TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Lukas Sulg 221489IABM

The Integration of Business Intelligence and Third-Party Applications for Data Management

Master's thesis

Supervisor: Ahti Lohk
PhD

Tallinn 2024

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Lukas Sulg 221489IABM

Äritehnoloogia platvormi ja kolmandate osapoolte rakenduste integreerimine andmehalduse jaoks

Magistritöö

Juhendaja: Ahti Lohk
PhD

Tallinn 2024

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Lukas Sulg

08.05.2024

Abstract

The integration of a lineage layer into Business Intelligence (BI) systems presents a pivotal enhancement for addressing architectural, operational, and governance challenges inherent in modern digital environments. This thesis investigates the seamless incorporation of third-party applications into proprietary BI platforms, aiming to bolster organizational efficiency and improve decision-making through enhanced data governance. Utilizing a theoretical case study, the author aims to integrate a lineage layer into a BI architecture. The main focus is on creating compatibility with current data structures and workflows.

Findings from the study highlight that the addition of a lineage layer significantly augments decision-making capabilities by offering deeper insights into data origins and transformations. This aids in identifying inconsistencies and optimizing operational strategies. Furthermore, the integration necessitates robust security protocols to protect sensitive metadata and ensure compliance with regulations like GDPR [1].

Ultimately, this thesis contributes to the discourse on BI system enhancement, illustrating that strategic third-party integrations can cultivate a more informed organizational culture and elevate overall data management, albeit requiring meticulous consideration of architectural and security implications.

This thesis is written in english and is 51 pages long, including 5 chapters, 8 figures and 2 tables.

Annotatsioon

Äritehnoloogia platvormi ja kolmandate osapoolte rakenduste integreerimine andmehalduse jaoks

Päritolu kihi integreerimine äritehnoloogia platvormidega kujutab endast olulist uuendust, mis aitab lahendada kaasaegsetele digitaalsetele keskkondadele omaseid probleeme. Käesolev lõputöö keskendub kolmanda osapoole rakenduste sujuvale integreerimisele BI-platvormidesse, eesmärgiga suurendada organisatsiooni tõhusust ja parandada andmehaldust.

Lõputöö viiakse läbi kui teoreetiline juhtumiuuring, mille peamine eesmärk on BI-arhitektuuri päritolu kihi loomine ja integreerimine. Uuring keskendub erinevate andmestruktuuride ja töövoogude ühtlustamisele.

Uuringu tulemused toovad esile, kuidas päritolu kiht muudab andmetega töötlemist, võimaldades kiirelt näha, milliseid operatsioone on andmetega varem tehtud. Tänu päritolu kihile on lihtne tuvastada andmetes ebakõlasid, mis omakorda aitab probleeme kiiremini lahendada ja suurendab andmete usaldusväärsust.

Lõppkokkuvõttes aitab see lõputöö kaasa äritehnoloogia platvormide funktsionaalsuse suurendamisele, illustreerides samas, kuidas kolmanda osapoole rakenduste integreerimine BI-platvormidesse mõjutab ettevõtte kultuuri ja andmehaldust.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 51 leheküljel, 5 peatükki, 8 joonist, 2 tabelit.

List of abbreviations and terms

BI	Business intelligence
ETL	Extract, Transform, Load
SDK	Software development kit
API	Application Programming interface
SSO	Single sign-on
SSL	Secure Sockets Layer
ITIL	Information Technology Infrastructure Library
JSON	JavaScript Object Notation
XML	Extensible Markup Language
OOP	Object-Oriented Programming
REST	Representational State Transfer
TLS	Transport Layer Security
HTTP/HTTPS	Hypertext Transfer Protocol/Secure Hypertext Transfer Protocol
DTO	Data Transfer Object

Table of contents

List of figures

List of tables

1 Introduction

In today's rapidly evolving digital landscape, data has become a foundational element for businesses, driving both strategic decisions and daily operations. Business Intelligence (BI) platforms play a critical role in this environment by enabling organizations to effectively analyze and leverage their data. These platforms are more than mere data repositories; they offer a comprehensive analytical framework that consolidates diverse data sources, fostering a unified, data-driven culture within organizations.

However, as these digital ecosystems grow more complex, it becomes evident that BI systems require further enhancements to keep pace. This need often leads to the integration of third-party applications via APIs, adding specialized functions that address specific operational challenges and help navigate the intricacies of the BI field. One of the pervasive challenges in this context is data inconsistency, where disparate versions of the same data are spread across different segments of an organization. This situation highlights the critical need for complete transparency and meticulous data management to support sound decision-making.

As organizations continue to integrate these sophisticated applications, several key inquiries emerge, focusing on how these integrations affect the broader landscape of organizational efficiency, decision-making processes, and data governance:

- What architectural considerations become paramount when introducing third-party applications, such as the linage layer, into existing BI systems?
- How do these integrations transform the efficiency of organizations and the landscape of decision-making?
- What are the implications for data governance, security, and compliance in the wake of these integrations?

This thesis aims to tackle these questions by theoretically creating and implementing a linage layer into BI systems as a case study. Given the proprietary nature of many BI platforms, which are typically available only as paid solutions, a theoretical approach is adopted. This method allows for an exploration of potential integrations without the need for direct access to these systems. The exploration aims to uncover the architectural, operational, and governance challenges and opportunities that arise from such integrations, thereby contributing to the broader discourse on enhancing BI capabilities and advancing a more informed organizational culture.

2 Theoretical Background

The BI Platforms are comprehensive software solutions that facilitate the process of extracting insights from an organization's data. Designed to aggregate, consolidate, and refine data from diverse sources, the platform serves as the primary repository for analysis and reporting. This centralization plays a critical role in enabling comprehensive insights and fostering a data-driven culture across the organization.

The BI Platform's integration of various data streams ensures that users throughout the organization have access to timely, accurate, and relevant information. It supports a wide array of analytical processes, including data visualization, trend analysis, predictive analytics, and dashboard creation. These capabilities allow users to derive actionable insights, supporting strategic decisions that align with organizational goals.

Despite the extensive functionalities provided by SAP Business Objects and MicroStrategy, the potential for enhancement through the integration of third-party applications via APIs is significant. This strategy not only introduces specialized functionalities to meet specific operational needs but also addresses challenges within the BI landscape.

This chapter outlines the existing BI architecture, emphasizing the main technologies employed, the challenges encountered, and the innovative solutions being implemented. It highlights the necessity of an evolved architectural strategy to meet the organization's dynamic needs, detailing the functional requirements the BI system must fulfill to support effective decision-making and strategic planning. Through strategic third-party integrations, the BI Platform is poised to extend its utility beyond core capabilities, demonstrating the organization's commitment to innovation and the continuous improvement of its data-driven decision-making processes.

2.1 Methodology

This section presents the methodology employed to theoretically analyze the incorporation of the Lineage Layer into BI systems. Due to the proprietary and often expensive nature of BI platforms, a theoretical approach becomes imperative. This approach facilitates the exploration of potential integrations without requiring direct access to these systems.

The process commences with a systematic examination of various functionalities within BI platforms. Following this, an in-depth analysis of existing solutions related to BI, data lineage, programming, legal considerations, and other relevant areas is conducted. By delineating the prerequisites for integrating a new layer into BI platforms, the author draws upon research findings to introduce a framework for creating a lineage layer.

The research also involves analyzing potential architectural modifications and the operational impact of integrating a lineage layer with BI systems. This includes evaluating the theoretical effects on organizational efficiency, decision-making, and data governance. Due to constraints on accessing real BI platforms, the study uses architectural simulations with diagrams and models to visualize the integration.

Further, the study discusses how a lineage layer could theoretically enhance BI functionalities like data accuracy and processing efficiency. Hypothetical case studies based on realistic business scenarios will illustrate potential changes in BI operations due to enhanced data management capabilities and expected benefits such as improved decision-making and competitive advantage.

Lastly, the application of theoretical frameworks related to data management includes exploring how enhanced metadata management aligns with and improves data governance frameworks and the theoretical impacts on the security of BI systems, highlighting potential improvements and vulnerabilities. This methodology aims to contribute a structured theoretical analysis to the academic and practical discourse on BI systems enhancement through third-party integrations, providing insights into architectural considerations, operational enhancements, and strategic benefits, aiming to inform future empirical research and guide practical implementations in the field of business intelligence.

