately, Cosine and Pearson identify similarities that fall into classification 1 or 2 and thus provide higher performance for classes with the highest Support. Table 5.11 shows that the overall performance for Cosine is 0.79 and Pearson 0.76. Although the overall performance appears promising, considering classification results from Table 5.10, the conclusion can be drawn that high accuracy is due to the lack of diversity in similarities and distribution of classes across the pairwise resources similarities.

Regarding error metrics, the MAE and RMSE for Cosine are 0.44 and 0.85, respectively, while for Pearson, they are 0.40 and 0.71. These values indicate that, on average, the recommender system has slightly lower errors when using the Pearson similarity metric than the Cosine similarity metric. In addition, the lower MAE for Pearson indicates that, on average, the absolute differences between the predicted and actual values are smaller for Pearson than for Cosine. Similarly, the lower RMSE for Pearson suggests that the system is less sensitive to more significant errors when using Pearson as the similarity metric.

Despite the high overall accuracy of the recommender system, the system's reliability cannot be determined due to high data sparsity and a small dataset with reliable



Table 5.10: Comparing the recommender system results for Cosine and PCC distance metrics on each class using IICF recommender.

	Precision		Recall		F1		Support	
	Cos.	PCC	Cos.	PCC	Cos.	PCC	Cos.	PCC
Class 1	0.90	0.60	0.86	0.10	0.88	0.18	181	29
Class 2	0.13	0.77	0.36	0.97	0.20	0.86	11	154
Class 3	0.00	0.25	0.00	0.09	0.00	0.13	3	11
Class 4	0.00	0.00	0.00	0.00	0.00	0.00	0	1
Class 5	0.00	0.00	0.00	0.00	0.00	0.00	8	8
Macro Avg.	0.26	0.32	0.31	0.23	0.27	0.23	203	203
Weighted Avg.	0.81	0.69	0.79	0.76	0.80	0.69	203	203

data. In conclusion, the current analysis reveals some limitations in the recommender system's performance, which could be addressed in future work by employing larger, more diverse datasets and exploring alternative similarity metrics and classification techniques.

### 5.4.3 Summary

In conclusion, this study has investigated the performance of an IICF recommender system and compared it to a CB recommender system. Although IICF has some advantages, such as its ability to handle large datasets and provide recommendations based on user behavior, the findings indicate that it may not be the most suitable approach for recommending data repositories according to user interaction models.

One reason for this is the data sparsity issue inherent in IICF. Due to the limited interactions between users and resources, the recommendation quality may suffer as the system struggles to identify meaningful patterns in user behavior. Moreover, the performance of the IICF system relies heavily on the quality and quantity of user interactions, which might not have been sufficient in this case.

The study's quantitative and qualitative evaluations have offered valuable insights into the effectiveness of the recommender system in providing relevant recommendations and identifying similar repositories. These findings are consistent with previous

Table 5.11: Overall performance of Cosine and PCC distance metrics to suggest correct items.

	RMSE	MAE	Accuracy
Cosine	0.85	0.44	0.79
PCC	0.71	0.40	0.76

research, highlighting the importance of domain expertise in evaluating recommender systems [236]. The work also builds upon recent research on recommender systems, such as Zhang et al.'s study [234] on deep learning-based recommendation models.

The study demonstrates that a CB recommender system can effectively enable the reusability of research data repositories by utilizing RDF-based graph data models. By leveraging the relationships between research data items and their structured metadata, these systems can significantly improve the discoverability and reuse of research data. This fosters collaboration among researchers and reduces the research community's repetition of efforts and replication of resources.

The research has yielded promising outcomes for the CB recommender system by utilizing the Euclidean distance metric after applying NLP to an unstructured sample dataset. However, the recommender system is limited to functioning for datasets compelled by uniform metadata profiles. To overcome this constraint, it is suggested that OSPs facilitate inheriting RDF graphs or reusing metadata profiles to enable the CB recommender system effectivity across a wider domain of data repositories. This would allow for achieving interdisciplinary data reusability and collaborations.

On the other hand, CB recommender systems use resource features and metadata to provide recommendations. This approach can be more robust in handling sparse datasets and offer personalized suggestions based on the content rather than relying on user behavior patterns. Furthermore, CB recommender systems can address the cold start issue more effectively, as they do not require many user interactions to make meaningful recommendations.

Throughout the investigation, the strive has been to contribute to the expanding body of literature on RDM, providing valuable insights into the potential of CB recommender systems in this context. By implementing these systems with standardized metadata profiles, such as those offered by RDM platforms like Coscine, researchers can benefit from a more streamlined approach to managing their research data. This facilitates greater interoperability and discoverability of research artifacts, ultimately promoting the FAIR principles of research data management.

The findings underscore the importance of further exploration and development of CB recommender systems in the context of RDM and the need for continued collaboration between researchers, domain experts, and institutions in creating standardized metadata profiles. By doing so, the research community can continue to enhance the discoverability and reuse of research data, ultimately driving scientific progress and innovation.

In light of these findings, it is suggested that CB recommender systems might better fit the requirements. However, it is essential to note that the choice of the recommender system should be tailored to the specific conditions and constraints of the problem domain. Further research could explore the potential benefits of a hybrid

recommender system, combining the strengths of IICF and CB approaches to provide users with more accurate and personalized recommendations.



## 6 Conclusion

In this Chapter, the key findings of the research are incorporated, revisiting the central research questions and examining the implications of the results in the context of the broader literature on RDM and Open Science processes. Next, this thesis's main contributions are presented, encompassing a combination of methodologies, design criteria for improving RDM practices, and proof of concept through implementations and case studies. Moreover, potential future research directions that could build upon and extend the work are outlined, addressing the evolving needs of the RDM community and further enhancing the effectiveness of OSPs. Finally, by reflecting on the achievements and limitations of the study, an attempt is made to provide a comprehensive understanding of the research's impact and the potential for future advancements in the field.

### 6.1 Answering Research Questions

Here, the findings are presented and discussed concerning the research questions posed at the beginning of this dissertation. By systematically exploring each research question, various aspects of the proposed framework, methodologies, and design criteria are clarified, highlighting their effectiveness in enhancing process-aware RDM activities and modeling the underlying actual practices.

***Main Research Question:*** How to enable discovering and enhancing RDM practices via modeling the underlying activities?

Figure 6.1 illustrates a comprehensive meta-model encompassing all sub-research questions and their corresponding research questions. This meta-model is the foundation for the investigation into enabling the discovery and enhancement of process-aware RDM activities via modeling the underlying user actions. In Chapter 2, a universal reference software architecture tailored to OSPs was established to scope the conclusions and provide a robust and flexible basis for addressing the sub-research questions.

To comprehensively address the main research question, the main research question is partitioned into five data-driven sub-research questions, each focusing on specific research gaps pinpointed in the existing literature revealed in Figure 1.1. These

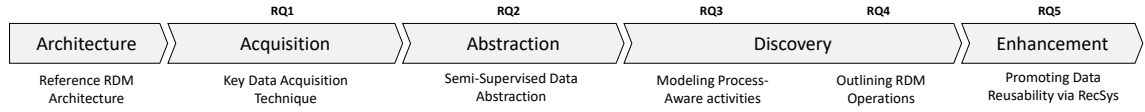


Figure 6.1: A schematic representation of a meta-model for research questions.

sub-research questions facilitated a systematic and targeted approach, allowing for a deeper dive into the complexities of research data management processes and challenges. The findings for these sub-research questions have significantly contributed to the field of RDM, both methodologically and empirically, through developing novel approaches and implementing case studies.

The meta-model presented in Figure 6.1 provides a coherent and structured framework for the investigation, highlighting the interconnectedness of the sub-research questions and the overarching primary research question. Adopting this meta-model ensured a comprehensive and cohesive exploration of the research problem, allowing for addressing the complex and multifaceted nature of RDM. Furthermore, the findings could potentially inform and improve RDM practices across various domains, promoting efficient and effective data management within the broader research community.

**Research Question 1:** How to overcome the challenges of acquiring datasets suitable for process and data analytics from continuously evolving RDM services while maintaining data quality standards?

The research question was defined in the context of the challenges associated with dynamic data capturing in distributed environments, focusing on the need for scalable, maintainable, and high-quality datasets that adhere to data protection laws.

In Chapter 3, the dataset qualities necessary for analyzing user and software interactions in distributed settings are characterized and defined. Correspondingly, a series of methodologies were proposed to answer the research question, and case studies were conducted, each offering valuable insights and contributions to the field. Based on the reference architecture for RDM services, techniques for acquiring datasets were discussed by instrumentation of Authorization service (Centralized logger) or extending of Client-Side and Server-Side software components.

The case studies examined in Section 5.1.1 and Section 5.1.2, concentrating on Centralized logger and Server-Side logger methods, respectively, demonstrated the effectiveness of presented approaches in providing valuable insights, discovering process models, identifying bottlenecks, and enabling adaptable datasets for various data analysis projects from distributed systems. Nevertheless, both methods had challenges, such as integration and maintenance efforts, system overhead, and issues analyzing aggregated data.

To overcome the limitations of individual methods, in Section 3.4, a novel methodology for acquiring datasets from OSPs by combining and incorporating the strengths of the Centralized logger and Client-Side logger was introduced.

The Hybrid logger provided a comprehensive view of user activities and software component executions. It enables adequate accuracy in specifying user behavior on Client-Side by merely observing traces of software executions captured by the Authorization service. The suggested Hybrid technique generates datasets without needing cookies, explicitly tracking every user action or finding and excluding irrelevant activities within distributed systems. However, the methodology also revealed limitations, such as the lack of capturing events that do not require authorization service for execution, and discussed the algorithm’s sensitivity towards similar Client activities.

The methodologies and case studies presented, offer valuable insights and provide a comprehensive approach to addressing the challenges associated with data capturing in distributed environments. Likewise, the benefits and limitations of all methods are detailed and presented, indicating their applicability according to a data-driven study. All established approaches comply with the GDPR requirement of not disclosing user-specific information using pseudonymized user Ids.

**Research Question 2:** How to effectively abstract and transform low-level datasets into interpretable high-level representations for RDM services while balancing granularity and model fitness?

The research question was defined in the context of the complex and unstructured landscape of user processes in institutional OSPs with heterogeneous and distributed IT infrastructure. The goal was to propose a methodology that deals with unordered event data while ensuring the discovery of structured and analyzable process models.

A novel methodology was proposed in Section 4.1, and two case studies were conducted to answer the research question, each offering valuable insights and contributions to the field. The offered method adopts a semi-supervised learning approach, utilizing continuous and discrete data to discover process models and measure the similarity between activities to help with the task of abstraction and activities reannotation. The technique accommodates non-sequential events due to the nature of distributed systems and has successfully abstracted event logs, transforming low-level logs into high-level representations. By identifying the optimal level of abstraction, an accurate representation of real-world behaviors was ensured while maintaining sufficient detail for analysis and interpretation.

The case studies explained in Section 5.2, employed the Hybrid logger to acquire the sample dataset and demonstrated the approach’s effectiveness in providing meaningful insights into the underlying processes and identifying areas for improvement. In addition, a correlation was found between model fitness and the accuracy of signifying user activities based on the corresponding sequence of software component

executions using the DT and RF machine learning algorithms on the Data Archiving Tool and Metadata Management Tool, indicating the importance of maintaining a balance between abstraction and achieving meaningful insights.

In summary, the findings for this research question provide a strong foundation for future research in abstracting process models for information services while balancing data granularity with respect to models' fitness. In addition, the methodology and case study presented offer valuable insights and a comprehensive approach to addressing the challenges associated with unordered event data and the discovery of structured and analyzable process models.

**Research Question 3:** How can business process intelligence be incorporated into existing distributed RDM services to provide insights for user and system activities?

The research question aimed to investigate incorporating BPI algorithms across available services to deliver insights into hierarchical control-flow modeling of microservices and software components supporting the Open Science processes. The goal was to efficiently enable reverse-engineering the non-trivial processes in a decentralized landscape of IT systems and provide means to generate RDM process models based on user activities.

A proposed method led to the development of an information system called DA4RDM and case studies were conducted to answer the research question, each offering valuable insights and contributions to the field. The software architecture for DA4RDM reviewed in Section 4.2 employs a modular and layered architecture with APIs for data source handling, pre-processing, process discovery, and user session management. The choice of design and technologies, including Python, Flask, PM4PY, SQLAlchemy, and SocketIO, provide flexibility, extensibility, and a rich ecosystem of libraries.

In the case study examined in Section 5.1.1, middleware microservices and services in the highest demand were identified by evaluating the discovered process model. Furthermore, hierarchical modeling processes within distributed services allowed for assessing the impact of newly developed software components on related services and identifying bottlenecks in the system.

The further case study presented in Section 5.3.1 demonstrates the approach's effectiveness in providing deep insights into the system's actual behavior and identifying services that function independently or as groups.

Moreover, the case study in Section 5.3.2 demonstrates the application of DA4RDM in providing valuable insights for research data management, including checking compliance with FAIR principles or RDM processes and identifying areas where deviations frequently occur. Analyzing user behavior patterns can determine if certain activities or stages require more assistance, helping to improve user experience and provide targeted support.



In Summary, the findings of this research question provide a strong foundation for utilizing BPI and its applications in RDM. The methodology and case study offered valuable insights and a comprehensive approach to addressing the challenges of incorporating process analytics algorithms across heterogeneous and distributed services. Despite the limitations and areas for improvement, the proposed practice has the potential to significantly improve the understanding and optimization of RDM processes in OSPs.

**Research Question 4:** How to outline the RDM lifecycle for research projects and assess its corresponding process-aware requirements in collaborative scientific platforms?

