Why DataOps Matters More Than Ever:

A Technical Debt Perspective on Data Quality and MLOps in AI Systems

Sergio Moreschini - 16.12.2025

Tampere University

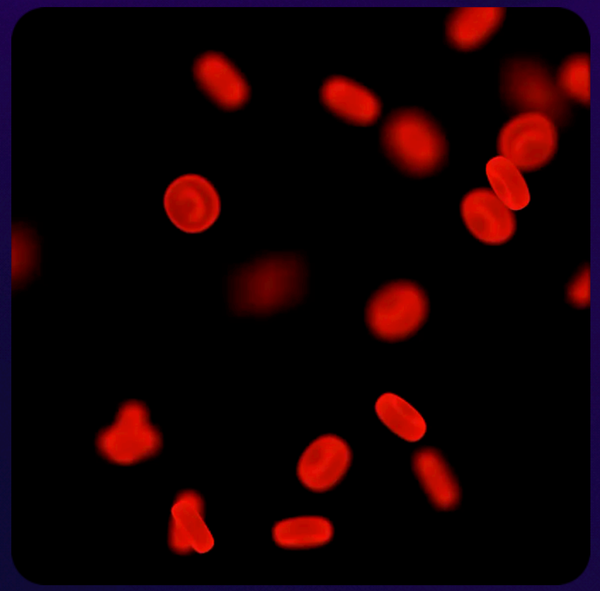
Summary

- The Problem: Hidden Data Technical Debt
- Empirical strategy: Mining Software Repositories
- The Framework: DataOps meets ISO/IEC 25012
- My Lesson Learned: Industrial Collaboration
- The Takeaway: Why it matters for you

Who I Am

Researcher, Engineer, Bridge Builder.

- Master Degree ITC:
  Università degli Studi **Roma Tre** 110L
- Exchange “Erasmus Program”:
  Tampere University of Technology
- Started Ph.D in Signal Processing and Machine Learning
- Moved to Software Engineering



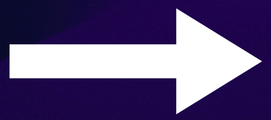
Tampere University

What was like coding before ML

Navigating in Technical Debt

Developing classical Signal Processing in Matlab

- Unversioned Code
- Lots of #ToDo
- Unversioned Data



Developing ML in Python

- Unversioned Code
- Lots of #ToDo
- Unversioned Data
- Spaghetti dependencies on Libraries

The Problem
Technical Debt

"Shipping first-time **code** is like going into debt. A little debt speeds development so long as it is paid back promptly with a rewrite. Objects make the cost of this transaction tolerable. The danger occurs when the debt is not repaid. Every minute spent on not-quite-right code counts as interest on that debt. Entire engineering organizations can be brought to a stand-still under the debt load of an unconsolidated implementation, object-oriented or otherwise."

Ward Cunningham, 1992

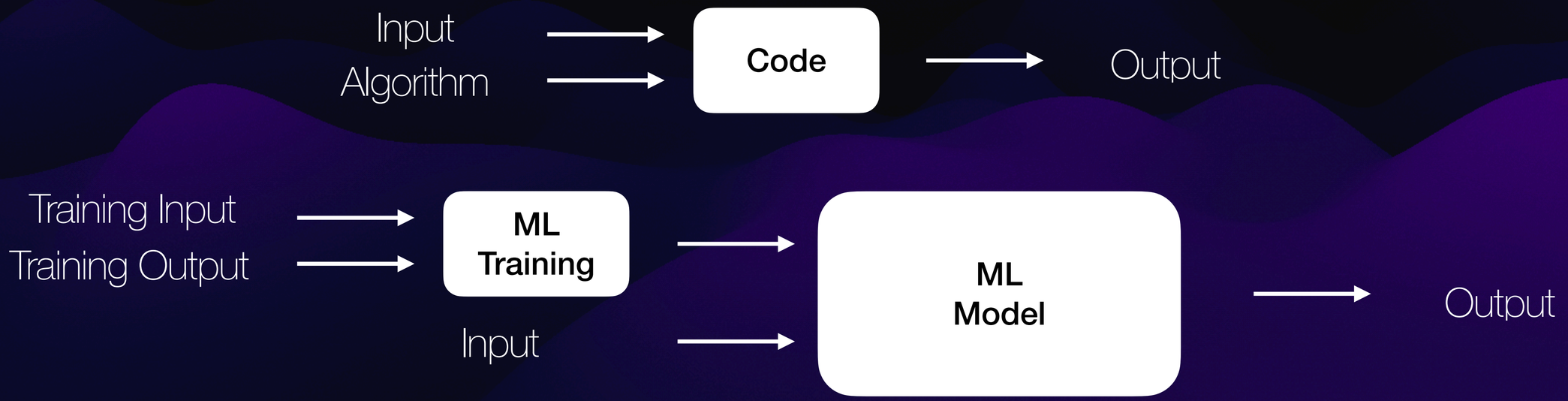
The Problem
Technical Debt

Technical Debt refers to the consequence of choosing sub-optimal technical decisions that deliver ***short-term benefits at the cost of increased complexity, rework, or reduced quality*** *in the future.*

