Eindhoven University of Technology

MASTER

Designing an integration system for financial data in Accounting Information Systems

van Son, N.L.J.

Award date:
2020

Link to publication

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain

Technische Universiteit
Eindhoven
University of Technology

Department of Mathematics and Computer Science

Designing an integration system for financial data in Accounting Information Systems

Master Thesis

Niek van Son

Supervisors:
Dr. Ekaterini Ioannou (Data Mining Group)
Dr. George Fletcher (Database Group)
Dr. Vlado Menkovski (Data Mining Group)
Frits van de Water (SpendLab)

Version 1.0

Eindhoven, January 2020

Abstract

IBM defines data integration as "The combination of technical and business processes used to combine data from disparate sources into meaningful and valuable information." [1] Spendlab Recovery is an organization that specializes in financial, administrative services and provides financial audits for its customers. These customers are organizations within the public and private sectors. In order to conduct an audit, the financial systems(s) of the organization are combined into a single unified view. From this single unified view, analysis files are created. Currently, the process of integrating financial systems is done without the support of specific tooling. Therefore, the process of the integration of financial systems is challenging and time-consuming.

Moreover, the process is not standardized. That is, two different data specialists, execute different steps, and make different assumptions while integrating (the same) financial systems. Furthermore, past integrations are not considered. Resulting in continuously having to redo all the steps when the same financial system is encountered. As a consequence, the quality of the resulting integration does not only differ between data specialists but also between integrations of the same data specialist. For these reasons, and the fact that the goal of each financial system is nearly identical, namely keeping track of the account and business processes of the organization. The question arose whether it is possible to ease up, standardize, or even fully automate the integration process implemented by Spendlab Recovery.

Based on a review of the literature concerning data integration, schema matching, and data exchange, a system is proposed. There exist two different approaches to querying a set of (heterogeneous) data sources, namely: schema first and no schema. The proposed system is based on the schema first approach. Schema first means that we create a mediator schema, to which the financial systems map. We are then able to query the mediator schema, which provides the semantically integrated view, to create the analysis file. Therefore, our system consists of four components, namely: Data warehouse, Semantic mapper, Data exchange, and a Graphical User Interface. The data warehouse provides a central storage and uses a mediator schema as its underlying model. Given the data warehouse and a financial system, the semantic mapping component proposes semantic mappings. During this thesis, two semantic mapping algorithms were created. When the proposed semantic mapping is deemed correct by the data specialist, then given the data warehouse, financial system, and semantic mapping, the data exchange component generates a query. This query transports the data from the financial system into the data warehouse. The correct semantic mappings can be re-used. As a result, financial systems of the same version can be integrated automatically. The Graphical user interface combines all the other components and makes it easier to use. The main contribution of this work is uniquely combining the four described components. The project resulted in a system that, according to the users, eases up the process, saves time and introduces a standard.

[1] https://www.ibm.com/analytics/data-integration

Preface

First of all, I would like to thank everyone that helped and supported me during this project. Secondly, I would like to thank my supervisor Ekaterini Ioannou for her support, collaboration, and feedback to help me finish my studies. Thirdly, I would like to thank my company supervisor Frits van de Water for his continues support and motivation. Also, I would like to thank David Michallik for his collaboration, explanations, and sharing ideas with me. Lastly, I would like to thank all the other people involved with SpendLab Technology for all the discussions we shared and giving me the opportunity for this project.

13 January 2020

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Data integration is becoming more and more important. Over the years, data integration has received significant attention. This rise in attention is partly explained by the fact that machine learning models are increasingly applied in real-world contexts, as well as organizations using different systems to manage their data. For these reasons preserving high data quality through a proper data integration system is becoming ever more critical. In this first chapter, we introduce the research problem. First, section 1.1 describes the company and the domain it operates in. Section 1.2 shortly introduces the self-developed platform. In section 1.3, we describe the research problem. After that, section 1.4 shows the scope and goals of the project. Followed by the research method in section 1.5. Finally, the outline of the thesis is given in section 1.6.

1.1 Spendlab Recovery

Spendlab Recovery [1] is an organization that specializes in administrative services and provides financial audits for its customers. These customers are organizations within the public and private sectors. Spendlab Recovery strives to increase the working capital of these organizations by recovering liquidity losses. Some examples of organizations are:

- ASML- ASML is the largest supplier in the world of photolithography systems in the semi-conductor industry `https://www.asml.com/en`.
- DSM- DSM is an international dutch chemical company. The head office of the company is located in Heerlen, The Netherlands `https://www.dsm.com/corporate/home.html`.
- Gemeente Amsterdam- Gemeente Amsterdam is the municipality of Amsterdam `https://www.amsterdam.nl/en/`.

Currently, the core focus of Spendlab Recovery consists of the analysis of the organization's financial data, detecting weaknesses in administrative processes, reporting the findings, and with permission of the organizations reclaim the undue payments from its vendors. The analysis is performed with unique self-developed software called Accounts Payable Recovery Analysis (APRA), which we will explain in section 1.2. The main goals of the financial analysis provided by Spendlab Recovery are:

- Identify and recover undue payments. For example, if the organization pays the same invoice twice, then the second payment is undue and can be reclaimed.
- Identify and reclaim VAT errors. For example, the organization pays 21% VAT on received goods while this had to be 9%. Or the organization charges 9% VAT on supplied goods, while, this had to be 21%.

[1] https://www.spendlab.com/home

- Identify unjustified invoices. For instance, the administration of the organization contains an invoice for goods; however, these goods were never received or ordered.
- Identify missed claims. For example, if the organization delivered goods, but forgot to send an invoice. Therefore, losing money.

The financial analysis consists of six main steps; these are:

1. Preparation- Preparation aims at preparing the data for the following steps. It consists of exploratory research. During this exploratory research, Spendlab Recovery looks at the financial department and the implemented processes within the organization. One of the goals of this exploratory research is to determine risk areas to improve the audit.
2. Data- In this step, Spendlab Recovery collects the necessary data, that is used to perform the financial audit. This data consists of:
   - Master data of vendors
   - General ledger
   - Bank details of the vendors
   - Contact information of the vendors
   - Invoices
   - Invoice lines

   We give a more detailed description of the collected data in chapter 5. The collected data is combined into a single unified view. From the unified view, the analysis file is created. The resulting analysis file is then used in the risk analysis and risk verification steps. Challenges during the data step arise from the fact that each organization uses (multiple) different financial systems; for example, DSM might use SAP and EXACT implementations, whereas, ASML might use Microsoft dynamics. The data residing in these different systems have to be integrated correctly to preserve data quality. Otherwise, the financial audit will become less reliable and profitable.
3. Risk analysis- During the risk analysis, the invoices existing in the data of the organization are indicated with categories. The APRA platform generates the risk categories (section 1.2).
4. Risk verification, in the verification step, the given risk categories are judged. The verification step can result in claims, for instance, an undue payment. These claims are used in the collection step.
5. Collection- In the collection step, the claims from the risk verification are collected. This is done in consultation with the organization.
6. Closure- In the closure step, a final report is made. This report contains:
   - The claims and the money that was reclaimed.
   - A detailed description of the analysis, which includes the found risks.
   - Mutations that the organization can perform to balance the financial statements/accounts.
   - The root-cause-analysis of the found risks, and if possible, advice to prevent these risks in the future.

1.2 Account Payable Recovery Analysis

Accounts payable recovery analysis (APRA), is the platform, which is currently developed and maintained by Spendlab Technology. Spendlab Technology is a separate business unit, responsible for the maintenance and development of APRA, as well as the technical issues and challenges encountered by Spendlab Recovery during the audits. APRA aims to provide a platform, by which the audits performed by Spendlab Recovery become:

- Faster- By indicating risks, the audit can be performed in a more structured manner.
- More reliable- By making use of algorithms, the audit becomes less prone to human errors.
- More profitable- By making use of algorithms risks that otherwise would not be found or would have taken a long time to find, are indicated.

1.3 Research problem

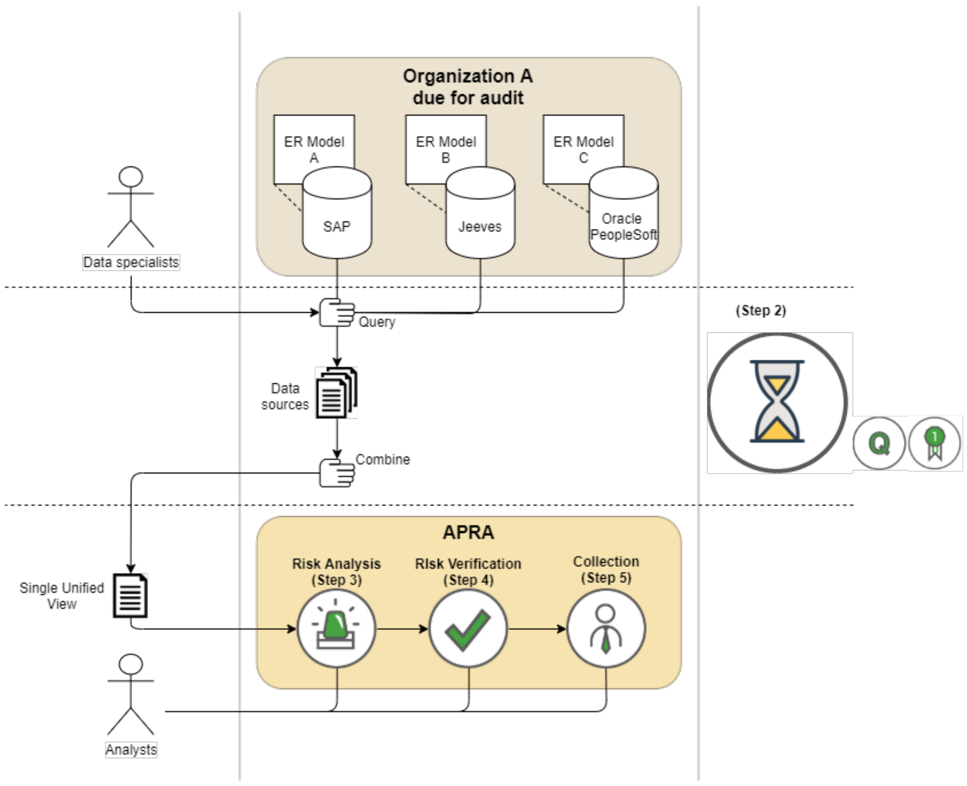


Figure 1.1: The current situation involving the APRA platform and the financial systems of the customer

Figure 1.1 illustrates the current situation; as can be seen, organization A uses three financial systems, namely: SAP, Jeeves, and Oracle PeopleSoft. These different financial systems capture the same type of data [2]. However, these systems differ in underlying techniques and schema, which

[2]We define a financial system as a set of one or various related data sources that originate from enterprise resource planning (ERP) systems or Best Billing/Accounting systems. ERP systems seek to integrate the complete range of business processes. Whereas, Best Billing/Accounting systems solely are responsible for the collection, storing and processing of financial and accounting data. Basically, Best Billing/Accounting systems are a sub part of ERP systems

are used to capture the data. The data specialist has to retrieve data from the financial systems separately. After which the data specialist has to combine the retrieved data into a unified view. Key challenges during this process are:

- Identify the tables and columns that are necessary to create a unified view.
- Identify how the necessary tables and columns of the financial systems relate to each other.
- Retrieve and combine the data from the different financial systems into the unified view.

Currently, integrating the retrieved data is executed with generic tooling (MS ACCESS [3]). As a consequence, the integration process is challenging, tedious, error-prone, and lengthy. Further being complicated by the fact that large organizations are often composed of several business units. Therefore, (large) organizations mostly use multiple financial systems. Examples of these financial systems are:

- SAP
- Exact Globe
- Navision
- Oracle ERP cloud
- PeopleSoft (Oracle)
- Microsoft dynamics
- Jeeves ERP

Because the goal of financial systems is nearly identical, namely keeping track of the accounting and business processes of a company. The question arose whether it is possible to develop a data integration system that eases up or even fully automates the integration process implemented by Spendlab Recovery. Figure 1.2 shows, on a high level, the situation this work aims for, as can be seen, the different financial systems are integrated through the proposed data integration system. Making the integration process less challenging, tedious, error-prone, and lengthy.

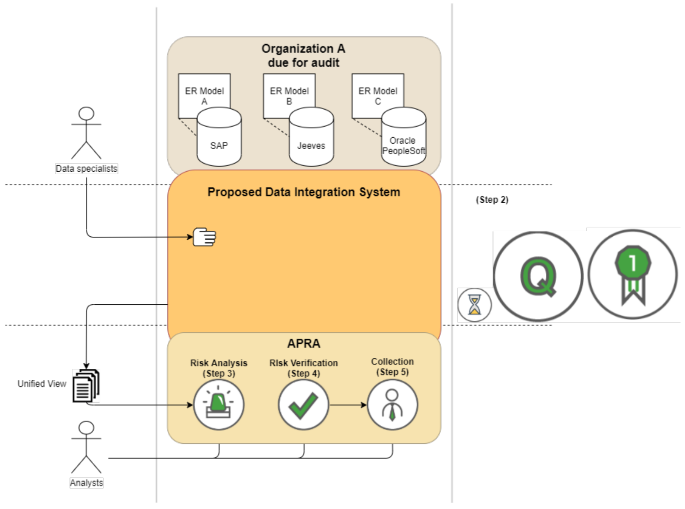


Figure 1.2: The desired situation involving the APRA platform and the financial systems of the customer

[3]https://products.office.com/en-us/access

1.4 Project scope and goals

The main objective of this project is first and foremost to explore the possibility if the integration process can be made easier, ideally fully automated through realizing and creating a prototype. This prototype aims to integrate two frequently encountered financial systems by Spendlab Recovery, which are SAP, and EXACT Globe.

The main contributions are:

1. A study of the current situation, the wished situation, and how to best realize this.
2. Design and implement the prototype that integrates, as well as centralizes the data from financial systems.
3. Examine the capabilities of the prototype through an extensive experimental evaluation.

1.5 Research method

Research question	Goal	Method
What are the prerequisites and requirements of the system?	Identify the wishes of the stakeholders, and translate them into requirements	Interview
What techniques are suitable to create the integration system adhering to the requirements of Spendlab Recovery?	Identify techniques/components (Method fragments) that best suit the proposed integration system	Literature review
To what extent does the created integration system ease up, standardize, or fully automate the integration process of financial systems?	Asses the effectiveness of the proposed system, and identify shortcomings that can be future work	User experiments in which users of the system will perform integrations of financial systems. The integrations are done with the proposed integration system.

Table 1.1: Research questions and methods of this project

Given the problem context, the main research question of this study becomes: *Can we design an integration system that eases up, standardize, or fully automate the integration process of financial systems implemented by Spendlab Recovery*? To answer the main research question, we defined the following three research questions, their goals and the method used to achieve the goal. These are shown in table 1.1

1. *What are the prerequisites and requirements of the system*?

   As table 1.1 shows, we try to answer this research question by conducting interviews with the stakeholders. The interview is set up to answer the following sub-questions:

   - What is the current situation?
   - Given the current situation, what are the encountered problems?
   - How can the integration system resolve these problems?

   The interviews are listed in Appendix B.5.

2. *What techniques are suitable to create the integration system adhering to the requirements of Spendlab Recovery?*

   As table 1.1 shows, we try to answer this research question by reviewing existing literature on integration systems, and we categorize data integration systems and techniques. Identify different components and set-ups. After this, we select the components that best satisfy the requirements of the proposed integration system.

3. *To what extent does the created integration system ease up, standardize, or fully automate the integration process of financial systems?*

   As table 1.1 shows, we use user experiments to measure the effectiveness of the proposed integration system. In order to measure the effectiveness, we implemented a fully functioning prototype based on the results of the previous question. Multiple end-users will then use this prototype to perform the integration process. After which we can measure the effectiveness.

Furthermore, the methods used for the design of the integrated system are prototyping and method engineering [7]. In this context, the method is the proposed integration system. The integration system consists of different components. The various components are the method fragments. These method fragments (components) can be combined uniquely or be newly created to develop a new method (proposed integration system). Prototyping is used to develop the proposed integration system.

1.6 Outline

This remainder of this thesis is organized as follows. Firstly chapter 2 describes, based on the results from the interview, the prerequisites and requirements of the proposed integration system. Next, chapter 3 will discuss related work to the field of data integration and schema matching, identifying techniques and components. Followed by chapter 4 describing the overall system architecture. The system architecture shows how the different components are combined into the integration system. Subsequently, the various components of the proposed integration system are described.

Chapter 5 introduces the centralized model used to store the data

Chapter 6 describes the used schema matching algorithms

Chapter 7 lists the algorithm, which given two schema **A** and **B** generates a query which moves the data from **A** to **B**

Chapter 8 describes the graphical user interface

Chapter 9 gives a thorough experimental evaluation of the performance of the proposed integration system. Finally, concluding the work and listing future work in chapter 10.

Chapter 2

Prerequisites

In the previous chapter, we introduced Spendlab Recovery, the research problem, and the research method. This chapter describes, based on the interview results, the prerequisites and requirements of the proposed integration system. Firstly, we introduce the stakeholders in section 2.1. Secondly, we describe the setting in section 2.2. Followed by the interview results in section 2.3. Finally, section 2.4 concludes the results and lists the requirements of the proposed system.

2.1 Stakeholders

An individual, group, or organization who is impacted by the outcome of the project or have rights, claims, shares, and interests in the project are called stakeholders. Stakeholders influence the project. Therefore, all stakeholders related to the project will be listed here.

TUe

Name	Role
Niek van Son	Graduate Student
Ekaterini Ioannou	Graduation Supervisor

Table 2.1: Stakeholders of the TU/e

In order to graduate at the TU/e, standards have to be met. The TU/e stakeholders concern themselves with the research and implementation of the system to meet the requirements of graduation.

Data specialist

Name	Role
David Michalik	Data specialist
Hugo Bartels	Data specialist

Table 2.2: Stakeholders with the role data specialist

Data specialists are the primary users of the proposed integration system.

Spendlab Recovery

Name	Role
Frits van de Water	CTO
Iris Lopez	CEO

Table 2.3: Stakeholders of Spendlab Recovery

The stakeholders of Spendlab Recovery are responsible for the strategic vision of the APRA platform and how the proposed integration system supports this.

2.2 Setting

Spendlab Recovery operates in the financial domain. Therefore, Spendlab Recovery mostly works with ERP, Best Billing, and Invoicing systems. The main difference between ERP systems and Best Billing/Invoicing systems is that Best Billing and Invoicing systems are strictly aimed at the financial aspect of the business. The financial aspect of the business includes account receivable, account payable, and payroll. ERP systems include all these accounting software features expanded by, for example:

- Human resource management.
- Material management.

We illustrate the challenges of financial data concerning Spendlab Recovery using the 5V's of big data. The 5V's of big data stands for volume, velocity, variety, veracity, and value [24].

- **Volume**- Volume stands for the large amounts of data that are generated. Each company has to handle its accounting; The larger the company, the more invoices that need to be handled. Still, even small companies might handle lots of invoices.
- **Velocity**- Velocity refers to the speed at which new data is generated. The rate at which invoices are sent/received is quite high. Large(r) companies consisting of different entities and business units generate invoice data even faster.
- **Variety**- Variety stands for the difference in the structure of the received data. Each company uses it's own ERP, Best Billing, Accounting system. These systems differ on an implementation level. Moreover, within a company, a multitude of these systems can be used.
- **Veracity**- Veracity indicates the quality of the data. Because handling invoices most of the time is done by humans, the data will not be entirely correct. Past projects have shown that null values are common, and also duplicate vendors/invoices. Furthermore, the quality of the data varies significantly between the companies that are being audited.
- **Value**- Value describes the worth of the data being extracted.

2.3 Interview results

2.3.1 Data specialists

Spendlab Recovery performs around 150 analyses each year. The current process that the data specialists execute consists of three steps. We identify the three steps as follows:

1. Retrieve data (financial systems) from the customer.

2. Integrate the different financial systems. The integration will result in a unified view named the “51”.

3. From the unified view, create the analysis file. The analysis file is named 84.

According to the data specialists, the most challenging parts of the process are steps 2 and 3. Therefore, we aim to improve the process once the data is retrieved. Figure 2.1 shows steps 2 and 3 in more detail. In step 2 the data from the different financial systems are integrated and combined in a unified view. This unified view is named the *51*. Step 3 takes this unified view and creates the *84*, which is used to create the analysis file. Note that both the *51* and *84* are single tables. We will describe both steps in more detail.

Step 2 starts with mapping data from the financial system to four different sub tables, namely:

- **Master-** The master table contains all the information regarding the creditors, including addresses, bank accounts, etc.
- **VAT-** The VAT table contains data related to VAT, such as the VAT amount.
- **Invoice-** The invoice table contains the invoice related data, such as the amount to be paid, invoice number, and invoice description. Although an invoice consists of multiple invoice lines, the current integration process discards these lines.
- **Payment-** The payment table contains data related to how and when the payment has been made.

Deciding how the data, existing in the financial system, maps to one of the four tables is based on previous experience. However, if a new ERP, Accounting, or Billing system is encountered, then the column names and the data stored within these columns determine the mapping. After the data from the financial system is stored in these subtables. The subtables are combined into a single table, which is called the *50*. Combining the sub tables means that the data from each sub-table is joined. The *50* table still is system-specific. That is, columns/tables (and their names) differ between the *50* tables created from the SAP system and the Exact globe system. The final part of step 2 is combining the resulting *50* tables into the *51* table. Because the *50* table is system-specific, another mapping has to be made. This mapping determines how the columns

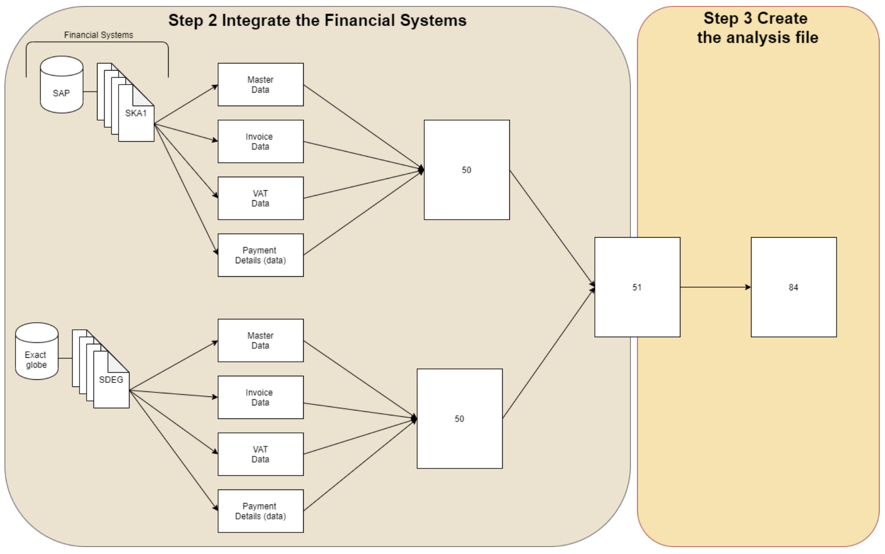


Figure 2.1: Steps 2 and 3 taken during the integration process of two financial systems

from the *50* table(s) relate to the *51* table. The *51* table adheres to a standard, and thus providing a unified view concluding step 2.

