

Kalle Karlin

**UTILIZING DATA INTEGRATION MOD-
ELS IN DATA INTEGRATIONS**
Case Study

ABSTRACT

Kalle Karlin: Utilizing data integration models in data integrations
Master of Science Thesis
Tampere University
Master of Science Degree Program in Information and Knowledge Management
May 2019

Data integration is a process that enables communication between different systems. EDI has been used for a long time and any-to-any integrations allows to transfer data from source to target destinations while transforming and mapping the data between different data models. Many data integration project fails due to not understanding the business objectives and technical requirements. Conceptual, logical and physical models are typically used in data model context, but similar approach can be utilized in data integration modeling. Data integration modeling strives to explain what functionalities are expected from a technical solution for desired results to support business processes. Data integration modeling can be a complex process and therefore, different modeling methods can be utilized to communicate with business as well as with developers.

The objective of this thesis was to identify practical solutions of leveraging data integration models in data integration solutions. This research was carried out as a deductive case study in a company that provides data integration solutions to its customers using the case company's integration platforms. At first, literature review was made to identify existing theory about the research topic. After that, empirical material was collected by interviewing people who are participating to customer projects in the case company. Thematic analysis was used to identify occurring themes and patterns from the existing theory and empirical research results.

Research results show that successful data integration solution requires understanding both the business as well as the technology. The modeling process starts by identifying business processes and business objectives. After the scope and goals are clear, technology is added to support the business. The modeling process is usually iterative since minor technical details can have big impact to the overall solution. Therefore, technical knowledge is needed to understand what the boundaries are of creating a solution. Different modeling methods, such as UML diagrams, can be used to ensure that business goals are identified correctly. Data integration models can also be utilized to give guidelines for the developers to implement a solution with required functionalities.

Keywords: Data integration, EDI, B2B integration, modeling, integration brokerage, mapping

The originality of this thesis has been checked using the Turnitin OriginalityCheck service.

TIIVISTELMÄ

Kalle Karlin: Dataintegraatiomallien hyödyntäminen dataintegraatioissa
Diplomityö
Tampereen yliopisto
Tietojohdamisen diplomi-insinöörin tutkinto-ohjelma
Toukokuu 2019

Dataintegraatio on prosessi, joka mahdollistaa kommunikaation eri järjestelmien välillä. EDI-integraatioita on käytetty jo pitkään ja any-to-any integraatio mahdollistaa datan liikuttamisen eri lähde- ja kohdesijaintien välillä muuntaen ja mapaten datan eri tietomallien välillä. Monet dataintegraatioprojektit epäonnistuvat, koska liiketoiminnan tavoitteita ja teknisiä vaatimuksia ei olla ymmärretty oikein. Konseptuaalista, loogista ja fyysistä mallia on käytetty tyypillisesti tietomallien kontekstissa, mutta samaa lähestymistapaa voidaan hyödyntää dataintegraatioiden mallinnuksessa. Dataintegraatiomalli pyrkii kuvaamaan toiminnallisuuksia, joita vaaditaan tekniseltä ratkaisulta tukemaan liiketoimintaprosesseja. Dataintegraatioiden mallinnus voi olla monimutkainen prosessi. Erilaisia mallinnusmenetelmiä voidaan hyödyntää, jotta pystytään kommunikoimaan liiketoiminnan ja ohjelmistokehittäjien kanssa.

Tämän diplomityön tavoite oli tunnistaa käytännönläheisiä ratkaisuja, miten dataintegraatiomalleja voidaan hyödyntää dataintegraatoratkaisuissa. Tämä tutkimus toteutettiin deduktiivisena tapaustutkimuksena. Tapaustutkimuksen kohde on yritys, joka tarjoaa dataintegraatoratkaisuja asiakkailleen hyödyntäen kohdeyrityksen integraatioalustoja. Tutkimusprosessi aloitettiin etsimällä tieteellistä aineistoa, jotta saatiin ymmärrys mitä olemassa oleva teoria on tunnistanut tutkimusaiheesta. Sen jälkeen empiirinen toteutus toteutettiin haastattelemalla kohdeyrityksen työntekijöitä, jotka ovat mukana asiakasprojekteissa. Teema-analyysin avulla tunnistettiin toistuvat teemat ja mallit teorian ja empiirisen aineiston välillä.