2.2 Sap Business Objects

SAP Business Objects is a comprehensive reporting and analytics platform engineered to consolidate data from various databases. Its main purpose is to enable users to discover data, perform analyses to derive insights, and generate reports that visualize these insights. The interface is designed to be user-friendly, facilitating the analysis of data from data warehouses and aiding in the decision-making process. The platform offers a range of functionalities that support the creation of both simple and complex reports, and it includes features for sharing information within an organization efficiently. This image aims to further illuminate the platform's functionality. [2]

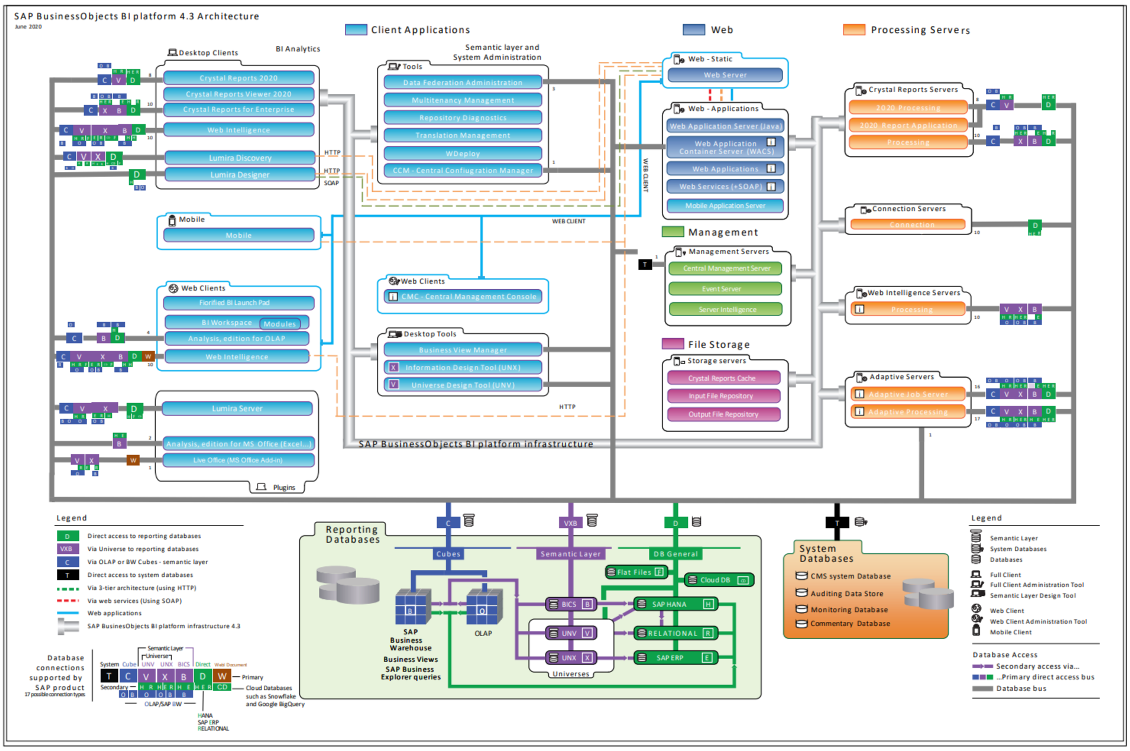

Figure 1 SAP BO component diagram

2.3 MicroStrategy

MicroStrategy is a BI and analytics software platform designed to provide users with the tools needed to analyse, report, and visualize data. This platform is utilized by organizations seeking to make data-driven decisions using comprehensive analytics. MicroStrategy's architecture supports the analysis of large datasets, enabling users to generate reports, dashboards, and visualizations that facilitate the understanding of complex data patterns and trends. This image serves to enhance understanding of the platform's capabilities. [3]

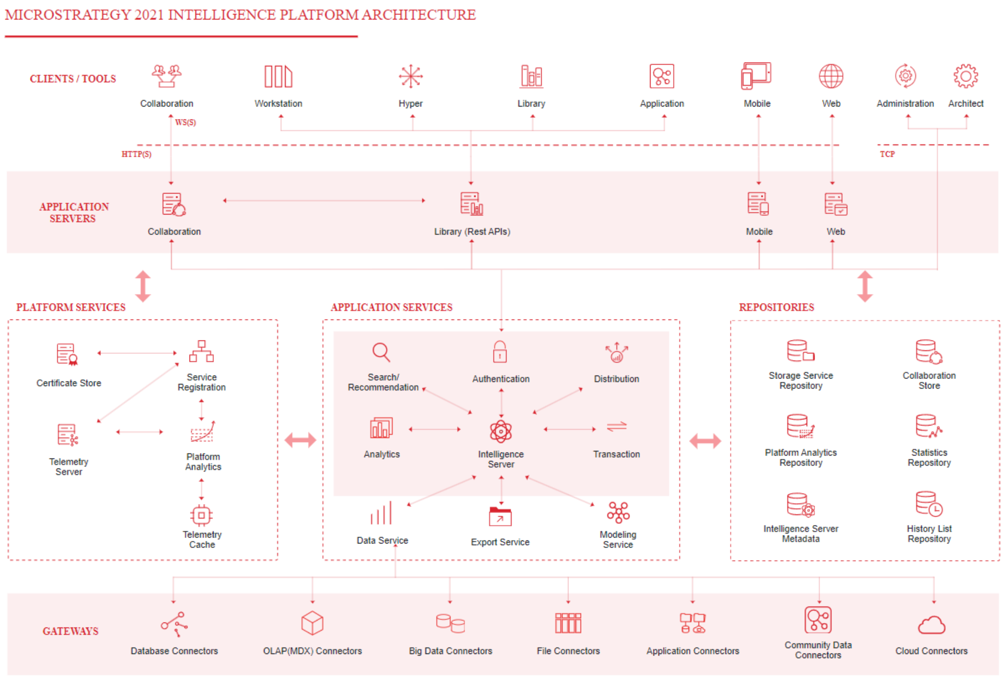

Figure 2 MicroStrategy Products and Components

2.4 Novelty

The novelty lies in the detailed exploration of how these integrations can be achieved within existing BI systems, an area that remains underexplored despite the recognized potential and emerging market solutions. While there are companies like Wiiisdom [4] that are at the forefront of creating third-party applications specifically designed to augment the capabilities of BI systems, comprehensive academic or industry-specific studies focusing on the methodologies and best practices for integrating such applications with established BI systems are scarce.

The gap in research is especially pronounced when it comes to the pragmatic, step-by-step integration processes, which require a deep dive into technical strategies, data governance protocols, and user adoption methodologies. There is a clear need for systematic studies that not only evaluate the technical feasibility and business impacts of these integrations but also provide a blueprint for successful implementation. [5] This includes considerations of data consistency, security policies, and ensuring the agility of BI systems to adapt to the addition of third-party applications without compromising performance or user experience.

Moreover, the study is set to investigate the symbiotic relationship between BI systems and third-party applications, hypothesizing that an optimal integration strategy could lead to an enhancement of data management practices, improved decision-making processes, and a potential increase in overall business intelligence agility and efficiency. As businesses continue to seek competitive advantages through data analytics, the findings of this research could provide valuable insights and contribute significantly to the existing body of knowledge in BI system enhancement.

2.5 Data

Data stands as a cornerstone of enterprise operations, significantly influencing decision-making processes, operational efficiency, and customer satisfaction. The prevalence of data-related challenges has rapidly increased the advancement of data science as an essential discipline in recent years [6]. Inconsistencies or gaps in data can lead to a host of issues, undermining the integrity and reliability of insights derived from data analytics.

Data inconsistency [7] arises when disparate versions of the same data are spread across different segments of an organization's data ecosystem. This issue can stem from a variety of causes, including errors during data entry, challenges in integrating data from diverse sources, and the absence of standardized processes for data management [8]. Moreover, as data volumes grow exponentially, managing it effectively becomes increasingly complex, elevating the importance of robust data management and data lineage practices.

Metadata is a fundamental component of information systems, taking on various forms and underpinning the core functionalities of many everyday software applications. Each piece of content within these systems is accompanied by metadata, which includes details about the content's creation, name, topic, features, and other relevant attributes. This metadata is crucial for the operation of the systems that house the content, as it enables users to locate specific items of interest, document critical information about these items, and facilitate the sharing of this information with others. In essence, metadata not only organizes and contextualizes the data but also enhances the usability and accessibility of the information within these systems. [9]

In this context, the focus on data management and data lineage is not just about keeping pace with data growth but about ensuring the quality and consistency of data across the organization. These efforts are vital for maintaining the utility and value of data as a strategic asset, emphasizing the need for ongoing attention to data quality, standardization, and governance.

3 Architectural Considerations for Integration

In this section, we will explore the architectural considerations required for integrating third-party applications with BI platforms. Our investigation will involve an exploration of a lineage layer, alongside an examination of various software development strategies, integration techniques, and hosting solutions.

3.1 Identifying Integration Needs

Evaluating the efficacy of a system and pinpointing areas for enhancement often involves a degree of subjectivity. In the realm of BI platforms, while the overarching framework is robust for data management, certain areas could benefit from refinement [10]. From the author's perspective, a notable gap lies in the tracing of data lineage for dashboards, reports, or dossiers—terms that are used interchangeably to describe the analytical output.