This research question aimed to utilize the gathered event data to bridge the knowledge gap regarding the actual practices of researchers within OSPs and RDM guidelines. The goal was to enable visualizing descriptive RDM stages for research projects while identifying RDM-related requirements using process-aware activities. Furthermore, outlining the RDM operations would encourage troubleshooting the potential barriers for each RDM phase and provide self-awareness regarding the current collective RDM activities within a research project.

A methodology was proposed in Section 4.3, and a case study was conducted to answer the research question, each offering valuable insights and contributions to the field. The developed methodology involves discovering and outlining RDM phases and determining the current RDM phase of a research project using Inductive Miner and Alignment Analysis as the two most reliable and least expensive methods for conducting conformance checking.

The case study in Section 5.3.4 demonstrates the proposed methodology's potential in outlining RDM phases and determining the current RDM phase of research projects. The discovered process models and the radar visualization of RDM phases provide valuable insights into the progression and activities of different research groups. In addition, the granularity of the collected information and domain knowledge play crucial roles in the analysis.

The methodology and case study offered valuable insights into discovering the actual RDM lifecycle via a bottom-up investigation of RDM activities within an OSP, contributing significantly to understanding and optimizing the RDM process within a research community. It facilitates locating the researchers' challenges in complying with RDM guidelines and principles and assessing their corresponding process-aware requirements. By incorporating the introduced methodology and domain knowledge, research steering committees can adapt FAIR guidelines or respond to researchers' needs within an RDM platform.

**Research Question 5:** How can recommender systems enhance research data discoverability and reusability in RDM Platforms while maintaining user control over data artifacts?

The primary goal was to develop methods to enable the reusing of research data repositories without compromising sensitive information, leveraging available datasets on user activities and knowledge graphs.

Two recommender systems were proposed and evaluated in Section 4.4: Content-Based (CB) and Item-Item Collaborative Filtering (IICF) recommender systems. The findings suggest that the CB recommender system better suits the specific use case due to leveraging application profiles (specialized metadata schema) and offering personalized suggestions based on unstructured content rather than user behavior patterns, making it more robust in handling sparse datasets and addressing the cold start issue.

The offered methodologies ensure user control over research data by aggregating files' specialized metadata or studying collective user activities rather than looking into individual classes or pairwise file comparisons. The resulting recommenders return PIDs as data collections (repository location) that require the data manager's explicit authorization to view files.

The case study in Section 5.4 demonstrated that the CB recommender system could effectively enable the reusability of research data repositories by utilizing RDF-based graph data models. By leveraging the relationships between research data items and their selected metadata, the system can significantly improve the discoverability and reuse of research data, fostering collaboration among researchers and reducing the replication of efforts and resources in the research community.

Furthermore, the case study presented in Section 5.4.2 utilized the IICF recommender, which yielded poor results. The study suggests that IICF might not be ideal for recommending data repositories based on user interaction models. This stems from IICF's inherent data sparsity issue, which can compromise recommendation quality due to inadequate user-resource interactions. Additionally, IICF performance relies heavily on the quantity and quality of these interactions, which were insufficient in this context.

The study's findings contribute to the expanding body of literature on RDM and provide valuable insights into the potential of CB recommender systems in this context. Also, by implementing these systems with standardized metadata profiles, such as those offered by RDM platforms like Coscine, researchers can benefit from a more streamlined approach to find and reuse research data, promoting the findability and reusability of FAIR principles.

Finally, the research results highlight the importance of further exploring and developing CB recommender systems in the context of RDM. Continued collaboration between researchers, domain experts, and institutions in creating standardized metadata profiles is crucial to enhance the discoverability and reuse of research data, ultimately driving scientific progress and innovation. Future work could investigate

a Hybrid recommender system, combining the strengths of Item-Item Collaborative Filtering and CB approaches to provide users with more accurate and personalized recommendations to foster the reusability of research data collections.

## 6.2 Contributions

In this dissertation, significant contributions have been made to the body of RDM and OSPs by introducing novel methodologies, presenting design criteria, and providing proof of concept through implementations and case studies. The contributions can be summarized in three prominent folds:

- 1) *A Set of Comprehensive Methodologies:* A series of methodologies addressing various aspects of RDM and OSPs, from acquiring and abstracting datasets to discovering new knowledge and recommending similar data repositories, were presented. These methodologies were designed to work together and complement each other, providing a comprehensive solution for analyzing process-aware RDM activities. Specifically, the following were introduced:
  - A Hybrid logger approach for acquiring datasets from OSPs that combines the strengths of Centralized logger and Client-Side logger methods.
  - A semi-supervised learning approach for abstracting datasets in order to discover process models and measure the similarities between user activities and reannotating activity labels in Client-Server applications.
  - A modular web-based application (DA4RDM) incorporating business process intelligence algorithms to discover and assess user and software interactions across heterogeneous and distributed services.
  - A methodology for discovering and outlining the actual RDM lifecycle and determining the current RDM phase of research projects.
  - Two recommender systems for enhancing the findability and reusability of research data repositories.
- 2) *Design Criteria for Improving RDM Practices:* Based on the findings, specific design criteria and guidelines for improving RDM practices were derived, which can be applied to various OSPs and RDM providers. These criteria include:
  - Identifying and selecting the suitable strategy for acquiring datasets for a data-driven study, according to a project requirement, while ensuring user privacy compliance in data acquisition.
  - Balancing data granularity and model fitness for process model abstraction with respect to demands of domain experts.

- Promoting the use of standardized and inheriting metadata profiles to enable recommender systems and enhance data discoverability and reusability.
- Fostering collaboration between researchers, domain experts, and institutions to develop and implement RDM practices by modeling and study of actual user interaction models.

3) *Proof of Concept with Implementations and Case Studies:* The applicability and effectiveness of the proposed techniques and design criteria have been demonstrated by implementing proof-of-concept solutions and conducting case studies. The implementations and case studies provided valuable insights into the real-world applications of the methodologies and confirmed their potential to improve RDM practices in OSPs. Furthermore, these proof-of-concept solutions apply to other RDM systems, provided they adhere to the general architecture of RDM systems.

The contributions of this interdisciplinary study have enabled a more comprehensive understanding of process-aware RDM activities, which were previously inadequate and limited. Furthermore, it lays the groundwork for further discipline-oriented studies, which were before unexplored and ambiguous. The study's goals have been achieved in advancing RDM practices by providing guidelines on implementing the necessary architecture and methodologies to enhance process-aware RDM activities and services and demonstrating the applicability of these methods to real-world scenarios. These contributions can be leveraged by researchers, domain experts, institutions, and OSPs maintainers to enhance the findability, interoperability, and reusability of research data.

### 6.3 Outlook

As with any research area, there are opportunities for further advancements and exploration. This Section provides an outlook on potential work and improvements that can build upon the findings.

1. **Enhancing Data Acquisition Techniques:** While the Hybrid logger approach has demonstrated promising results in acquiring datasets from OSPs, further refinements could be explored to address its limitations, such as capturing events that do not require authorization service (OAuth) and reducing the sensitivity of the ELG algorithm towards similar client activities. Developing new data acquisition techniques and tools with higher privacy and security compliance would also contribute to the field.

2. **Adapting to Evolving RDM Standards and Practices:** As the RDM landscape evolves, the methodologies and design criteria should be updated and refined to accommodate new standards, guidelines, and best practices. This may involve incorporating new data models, metadata standards, and vocabularies and adapting to changing legal and ethical requirements.
3. **Expanding the Scope of Recommender Systems:** Future work could investigate Hybrid recommender systems that combine the strengths of IICF and CB approaches to provide users with more accurate and personalized recommendations. Additionally, exploring the integration of other recommender systems or incorporating contextual information, such as users' research domains and collaboration networks, could further enhance the recommendations and foster the reusability of research data collections.
4. **Scaling and Optimizing the Proposed Architectures:** As OSPs and RDM systems grow in size and complexity, scalability and optimization of the proposed architectures will become increasingly important to process large-scale datasets. Future work could investigate the development of more efficient algorithms, parallel processing techniques, and distributed computing solutions to ensure the proposed methodologies can handle the increasing demands of large-scale RDM systems.
5. **Providing complementary Data Provenance Feature:** Future work could explore integrating data provenance techniques into the RDM processes and OSPs to provide a comprehensive view of the data's lifecycle, including its origin, processing history, and ownership. Although the method for discovering process-aware activities can provide a holistic view of user activities, one still needs to track (via versioning) and tailor the data changes over time to user actions to provide a suitable and comprehensive representation to end users. Incorporating data provenance into RDM systems can enhance data trustworthiness and reproducibility of research results. Developing methodologies to capture, store, and query provenance information efficiently and effectively would be valuable to the field.
6. **Evaluating the Impact of the Proposed Methodologies:** Conducting long-term evaluations of the impact of my proposed methodologies on RDM practices and OSPs would provide valuable insights into their effectiveness and potential areas for improvement. This could involve analyzing the adoption of the methodologies across different OSPs, their effect on research collaboration and data reuse, and assessing their impact on the overall research performance.

The contributions made in this dissertation serve as a foundation for tackling the newly identified areas outlined here. This work has presented a comprehensive data analytics framework that has established novel methodologies, design criteria, and proof-of-concept implementations that pave the way for further exploration and improvements

in RDM and OSPs software architecture. Furthermore, the groundwork has been laid for future research in these areas by addressing current challenges and providing practical solutions. This enables the scientific community to delve into these new directions more effectively and efficiently by extending the presented methodologies, a task that would have otherwise been challenging without these contributions.

# Appendix

The datasets and developed applications that have been extensively discussed and demonstrated in this dissertation are available for further study. The raw datasets utilized throughout this research can be accessed here [237]. These datasets have been meticulously curated and processed to ensure their suitability for the various case studies. The developed web application (DA4RDM) and the source code to Python packages developed during this research can be accessed here [238]. These software applications are integral to the research methodologies and findings presented. Both the datasets and applications have been preserved for long-term availability and can be accessed by future researchers to replicate the study or to conduct new research that builds upon this work.

## A.1 Screenshots of Case Studies

The case studies under discussion have been effectively executed and functioned across the subsequent systems.

## DA4RDM

The screenshot shows the DA4RDM web interface with the 'Data Pre-processing' tab selected. The 'Data Source' section is active, displaying a form for defining a new data source. The form includes a 'Datasource name' input field, a 'Parameters' section with radio buttons for 'CSV', 'XES', and 'Database' (with 'CSV' selected), and a 'Choose file' button. The 'Save' button is highlighted in blue.

Figure A.1: Dataset source definition and parameters.

The screenshot shows the DA4RDM web interface with the 'Data Pre-processing' tab selected. The 'Data Source' section is active, displaying a dropdown menu with 'SampleData 2022' selected. The 'Data Pre-processing Pipeline' section is also active, displaying a dropdown menu with 'KPI\_analysis' selected. The 'Save' button is highlighted in blue.

Figure A.2: Selection of pre-defined data source and pre-processing pipeline.



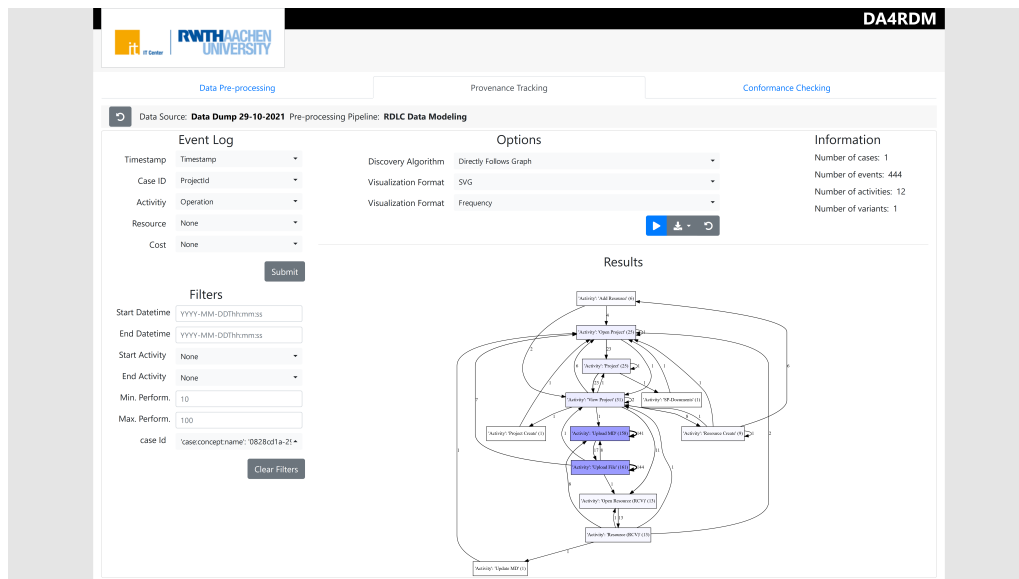


Figure A.3: Discovery of process models and filter options.