In step 3, the data specialist created the analysis file. This step consists of executing queries. These queries take the data from the *51* table and augment it with, for example, VAT percentages. After this, the augmented data is stored in the *84*. The resulting analysis file (chapter 1) consists of the data from the *84* table.

Step 2 is challenging because it is not standardized, meaning that each data specialist executes step 2 differently. Therefore, relating columns between the tables in a different manner. Furthermore, each integration always starts anew, and past integrations are not considered. As a result, data specialists not only approach the financial systems differently but also the same data specialist can approach the same financial system (from another organization) differently. For these reasons, multiple errors can occur, namely:

- The data can be interpreted differently. Data specialists can map the same column in a data source to different columns in the target database (ACCESS).
- Decision making can differ between data specialists during the process.
- The data specialists can use different joins.
- Data specialists can execute wrong joins. Therefore, not being able to match records or introduce incorrect matches. Incorrect matches will result in corrupt data.
- Key fields in the data sources might be determined differently.
- Not paying attention to, and thus not noticing errors.

Besides these human errors, other errors can occur, such as:

- Incomplete/unreliable data present in the data sources.
- Data conversion errors. The data specialists sometimes export the raw data from the system of the customer to CSV/TXT. Therefore, for example, if an invoice number exists with the value of 1000000000, then this might be parsed to $1E + 9$.
- Different data formats used to describe the same data across the financial systems.

Likewise, in step 3, the creation of the *84* table from the *51* table is not standardized.
Besides these errors, the moment at which an error is noticed varies greatly. This is a consequence of the fact that the integration process is not standardized. As a result, errors can be noticed anywhere in the process, or (worst of all) not at all. In any case, errors result in money loss because either the risk analysis will be performed on incorrect data or the data integration has to be redone.

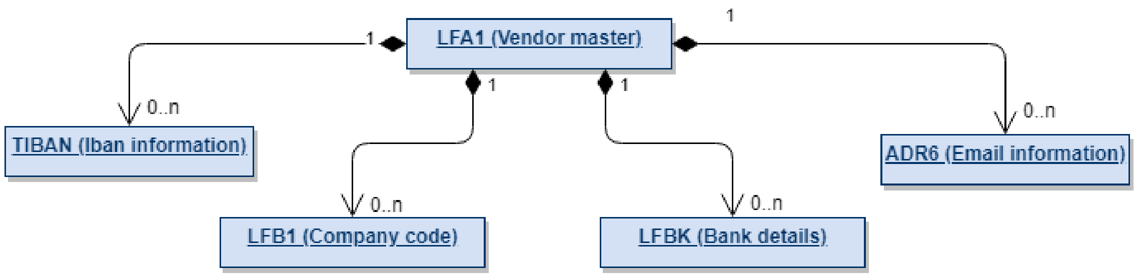


Figure 2.2: Structure of Master data in SAP

Another problem arises from the fact that data from the financial systems are mapped to four sub tables. Because these sub tables are used, existing relationships in the data are discarded. For example, a single vendor can have multiple account numbers stored in 1:N relationships. Figure 2.2 shows how these relationships are stored in SAP systems. If this is mapped to the sub-table master, then the data specialist picks a single entry. Therefore, the data is denormalized to the first normal form. Likewise, this holds for all relations present in the data. Consequently, data integrity is not maintained. For example, if the amount of account numbers for a vendor increases, then the chance of picking the correct one becomes smaller.

As said step 3, the creation of the .84 is not standardized. Each data specialist can execute the required queries differently. Introducing discrepancy in the resulting 84 table(s). Additionally, the following errors can occur:

- The data specialist forgets a query.
- The data specialist executes the same query multiple times, resulting in duplicate data.
- The data specialist executes self-made queries containing errors.

Another drawback is that MS ACCESS is used, which can not handle files larger than 2GB for the integration process. The reason that the data specialists use MS ACCESS is that they do not have a profound knowledge of SQL or other query languages. The main aspects of the proposed integration system, according to the data specialists, are:

- The system supports integrating different data source(s)
- The system shows potential mappings, making the integration process less complicated, especially for newly encountered systems.
- The relationships present in the data should remain intact
- The tool introduces standards; for example, correct mappings can be reused for the same financial system
- The data specialist should get a clear view of the data in the financial system to notice errors more easily

2.3.2 Spendlab Recovery Stakeholders

Spendlab Recovery does 150 analyses each year; the aim is to increase this to 200.

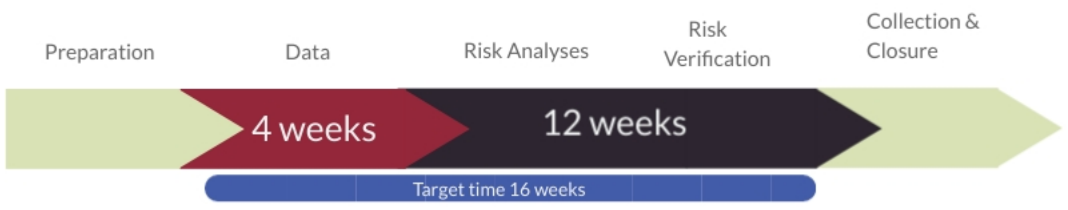


Figure 2.3: The timeline, with target times, of a single financial audit

Figure 2.3 shows the six steps of the financial audit (chapter 1), and the target time for the Data, Risk analyses, and Risk Verification step. We do not show the times for preparation and collection & closure because these times vary considerably. If we address financial audits, then we mean the data, risk analyses, and risk verification step. After these three steps, the audit is completed, and the resulted claims are collected, which can take some time. There are two reasons the target time of 16 weeks is not always achieved.

These two reasons are:

- **Internal**: Internal reasons are errors within the processes of Spendlab Recovery. For example, errors during data integration that are not noticed, resulting in time loss because it has to start over.
- **External:** External reasons are errors outside of the company processes. For example, failures at the organization or suppliers of the organization, that can result in communication errors or having to wait for a reply.

Figure 2.3 indicates the vital role the integration process takes, and the impact it has on the target time. That is, the integration process has to be (correctly) completed before the other services can start. Therefore, if the target time (4 weeks) of the data step is not achieved, then the delay will propagate through the rest of the audit.

According to the Spendlab Recovery stakeholders, there is a discrepancy between the data specialists and the other functions. Meaning that, across the company, most of the time, only the general idea of the integration process is known. However, the details of the integration process are not. Therefore, only data specialists can judge the quality of the data integration. Consequently, audits can be completed while containing integration errors. By creating a data integration system, internal errors in the data step should become less frequent. As a result, the resulting analysis file from the data step is of higher quality, as well as the time to create the analysis file is reduced. Furthermore, by not discarding the 1:N relations present in the financial system of the organization, not only the analysis might improve, but also Spendlab Recovery can introduce a new service called clean master data. This service cleans the data, for example, removing not used account numbers and adding correct ones.

2.4 Conclusion

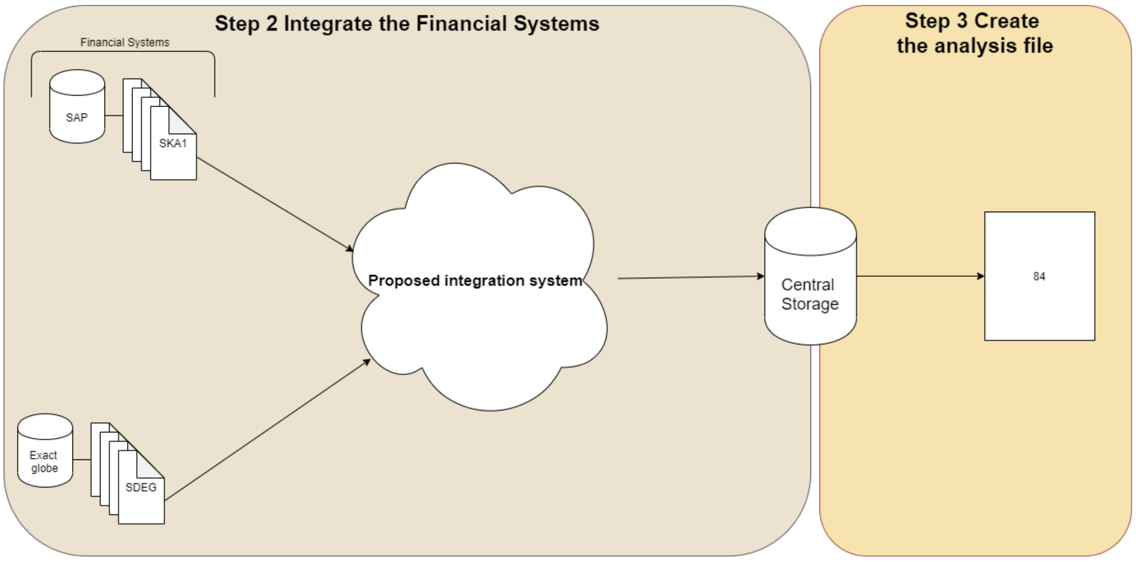

Step 2 Integrate the Financial Systems
Financial Systems
SAP
SKA1
Exact globe
SDEG
Proposed integration system
Central Storage
Step 3 Create the analysis file
84

Figure 2.4: The proposed integration system, and how it fits the current steps taken by the data specialist

This chapter introduced the stakeholders, described the setting, and summarized the interview results. Based on the interviews, we identified three main issues. These main issues are:

- The integration process is not standardized.
- Relations in the data, existing in the financial systems, are discarded.
- There is a lack of quality control regarding the integrated data throughout the audit.

From this, we conclude that the proposed integration system should standardize the integration process. Standardization means that once a financial system is integrated, data specialists can reuse the executed steps. Reusing steps decreases the chance of errors, but also saves time. Secondly, instead of using single tables, central storage (data warehouse) should be used, to which all the financial systems can be mapped. The model used for the central storage should keep the relations intact. By querying or creating metrics on the central storage, the system can provide insight into the quality of the resulting integration. Furthermore, the central storage can be queried to create the “84” table. The “84” table then can be used to generate the analysis file (figure 2.4). Lastly, because the data specialists do not have a profound knowledge of SQL or other query languages, the system should be useable through a graphical user interface. Therefore, not requiring knowledge of query languages. In short, a system which provides the following functionality:

- Supports integrating different data sources.
- Makes integrating new data sources easier through an automated mapper.
- Automates integrating existing data sources through saved mappings.
- Overview of the data after the integration to notice errors more easily.
- Remains close to the raw data of the customer, meaning that the relations in the data remain intact.

We translate the functionality into the following features (Table 2.4), system requirements (Table 2.5) and user requirements (Table 2.6)

Index	Title	Description
F01	Mediated schema	DISAP provides datawarehouse in which the financial data from the different data sources can be stored
F02	Clean data	DISAP provides functionality to transform the data from the data sources
F03	Integrate data	DISAP provides functionality which integrates data from the data sources into the data warehouse
F04	Access data warehouse	DISAP provides API through which (integrated) data from the data warehouse can be accessed

Table 2.4: Main features of DISAP

Index	Title	Description	Owner	Priority
S01	Connect to data sources	As system I want to provide a interface to connect to different data sources	Student	Must

Table 2.5: System requirements

System requirements

Index	Title	Description	Owner	Priority
S02	Extract data	As system I want to be able to extract data from the different data sources	Student	Must
S03	Save mapping	As system I want to be able to save the mapping from the data sources to the data warehouse	Student	Must
S04	Retrieve saved mappings	As system I want to be able to retrieve saved mappings	Student	Must
S05	View mappings	As system I want to be able to show an overview of the mapping	Student	Must
S06	Generate schema mapping	As system given a source schema **S** and the mediated schema **T** I want to generate a mapping	Student	Should
S07	Extract schema	As system given a data source **D** I want to be able to extract a schema **S**	Student	Should

Index	Title	Description	Owner	Priority
U01	Add Project	As data specialist I want to be able to add a project in order to start the integration process	Student	Must
U02	Add system	As data specialist I want to be able to add a system to the project	Student	Must
U03	Open data system	As a data specialist I want to be able to open the data system in order to integrate	Student	Must
U04	Add sources to data sytem	As a data specialist I want to be able to add data sources to the data system	Student	Must
U05	Overview Mapping	As data specialist I want to have an overview of the suggested mapping from schema **S** to mediated schema **T**	Student	Must

Table 2.6: User requirements

User requirements (continued)

Index	Title	Description	Owner	Priority
U06	Augment generated mapping	As data specialist I want to be able to augment the generated mapping from schema **S** to mediated schema **T** in order to be able to make changes to the mapping	Student	Must
U07	Finalize mapping	As data specialist I want to be able to finalize the mapping from schema **S** to mediated schema **T**	Student	Must
U08	Integrate data	As data specialist I want to be able to integrate the data from the data sources to the data warehouse (mediated schema)	Student	Must
U09	Query data	As data specialist I want to be able to access the data warehouse through API	Student	Must
U010	Generate .84 file	As data specialist I want to be able to generate the .84 file from the data warehouse using the API	Student	Must
U011	Dashboard	As data specialist I want to analyze the resulting data from the integration process using a dashboard	Student	Should

Chapter 3

Related work

In the previous chapter, we described the prerequisites and requirements of the proposed integration system. In this chapter, we review existing literature on data integration systems. The goal of this chapter is to identify techniques and components that best fulfill the requirements of the proposed integration system and how these techniques and components can be combined. The proposed integration system touches related work in the following three areas:

- Schema mapping, which is responsible for generating mappings from the data source(s) to the data warehouse.
- Data exchange, this area is responsible for correctly moving data from the data source(s) to the data warehouse.
- GUI, which is responsible for providing a user-interface enabling the users to augment generated mappings.

We focus on two key components: Schema mapping and Data exchange, and uniquely combining these, greatly simplifying data integration for financial data. The third key component GUI will be described in chapter 8. First, in section 3.1, we summarize the motivation for data integration systems. In section 3.2, we will review the literature on data integration systems as a whole, followed by schema mapping and data exchange individually.

3.1 Motivation

IBM defines data integration as "the combination of technical and business processes used to combine data from disparate sources into meaningful and valuable information." [1] Because enterprises today generate & manage huge amounts of data in their daily operations, mostly using separated systems, the problem of designing data integration systems has become ever more important. Furthermore, enterprises are becoming progressively more data-oriented. That is, data science techniques are increasingly applied, in real-world contexts, to support amongst others decision making. Therefore, preserving high data quality is evermore important [39]. One of the central issues to preserve high data quality is a proper data integration system [41]. As described enterprises today mostly use separate systems, which use different formats: database formats (e.g., relation model), semi-structured formats (e.g., DTDs, SGML or XML schema), scientific formats etc. [36]. The effort involved to integrate these systems is considerable. Moreover, in practice, this is not always done by computer scientists, but rather domain experts. Therefore, designing and maintaining a proper data integration system is important for every data-oriented business.

[1]https://www.ibm.com/analytics/data-integration

3.2 Literature review

3.2.1 Data integration systems

Data integration systems aim to combine data residing at different sources and providing users with a unified view [27]. There exist two different approaches to querying a set of (heterogeneous) data sources, namely:

- Schema first, the schema first approach queries a (mediator) schema, to which multiple data sources are mapped to provide a semantically integrated view.
- No schema, the no schema approach considers each data source separately and considers all data from these data sources right from the start. It provides basic keyword and structure queries.

Mediator architectures and data warehouses use the schema first method. The drawback of this approach is that the semantically integrated views only can be provided if there are precise mappings between the source(s) schema and the target schema. The no schema approach does not require such a mapping to create a semantically integrated view over the data source(s), it uses the bag of words model or XML data model to query the data source(s) [42]. The no schema approach is mostly implemented by search engines such as Google and Beagle. However, the no schema approach does not provide any form of information integration [42].

The data integration systems we are interested in this thesis are the ones categorized by the schema approach. In other words, by an architecture based on a global schema. Most data integration systems either express the global schema as a view on the sources called Global as a View (GAV) or the source schema as views on the global schema called Local as a View (LAV) [27] [42]. Global as a view, the GAV approach is based on the idea that the contact of each element g of the global schema should be characterized in terms of a view q_s over the sources [27]. The Local as a View approach is based on the idea that the contact of each source s should be characterized in terms of a view q_g over the global schema [27]. A mix of these two approaches is also possible, which is called GLAV [15] [27]. Other methods that we found in the literature are Peer-to-peer integration systems [19] [6]. In Peer-to-peer integration systems, every peer acts as both client and server and provides part of the overall information available from a distributed environment without relying on a single global view [28]. Nevertheless, peers still have to be semantically mapped to one another.

3.2.2 Schema mapping

Schema matching is the process of identifying elements in two or more databases for integration and is a fundamental problem since the 1980s [14] [40]. During schema matching the semantic correspondence between two or more schema is determined [11]. Schemas are defined as a formal structure that represents an engineered artifact [5]. Examples of these artifacts are:

- Database schema
- XML schema
- Message formats
- Ontology's

In the case of Spendlab Recovery, engineered artifacts are database schema. Schema matching done by hand is a tedious, time consuming, and error-prone process. Being further complicated by the increasing amount of data [37]. To reduce the manual effort associated with schema matching, fortunately, lots of work has been done on (semi) automatically schema matching. Automatically schema matching is only possible if best-effort matching is satisfactory [5]. Mostly, as is the case with Spendlab Recovery; an analyst needs to examine the resulting schema match to prevent integration errors. Challenges of schema matching are:

- Design conflicts, schema are developed independently, resulting in different structures and terminology even if they model the same domain. [37]
- Non-monolithic schema vs. distributed schema.
- Naming conflicts between schema modeling the same domain. [40] [11]

The main function of schema matching is match. The match function takes as input two schema and outputs a mapping [37]. Let a mapping be defined as a set of mapping expressions, which relates one or more elements from source schema **S1** to one or more elements existing in target schema **S2**. These mapping expressions can be directional and non-directional. A non-directional relation is a relation between a combination of elements from S1 and S2 [37]. Furthermore, the mapping expressions contain a similarity score.

This score $[0, 1]$ indicates how similar the elements in the mapping expression are:

$$Similarityscore = \begin{cases} 1 & \text{If the elements are identical.} \\ 0 & \text{If the elements are completely indifferent.} \end{cases} \tag{3.1}$$

The closer the score comes to 1, the more related the elements are.

The schema matching process can involve a wide variety of matching algorithms. Therefore, the implementation of the match function can use one or more matching algorithms [37]. These matching algorithms determine correspondences based on a given criterion, for instance: name similarity between a table name from source schema **S1** and target schema **S2**. Combinations of matching criterion are possible by either combining the results of multiple individual matching algorithms or implementing various matching criterion within a single matching algorithm [37] [11]. We define the method of combing the results of multiple individual matching algorithms as a combined matcher and implementing various matching criterion, within a single matching algorithm, as a hybrid matcher. Each of these methods has its advantages and disadvantages which we will list below:

- Combined matcher
  - Advantages
    - Extensibility, a combined matcher, can easily add or remove different individual matchers.
  - Disadvantages
    - Inefficient, the combined matcher uses multiple passes over the schema and combines the results of each pass. The hybrid matcher only uses a single pass.
- Hybrid matcher
  - Advantages
    - Efficiency, a hybrid matcher only needs one pass over the schema to determine the mapping, whereas the combined matcher has multiple passes.
    - Provide better match candidates. Because poor match candidates matching only one of several criteria are filtered out. [37]
  - Disadvantages
    - Inextensible, using a hybrid matcher makes the setup less extensible. Because the various criterion all are programmed in the single algorithm. If we want to add a new criterion or remove some existing criterion, then this has to be done pragmatically, whereas, with the combined matcher, it is possible to just remove the individual matcher.

The wide variety of algorithms involved in schema matching can be categorized. We use the categorization described in [14] [40] [37] consisting of schema-level matching methods and instance-level matching methods. Schema-level matching only uses available schema information. Schema

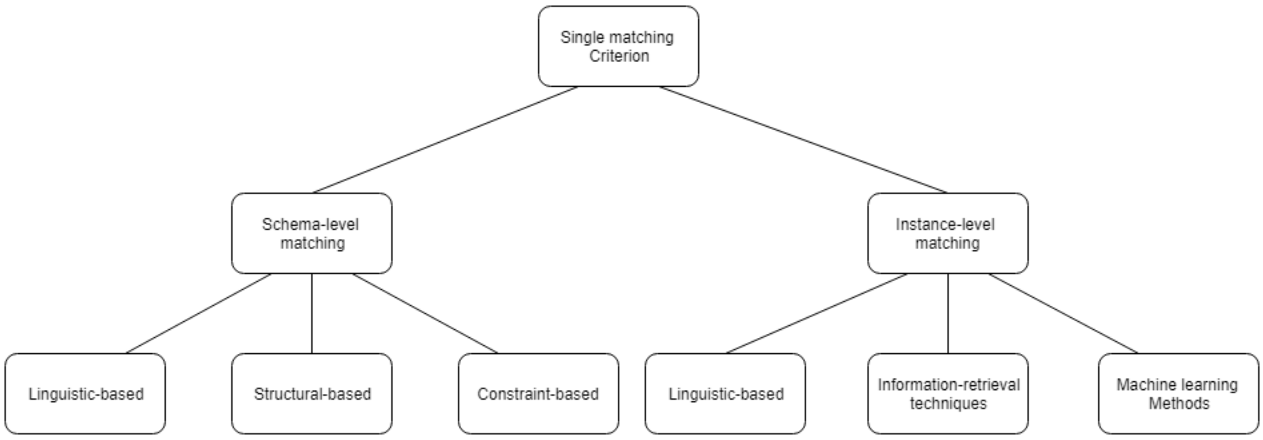


Figure 3.1: The categorization consisting of schema-level and instance-level methods

information consists of table-names, column-names, table-description, foreign-key indicators, etc. It only looks at the meta-data of the schema but not at the data that is stored. Schema-level matching methods are linguistic-based, structural based, or constraint-based.

- Linguistic-based methods look at the table/column names and use string similarity methods to determine correspondences [14].
- Structural-based algorithms use relationships between tables, foreign-key dependencies, and, for instance: use a graph comparison algorithm to determine correspondences [14].
- Constraint-based methods look at the data-constraints, optionality constraints, uniqueness constraints of the columns [14].