Tutkimustulokset osoittavat, että onnistunut dataintegraatiototeutus vaatii sekä liiketoiminnan ymmärrystä että teknistä osaamista. Mallinnusprosessi alkaa liiketoimintaprosessien ja -tarpeiden tunnistamisesta. Sen jälkeen teknologia lisätään tukemaan liiketoimintaa. Mallinnus on usein iteraatiivinen prosessi, koska pienet tekniset asiat saattavat vaikuttaa oleellisesti kokonaisratkaisuun. Tämän takia tarvitaan teknistä osaamista, jotta osataan tunnistaa ratkaisun tekniset rajoitteet. Erilaisia mallinnusmenetelmiä, kuten UML-kaavioita, voidaan hyödyntää, jotta pystytään varmistamaan, että liiketoiminnan tarpeet on tunnistettu oikein. Dataintegraatiomalleja voidaan myös käyttää ohjeena ohjelmistokehittäjille, jotta he tietävät miten ratkaisu toteutetaan vaadituilla toiminnallisuuksilla.

Avainsanat: Dataintegraatio, EDI, B2B-integraatio, mallinnus, integraatio-operaattori, mappaus

Tämän julkaisun alkuperäisyys on tarkastettu Turnitin OriginalityCheck –ohjelmalla.

PREFACE

This was my last project during my studies that started at 2012 in Tampere University of Technology and finally ended at 2019 in Tampere University. My interest towards the thesis topic was formed during past few years while I have worked with data integration solutions. Even though I have now finally finalized my thesis and therefore, my studies, my journey with data integrations as well as with information and knowledge management in general, continues.

I would like to thank Opentext+Liaison Technologies for the opportunity to do this thesis in the company. I really appreciate the flexibility the company has provided to me during the previous years when I have finalized my studies. Big thanks to all my colleagues and other stakeholders who have taught me a lot about the world of data integrations. Special thanks to all my architect and architect-minded colleagues, especially to my superior Janne Edelman.

I would also like to thank my thesis supervisor Samuli Pekkola who has guided me through the research process and who was very flexible with my schedule. Thanks to all people I have been involved with during my studies. Greetings especially to MJTJP – I will never forget the fun times we had during our studies.

Helsinki, 20.5.2019

Kalle Karlin

TABLE OF CONTENTS

1. INTRODUCTION	1
1.1 Research background	1
1.2 Objectives, scope and limitations	2
1.3 Research methodology	3
1.4 Research structure	6
2. DATA INTEGRATION	7
2.1 Clarifying the concept	7
2.2 Data models	9
2.3 Common message formats	14
2.4 Understanding the data within a message	17
2.5 Data transformation and mapping	21
3. DATA INTEGRATION MODELING	26
3.1 Modeling process	26
3.2 Understanding the business	28
3.3 Data flow diagram	31
3.4 Component integration model	33
3.5 Sequence diagram	35
3.6 Flowcharts	37
4. RESEARCH METHOD AND RESEARCH SETUP	40
4.1 Introducing the case company	40
4.2 Research process	41
4.3 Empirical study	42
4.4 Analyzing the research results	44
5. KEY RESULTS	46
5.1 Reasons to design and document a solution	46
5.2 The importance of business processes	48
5.3 Understanding technical boundaries and limitations	50
5.4 Understanding the data	52
5.5 Modeling process and methods	54
5.6 Common challenges	57
6. DISCUSSION	60
6.1 Modeling process	60
6.2 Modeling methods	63
7. CONCLUSIONS AND SUMMARY	66
7.1 Conclusions	66

7.2 Evaluation of the research and future study	68
REFERENCES.....	71
APPENDIX A: INTERVIEW QUESTIONNAIRE.....	77

1. INTRODUCTION

1.1 Research background

The number of integrations between different system and organizations is continuously increasing (Gartner 2017). As Kuchibhotla et al. (2009) states, data is nowadays a key element in organization's success. Organizations need to connect with their suppliers, customers, and internal business units, to remain competitive (Kuchibhotla et al. 2009). Organizations emphasize the value of relationships and create integrations to allow interactions with their business partners (Cardoso & Bussler 2011). Veselá (2017) research shows that data integrations are either required or recommended by most of the business partners, which is one of the reasons companies are obliged to have data integrations. All these different stakeholder's often store and process data in different formats, which is the reason why these different data formats needs to be integrated (Kuchibhotla et al. 2009).