3.2 Functional requirements

To effectively integrate third-party applications with our BI platform, it is essential to specify the criteria that the new tool must satisfy. By meticulously reviewing existing literature and tapping into the author's domain knowledge and expertise, the author has systematically curated a comprehensive set of functional requirements for the lineage layer tool.

#	Category	Description
1.1	**Metadata Parsing and Storage**	The application must support parsing metadata ensuring comprehensive data coverage.
2.1	**Data Access and Integration**	The Metadata Repository must ensure seamless integration of structured metadata into our relational database systems.
2.2		The application should ensure compatibility with a range of database technologies, focusing on lightweight integration and optimal performance.
3.1	**Transformation and Mapping**	The Metadata Repository must offer capabilities for defining and applying complex data transformations, mappings, and functional dependencies to the parsed metadata.
3.2		Users should be able to define custom transformations leveraging the underlying framework capabilities without needing in-depth technical knowledge.
4.1	**Security and Data Access Control**	The application must implement robust security measures, including authentication and authorization, to manage access to sensitive metadata.
5.1	**Scalability and Performance**	The Metadata Repository should be scalable, capable of handling large volumes of metadata and high levels of concurrent access without performance loss.

5.2		It must incorporate optimization techniques for enhancing query performance and ensuring efficient metadata storage and retrieval.
6.1	**Integration and Extensibility**	The application should seamlessly integrate with existing systems and technologies, utilizing the extensibility of its underlying framework.
6.2		The Metadata Repository must allow for the addition of new data access methods, transformation capabilities, and database technology support, adapting to evolving metadata management needs.

Table 1 Functional requirements

3.3 Software Development Strategy

There exist two primary approaches to bridge technical gaps and leverage data more effectively: the selection of third-party applications and in-house application development. Each strategy has its own set of advantages and considerations, necessitating a careful comparison to determine the most suitable approach for an organization's specific needs. in our context, the focus is on evaluating the theoretical viability of these strategies rather than their practical implementation.

3.3.1 Selection of Third-Party Applications

Selecting appropriate third-party applications is crucial for enhancing the functionality and efficiency of our systems. This process involves a thorough evaluation to ensure that the chosen external software aligns with our system's objectives and meets its technical specifications. The right third-party application will integrate seamlessly into our existing infrastructure, support our operational goals, scale with our growth, offer robust security features, and provide reliable vendor support.

Teradata Data DNA [11] could serve as an exemplary candidate for integration, particularly if our focus is on bolstering data analytics capabilities. This application is adept at handling large volumes of data and offers robust data integration, management, and warehousing capabilities, making it a prime choice for enterprises aiming to exploit big data analytics. Additionally, it provides advanced analytical tools that foster deep insights, facilitating more informed decision-making and predictive analytics.

3.3.2 In-House Application development

In contrast, in-house application development offers full ownership and control over the software, allowing for tailor-made solutions that precisely meet an organization's requirements.

3.3.2.1 API Integration Techniques

BI systems are inherently complex and designed to accommodate the vast and varied data needs of modern enterprises. To enhance flexibility and enable seamless integration with other applications, BI platforms provide a wide array of SDK and APIs. These tools are essential for customizing, extending, and integrating the BI system with existing organizational workflows and external applications. For instance, to showcase the variety of tools available for developers to integrate and extend the functionality of BI systems.

SAP BusinessObjects Examples:

- SAP BI Semantic Layer Java SDK [12]: Facilitates manipulation of the semantic layer, enabling the creation of contextual data models.
- SAP BusinessObjects BI Platform Java SDK [13]: Provides Java libraries for BI platform management, covering user, report, and document management.
- SAP BusinessObjects RESTful Web Service SDK [14]: Offers a RESTful interface for BI platform interaction, supporting report viewing, scheduling, and distribution.

MicroStrategy Examples:

- MicroStrategy REST API [15]: Allows RESTful access to BI capabilities, enabling data querying, report generation, and user management in custom applications.
- MicroStrategy mstrio [16]: A REST wrapper that eases MicroStrategy API interactions, streamlining the integration of BI insights into applications.

3.3.2.2 Authentication and Security

To demonstrate the process of connecting to BI systems, I have developed pseudocode utilizing the MicroStrategy REST API [15] as a case study. This example specifically showcases the authentication process, a critical step in establishing a connection with the BI system to execute data queries, as detailed in the supplementary sections. A noteworthy aspect of the code is the incorporation of a certificate in the login function. This requirement stems from the BI system's extensive interaction with various components, leading to significant server-to-server traffic. The use of TLS+- encryption to secure this communication. Consequently, the inclusion of a certificate is not just a best practice but a necessity to ensure that all data traffic across servers remains encrypted, upholding a secure operational environment.

```
# Semi-Pseudocode for Interacting with MicroStrategy REST API

# Parameters
url = 'https://mstr/MicroStrategyLibrary/api/'
login_mode = 1  # Standard login
certificate_path = "path/to/your_certificate.crt"

# Function to login and obtain authToken and cookies
def login(base_url, login_mode, certificate_path):
    username = input("Enter your username: ")
    password = input("Enter your password: ")

    # Setup payload and headers
    payload = {'username': username, 'password': password, 'loginMode': login_mode}
    headers = {"Content-type": "application/json"}

    # Make POST request to login endpoint
    response = requests.post(f"{url}auth/login", json=payload, headers=headers, verify=certificate_path)

    if response.status_code == 200:
        authToken = response.headers.get('X-MSTR-AuthToken')
        cookies = {"JSESSIONID": response.cookies['JSESSIONID']}
        print("Login successful! Token: ", authToken)
        return authToken, cookies
    else:
        print("Login failed. Status: ", response.status_code, "Message: ", response.text)
        return None

# Function to set headers for further API requests
def set_headers(authToken):
    return {'X-MSTR-AuthToken': authToken, 'Content-Type': 'application/json', 'Accept': 'application/json'}

# Main
if __name__ == "__main__":
    authToken, cookies = login(url, login_mode, certificate_path)
    if authToken:
        headers = set_headers(authToken)
        # Further API requests can be made using authToken and headers
```

3.3.2.3 Data Exchange Formats

REST APIs are commonly utilized for data exchange in BI systems, employing HTTP/HTTPS protocols to efficiently send and receive data, typically in JSON or XML formats. However, the landscape of SDKs and APIs is diverse, and not all technologies conform to the REST model or explicitly utilize familiar data exchange formats like JSON or XML [17]. A prime example of this diversity is demonstrated using the SAP BI Semantic Layer Java SDK [12] for information extraction, as showcased by the author. This SDK leverages structured data models that resonate with the organizational principles seen in widely recognized data formats. This compatibility facilitates the seamless serialization of object models into structured formats such as JSON or XML when data needs to be communicated between systems, transmitted over networks, or stored persistently, thus preserving data integrity and ensuring broad accessibility.

In the pseudocode example provided, the author introduces a DTO, a common construct in OOP with Java, designed for data reading and writing. This DTO, structured around key-value pairs, aligns naturally with the JSON format, allowing for straightforward conversion.

```
//pseudocode for extracting table information
class TableExtraction {
  // Function to extract and compile table information into DTOs
  function getTable(dataFoundation) {
    tableDTOS = new ArrayList() // Initialize list for table DTOs
    tables = dataFoundation.getTables() // Retrieve tables

    // Loop through each table to gather and set properties
    for table in tables {
      // Prepare table properties
      identifier = table.getIdentifier()
      name = table.getName()
      description = table.getDescription() ? table.getDescription() : ""
      aliasName = table instanceof AliasTable ? "true" : "false"
      aliasTable = aliasName == "true" ? ((AliasTable)table).getAliasedTable().getName() : ""
      qualifier = table instanceof DatabaseTable ? ((DatabaseTable)table).getQualifier() : ""
      derived = table instanceof DerivedTable ? "true" : "false"
      expression = derived == "true" ? ((DerivedTable)table).getExpression() : ""
      sourceShortName = derived == "true" ? ((DerivedTable)table).getSourceShortName() : ""
      federated = table instanceof FederatedTable ? "true" : "false"
      // Create a TableDTO with the properties
      tableDTO = new TableDTO(identifier, name, description, qualifier, aliasName, aliasTable, derived,
expression, sourceShortName, federated)
      // Add the DTO to the list
      tableDTOS.add(tableDTO)
    }
    return tableDTOS // Return the compiled list of TableDTOs
  }
```

3.3.2.4 Error Handling and Monitoring

This section aims to outline best practices for identifying, handling, and logging errors, as well as strategies for effective system monitoring to ensure reliability, performance, and user satisfaction.