Whereas, schema-level matching looks at the schema-information instance-level matching look at the data stored inside the source. Instance-level matching algorithms are linguistic-based, use information-retrieval techniques or machine-learning techniques. Linguistic-based methods can look at key-terms; Information-retrieval techniques use distributions, frequencies, averages, patterns such as phone numbers, or IBAN-numbers. Machine-learning methods apply, for instance: neural networks to determine correspondences.[3] [31] [12] [30] [8] Note, that by solely operating on the financial domain, schema matching algorithms can be specially adapted to work well within the financial domain.

3.2.3 Data exchange

We follow [15][16] and define the data exchange as Data structured under one schema (which we call a source schema) must be restructured and translated into an instance of a different schema (a target schema). More formally, this translates to the following definition. Given a source schema $\mathbf{S}$, a target schema $\mathbf{T}$, a set of constraints over $\mathbf{t}$ denoted as $\sum_t$, and a set of source-to-target dependencies denoted as $\sum_{st}$, which relate elements in $\mathbf{S}$ to elements in $\mathbf{T}$. The problem of data exchange becomes given an instance I over source schema $\mathbf{S}$, materialize an instance J over the target schema $\mathbf{T}$ such that the target dependencies $\sum_t$ are satisfied by J, and the source-to-target dependencies $\sum_{st}$ are satisfied by I and J together.

According to [16] in data exchange, the target schema $\mathbf{T}$ is often independently created and comes with its own constraints, whereas, in data integration, the target schema is commonly assumed to be a reconciled, virtual view of a heterogeneous collection of sources, and as such has no constraints. Furthermore, in data exchange, the target instance is actually materialized, in a data integration system; this is not required. Both systems apply queries over the target system. However, in a data integration system, the data sources are used to answer queries over the target schema. In contrast, in a data exchange setting queries, are answered over the materialized target instance, and therefore require no query reformulation.

3.3 Conclusion

Most research on schema matching and data exchange has been done separately. Meaning that research on data exchange does not take schema matching algorithms into account, but instead assumes that a set of correspondences is available. Whereas, research on schema matching does not take data exchange into account, but rather only how two schemas relate to each other. This thesis aims to provide a data integration system for financial data by combining schema matching and data exchange in a unique way. We use schema matching algorithms to help the user define semantic mappings between data sources and a global schema. Instead of using these semantic mappings to answer queries over the shared data sources by reformulating queries. We propose an algorithm inspired by CLIO [15] to generate queries, which transport the data from these data sources into the data warehouse. This data warehouse will provide a unified view of the data sources.

Chapter 4

System architecture

In the previous chapter, we reviewed the literature on data integration systems. Based on this literature review and the requirements described in chapter 1, we propose an integration system centered around a data warehouse. This chapter will, in section 4.1, describe the motivation. Section 4.2 shows the architecture. After which we describe the proposed technologies in section 4.3.

4.1 Motivation

In chapter 2, we retrieved the requirements of the proposed integration system. Chapter 3 reviewed literature on integration systems. Combining found techniques based on the requirements of the proposed integration system is important. Therefore a clear system architecture has to be designed, which is extensible, as well as flexible.

4.2 Architecture

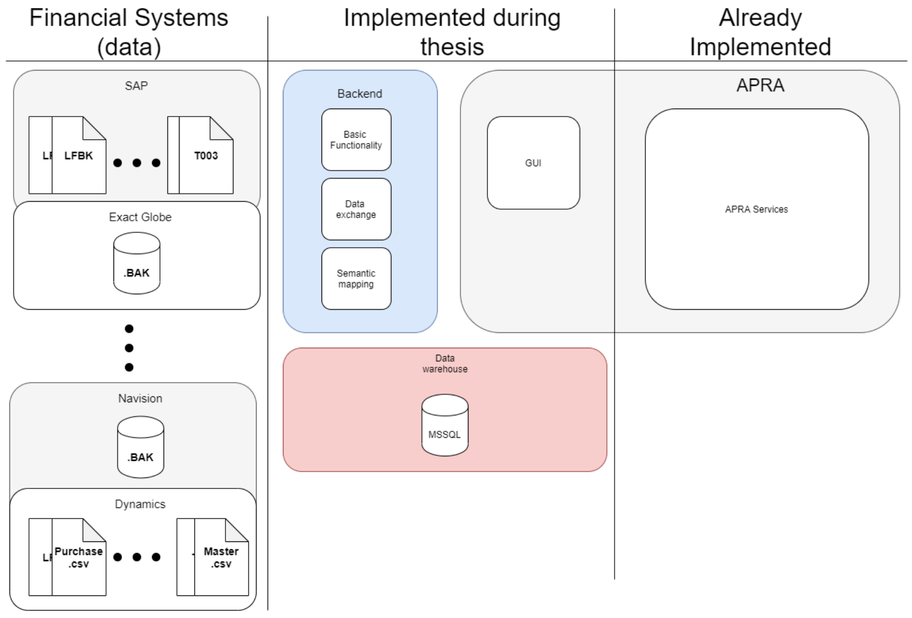


Figure 4.1: The components of the proposed integration system and how these components are compatible with each other

Figure 4.1 illustrates the architecture of the proposed integration system. It shows the components partitioned into three categories.

- Financial systems
- Implemented during the thesis
- Already implemented

First of all, we describe financial systems. Financial systems, as described in chapter 1, are ERP, Best Billing, and Invoicing Systems. Figure 4.1 shows four examples of financial systems that are encountered by Spendlab Recovery. As can be seen, each financial system consists of one or multiple different data sources. Therefore, we define a financial system as a set of one or various related data sources. For example, SAP consists of 9 files; each file corresponds to a table extracted from the SAP database. As a result, the data specialists do not have to retrieve the entire SAP database, but only the tables that are required to conduct the audit. The same holds for the other financial systems encountered by Spendlab Recovery. In appendix A, the financial systems encountered are listed.

Secondly, we describe the already implemented category. This category only consists of APRA services. The APRA services make up the platform by which Spendlab Recovery aims to make the audits faster, more reliable, and more profitable. This component is not described in full detail because it is not in the scope of this project. However, this project aims to create a data integration system that becomes part of the APRA platform (chapter 2). Therefore, as illustrated in figure 4.1, the GUI component of the proposed integration system has to become part of the APRA platform. As a result, it is essential to know what technologies are used to create the APRA platform. The main programming language of the APRA platform is C#; the views are created using HTML5, CSS, and JavaScript. For the database(s) of the APRA platform, MSSQL is used.

Finally, we describe the "*implemented during the thesis*" category. As figure 4.1 shows, the proposed system consists of three main components. These main components are

- Data warehouse, the data warehouse component provides central storage. The underlying central data warehouse model is based on the commonality between encountered financial system. Therefore, it is possible to consolidate the financial systems by mapping them to the central data warehouse. This data warehouse not only can be used to create the analysis file (chapter 1) but also provide a business intelligence platform (in the future). We describe the data warehouse in chapter 5.
- Backend, the backend component provides the functionality of the proposed integration system. The functionality it provides is:
  - Data can be extracted from the financial system, and stored in a temporary database (basic)
  - Consult data in the data warehouse. (basic)
  - Semantic mappings can be generated between the financial system and the data warehouse. (semantic mapping)
  - Augment generated semantic mappings. (semantic mapping)
  - Given the financial system, the data warehouse, and the semantic mapping generate queries that transfer the data from the temporary storage into the data warehouse. (data exchange)

  We classify the functionality that the backend provides in three categories. These categories are basic, semantic mapping, and data exchange. The classified back-end functionality is shown in figure 4.1; this thesis will describe semantic mapping (chapter 6) and data exchange (chapter 7) in more detail.

- GUI, the goal of the graphical user interface component is to make the integration system, and all of its functionality easy to use. The system has to be easy because the data specialist does not have a profound knowledge of SQL or any other query language (chapter 2). Therefore, the users of the integration system are domain scientists rather than computer scientists. Hence, the proposed integration system requires a simple graphical user interface.

4.3 Technology

Selecting suitable technologies is an important step in the design process. If this is not done correctly, then the resulting solution might not be suitable at all. For each of the components listed in section 4.2, we describe the chosen technology.

4.3.1 Data warehouse

One of the main decisions is to select a relational database or a non-relational database. E.F Codd invented the relational database in 1970. A relational database is a collection of data items stored in formally described tables. Through these tables, data can be accessed or reassembled in many different ways. Accessing and reassembling data is mostly done with the Structured Query Language (SQL). Relational databases are well suited for highly structured data. However, relational databases can be highly complex when working with unstructured data. On the other hand, non-relational databases are a class of system which does not use relations as its structure. Non-relational databases can be Key-Value Stores, Document Stores, Graph Database, Column Oriented databases, Object-Oriented Databases, Grid and Cloud Databases, XML Databases, Multidimensional Databases, Multivalue Databases, and Multimodel Databases. Non-Relational databases have an advantage that is they scale easily with large amounts of data. The main disadvantage of non-relational databases is that joins cannot be performed, making them less use full for well-structured data. [23] The data from ERP, Best Billing, and Accounting systems is highly structured; moreover, most of these systems use a relational database to store their data. Therefore, we chose a relational database for the data warehouse. There are many different relational databases, for example, PostgreSQL, MySQL, and MariaDB. However, because the APRA platform uses MSSQL (Microsoft SQL server). We use Microsoft SQL Server to implement the data warehouse. Not only is Spendlab Recovery most acquainted with MSSQL, but because Microsoft Azure [1] is used for managing and monitoring the APRA platform. The data warehouse can easily be deployed to Azure Synapse Analytics [2].

4.3.2 Backend

The APRA platforms backend main programming language is C#. Therefore, the programmers of the APRA platform are most acquainted with the C# programming language. Consequently, we chose C# as the programming language of the Backend.

4.3.3 GUI

The aim is to make the proposed integration system part of the APRA platform (chapter 2). As described in section 4.2, the graphical user interface of the APRA platform is created using HTML5, CSS, and JavaScript. Therefore, we chose also to use HTML5, CSS, and Javascript to implement the graphical user interface. Furthermore, we decided to use D3.js[3] to create, if necessary, more complex visualizations.

[1]https://azure.microsoft.com/en-us/
[2]https://azure.microsoft.com/en-us/services/synapse-analytics/
[3]https://d3js.org/

4.4 Conclusion

The proposed integration system consists of three main components. These main components are:

- Data warehouse
- Backend
    - Basic
    - Semantic mapping
    - Data exchange
- GUI

The data warehouse component provides central storage. Because the underlying model of the data warehouse is based on the commonality between encountered financial systems. The financial systems can be mapped to the data warehouse. The backend provides functionality that can be classified into three categories. These categories are basic, semantic mapping, and data exchange. The semantic mappings functionality generates semantic mappings between the financial system and the data warehouse. These semantic mappings then have to be finalized by the users (data specialists). Given the finalized semantic mappings, the data exchange functionality can generate a query, which transports the data from the temporary storage into the data warehouse. The graphical user interface simplifies using the different components.

Chapter 5

Data warehouse

In chapter 4, we show the architecture of the proposed integration system. In this chapter, we describe the data warehouse. The data warehouse is one of the main components of the proposed integration system. In section 5.1, we describe the motivation of the data warehouse component. After that, section 5.2 describes in detail the data that is going to be stored in the data warehouse. Finally, in section 5.3, the model for the data warehouse is described.

5.1 Motivation

As described in chapter 3, we are interested in the schema-first approach. As a result, we designed an architecture based around a data warehouse. The goal of the data warehouse is to provide central storage (the unified view). In order to make the proposed integration system flourish, the underlying model of the data warehouse must be set up correctly. That is, the underlying model must be based on the commonality between ERP/Financial accounting systems, thereby making it possible to map each ERP/Financial accounting system (section 5.2) to the data warehouse. Moreover, given the requirements of the proposed integration system (chapter 2), the relations present in the ERP/Financial accounting systems must remain intact. In other words, no longer denormalizing the data, which is current practice. Therefore, an '*understandable*' or perspicuous model has to be designed, which is extendable for future purposes.

5.2 Data

Spendlab Recovery provides financial analysis for organizations. To conduct these analyses, Spendlab Recovery operates with the financial data of the organization. Financial data is mostly stored in enterprise resource planning (ERP) systems or Best Billing/Accounting systems, which we call financial systems. ERP systems seek to integrate the complete range of business processes, especially procurement, material management, production, logistics, maintenance, sales distribution, financial accounting, asset management, cash management, controlling, strategic planning, and quality management [25]. Its goal is to organize, define, and standardize these processes to plan and control the organization effectively [21]. Contrary to ERP systems, which focus on the complete range of business processes, accounting information systems are responsible for the collection, storing, and processing of financial and accounting data. Accounting information systems are typically composed of three major subsystems [4]:

- Transaction processing system.
- General ledger system and financial reporting system.
- Management reporting system.

Basically, accounting information systems are a part of enterprise resource planning systems.

The most important concerns for accounting are accounting operations (transaction processing, accounts payable and receivable, internal financial regulation, including internal audits, compliance with regulatory requirements and taxes), management accounting (forecasting, budgeting, costing and reporting on variances (cost control, detailed reports about performance and budget) as well as cash flow management), management support (which includes identifying and analyzing strategic options, decision support, designing and tracking key personnel indicators, benchmarking, strategic management accounting, and business risk management), staff management, training scrutiny of capital projects, emphasis on customers and products, reports about debtor and creditor aging, real time reporting, interactive reporting, auditing, internal controls implementation, risk management, error or fraud detection, accountability [4].

The accounting part of ERP systems and Accounting systems more or less capture the same concepts and data. Because these systems share the same concepts, it is possible to create a mediator schema. We use the mediator schema as a model for the data warehouse. The data retrieved from these systems are composed of:

- Master data of the vendors- e.g., the vendor name, vendor number, VAT number, the address of the vendor, the country of the vendor.
- General ledger- Basically the general ledger, centralizes the accounting data. Data of the general ledger consist of a date, description, balance or total amount for each account.
- Bank details of the vendors- This contains the bank names, account numbers, and IBAN numbers of the vendor.
- Contact information of the creditors- This contains email-addresses and phone numbers.
- Invoices- An invoice is a commercial document issued by a seller or buyer, relating to a sale transaction. The typical invoice contains a unique reference, date, credit term, tax amount, details of the seller, purchase order number, description, the amount charged.
- Invoice lines- The invoice lines describe a single product/item. These lines make up an invoice, i.e., an invoice consists of one or more invoice lines. Typically an invoice consists of unique reference, date, tax amount, description, amount.

5.3 Database diagram

Figure 5.1 shows the mediator schema. The mediator schema is composed of two sections. These sections are:

- Vendor Section- The vendor section consists of data related to the master data, bank details, and contact information of the vendor (section 5.2). Most of the relations are trivial and are one-to-many relations. There is a one-to-many relation between Vendor and Address-Information because a Vendor can have multiple locations. The locations all have different contact information explaining their one-to-many relationship. The same holds for Vendor and AccountInformation; one vendor can have multiple bank accounts. The noteworthy part is the relation between the vendor and the unit entity. Because an organization might consist of various units, in other words, the organization is conglomerate. Meaning that the organization consists of different independent business units. Each of the business units handles its financial processes separately. However, it has to come together in the same financial systems. Because of this and the fact that business units can contract the same vendor, the relation between vendor and unit is many-to-many.

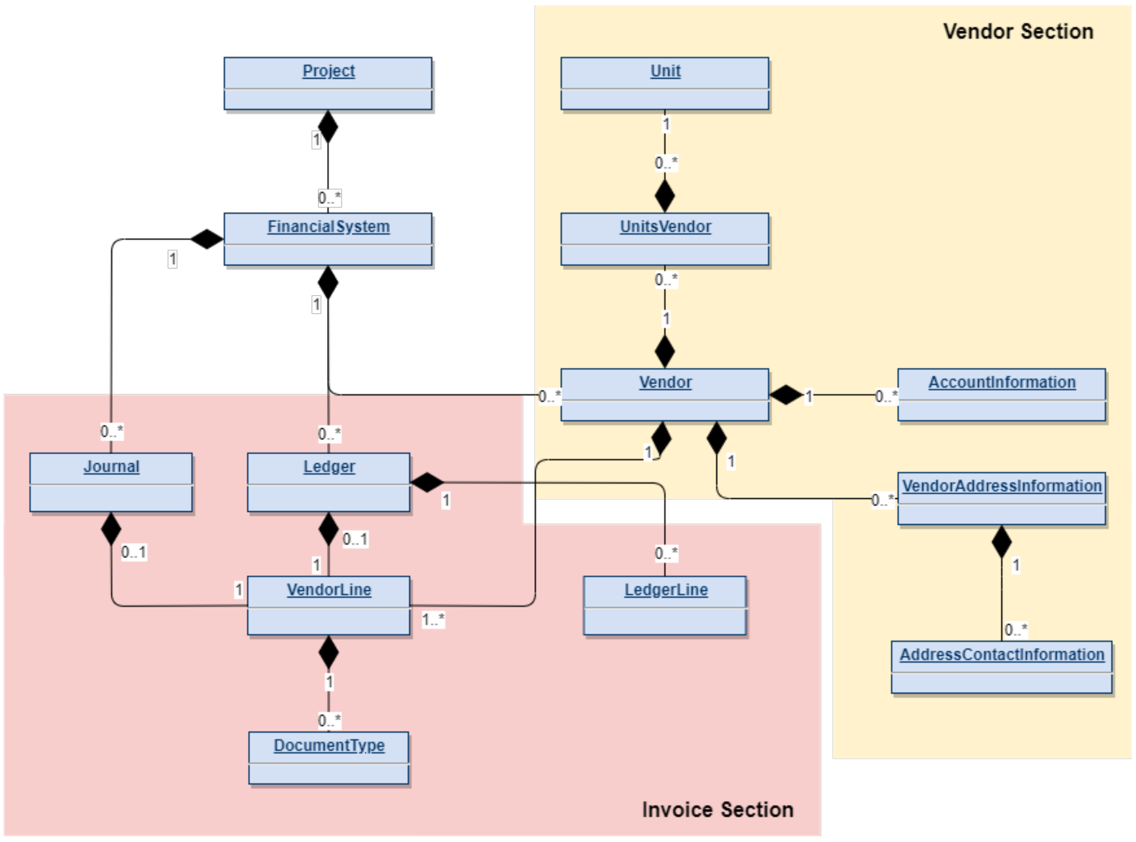


Figure 5.1: ER diagram representing the data warehouse

- Invoice Section- The invoice section consists of data related to the general ledger, invoices, and invoice lines (section 5.2). Noteworthy is that the vendor line can belong to either a ledger or journal. This relation is explained because accounting information systems either use ledgers in combination with document types (SAP) or Journals (EXACT, Navision). After discussing this matter with the stakeholders, it was decided not to combine these into a single entity, but rather keep them separated. Therefore, the vendor line has 0..1 relations, with the Journal, Ledger, and DocumentType entities. Furthermore, there is a one-to-many relation between Vendor and VendorLine. Because multiple invoices can belong to the same vendor. In other words, multiple goods were ordered at the same vendor. Through this one-to-many relation, the invoice section and the vendor section can be joined together.

5.4 Conclusion

In order to make the proposed integration system flourish, the data warehouse component has to be set up correctly. That is, the underlying model must be based on the commonality between ERP/Financial accounting systems, thereby making it possible to map each ERP/Financial accounting systems to the data warehouse. Moreover, given the requirements of the proposed integration system, the relations present in the ERP/Financial accounting systems must remain intact. ERP systems seek to integrate the complete range of business processes, especially procurement, material management, production, logistics, maintenance, sales distribution, financial accounting, asset management, cash management, controlling, strategic planning, and quality management. Its goal is to organize, define, and manage these processes to plan and control the organization effectively. Contrary to ERP systems, which focus on the complete range of business processes, accounting information systems are responsible for the collection, storing, and processing of financial and accounting data. Accounting information systems are a part of enterprise resource planning systems. The data SpendLab retrieves from these systems is composed of:

- Master data of the vendors, e.g., the vendor name, vendor number, VAT number, the address of the vendor, the country of the vendor.
- General ledger, basically the general ledger, centralizes the accounting data. Data of the general ledger consists of descriptions, balances, and total amounts for each account.
- Bank details of the vendors; this contains the bank names, account numbers, and IBANs of the vendor.
- Contact information of the creditors; this contains email-addresses and phone numbers.
- Invoices, an invoice is a commercial document issued by a seller or buyer, relating to a sale transaction. The typical invoice contains a unique reference, date, credit term, tax amount, details of the seller, purchase order number, description, the amount charged.
- Invoice lines, the invoice line describes a single product/item. These lines make up an invoice, i.e., an invoice consists of one or more invoice lines. Typically an invoice consists of unique reference, date, tax amount, description, amount.

We designed a mediator schema consisting of two sections. The vendor section includes data related to the master data, bank details, and contact information of the vendor. And, the invoice section consists of data related to the general ledger, invoices, and invoice lines. By a relation between the Vendor and Invoice entity, the two sections can be joined together. We use the mediator schema as the underlying model for the data warehouse.

Chapter 6

Schema matcher

The previous chapter described one of the main components, namely the data warehouse. This brings us to another main component, which is the semantic mapping component. In section 6.1, we describe why we created the semantic mapping component. Section 6.2 describes the setup of the semantic mapping component. After which in section 6.3 & 6.4, we describe the algorithms.

6.1 Motivation

In chapter 4, we explained the architecture of the proposed integration system and its requirements. Based on the requirements (chapter 2), we decided to use a schema-based integration system. Hence the proposed integration system is based around a data warehouse (chapter 4). As a result, each financial system that we integrate has to be mapped to the data warehouse. After which the mapping can be reused. However, creating this mapping can be a tedious process (chapter 3). Moreover, the users (data specialists) have no profound knowledge of SQL or other query languages (chapter 2). For these reasons, we aim to ease up creating the semantic mappings. We think this can be achieved by proposing semantic mapping through algorithms. As a result, the user only has to examine the resulting semantic mappings, which reduces manual effort.

6.2 Setup

There are two different setups for schema matching. We classify the two different schema matching setups as a combined matcher or hybrid matcher (chapter 3). We choose to implement a combined matcher setup such that new semantic mappings can be easily added and old algorithms easily removed. The combined matcher setup is chosen mainly because of its flexibility and extensibility. The multiple different financial systems encountered might benefit from different matcher set-ups. For example, for SAP systems, matcher set **A** might work best, whereas other financial systems benefit from different set-ups. Secondly, we might want to create system-specific schema matching algorithms. Moreover, if we want to extend the proposed integration system with new state of the schema matching algorithms, then this is more easily achieved using the combined matcher.

The drawback of using the combined matcher setup is that the result of each separate individual matcher has to be combined [38] [37]. However, during this thesis, we regard the implemented matching algorithms separately.