According to Gartner (2015; 2016), companies are outsourcing their data integration projects and according to Gartner (2015), over one million companies utilize integration brokerages to create and maintain their integrations. Gartner (2016) has defined integration brokerage as a service provider who offers EDI, A2A, B2B, or some other data integration solutions. Integration brokerage companies are providing different technologies to create the data integrations, such as integration-platform-as-a-service (iPaaS) (Gartner 2016; Gartner 2017) and data-platform-as-a-service (dPaaS) (Gupta 2015; Opentext+Liaison 2019). Gartner (2017) forecasts that "*the integration brokerage (outsourcing) service market will reach \$1.9 billion by 2020*".

Modeling data integration solutions can be a complicated process (Al-Naeem et al. 2014). Gartner (2016) estimates that by 2020, over one third of data integration projects will fail. To prevent this happening and to ensure that data integration projects are executed successfully, different modeling methods can be utilized to capture the business needs and to understand what the solution needs to do and how it can be implemented. As Karimpour et al. (2013) mentions, architectural decisions affect to the success of projects and the sooner problems are discovered the less the project will cost.

Before any solution can be implemented, first must be defined what needs to be done. Planning a solution can be a complex process, because it requires understanding the

business processes as well as technical knowledge of how to implement a solution to support the business needs. Data integration modeling is one part of design process to achieve understanding what the solution needs to do as well as to provide information how the solution can be technically implemented. (Reeve 2013)

Because complexity level differs between projects and since there are different stakeholders involved in projects, the appropriate detail level of models varies. Existing literature about leveraging data integration models in case company's context is limited and thus, there is a need for a research that strives to understand how to utilize data integration models for designing a solution. This allows to identify best practices towards future solutions in the case company.

1.2 Objectives, scope and limitations

The primary objectives of this research are to define how to utilize data integration modeling in solution architecture in the case organization and to define methods to model data integrations. This research aims to explain how data integration solutions should be designed. The primary research question is following:

- How to leverage data integration modeling in data integration solutions?

The primary research question can be divided to following secondary research questions:

- What is data integration?
- How to model data integrations?
- What information is needed to design a data integration solution in the case company?
- What are the challenges of planning a solution?

Two first secondary research questions are answered based on existing literature. Other secondary research questions are answered based on the empirical research. After the secondary questions are answered, there should be an answer to the primary research question.

The scope of the research was discussed and agreed with the case company. Data integration is a commonly used term in many different contexts, such as data warehousing and databases, but this research focuses to data integration solutions provided by the case company to its customers. Because the case company has multiple platforms

where the data integration solutions are provided, technical details how different technologies are working in backend or frontend point of view, are not part of the research.

This research focuses to data integrations from case company's point of view and all processes beyond the case company's solutions are left out of the scope. For example, integrating real data from a data warehouse to a global schema or integrating organizations external and internal data models are processes that are usually beyond the reach of the case company, and therefore left out of the scope. In this research context, the assumption is that documentation and other information what the data presents in data models is either given by a customer or gathered from the customer.

Business reasons *why* the data integrations are needed and *what* are the business benefits, are left out of the scope. From case company's point of view, these are more related to sales, marketing and R&D and therefore, not part of the research objectives. Even though it is important to understand business processes behind the data integrations, business processes are covered only at the level which is relevant for this research. The objective is to achieve an understanding of utilizing data integration modeling methods in solutions provided by people working with customers in the case organization.

Since the case organization provides data integration solutions for many different industries, the goal was not to focus profoundly to data integrations related to certain industry's business processes, but rather to achieve a general understanding of designing solutions regardless of the business domain. This way the research results can be utilized in wider scale.