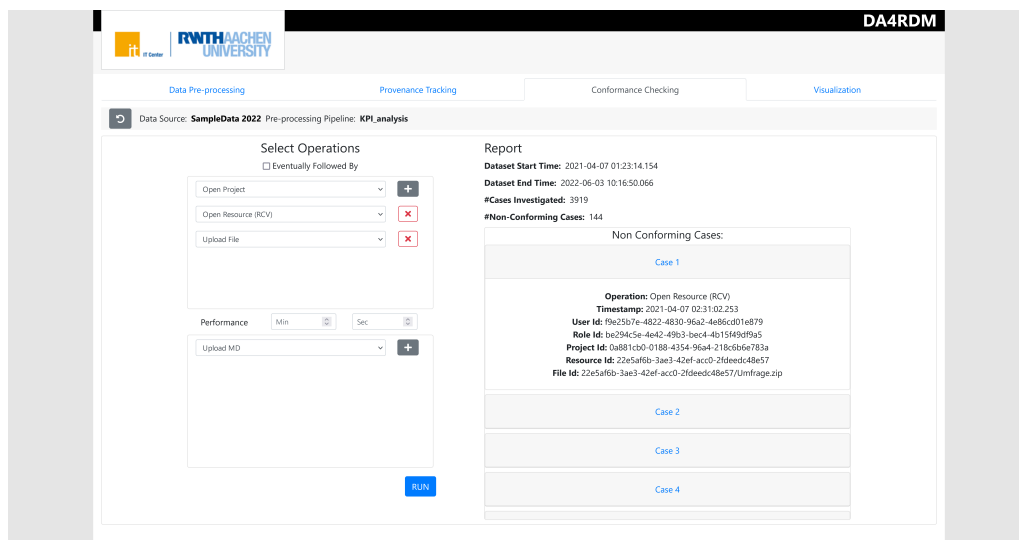


Figure A.4: Workflow conformance checking and requirement engineering.

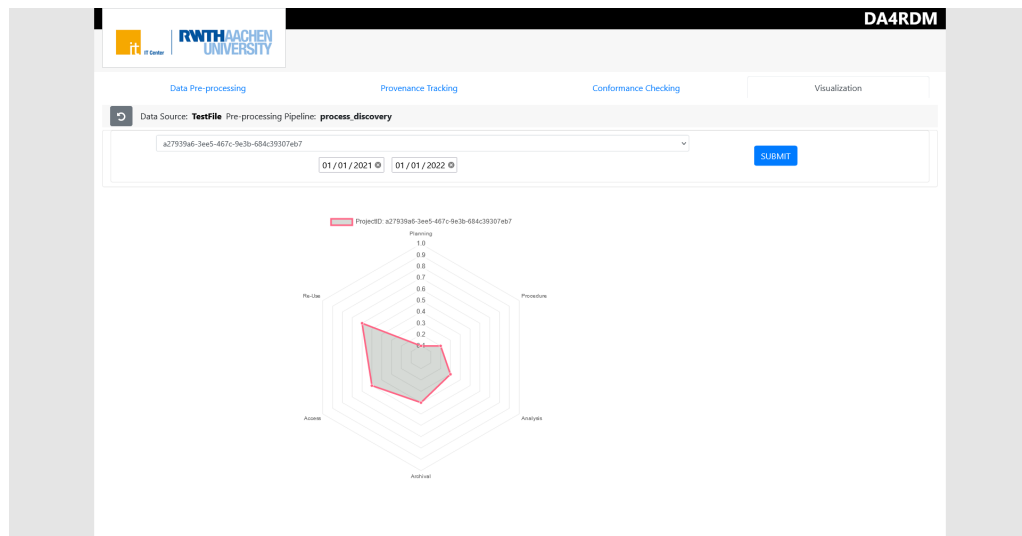


Figure A.5: Visualization of RDM phases for selected projects within a selected timeframe.

## Coscine Platform



Figure A.6: Coscine login page.

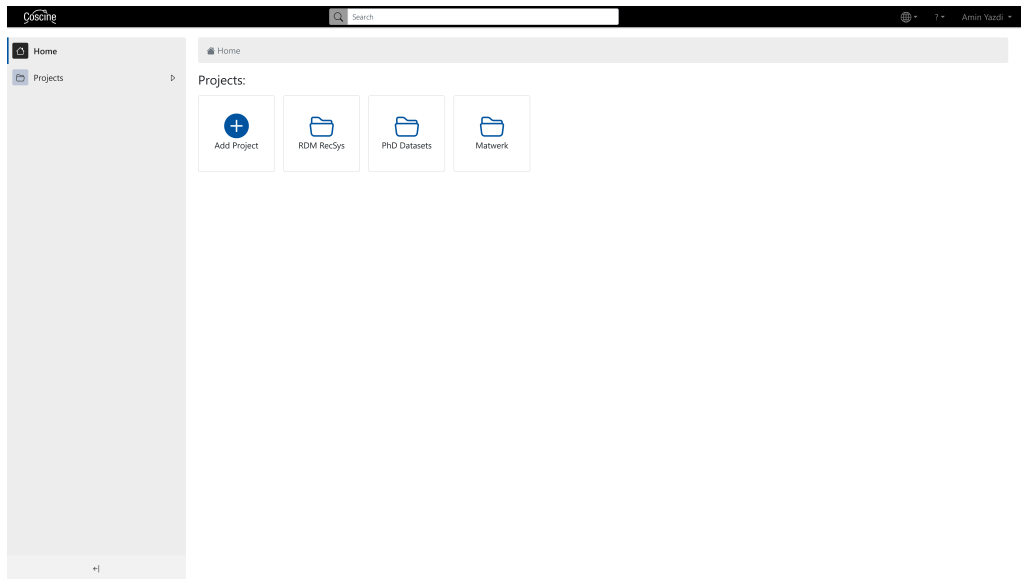


Figure A.7: Coscine dashboard.

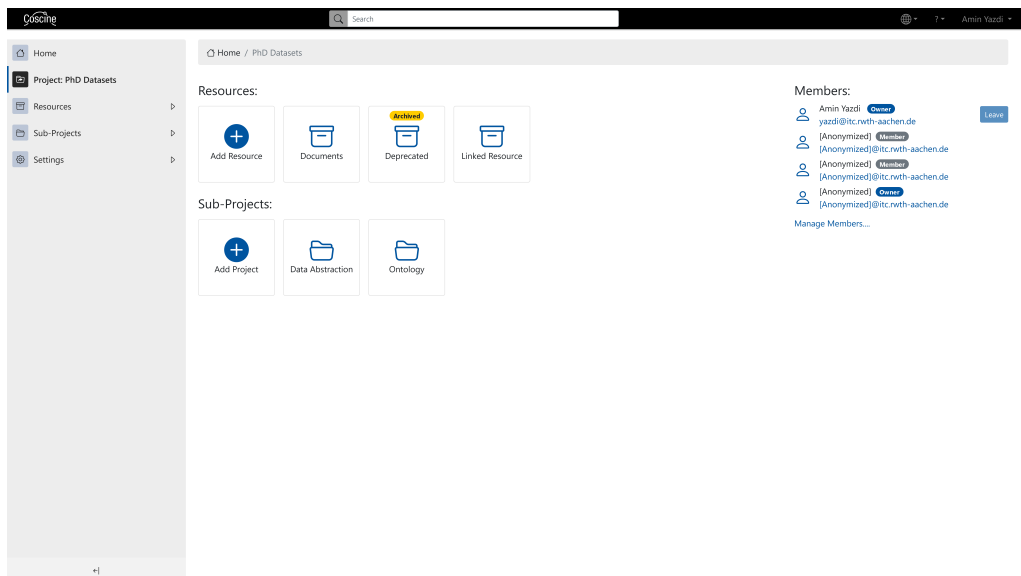


Figure A.8: Coscine project view, listing resources, sub-projects, and members.

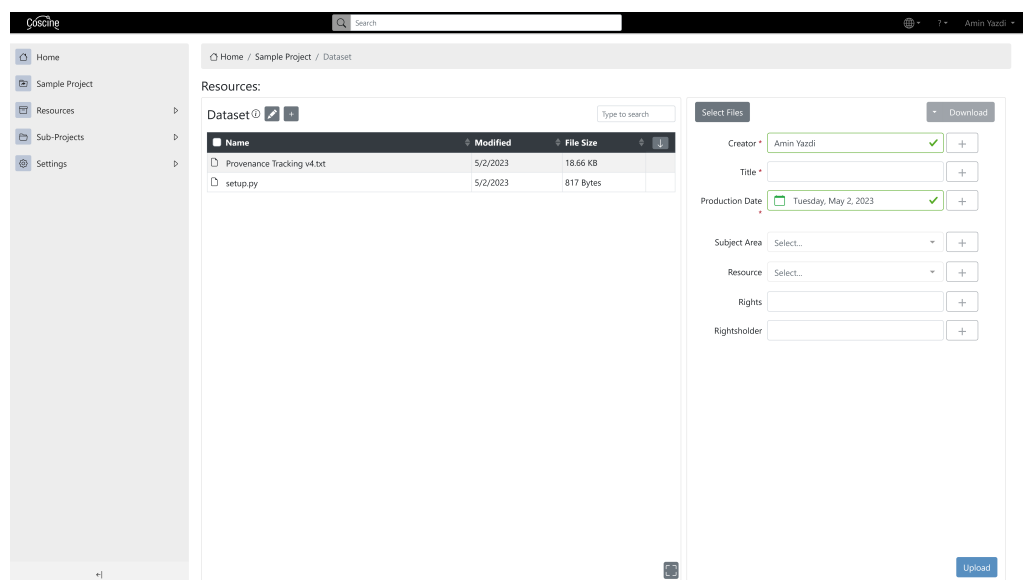


Figure A.9: Resource content view for metadata management.

## SimpleArchive Tool

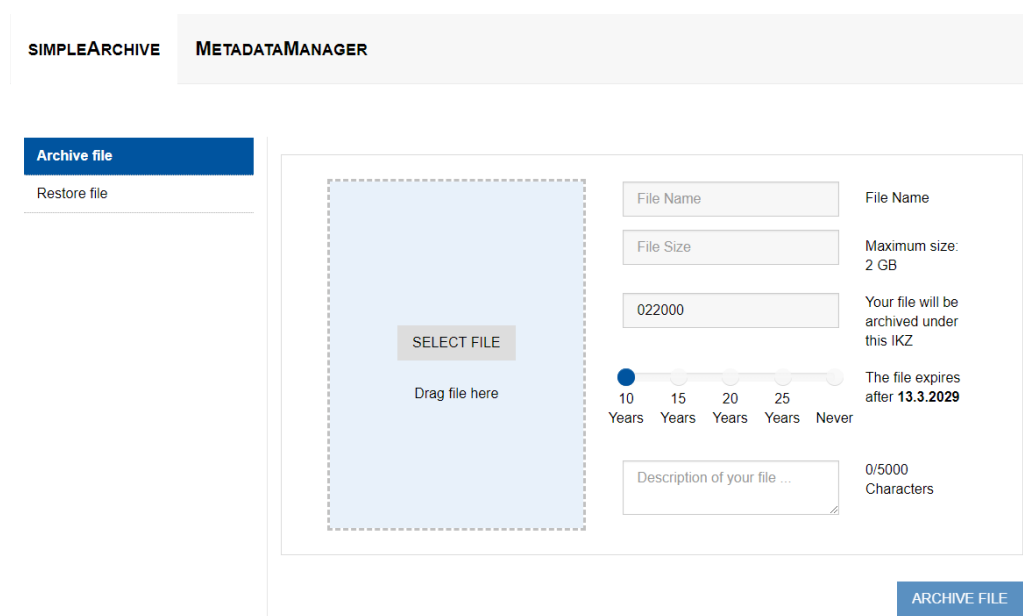


Figure A.10: Interface for archiving research data.

Paper.zip

Link to RWTH Publications

Upload Date

The file expires after

10

Years

15

Years

20

Years

25

Years

Never

Expiry Date ?

Your file will be archived under this IKZ

Description

0/5000 Characters

Restore

Download certificate

Add metadata

Copy PID

UPDATE

Figure A.11: List of archived nodes ready to be restored.

## Metadata Manager Tool

### Metadata Fields


The following fields are saved to the metadata file.

**Publish Metadata \*** ☐ private  
☐ institute  
☐ public

**IKZ \*** ☐ 022000  
☐ 123456  
☐ 012345

**Creator \***

**Title \***

**Production Date**  

**Subject Area**

**Resource**

**Rights**

**Rightsholder**

Figure A.12: Submitting customized metadata information (Base Application Profile).

SIMPLEARCHIVE

METADATAMANAGER

Submit

Manage

Search

Display 10 records per page

Search:








Title	Creator	Visibility	Issued	Schema	Referenced Data
temp.txt	Quilla Botester	private	02/25/2019	simpleArchive file	
d-sp05.rdp	Quilla Botester	private	02/25/2019	simpleArchive file	
package-lock.json	Quilla Botester	private	02/08/2019	simpleArchive file	
gulpfile.js	Quilla Botester	private	02/08/2019	simpleArchive file	
gulpfile.js	Quilla Botester	private	02/08/2019	simpleArchive file	
tfs.ps1	Quilla Botester	private	02/08/2019	simpleArchive file	
12341234.txt	Quilla Botester	private	02/08/2019	simpleArchive file	

Figure A.13: List of metadata entries that are linked with an archived node.

SIMPLEARCHIVE

METADATAMANAGER

Submit

Manage

Search

Select a state

only simpleArchive files? ☐

SEARCH

Display 10 records per page

Search:

Title	Creator	Visibility	Issued	Schema
temp.txt	Quilla Botester	private	02/25/2019	simpleArchive file
d-sp05.rdp	Quilla Botester	private	02/25/2019	simpleArchive file
package-lock.json	Quilla Botester	private	02/08/2019	simpleArchive file
gulpfile.js	Quilla Botester	private	02/08/2019	simpleArchive file
gulpfile.js	Quilla Botester	private	02/08/2019	simpleArchive file
tfs.ps1	Quilla Botester	private	02/08/2019	simpleArchive file

Figure A.14: Search user interface for metadata.

## A.2 Python Packages Developed

The Python packages developed and showcased in this dissertation are readily available for use, demonstrating their practical applications and ease of integration. Users can leverage these packages to enhance their projects and streamline various processes.

### Content-Based Recommender System

The `DA4RDM_RecSys_ContentBased` is a Python package that recommends related data collections by leveraging unstructured and explicitly supplied metadata associated with files. The package's repository contains a detailed README file containing in-depth usage guidelines. The package can be accessed publicly at: <https://pypi.org/project/DA4RDM-RecSys-ContentBased/>.