Ward Cunningham, 1992

The Problem

Technical Debt

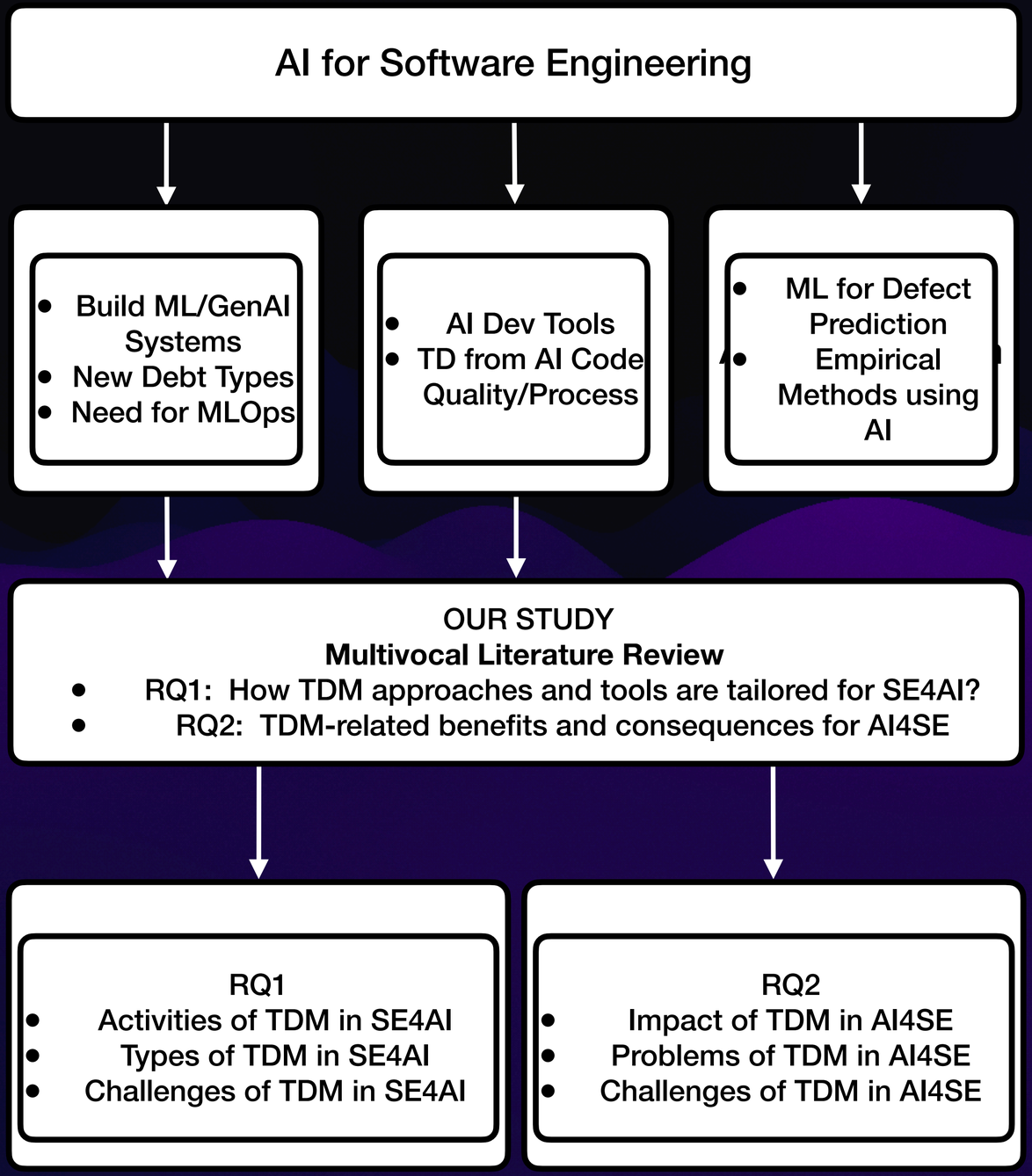


Tampere University

The Evolution of Technical Debt from DevOps to Generative AI

A multivocal literature review

- SE4AI
  - specialized SE approaches for developing software that is driven by or uses AI
- AI4SE
  - Use of tools for increasing productivity software engineering lifecycle
- AI4SE - Research
  - Use of tools to extend and validate research contributions



RQs

RQ1: How does TDM need to be tailored to match the needs of
developing AI-driven software?
RQ2: How does the use of AI during code development affect TDM?

Search Strategy

Multivocal Approach:

- 4 digital libraries + **3** Search engines:
  - ACM Digital Library, IEEEXplore Digital Library, Web of Science, and Scopus.
  - Google, Yahoo, Bing
- Systematic screening following Kitchenham guidelines
- 10+ Years of AI-Enabled SE evolution