Error handling is a critical facet of BI system architecture, designed to ensure that systems remain functional and responsive in the face of unexpected issues. It involves the implementation of comprehensive mechanisms for catching errors, particularly around critical operations that involve external API interactions or complex data processing tasks. The aim is to gracefully manage failures by providing clear and informative error messages that aid in diagnosing and resolving issues. Moreover, systematically logging errors with detailed information, including when and where they occurred, is essential for effective troubleshooting. Incorporating retry mechanisms for handling transient errors, especially those related to network communications or temporary service disruptions, can also enhance system robustness, employing appropriate strategies for exponential backoff or delay between attempts.

Monitoring plays a crucial role in the continuous observation and analysis of system operations. It helps in detecting performance bottlenecks, preempting potential issues, and acquiring insights that drive continuous improvement [18]. This encompasses tracking vital performance indicators such as response times, throughput, and resource utilization to identify trends and detect anomalies. Establishing alerting mechanisms based on specific thresholds or patterns of anomalies ensures that system administrators and relevant stakeholders are promptly notified of potential issues. Conducting regular health checks on key system components, including databases, APIs, and external services, guarantees their operational integrity. Additionally, leveraging log analysis tools to examine error logs and operational data can unearth common issues, security concerns, or patterns warranting attention. It's also beneficial to integrate mechanisms for capturing user feedback, which can reveal usability issues or areas for enhancement not immediately evident through system metrics.

3.3.3 Selecting Hosting Solution

The pivotal task in integrating a third-party BI application involves selecting the most suitable server. Traditionally, the application server is the favored choice for deployment, primarily because it serves as the central hub for BI systems, managing all computational tasks, processes, and essential SDK services like RESTful services.

One notable benefit of this approach is the enhancement of security. By hosting the application on a single server, it reduces the need for inter-server communication, thereby boosting efficiency and eliminating the risk associated with data encryption during transit. However, it is important to note that this configuration does not completely obviate the need for data queries from the database to fully trace data lineage.

Furthermore, the integration of a third-party application necessitates a careful reevaluation of server capacity. It may require resizing of the application servers to ensure they can effectively support the additional workload introduced by the new application. This resizing is critical to maintain smooth operations and optimal performance across the system.

3.4 Minimalistic Based Architecture for BI Platforms

The simplest configuration for a BI system typically involves at least three servers; however, this setup is often not used due to several drawbacks. For instance, it lacks a dedicated backup solution, leaving critical data vulnerable since all components reside on a single server. Additionally, this approach presents substantial security risks, as it omits additional protective layers that would typically shield the server during access and interaction.

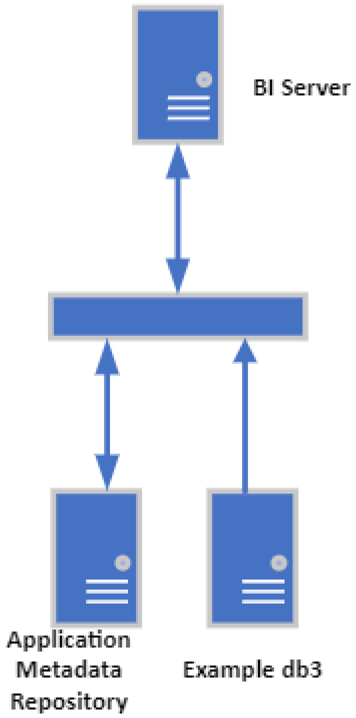

Figure 3 Minimalistic Based Architecture for BI Platforms

3.5 Classical Architecture for BI Platforms

The diagram represents an implementation of a three-tier architecture [19] model, tailored for a BI platform. This is the most common way to setup and BI platform and is often used as proof of work. This means that this architecture is mainly used to test out different BI systems. There are many advantages of separating the BI system into a 3 layer This configuration is posited as the foundational requirement for the application's operation, signifying the essential components despite its seemingly intricate structure.

Client Tier: This is identified by the "Client Tools/Management Server" and serves as the interface through which users interact with the BI platform. It hosts the user interfaces and client-side applications.

Middle Tier: This layer is integral to the architecture, consisting of a "Web Server" and an "Application Server". The web server's primary role is to handle HTTP requests and deliver responses, acting as the intermediary between the client and the application server. The application server, on the other hand, is responsible for executing the core business logic of the BI platform and facilitating interactions with the database systems. Data Tier: The lowermost layer in the diagram, encompassing the "Application Metadata Repository", databases "Example db3", "Example db2", "Example db1", and "NAS" for Network-Attached Storage, is where the data is stored, managed, and retrieved. This tier ensures that the structured and unstructured data essential to BI processes are effectively organized and accessible.

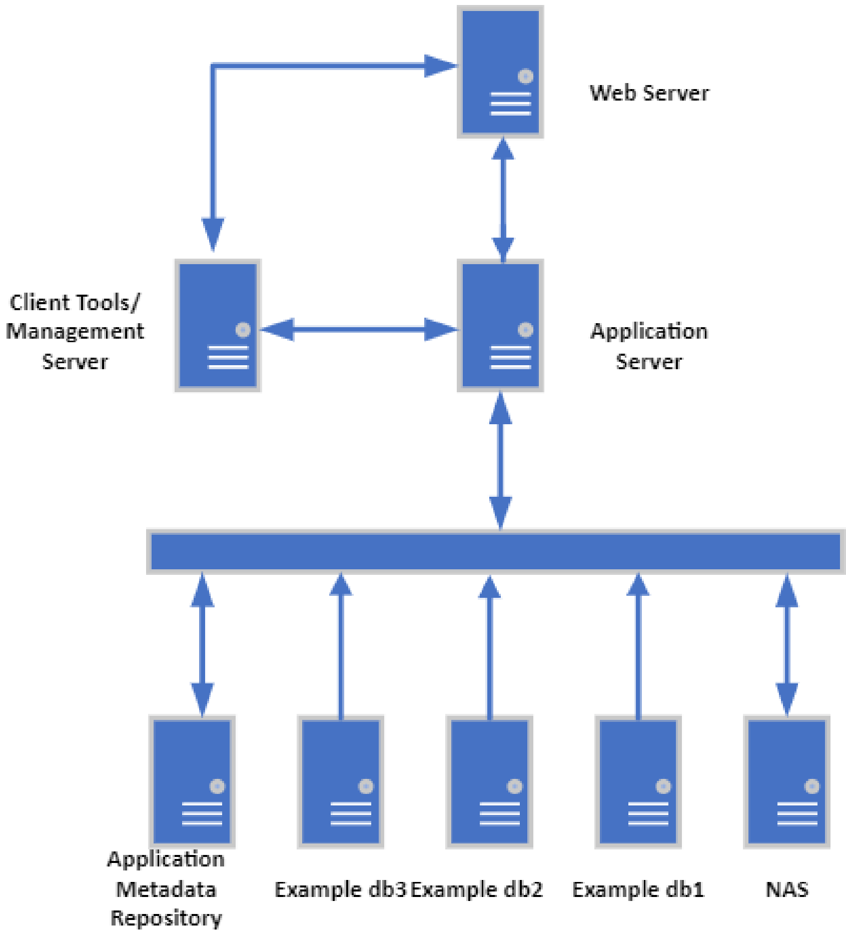

Figure 4 Classical Architecture for BI Platforms

3.6 Clustered Architecture for BI Platforms

This section outlines a clustered architecture [20] specifically designed for BI platforms, emphasizing scalability through the replication of foundational structures. In enterprises where BI systems are crucial, adopting a clustered configuration is essential for ensuring high availability that supports critical business operations. The architecture, as illustrated, utilizes black lines to indicate the capability of components to act as backups for one another. This design facilitates the creation of duplicate servers in alternate logical locations, significantly bolstering the system's security and stability. Such strategic redundancy is indispensable, as it guarantees continuous service while preserving data integrity and accessibility under various circumstances.

Incorporated into this architecture is the use of load balancers [21], which play a dual role in enhancing both operational efficiency and security. The primary function of a load balancer is to distribute incoming network traffic evenly across the server cluster, optimizing resource utilization and promoting efficient load management. More critically, the load balancer serves as a protective barrier for the web servers, concealing their IP addresses from external exposure. This security measure is a key component of contemporary best practices for web applications, aligning with recommendations from BI platform vendors. Consequently, the integrated approach of using both redundant clustered architecture and strategic load balancing ensures robust, uninterrupted availability of BI services, making it a recommended strategy for deploying resilient and scalable BI systems.