The result of the schema matcher will be a set of correspondences, which indicate how the schemas of financial systems relate to the mediated schema of the data warehouse. Therefore, it is necessary to define the structure of a correspondence. We define a correspondence as a tuple consisting of:

- FromTablename (source schema)

- FromColumnName (source schema)
- ToTableName (target schema)
- ToColumName (target schema)

Our correspondence structure relates a column from a table in the source schema to a column in the target schema.

In chapter 2, the current situation of the integration process is described. As shown when a new system is encountered, then the data specialist analyses the table names, the column names, and the data that is stored within each column. If, for example, the table name is TIBAN, the column name is IBAN, and data stored within this column adheres to the IBAN pattern (AF21 1422 5251 7313) then the analyst will make the decision this column contains IBANs. Based on this knowledge, the data specialist decides how the column should be mapped. This example nicely describes the two categories of schema matching algorithms.

1. Schema level: the analyst inspects the table and column names
2. Instance-level: the analyst examines the data stored in the column

Therefore, we created two schema matching algorithms. The first matching algorithm is schema-based. The second matching algorithm is instance-based.

6.3 Schema-based

The schema-based matcher is linguistic-based and uses the table and column names to find semantically similar elements. Name matching can be done in the following ways: [37]

- Equality of names.
- Equality of canonical name representations after stemming and other preprocessing.
- Equality of synonyms.
- Equality of hypernyms.
- Similarity of names based on common substrings, pronunciation, soundex, etc.
- User-provided name matches

Algorithms that determine a match based on similarity metrics can be categorized into two groups [32] [13]:

- Character-based
- Token-based similarity measures

Character-based methods are based on the character composition. The most common character-based measure is the edit distance. The edit distance can be defined as: Given two strings a and b on an alphabet $\sum$, the edit distance $d(a, b)$ is the minimum-weight series of edit operations that transform a into b. There exist multiple algorithms that determine the edit distance for example:

- Levenshtein [29]
- Jaro distance [22]
- Q-grams [34]

Token-based methods are used for recognizing the rearrangement of words, for example, "Invoice Line" and "Line Invoice." Token-based methods split the strings into substrings. Examples of token-based methods are:

- Jaccard similarity [20]
- Atomic strings [33]
- Cosine similarity [13] [10]

We implement a matcher based on equality of names in combination with a domain specific dictionary. The dictionary is used as an extension on the table and column names. And will be filled by the data specialist throughout integrating financial systems. Figure 6.1 illustrates the dictionary extending the table and column names. As can be seen, it is possible to have multiple translations. The proposed algorithm is character-based and uses the levenhstein metric. The levenhstein metric is used to determine the similarity between pairs of table and column names.

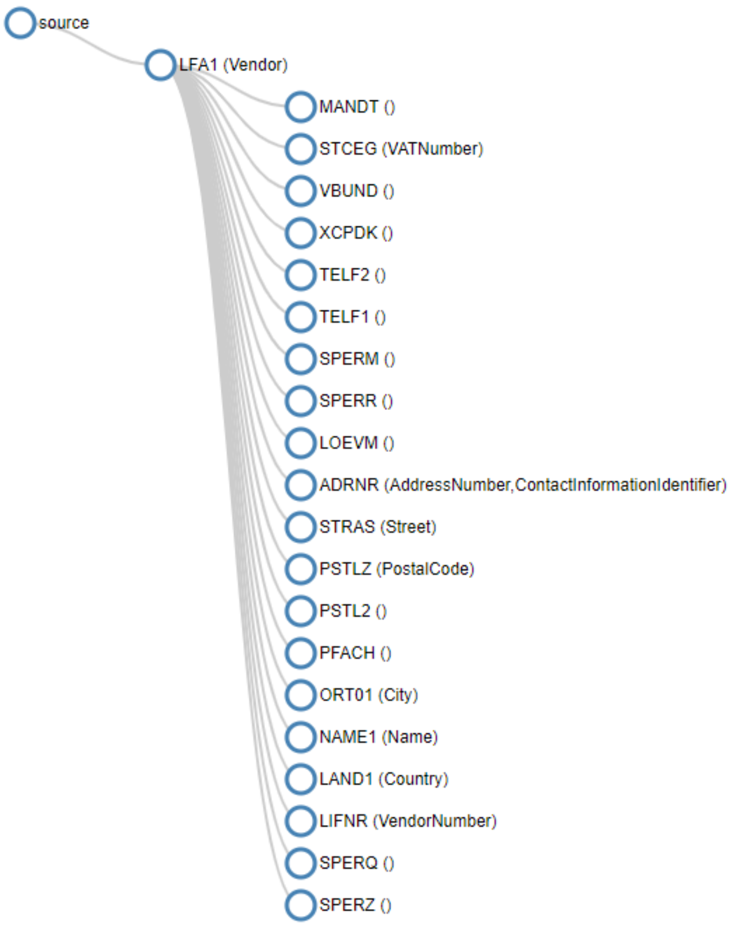


Figure 6.1: Schema consisting of one table named LFA1 with translations

The intuition behind the algorithm is that each combination of table and column pairs in the source schema are matched against each table and column pair in the target schema. For illustration purposes, we list some table and column pairs. Figure 6.1 shows the table LFA1 with its columns. Examples of table and column pairs are LFA1.MANDT, LFA1.STEC and LFA1.VBUND. If the similarity score between table and column pairs, from the source and target schema, is above threshold ρ, then the correspondence is added to the resulting correspondence set. We define the similarity score between a single combination of such pairs as follows:

$sim(s_1, s_2) = a \cdot prefix\_sim(s_1, s_2) + b \cdot suffix\_sim(s_1, s_2)$

- where a, b are constants such that $a + b = 1.0$.
- S_1, S_2 are tuples $\{t, c, t_1, c_1\}$ where t is the table name, c is the column name, t_1 are translations for t, c_1 are translations for c. S_1 relates to a tuple from the source schema, whereas S_2 relates to a tuple from the target schema.
- $prefix\_sim(s_1, s_2)$ is the levenhstein distance between the table name of $S_1 and S_2$.
- $suffix\_sim(s_1, s_2)$ is the levenhstein distance between the column name of $S_1 and S_2$.

The intuition translates to algorithms 1 & 2.

```
Data: Source schema $A$, Target schema $B$
Result: Correspondence set that relates schema $A$ to schema $B$
begin
    $result \longleftarrow []$
    $sourceTuples \longleftarrow ParseSchemaToTuples(A)$ $targetTuples \longleftarrow ParseSchemaToTuples(B)$
    for $source \in A$ do
        for $target \in B$ do
            $sim \longleftarrow CalculateScore(source, target)$
            if $score > \rho$ then
                $correspondence \longleftarrow$ Create correspondence between $source$ and $target$
                $result.add(correspondence)$
            end
        end
    end
    return $result$
end
```

Algorithm 1: GenerateCorrespondences

```
Data: source tuple $A$, target tuple $B$
Result: Similarity score between $A$ and $B$
begin
    $prefixScore \longleftarrow Levenhstein(A.t, B.t)$
    $suffixScore \longleftarrow Levenhstein(A.c, B.c)$
    for $tableNames \in A.t_1$ do
        $tempPrefixScore \longleftarrow Levenhstein(tableNames, B.t)$
        if $tempPrefixScore > prefixScore$ then
            $prefixScore = tempPrefixScore$
        end
    end
    for $columnNames \in A.c_1$ do
        $tempSuffixScore \longleftarrow Levenhstein(columnNames, B.c)$
        if $tempSuffixScore > suffixScore$ then
            $suffixScore = tempSuffixScore$
        end
    end
    return $((a \cdot prefixScore) + (b \cdot suffixScore))$;
end
```

Algorithm 2: CalculateScore

The first step of generating the correspondences is parsing the source and target schema to tuple sets. For every tuple in the tuple set of the source schema, we determine the similarity scores with the tuples in the tuple set of the target schema. As can be seen, the levenhstein distance is not only calculated based on the table and column name, but also the related translations from the domain-specific dictionary. Note that the highest score for both prefix and suffix is chosen. Given values for the constants a and b the final score is determined. If the final score is above the threshold ρ then the correspondence is added to the result set. Figure 6.2 shows an example of the resulting semantic mapping. On the left side, the source schema is shown, which consists of table LFA1. The right side shows the mediator schema of the data warehouse.

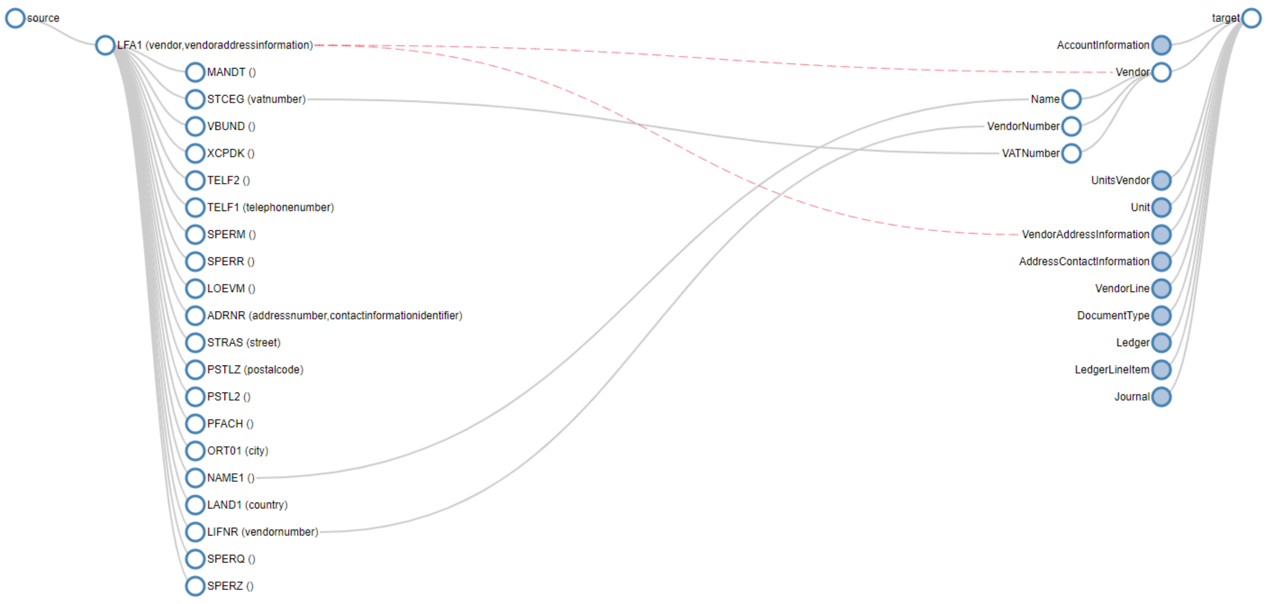


Figure 6.2: Semantic mapping that relates LFA1 to the Vendor table of the data warehouse

6.4 Instance based

In contrast to the schema-based matcher, the instance-based matcher relies solely on the data stored inside the tables. Therefore, whereas the schema-based matcher relies heavily on a knowledge base, thesaurus, or other word matching databases, the instance-based matcher does not have this drawback [9].

Spendlab Recovery operates in the financial domain. More specifically, the data retrieved are the ledger mutations, invoice and invoice lines, vendor data, payment data, and purchase order data (chapter 5). Because Spendlab Recovery operates on such a specific domain, we propose to perform a feature engineering to classify the columns of a table. Based on the classification of the columns, we then classify the table. Finally, based on the classification of the tables and their columns, we determine the semantic mapping. Intuitively this translates to an algorithm consisting of the following steps:

1. Classify the data warehouse; this classification always remains the same unless the mediator schema describing the data warehouse changes.

2. Classify the financial system; this is achieved through data mining and information retrieval techniques.

3. Determine correspondences between the classified tables of the financial system and the data warehouse. If the table from the classified financial system has the same classification as a table from the classified data warehouse, then the two tables relate.

4. Determine column correspondences between the related tables **A** and **B**. If a column from classified table **A** has the same classification as a column from table **B** then the columns relate.
5. Based on the related tables and sequentially, their related columns create the set of correspondences.

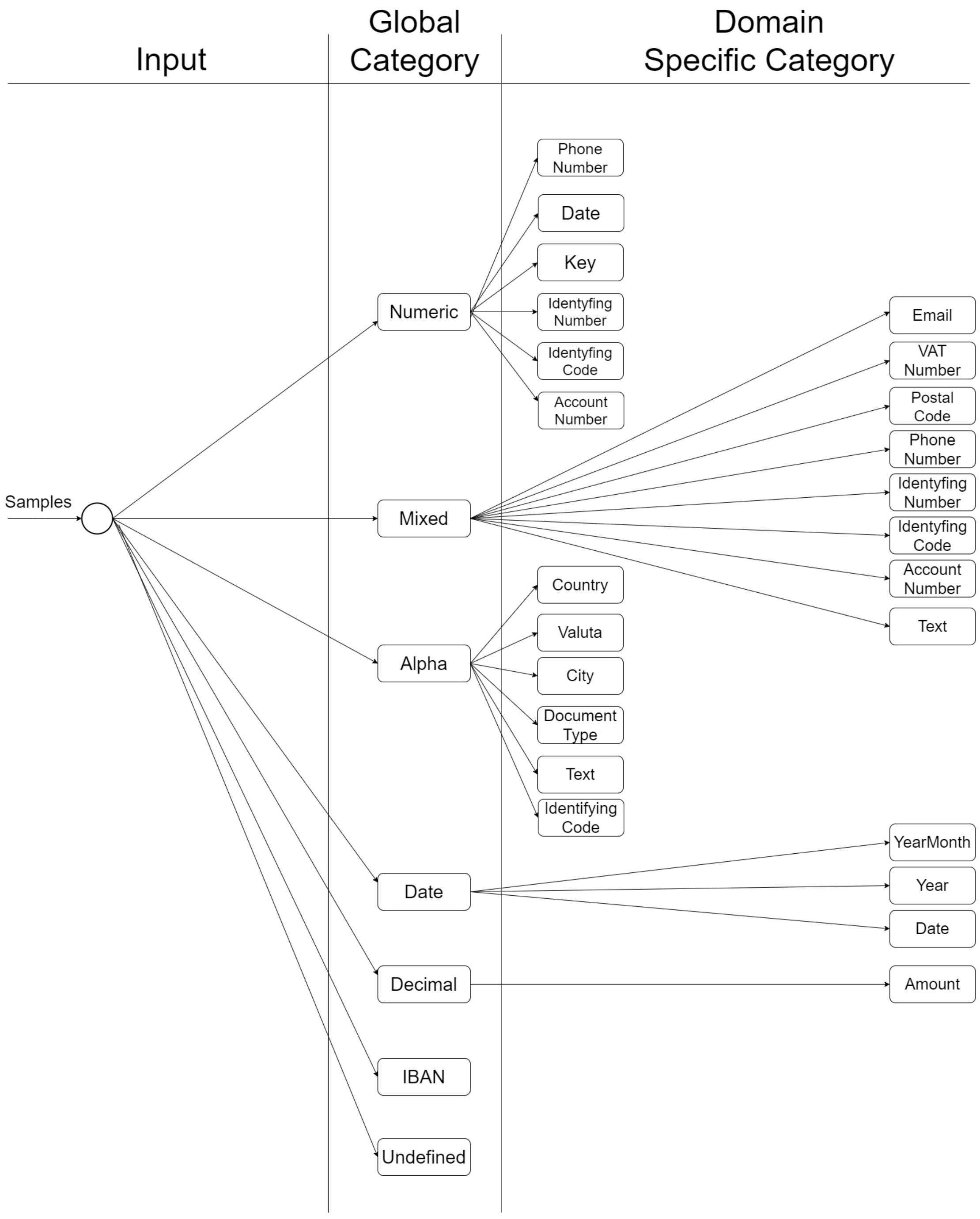


Figure 6.3: The process of classifying columns that the instance-based algorithm implements

Figure 6.3 illustrates the process of classifying a single column. The first step is to take samples of the data stored in the column. These samples are then given a global category. We determine the global category based on regexes. However, instead of using three categories: string, number, and mixed [32]. We introduce four extra global categories. The seven global categories are:

- Numeric- This category describes numeric data only.
- Alpha- This category describes alphabetic data only.
- Mixed- This category describes alphanumeric data, including special characters.
- Decimal- This category describes decimal values only.
- Date- This category describes date values only.
- IBAN- This category describes values that satisfy the international banking account number standard.

.

```
Data: source schema $A$
Result: Mappings between source schame and data warehouse
begin
    $categorizedTables \longleftarrow []$
    for $table \in A$ do
        $tempTable \longleftarrow$
        $tempTable.Name \longleftarrow table.Name$
        $tempTable.CategorizedColumns \longleftarrow []$
        for $column \in table.Columns$ do
            $samples \longleftarrow RetrieveRandomSamples(table, column)$
            $globalCategory \longleftarrow DetermineGlobalCategory(samples)$
            if $globalCategory = Numeric$ then
                $domainCategory \longleftarrow applyNumericRules(samples, tempTable)$
            end
            if $globalCategory = Alpha$ then
                $domainCategory \longleftarrow applyAlphaRules(samples, tempTable)$
            end
            if $globalCategory = Mixed$ then
                $domainCategory \longleftarrow applyMixesRules(samples, tempTable)$
            end
            if $globalCategory = Decimal$ then
                $domainCategory \longleftarrow applyDecimalRules(samples, tempTable)$
            end
            if $globalCategory = Date$ then
                $domainCategory \longleftarrow applyDateRules(samples, tempTable)$
            end
            if $globalCategory = Iban$ then
                $domainCategory \longleftarrow applyIbanRules(samples, tempTable)$
            end
            $tempTable.CategorizedColumns.add(domainCategory)$
        end
        $categorizedTables.add(tempTable)$
    end
    $mapping \longleftarrow MapCategorizeTableWithDatawarehouse(categorizedTables)$
end
```

Algorithm 3: Pseudo-code of the instance based semantic mapper

Based on the given global category, we apply different data mining techniques to determine metrics. For example, distribution, the length distribution of the data, average values, average numeric difference between data values, maximum value, minimum value, the sum of the values, but also using the GeoNames data set to determine how many values in the samples are a valid city. Using these metrics, we can classify the column. For example, a column given alpha as global classification will only get domain classification city if the percentage of valid city's amongst the samples is greater than 70%. Another example is that by looking at the length distribution and average length, we are able to differentiate between IdentyfingNumber and Accountnumber. As can be seen, we differentiate the techniques because for numeric values, we do not want to check if these values are valid city's. Moreover, a column containing only alpha values will never contain e-mail addresses. As a result, we spare execution time because we have to calculate fewer metrics. The used techniques are based on data mining and information retrieval methods. This process is applied for every column in the financial system and resulted in algorithm 3After all the columns are classified, we use this knowledge to determine table categories. Table 6.1 shows the fully classified data warehouse. We use this knowledge to set up the table classes based on column classes. For example, if the set of classified columns from a table of the financial system contains IdentyfingNumber, VATNumber, and Text, then we classify the table as Vendor. Another example is if the set of classified columns contains Iban or AccountNumber, then the table is categorized as AccountInformation. Finally, we use the classified table and the classified data warehouse to create the semantic mappings. The semantic mapping is created using steps 3, 4, and 5.

Table	Column	Column classification
AccountInformation	AccountNumber VendorNumber IbanNumber	AccountNumber VendorNumber Iban
Vendor	VendorNumber VATNumber Name	IdentifyingNumber VATNumber Text
UnitsVendor	VendorNumber CompanyCode	IdentifyingNumber IdentifyingCode
Unit	Name CompanyCode	Text IdentifyingCode
AddressInformation	VendorNumber Street PostalCode TelephoneNumber Country City ContactInformationIdentifier	IdentifyingNumber Text PostalCode PhoneNumber Country City IdentifyingCode
AddressContactInformation	EmailAddress ContactInformationIdentifier	Email IdentifyingCode
Journal	JournalNumber Description	IdentifyingNumber Text
Ledger	LedgerCode LedgerNumber Description	IdentifyingCode IdentifyingNumber Text
LedgerLineItem	InvoiceNumber ReferenceNumber LedgerNumber Amount OriginalAmount Description	IdentifyingNumber IdentifyingNumber IdentifyingNumber Amount Amount Text

Table 6.1: The classification of the mediator schema describing the data warehouse

The classification of the mediator schema describing the data warehouse (continued)

Table	Column	Column classification
VendorLine	VendorNumber	IdentifyingNumber
	CompanyCode	IdentifyingCode
	InvoiceNumber	IdentifyingNumber
	ReferenceNumber	IdentifyingNumber
	DocumentTypeCode	DocumentType
	ClearingDocumentNumber	IdentifyingNumber
	Valuta	Valuta
	Amount	Amount
	VATAmount	Amount
	AmountExcludingVAT	Amount
	JournalNumber	IdentifyingNumber
	OriginalAmount	Amount
	DocumentDate	Date
	PostingDate	Date
	PostingKey	Key
	Description	Text
DocumentType	DocumentTypeCode	DocumentType
	DocumentDescription	Text

6.5 Conclusion

In this chapter, we describe the chosen algorithms, which we use to propose semantic mappings. Because the proposed integration system is based around a data warehouse, each financial system has to be mapped. Creating these mappings can be tedious. Therefore, we aim to ease up this process by proposing semantic mappings. For this, we chose to implement a combined matcher setup. We decided to implement the combined matcher because of its flexibility and extensibility. Two algorithms are implemented, namely a schema-based algorithm and an instance-based algorithm. Whereas, the schema-based algorithm only considers data on the schema level. The instance-based matcher operates solely on the data stored in the table. Because we choose the combined matcher setup, we can easily add new (state of the art) algorithms in the future.

Chapter 7

Data exchange

In chapter 4, we described the architecture of the proposed integration system. The previous chapter describes the data warehouse (chapter 5) and the semantic mapping component (chapter 6). This chapter will describe another main component of the proposed integration system, which is the query generator. In section 7.1, we describe the motivation behind the query generator. After this, in section 7.2, we explain how we implemented the query generator.

7.1 Motivation

Thus far, we described the architecture of the proposed integration system (chapter 4) and two of the main components. The data warehouse (chapter 5) and the semantic mapper component (chapter 6). These chapters describe how the integrated data will be stored and how we aim to ease up creating the necessary semantic mappings. Another step is moving the data from the financial systems into the data warehouse. Moving the data can be achieved by writing queries that transfer the data from (tables of) the financial system into the data warehouse. However, the users (data specialists) have no profound knowledge of SQL or other query languages. Therefore, writing these queries becomes very tedious, time-consuming, and error-prone. Hence, we aim to fully automate this by given the financial system, the data warehouse, and the semantic mapping generate these queries.