1

```
pip install DA4RDM-RecSys-ContentBased
```

Listing 6: Installation of the package

```
1 from DA4RDM_RecSys_ContentBased import preprocessor
2 from DA4RDM_RecSys_ContentBased import »
   distance_similarity_calculator
```

Listing 7: Importing the methods

```
1 #:param filepath: filepath to the csv file with '|' as »
   ↳ the seperator
2 #:param features: array of features to consider
3 #:param separator: the seperator used when loading the »
   ↳ csv-file
4 #:param encoder: set the language model: »
   ↳ 'mobilebert_multi_cased' or 'bert_multi_cased'
5 #:param minmaxScaleFlage: minmax scaling resource vector
6 #:param debug: debug mode
7 #:return: a preprocessed pandas.DataFrame
8
9 df = preprocessor.loadAndPreprocess_function( »
   ↳ filepath="PATH", features=['Predicate-1', 'Predicate-
   ↳ 2', 'Predicate-3', 'Predicate-4', 'Predicate-
   ↳ 5'], debug=False )
```

Listing 8: Usage example- load and preprocess the dataset

```
1 #:param df: preprocessed dataframe
2 #:param key: compare resources to this key
3 #:param distanceMethod: 'euclidean' or 'cosine' distance
4 #:param sortAscending: True = sort output ascending
5 #:param DEBUG_MODE: debug mode
6 #:param outputFormatJson: Trigger Json format
7 #:return: relative distance between key and furthest »
   ↳ resource
8
9 jsonOutPut = »
   ↳ distance_similarity_calculator.result_function(df, »
   ↳ "Key-Resource-Id", distanceMethod="euclidean", »
   ↳ outputFormatJson=True, DEBUG_MODE=False )
```

Listing 9: Distance similarity calculator



```
1      {"distance":{  
2          "Key-Resource-Id":0.0,  
3          "302231B4...":0.13,  
4          "4EFD8371...":0.34,  
5          "F8BE75F7...":0.36,  
6          "FAF13DF1...":0.36,  
7          "24CE68AD...":0.43,  
8          "632AD746...":0.44  
9      }}
```

Listing 10: Example JSON output

## Collaborative Filtering Recommender System

The `DA4RDM_RecSys_UserBased` is a Python package designed to recommend data-collections by analyzing user behavior patterns toward data-collections and conducting an exploratory examination of user-resource interactions. The package's repository contains a comprehensive README file with detailed usage instructions. The package is publicly available at: <https://pypi.org/project/DA4RDM-RecSys-UserBased/>.

```
1      pip install DA4RDM-RecSys-UserBased
```

Listing 11: Installation of the package

```
1      from DA4RDM_RecSys_UserBased import get_recommendation
```

Listing 12: Importing the method

```
1  ....#:param datapath: filepath to the csv file, a string is expected
2      #:param ref_user: the user for which resources are to be
   ↪ recommended
3      #:param num_recommendation: number of recommendations
4      #:param outlier_detection_method: percentile, zscore, iqr
5      #:return: a json file with resource recommendation
6
7      jsonOutPut = get_recommendation.get_recommendations("PATH", "Key-
   ↪ User-Id", 3, "percentile")
```

Listing 13: Usage example- preprocess and recommendations

```
1      {"Recommendations": {
2          "Ref_User_ID": "Key-User-Id",
3          "Recommendations": [
4              ["54tc274h...", 0.73],
5              ["e5vf5007...", 0.42],
6              ["b56ednkj...", 0.02]]
7      }}
```

Listing 14: Example JSON output

## RDM Visualizer (Processed Based)

The DA4RDM\_Vis\_ProcessBased is a Python package designed to calculate and determine RDM phases for a specified project ID according to user interactions of collective project members. The package ultimately offers a radar visualization. The package's repository contains a comprehensive README file providing detailed usage guidelines. The package is publicly accessible at: <https://pypi.org/project/DA4RDM-Vis-ProcessBased/>.

```
1      pip install DA4RDM_Vis_ProcessBased
```

Listing 15: Installation of the package

```
1      from DA4RDM_Vis_ProcessBased import Vizualize
```

Listing 16: Importing the method

```
1      #:param dataset_user_interactions: filepath to the csv ↵
      ↪ file
2      #:param project_id: the key project to be evaluated
3      #:param earliest_timestamp: the earliest timestamp to ↵
      ↪ consider for filtering records
4      #:param last_timestamp: the latest timestamp to consider ↵
      ↪ for filtering records
5      #:return: path to RDM visualization in png format
6
7      RdmPath = Visualize.process_vis("PATH", "Key-Project-
      ↪ Id", "timestamp-end", "timestamp-start")
```

Listing 17: Usage example- processing and return of the RDM visualization



# List of Publications

As the research presented in this dissertation has progressed, several preliminary results have been published in academic journals and conference proceedings. These publications have contributed to the scientific community’s knowledge and helped shape and refine the ideas and methodologies presented in this thesis. In addition, the valuable feedback and discussions from these publications and conferences have informed the evolution of the research presented herein.

This section lists several research papers that were published during the course of my Ph.D. studies:

- M. A. Yazdi, M. Politze, and M. Müller, “A Novel Approach to Outlining Research Data Management Life Cycle: A Case Study,” in *Proceedings of 10th International Conference on Information Management (ICIM)*, IEEE, 2024, [196]
- M. A. Yazdi, M. Politze, and M. Müller, “A Hybrid Event Log Acquisition Technique in Distributed Systems,” in *Proceedings of the Future Technologies Conference (FTC), Advances in Intelligent Systems and Computing*, Springer, 2023. DOI: 10.1007/978-3-031-47451-4\_23, [25]
- M. A. Yazdi, M. Politze, and B. Heinrichs, “Research Data Reusability with Content-Based Recommender System,” in *Proceedings of the 4th International Conference on Deep Learning Theory and Applications (DeLTA)*, Springer, 2023. DOI: 10.1007/978-3-031-39059-3\_10, [199]
- B. Heinrichs and M. A. Yazdi, “Determining the Similarity of Research Data by Using an Interoperable Metadata Extraction Method,” in *Proceedings of the 1st International Conference on Research Data Infrastructure (CoRDI)*, Open Science, 2023, [198]
- M. Politze, U. Christoph, B. Decker, *et al.*, “Supporting Software Development Processes for Academia with GitLab,” in *Proceedings of European University Information Systems Congress*, EasyChair, 2023. DOI: 10.29007/9157, [166]
- M. Müller, C. Terboven, M. Nellesen, *et al.*, *Combining HPC, AI and RDM: Challenges and Approaches*, International Conference on Society for Industrial and Applied Mathematics (SIAM), 2023. [Online]. Available: [https://meetings.siam.org/session/dsp\\_talk.cfm?p=123896](https://meetings.siam.org/session/dsp_talk.cfm?p=123896), [195]

- B. Heinrichs, M. Politze, and M. A. Yazdi, “Evaluation of Architectures for FAIR Data Management in a Research Data Management Use Case,” in *Proceedings of the 11th International Conference on Data Science, Technology and Applications (DATA)*, SciTePress, Jul. 11, 2022. DOI: 10.5220/0011302700003269. [Online]. Available: <https://publications.rwth-aachen.de/record/853932>, [162]
- M. A. Yazdi, D. Schimmel, M. Nellesen, *et al.*, “DA4RDM: Data Analysis for Research Data Management Systems,” in *Proceedings of the 13th International Conference on Knowledge Management and Information Systems (KMIS)*, SciTePress, 2021, pp. 177–183. DOI: 10.5220/0010678700003064. [Online]. Available: <https://publications.rwth-aachen.de/record/834785>, [167]
- M. A. Yazdi, P. Farhadi, and B. Heinrichs, “Event Log Abstraction in Client-Server Applications,” in *Proceedings of the 13th International Conference on Knowledge Discovery and Information Retrieval (KDIR)*, SciTePress, 2021, pp. 27–36. DOI: 10.5220/0010652000003064. [Online]. Available: <https://publications.rwth-aachen.de/record/834786>, [27]
- M. A. Yazdi and M. Politze, “Reverse Engineering: The University Distributed Services,” in *Proceedings of the Future Technologies Conference (FTC), Advances in Intelligent Systems and Computing*, vol. 2, Springer, 2020, pp. 223–238. DOI: 10.1007/978-3-030-63089-8\_14. [Online]. Available: <https://publications.rwth-aachen.de/record/807731>, [24]
- M. Politze, F. Claus, B. Brenger, *et al.*, “How to Manage IT Resources in Research Projects? Towards a Collaborative Scientific Integration Environment,” in *European University Information Systems (EUNIS)*, vol. 2, European Journal of Higher Education, 2020. DOI: 10.18154/RWTH-2020-11948. [Online]. Available: <https://publications.rwth-aachen.de/record/808269>, [21]
- M. A. Yazdi, “Enabling Operational Support in the Research Data Life Cycle,” in *Proceedings of the First International Conference on Process Mining (ICPM), Doctoral Consortium*, CEUR, 2019, pp. 1–10. [Online]. Available: <https://publications.rwth-aachen.de/record/794985>, [22]
- M. A. Yazdi, A. C. Valdez, L. Lichtschlag, *et al.*, “Visualizing Opportunities of Collaboration in Large Research Organizations,” in *International Conference on HCI in Business, Government and Organizations (HCIBGO)*, Springer, 2016, pp. 350–361. DOI: 10.1007/978-3-319-39396-4\_32. [Online]. Available: <https://publications.rwth-aachen.de/record/678527>, [26]
- A. C. Valdez, M. A. Yazdi, A. K. Schaar, *et al.*, “Orchestrating Collaboration using Visual Collaboration Suggestion for Steering of Research Clusters,” in *6th International Conference on Applied Human Factors and Ergonomics (AHFE)*, vol. 3, Elsevier, 2015, pp. 363–370. DOI: 10.1016/j.promfg.2015.07.176. [Online]. Available: <https://publications.rwth-aachen.de/record/571527>, [23]

# Statement of Originality

I, M. Amin Yazdi, confirm that the content of this Ph.D. thesis is original and was created solely by me, except where explicitly stated otherwise. This thesis reflects my intellectual efforts and research outcomes, conducted under the expert guidance of my supervisor, Prof. Matthias S. Müller. I guarantee that all sources used in this research and thesis preparation have been accurately cited and referenced. Any assistance received, be it in the form of data, ideas, or materials from other individuals, organizations, or published works, is appropriately acknowledged in the relevant sections of references and acknowledgments. Contributions made by others have been appropriately recognized and credited. Moreover, I declare that this thesis has not been submitted previously for any other degree or academic qualification.

The findings of each previously published paper have been carefully integrated and distributed throughout this dissertation, ensuring that relevant aspects of the papers contribute to the literature review, methodology, and/or results, thereby enriching the overall quality and depth of the presented research.

Furthermore, I would like to express my profound gratitude to my colleagues at the IT Center, particularly the members of the RPDm, PDSL, and HPC departments, for their invaluable expertise and assistance. Their collaborative efforts have been an essential component of the projects discussed in this dissertation. I am truly grateful for their support throughout my research journey.





# References

- [1] N. Kroes, “Setting up the european cloud partnership,” in *World Economic Forum*, vol. 1, 2012, p. 26. [Online]. Available: [http://www.eurocloud.fr/doc/Kroes\\_Davos\\_2012.pdf](http://www.eurocloud.fr/doc/Kroes_Davos_2012.pdf).
- [2] G. Mosconi, Q. Li, D. Randall, H. Karasti, P. Tolmie, J. Barutzky, M. Korn, and V. Pipek, “Three gaps in opening science,” *Computer Supported Cooperative Work (CSCW)*, vol. 28, no. 3, pp. 749–789, 2019. DOI: 10.1007/s10606-019-09354-z.
- [3] I. C. RWTH Aachen University, *Coscine*, Accessed: 01.04.2022, 2022. [Online]. Available: <https://coscine.de/>.
- [4] C. for Open Science, *Open science framework*, Accessed: 01.04.2022, 2022. [Online]. Available: <https://osf.io/>.
- [5] E. O. F. N. Research, *Zenodo*, Accessed: 01.04.2022, 2022. [Online]. Available: <https://zenodo.org/>.
- [6] A. Lefebvre, E. Schermerhorn, and M. Spruit, “How research data management can contribute to efficient and reliable science,” *ECIS 2018 Proceedings Collections*, 2018.
- [7] P. Ayris, J.-Y. Berthou, R. Bruce, S. Lindstaedt, A. Monreale, B. Mons, Y. Murayama, C. Södergård, K. Tochtermann, and R. Wilkinson, *Realising the European open science cloud*. European Union, 2016, ISBN: 978-92-79-61762-1. DOI: 10.2777/940154.
- [8] C. Heise and J. M. Pearce, “From open access to open science: The path from scientific reality to open scientific communication,” *SAGE open*, vol. 10, no. 2, p. 2158244020915900, 2020. DOI: 10.1177/2158244020915900.
- [9] P. Lord and A. Macdonald, *E-science curation, report data curation for, escience in the uk: An audit to establish requirements for future, curation and, provision*, 2015.
- [10] E. Commission, *Horizon 2020*, Accessed: 14.03.2022, EU, 2022. [Online]. Available: [https://ec.europa.eu/info/research-and-innovation/funding/funding-opportunities/funding-programmes-and-open-calls/horizon-2020\\_en](https://ec.europa.eu/info/research-and-innovation/funding/funding-opportunities/funding-programmes-and-open-calls/horizon-2020_en).
- [11] E. Commission, *Guidelines on fair data management in horizon 2020*, Accessed: 14.03.2022, EU, 2022. [Online]. Available: [https://ec.europa.eu/research/participants/data/ref/h2020/grants\\_manual/hi/oa\\_pilot/h2020-hi-oa-data-mgt\\_en.pdf](https://ec.europa.eu/research/participants/data/ref/h2020/grants_manual/hi/oa_pilot/h2020-hi-oa-data-mgt_en.pdf).