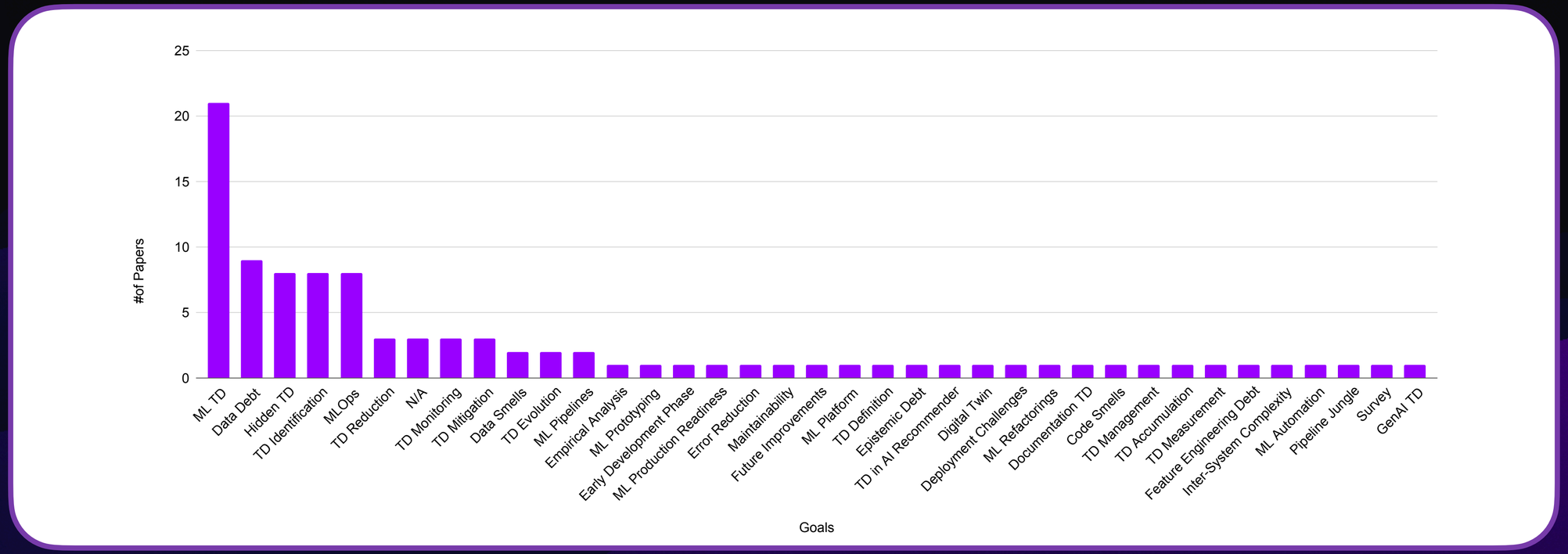
Corpus Overview

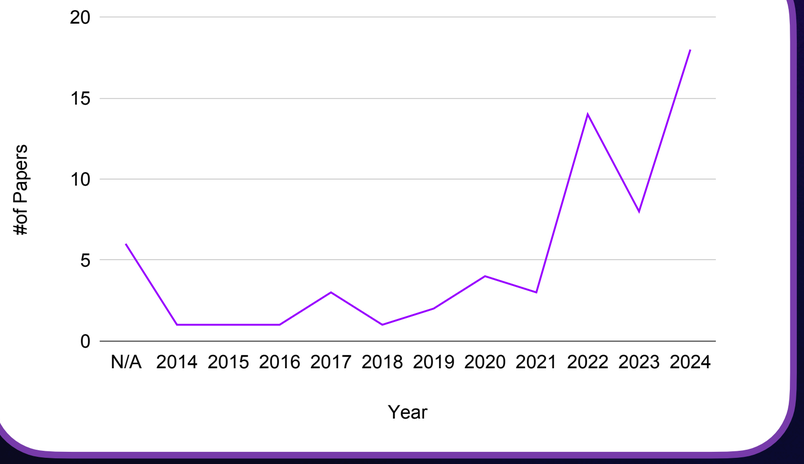
Retrieval from bibliographic sources	894
Reading by Title and Abstract	-812
Full Reading	-20
Quality Assessment	0
Primary Studies	**62**

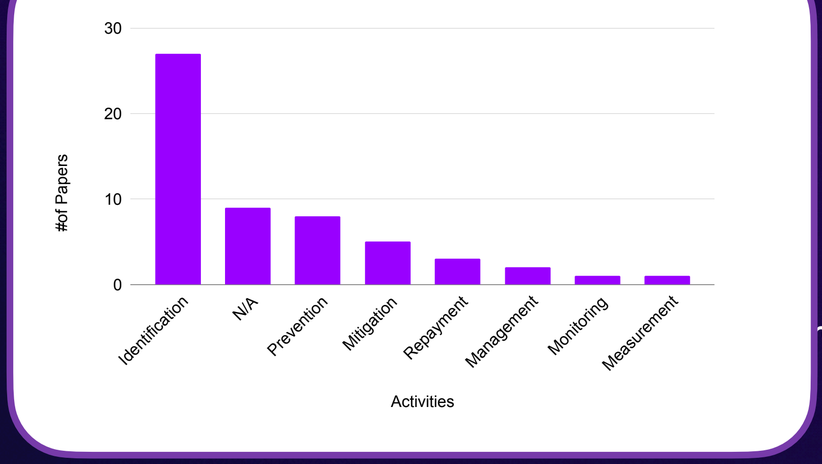
Data Extraction

RQ1	Study theme
	Activities of TD management
	Type of Td identified in the work
	Challenges faced in the work
	Impact of the work
RQ2	Main problem to be addressed
	Challenges faced in the work
	Opportunities encountered in the work

RQ1: How does TDM need to be tailored to match the needs of developing AI-driven software?