1.3 Research methodology

As a start of a research process, the research methods need to be decided. Decisions during the research process are affected by broader assumptions and ideologies. (Saunders et al. 2016) Saunders et al. (2016, p. 124) has presented a research onion to describe different methodological choices and methods. It describes the research philosophies, approaches, choices, strategies, time horizon and data collection techniques and procedures (Saunders et al. 2016, p. 124). That research onion is adapted for this thesis and figure 1 presents selected research methods.

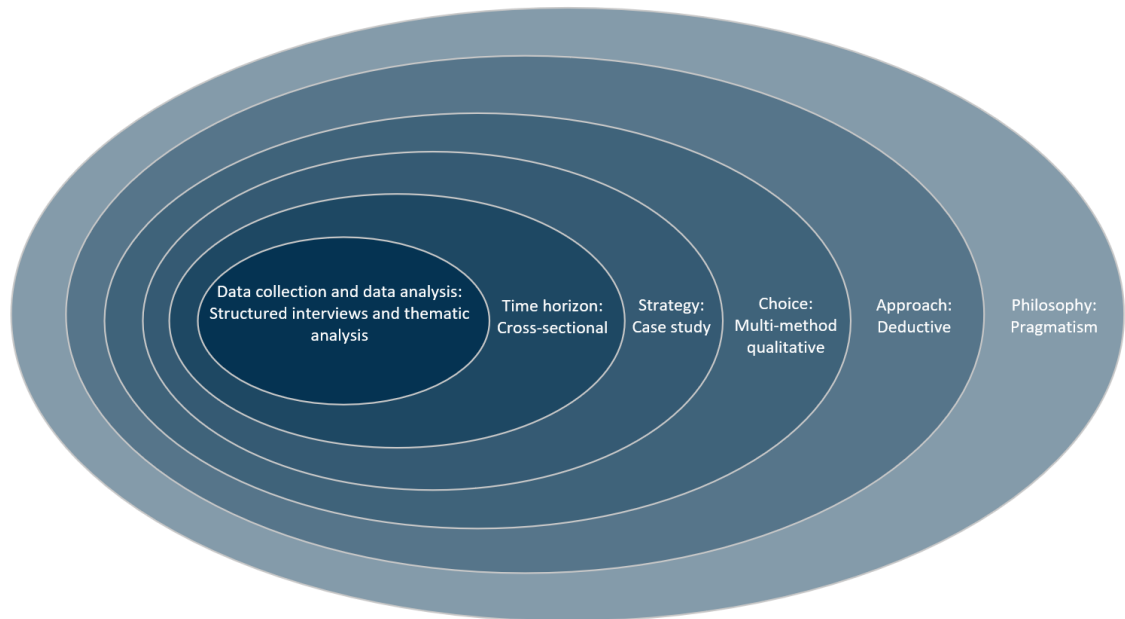


Figure 1: Research onion (Adopted from Saunders et al. 2016, p. 124)

Research philosophy describes how the world around is viewed and sets assumptions how the knowledge is developed (Saunders et al. 2016, p. 124). Selected research philosophy affects further decisions and therefore all strategies are not suitable for all philosophies (Saunders et al. 2016, p. 178). According to Saunders et al. (2016, pp. 135-144), there are five major philosophies that are positivism, critical realism, interpretivism, postmodernism and pragmatism. This research follows pragmatism philosophy that strives to find out practical solutions by giving answers for the research problems and questions (Saunders et al. 2016, pp. 137-144). The pragmatism philosophy was suitable because that allows to provide practical solutions for the case company based on the research results and therefore, create value for the case company.

Saunders et al. (2016, p. 145) presents three different approaches to theory development. Deductive approach means that the research starts with theory and the research tests that theory. Inductive approach starts by collecting and analyzing data that is used to develop a theory. Abductive approach contains elements from both approaches. Abductive approach starts similar way than inductive approach, but after the theory is created, it is then tested via empirical data collection and analyses. (Saunders et al. 2016, p. 145). In this research, literature review was made first to formulate the theory background which was then tested with the empirical research. Thus, deductive approach was selected. This approach was chosen because that allowed to study what existing literature has found about the research topic to find out how those methods could be utilized in case company's context.