- [12] U. of Jena, *Data management plan*, Accessed: 14.03.2022, Jena University Website, 2022. [Online]. Available: <https://www.researchdata.uni-jena.de/en/information/data-management-plan>.
- [13] H. Karasti, K. S. Baker, and E. Halkola, "Enriching the notion of data curation in e-science: Data managing and information infrastructuring in the long term ecological research (lter) network," *Computer Supported Cooperative Work (CSCW)*, vol. 15, no. 4, pp. 321–358, 2006. DOI: 10.1007/s10606-006-9023-2.
- [14] I. Eberhard and W. Kraus, "Der elefant im raum. ethnographisches forschungsdatenmanagement als herausforderung für repositorien," *Mitteilungen der Vereinigung Österreichischer Bibliothekarinnen und Bibliothekare*, vol. 71, no. 1, pp. 41–52, 2018. DOI: 10.31263/voebm.v71i1.2018.
- [15] A. Treloar and C. Harboe-Ree, "Data management and the curation continuum: How the monash experience is informing repository relationships," 2008. DOI: 10.4225/03/5a16485de1981.
- [16] K. Kervin, R. B. Cook, and W. K. Michener, "The backstage work of data sharing," in *Proceedings of the 18th International Conference on Supporting Group Work*, 2014, pp. 152–156. DOI: 10.1145/2660398.2660406.
- [17] B. Rolland and C. P. Lee, "Beyond trust and reliability: Reusing data in collaborative cancer epidemiology research," in *Proceedings of the 2013 conference on Computer supported cooperative work*, 2013, pp. 435–444. DOI: 10.1145/2441776.2441826.
- [18] S. Gupta and C. Müller-Birn, "A study of e-research and its relation with research data life cycle: A literature perspective," *Benchmarking: An International Journal*, 2018. DOI: 10.1108/bij-02-2017-0030.
- [19] C. L. Borgman, "The conundrum of sharing research data," *Journal of the American Society for Information Science and Technology*, vol. 63, no. 6, pp. 1059–1078, 2012. DOI: 10.1002/asi.22634.
- [20] K. Eschenfelder and A. Johnson, "The limits of sharing: Controlled data collections," *Proceedings of the American Society for Information Science and Technology*, vol. 48, no. 1, pp. 1–10, 2011. DOI: 10.1002/meet.2011.14504801062.
- [21] M. Politze, F. Claus, B. Brenger, M. A. Yazdi, B. Heinrichs, and A. Schwarz, "How to Manage IT Resources in Research Projects? Towards a Collaborative Scientific Integration Environment," in *European University Information Systems (EUNIS)*, vol. 2, European Journal of Higher Education, 2020. DOI: 10.18154/RWTH-2020-11948. [Online]. Available: <https://publications.rwth-aachen.de/record/808269>.
- [22] M. A. Yazdi, "Enabling Operational Support in the Research Data Life Cycle," in *Proceedings of the First International Conference on Process Mining (ICPM), Doctoral Consortium*, CEUR, 2019, pp. 1–10. [Online]. Available: <https://publications.rwth-aachen.de/record/794985>.

- 
- [23] A. C. Valdez, M. A. Yazdi, A. K. Schaar, and M. Zieffle, "Orchestrating Collaboration using Visual Collaboration Suggestion for Steering of Research Clusters," in *6th International Conference on Applied Human Factors and Ergonomics (AHFE)*, vol. 3, Elsevier, 2015, pp. 363–370. DOI: 10.1016/j.promfg.2015.07.176. [Online]. Available: <https://publications.rwth-aachen.de/record/571527>.
- [24] M. A. Yazdi and M. Politze, "Reverse Engineering: The University Distributed Services," in *Proceedings of the Future Technologies Conference (FTC), Advances in Intelligent Systems and Computing*, vol. 2, Springer, 2020, pp. 223–238. DOI: 10.1007/978-3-030-63089-8\_14. [Online]. Available: <https://publications.rwth-aachen.de/record/807731>.
- [25] M. A. Yazdi, M. Politze, and M. Müller, "A Hybrid Event Log Acquisition Technique in Distributed Systems," in *Proceedings of the Future Technologies Conference (FTC), Advances in Intelligent Systems and Computing*, Springer, 2023. DOI: 10.1007/978-3-031-47451-4\_23.
- [26] M. A. Yazdi, A. C. Valdez, L. Lichtschlag, M. Zieffle, and J. Borchers, "Visualizing Opportunities of Collaboration in Large Research Organizations," in *International Conference on HCI in Business, Government and Organizations (HCIBGO)*, Springer, 2016, pp. 350–361. DOI: 10.1007/978-3-319-39396-4\_32. [Online]. Available: <https://publications.rwth-aachen.de/record/678527>.
- [27] M. A. Yazdi, P. Farhadi, and B. Heinrichs, "Event Log Abstraction in Client-Server Applications," in *Proceedings of the 13th International Conference on Knowledge Discovery and Information Retrieval (KDIR)*, SciTePress, 2021, pp. 27–36. DOI: 10.5220/0010652000003064. [Online]. Available: <https://publications.rwth-aachen.de/record/834786>.
- [28] M. Dreyer and A. Vollmer, "An integral approach to support research data management at the humboldt-universität zu berlin," *Y. Salmatzidis. Thessaloniki, Greece*, 2016. [Online]. Available: [https://www.eunis.org/wp-content/uploads/sites/8/2016/02/EUNIS2016\\_paper\\_66.pdf](https://www.eunis.org/wp-content/uploads/sites/8/2016/02/EUNIS2016_paper_66.pdf).
- [29] C. Tenopir, R. J. Sandusky, S. Allard, and B. Birch, "Research data management services in academic research libraries and perceptions of librarians," *Library & Information Science Research*, vol. 36, no. 2, pp. 84–90, 2014. DOI: 10.1016/j.lisr.2013.11.003.
- [30] A. M. Cox, M. A. Kennan, L. Lyon, and S. Pinfield, "Developments in research data management in academic libraries: Towards an understanding of research data service maturity," *Journal of the Association for Information Science and Technology*, vol. 68, no. 9, pp. 2182–2200, 2017. DOI: 10.1002/asi.23781.
- [31] M. Ashiq, M. H. Usmani, and M. Naeem, "A systematic literature review on research data management practices and services," *Global Knowledge, Memory and Communication*, 2020. DOI: 10.1108/gkmc-07-2020-0103.

- [32] A. M. Cox, M. A. Kennan, L. Lyon, S. Pinfield, and L. Sbaffi, “Maturing research data services and the transformation of academic libraries,” *Journal of Documentation*, 2019. DOI: 10.1108/jd-12-2018-0211.
- [33] H. Nie, P. Luo, and P. Fu, “Research data management implementation at peking university library: Foster and promote open science and open data,” *Data Intelligence*, vol. 3, no. 1, pp. 189–204, 2021. DOI: 10.1162/dint\_a\_00088.
- [34] W. W.-U. Münster, *Sciebo*, Accessed: 01.04.2022, 2022. [Online]. Available: <https://hochschulcloud.nrw/>.
- [35] I. Dropbox, *Dropbox*, Accessed: 01.04.2022, 2022. [Online]. Available: <https://www.dropbox.com/>.
- [36] A. Inc., *Google drive*, Accessed: 01.04.2022, 2022. [Online]. Available: <https://www.google.com/drive/>.
- [37] G. Inc, *Gitlab*, Accessed: 01.04.2022, 2022. [Online]. Available: <https://about.gitlab.com/>.
- [38] C. Curdt, “Design and implementation of a research data management system: The crc/tr32 project database (tr32db),” Ph.D. dissertation, Universität zu Köln, 2014. [Online]. Available: <http://kups.ub.uni-koeln.de/id/eprint/5882>.
- [39] M. Politze, A. Schwarz, S. Kirchmeyer, F. Claus, and M. S. Müller, “Kollaborative forschungsunterstützung: Ein integriertes probenmanagement,” *Tage 2019*, p. 58, 2019. DOI: 10.11588/heibooks.598.c8417.
- [40] T. Kirsten, A. Kiel, J. Wagner, M. Rühle, and M. Löffler, “Selecting, packaging, and granting access for sharing study data,” *INFORMATIK 2017*, 2017. DOI: 10.18420/in2017\_138.
- [41] G. F. Coulouris, J. Dollimore, and T. Kindberg, *Distributed systems: concepts and design*. pearson education, 2005, ISBN: 0-201-62433-8.
- [42] R. Bolton, S. Campana, A. Ceccanti, X. Espinal, A. Fkias, P. Fuhrmann, and Y. Grange, “Escape prototypes a data infrastructure for open science,” in *EPJ Web of Conferences*, EDP Sciences, vol. 245, 2020, p. 04 019. DOI: 10.1051/epjconf/202024504019.
- [43] M. Herschel, R. Diestelkämper, and H. Ben Lahmar, “A survey on provenance: What for? what form? what from?” *The VLDB Journal*, vol. 26, no. 6, pp. 881–906, 2017. DOI: 10.1007/s00778-017-0486-1.
- [44] Z. B. Mufti and M. Elkhodr, “Data provenance in the internet of things: Views and challenges,” *CSEN, NCWC*, pp. 1–7, 2018. DOI: 10.5121/csit.2018.81201.
- [45] R. Hu, Z. Yan, W. Ding, and L. T. Yang, “A survey on data provenance in iot,” *World Wide Web*, vol. 23, no. 2, pp. 1441–1463, 2020. DOI: 10.1007/s11280-019-00746-1.
- [46] E. D. Foster and A. Deardorff, “Open science framework (osf),” *Journal of the Medical Library Association: JMLA*, vol. 105, no. 2, p. 203, 2017. DOI: 10.5195/jmla.2017.88.

- 
- [47] C. for Open Science, *Waterbutler*, Accessed: 01.04.2022, 2022. [Online]. Available: <https://waterbutler.readthedocs.io>.
- [48] D. Schmitz and M. Politze, "Forschungsdaten managen—bausteine für eine dezentrale, forschungsnahe unterstützung," *o-bib. Das offene Bibliotheksjournal/Herausgeber VDB*, vol. 5, no. 3, pp. 76–91, 2018. DOI: 10.5282/o-bib/2018H3S76-91.
- [49] W. Smith, T. Moyer, and C. Munson, "Curator: Provenance management for modern distributed systems," in *10th USENIX Workshop on the Theory and Practice of Provenance (TaPP 2018)*, 2018. DOI: 10.5555/3319379.3319387.
- [50] E. Stephan, B. Raju, T. Elsethagen, L. Pouchard, and C. Gamboa, "A scientific data provenance harvester for distributed applications," in *2017 New York Scientific Data Summit (NYSDS)*, IEEE, 2017, pp. 1–9. DOI: 10.1109/nysds.2017.8085041.
- [51] D. Donoho, "50 years of data science," *Journal of Computational and Graphical Statistics*, vol. 26, no. 4, pp. 745–766, 2017. DOI: 10.1080/10618600.2017.1384734.
- [52] G. Press, "A very short history of data science," *Forbes. com*, 2013.
- [53] F. Provost and T. Fawcett, "Data science and its relationship to big data and data-driven decision making," *Big data*, vol. 1, no. 1, pp. 51–59, 2013. DOI: 10.1089/big.2013.1508.
- [54] W. van der Aalst, *Process mining: data science in action*. Springer, 2016. DOI: 10.1007/978-3-662-49851-4.
- [55] W. van der Aalst, *Process Mining Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011, vol. 2. DOI: 10.1007/978-3-642-19345-3.
- [56] W. Van der Aalst, T. Weijters, and L. Maruster, "Workflow mining: Discovering process models from event logs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 9, pp. 1128–1142, 2004. DOI: 10.1109/tkde.2004.47.
- [57] W. M. Van der Aalst, A. A. De Medeiros, and A. Weijters, "Genetic process mining," in *International conference on application and theory of petri nets*, Springer, 2005, pp. 48–69. DOI: 10.1007/11494744\_5.
- [58] J. C. Buijs, B. F. Van Dongen, and W. M. van Der Aalst, "On the role of fitness, precision, generalization and simplicity in process discovery," in *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*, Springer, 2012, pp. 305–322. DOI: 10.1007/978-3-642-33606-5\_19.
- [59] L. Cheng, B. F. van Dongen, and W. M. van der Aalst, "Efficient event correlation over distributed systems," in *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, IEEE Press, 2017, pp. 1–10. DOI: 10.1109/ccgrid.2017.94.
- [60] M. Kumar, L. Thomas, and B. Annappa, "Distilling lasagna from spaghetti processes," in *Proceedings of the 2017 International Conference on Intelligent*