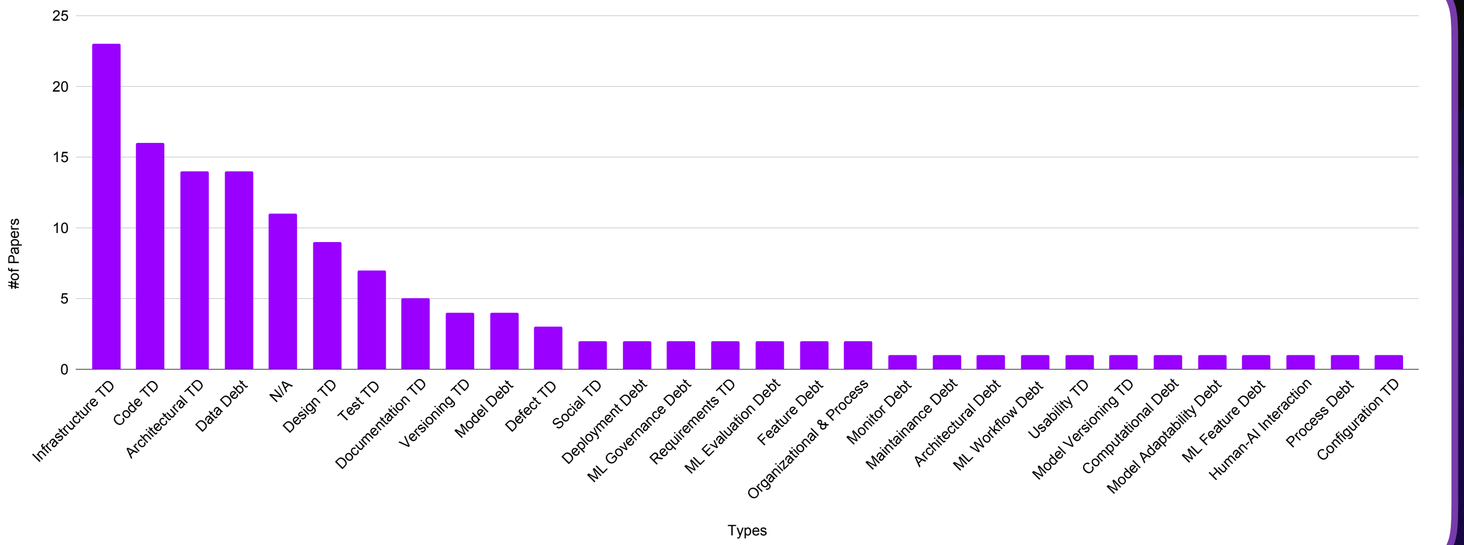


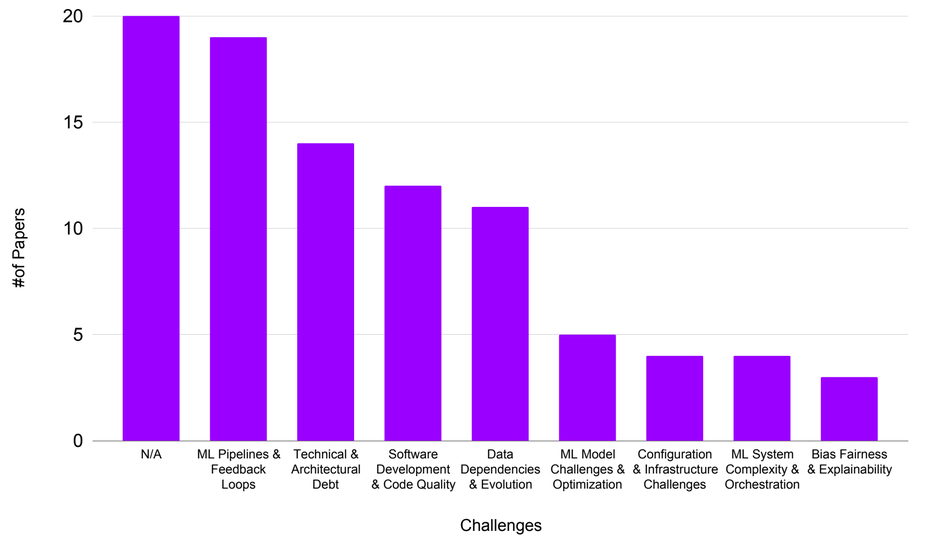




re University

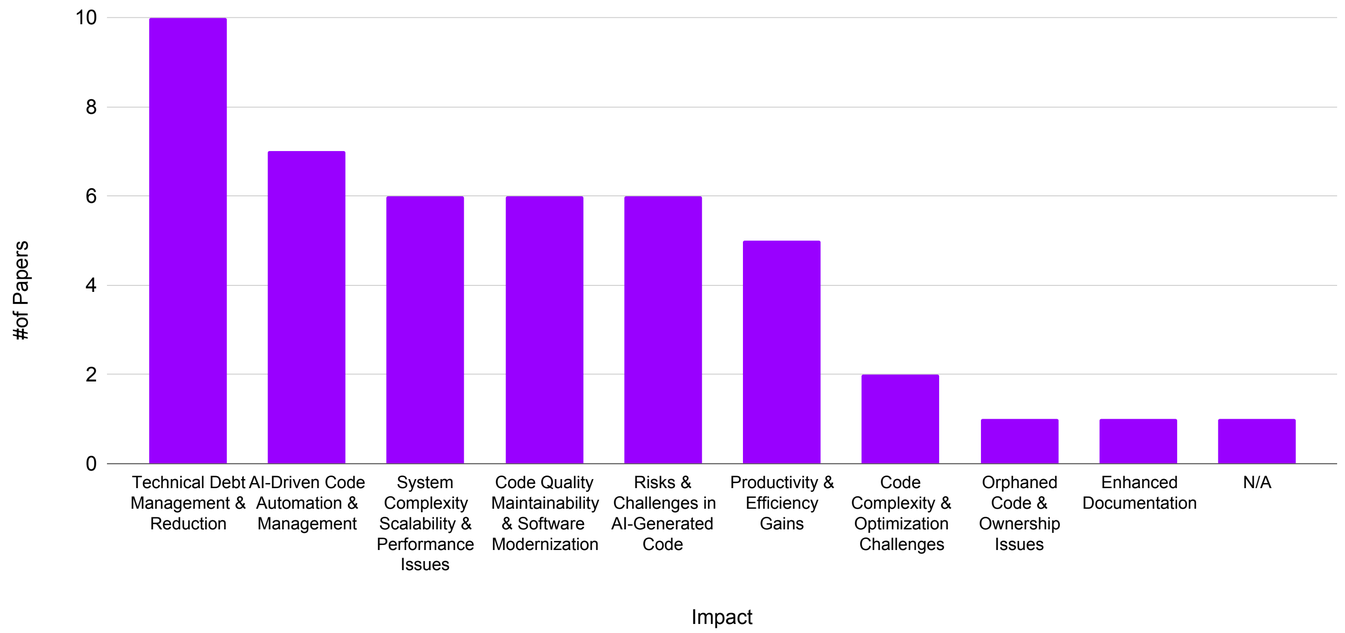
RQ1: How does TDM need to be tailored to match the needs of developing AI-driven software?