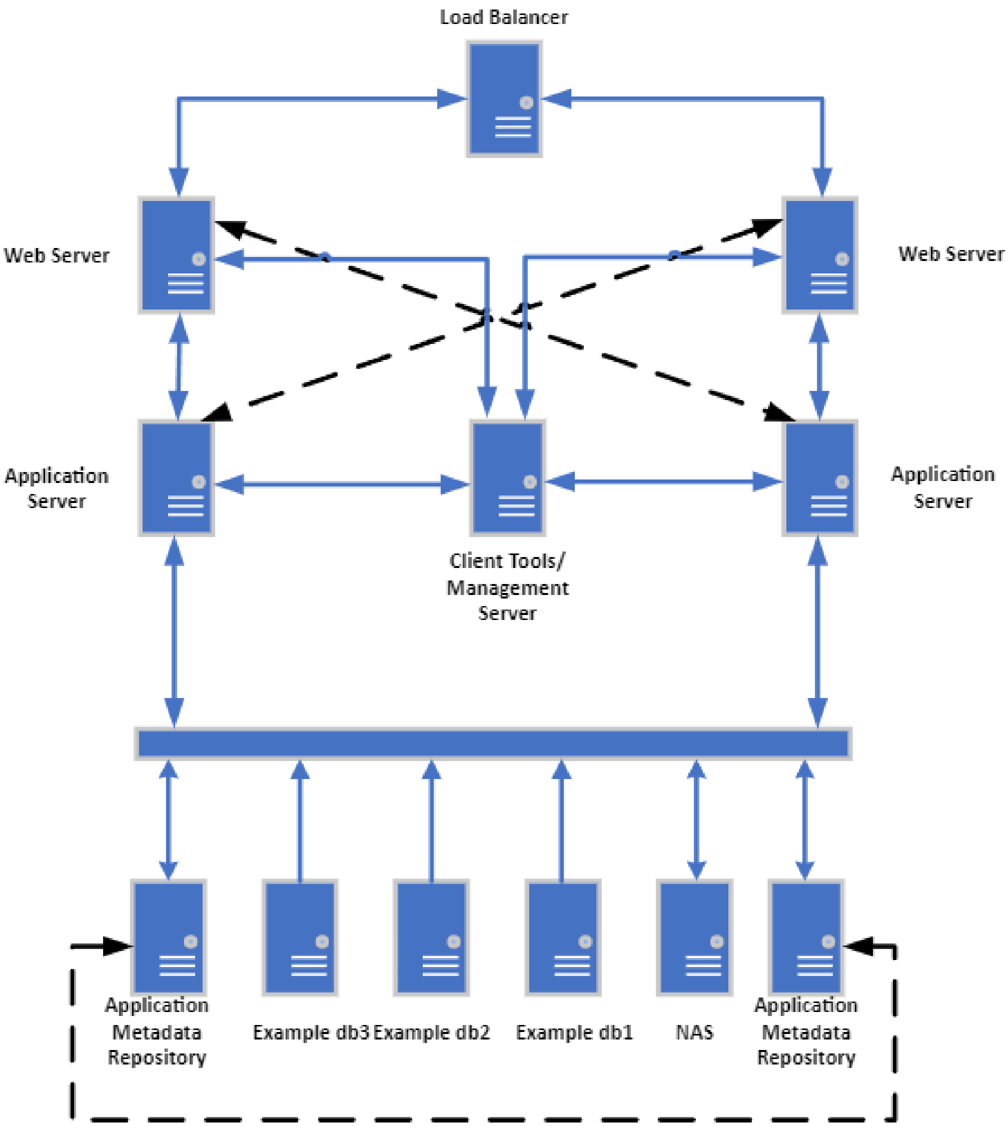

Figure 5 Clustered Architecture for BI Platforms

4 Theoretical Frameworks and Analytical Challenges

This section explores the intricate relationship between established theoretical frameworks and the practical complexities that arise when integrating a lineage layer into BI systems. It examines the foundational principles that guide the creation, incorporation, and enhancement of these systems, while also confronting the inherent challenges that come with safeguarding security, enabling scalability, and maintaining high performance.

4.1 Theoretical Architecture For Linage Layer

The creation of a comprehensive data lineage layer is encapsulated within an intricate Component Diagram designed by the author. This diagram serves as a strategic framework, mapping out the critical components required to forge connections between data's origin and its end-use in BI reporting. It highlights 'Documents and Specifications', 'ETL', 'BI System', and 'Source DBs' as the pivotal stages where data resides and requires detailed parsing to trace its lineage.

Within this framework, the diagram unites key analytical instruments: the 'SQL Parser', 'ETL Scanner', 'BI Scanner', and 'DB Scanners'. These tools are integral to extracting and distilling the data, piecing together metadata and insights crucial for constructing a lineage layer that is both informative and accurate.

At the system's core lies the 'Application Repository', a vault of valuable metadata and parsed data. This repository is the lifeline to the 'Service Class', which undertakes the logical operations necessary for constructing the lineage layer, knitting together various data strands into a coherent narrative.

The 'Service Class' employs 'Data Models' as temporary constructs to hold and manage the data. These models become the foundation upon which the 'Visualizer' later builds, transforming complex data paths into a visual lineage that enhances understanding and interaction.

Finally, 'UI Controllers' act as the user-facing element of the system, translating user inputs into actions. They represent the operational facet of the user interface, allowing users to engage with the system and access the lineage layer, effectively bridging the gap between backend data processes and frontend user experience. This comprehensive system thus enables a thorough understanding of data provenance, supporting effective data governance and compliance.

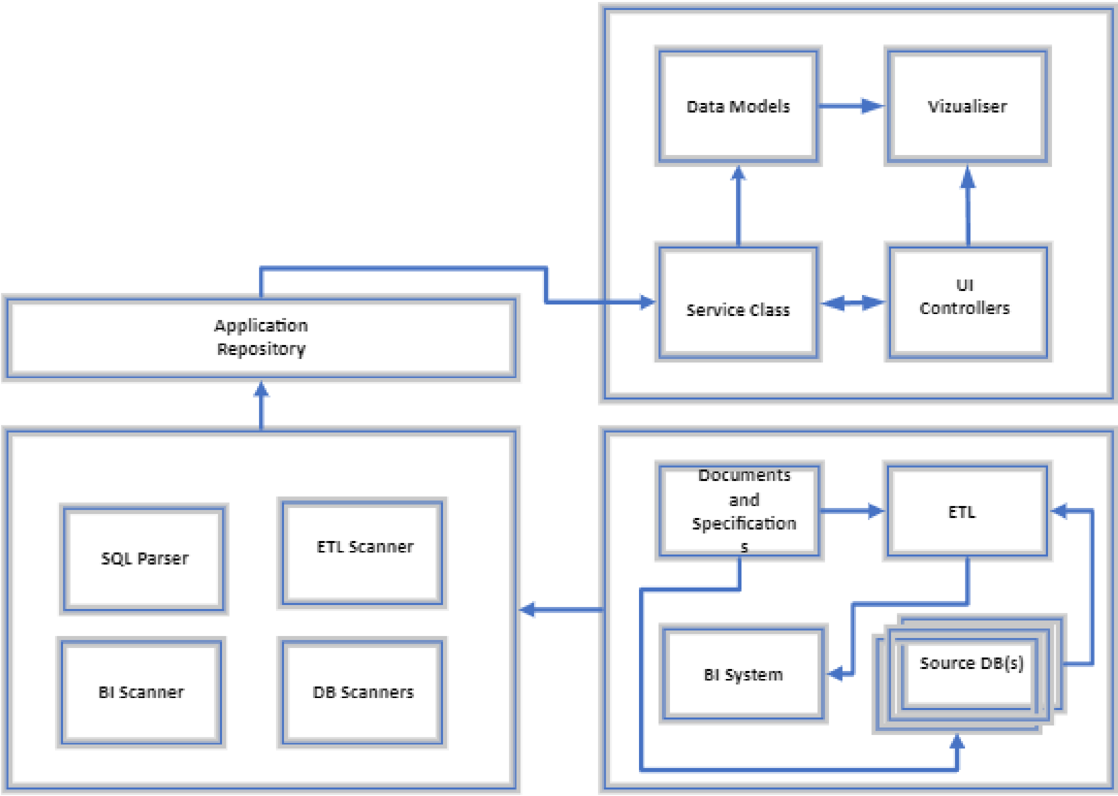

Figure 6 Framework Components

4.2 Theoretical Challenges

Integrating a new application into a BI platform presents several theoretical challenges that need to be addressed to ensure successful implementation.

Challenge	Description	Proposed Solution
Dependency management	Reliance on multiple database and BI application vendors can lead to functionality issues due to lack of backward compatibility.	Implement direct parsing from metadata repositories to manage database structure variations and maintain reliability amidst SDK updates. Emphasize modularity, reusability, and extensibility.
Data Structure Storage	BI applications use diverse data types, and systems like MicroStrategy and Business Objects differ significantly in their data structuring.	Develop a flexible database structure capable of handling various metadata types, including both descriptive and structural metadata, to enhance data extraction and integration. [22]
Framework Construction	The complexity of real database systems requires users to understand the schema or BI semantic layer, which can be daunting for unfamiliar users.	Utilize schema summarization techniques [23] to simplify user interaction with complex databases. Address legal concerns such as GDPR to focus only on technical data parsing.
Integration Challenges	Integrating a new solution into an existing complex IT environment may face resistance due to the scale and scope of changes required.	Apply systems theory [24] to demonstrate the long-term benefits of improved data management and system interoperability, justifying the efforts involved in overcoming initial resistance.