- Systems, Metaheuristics & Swarm Intelligence*, ACM, 2017, pp. 157–161. DOI: 10.1145/3059336.3059362.
- [61] R. Agrawal, D. Gunopulos, and F. Leymann, “Mining process models from workflow logs,” in *International Conference on Extending Database Technology*, Springer, 1998, pp. 467–483. DOI: 10.1007/bfb0101003.
  - [62] W. van der Aalst and C. W. Gunther, “Finding structure in unstructured processes: The case for process mining,” in *Application of concurrency to system design, 2007. ACSD 2007. Seventh international conference on*, IEEE, 2007, pp. 3–12. DOI: 10.1109/acsd.2007.50.
  - [63] P. N. Edwards, S. J. Jackson, M. K. Chalmers, G. C. Bowker, C. L. Borgman, D. Ribes, M. Burton, and S. Calvert, “Knowledge infrastructures: Intellectual frameworks and research challenges,” 2013. [Online]. Available: <https://escholarship.org/uc/item/2mt6j2mh>.
  - [64] M. Jirotko, C. P. Lee, and G. M. Olson, “Supporting scientific collaboration: Methods, tools and concepts,” *Computer Supported Cooperative Work (CSCW)*, vol. 22, pp. 667–715, 2013. DOI: 10.1007/s10606-012-9184-0.
  - [65] M. J. Bietz, E. P. Baumer, and C. P. Lee, “Synergizing in cyberinfrastructure development,” *Computer Supported Cooperative Work (CSCW)*, vol. 19, pp. 245–281, 2010. DOI: 10.1007/s10606-010-9114-y.
  - [66] A. C. Valdez, S. Bruns, C. Greven, M. Zieffle, and U. Schroeder, “What should i read next? a personalized visual publication recommender system,” in *International Conference on Human Interface and the Management of Information*, Springer, 2015, pp. 89–100. DOI: 10.1007/978-3-319-20618-9\_9.
  - [67] M. Rafiei, L. von Waldthausen, and W. M. van der Aalst, “Ensuring confidentiality in process mining,” *Simpda*, vol. 18, pp. 3–17, 2018.
  - [68] L. Perrier and L. Barnes, “Developing research data management services and support for researchers: A mixed methods study,” *Partnership: The Canadian Journal of Library and Information Practice and Research*, vol. 13, no. 1, 2018. DOI: 10.21083/partnership.v13i1.4115.
  - [69] S. Kokolakis, “Privacy attitudes and privacy behaviour: A review of current research on the privacy paradox phenomenon,” *Computers & Security*, vol. 64, pp. 122–134, 2017. DOI: 10.1016/j.cose.2015.07.002.
  - [70] Y. Kim and M. Adler, “Social scientists data sharing behaviors: Investigating the roles of individual motivations, institutional pressures, and data repositories,” *International Journal of Information Management*, vol. 35, no. 4, pp. 408–418, 2015. DOI: 10.1016/j.ijinfomgt.2015.04.007.
  - [71] D. S. Sayogo and T. A. Pardo, “Exploring the determinants of scientific data sharing: Understanding the motivation to publish research data,” *Government Information Quarterly*, vol. 30, S19–S31, 2013. DOI: 10.1016/j.giq.2012.06.011.

- 
- [72] E. Union, *General data protection regulation*, Accessed: 01.04.2022, 2022. [Online]. Available: <https://gdpr-info.eu/>.
- [73] I. Labastida, "Legal requirements, rdm and open data," in *Leaders Activating Research Networks (LEARN)*, 2017. DOI: 10.14324/000.learn.22.
- [74] Y. Sedelmaier and D. Landes, "How can we find out what makes a good requirements engineer in the age of digitalization?" In *2017 IEEE Global Engineering Education Conference (EDUCON)*, IEEE, 2017, pp. 230–238. DOI: 10.1109/educon.2017.7942853.
- [75] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, *Non-functional requirements in software engineering*. Springer Science & Business Media, 2012, vol. 5, ISBN: 978-3-642-02462-7. DOI: 10.1007/978-3-642-02463-4\_19.
- [76] W. Maalej, M. Nayebi, T. Johann, and G. Ruhe, "Toward data-driven requirements engineering," *IEEE software*, vol. 33, no. 1, pp. 48–54, 2015. DOI: 10.1109/ms.2015.153.
- [77] A. Hemmati, S. D. Al Alam, and C. Carlson, "Utilizing product usage data for requirements evaluation," in *2018 IEEE 26th International Requirements Engineering Conference (RE)*, IEEE, 2018, pp. 432–435. DOI: 10.1109/re.2018.00056.
- [78] N. El Moukhi, I. El Azami, A. Mouloudi, and A. El Mounadi, "Requirements-driven modeling for decision-making systems," in *2018 International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS)*, IEEE, 2018, pp. 1–7. DOI: 10.1109/icecocs.2018.8610523.
- [79] N. G. Mohammadi and M. Heisel, "A framework for systematic refinement of trustworthiness requirements," *Information*, vol. 8, no. 2, p. 46, 2017. DOI: 10.3390/info8020046.
- [80] J. Dbrowski, F. M. Kifetew, D. Muñante, E. Letier, A. Siena, and A. Susi, "Discovering requirements through goal-driven process mining," in *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*, IEEE, 2017, pp. 199–203. DOI: 10.1109/rew.2017.61.
- [81] A. Filipowska, P. Kauny, and M. Skrzypek, "Improving user experience in e-commerce by application of process mining techniques," *Zeszyty Naukowe Politechniki Czstochowskiej Zarzdzanie*, vol. 33, no. 1, pp. 30–40, 2019. DOI: 10.17512/znpcz.2019.1.03.
- [82] D. Forschungsgemeinschaft, *Gute wissenschaftliche praxis*, Accessed: 15.06.2022, 2022. [Online]. Available: [https://www.dfg.de/foerderung/grundlagen\\_rahmenbedingungen/gwp/index.html](https://www.dfg.de/foerderung/grundlagen_rahmenbedingungen/gwp/index.html).
- [83] A. M. Ketchum, "The research life cycle and the health sciences librarian: Responding to change in scholarly communication," *Journal of the Medical Library Association: JMLA*, vol. 105, no. 1, p. 80, 2017. DOI: 10.5195/jmla.2017.110.
- [84] C. Humphrey, "E-science and the life cycle of research," 2006.

- [85] E. V. Luneva, "Key performance indicators (kpi) system in education," *Asian Social Science*, vol. 11, no. 8, p. 194, 2015. DOI: 10.5539/ass.v11n8p194.
- [86] U. FORCE11 Initiative, *The fair data principles*, Accessed: 01.04.2022, 2022. [Online]. Available: <https://force11.org/info/the-fair-data-principles/>.
- [87] I. O. Data and I. Exchange, *Iode*, Accessed: 01.04.2022, 2022. [Online]. Available: <https://www.iode.org/>.
- [88] E. O. S. Cloud, *Eosc*, Accessed: 01.04.2022, 2022. [Online]. Available: <https://eosc-portal.eu/>.
- [89] F. Berman and M. Crosas, "The research data alliance: Benefits and challenges of building a community organization," 2020. DOI: 10.1162/99608f92.5e126552.
- [90] N. Hartl, E. Wössner, and Y. Sure-Vetter, "Nationale forschungsdateninfrastruktur (nfdi)," *Informatik Spektrum*, vol. 44, no. 5, pp. 370–373, 2021. DOI: 10.1007/s00287-021-01392-6.
- [91] R. Higman, D. Bangert, and S. Jones, "Three camps, one destination: The intersections of research data management, fair and open," *Insights*, vol. 32, no. 1, 2019. DOI: 10.1629/uksg.468.
- [92] T. Tanhua, S. Pouliquen, J. Hausman, K. Obrien, P. Bricher, T. De Bruin, J. J. Buck, E. F. Burger, T. Carval, K. S. Casey, *et al.*, "Ocean fair data services," *Frontiers in Marine Science*, vol. 6, p. 440, 2019. DOI: 10.3389/fmars.2019.00440.
- [93] P. Wittenburg, "From persistent identifiers to digital objects to make data science more efficient," *Data intelligence*, vol. 1, no. 1, pp. 6–21, 2019. DOI: 10.1162/dint\_a\_00004.
- [94] K. De Smedt, D. Koureas, and P. Wittenburg, "Fair digital objects for science: From data pieces to actionable knowledge units," *Publications*, vol. 8, no. 2, p. 21, 2020. DOI: 10.3390/publications8020021.
- [95] S. Corral, "Designing libraries for research collaboration in the network world: An exploratory study," *Liber Quarterly*, vol. 24, no. 1, 2014. DOI: 10.18352/lq.9525.
- [96] A. M. Cox and W. W. T. Tam, "A critical analysis of lifecycle models of the research process and research data management," *Aslib Journal of Information Management*, 2018. DOI: 10.1108/ajim-11-2017-0251.
- [97] A. Surkis and K. Read, "Research data management," *Journal of the Medical Library Association: JMLA*, vol. 103, no. 3, p. 154, 2015. DOI: 10.3163/1536-5050.103.3.011.
- [98] I. Souilah, "Provenance in distributed systems: A process algebraic study of provenance management and its role in establishing trust in data quality," Ph.D. dissertation, University of Southampton, 2013. [Online]. Available: <http://eprints.soton.ac.uk/id/eprint/353288>.
- [99] R. Tang, Z. Hu, N. Henry, and A. Thomas, "A usability evaluation of research data management librarian academy (rdmla): Examining the impact of learner



- 
- differences in pedagogical usability,” *Journal of Web Librarianship*, vol. 15, no. 3, pp. 154–193, 2021. DOI: 10.1080/19322909.2021.1937442.
- [100] B. Raju, T. Elsethagen, E. Stephan, and K. K. Van Dam, “A scientific data provenance api for distributed applications,” in *2016 International Conference on Collaboration Technologies and Systems (CTS)*, IEEE, 2016, pp. 104–111. DOI: 10.1109/cts.2016.0036.
- [101] J. Bacon and K. Moody, “Toward open, secure, widely distributed services,” *Communications of the ACM*, vol. 45, no. 6, pp. 59–64, 2002. DOI: 10.1145/508448.508475.
- [102] L. C. Briand, Y. Labiche, and J. Leduc, “Toward the reverse engineering of uml sequence diagrams for distributed java software,” *IEEE Transactions on Software Engineering*, vol. 32, no. 9, pp. 642–663, 2006. DOI: 10.1109/tse.2006.96.
- [103] A. van Hoorn, M. Rohr, W. Hasselbring, J. Waller, J. Ehlers, S. Frey, and D. Kieselhorst, “Continuous monitoring of software services: Design and application of the kieker framework,” 2009, ISSN: 2192-6247. [Online]. Available: [https://macau.uni-kiel.de/receive/macau\\_mods\\_00001840](https://macau.uni-kiel.de/receive/macau_mods_00001840).
- [104] M. Leemans and W. M. van der Aalst, “Process mining in software systems: Discovering real-life business transactions and process models from distributed systems,” in *2015 ACM/IEEE 18th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, Ieee, 2015, pp. 44–53. DOI: 10.1109/models.2015.7338234.
- [105] I. Beschastnikh, Y. Brun, M. D. Ernst, and A. Krishnamurthy, “Inferring models of concurrent systems from logs of their behavior with csight,” in *Proceedings of the 36th International Conference on Software Engineering*, 2014, pp. 468–479. DOI: 10.1145/2568225.2568246.
- [106] C. Ackermann, M. Lindvall, and R. Cleaveland, “Recovering views of inter-system interaction behaviors,” in *2009 16th Working Conference on Reverse Engineering*, Ieee, 2009, pp. 53–61. DOI: 10.1109/wcre.2009.34.
- [107] J. Moe and D. A. Carr, “Using execution trace data to improve distributed systems,” *Software: Practice and Experience*, vol. 32, no. 9, pp. 889–906, 2002. DOI: 10.1002/spe.466.
- [108] M. Salah and S. Mancoridis, “Toward an environment for comprehending distributed systems,” in *WCRE*, 2003, pp. 238–247, ISBN: 0769520278. DOI: 10.5555/950792.951369.
- [109] W. van der Aalst, A. Adriansyah, and B. van Dongen, “Replaying history on process models for conformance checking and performance analysis,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 182–192, 2012. DOI: 10.1002/widm.1045.
- [110] B. F. Van Dongen, A. K. A. de Medeiros, H. Verbeek, A. Weijters, and W. M. van Der Aalst, “The prom framework: A new era in process mining tool

- support,” in *International conference on application and theory of petri nets*, Springer, 2005, pp. 444–454. DOI: 10.1007/11494744\_25.
- [111] C. W. Günther and A. Rozinat, “Disco: Discover your processes.,” *BPM (Demos)*, vol. 940, pp. 40–44, 2012.
- [112] R. Mans, W. M. van der Aalst, and H. Verbeek, “Supporting process mining workflows with rapidprom.,” *BPM (Demos)*, vol. 56, 2014. DOI: 10.1007/978-3-319-15895-2.
- [113] S. J. van Zelst, F. Mannhardt, M. de Leoni, and A. Koschmider, “Event abstraction in process mining: Literature review and taxonomy,” *Granular Computing*, pp. 1–18, 2020. DOI: 10.1007/s41066-020-00226-2.
- [114] S. Smirnov, H. A. Reijers, and M. Weske, “From fine-grained to abstract process models: A semantic approach,” *Information Systems*, vol. 37, no. 8, pp. 784–797, 2012. DOI: 10.1016/j.is.2012.05.007.
- [115] F. Mannhardt and N. Tax, “Unsupervised event abstraction using pattern abstraction and local process models,” *arXiv preprint arXiv:1704.03520*, 2017. DOI: 10.48550/arXiv.1704.03520.
- [116] M. de Leoni and S. Dünder, “Event-log abstraction using batch session identification and clustering,” in *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, 2020, pp. 36–44. DOI: 10.1145/3341105.3373861.
- [117] A. K. Begicheva and I. A. Lomazova, “Discovering high-level process models from event logs,” *Modeling and Analysis of Information Systems*, vol. 24, no. 2, pp. 125–140, 2017. DOI: 10.18255/1818-1015-2017-2-125-140.
- [118] F. Mannhardt, M. De Leoni, H. A. Reijers, W. M. Van Der Aalst, and P. J. Toussaint, “From low-level events to activities-a pattern-based approach,” in *International conference on business process management*, Springer, 2016, pp. 125–141. DOI: 10.1007/978-3-319-45348-4\_8.
- [119] N. Tax, N. Sidorova, R. Haakma, and W. M. van der Aalst, “Event abstraction for process mining using supervised learning techniques,” in *Proceedings of SAI Intelligent Systems Conference*, Springer, 2016, pp. 251–269. DOI: 10.1007/978-3-319-56994-9\_18.
- [120] C. Liu, S. Wang, S. Gao, F. Zhang, and J. Cheng, “User behavior discovery from low-level software execution log,” *IEEE Transactions on Electrical and Electronic Engineering*, vol. 13, no. 11, pp. 1624–1632, 2018. DOI: 10.1002/tee.22727.
- [121] S. J. Leemans, D. Fahland, and W. M. Van Der Aalst, “Process and deviation exploration with inductive visual miner.,” *BPM (Demos)*, vol. 1295, no. 46, p. 8, 2014. DOI: 10.1007/978-3-319-15895-2.
- [122] P. Ceravolo, A. Azzini, M. Angelini, T. Catarci, P. Cudré-Mauroux, E. Damiani, A. Mazak, M. Van Keulen, M. Jarrar, G. Santucci, *et al.*, “Big data semantics,” *Journal on Data Semantics*, vol. 7, no. 2, pp. 65–85, 2018. DOI: 10.1007/s13740-018-0086-2.