According to Saunders et al. (2016, p. 164), first methodological choice is to decide whether to select quantitative, qualitative or mixed methods for data collection. Theory background was created by collecting qualitative data from existing literature. The data collection of the empiric part of this thesis was made with interviews. The goal of the interviews was to collect non-numeric qualitative data to research data integration modeling in the case company's context. Since there are two different data collection methods it makes this paper a multi-method qualitative study. (Saunders et al. 2016, p. 168) Qualitative data was suitable to understand reasons behind actions and choices.

Saunders et al. (2016, p. 177) describes research strategies as a plan to achieve the research objectives. For this thesis, case study strategy was chosen, because the goal was to study data integration modeling in the case company's context. Single case was selected, because it allowed to research unique case profoundly (Saunders et al. 2016, p. 186). Case study was suitable strategy due the pragmatism philosophy, meaning that this research produces valuable knowledge from case company's point of view and the results can easily be utilized to improve the case company's operations.

Time horizon defines whether the research studies a "snapshot" of certain time (cross-sectional) or if the research studies something over time (longitudinal) (Saunders et al. 2016, p. 200). Time horizon for this research is cross-sectional, because this paper focuses to understand a phenomenon at a certain time. This choice was made to get an understanding about the current state to provide practical solutions towards the future. The environment where the case company operates is changing rapidly, which is why longitudinal time horizon could have provided results that are not valid today.

For this research, data collection methods were literature review and structured interview. The research interviews were held as a conversation between two people, as Saunders et al. (2016, p. 388) suggests doing. The interviewer asks questions, to which the interviewee responds, and the interviewer carefully listens the answers. Interviews are used to collect valid and reliable data to achieve the research goals. (Saunders et al. 2016, p. 388) Collected data was analyzed using thematic analysis, which is according to Saunders et al. (2016, p. 579) a suitable method for a deductive research approach. Interviews allowed to understand better how the case company operates to find out what could be improved. Sections 4.2 and 4.3 describes in more detail how the interviews were done in this research.

1.4 Research structure

This research contains seven chapters, which are introduction, data integration, data integration modeling, research method and research setup, analyzing the research results, discussion, and finally, conclusions and summary.

First three chapters introduce the research objectives and provides background knowledge of the research topic based on existing literature and theory. The goal of chapters two and three is to describe data integrations and data integration modeling in the research context.

Fourth chapter introduces the empirical part of the research. It introduces the case company and defines how the empirical research was made and what were the reasons for research decisions.

Fifth chapter presents material collected through interviews. Sixth chapter analyses and combines interview results with existing literature and sums up differences and similarities in them. It describes the outcome of the research and provides answers for the research questions.

Last chapter summarizes the research and concludes the results. It provides a conclusion how data integration modeling could be utilized in the case company. Also, possible suggestions towards future researches related to the topic are mentioned.

2. DATA INTEGRATION

2.1 Clarifying the concept

Dejan (2007) defines integration as a process of enabling a communication between different software components. According to Lenzerini (2002), data integration combines data from different sources and provides the user a unified view of that data. IDC (2019) defines the term as following: *“Data Integration enables the access, blending, movement, and integrity of data among multiple data sources”*. IBM (2019b) describes data integration as *“the combination of technical and business processes used to combine data from disparate sources into meaningful and valuable information”*.

Organizations exchange data to support their business processes, such as purchasing, sales, shipping and invoicing. These business processes create business transactions, such as purchase orders, sales orders, shipping notices and invoices, that needs to flow between the trading partners. When these transactions are processed in an automated and standardized way, the interchange of data is called as Electronic Data Interchange (EDI). (Hill & Ferguson 1989; Manouvrier et al. 2008) Edifactory (2019) defines EDI as the application-to-application exchange of business documents between trading partners or business in a standardized electronic format. Hill & Ferguson (1989) defines EDI as the movement of business data electronically from one business application to another in a structured and computer-processable data format, whereas Opentext (2018a) describes EDI as *“the computer-to-computer exchange of business documents in a standard EDI document format between business partners”*. Veselá (2017) summarizes the idea of EDI as the *“exchange of standardized structured business and other documents in electronic form”*, as presented in figure 2.