Regulatory Compliance	Compliance with data protection laws like GDPR [1] is critical, particularly concerning the handling of sensitive personal data.	Focus on parsing only the technical data and formats, not the content itself. This approach ensures compliance with GDPR by avoiding the use of sensitive data and focusing on metadata structures.

Table 2 Theoretical Challenges

4.3 Security Considerations in Data Integration

When discussing the security of BI systems, several key factors must be considered, including the intended setup, the data stored within the system, the authorized users, and the governance rules for data processing. Additionally, technical complexities arise due to the diverse applications integrated within the BI framework, necessitating a holistic approach to system operation. For systems employing classic or clustered architectures, all communications should be secured using TLS protocols [25]. Although encryption is a critical component of securing data, it is also resource-intensive, requiring frequent updates and installations of certificates on each server.

An alternative method to alleviate some of these burdens is load balancer offloading [21], which can enhance performance by reducing the load on individual servers. To illustrate the trade-offs between these approaches, the author has conducted a comparative analysis, highlighting the strengths and weaknesses of each. Additionally, the adoption of game theory can offer valuable insights into strategic decision-making processes in security implementation.

Furthermore, employing comprehensive risk management strategies, as outlined in the reference [26], is crucial for developing effective security measures. These strategies should encompass encryption technologies to protect data at rest and in transit, robust access control mechanisms to manage data access, and secure communication protocols to safeguard data during transmission. Together, these elements form the backbone of a well-rounded security framework for BI systems.

4.4 Scalability and Performance Optimization

When discussing the performance of the metadata parser, it is crucial to consider both the time complexity of the underlying framework and the volume of data managed by the company. Typically, as data volume increases, so does processing time, making efficient data handling crucial for maintaining system performance. The frequency with which the repository that this parser populates must be updated to reflect real-world scenarios highlights the importance of performance. More frequent refreshes enable better visibility and accuracy in data lineage, making efficient processing a priority.

To achieve the highest possible efficiency in processing, implementing design patterns from the [27]GoF can be particularly effective during framework development. These patterns aid in creating a flexible and scalable architecture that can adapt to varying data loads without compromising on speed.

For optimal performance and scalability, the framework should excel in managing large and fluctuating data volumes. This involves a design that minimizes the need for clustering by incorporating advanced multi-threading capabilities, as noted in reference [28]. Such capabilities are essential for ensuring that the application remains responsive and maintains high throughput, even under substantial data loads.

A critical component of this design strategy involves sophisticated state management within the application. It is crucial to prevent redundant executions of the same query within a single operation cycle. To address this, the system should support the parallel execution of queries with a robust mechanism for managing dependencies among them. This setup allows multiple queries to run simultaneously, while the system smartly coordinates these operations. For instance, if one query depends on the outcome of another, the system ensures seamless integration of results, thereby optimizing the overall execution flow and enhancing performance.

Furthermore, the adoption of techniques such as caching frequently accessed data and employing efficient data indexing can further enhance system responsiveness and reduce latency. By strategically managing how data is accessed and processed, the framework can deliver quicker insights and support more complex data analyses without slowing down.

Overall, by leveraging advanced programming paradigms, thoughtful architecture design, and state-of-the-art data handling techniques, a BI framework can dramatically improve its performance metrics. This improvement enables businesses to enjoy rapid and reliable data processing capabilities that are essential for effective decision-making and timely updates to the repository, thereby providing comprehensive visibility into data lineage.

4.5 Lessons Learned

Integrating BI with additional data management applications is essential for enterprises aiming to enhance their analytical capabilities and operational efficiency. Here are some key insights from experiences in the theoretical integration of such tools.

Choosing the right partners or tools for integration into your system is crucial for maximizing the value from your BI platform. It's important to recognize that not all applications and BI tools are the same. The chosen software and partners must align well with the enterprise's existing landscape and meet both functional and non-functional requirements.

The technical challenges of integrating different systems should not be underestimated. Compatibility issues between the BI platform and third-party applications can arise, often requiring customized solutions. Alternatively, developing these applications in-house is an option, though it tends to be more costly and time-consuming.

Effective use of BI platforms typically necessitates employee training. To fully leverage the benefits of new features, comprehensive training is essential. This not only helps in extracting maximum value but also mitigates resistance to change. Demonstrating clear benefits can further help in overcoming such resistance.

4.6 Best Practices: Guiding Principles and Theories

The successful integration of BI systems with third-party applications is grounded in established best practices, which draw upon foundational principles and theories from information systems, software engineering, and data science. Key among these is the principle of data governance, which is critical for maintaining data quality and consistency across platforms. This principle ensures that data remains accurate, reliable, and secure, serving as a trustworthy foundation for decision-making processes.

In addition to data governance, agile methodologies [29] play a crucial role in the integration process. These methodologies advocate for flexible, iterative development and deployment, enabling organizations to adapt quickly to changes and refine their systems based on real-world feedback and evolving requirements. This approach facilitates continuous improvement and ensures that BI systems remain responsive to the needs of the business.

Furthermore, the concept of data democratization [30] is pivotal, promoting the idea that BI insights should be widely accessible and actionable across the organization. This theory supports the empowerment of non-technical users to engage with BI tools and make informed decisions, thereby fostering a data-driven culture within the organization.

By embracing these guiding principles and theoretical frameworks, organizations can significantly bolster the efficiency, effectiveness, and overall value of their integrated BI systems. This holistic approach not only enhances technical integration but also aligns BI initiatives with strategic business objectives, ensuring that BI systems contribute meaningfully to organizational success.

5 Results

The theoretical implementation of a metadata parser within BI systems, as explored in this thesis, unveils significant findings across various dimensions of data management and utilization. These results highlight the potential for improved decision-making capabilities, enhanced data quality and transparency, and notable business impacts stemming from the integration of such third-party applications.

5.1 Improved Decision-Making Capabilities

The strategic integration of a metadata parser with BI systems marks a transformative advancement in how organizations approach decision-making. This integration facilitates a robust, granular view of data lineage that delineates the origins, transformations, and final applications of data streams. By equipping decision-makers with an in-depth, accurate understanding of data’s provenance and modifications, the system ensures a high degree of data context and reliability. This comprehensive visibility is pivotal in boosting the quality of decisions, as it fosters more informed, precise, and prompt decision-making processes. Stakeholders can place enhanced trust in the data's integrity and depth, enabling them to make decisions based on a solid foundation of verified information.

Moreover, the detailed tracking capabilities of the integrated metadata parser illuminate the entire data transformation journey. This visibility into each transformation phase allows for the identification and analysis of inefficiencies and bottlenecks, which might otherwise impede operational fluidity. Addressing these issues not only enhances operational efficiency but also catalyzes the refinement of strategic and tactical maneuvers within the organization. Ultimately, this leads to optimized business processes and a competitive edge in rapidly changing markets. Enhanced metadata management also supports regulatory compliance and risk management, adding another layer of strategic advantage to the organization [31].

5.2 Enhanced Data Quality and Transparency

Implementing a lineage layer is crucial for enhancing data quality and transparency across the BI landscape. This tool automates the documentation and monitoring of data sources and their transformation processes. Consequently, organizations can swiftly identify and rectify any data inaccuracies, inconsistencies, or outdated information. Such capabilities ensure that the data fed into analytics platforms is current, high-quality, and aligned with the organization's operational requirements. [32]

This proactive data management approach significantly bolsters the integrity of the data pool. As high-quality data is vital for effective BI systems, improvements in data accuracy lead to more dependable analytics outputs. Moreover, automated tracking promotes a culture of transparency within the organization. It provides stakeholders, from executives to operational staff, with a transparent view of the data processing and refinement procedures across various BI systems. This transparency fosters deeper trust in the analytics results and the strategic insights derived, thereby enhancing decision-making confidence.

Additionally, this improved transparency is essential for compliance and auditing. With detailed, automated records of all data actions and pathways, organizations can more readily demonstrate adherence to applicable data protection regulations and standards. This feature is especially critical in sectors with stringent regulatory requirements, such as finance, healthcare, and public services.

To illustrate this concept, the author has developed a mockup. The first view displays the explorer view before any query has been executed. The second view is what the drilling process would look like. It is important to note that the data shown in these visualizations is fictional, tailored to highlight the potential value of such information.