- 
- [123] J. Makani, “Knowledge management, research data management, and university scholarship: Towards an integrated institutional research data management support-system framework,” *Vine*, 2015. DOI: 10.1108/vine-07-2014-0047.
- [124] X. Specka, P. Gärtner, C. Hoffmann, N. Svoboda, M. Stecker, U. Einspanier, K. Senkler, M. M. Zoorder, and U. Heinrich, “The bonares metadata schema for geospatial soil-agricultural research data—merging inspire and datacite metadata schemes,” *Computers & Geosciences*, vol. 132, pp. 33–41, 2019. DOI: 10.1016/j.cageo.2019.07.005.
- [125] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, R. Wirth, *et al.*, “Crisp-dm 1.0: Step-by-step data mining guide,” *SPSS inc*, vol. 9, p. 13, 2000.
- [126] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne, *et al.*, “The fair guiding principles for scientific data management and stewardship,” *Scientific data*, vol. 3, no. 1, pp. 1–9, 2016. DOI: 10.1038/sdata.2016.18.
- [127] L. Hofeditz, B. Ross, K. Wilms, M. Rother, S. Rehwald, B. Brenger, A. López, R. Vogl, and D. Rudolph, “How to design a research data management platform? technical, organizational and individual perspectives and their relations,” in *International Conference on Human-Computer Interaction*, Springer, 2020, pp. 324–337. DOI: 10.1007/978-3-030-50017-7\_23.
- [128] RDMO Development Team, *Rdmo*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://rdmorganiser.github.io/>.
- [129] simplearchive Development Team, *Simplearchive*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://app.rwth-aachen.de/simplearchive/>.
- [130] Metadata Manager Development Team, *Metadata manager*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://app.rwth-aachen.de/metadata/>.
- [131] Coscine Development Team, *Coscine*, Accessed: 2022-06-09, European Journal of Higher Education IT, 2022. [Online]. Available: <https://coscine.de/>.
- [132] Zenodo Development Team, *Zenodo*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://zenodo.org/>.
- [133] OSF Development Team, *Osfi*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://osf.io/>.
- [134] Sciebo-RDS Development Team, *Sciebo-rds*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://www.research-data-services.org/>.
- [135] eLabFTW Development Team, *Elabftw*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://www.elabftw.net/>.
- [136] CKAN Development Team, *Ckan*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://ckan.org/>.
- [137] Nextcloud Development Team, *Nextcloud*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://nextcloud.com/>.

- [138] Dataverse Development Team, *Dataverse*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://dataverse.org/>.
- [139] EUDAT CDI Development Team, *Eudat cdi*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://www.eudat.eu/>.
- [140] DSpace Development Team, *Dspace*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://dspace.org/>.
- [141] Figshare Development Team, *Figshare*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://figshare.com/>.
- [142] GitLab Development Team, *Gitlab*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://gitlab.com/>.
- [143] A. Zuiderwijk, R. Shinde, and W. Jeng, “What drives and inhibits researchers to share and use open research data? a systematic literature review to analyze factors influencing open research data adoption,” *PloS one*, vol. 15, no. 9, e0239283, 2020. DOI: 10.1371/journal.pone.0239283.
- [144] C. Brindescu, M. Codoban, S. Shmarkatiuk, and D. Dig, “How do centralized and distributed version control systems impact software changes?” In *Proceedings of the 36th international conference on Software Engineering*, 2014, pp. 322–333. DOI: 10.1145/2568225.2568322.
- [145] B. De Alwis and J. Sillito, “Why are software projects moving from centralized to decentralized version control systems?” In *2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering*, IEEE, 2009, pp. 36–39. DOI: 10.1109/chase.2009.5071408.
- [146] B. Mons, *Data stewardship for open science: Implementing FAIR principles*. Chapman and Hall/CRC, 2018. DOI: 10.1201/9781315380711.
- [147] J. Parland-von Essen, K. Fält, Z. Maalick, M. Alonen, and E. Gonzalez, “Supporting fair data: Categorization of research data as a tool in data management,” *Informaatitutkimus*, vol. 37, no. 4, 2018. DOI: 10.23978/inf.76084.
- [148] International DOI Foundation, *Doi resolver*, Accessed: 2022-06-09, 2022. [Online]. Available: <http://dx.doi.org/>.
- [149] I. Wolff, D. Broneske, and V. Köppen, “Fair research data management for learning analytics,” in *Proceedings of DELFI Workshop*, 2021, pp. 158–163.
- [150] R. Cyganiak, D. Wood, M. Lanthaler, G. Klyne, J. Carroll, and B. McBride, “Rdf 1.1 concepts and abstract syntax, february 2014,” *URL https://www.w3.org/TR/rdf11-concepts*, 2018.
- [151] *Meilisearch*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://www.meilisearch.com/>.
- [152] Apache Software Foundation, *Apache solr*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://solr.apache.org/>.

- 
- [153] Elasticsearch B.V., *Elasticsearch*, Accessed: 2022-06-09, 2022. [Online]. Available: <https://www.elastic.co/>.
  - [154] E. Dell, “Ecs overview and architecture,” *A Dell EMC Technical Whitepaper*, 2018.
  - [155] B. Leiba, “Oauth web authorization protocol,” *IEEE Internet Computing*, vol. 16, no. 1, pp. 74–77, 2012. DOI: 10.1109/mic.2012.11.
  - [156] V. Koutsonikola and A. Vakali, “Ldap: Framework, practices, and trends,” *IEEE Internet Computing*, vol. 8, no. 5, pp. 66–72, 2004. DOI: 10.1109/mic.2004.44.
  - [157] A. Armando, R. Carbone, L. Compagna, J. Cuellar, and L. Tobarra, “Formal analysis of saml 2.0 web browser single sign-on: Breaking the saml-based single sign-on for google apps,” in *Proceedings of the 6th ACM workshop on Formal methods in security engineering*, 2008, pp. 1–10. DOI: 10.1145/1456396.1456397.
  - [158] D. Patel, “Research data management: A conceptual framework,” *Library review*, 2016. DOI: 10.1108/lr-01-2016-0001.
  - [159] K. Gierend, F. Krüger, D. Waltemath, M. Fünfgeld, T. Ganslandt, A. A. Zeleke, *et al.*, “Approaches and criteria for provenance in biomedical data sets and workflows: Protocol for a scoping review,” *JMIR Research Protocols*, vol. 10, no. 11, e31750, 2021. DOI: 10.2196/31750.
  - [160] T. Lebo, S. Sahoo, D. McGuinness, K. Belhajjame, J. Cheney, D. Corsar, D. Garijo, S. Soiland-Reyes, S. Zednik, and J. Zhao, “Prov-o: The prov ontology,” 2013. [Online]. Available: <http://www.w3.org/TR/2013/REC-prov-o-20130430/>.
  - [161] A. Latif, F. Limani, and K. Tochtermann, “On the complexities of federating research data infrastructures,” *Data Intelligence*, vol. 3, no. 1, pp. 79–87, 2021. DOI: 10.1162/dint\_a\_00080.
  - [162] B. Heinrichs, M. Politze, and M. A. Yazdi, “Evaluation of Architectures for FAIR Data Management in a Research Data Management Use Case,” in *Proceedings of the 11th International Conference on Data Science, Technology and Applications (DATA)*, SciTePress, Jul. 11, 2022. DOI: 10.5220/0011302700003269. [Online]. Available: <https://publications.rwth-aachen.de/record/853932>.
  - [163] M. Politze, B. Decker, and T. Eifert, “Pstaix-a process-aware architecture to support research processes,” *INFORMATIK 2017*, 2017. DOI: 10.18420/in2017\_137.
  - [164] G. Wiedherhold, “Mediators in the architecture of future information system,” *IEEE Computer*, vol. 25, pp. 38–49, 1992. DOI: 10.1109/2.121508.
  - [165] M. Politze, “A reference architecture and implementation enabling data protection in distributed elearning and escience processes; 1. Auflage,” Druckausgabe: 2019. - Auch veröffentlicht auf dem Publikationsserver der RWTH Aachen University 2020; Dissertation, RWTH Aachen University, 2019, Dissertation, RWTH Aachen University, Aachen, 2019, 1 Online-Ressource (xviii, 232 Seiten) : Illustrationen, Diagramme, ISBN: 978-3-86359-795-5. DOI: 10.18154/RWTH-2019-11476. [Online]. Available: <https://publications.rwth-aachen.de/record/774326>.

- [166] M. Politze, U. Christoph, B. Decker, P. Hristov, I. Lang, M. Nellesen, and M. A. Yazdi, "Supporting Software Development Processes for Academia with GitLab," in *Proceedings of European University Information Systems Congress*, EasyChair, 2023. DOI: 10.29007/9157.
- [167] M. A. Yazdi, D. Schimmel, M. Nellesen, M. Politze, and M. Müller, "DA4RDM: Data Analysis for Research Data Management Systems," in *Proceedings of the 13th International Conference on Knowledge Management and Information Systems (KMIS)*, SciTePress, 2021, pp. 177–183. DOI: 10.5220/0010678700003064. [Online]. Available: <https://publications.rwth-aachen.de/record/834785>.
- [168] S. Drenckberg, "Central Logging System for User Behaviour Detection," Tech. Rep., Feb. 2019. DOI: 10.5281/zenodo.7865228.
- [169] J. Treadway, M. Hahnel, S. Leonelli, D. Penny, D. Groenewegen, N. Miyairi, K. Hayashi, D. O'Donnell, D. Hook, *et al.*, "The state of open data report," 2016. DOI: 10.6084/m9.figshare.4036398.
- [170] O. Ålander *et al.*, "A descriptive analysis of research data management practices in european educational institutions," 2018. [Online]. Available: <http://urn.fi/URN:NBN:fi:aalto-201806273621>.
- [171] J. C. Teitelbaum and K. Zeiler, *Research handbook on behavioral law and economics*. Edward Elgar Publishing, 2018, ISBN: 9781849805674.
- [172] B. Mons, C. Neylon, J. Velterop, M. Dumontier, L. O. B. da Silva Santos, and M. D. Wilkinson, "Cloudy, increasingly fair; revisiting the fair data guiding principles for the european open science cloud," *Information services & use*, vol. 37, no. 1, pp. 49–56, 2017. DOI: 10.3233/ISU-170824.
- [173] E. Curry, *Real-time linked dataspace: Enabling data ecosystems for intelligent systems*. Springer Nature, 2020. DOI: 10.1007/978-3-030-29665-0.
- [174] A. Freitas, S. O'Riáin, and E. Curry, "Querying and searching heterogeneous knowledge graphs in real-time linked dataspace," in *Real-time Linked Dataspace*, Springer, 2020, pp. 105–124. DOI: 10.1007/978-3-030-29665-0\_7.
- [175] R. J. C. Bose, R. S. Mans, and W. M. van der Aalst, "Wanna improve process mining results? its high time we consider data quality issues seriously," in *2013 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, 2013, pp. 127–134. DOI: 10.1109/CIDM.2013.6597227.
- [176] R. Andrews, S. Suriadi, C. Ouyang, and E. Poppe, "Towards event log querying for data quality," in *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*, Springer, 2018, pp. 116–134. DOI: 10.1007/978-3-030-02610-3\_7.
- [177] W. v. d. Aalst, A. Adriansyah, A. K. A. d. Medeiros, F. Arcieri, T. Baier, T. Blickle, J. C. Bose, P. v. d. Brand, R. Brandtjen, J. Buijs, *et al.*, "Process mining manifesto. iee task force on process mining.," in *International conference on business process management*, Springer, 2011, pp. 169–194.