7.2 Implementation

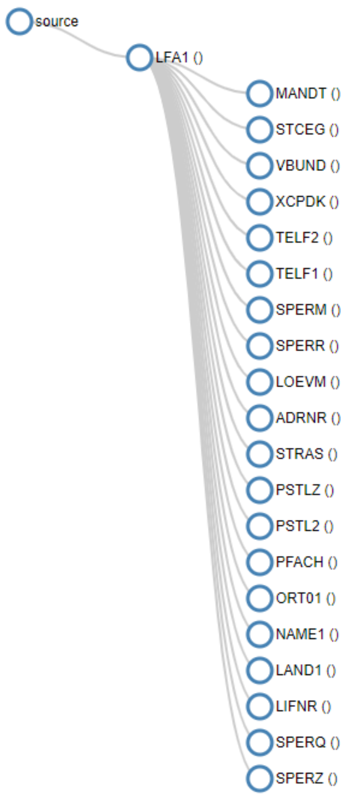
The goal of the query generator is given a financial system, the data warehouse and the semantic mapping generate a query that transfers the data from the financial system into the data warehouse. Lots of research has been done on query generation and reformulation. This research is described in chapter 3. However, there has been done little research into given a semantic mapping, which is partially generated by algorithms described in chapter 6, generate a query that transfers the data. As a matter of fact, we only found the CLIO project that describes data exchange [15]. Whereas CLIO for data exchange, given a source instance, will produce an instance of the target schema that satisfies the mapping, and that represents the source data as accurately as possible. We propose a query generator that given a financial system (source instance). Generate a query, that transfers the data from the financial system into the data warehouse (target instance). For this, we developed an algorithm based on the CLIO project. The main difference between the CLIO approach and our approach is the following:

- CLIO can handle nested schema. In our approach, we only have to be able to map between relational schema.

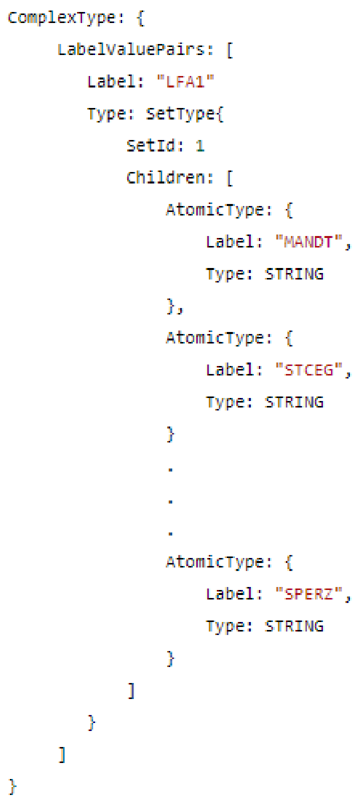
The query generator algorithm consists of the following steps:

1. Parse source and target schema to standard representation.
2. Generate relational integrity constraints for the source schema.
3. Generate relational integrity constraints for the target schema.
4. Generate the maximal set of logically related elements in source schema (through chase algorithm).
5. Generate the maximal set of logically related elements in source schema (through chase algorithm).
6. Given the semantic mapping and the logically related elements for the source and target schema, generate the source-to-target dependencies.
7. Use the source-to-target dependencies to create the MSSQL query.

7.2.1 Schema representation



(a) Source schema consisting of one table named LFA1



```
ComplexType: {
    LabelValuePairs: [
      Label: "LFA1"
      Type: SetType{
         SetId: 1
         Children: [
            AtomicType: {
               Label: "MANDT",
               Type: STRING
            },
            AtomicType: {
               Label: "STCEG",
               Type: STRING
            }
            .
            .
            .
            AtomicType: {
               Label: "SPERZ",
               Type: STRING
            }
         ]
      }
    ]
}
```

(b) Standardized schema consisting of one table named LFA1

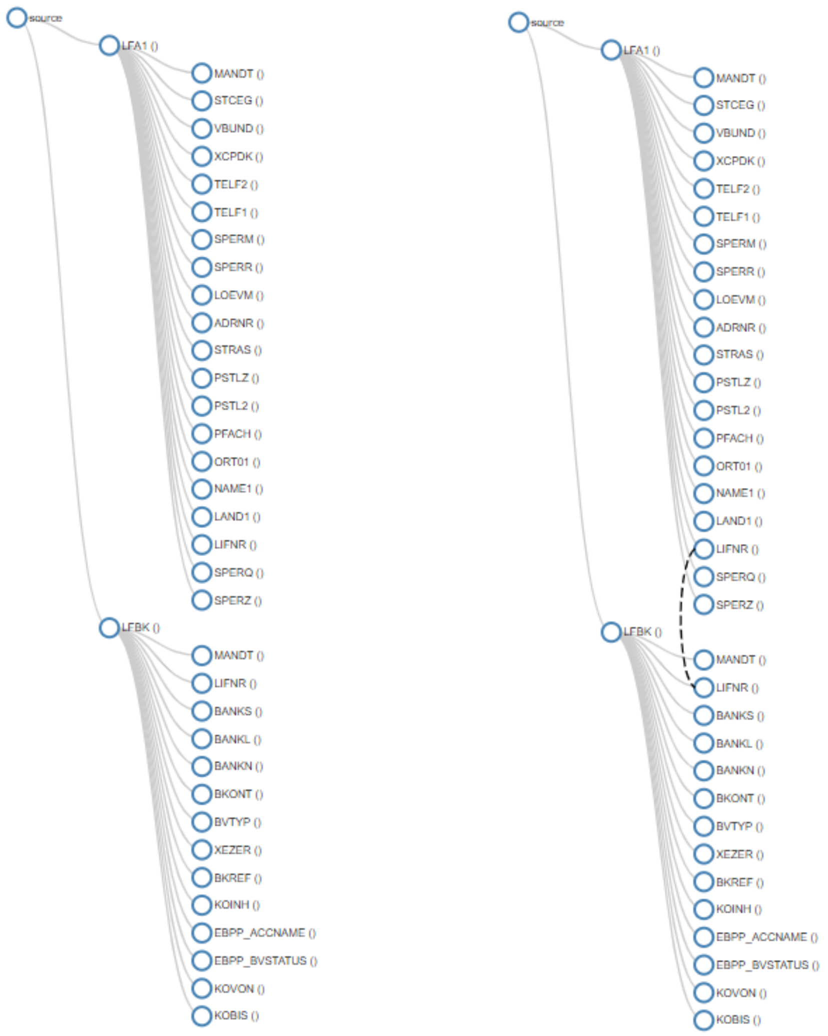
Figure 7.1: Source schema A represented normally and standardized

The first step of the Query generator is parsing the source and target schema to a standard representation. Figure 7.1 illustrates a schema (left) and its standard representation (right). For the standard representation, we use a nested referential model [15]. This model consist of three types:

- **AtomicType**, AtomicTypes contain the basis and relate to a column. AtomicTypes consist of a Label and a DataType. Datatypes can be String, Boolean, Integer, and Double.
- **SetType**. SetTypes consist of a SetId and a Collection of AtomicTypes or ComplexTypes.
- **ComplexType**, a ComplexType, basically is a collection of LabelValuePairs.
- **LabelValuePairs**, labelValuePairs consist of name and type. Type can be either an AtomicType, ComplexType, or SetType.

By using this definition, the schema can be modeled as a ComplexType, whose LabelValuePairs contain SetTypes, the SetTypes contain a collection of AtomicTypes.

7.2.2 Relational integrity constraints



(a) Source schema consisting of two tables named LFA1 and LFBK

(b) Source schema consisting of two tables named LFA1 and LFBK with a foreign key

Figure 7.2: Source schema with and without foreign key relation

Integrity constraints are rules that enforce basic information based constraints. These rules ensure data accuracy and consistency. Relational integrity constraints are such rules that are specifically specified between two tables. Take, for example, figure 7.2; on the right side, we see two tables with a foreign key. Meaning that every value of the foreign key in LFB1 must be available in LFA1. This constraint is a relational integrity constraint. It is important to satisfy these constraints when generating the query,

In order to generate the relational integrity constraints, we, first of all, represent the related schema elements without constraints. For example, in figure 7.2, all the columns of LFA1 are related without any constraint. In short, all the columns in a single table are related to each other. We define the set of related elements without any constraints as a **primary path** [15]. In figure 7.2, the **primary paths** will be:

- lfa1 in LFA1
- lfbk in LFBK

After the generation of the primary paths, we generate the referential constraints. We define a referential constraint as:

- **foreach** P_1 **exists** P_2 **where** B
  - P_1 is a Primary Path
  - P_2 is a Primary Path
  - B is a conjunction of equality's in the form $e_1 = e_2$

In short primary paths represent related schema elements without constraints, whereas referential constraints represent related schema elements through constraints. In figure 7.2, representation A does not have any referential constraints because it does not contain a foreign key; therefore, it does not relate two primary paths (schema elements) through a constraint. Representation B will generate the following referential constraint:

- **foreach** lfa1 in LFA1 **exists** lfbk in LFBK **where** lfa1.LIFNR = lfbk.LIFNR.

7.2.3 Associations

After determining the primary paths and the relational integrity constraints, the set of associations is generated. Formally, an association is a form of a query (with no return or select clause) over a single schema; intuitively, all the atomic type elements reachable from the query variables are considered to be "associated" [15]. We define two types of associations, namely:

- Structural association: a structural association is an association defined solely by a primary path P; thus, having the following form: **from** P [15].
- Logical association: a logical association specifies semantic relations between schema elements using constraints; thus, having the following form: **from** $P_1 ... P_n$ **where** B [15]. B is a conjunction of equality's between expressions over $P_1 ... P_n$.

We define an association as **from** X **where** B, where X is the set of primary paths, and B the set of expressions. The set associated elements are determined using the chase algorithm [35]. Briefly, a chase step is applied to association A using a referential constraint (**foreach** X **exists** Y **where** B). If association A contains the primary path X, but Y is not contained, then the Y clause and B expressions are added to the association. This is shown in algorithm 4. If we take the schema A from figure 7.2, then the resulting logical associations will be:

- **from** lfa1 in LFA1

- **from** lfbk in LFBK

If we take schema B from figure 7.2, then the resulting logical associations will be:

- **from** lfa1 in LFA1, lfbk in LFBK **where** lfa1.LIFNR = lfbk.LIFNR

```
Data: primary paths $P$, relational integrity constraints $N$.
Result: The set of logical associated elements
begin
    $logicalAssociatedElements \longleftarrow []$
    for $p \in P$ do
        $association \longleftarrow$ new $Association()$
        $association.X.add(p)$
        for $n \in N$ do
            if $association.X.contains(n.X)$ & $!association.X.contains(n.Y)$ then
                $association.X.add(n.X)$
                $association.B.add(n.B)$
            end
        end
        $logicalAssociatedElements.add(association)$
    end
    return $logicalAssociatedElements$
end
```

Algorithm 4: Generate logical associated elements

7.2.4 source-target dependencies

```
Data: associations $A$, correspondences $C$
Result: The source to target dependencies
begin
    $stDependencies \longleftarrow []$
    $groupedCorrespondences \longleftarrow C.groupBy(ToTablename)$ for
     $g \in groupedCorrespondences$ do
        $association \longleftarrow findSourceAssociation(A, g.Values)$
        if $association! = null$ then
            $stDependency \longleftarrow$ new $StDependency()$
            $stDependency.A = association$
            $stDependency.T = g.Key$
            $stDependency.C = g.Values$
            $stDependencies.add(stDependency)$
        end
    end
    return $stDependencies$
end
```

Algorithm 5: Generate source-to-target dependencies

As described in chapter 6, semantic mappings are created. Relating elements between the financial system and the data warehouse. The semantic mappings are a set of correspondences. Given a semantic mapping, which relates elements between the financial system and the data warehouse, we generate source-to-target dependencies. We define source-to-target dependencies as $std = \{A, T, C\}$, where A = Association, T = TargetTableName and C = set of correspondences.

We generate the source-to-target dependencies by grouping the correspondences of the semantic mapping using $ToTablenName$. We group because we want to generate a single query per table. For each group, we search for the association (section 7.2.3). This results in the algorithm 5. Lastly, we translate the generated source-to-target dependencies into (SQL) insert into queries. For example, given the mapping shown in figure 7.3 will result in the following insert queries:

```
INSERT INTO Vendors (VATNumber,VendorNumber,Name, DataStatus, ProjectId)
SELECT [lfa1].[STCEG],[lfa1].[LIFNR],[lfa1].[NAME1], 0, 1
FROM [db86561a4074f54cc299ebb7630756b975].[dbo].LFA1 AS lfa1,
[db86561a4074f54cc299ebb7630756b975].[dbo].LFBK AS lfbk
WHERE lfa1.LIFNR = lfbk.LIFNR


INSERT INTO AccountInformations (VendorNumber,Identifier,AccountNumber,
DataStatus, ProjectId)
SELECT [lfbk].[LIFNR],[lfbk].[BANKL],[lfbk].[BANKN], 0, 1
FROM [db86561a4074f54cc299ebb7630756b975].[dbo].LFA1 AS lfa1
[db86561a4074f54cc299ebb7630756b975].[dbo].LFBK AS lfbk
WHERE lfa1.LIFNR = lfbk.LIFNR
```

These insert into queries are collected and executed in a single transaction.

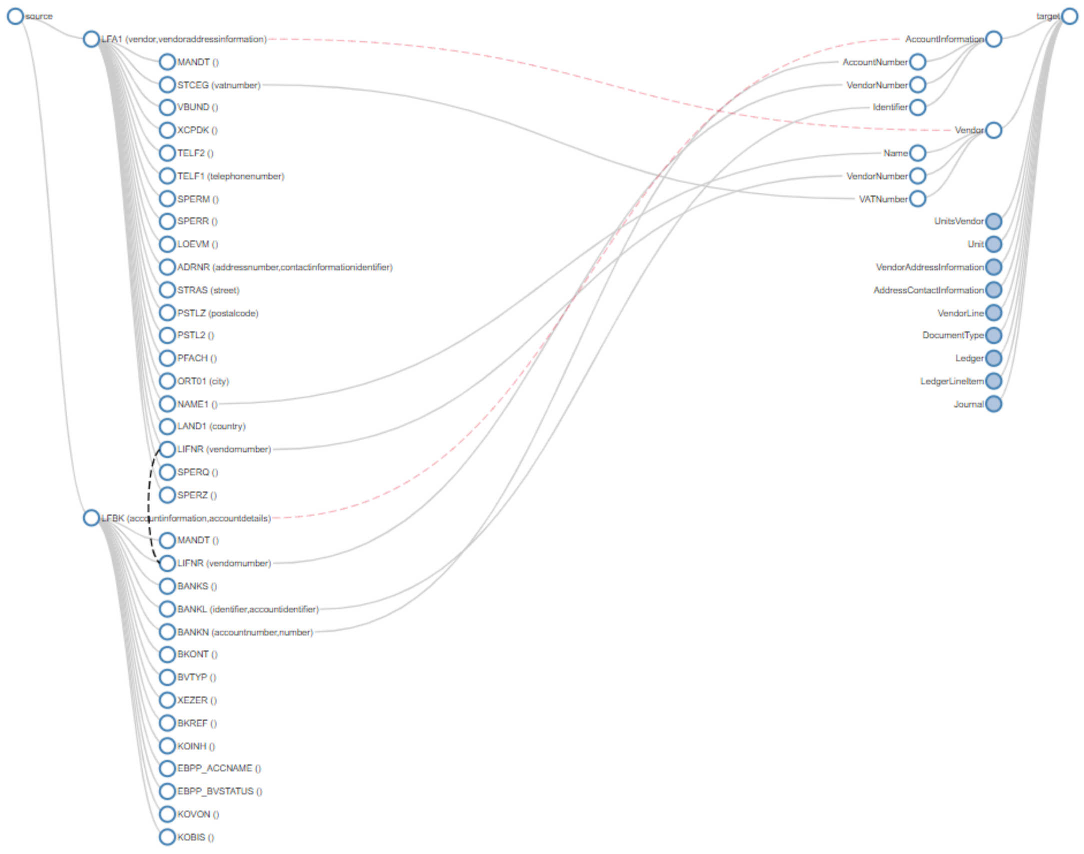
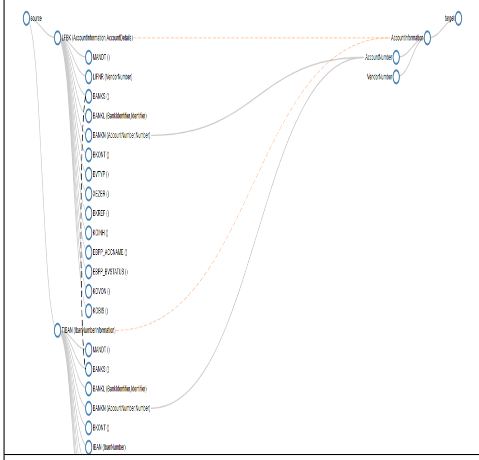


Figure 7.3: Semantic mapping that relates LFA1 and LFBK to the AccountInformation and Vendor table

Table 7.1 lists some more examples of semantic mappings and the generated queries. As can be seen, we support N:1 relations by concatenating.

Semantic mapping	Generated MSSQL query

 source
target
LFBK (AccountInformation,AccountDetails)
MANDT ()
LIFNR (VendorNumber)
BANKS ()
BANKL (BankIdentifier,Identifier)
BANKN (AccountNumber,Number)
BKONT ()
BVTYP ()
XEZER ()
BKREF ()
KOINH ()
EBPP_ACCNAME ()
EBPP_BVSTATUS ()
KOVON ()
KOBIS ()
TIBAN (IbanNumberInformation)
MANDT ()
BANKS ()
BANKL (BankIdentifier,Identifier)
BANKN (AccountNumber,Number)
BKONT ()
IBAN (IbanNumber)
AccountInformation
AccountNumber
VendorNumber
	BEGIN TRAN
	
	INSERT INTO AccountInformations (AccountNumber)
	SELECT CONCAT([tiban].[BANKN],[lfbk].[BANKN])
	FROM LFBK AS lfbk, TIBAN AS tiban
	WHERE lfbk.BANKS = tiban.BANKS
	
	COMMIT TRAN

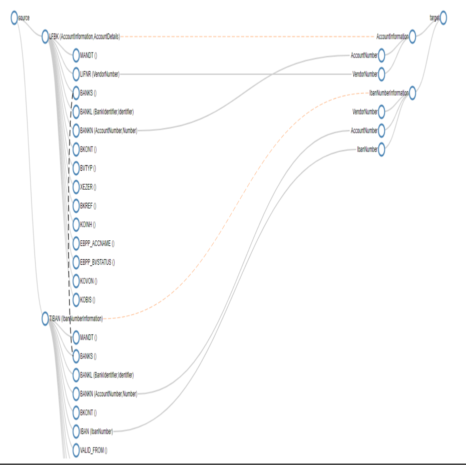
 source
target
LFBK (AccountInformation,AccountDetails)
MANDT ()
LIFNR (VendorNumber)
BANKS ()
BANKL (BankIdentifier,Identifier)
BANKN (AccountNumber,Number)
BKONT ()
BVTYP ()
XEZER ()
BKREF ()
KOINH ()
EBPP_ACCNAME ()
EBPP_BVSTATUS ()
KOVON ()
KOBIS ()
TIBAN (IbanNumberInformation)
MANDT ()
BANKS ()
BANKL (BankIdentifier,Identifier)
BANKN (AccountNumber,Number)
BKONT ()
IBAN (IbanNumber)
VALID_FROM ()
AccountInformation
AccountNumber
VendorNumber
IbanNumberInformation
VendorNumber
AccountNumber
IbanNumber
	BEGIN TRAN
	
	INSERT INTO AccountInformations (VendorNumber,AccountNumber)
	SELECT [lfbk].[LIFNR],[lfbk].[BANKN]
	FROM LFBK AS lfbk,TIBAN AS tiban
	WHERE lfbk.BANKS = tiban.BANKS
	
	INSERT INTO IbanNumberInformations (AccountNumber,IbanNumber)
	SELECT [tiban].[BANKN],[tiban].[IBAN]
	FROM LFBK AS lfbk, TIBAN AS tiban
	WHERE lfbk.BANKS = tiban.BANKS
	
	COMMIT TRAN

Table 7.1: Examples of generated queries given semantic mappings.

7.3 Conclusion

Given semantic mappings between financial systems and the data warehouse, queries can be written that transfer the data from the financial systems into the data warehouse. However, because the users (data specialist) have no profound knowledge of SQL or other query languages, this becomes a time consuming, tedious, and error-prone process. Therefore, we implemented a query generator that given a financial system, the data warehouse, and the corresponding semantic mapping generates a query. This query can be executed to transfer the data from the financial system into the data warehouse. Because, CLIO was the only project encountered, during the literature review, that provides data exchange [15]. We based the implementation of the data exchange component on the CLIO project. The main difference between the CLIO approach and ours is that:

1. We can not handle nested schema. However, we do not encounter nested schemas.
2. Instead of materializing, we generate a query. We do this by grouping the correspondences by ToTableName and use these groups to generate the source to target dependencies. After this, we use these dependencies to create the insert into queries.

In the future, the algorithm can be extended to be able to handle nested schema. Consequently making it possible to handle document-oriented databases.

Chapter 8

Graphical User Interface

In the previous chapters, we described the architecture of the proposed integration system (chapter 4). After which we addressed three main components. These main components are:

1. Data warehouse (chapter 5)
2. Semantic mapping (chapter 6)
3. Query generator (chapter 7)

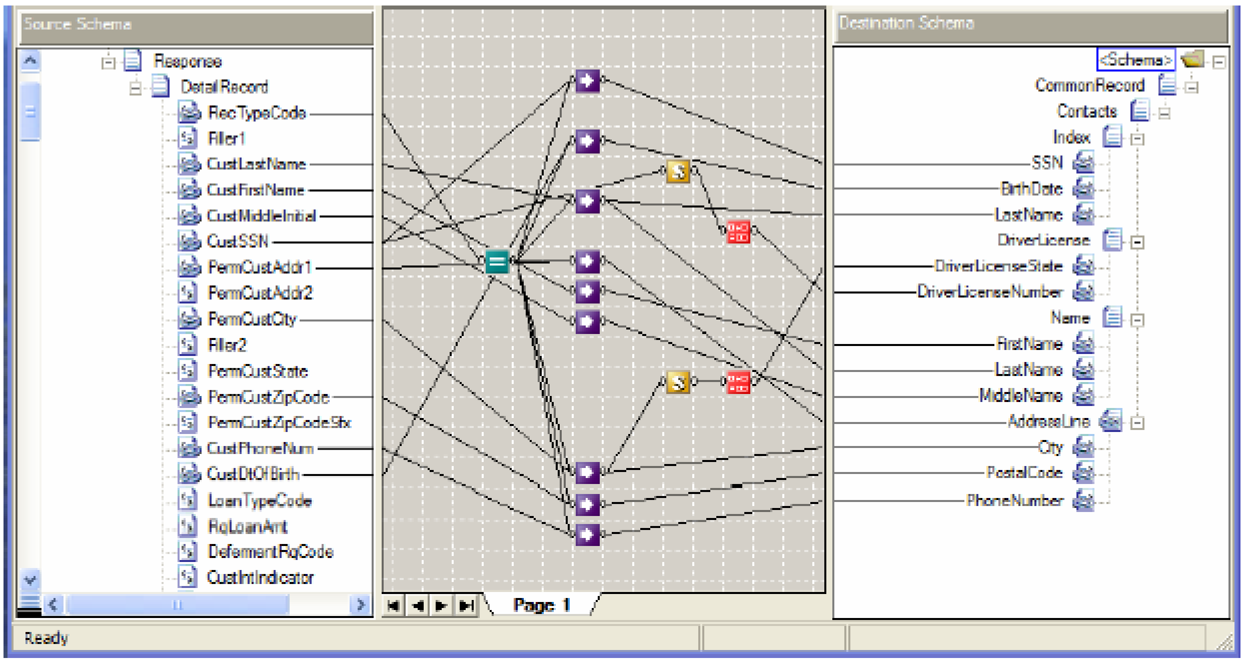
This brings us to the last main component of the proposed integration system. That is the graphical user interface. In section 8.1, we describe the motivation. Section 8.2 shows existing visualization techniques for schema matching. After which in section 8.3, we describe the implementation of the visualizations techniques we chose.