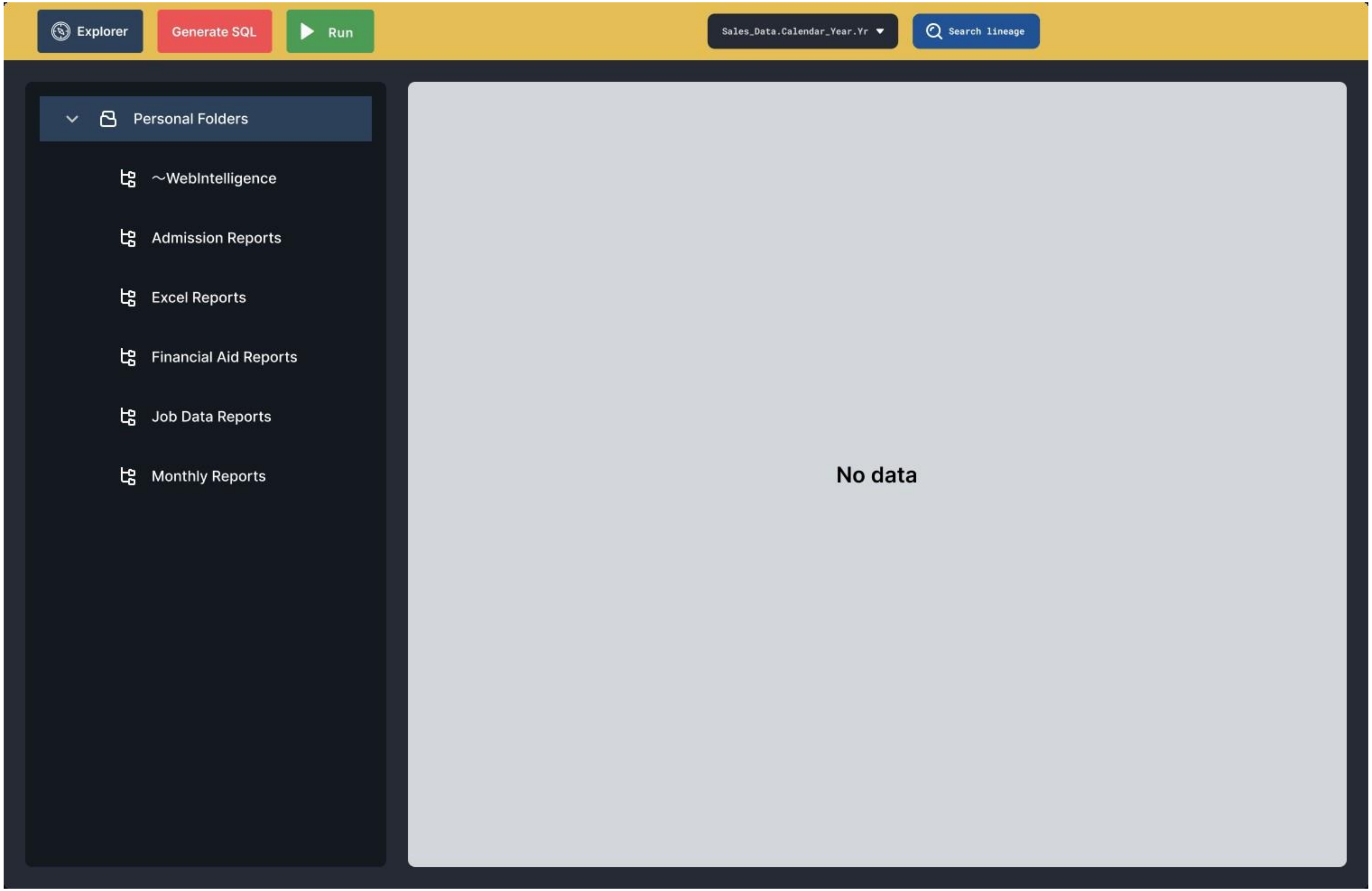

Figure 7 Lineage Layer Explorer

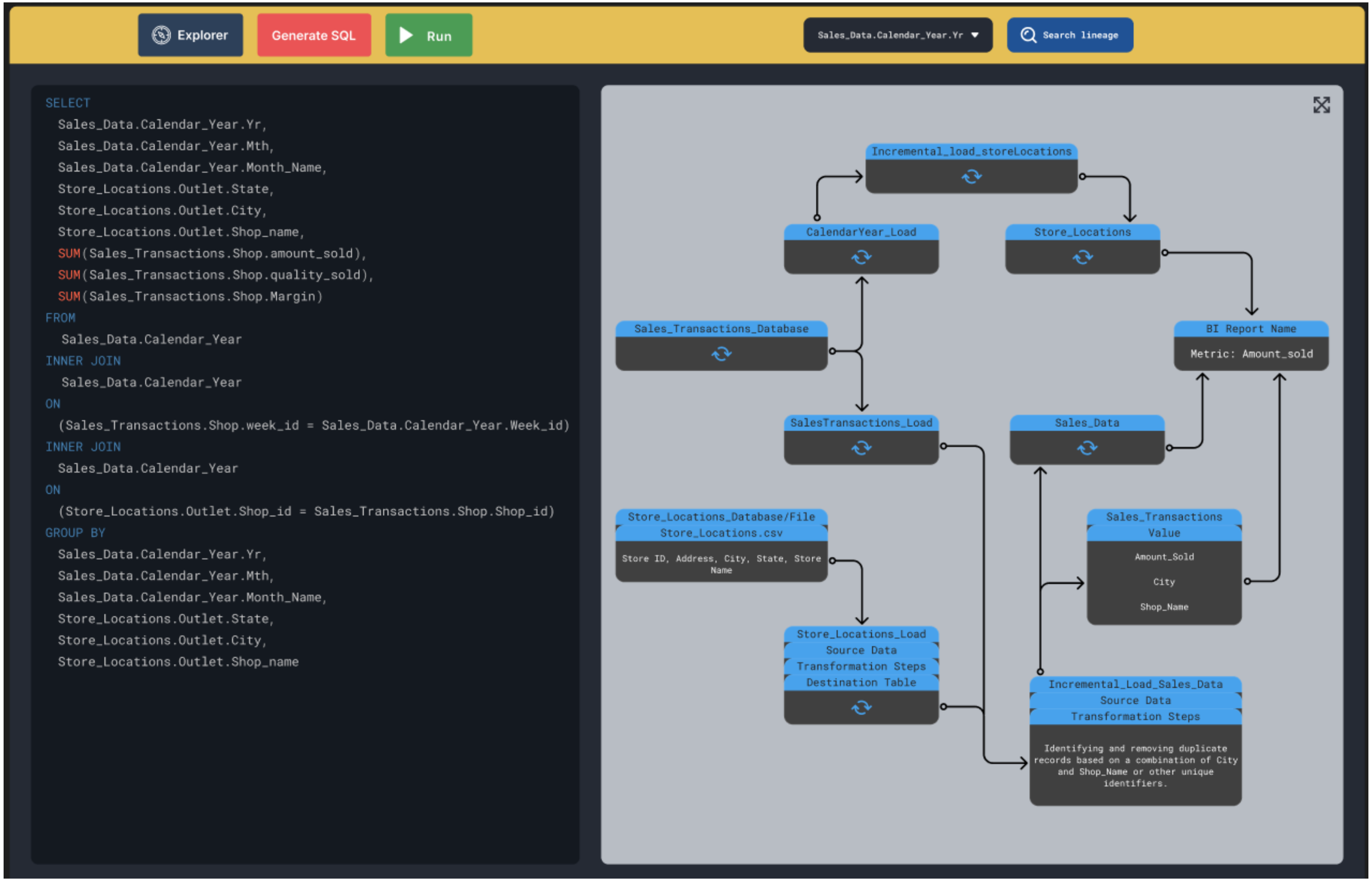

Figure 8 Lineage Layer

5.3 Business Impact of Integration

The integration of the lineage layer into BI systems carries profound implications for business operations, driving significant value from enterprise assets. By elucidating the data's pathway from source through ETL processes to BI databases, organizations can unlock hidden value, leveraging their data assets more effectively and efficiently. This clarity facilitates impact analysis, allowing businesses to understand the ramifications of data changes and to conduct fact-based governance with minimal effort yet maximal impact.

Moreover, this integration has the potential to reduce analysis and development time significantly. By automating and streamlining the data governance processes, businesses [31] can expedite the development of new analytics capabilities and the refinement of existing ones. This acceleration not only enhances agility and responsiveness to market changes but also frees up valuable resources for innovation and strategic initiatives, ultimately leading to improved competitiveness and market positioning.

5.4 Future Directions

The theoretical implementation of a lineage layer within BI systems has unveiled a plethora of avenues for future exploration in data management and analytics, signaling a transformative shift in enhancing decision-making processes, data quality, and business impact through advanced BI system integration. Among the most promising of these avenues is the integration of AI and ML technologies with the lineage layer.