- 
- [178] R. J. C. Bose, W. M. Van Der Aalst, I. Iobait, and M. Pechenizkiy, "Dealing with concept drifts in process mining," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 1, pp. 154–171, 2013. DOI: 10.1109/TNNLS.2013.2278313.
- [179] S. Suriadi, R. Andrews, A. H. ter Hofstede, and M. T. Wynn, "Event log imperfection patterns for process mining: Towards a systematic approach to cleaning event logs," *Information systems*, vol. 64, pp. 132–150, 2017. DOI: 10.1016/j.is.2016.07.011.
- [180] R. van Cruchten, "Data quality in process mining: A rule-based approach," *Interactive Process Mining in Healthcare*, pp. 53–79, 2020.
- [181] D. Hardt, "The oauth 2.0 authorization framework," Tech. Rep., 2012.
- [182] M. Politze and B. Decker, "Extending the OAuth2 workflow to audit data usage for users and service providers in a cooperative scenario," in *DFN-Forum Kommunikationstechnologien*, 2017, pp. 41–50. DOI: 10.18154/RWTH-2017-04864.
- [183] G. T. Lakshmanan and R. Khalaf, "Leveraging process-mining techniques," *IT Professional*, vol. 15, no. 5, pp. 22–30, 2012. DOI: 10.1109/MITP.2012.88.
- [184] M. Sahlabadi, R. C. Muniyandi, and Z. Shukur, "Detecting abnormal behavior in social network websites by using a process mining technique," *Journal of Computer Science*, vol. 10, no. 3, p. 393, 2014. DOI: 10.3844/jcssp.2014.393.402.
- [185] M. Politze, "Extending OAuth2 to join local services into a federative SOA," in *EUNIS 23rd annual congress ; shaping the digital future of Universities*, 2017, pp. 124–132. DOI: 10.17879/21299722960. [Online]. Available: <https://publications.rwth-aachen.de/record/690941>.
- [186] W. Yang, "Ideal Process Abstraction Methods for Heaps of User Activities in Distributed Services," Tech. Rep., Jan. 2020. DOI: 10.5281/zenodo.7865267.
- [187] J. Benesty, J. Chen, Y. Huang, and I. Cohen, "Pearson correlation coefficient," in *Noise reduction in speech processing*, Springer, 2009, pp. 1–4. DOI: 10.1007/978-3-642-00296-0\_5.
- [188] T. Jouck, A. Bolt, B. Depaire, M. de Leoni, and W. M. van der Aalst, "An integrated framework for process discovery algorithm evaluation," *arXiv preprint arXiv:1806.07222*, 2018.
- [189] T. Nind, J. Galloway, G. McAllister, D. Scobbie, W. Bonney, C. Hall, L. Tramma, P. Reel, M. Groves, P. Appleby, *et al.*, "The research data management platform (rdmp): A novel, process driven, open-source tool for the management of longitudinal cohorts of clinical data," *GigaScience*, vol. 7, no. 7, p. g060, 2018. DOI: 10.1093/gigascience/giy060.
- [190] A. Cox and E. Verbaan, *Exploring research data management*. Facet publishing, 2018. DOI: 10.29085/9781783302802.
- [191] L. Ellenbeck, "Methods and Best Practices for User Journey Mapping within Distributed Services," Tech. Rep., Jan. 2020. DOI: 10.5281/zenodo.7865284.

- [192] D. Schimmel, “Automated Pipeline for Data Preprocessing,” Tech. Rep., Dec. 2019. DOI: 10.5281/zenodo.7865245.
- [193] D. Schimmel, “Data Analysis for Distributed Services: Web Application for Process Discovery,” FH Aachen University of Applied Sciences, Tech. Rep., Aug. 2020. DOI: 10.5281/zenodo.7865119.
- [194] A. Berti, S. J. van Zelst, and W. van der Aalst, “Process mining for python (pm4py): Bridging the gap between process-and data science,” *arXiv preprint arXiv:1905.06169*, pp. 13–16, 2019. DOI: 10.48550/arXiv.1905.06169.
- [195] M. Müller, C. Terboven, M. Nellesen, M. A. Yazdi, M. Polize, I. Lang, B. Heinrichs, F. Claus, and T. Eifert, *Combining HPC, AI and RDM: Challenges and Approaches*, International Conference on Society for Industrial and Applied Mathematics (SIAM), 2023. [Online]. Available: [https://meetings.siam.org/session/dsp\\_talk.cfm?p=123896](https://meetings.siam.org/session/dsp_talk.cfm?p=123896).
- [196] M. A. Yazdi, M. Politze, and M. Müller, “A Novel Approach to Outlining Research Data Management Life Cycle: A Case Study,” in *Proceedings of 10th International Conference on Information Management (ICIM)*, IEEE, 2024.
- [197] S. Dunzer, M. Stierle, M. Matzner, and S. Baier, “Conformance checking: A state-of-the-art literature review,” in *Proceedings of the 11th international conference on subject-oriented business process management*, 2019, pp. 1–10. DOI: 10.1145/3329007.3329014.
- [198] B. Heinrichs and M. A. Yazdi, “Determining the Similarity of Research Data by Using an Interoperable Metadata Extraction Method,” in *Proceedings of the 1st International Conference on Research Data Infrastructure (CoRDI)*, Open Science, 2023.
- [199] M. A. Yazdi, M. Politze, and B. Heinrichs, “Research Data Reusability with Content-Based Recommender System,” in *Proceedings of the 4th International Conference on Deep Learning Theory and Applications (DeLTA)*, Springer, 2023. DOI: 10.1007/978-3-031-39059-3\_10.
- [200] M. Klein, “Recommender Systems for Research Data Management,” FH Aachen University of Applied Science, Tech. Rep., Jan. 2023. DOI: 10.5281/zenodo.7865199.
- [201] Y. Ünal, G. Chowdhury, S. Kurbanolu, J. Boustany, and G. Walton, “Research data management and data sharing behaviour of university researchers,” in *Proceedings of ISIC: The Information Behaviour Conference*, vol. 3, 2019, p. 15.
- [202] C. Tenopir, S. Allard, L. Baird, R. J. Sandusky, A. Lundeen, D. Hughes, and D. Pollock, “Academic librarians and research data services: Attitudes and practices,” *IT Lib: Information Technology and Libraries Journal. Issue 1*, 2019. DOI: 10.1177/0340035212473089.
- [203] M. Färber and A.-K. Leisinger, “Recommending datasets for scientific problem descriptions,” in *CIKM*, 2021, pp. 3014–3018. DOI: 10.1145/3459637.3482166.



- 
- [204] A. M. Nair, O. Benny, and J. George, "Content based scientific article recommendation system using deep learning technique," in *Inventive Systems and Control: Proceedings of ICISC 2021*, Springer, 2021, pp. 965–977. DOI: 10.1007/978-981-16-1395-1\_70.
  - [205] M. Vardigan, D. Donakowski, P. Heus, S. Ionescu, and J. Rotondo, "Creating rich, structured metadata: Lessons learned in the metadata portal project," *IASSIST Quarterly*, vol. 38, no. 3, pp. 15–15, 2015. DOI: 10.29173/iq123.
  - [206] M. Politze, S. Bensberg, and M. S. Müller, "Managing discipline-specific metadata within an integrated research data management system.," in *ICEIS (2)*, 2019, pp. 253–260. DOI: 10.5220/0007725002530260.
  - [207] V. Revathy and S. P. Anitha, "Cold start problem in social recommender systems: State-of-the-art review," *Advances in Computer Communication and Computational Sciences: Proceedings of IC4S 2017, Volume 1*, pp. 105–115, 2019. DOI: 10.1007/978-981-13-0341-8\_10.
  - [208] S. Phung, A. Kumar, and J. Kim, "A deep learning technique for imputing missing healthcare data," in *2019 41st annual international conference of the IEEE engineering in medicine and biology society (EMBC)*, IEEE, 2019, pp. 6513–6516. DOI: 10.1109/embc.2019.8856760.
  - [209] D. A. Bennett, "How can i deal with missing data in my study?" *Australian and New Zealand journal of public health*, vol. 25, no. 5, pp. 464–469, 2001. DOI: 10.1111/j.1467-842x.2001.tb00294.x.
  - [210] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018. DOI: 10.48550/arXiv.1810.04805.
  - [211] A. Rogers, O. Kovaleva, and A. Rumshisky, "A primer in bertology: What we know about how bert works," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 842–866, 2021. DOI: 10.1162/tacl\_a\_00349.
  - [212] C. C. Aggarwal and P. S. Yu, "Outlier detection for high dimensional data," in *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, 2001, pp. 37–46. DOI: 10.1145/375663.375668.
  - [213] A. Boukerche, L. Zheng, and O. Alfandi, "Outlier detection: Methods, models, and classification," *ACM Computing Surveys (CSUR)*, vol. 53, no. 3, pp. 1–37, 2020. DOI: 10.1145/3381028.
  - [214] N. N. Ho-Dac, S. J. Carson, and W. L. Moore, "The effects of positive and negative online customer reviews: Do brand strength and category maturity matter?" *Journal of marketing*, vol. 77, no. 6, pp. 37–53, 2013. DOI: 10.1509/jm.11.0011.
  - [215] C. L. Borgman, *Big data, little data, no data: Scholarship in the networked world*. MIT press, 2017, ISBN: 9780262028561.
  - [216] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," *The adaptive web: methods and strategies of web personalization*, pp. 291–324, 2007. DOI: 10.1007/978-3-540-72079-9\_9.

- [217] G. Linden, B. Smith, and J. York, “Amazon. com recommendations: Item-to-item collaborative filtering,” *IEEE Internet computing*, vol. 7, no. 1, pp. 76–80, 2003. DOI: 10.1109/mic.2003.1167344.
- [218] V. Vassallo and A. Felicetti, “Towards an ontological cross-disciplinary solution for multidisciplinary data: Vi-seem data management and the fair principles,” *International Journal on Digital Libraries*, vol. 22, pp. 297–307, 2021. DOI: 10.1007/s00799-020-00285-5.
- [219] C. Desrosiers and G. Karypis, “A comprehensive survey of neighborhood-based recommendation methods,” *Recommender systems handbook*, pp. 107–144, 2010. DOI: 10.1007/978-0-387-85820-3\_4.
- [220] L. Candela, D. Castelli, P. Manghi, and A. Tani, “Data journals: A survey,” *Journal of the Association for Information Science and Technology*, vol. 66, no. 9, pp. 1747–1762, 2015. DOI: 10.1002/asi.23358.
- [221] S. Khan, X. Liu, K. A. Shakil, and M. Alam, “A survey on scholarly data: From big data perspective,” *Information Processing & Management*, vol. 53, no. 4, pp. 923–944, 2017. DOI: 10.1016/j.ipm.2017.03.006.
- [222] H. Abdollahpouri, R. Burke, and B. Mobasher, “Controlling popularity bias in learning-to-rank recommendation,” in *Proceedings of the eleventh ACM conference on recommender systems*, 2017, pp. 42–46. DOI: 10.1145/3109859.3109912.
- [223] L. Vuong Nguyen, T.-H. Nguyen, J. J. Jung, and D. Camacho, “Extending collaborative filtering recommendation using word embedding: A hybrid approach,” *Concurrency and Computation: Practice and Experience*, e6232, 2021. DOI: 10.1002/cpe.6232.
- [224] L. Cai and Y. Zhu, “The challenges of data quality and data quality assessment in the big data era,” *Data science journal*, vol. 14, 2015. DOI: 10.5334/dsj-2015-002.
- [225] R. E. Shiffler, “Maximum z scores and outliers,” *The American Statistician*, vol. 42, no. 1, pp. 79–80, 1988. DOI: 10.2307/2685269.
- [226] E. Kindler, V. Rubin, and W. Schäfer, “Process mining and petri net synthesis,” in *International Conference on Business Process Management*, Springer, 2006, pp. 105–116. DOI: 10.1007/11837862\_12.
- [227] A. Dean, D. Voss, D. Dragulji, *et al.*, *Design and analysis of experiments*. Springer, 1999, vol. 1. DOI: 10.1007/978-3-319-52250-0.
- [228] M. Rafiei and W. M. van der Aalst, “Mining roles from event logs while preserving privacy,” in *International Conference on Business Process Management*, Springer, 2019, pp. 676–689. DOI: 10.1007/978-3-030-37453-2\_54.
- [229] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: Synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002. DOI: 10.1613/jair.953.

- 
- [230] T. S. Wiens, B. C. Dale, M. S. Boyce, and G. P. Kershaw, “Three way k-fold cross-validation of resource selection functions,” *Ecological Modelling*, vol. 212, no. 3-4, pp. 244–255, 2008. DOI: 10.1016/j.ecolmodel.2007.10.005.
- [231] M. Hofmann and R. Klinkenberg, *RapidMiner: Data mining use cases and business analytics applications*. ACM, 2013. DOI: 10.5555/2543538.
- [232] C. Tenopir, N. M. Rice, S. Allard, L. Baird, J. Borycz, L. Christian, B. Grant, R. Olendorf, and R. J. Sandusky, “Data sharing, management, use, and reuse: Practices and perceptions of scientists worldwide,” *PloS one*, vol. 15, no. 3, e0229003, 2020. DOI: 10.1371/journal.pone.0229003.
- [233] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, “On the surprising behavior of distance metrics in high dimensional space,” in *Database TheoryICDT 2001: 8th International Conference London, UK, January 4–6, 2001 Proceedings 8*, Springer, 2001, pp. 420–434. DOI: 10.1007/3-540-44503-x\_27.
- [234] S. Zhang, L. Yao, A. Sun, and Y. Tay, “Deep learning based recommender system: A survey and new perspectives,” *ACM computing surveys (CSUR)*, vol. 52, no. 1, pp. 1–38, 2019. DOI: 10.1145/3285029.
- [235] X. Zhou, J. He, G. Huang, and Y. Zhang, “Svd-based incremental approaches for recommender systems,” *Journal of Computer and System Sciences*, vol. 81, no. 4, pp. 717–733, 2018. DOI: 10.1016/j.jcss.2014.11.016.
- [236] A. M. Jones, A. Arya, P. Agarwal, P. Gaurav, and T. Arya, “An ontological sub-matrix factorization based approach for cold-start issue in recommender systems,” in *2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC)*, IEEE, 2017, pp. 161–166. DOI: 10.1109/ctceec.2017.8455147.
- [237] M. A. Yazdi, *Dataset used for this phd dissertation*, 2023. DOI: 10.18154/RWTH-2023-05866.
- [238] M. A. Yazdi, *Software applications for this phd dissertation*, 2023. DOI: 10.18154/RWTH-2023-05865.