Figure 2: EDI in a nutshell

Hill & Ferguson (1989) argued decades ago that EDI has multiple benefits that have been confirmed during the years (Kuchibhotla et al. 2009; Debicki & Kolinski 2018). EDI

reduces delays in business processes, because transactions can be processed automatically minimizing the time of information flow (Hill & Ferguson 1989; Kuchibhotla et al. 2009; Debicki & Kolinski 2018). Automatic processing also reduces labor costs, because it reduces the need of paper-based business processes and manual work (Hill & Ferguson 1989; Kuchibhotla et al. 2009). When data is processed automatically, it reduces errors caused by human and therefore reduces costs (Hill & Ferguson 1989; Kuchibhotla et al. 2009; Debicki & Kolinski 2018). In summary, EDI allows organizations to increase their productivity and efficiency by exchanging real-time information in interactive and standardized way with their trading partners (Kuchibhotla et al. 2009).

To summarize the purpose of EDI in practice, it helps organizations to automatize their business processes. EDI integration transfers data between systems using standardized electronic formats and enables computers to process the data. (Hill & Ferguson 1989; Manouvrier et al. 2008; Veselá 2017; Edifactory 2019) For example, an enterprise resource planning (ERP) systems have interfaces for EDI (Saeed et al. 2005), which means that EDI integration allows different ERP systems to connect between each other. Therefore, a business transaction processed in customer's ERP system flows to supplier's own system via EDI, allowing the customer and the supplier to process business transactions in their own systems while minimizing manual work (Mukhopadhyay & Kekre, 2002; Saeed et al. 2005).

Application-to-application (A2A) integration refers to integrations between different applications, such as integration between front-office applications and back-office applications (Manouvrier et al. 2008). It typically means integrations inside an organization (Manouvrier et al. 2008; Gartner 2016). Informatica (2019) defines application integration as *"the merging and optimization of data and workflows between two disparate software applications, often a new cloud application with a legacy on-premise application."*

Manouvrier et al. (2008) claims that business-to-business (B2B) integration corresponds to A2A integration between different organizations. B2B integration allows organizations to exchange business data electronically and B2B integration term also includes EDI (Bussler 2002; Gartner 2015; Gartner 2016). B2B integration integrates different data formats that have different data representations (Cardoso & Bussler 2011). Opentext (2019) defines B2B integration as *"the digital integration, automation and optimization of key business processes that extend outside the four walls of your organization"*, whereas IBM (2019a) defines B2B integration as *"the automation of business processes and communication between two or more organizations"*. Thus, B2B integration contains both A2A and EDI integrations between different organizations (Manouvrier et al. 2008; Bussler 2002; Gartner 2015; Gartner 2016; Cardoso & Bussler 2011).

In summary, B2B integration contains multiple message flows between two or more organizations and thus, also between different applications, allowing them to automatize their business processes. One simple example is the exchange of purchase order (PO) and purchase order confirmation (POC) between customer and supplier. In this message flow, the customer (sender) sends a PO to a supplier (receiver) and the supplier (sender) confirms the PO by sending a POC back to the customer (receiver). These message flows together create a B2B integration. (Bussler 2002)

It is important to notice that the terms related to integrations are referred in multiple contexts and therefore, the definitions vary even between organizations in the industry. However, all definitions have the same main idea: Move data from one location to another location and transform the data from one format to another. Data integration in the research context is presented in figure 3.