8.1 Motivation

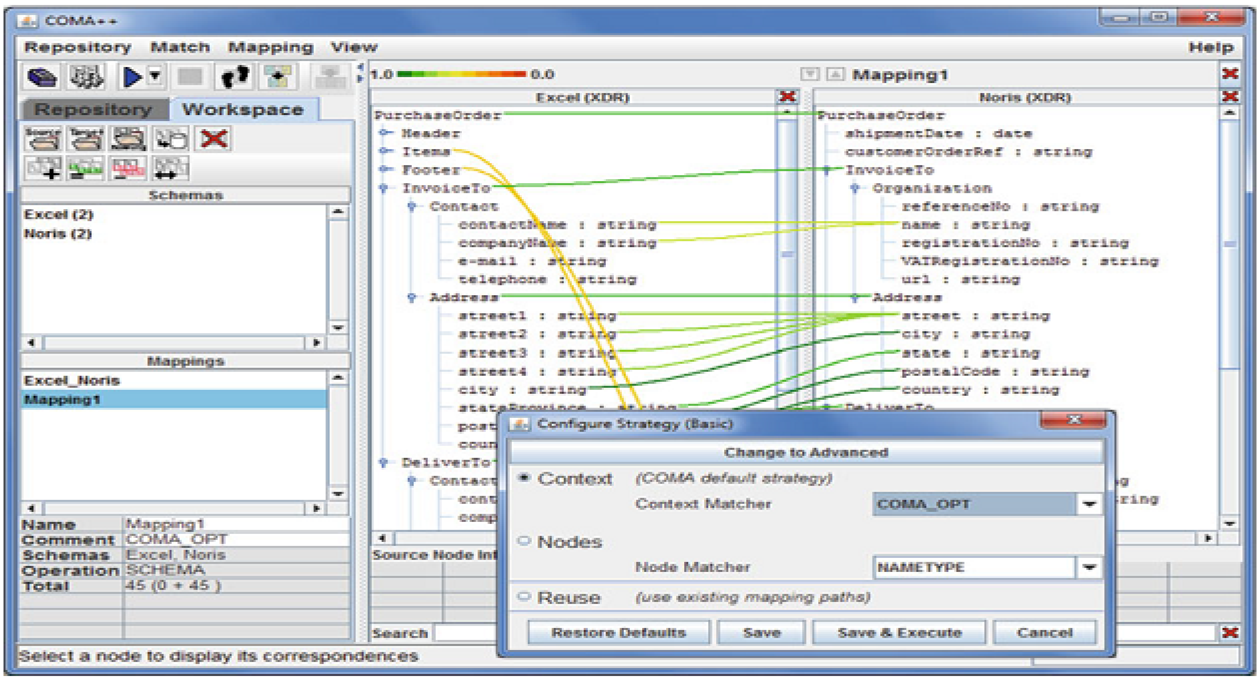
In the previous chapters, we described the technical components of the proposed integration system. As described in chapter 3, fully automatic schema matching is only possible if best-effort matching is satisfactory. As may be expected, this is not the case with Spendlab Recovery. Therefore, the users (data specialists) have to examine the generated semantic mappings. Because the users (data specialists) of the integration system are domain specialists rather than computer scientists, the proposed integration system requires a simple graphical user interface.

8.2 Existing visualization techniques

Schema matching is far from being a fully automated task. In cases where high precision is necessary, verification and fine-tuning the generated correspondences is required [17]. Spendlab Recovery operates in the financial domain; therefore, in order to make judgments about the generated correspondences, a data analyst is required to have a vast amount of domain knowledge, which a computer will never have. The task of the data analyst is to determine the correct correspondence, remove false-positives, and add missed correspondences. Research has largely been focused on improving the algorithms, whose purpose is to generate these correspondences. However, in cases where domain-knowledge is required, the users and tools must be paired together [5]. Therefore, lately, more and more research has been done on tools that support the semiautomatic process, resulting in a more human-centered approach [5]. Moreover, the international Work-shop on Ontology Alignment and Visualization was created to enable researchers to share and explore new visualizations techniques to support the matching. A current well-received solution is having the source schema on the left, the target schema on the right, which both are represented as tree



(a) UI of BizTalk



Name	Mapping1
Comment	COMA_OPT
Schemas	Excel, Noris
Operation	SCHEMA
Total	45 (0 + 45)

(b) UI of Coma++

Figure 8.1: Example of two different tools using the same solution for their GUI. Both tools have the source schema on the left and the target schema on the right represented as tree views, and the mapping shown as a network of links connecting elements from the source and target schema.

views, the mapping is shown as a network of links connecting elements from the source and the target schema.

Fig 8.1 shows two examples of systems, namely BizTalk and Coma++. Next to BizTalk and Coma++ [2] WebLogic Workshops, IBM websphere [26], TibCo BusinessWorks, Altova, Stylus Studio, Cape Clear, Sonic Software and ActiveState all use a similar solution [18]. Other approaches are a set of yes/no questions that the user must answer, a graph-based visualization of ontology's, which are then compared side by side, and a list view. [1] [17] [17].

As described, the most common method used is the system in which the source schema on the left, the target schema on the right, which both are represented as tree views, the mapping is shown as a network of links connecting elements from the source and the target schema. Figure 8.4 illustrates the drawback of this approach. As can be seen, if the schema or mappings become large, then the details of interests are lost. More specifically, the solution does not scale well to a large schema, and yet systems are getting bigger and bigger, also creating larger schema [18]. Consequently, it becomes harder to examine the mapping without much scrolling of both schemas. Moreover, having many mappings, therefore, having many lines, it will be hard to distinguish between mapped elements.

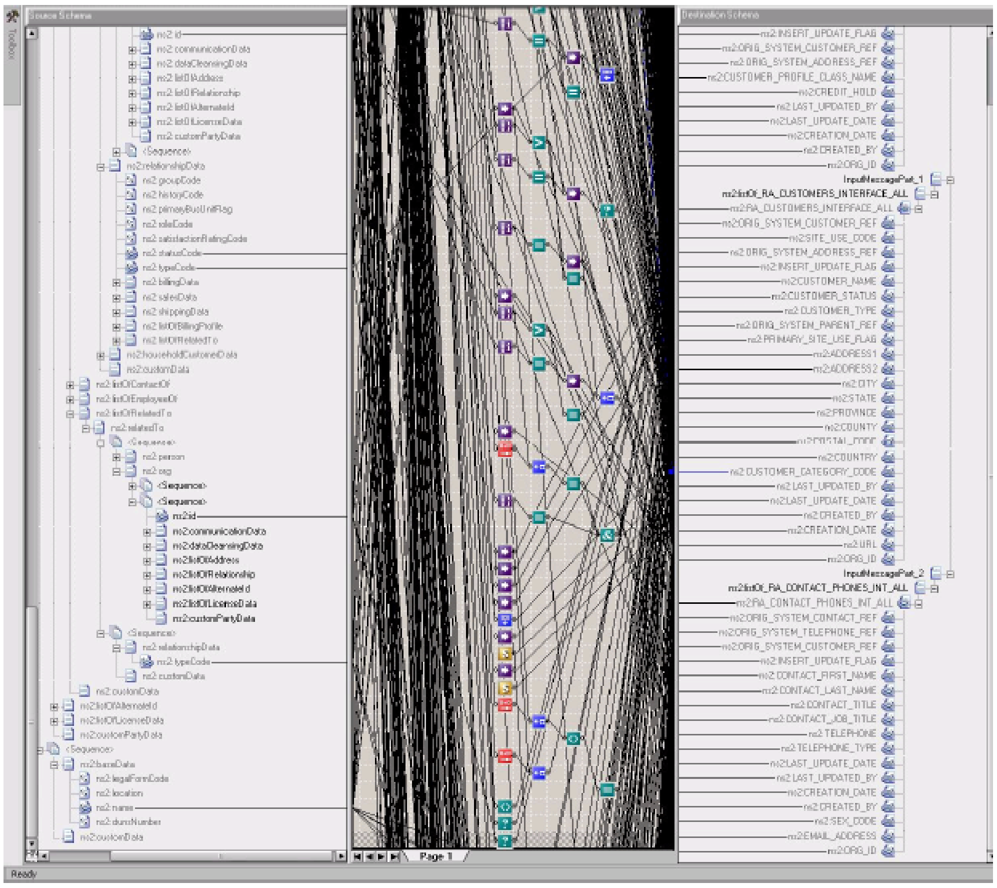


Figure 8.2: Example of large schema, losing details of interest

8.3 Implementation

The most used method is having the source schema on the left, the target schema on the right, which both are represented as tree views, the mapping is shown as a network of links connecting elements from the source and the target schema. Therefore, we also implement these visualization

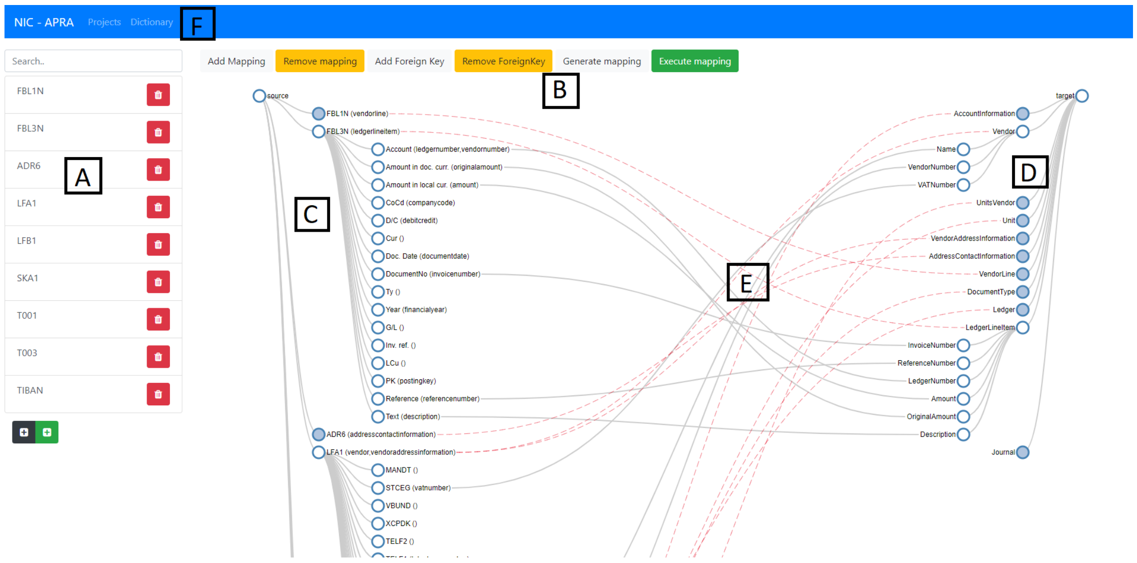


Figure 8.3: User Interface of DISAP

techniques. The techniques used for implementing the graphical user interface are HTML5, CSS, and Javascript. For the tree views and network of lines, we use D3.js [1] (chapter 4). Figure 8.3 illustrates the graphical user interface of the proposed integration system. The graphical user interface consists of:

- **A**. The side panel A visualizes all the added data source(s). It is possible to search through all added data sources through the search bar located on top of the panel. Furthermore, it is possible to delete each data source by clicking the red thrash-can of the specific data source. Data sources are added to the financial system by the two + buttons below the panel. The green one is used for the physical data source(s), whereas, the gray one is used for the virtual data source(s).

- **B**. B is the main toolbar for the integration process. If elements from the source and target schema are selected, which is done by clicking on the specific node(s) and the user presses the add mapping button than a mapping is made between the source and target elements. The same holds for adding a foreign key. Other functions that the user can perform are:

  – Remove mapping

  – Remove Foreign Key

  – Generate mapping

  – Execute mapping

- **C**. C is the source schema visualized as a Tree View. Each root node can be selected to execute the actions from the toolbar. Furthermore, the nodes with children can be collapsed.

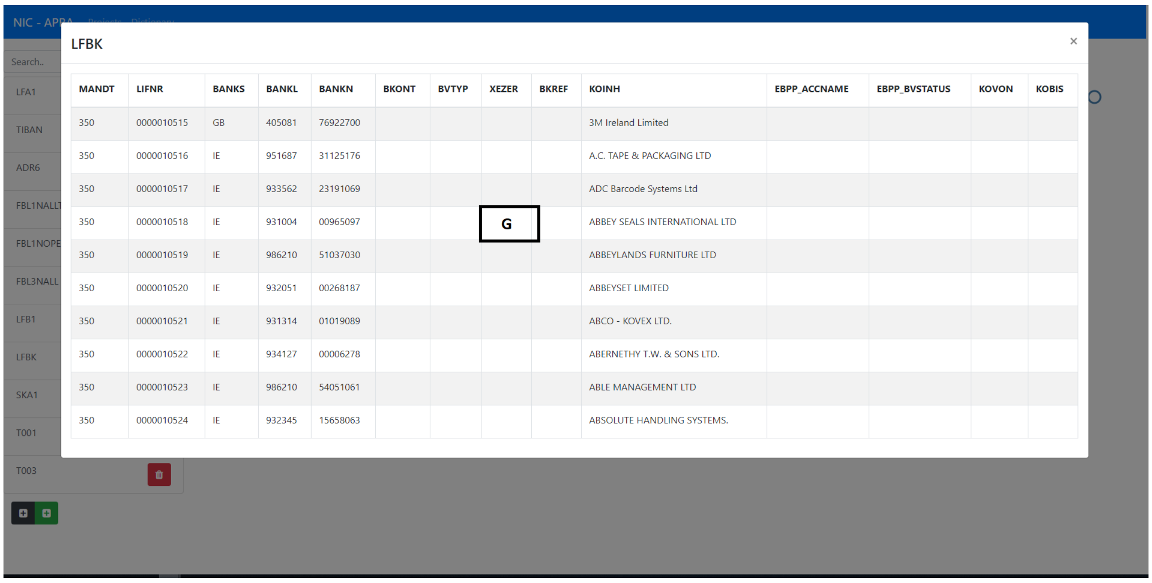
- **D**. D is the target schema visualized as a Tree View. It has the same functionality as **C**.

- **E**. E shows the line(s), which represents the mappings between a source schema and target schema. Line(s) are highlighted on hover. It is also possible to select a line in order to remove the mapping from the integration (see functionality **B**).

- **F**. F shows the menu bar. Through this menu bar, the user can go back to the projects section or the dictionary.

[1]https://d3js.org/

- **G**. If the user right-clicks on a node in **C**, then the data of the specific clicked table or column will be shown in a pop-up modal.



LFBK

MANDT	LIFNR	BANKS	BANKL	BANKN	BKONT	BVTYP	XEZER	BKREF	KOINH	EBPP_ACCNAME	EBPP_BVSTATUS	KOVON	KOBIS
350	0000010515	GB	405081	76922700					3M Ireland Limited				
350	0000010516	IE	951687	31125176					A.C. TAPE & PACKAGING LTD				
350	0000010517	IE	933562	23191069					ADC Barcode Systems Ltd				
350	0000010518	IE	931004	00965097					ABBEY SEALS INTERNATIONAL LTD				
350	0000010519	IE	986210	51037030					ABBEYLANDS FURNITURE LTD				
350	0000010520	IE	932051	00268187					ABBEYSET LIMITED				
350	0000010521	IE	931314	01019089					ABCO - KOVEX LTD.				
350	0000010522	IE	934127	00006278					ABERNETHY T.W. & SONS LTD.				
350	0000010523	IE	986210	54051061					ABLE MANAGEMENT LTD				
350	0000010524	IE	932345	15658063					ABSOLUTE HANDLING SYSTEMS.				

Figure 8.4: Modal showing data contained in a table, which is being integrated

8.4 Conclusion

Fully automatic schema matching is only possible if best-effort matching is satisfactory. As may be expected, this is not the case with Spendlab Recovery. Therefore, the users (data specialists) have to examine the mapping. Because the users are domain experts rather than computer scientists, a simple graphical user interface is required. We review existing visualizations techniques for schema matching. The most encountered technique is having the source schema on the left, the target schema on the right, which both are represented as tree views. The mapping is shown as a network of links connecting elements from the source and target schema. Therefore, we chose this technique for the graphical user interface of the proposed integration system. However, one drawback is that if the schema or mapping becomes more substantial, then the details of interest are lost. In the future, this drawback can be countered by implementing (one of) the following techniques: [18]

- Highlight propagation
- Auto-scrolling
- Coalescing Trees
- Multi-select
- Incremental Search
- Bendable Links
- Focus on Linked Elements

Chapter 9

Evaluation

In the previous chapters, the proposed integration system is described including the reason for creating the proposed integration system (chapter 1), the requirements (chapter 2), related work (chapter 3), the system architecture (chapter 4), and the components (chapters 5, 6, 7, 8). This brings us to the last research question: "*To what extent does the created integration system ease up, or fully automate the integration process?*". In this chapter, we answer that question. Section 9.1 describes the motivation and goal of the evaluation. In section 9.2, we describe the set up of the evaluation. After which in section 9.3 we show the results of the semantic mapper evaluation. We conclude the chapter with the user evaluation described in section 9.4.

9.1 Motivation

In order to test if we could ease up, standardize, or fully automate the integration process, we created a fully functioning prototype. This prototype consists of separate components. The separate components not only can be evaluated individually but also as a whole. The main question is, "*are we able to ease up or fully automate the integration process*?" We answer this question through a user test, making use of the fully-functioning prototype.

Furthermore, the schema matcher is an important component of the proposed integration system. The goal of the schema matcher is to ease up creating semantic mappings. Therefore, the quality of the proposed semantic mappings influences the experience of the users. As a result, we decided to evaluate the schema matcher individually. We use a score to assess the schema matcher algorithms. The score makes it possible to reason about the different semantic mapping algorithms. Not only indicates this score, how well the implemented algorithms generate the semantic mappings, but it can also be used to indicate how changes or complete new semantic mappings algorithms perform.

9.2 Setup

Section 9.1 shows that not only do we evaluate the system as a whole, but also the schema matcher component individually. We will first describe the used datasets. Secondly, we explain how we assess the schema matcher algorithms. After this, we will describe how we evaluate the proposed integration system.

9.2.1 Datasets

The datasets used for the evaluation are four real-world datasets and are financial systems that are from organizations, which were audited in the past. More specifically, the financial systems are all SAP and Exact.

9.2.2 Schema matcher setup

In order to indicate the performance of the semantic mappers, we run experiments. For the experiments, we use the datasets described in section 9.2.1. For the evaluation, we first create a ground-truth. As ground-truth, we use two semantic mappings, which relate SAP and Exact to the data warehouse. These semantic mappings were created by hand and verified with the data specialist. The ground-truths is the ideal result of the semantic mapper algorithms.

To determine the accuracy of a semantic mapper algorithm, we use the F-measure. The F-measure uses precision and recall, which are well known and commonly used [11] [32] [41]. For the evaluation, we determine two scores. The scores are based on precision, recall, and F-measure. The first score indicates the accuracy of the semantic mapping algorithm. Correct column matches determine this score. The second score indicates the ability of the semantic mapping algorithm to identify table matches correctly. We introduce the second score because even if the column matches within a table match are incorrect, having correct table matches simplifies the process of creating a semantic mapping. We compare the result of the semantic mapping algorithms with the ground truths. This comparison gives us the false negatives **FN**, the true positives **TP**, and the false positives **FP**. The precision, recall, and F-measure are then determined as follows:

$Precision = \frac{|\mathbf{TP}|}{|\mathbf{TP}|+|\mathbf{FP}|}$ is the fraction of relevant correspondences among all the found ones

$Recall = \frac{|\mathbf{TP}|}{|\mathbf{TP}|+|\mathbf{FN}|}$ is the fraction of relevant correspondences that were generated.

F-measure $= 2 \cdot \frac{Precision*Recall}{Precision+Recall}$ harmonic mean of precision and recall.

9.2.3 User test setup

The user test is designed to measure the effect of the proposed integration system on the integration process. More specifically, does the proposed integration system ease up the integration process. As well as, what are the shortcomings of the proposed integration system, and how can we improve the proposed integration system? The user test lets the users integrate four financial systems (section 9.2.1). The integration test consists of the following steps:

- Create a new integration project.
- Add a new or existing financial system.
- Start the integration process for the financial system.
- Add data-sources to the specific financial system.
- Depending on the fact that the financial system is new or existing, the user generates a new mapping or reuses an existing one.
- Augment the semantic mapping.
  - Add/remove mappings.
  - Add/remove foreign keys.
- Execute the mapping (transfer data into the data warehouse).
- Create the analysis file.

We introduced some errors in order to check whether the user notices them. For example, during one of the SAP integration's, a mapping has to be added (which is purposely missed by the algorithm). During the EXACT integration, a foreign key has to be added to join the right creditor number and account number together. Furthermore, we chose two SAP systems, which are the same version. Therefore, if the user completes integrating one of the SAP systems, then the semantic mapping can be reused when integrating the other SAP system. We choose to use two SAP systems of the same version to test the reusability/standardization of the proposed integration system.

After each test, first of all, the user will be asked to score the proposed integration system. For the scoring, we use a questionnaire based on a 7-point Liker scale 1=Low and 7=High. The users we asked can be classified into three groups. These groups are:

- Domain experts. Domain experts are the users currently performing the integration of financial systems.
- Semi-domain expert. Semi-domain experts are users that know the domain but are not performing the integration process of financial systems.
- No domain expert. No domain experts are users that do not know much about the domain and are not performing the integration process of financial systems.