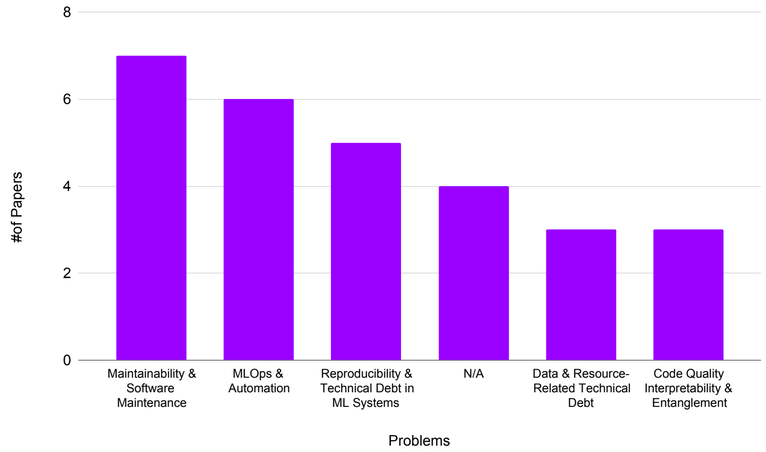


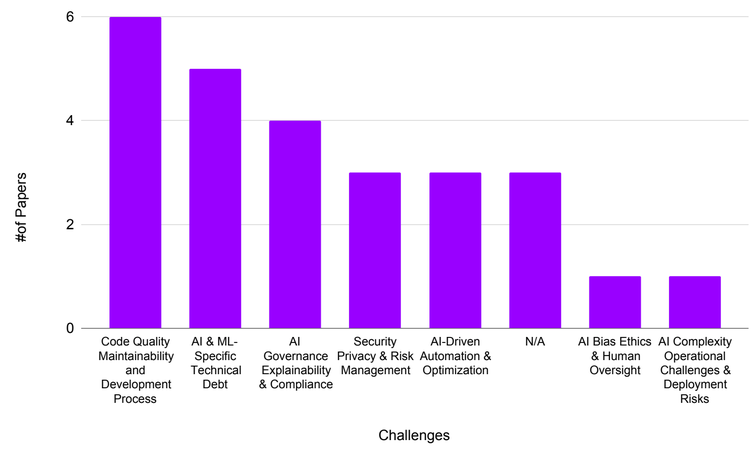


Tampere University

RQ2: How does the use of AI during code development affect TDM?







ere University

The Evolution of Technical Debt from DevOps to Generative AI

A multivocal literature review

- New Debts are related to:
  - ML Pipelines and Feedback Loops
  - ML Models Challenges and Optimizations
  - Explainability
  - Architectures

Tampere University

The Evolution of Technical Debt from DevOps to Generative AI

A multivocal literature review

All of these new debts are all connected directly or indirectly to:

Data Debt

Tampere University

Empirical Strategy
Mining Software Repository

- Motivation:
  - Imbalance of TDM for Code and Infrastructure compared to Data
  - Need to strengthen the role of DataOps as a foundational discipline in the management of ML and AI-driven software systems.

- Goal:
  "Investigating how DataOps practices are :
  - adopted in open-source and industrial ML projects,
  - assessing their maturity levels, and
  - analyzing the correlation between the maturity of DataOps practices and the accumulation of TD."

Empirical Strategy
Mining Software Repository

RQS:

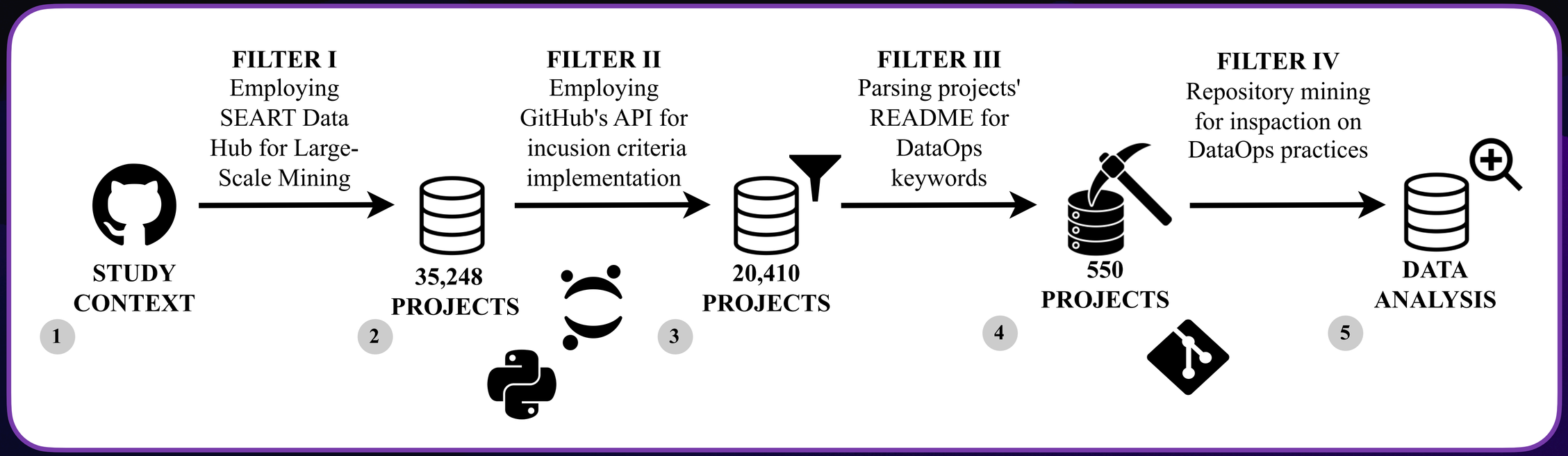
- Are DataOps practices systematically adopted by OSS projects?
- What is the distribution of the DataOps maturity among projects following DataOps practices?
- What is the relationship between DataOps projects' maturity level and the TD symptoms detected in their development process?

- **Hypothesis:** there must be a highly correlated relationship between the DataOps maturity level adopted by a project and its TD symptoms

Tampere University

Empirical Strategy

RQ1: DataOps practices systematically adopted by OSS projects



Tampere University

Empirical Strategy

RQ2: DataOps maturity

Level	Description
0 - None	No evidence of DataOps practices
1 - Foundational	Minimal automation
2 - Basic	Some automation and limited integration
3 - Structued	Consistent application of DataOps practices
4 - Advanced	Full Automation

Empirical Strategy

RQ3: Relationship between DataOps maturity and TD

Hotspot code health

Healthy

A weighted average of the Code Health in your hotspots. Generally, this is the most critical metric since low Code Health in a hotspot will be expensive.

Average code health

Problematic

A weighted average of all the files in the codebase. This KPI indicates how deep any potential Code Health issues go.

Worst performer

Unhealthy

A single file Code Health score representing the lowest Code Health in any module across the codebase, and it points out long-term risks.

Source: https://codescene.com/product/code-health

Tampere University

The work so far…

Main problem raised

- Lack of tools offering a comprehensive alternative to static analysis tools in software engineering
- RAG, powerful ally or increasing debt threat?
- What opportunities are we missing?