Integrating AI and ML with the lineage layer could fundamentally alter how businesses analyze and leverage their data. The development of predictive models that utilize insights gleaned from metadata to forecast trends, pre-emptively identify data quality issues, and recommend workflow optimizations represents a proactive approach to data management. This shift towards predictive analytics and proactive data quality management could significantly reduce the time and resources traditionally allocated to data cleaning and preparation, thereby enabling organizations to concentrate on extracting and acting upon valuable insights.

Summary

This thesis explores the integration of a lineage layer into BI systems, addressing architectural, operational, and governance challenges. The problem centers on how to seamlessly incorporate third-party applications into proprietary BI environments without compromising system integrity. The problem arises since data goes though various transformations and can cause inconsistencies. It aims to enhance organizational efficiency and decision-making through improved data governance and security measures. The methodology involves a theoretical approach, using a case study to model the integration of a lineage layer and its implications on existing BI architectures. Emphasis is placed on maintaining compatibility with current data structures and workflows, while ensuring minimal disruption to operations. Results indicate that integrating a lineage layer can significantly enhance decision-making capabilities by providing deeper insights into data origins and transformations, aiding in the detection of inconsistencies and optimizing operational strategies. Additionally, robust security protocols are required to safeguard sensitive metadata and comply with regulations like GDPR. In conclusion, the thesis contributes to the discourse on BI system enhancement by demonstrating that strategic third-party integrations can foster a more informed organizational culture and improve overall data management, albeit with careful consideration of architectural and security implications.

References

[1] GDPR.EU, “What is GDPR, the EU’s new data protection law?.,” [Online]. Available: https://gdpr.eu/what-is-gdpr/.. [Accessed 2024].

[2] “Business Intelligence Platform Administrator Guide,” SAP, 2 03 2024. [Online]. Available: https://help.sap.com/doc/6dba07dcab5f44e0a29008d4a7c648f6/4.2.5/en-US/sbo42sp4_bip_user_en.pdf.

[3] “MicroStrategy Products and Components,” MicroStrategy, [Online]. Available: https://www2.microstrategy.com/producthelp/Current/InstallConfig/en-us/Content/MicroStrategy_products_and_components.htm. [Accessed 02 03 2024].

[4] Wiiisdom, “Wiiisdom,” [Online]. Available: https://wiiisdom.com. [Accessed 2024].

[5] C. S. P. G. Barbara Dinter, “A Stakeholder Lens on Metadata Management in Business Intelligence and Big Data – Results of an Empirical Investigation,” 2015.

[6] L. H. O. A. Dilovan Haji, “Big Data: Management, Technologies, Visualization,,” 2020. [Online]. [Accessed 2024].

[7] X. G. H. W. Guannan Liu, “An Investigation Of Identity-Account Inconsistency In Single Sign-On,” 2021. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3442381.3450085. [Accessed 2024].

[8] S. M. S.-M. M. H. H. Mahmoud Mohammad Ahmad Ibrahim, “Managing Quality Assurance Challenges Of Devops Through Analytics,” 2019. [Online]. [Accessed 2024].

[9] J. Riley, “Understanding Metadata,” vol. 23, pp. 7-10, 2017.

[10] A. K.-G. Raphael Grytz, “Business Intelligence and Analytics Cost Accounting: An Empirical Study of the Perceptions of Stakeholders and Their Implications,” 2018. [Online]. Available: https://www.semanticscholar.org/paper/Business-Intelligence-and-Analytics-Cost-An-Study-Grytz-Krohn-Grimberghe/d03f40d632d10a80b6ae9de50fd86709ef47c39a. [Accessed 15 04 2024].

[11] Teradata, “Teradata Launches New ‘DataDNA’ Data Forensics Tool,” Teradata, 2020. [Online]. Available: https://www.teradata.com/press-releases/2020/teradata-launches-new-datadna-data-forensics-tool. [Accessed 2024].

[12] S. SE, “SAP BI Semantic Layer Java SDK API,” 2015 . [Online]. Available: https://help.sap.com/doc/javadocs_bip_41/4.1/en-US/slsdk/en/overview-summary.html. [Accessed 25 03 2024].

[13] S. SE, “SAP BusinessObjects Business Intelligence Platform Java SDK,” [Online]. Available: https://help.sap.com/doc/javadocs_bip_42/4.2/en-US/bip/en/overview-summary.html. [Accessed 25 03 2024].

[14] S. SE, “REST API Quick Reference,” SAP SE, [Online]. Available: https://help.sap.com/docs/SAP_BUSINESSOBJECTS_WEB_INTELLIGENCE/58f583a7643e48cf944cf554eb961f5b/7da2e2d66f701014aaab767bb0e91070.html?version=4.2.4. [Accessed 25 03 2024].

[15] Microstrategy, “Introduction to REST API,” [Online]. Available: https://microstrategy.github.io/rest-api-docs/. [Accessed 25 03 2024].

[16] Microstrategy, “Mstrio-py,” [Online]. Available: https://www2.microstrategy.com/producthelp/Current/mstrio-py/index.html. [Accessed 25 03 2024].

[17] A. ARCURI, “restful Api Automated Test Case Generation With Evomaster,” 2019. [Online].

[18] T. R. A. A. R. Silvia Esparrachiari, “Tracking And Controlling Microservice Dependencies,” 2018. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/3267118.

[19] www.geeksforgeeks.org, “Three-Tier Client Server Architecture in Distributed System,” 2023. [Online]. Available: https://www.geeksforgeeks.org/three-tier-client-server-architecture-in-distributed-system/.

[20] “MicroStrategy Intelligence Server Clustering FAQ,” MicroStrategy, 2024. [Online]. Available: https://community.microstrategy.com/s/article/KB16018-MicroStrategy-Intelligence-Server-Clustering-FAQ?language=en_US&_gl=1*djk98d*_ga*MTEzMjEyOTg0LjE3MDc3MjQ2ODE.*_ga_0C9LVNZBZY*MTcxNDA3MDI0Ny4xMi4xLjE3MTQwNzAzMDUuMC4wLjA.. [Accessed 2024].

[21] S. R. M. Z. R. R. Z. Omid H Jader, “A State Of Art Survey For Web Server Performance Measurement And Load Balancing Mechanisms,” 2019. [Online]. Available: https://www.researchgate.net/profile/Subhi-Zeebaree/publication/337972812_A_State_Of_Art_Survey_For_Web_Server_Performance_Measurement_And_Load_Balancing_Mechanisms/links/5f115b8092851c1eff183f6f/A-State-Of-Art-Survey-For-Web-Server-Performance-Measuremen. [Accessed 2024].

[22] R. H. I. V. C Quix, “Metadata Extraction and Management in Data Lakes With GEMMS,” 2016. [Online]. Available: https://csimq-journals.rtu.lv/article/view/1548. [Accessed 2024].

[23] H. V. J. Cong Yu, “Schema Summarization,” 2006.

[24] G. G. W. &. L. L. Constantine, Systems Theory, 2009.

[25] K. G. P. H. W. Hugo Krawczyk, “On the Security of the TLS Protocol: A Systematic Analysis,” in *Advances in Cryptology,* 2013.

[26] B. X. Guofeng Piao, “Risk Management Analysis of Modern Commercial Banks Using Behavioral Finance Theory and Artificial Neural Networks,” 2022.

[27] N. &. M. A. A. Qamar, “Impact of design patterns on software complexity and size,” *Mehran University Research Journal Of Engineering & Technology,* 2020.

[28] J. M. C. a. T. -W. H. K. -Y. Chen, “Multithreading in Java: Performance and Scalability on Multicore System,” in *Transactions on Computers,* pp. 1521-1534.

[29] H. A.-N. S. S. Samar Al-Saqqa, “Agile Software Development: Methodologies And Trends,” 2022. [Online]. Available: https://www.semanticscholar.org/paper/Agile-Software-Development%3A-Methodologies-and-Al-Saqqa-Sawalha/53436eeef7672a4b982ba8872491635cbf211f1a?p2df. [Accessed 2024].

[30] C. L. M. F. Hippolyte Lefebvre, “Data democratization: toward a deeper understanding,” 2021. [Online]. Available: https://www.researchgate.net/profile/Christine-Legner-2/publication/354906721_Data_democratization_toward_a_deeper_understanding/links/61539051fd7b3d1215599a07/Data-democratization-toward-a-deeper-understanding.pdf. [Accessed 2024].

[31] S. B. E. F. a. D. M. E. Bertino, “Trustworthy AI and Data Lineage,” vol. 27, pp. 5-6, 2023.

[32] T. Suja Cherukullapurath Mana, “A Study On Various Semantic Metadata Standards To Improve Data Usability,” 2019. [Online]. [Accessed 2024].

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis[1]

I Lukas Sulg

1. Grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis “The integration of Business intelligence and third-party applications for data management”, supervised by Ahti Lohk
   1.1. to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;
   1.2. to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.
2. I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.
3. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

08.05.2024

1 The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.