Secondly, we interview the domain experts to check that the proposed integration system complies with the system and user requirements (chapter 2).

9.3 Semantic mapper evaluation

9.3.1 Schema-based

		Column Score			**Table Score**		
Test set	Threshold	Precision	Recall	**F-measure**	Precision	Recall	**F-measure**
SAP_1	0.9	0	0	**0**	1	0	**0**
SAP_1	0.8	0	0	**0**	0	0	**0**
SAP_1	0.7	0	0	**0**	0	0	**0**
SAP_1	0.6	0	0	**0**	0	0	**0**
EXACT	0.9	0	0	**0**	0	0	**0**
EXACT	0	0	0	**0**	0	0	**0**
EXACT	0.7	0	0	**0**	0	0	**0**
EXACT	0.6	0.5714	0.1052	**0.1777**	0.0976	1	**0.1778**

Table 9.1: F-measure scores of the schema based matcher, without the domain specific thesaurus

Table 9.1 shows the results of the schema-based algorithm that does not use the domain-specific thesaurus. As can be seen, accuracy is extremely bad. The precision and recall are 0 or near 0. Meaning that when using higher thresholds, the algorithm is unable to make any column and as a result table match. However, when we use very low thresholds, the schema-based algorithm makes a lot of matches. These matches are mostly false positives. The false positives explain the recall score of 0.1052 combined with the precision score of 0.5714 for the EXACT dataset using threshold 0.6. And therefore the poor F-measure of 0.1777. These extremely poor scores are due to the table names and column names, which are present in SAP and EXACT systems, being abbreviations. The abbreviations can not be matched with the table and column names present in the data warehouse unless the threshold is very low. However, using the very low thresholds will results in a lot of wrong column matches and, as a result, table matches. We counter the disadvantage of having abbreviations as column and table names by adding a domain-specific thesaurus (chapter 6).

		Column Score			Table Score		
Test set	Threshold	Precision	Recall	**F-measure**	Precision	Recall	**F-measure**
SAP_1	0.9	0.9772	0.9772	**0.9772**	1	1	**1**
SAP_1	0.8	0.7333	1	**0.8461**	0.5882	1	**0.7404**
SAP_1	0.7	0.3437	1	**0.5116**	0.3571	1	**0.5263**
SAP_1	0.6	0.1872	1	**0.3154**	0.1960	1	**0.3278**
SAP_2	0.9	1	0.9767	**0.9882**	1	1	**1**
SAP_2	0.8	0.7288	1	**0.8431**	0.5882	1	**0.7404**
SAP_2	0.7	0.3385	1	**0.5068**	0.3571	1	**0.5263**
SAP_2	0.6	0.1706	1	**0.2915**	0.1923	1	**0.3225**
SAP_3	0.9	1	0.9268	**0.9629**	1	0.8888	**0.9411**
SAP_3	0.8	0.7222	0.9512	**0.8210**	0.5333	0.8888	**0.6666**
SAP_3	0.7	0.3333	0.9512	**0.4936**	0.3333	0.8888	**0.4848**
SAP_3	0.6	0.1659	1	**0.2847**	0.1836	1	**0.3103**
EXACT	0.9	0.9142	0.9696	**0.9411**	1	0.8888	**0.9411**
EXACT	0.8	0.5535	0.9687	**0.7045**	0.6153	0.8888	**0.7272**
EXACT	0.7	0.2038	0.9696	**0.3368**	0.5	0.8888	**0.64**
EXACT	0.6	0.0662	1	**0.1242**	0.2	1	**0.3333**

Table 9.2: F-measure scores of the schema based matcher, with the domain specific thesaurus

Table 9.2 shows the results of the schema-based algorithm that uses the domain-specific thesaurus. The use of the domain-specific thesaurus results in much better scores. We even reach accuracy (f-measure) scores around 0.97%. Interestingly, the same drawback of lowering the threshold also applies. After all, incorrect matchings (false positives) become more frequent, while recall stays the same. Hence, the precision drops due to the increased number of false-positives. In other words, the system identifies the correct known matches, even if the threshold, becomes smaller. However, by decreasing the threshold, we generate more matches. Most of the newly generated matches are false positive.

Even though the schema-based algorithm that uses the domain-specific thesaurus improves significantly on the schema-based algorithm that does not use the domain-specific thesaurus, the schema-based method has a drawback. That is, if financial systems are encountered, which contain abbreviations and these abbreviations are not represented in the thesaurus, then the schema-based matching algorithm performs poorly. The poor quality of the generated semantic-mappings will result in the following:

1. Increased effort to create the semantic-mapping for the financial system.
2. Having to add the abbreviations to the thesaurus.

Still, once the semantic mappings have been made, they can be re-used. Reusing semantic mappings spares time and effort compared to the current situation. Moreover, if the thesaurus has been filled with translations for the abbreviations, then under the condition that the translations of the abbreviations do not change significantly, a different version of the same financial system, will generate a more meaningful semantic mapping.

9.3.2 Instance-based

In order to counter the drawback of abbreviations that are present on the schema level, we implemented an instance-based algorithm (chapter 6). Table 9.3 shows the results of the implemented instance-based algorithm. As can be seen, the accuracy (F-measure) for the column scores is

	Column Score			Table Score		
Test set	Precision	Recall	**F-measure**	Precision	Recall	**F-measure**
SAP_1	0.2364	0.8139	**0.3664**	0.8333	1	**0.9090**
SAP_2	0.2905	0.8095	**0.4276**	0.9090	1	**0.9523**
SAP_3	0.1939	0.7804	**0.3106**	0.6428	1	**0.7826**
EXACT	0.1560	0.6666	**0.2528**	0.7272	0.8888	**0.8**

Table 9.3: F-measure scores of the instance based matcher

	Column Score			Table Score		
Test set	Precision	Recall	**F-measure**	Precision	Recall	**F-measure**
SAP_1	0.4615	0.4186	**0.4390**	0.8333	1	**0.9090**
SAP_2	0.4545	0.4651	**0.4597**	0.9090	1	**0.9523**
SAP_3	0.3589	0.3255	**0.3414**	0.6428	1	**0.7826**
EXACT	0.2631	0.3030	**0.2816**	0.7	0.7777	**0.7368**

Table 9.4: F-measure scores of the instance based matcher that only matches the first encountered column

significantly lower than the accuracy (F-measure) for the table scores. Meaning that the system correctly identifies table matches, based on the domain types of the columns, as explained in chapter 6. However, within these table matches, there exist false positives that wrongfully match columns. Interestingly, if we look at the recall scores for the column matches, then we see that the scores are fairly high/average. This means that the instance-based matcher correctly identifies most of the known correct matches, but there is a lot of noise in the form of false positives (incorrect matches). The algorithm introduces noise because it creates a match between every column of the same type within the table match. For illustration purposes, take, for example, table **A** consisting of two domain types, namely IdentyfingCode and IdentyfingNumber, and table **B** consisting of IdentyfingCode, IdentyfingNumber(1), IdentyfingNumber(2) and Amount. The algorithm will create matches between **A**.IdentyfingCode and **B**.IdentyFindCode, **A**.IdentyfingNumber and **B**.IdentyfingNumber(1), but also **A**.IdentyfingNumber and **B**.IdentyfingNumber(2) introducing one false positive. Moreover, for each column in table **A** that matches multiple columns in **B**, additional false positives are introduced. Therefore, larger tables introduce more false positives.

We explored the possibility of only matching the first encountered column of the same type. Hence, reducing the number of false positives. Meaning that for every column in **A**; we loop over the columns in **B**. If a column is of the same domain type, then we create a match and remove the matched column from **B**. Ensuring that not every column from **A** of the same type, matches the identical column in **B**. The result of this method is shown in table 9.4. As can be seen, the overall F-measure of the column score improves. This improvement is due to the fact that the precision increases because fewer false positives are generated. However, as shown, the recall also decreases because we pick the first encountered column of the same type. Therefore, not knowing if it is the correct column to match. As a result, the amount of true positives is decreased, and hence the decrease in recall. Although the precision increases, the accuracy (F-measure) does not increase significantly compared to matching all columns (table 9.3) as a result of the decreased recall.

Each of the methods has advantages and disadvantages. By matching all columns, a higher recall is achieved, thus identifying the necessary matches. However, there are more false positives, whereas, matching the first encountered column has less false positives, but scores lower on identifying the necessary matches. This can be translated to: "*is it easier for the data specialist to remove incorrect matches than to add new correct matches?*". Expert opinion within NIC Finance projects denotes that through the created GUI (see chapter 8), removing incorrect matches is easier. Therefore, we choose to use the first method for matching columns with the instance-based algorithm. In the future, more research can be conducted to improve the precision of the column matching within a pair of matching tables.

9.4 User test

Question	Domain expert	No domain expert	Semi-domain expert	Semi-domain expert	Total
How efficient is the system?	6	7	5	6	24/28
Overall ease of use?	5	6	6	5	22/28
How well is the integration process facilitated by the system?	7	7	7	7	28/28
How many time is saved by the system?	5	7	7	7	26/28
How easy does the integration of multiple financial systems become?	5	5	7	7	24/28
How reusable is the system?	7	6	6	7	26/28
Does the system provide uniformity?	6	7	6	6	25/28
Did we introduce a standard?	6	6	6	6	24/28
Will the quality of the integration improve, because of the system?	6	6	5	7	24/28

Table 9.5: User ratings that are given to the system(1=Low, 7=High), higher ratings indicate higher satisfaction. The last column shows the average score

Table 9.5 shows the scores given by the users. The results show that the process of the integration of financial systems becomes significantly easier. In addition, the system is reusable, resulting in uniformity. This uniformity introduces a standard across the data specialists. Therefore, reducing time and errors. The domain expert gives the overall ease of use a score of 5/7. The biggest drawbacks of the overall ease of use are that the system does not indicate where foreign keys (in the source) are necessary. And, for larger tables, scrolling is required even though only a few columns amongst the large table relate to the data warehouse.

During the user tests, we checked if the prototype complies with the systems and user requirements from chapter 2. Each requirement can be satisfied or not satisfied. This is shown in tables 9.6 and 9.7. Meeting the system and user requirements results in satisfying or not satisfying the main features of the proposed integration system; this is shown in table 9.8.

S01 and S02 are satisfied because the system provides functionality to add physical data sources such as .txt and .csv, as well as virtual data sources. Once added or connected, the system copy's the data in a unique temporary database. S03 and S06 are satisfied because the system can generate a semantic mapping through schema-matching algorithms and store this generated semantic mapping for the financial system. S04 and S05 are satisfied because the user can open and augment the semantic mapping for the financial system through a GUI. U01 and U02 are satisfied through the GUI the user can create a new integration project to which different financial systems can be added. These financial systems can either be new or existing ones. Furthermore, U03 and U04 are satisfied; the user can open the financial system through the GUI and add data sources to the financial system. U05 and U06 are satisfied because in the same view as U03, the semantic mapping is visualized, and there are buttons that provide the functionality to generate a new semantic mapping, as well as functionality to augment the mapping. U07 and U08 are satisfied because we can execute the mapping, which finalizes it and also exchange(s) the data from the temporary storage into the data warehouse. This exchange is performed through a query generated by the query generation algorithm from chapter 7. Furthermore, U09 is satisfied because we provide an API. This API can be used by the users to query the data stored in the data warehouse. Finally, through a button, the user can execute a query. This query results in the analysis file. Therefore, satisfying the requirement U010.

In conclusion, the prototype successfully implements 15 of the 17 requirements. The requirements that are not satisfied have the priority should. From this, we can conclude that the main features of the data integration system are satisfied.

Index	Title	Description	Owner	Priority	Status
S01	Connect to data sources	As system I want to provide a interface to connect to different data sources	Student	Must	Satisfied
S02	Extract data	As system I want to be able to extract data from the different data sources	Student	Must	Satisfied

Table 9.6: System requirements

System requirements (continued)

Index	Title	Description	Owner	Priority	Status
S03	Save mapping	As system I want to be able to save the mapping from the data sources to the data warehouse	Student	Must	Satisfied
S04	Retrieve saved mappings	As system I want to be able to retrieve saved mappings	Student	Must	Satisfied
S04	Retrieve saved mappings	As system I want to be able to retrieve saved mappings	Student	Must	Satisfied
S05	View mappings	As system I want to be able to show a overview of the mapping	Student	Must	Satisfied
S06	Generate schema mapping	As system given a source schema **S** and the mediated schema **T** I want to generate a mapping	Student	Should	Satisfied
S07	Extract schema	As system given a data source **D** I want to be able to extract a schema **S**	Student	Should	Not satisfied

Index	Title	Description	Owner	Priority	Satisfied
U01	Add Project	As data specialist I want to be able to add a project in order to start the integration process	Student	Must	Satisfied

Table 9.7: User requirements

User requirements (continued)

Index	Title	Description	Owner	Priority	Satisfied
U02	Add financial system	As data specialist I want to be able to add a system to the project	Student	Must	Satisfied
U03	Open data system	As a data specialist I want to be able to open the data system in order to integrate	Student	Must	Satisfied
U04	Add sources to financial system	As a data specialist I want to be able to add data sources to the data system	Student	Must	Satisfied
U05	Overview Mapping	As data specialist I want to have a overview of the suggested mapping from schema **S** to mediated schema **T**	Student	Must	Satisfied
U06	Augment generated mapping	As data specialist I want to be able to augment the generated mapping from schema **S** to mediated schema **T** in order to be able to make changes to the mapping	Student	Must	Satisfied
U07	Finalize mapping	As data specialist I want to be able to finalize the mapping from schema **S** to mediated schema **T**	Student	Must	Satisfied
U08	Integrate data	As data specialist I want to be able to integrate the data from the data sources to the data warehouse (mediated schema)	Student	Must	Satisfied

User requirements (continued)

Index	Title	Description	Owner	Priority	Satisfied
U09	Query data	As data specialist I want to be able to access the data warehouse through API	Student	Must	Satisfied
U010	Generate .84 file	As data specialist I want to be able to generate the .84 file from the data warehouse using the API	Student	Must	Satisfied
U011	Dashboard	As data specialist I want to analyze the resulting data from the integration process using a dashboard	Student	Should	Not Satisfied

Index	Title	Description	Status
F01	Mediated schema	DISAP provides datawarehouse in which the financial data from the different data sources can be stored	Satisfied
F02	Clean data	DISAP provides functionality to transform the data from the data sources	Satisfied
F03	Integrate data	DISAP provides functionality which integrates data from the data sources into the data warehouse	Satisfied
F04	Access data warehouse	DISAP provides API through which (integrated) data from the data warehouse can be accessed	Satisfied

Table 9.8: Main features of DISAP

9.5 Conclusion

The main research question is, "*Are we able to ease up or fully automate the integration process implemented by NIC Finance projects?*" To answer this research question, we created a fully functioning prototype, which can be tested as a whole or individually. In order to answer the main research question, we set up a user test. During the user test, the users use the proposed integration system (as a whole) to integrate real-life datasets. These datasets are SAP and Exact systems that originate from organizations that were audited in the past.

Furthermore, the semantic mapping component influences the user experience. That is, higher quality proposed semantic mappings reduce the manual effort associated with finalizing semantic mapping. Therefore, easing up the user experience. In order to reason about the quality of the proposed semantic mappings, we introduce two accuracy scores. These scores are based on F-measure, Recall, and Precision.

The user test showed that the integration process becomes significantly easier. In addition, by reusing semantic mappings, time is saved, and uniformity across the users (data specialists) is increased. The semantic mapper evaluation showed us that the schema-based algorithm that uses the domain-specific thesaurus reaches high accuracy scores. However, the schema-based matcher has one drawback that is: If the financial systems contain abbreviations (that are not present in the domain-specific dictionary), for its column and table names, then matchings can not be made. Therefore, we implemented the instance-based algorithm. This algorithm reaches high accuracy for the table matches. As well as identifying most column matches. However, when identifying column matches, lots of false positives are introduced. As a result, less then average accuracy was achieved for the column matches. Nonetheless, by identifying correct table matches, creating the semantic mappings becomes easier. Furthermore, because we identify most of the necessary column matches. The user only has to remove incorrect matches (false positives).

In the future new semantic mappings, algorithms can be added, or the instance-based matcher can be improved by making the column matching generate less noise. As well as visualization tricks (chapter 8) can be applied to improve the user experience, for example, indicating where, in the source data, foreign keys are missing.

Chapter 10

Conclusion

10.1 Conclusion

The project aimed to explore whether "*We could design an integration system that eases up, standardize or fully automate the integration process implemented by Spendlab Recovery*"? Based on the study, three mentioned issues arose, these are:

- The integration process is not standardized.
- Relations in the data, existing in the financial systems, are discarded.
- There is a lack of quality control regarding the integrated data throughout the audit.

To resolve these issues, we proposed a data integration system, that:

- Supports integrating different data sources.
- Makes integrating new data sources easier through an automated mapper.
- Automates integrating existing data sources through saved mappings.
- Overview of the data after the integration to notice errors more easily.
- Remains close to the raw data of the customer, meaning that the relations in the data remain intact.

We proposed a data integration system that uses the schema-first approach. The schema-first approach queries a mediator schema (data warehouse), to which multiple data sources are mapped. The proposed integration system uses a data warehouse to provide central storage. The underlying model of the data warehouse is based on the commonality between encountered financial systems. Furthermore, this model adheres to the wish that relations in the data of the organization should remain intact. The proposed system consists of four main components. These components are:

- Data warehouse
- Semantic mapping
- Data exchange
- Graphical user interface

Because the proposed integration system is based around a data warehouse, each financial system has to be mapped. Creating these mappings can be tedious. Therefore, we aim to ease up this process by proposing semantic mappings. For this, we chose to implement a combined matcher setup. We decided to implement the combined matcher because of its flexibility and extensibility. Two algorithms are implemented, namely a schema-based algorithm and an instance-based algorithm. The first algorithm we created is schema-based; this means that the matcher only operates on table and column names. This matcher reached poor results because almost all financial systems consist of abbreviations. We countered this by adding a domain-specific thesaurus to the matching algorithm resulting in high accuracy scores. However, the schema-based matcher has one drawback that is: If the financial systems contain abbreviations (that are not present in the domain-specific dictionary), for its column and table names, then the schema-based matching algorithm performs poorly. Therefore, we also created an instance-based matcher. This matcher operated solely with the data stored, and through data mining and information retrieval techniques, classifies the columns. The classified columns are then used to classify the table. The classification of the tables and columns is then used to create the resulting semantic mapping. The instance-based matcher, reached fairly good results, especially on matching the tables correctly. Therefore, easing up the process of creating semantic mappings significantly.

Given semantic mappings between financial systems and the data warehouse, queries can be written that transfer the data from the financial systems into the data warehouse. However, because the users (data specialist) have no profound knowledge of SQL or other query languages, this becomes a time consuming, tedious, and error-prone process. Therefore, we implemented a query generator that given a financial system, the data warehouse, and the corresponding semantic mapping generates a query. This query can be executed to transfer the data from the financial system into the data warehouse. Because, CLIO was the only project encountered, during the literature review, that provides data exchange. We based the implementation of the data exchange component on the CLIO project. The main difference between the CLIO approach and ours is that:

- CLIO is able to handle nested schema, whereas, in our approach, we know that we will never encounter nested schema. Because we know that all the schema and data we encounter belong to ERP / Financial systems.
- CLIO, materializes a target schema, whereas, we translate our source-to-target dependencies into an MSSQL query.

Fully automatic schema matching is only possible if best-effort matching is satisfactory. As may be expected, this is not the case with Spendlab Recovery. Therefore, the users (data specialists) have to examine the mapping. Because the users are domain experts rather than computer scientists, a simple graphical user interface is required. We review existing visualizations techniques for schema matching. The most encountered technique is having the source schema on the left, the target schema on the right, which both are represented as tree views. The mapping is shown as a network of links connecting elements from the source and target schema. Therefore, we chose this technique for the graphical user interface of the proposed integration system.

By design, the proposed system adheres to all the wishes retrieved during the study.

- The system supports integrating different financial systems.
- The integration process is made easier using semantic mapping algorithms; once a mapping is made, it can be reused. Therefore, not only saving time but also introducing standardization.
- Using a centralized data warehouse, we created a model that remains close to the data of the organizations that are audited.

Based on a 7-point liker scale, filled in during the user test. It is shown that the integration process becomes significantly easier. In addition, the system is reusable, resulting in uniformity. This uniformity introduces a standard across the data specialist that execute the integration. Therefore, reducing time and errors. This is reflected in the final score given to the system.

10.2 Future work

During the project, we came up with future work. Future work relates to the different components of the proposed integration system. First of all, in our literature review, we categorized different semantic mapping techniques. A limitation is that there is much literature on semantic mapping algorithms. Therefore, it was not as systematic as it could be. Hence, we might have missed literature. Future research can be conducted on semantic mapping algorithms. This research can be:

- Study how we can combine the results of separate individual semantic mapping algorithms.
- Study state of the art techniques, such as deep-learning and create new semantic mapping algorithms

Secondly, the graphical user component can be extended with visualizations techniques (chapter 8). As well as creating new visualization techniques to ease up creating semantic mappings.

Thirdly, the proposed integration system can be augmented with the following:

- Add data transformation functionality to the system. Making it able to specify how the data can be cleansed and transformed into various data formats.
- Augment the data exchange component, enabling it to handle nested schema. Hence, making it able to handle document-oriented databases.

Finally, the data warehouse can be extended with the following:

- Entity linkage algorithms. Because the data originates from different financial systems, the same vendor (entity) can be stored differently (within the same organization). For illustration purposes, in one financial system (of the organization), the VendorNumber is 1, and in the other financial system (of the organization), the same vendor has VendorNumber 4. By correctly identifying that these vendors are identical, we can further improve the analysis.
- Machine learning. Because our method does not denormalize the data, we can improve the risk analysis. The improvement can be achieved by adding new algorithms next to the existing ones. The main difference is that the new algorithms can run on the data warehouse, whereas the existing ones still require the denormalized analysis file.
- The data in the data warehouse can be visualized. Making it easier to notice whether the integration contains errors (chapter 2). Furthermore, making it easier to asses the data quality, being able to generate reports et cetera.