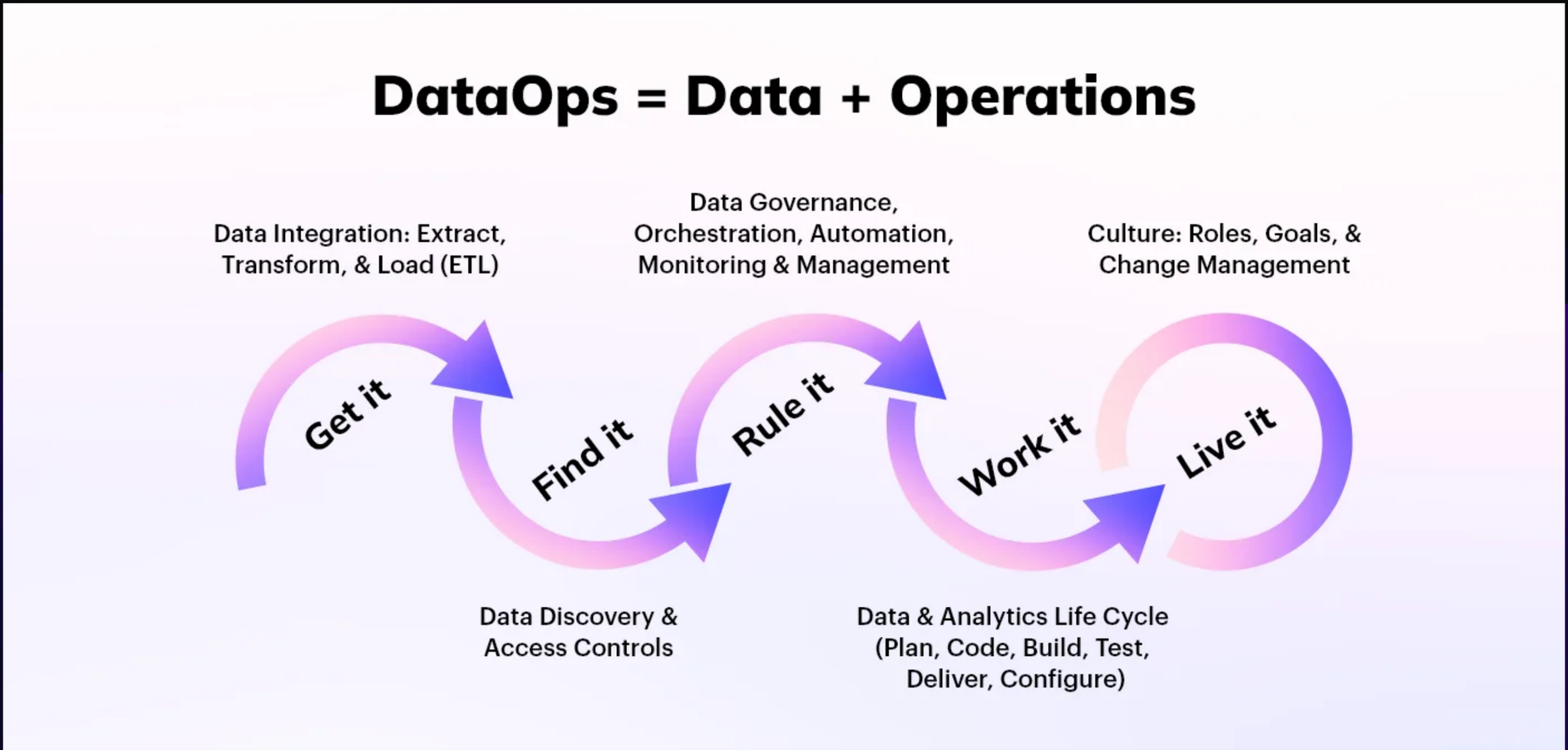
Data-related technical debt is the most pervasive and the most neglected type of debt in AI systems

The Framework

Symptoms of Data-Related TD:

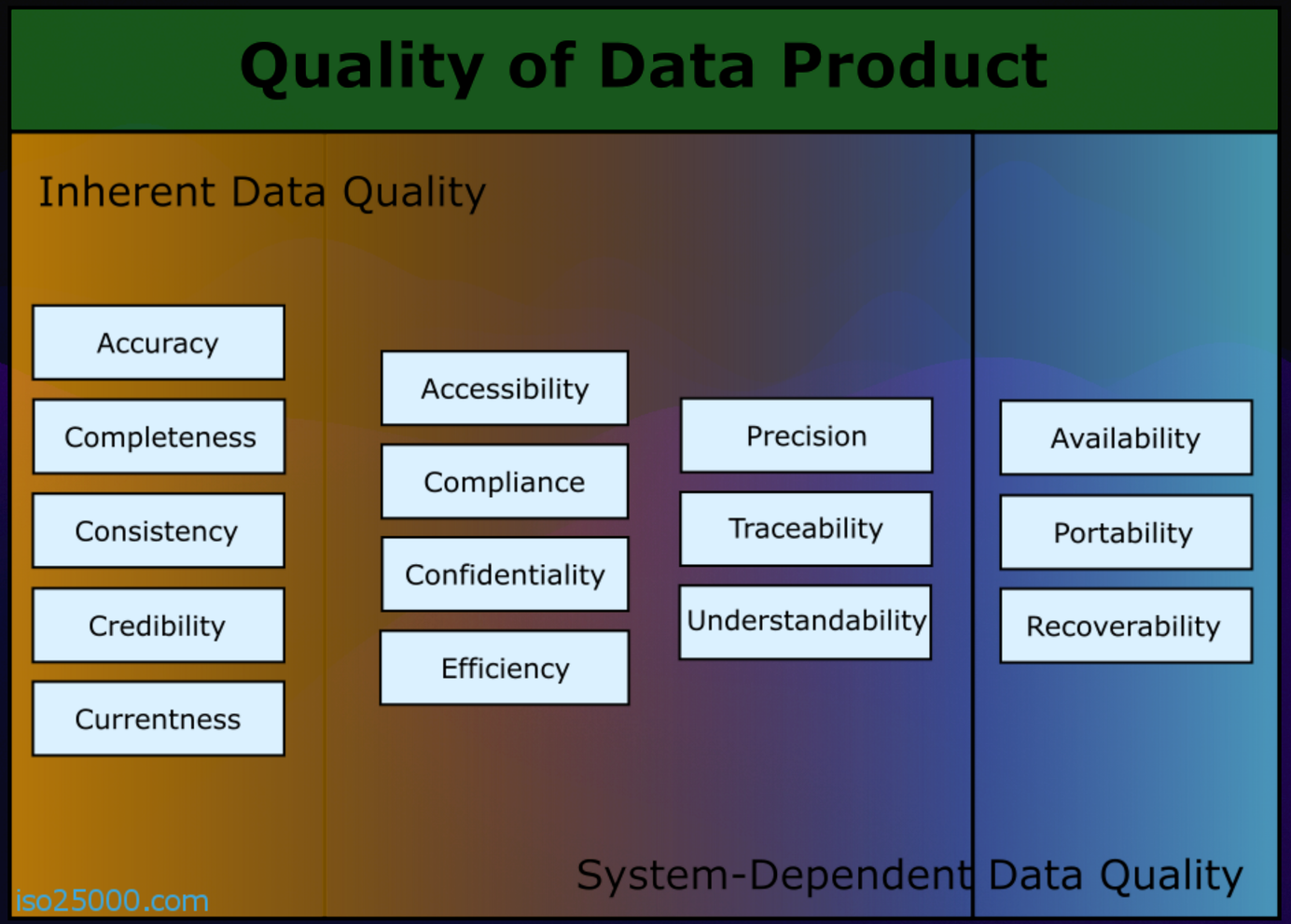
- Data drift & concept drift
- Data provenance/lineage
- Data quality validation
- Missing or inconsistent Data versioning
- Fragmented or hidden transformations
- Outdated, stale, or low-quality datasets