Bibliography

[1] B. Alexe, L. Chiticariu, R. J. Miller, and W. Tan. Muse: Mapping understanding and design by example. In *2008 IEEE 24th International Conference on Data Engineering*, pages 10–19, April 2008. 51

[2] David Aumueller, Hong Hai Do, Sabine Massmann, and Erhard Rahm. Schema and ontology matching with coma++. pages 906–908, 01 2005. 51

[3] Bao-hua Qiang, Kai-gui Wu, Xiao-feng Liao, and Zhong-fu Wu. Similarity determination based on data types in heterogeneous databases using neural networks. In *International Conference on Neural Networks and Signal Processing, 2003. Proceedings of the 2003*, volume 1, pages 377–380 Vol.1, Dec 2003. 20

[4] Fernando Belfo and António Trigo. Accounting information systems: Tradition and future directions. *Procedia Technology*, 9:536 – 546, 2013. CENTERIS 2013 - Conference on ENTERprise Information Systems / ProjMAN 2013 - International Conference on Project MANagement/ HCIST 2013 - International Conference on Health and Social Care Information Systems and Technologies. 27, 28

[5] Philip A. Bernstein, Jayant Madhavan, and Erhard Rahm. Generic schema matching, ten years later. *PVLDB*, 4:695–701, 2011. 18, 49

[6] Angela Bonifati, Elaine Chang, Terence Ho, Laks Lakshmanan, and Rachel Pottinger. Heptox: Marrying xml and heterogeneity in your p2p databases. volume 3, pages 1267–1270, 01 2005. 18

[7] Sjaak Brinkkemper. Method engineering: engineering of information systems development methods and tools. *Information and Software Technology*, 38(4):275 – 280, 1996. Method Engineering and Meta-Modelling. 6

[8] Niladri Chatterjee and Madhav Krishna. Semantic integration of heterogeneous databases on the web. pages 325–329, 03 2007. 20

[9] Cecil Chua, Roger Chiang, and Ee-Peng Lim. Instance-based attribute identification in database integration. *VLDB J.*, 12:228–243, 10 2003. 35

[10] William W. Cohen. Integration of heterogeneous databases without common domains using queries based on textual similarity. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, SIGMOD '98, pages 201–212, New York, NY, USA, 1998. ACM. 33

[11] Hong-Hai Do and Erhard Rahm. Matching large schemas: Approaches and evaluation. *Information Systems*, 32(6):857 – 885, 2007. 18, 19, 56

[12] AnHai Doan, Pedro Domingos, and Alon Levy. Learning source description for data integration. pages 81–86, 01 2000. 20

[13] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 19(1):1–16, Jan 2007. 32, 33

[14] Joerg Evermann. An exploratory study of database integration processes. *IEEE Trans. on Knowl. and Data Eng.*, 20(1):99–115, January 2008. 18, 20

[15] Ronald Fagin, Laura Haas, Mauricio Hernández, Renée Miller, Lucian Popa, and Yannis Velegrakis. Clio: Schema mapping creation and data exchange. pages 198–236, 01 2009. 18, 20, 21, 41, 42, 44, 47

[16] Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa. Data exchange: semantics and query answering. *Theoretical Computer Science*, 336(1):89 – 124, 2005. Database Theory. 20

[17] Sean M. Falconer and Natalya Fridman Noy. Interactive techniques to support ontology matching. In *Schema Matching and Mapping*, 2011. 49, 51

[18] George G. Robertson, Mary Czerwinski, and John Churchill. Visualization of mappings between schemas. pages 431–439, 01 2005. 51, 53

[19] Alon Halevy, Zachary Ives, Jayant Madhavan, Peter Mork, Dan Suciu, and Igor Tatarinov. The piazza peer data management system. *Knowledge and Data Engineering, IEEE Transactions on*, 16:787 – 798, 08 2004. 18

[20] Paul Jaccard. The distribution of the flora in the alpine zone.1. *New Phytologist*, 11(2):37–50, 1912. 33

[21] F. Robert Jacobs and F.C. 'Ted' Weston. Enterprise resource planning (erp)—a brief history. *Journal of Operations Management*, 25(2):357 – 363, 2007. Special Issue Evolution of the Field of Operations Management SI/ Special Issue Organisation Theory and Supply Chain Management. 27

[22] Matthew A. Jaro. Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida. *Journal of the American Statistical Association*, 84(406):414–420, 1989. 32

[23] Nishtha Jatana, Sahil Puri, Mehak Ahuja, Ishita Kathuria, and Dishant Gosain. A survey and comparison of relational and non-relational database. 25

[24] Ishwarappa Kalbandi and J Anuradha. A brief introduction on big data 5vs characteristics and hadoop technology. *Procedia Computer Science*, 48:319–324, 12 2015. 8

[25] Helmut Klaus, Michael Rosemann, and Guy G. Gable. What is erp? *Information Systems Frontiers*, 2(2):141–162, Aug 2000. 27

[26] C Lau and Arthur Ryman. Developing xml web services with websphere studio application developer. *IBM Systems Journal*, 41:178 – 197, 02 2002. 51

[27] Maurizio Lenzerini. Data integration: A theoretical perspective. pages 233–246, 01 2002. 18

[28] Maurizio Lenzerini. Principles of p2p data integration. pages 7–21, 01 2004. 18

[29] V. I. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10:707, February 1966. 32

[30] Wen-Syan Li, Chris Clifton, and Shu-Yao Liu. Database integration using neural networks: Implementation and experiences. *Knowl. Inf. Syst.*, 2:73–96, 03 2000. 20

[31] You Li, Dong-Bo Liu, and Wei-Ming Zhang. Schema matching using neural network. pages 743 – 746, 10 2005. 20

[32] Ahmed Mahdi and Sabrina Tiun. Utilizing wordnet and regular expressions for instance-based schema matching. *Research Journal of Applied Sciences, Engineering and Technology*, 8, 07 2014. 32, 37, 56

[33] Alvaro E. Monge and Charles P. Elkan. The field matching problem: Algorithms and applications. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 267–270. AAAI Press, 1996. 33

[34] Erwan Moreau, François Yvon, and Olivier Cappé. Robust similarity measures for named entities matching. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 593–600, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. 32

[35] Lucian Popa and Val Tannen. An equational chase for path-conjunctive queries, constraints, views. 12 2000. 44

[36] Lucian Popa, Yannis Velegrakis, Renée J. Miller, Mauricio A. Hernández, and Ronald Fagin. Translating web data. In *VLDB*, 2002. 17

[37] Erhard Rahm and Philip Bernstein. On matching schemas automatically. *Report Nr*, 1, 03 2001. 18, 19, 20, 31, 32

[38] Erhard Rahm and Philip Bernstein. A survey of approaches to automatic schema matching. *VLDB J.*, 10:334–350, 12 2001. 31

[39] Valerie Sessions and Marco Valtorta. The effects of data quality on machine learning algorithms. pages 485–498, 01 2006. 17

[40] Edhy Sutanta, Retantyo Wardoyo, Khabib Mustofa, and Edi Winarko. Survey: Models and prototypes of schema matching. *International Journal of Electrical Computer Engineering*, 6, 06 2016. 18, 19, 20

[41] M. Szymczak, A. Bronselaer, S. Zadrożny, and G. De Tré. Semantical mapping of attribute values for data integration. In *2014 IEEE Conference on Norbert Wiener in the 21st Century (21CW)*, pages 1–8, June 2014. 17, 56

[42] Marcos Antonio Vaz Salles, Jens-Peter Dittrich, Shant Karakashian, Olivier Girard, and Lukas Blunschi. itrails: Pay-as-you-go information integration in dataspaces. pages 663–674, 01 2007. 18

Appendix A

Encountered (Financial) Systems

Table A.1: Encountered (financial) systems by SpendLab Recovery

(Financial) Systems
FIS
SAP
Baan
Exact Globe
JDEdwards
Oracle
PeopleSoft
Fin+
Decade
Triton
MFG Pro
Crédit Réseau
Metacom
FMS
ACE
VILA
Credit
FINAD
Gisfa
CMD
Eagle
ROSS
AS 400
Bomas
AccountView
Navision

Encountered (financial) systems by SpendLab Recovery (continued)

(Financial) Systems
Acto unix
Hannibal (Centric)
Elite
JBA (Pink Rocade)
Fis for all van centric
Final
Davice
Auto-line
GFS
Coda
SCBA
AXI
Omnivers
infosoft van Centric
Kraan
Globit FBS
concorde - darmgard
Smart
Oliepakket (maatwerk)
Lodder
Damast
Darts
BPCS (ERP Systeem)
DBS Financieel
All Solutions
CAFAS
Atlas
Scala
Civision(sap voor gemeenten)
MultiRecord
Fis4All
DBS Financial
Xfis
IBS Financial
Nixdorf
MOVEX
AFAS 2003
Axapta
FIBU
RESY
Credplains
TSN
CubeWare
Grote Beer
Cedar
Improve
IPS
Multivers

Encountered (financial) systems by SpendLab Recovery (continued)

(Financial) Systems
ACTO
KING
Megacar
SG Tobias
Rimses / Remax
Flashware
Comtabel
KEA
20/20 vision
UNIT4/ REMS
MS Dynamics
Twinfield
Aremis
Extendas
KEY2FINANCIËN
AFAS
First Housing
IFS
CODA
CSB
Remmicom
CIPAL
DAX
KPD Bouwoffice
BBC (Schaubroeck)
Onbekend
Bis.Net
XPAND
fibru
Maconomy
ATAK
FinAcc
Finances
Interclean
Iventive
M Soft
My Project
Visma
Glovia
legal
Infohos

Encountered (financial) systems by SpendLab Recovery (continued)

(Financial) Systems
Agresso
Civision middelen
Empire
BaaN
LN
XAL
Cura
Rissis
ARCO Scanning
NCCW
ISGRAM
LISA
Uitzendmaster
Mamoet
Acto
Noco
Syntess
Pluriform (bouwworks)
Viewpoint
M3
POP
Roadmaster
Exact Online
Exact Financial
Profin

Appendix B

Conducted Interviews

B.1 Interview data specialist 1

Table B.1: Interview data specialist 1

Question	Answer	Annotation
What is the current workload?	Were a bit behind, each year we aim to do 150 integration's, up until now we have done 30	
What is the currently used process to integrate the data?	First there is a kickoff, then we retrieve the data. After which, we judge the correctness of the data and we start with the OPL	Things can go wrong, such as we do not get access to the data at the customer. Their IT specialist is not present etc.
How long does this process take in the best case?	After we retrieved the data, most of the time 2-3 weeks,	
How long does this process take in the worst case?	4+ weeks	
How long does this process take globally?	Unable to answer	
What are the pitfalls encountered during this process?	Many different business units, different databases (systems). How bigger the amount of data, ACCES becomes less reliable.	
Does it often happen that the data integration goes wrong?	Yes, there can be human errors, interpretation of the data can differ, data extraction errors, wrong joins, key fields can be determined differently	Data quality differs alot
What are the reasons it goes wrong?	See above	

Interview data specialist 1 (continued)

Question	Answer	Annotation
What happens if the data integration goes wrong? Do you guys start over or?	That depends when it is noticed and how the integration went wrong, most of the time we only have to redo a sub step.	
If it goes wrong when is this noticed? (late in the process or early)	That depends sometimes it is not noticed at all. Most of the time we notice errors when the data goes through APRA.	
How standardized is this process? i.e Does it happen that two data specialist for the same data source(s) use different methods/steps?	Yes this can happen, the output must be the same. However, we do not have a procedure also ACCES can be used differently by data specialists	Uniformity is a MUST
Is there any control on this or any idea what the differences are?	No, we only talk about it	
Is there a huge difference in the quality of integration between the data specialists?	Yes	
How many times do the steps differ for the same system. For example two customers with a SAP or EXACT system?		
How many times do the steps differ for the same customer. For example customer KPN in 2018 and customer KPN in 2019?	Every integration projects starts anew	
Did you collect meta data over the past projects?	No	
If there is no mediated schema how do you currently handle two different data sources for the same customer?	We integrate them seperatly and then join them together at the 51 table	
What are the steps, when the data is in ACCESS?	We join the 50 table(s) together into the 51 table, then we execute queries which create the 84 table. We export the 84 table resulting in the analysis file	At the 50 table the data is already denormalized (we choose 1 accountnumber out of N)
Are these steps different for each data source?	No, we always execute the queries	
Can these steps go wrong?	Yes, queries can be forgotten or queries can be executed multiply times. Or the wrong ones can be executed	

Interview data specialist 1 (continued)

Question	Answer	Annotation
Do data transformations happen for example when there are two different date formats within a column “2019-09-01” and “2019/09/01” is “2019/09/01” than translated to “2019-09-01” and vice versa?	Yes, it always goes to the same format	For example if we use the American date notation instead of the Dutch one, the analyses can go wrong.
What happens with data uncertainty for example null values?	They remain null values	
How do you want the tool to support you?	Standards/Uniformity should be introduced. We no longer have to use MS access. One a integration is done, we should be able to reuse it. Because, the steps are always the same	Non technical users should be able to use the integration tool
Where do you think most performance can be gained?	During integration	The steps up until the 51 table are the most work. After that it is just executing queries to create the 84 table
If the data is in the data warehouse do you have any wishes? For example dashboards etc.	Dashboard, reporting, overviews etc.	

B.2 Interview data specialist 2

Table B.2: Interview data specialist 2

Question	Answer	Annotation
What is the current workload?	7 out of 10	
How long does this process take in the best case?	A day if all data is complete and when it’s a simple system	
What is the currently used process to integrate the data?		
How long does this process take in the worst case?	More than a week	Depending on size and used system
How long does this process take globally?	32 hours	(for 1 system)
What are the pitfalls encountered during this process?	Many, incomplete data, unreliable data, wrong decision making in the process	

Interview data specialist 2 (continued)

Question	Answer	Annotation
Does it often happen that the data integration goes wrong?	sometimes	
What are the reasons it goes wrong?	Not paying attention, data incorrect, 1001 reasons	
What happens if the data integration goes wrong?	You have to be more specific what kind of errors you mean. Normally step 1 would be to correct your mistakes.	
If it goes wrong when is this noticed? (late in the process or early)	Depending on the mistake, certain errors are found immediately after making them, others are found prior to uploading it to APRA. Maybe even in the near future through the gatekeeper in APRA.	
What is the impact of the time such a error is noticed?	Simple errors can be handled quickly. Bigger errors can have huge impact on planning and costs.	
Are all errors noticed during this process? Can it be the case that a project wrongly integrated gets analyzed?	Not all errors were found during the Proces of Data. It could be the case that data is analyzed containing errors. It's standard that the data is checked by the PM at the customers vs the live erp system.	
How standardized is this process? i.e Does it happen that two data specialist for the same data source(s) use different methods/steps?	Its not standard. We are working on that for certain systems. (SQL)	
If so how much do these steps differ are there any examples?	The process steps are the same, but many things can differ in their approach.	
Is there any control on this or any idea what the differences are?	No	
Is there a huge difference in the quality of integration between the data specialists?	Sometimes there is, depending on complexity of the dataset.	

Interview data specialist 2 (continued)

Question	Answer	Annotation
How many times do the steps differ for the same customer. For example customer KPN in 2018 and customer KPN in 2019?	De dataset can always differ, the customer might have chanced its routine, changed their system, added software etc. So its hard to say, we always have a large scope , multiple years.	
Did you collect meta data over the past projects?	No	
How is the data loaded in ACCESS? A mediated schema or something else	See proces Data.	
If there is no mediated schema how do you currently handle two different data sources for the same customer?	Treat it as is.	
What are the steps, when the data is in ACCESS?	See proces Data.	
Are these steps different for each data source?	Different yet somewhat similar in approach. Not all systems have the same structure.	
Can these steps go wrong?	Yes.	
What happens if errors occur during these steps?	...	
Do data transformations happen for example when there are two different date formats within a column "2019-09-01" and "2019/09/01" is "2019/09/01" than translated to "2019-09-01" and vice versa?	Normally a data download is consistent in its output. So a data will always show the same format. But to every rule there is that 1 exception. . . .	Normally it would say "No". But I know it happened at least once.
What happens with data uncertainty for example null values?	When the value of a field is Null, we keep this as a value. If the field has a value filled in we use that value. If we cannot match a record it becomes a Null Value.	

Interview data specialist 2 (continued)

Question	Answer	Annotation
Why are the relation in the data not respected for example a creditor has multiple bank accounts. However, only one is picked during integration?	In the 51 export only 1 value can be used. otherwise you would see multiple lines per invoice created due to having more than 1 bank account. (same goes for address, phone number etc) It would be very difficult to analyze the lines visually. How to tell when is it a double created line, or when is it an actual double invoice line. Therefore, only 1 value can be used. This is also the reason we only show the amount VAT as a sum, and the amount paid as Sum. (so we throw out all the other useful info)	
How do you want the tool to support you?	Take away our repetitive unchallenging work and produce quicker results in the process. So we can do other more meaningful work.	
Where do you think most performance can be gained?	Automation of the steps after the 51 Export is created. (all steps now are in query and Macro)	

B.3 Interview SpendLab stakeholder 1

Table B.3: Interview SpendLab stakeholder 1

Question	Answer	Annotation
How many projects are done each year?	150	We aim to increase this to 200-250
How long does a project take in the best case?	In the best case 16 weeks	
How many weeks are reserved for the integration process	The target time is 4 weeks	The actually processing time (when the data is retrieved) is 1 week
How many days are reserved for the integration step	3 days	
How long does the integration step take in the current process	Around 3 weeks	

Interview SpendLab stakeholder 1 (continued)

Question	Answer	Annotation
In the best case how long does a project take?		
In the worst case how long does a project take	10-12 months	Ask Evelien
What is the target time for a average project?	The target time is 16 weeks	Reasons why this is not achieved can be internal and external. Communication errors with the organization or vendors of the organization are external. Internal errors are errors during the integration process or analysis
How many projects are within the target time?	Ask Evelien	
How many projects are outside of the target time		
Why?		
Is the data integration step, a bottleneck process?	Yes, it propagates through the audit en it makes the analysis less reliable.	
Describe the current process of data integration?	I cant describe it in detail. In mainlines data is retrieved then 50 then 51 and finally to 84.	
Are there details about the integration process you would like to know?	Yes, what is standardized and what is not?	
When do you think errors are noticed?	I hope in the gate, within the target time of the integration process. I know integration errors are also noticed after the project leader approved the integration	
Why do you think it happens at these moments	Because, the current measures are not adequate.	
Are there analysis, in which the errors are noticed after the integration process.	Yes	It depends on the project leader
Are there analysis, in which the errors are noticed after the complete audit	Yes, during the discussion with the organization about what was found.	Therefore, the analysis is not as profitable as could be
If this is the case, are there projects during which errors are not noticed at all	Yes, this has to happen because of the implemented process	

Interview SpendLab stakeholder 1 (continued)

Question	Answer	Annotation
Do you think the process is standardized	It should be	
What should he proposed system achieve?	Reduce the integration time, improve the quality, standardize the process ,provide more insight in how the data specialists operate and open up the possibility to implement new services such as clean master data	
What is the long term vision of the proposed integration system	Is should become part of the APRA platform, we even can provide it as SaaS to our partners	
Should the proposed integration system become part of the APRA platform	Yes	
Should partners be able to use the tool	Yes	

B.4 Interview SpendLab stakeholder 2

Table B.4: Interview SpendLab stakeholder 2

Question	Answer	Annotation
How many projects are done each year?	150	We aim to increase this to 200-300
How many days are reserved for the integration step	1 day	
In the best case how long does a project take	12 weeks	
In the worst case how long does a project take	52 weeks	
How many projects are within the target time?	60%	
How many projects are outside the target time?	40%	

Interview SpendLab stakeholder 2 (continued)

Question	Answer	Annotation
Why?	Large and complexity of systems at the customer (diversity of systems) the size of the project. Cooperation and time with the customer who must approve matters and / or must be coordinated with them; cooperation on the part of the supplier; the amount of complex files that require more research at the customer	
Is the data integration step, a bottleneck process?	Yes if they do not have the necessary evidence to accept the data and the data is not sufficient	
Describe the current process of data integration?	I do not know the details	
Are there things unknown about the data integration	I doubt the quality of the resulting analysis files	
Are there details about the integration process you would like to know?	Yes I would like to know about the completeness and quality of the resulting integration	
When do you think errors are noticed?	This is always too late! First time right must be the standard here. It may also be that this is not noticed at all because projects with a low result are closed without thorough checks and therefore money is not cashed or, in other words, turnover is lost	
Are there analysis, in which the errors are noticed after the integration process.	Yes	
Are there analysis, in which the errors are noticed after the complete audit	Yes this certainly happened. Then the choice is made: what is the current result on the project vs costs, as well as, time and costs to do it again	

Interview SpendLab stakeholder 2 (continued)

Question	Answer	Annotation
If this is the case, are there projects during which errors are not noticed at all	Definitely, this is because the details or data integration as a whole is not known across the organization	
Do you think the process is standardized	There should be, however currently this is not the case. Due to the fact that the data specialists are not computer scientist and it is most of the time not known how data integration's were executed. In other words every data specialists does it differently.	
What should the proposed data integration system achieve	It should standardize the process and implement uniformity across the data specialists and SpendLab. The details should become more clear, and the quality should be increased. As a result less errors should occur and time should be reduced.	

B.5 Interview SpendLab stakeholder 3

Table B.5: Interview SpendLab stakeholder 3

Question	Answer	Annotation
How many projects are done each year?	150-200	We aim to increase this to 200-300
How many days are reserved for data retrieval and data integration step	2 weeks	
How many days are reserved for the integration step	2 days	
In the best case how long does a project take	16 weeks	
In the worst case how long does a project take	32-52 weeks	
How many projects are within the target time?	40%	
How many projects are outside the target time?	60%	

Interview SpendLab stakeholder 3 (continued)

Question	Answer	Annotation
Why?	Because of project management, as well as, the size of the projects and many manual proceedings	
Is the data integration step, a bottleneck process?	Yes	
Describe the current process of data integration?	Standard tables are retrieved (downloaded), preferably in SQL. The data is then viewed and assessed as to what is available and which links must be made. What are the document types, what should you see as a payment and invoice. Then the connections are made and the tables are combined. After this is done, it is uploaded in APRA and risk-categories are generated.	
Are there things unknown about the data integration	I doubt the quality of the resulting analysis files	
Are there details about the integration process you would like to know?	Yes what happens to the data that in the current process is lost. For example since one account number is selected what happens to the other account numbers of the vendor? What is the data we use and what can we do with the not used data?	
When do you think errors are noticed?	Most of the time late in the process, hopefully when judging the resulted integration. However, it even happens during the analysis	
Are there analysis, in which the errors are noticed after the integration process.	Yes	
Are there analysis, in which the errors are noticed after the complete audit	Yes this certainly happened. Then we look what needs to be done to be able to correctly complete the audit. In other words, do we carry out an additional analysis. Or do we completely start over?	

Interview SpendLab stakeholder 3 (continued)

Question	Answer	Annotation
If this is the case, are there projects during which errors are not noticed at all	That is also possible. If a proper check has not been carried out, and if the analysts are not keen on it, then things may have been missed as a result.	
Do you think the process is standardized	I suspect that not everyone works in the same way and is just as precise. Therefore, there is a difference in quality between the data specialists	
What should the proposed data integration system achieve	Data that is lost due to denormalization should remain intact. As a result, we can improve the analysis and add new services. It should standardize the integration process	