Why DataOps



Source: https://codiant.com/blog/dataops-and-its-benefits-for-your-data-management/

ISO/IEC 25012/2008



Source: https://www.iso25000.com/index.php/en/iso-25000-standards/iso-25012

Towards AI-based systems

Inherent Characteristics

Characteristic	Meaning	DataOps Guidelines	DataOps Practices
Accuracy	Real-word entities are correctly represented	Rigorous Ingestion and Validation steps	Automated Data Validation tests
Completeness	All required data is present	Continuous Delivery and Monitoring	Monitor for missing-values count and Completeness Check
Consistency	No conflicts	ETL	Data Lineage Tools
Credibility	Trusted/ meets Normative expectations	Collaboration and Transparency	Reliability of Metadata
Cureteness	Data is up to date	Near real-time data	Triggers for data refresh

Towards AI-based systems

Inherent & System-dependent Characteristics (I)

Characteristic	Meaning	DataOps Guidelines	DataOps Practices
Accessibility	Available and Retrievable	Accessibilities for Teams	Data-Cataloguing
Compliance	Data meets Standard, Policies and Regulatory/Legal Requirements	Data Governance	Integration in Pipelines
Confidentiality	Protection against unauthorized access	Secure by Design	Dynamic Data Access policies
Efficiency	Process with minimal resources overhead	Emphasis in Pipelines	Resource Usage Monitoring, Data Optimization Formats

Towards AI-based systems

Inherent & System-dependent Characteristics (II)

Characteristic	Meaning	DataOps Guidelines	DataOps Practices
Precision	Enough details	Tests for precision losses across pipelines	Checks for rounding/truncations, enforce schema definition
Traceability	Trace and Track Origins, Changes, Lineage and Usage	Reproducibility and Version Control	Data lineage tools, Metadata and Data Versioning tools
Understandability	Clear, Interpretable and ready to be used	Democratization of Data	Data catalogues

Towards AI-based systems

System-dependent Characteristics (II)

Characteristic	Meaning	DataOps Guidelines	DataOps Practices
Availability	Accessible and Usable when needed	Continuous data delivery and reliability are core	Highly-available architectures
Portability	Can be moved/transferred without loss in quality	Pipelines are modular and infrastructure are cloud/hybrid	Standardized formats
Recoverability	Restorable after loss, corruption, or failure	Rollback available	Versioned Data Storage, Medallion approach

DataOps tools

Some alternatives

Commercial Tools

- Databricks
- Fabric (Microsoft)
- Snowflake

- Commercial Stacks Data Cloud
  - Google
  - AWS

Designing a Cloud-Native MLOps Pipeline

Industrial Collaboration

- Company making use of Databricks
- Making best use of the data they had
- Provide value though sensors
- *Incepting* MLOps Guidelines within the company

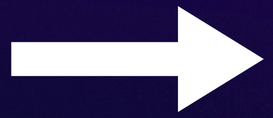
Tampere University

Why Databricks

- MLOps is a set of processes and automation for managing code, data, and models to improve performance, stability and long-term efficiency of ML systems

MLOps = DevOps + DataOps + ModelOps

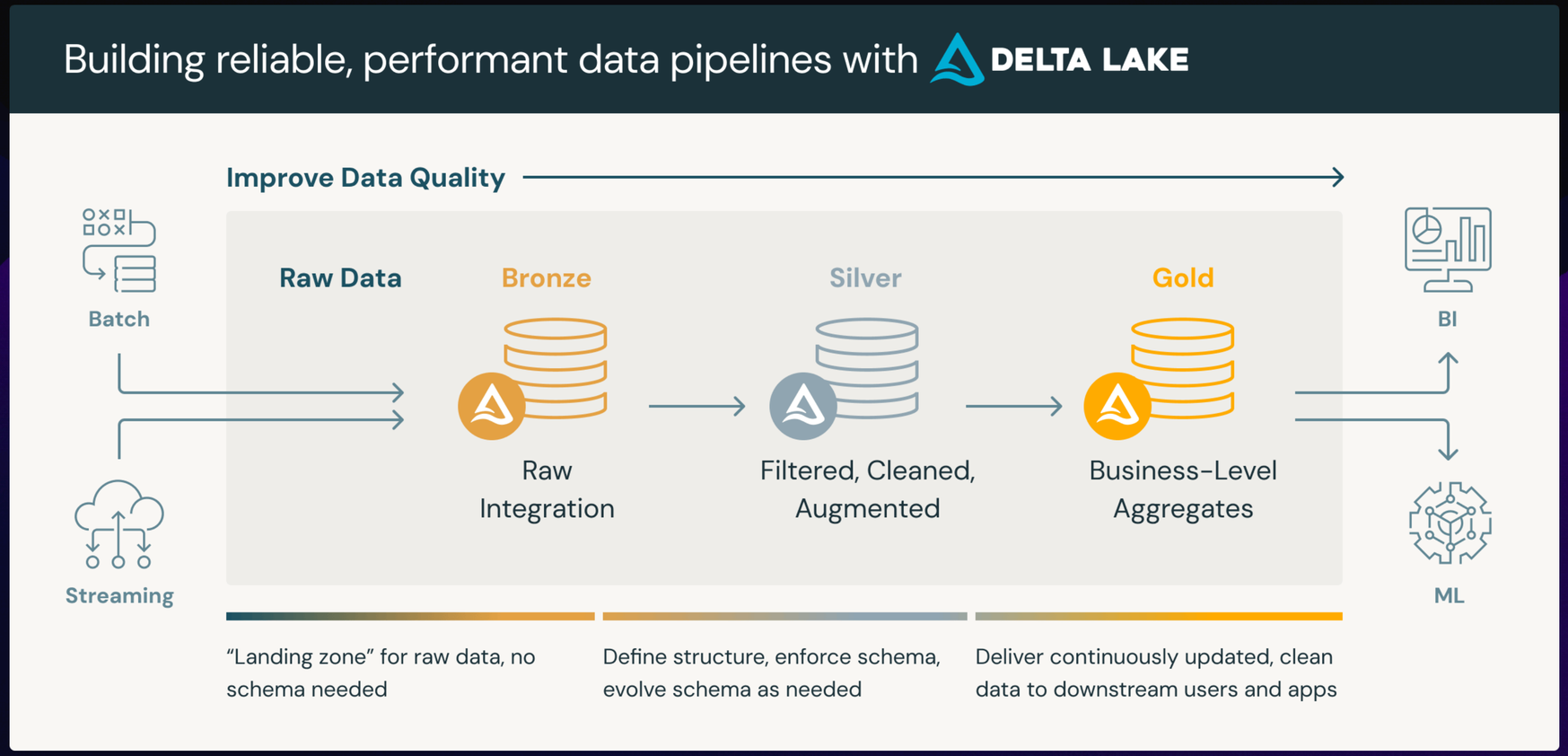
- Reproducibility:
  - Data: Data management
  - Code: Version Control
  - Models: Model Management
  - Automated integration of CI/CD



- Delta Lake
- Git
- mlFLow
- AzureML

Why Databricks

Data-centric approach

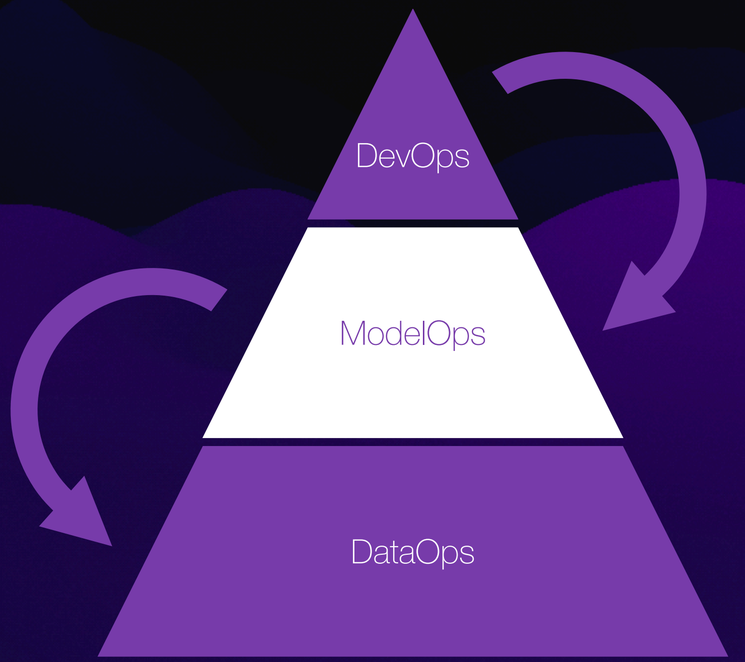


Source: https://www.databricks.com/glossary/medallion-architecture

Why Databricks

A new Perspective

- Through Databricks MLOps is an incremental and progressive approach
  - First DataOps is achieved (Silver Minimum)
  - ModelsOps is enforced and made available through mlflow
  - Finally Automation links all together



Tampere University

Lesson learned

Industrial Collaboration

- Around 60% of the time on the platform was related to DataOps; outside the platform most of the time was related to disseminating xOps practices
- All the data, and models were made available for anyone who had access inside the platform. → Reused for other projects
- Collaboration provided improvements also in other projects where the industrial team was working

Why this matters

1) Data Quality **IS** Technical Debt in AI Systems

- Data issues accumulate silently → Debt
- Data Pipelines are both the largest bottleneck and reason for ML systems to fail
- Poor data practices → highest cost in failures & downtime
- Investing in DataOps = reduce risk of Failing in AI

Why this matters

2) DataOps maturity accelerates AI adoption

- DataOps = stable and reproducible AI pipelines
- DataOps maturity = faster, and increased trust, in value for data
- DataOps is evolving to accelerate AI
- ISO 25012 → measurable governance

What does this mean?

3) Moving towards GenAi

- Europe is rapidly moving towards GenAI, neglecting data foundations
- Most companies make use of external available models
- We will soon move towards “in house” made models through SLMs
- **Your data is your business!**
    - Take care of it!

Tampere University

What does this mean?

- If you are planning to adopt/improve your GenAI/ML business: you need DataOps
- To build AI pipelines → you need data quality governance
- For sustainable AI → Reduce TD is vital

Let’s keep in touch!

Sergio Moreschini, Ph.D.
MLOps Architect | AI Infrastructure & Cloud-Native ML |
Tampere University and University of Oulu

Researcher, Engineer, Bridge Builder.