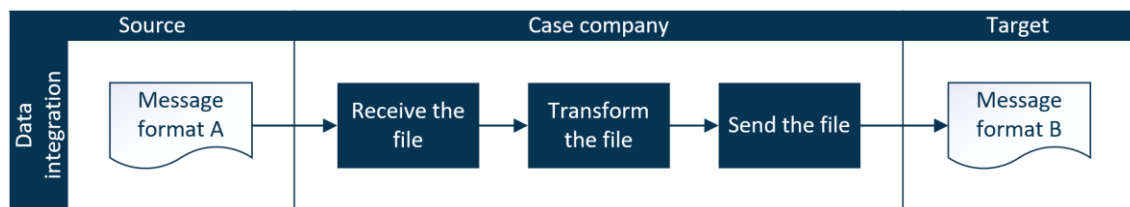


Figure 3: Data integration in the research context

In this research, the term data integration means process where messages are transformed from source destinations into another format and transferred to target destinations, utilizing the case company's integration platforms (Hill & Ferguson 1989; Bussler 2002; Manouvrier et al. 2008; Gartner 2016). As Gartner (2016) summarizes, the case company can provide any-to-any integrations. The term includes all integrations that the case company provides to its customers, regardless whether they are considered as EDI, A2A, B2B, or any other integrations.

2.2 Data models

Understanding data models is crucial in data integrations to enable understanding about what information the data portrays and how different entities are linked to each other. As Shahbaz (2015) claims, data models are necessary to understand the data. Therefore, general understanding of data models helps to create a data integration between different data models (Shahbaz 2015). Tillmann (2017) has given following definition for data models: *“A data model represents the definition, characterization, and relationships of data in a given environment”*. Data models are divided to conceptual, logical and physical models (Shahbaz 2015; Tillmann 2017).

Data models are results of data modeling (Shahbaz 2015; Tillmann 2017). Data modeling process starts by capturing the business requirements, which results in a high-level conceptual model (Gogolla & Hohenstein 1991; Batra & Marakas 1995; Hüsemann et al. 2000; Zuo & Zhang 2008; Shahbaz 2015). Conceptual model provides a documentation of the business requirements and it can be used as the starting point of the design (Fan et al. 2012). The next step of the data modeling is to create the logical model from the conceptual model (Gogolla & Hohenstein 1991; Shahbaz 2015; Tillmann 2017), which fills the gap between conceptual and physical model (Zuo & Zhang 2008). Physical data model adds technical details to logical data model, such as data types of attributes (Shahbaz 2015; Tillmann 2017) or in database context, tables and columns (Zuo & Zhang 2008; Tillmann 2017).

According to Fan et al. (2012), conceptual model increases the understanding of the context and provides a way for different stakeholders to communicate. Conceptual model contains information only in abstract level and thus, details are not presented (Shahbaz 2015). For example, street address, post office, postal code and country code can be combined into a single concept; address. Conceptual model is a system-independent model which means that it can be created before technology is selected (Gogolla & Hohenstein 1991; Zuo & Zhang 2008).

In logical data model, the objective is to identify logical entities, attributes and the relationships (Shahbaz 2015; Tillmann 2017). According to Tillmann (2017), entity is *"a person, place, or thing about which an organization wants to save information"* and attribute is *"a property or characteristic of an entity"*, meaning that an attribute gives additional information of an entity. An entity can be either an entity type defining a group of things, such as customers, or entity instance which is one occurrence of an entity type, such as "Customer A". Attribute can be used, for example, to define a name or a color of a product. Relationships are used to connect entities and they are usually verbs. For example, if a customer buys a product, the customer and the product entity form a relationship. (Tillmann 2017)

According to Tillmann (2017), the purpose of a logical data model is to help communication. Therefore, the logical data model does not have technical limitations such as maximum length of an entity (Tillmann 2017). Logical data model presents the information from end-user's perspective (Tillmann 2017), as is demonstrated in figure 4.



Figure 4: Logical data model according to definition of Tillmann (2017)

The figure 4 presents a logical data model of a real-life transaction where a customer buys product with certain color and name. Customer and Product are entities, Buy is a relationship and Product color and Product name are attributes of a product.

Minimum and maximum number of possible relations between entities can vary, as well as relationships can be mandatory or optional. A Finnish person has only one personal identification number and one personal identification number can be assigned only to one person, which is an example of a one-to-one relationship. A customer can buy multiple products as well as a product can be bought by multiple customers, so they create many-to-many relationship. Invoice can have multiple invoice lines, but all invoice lines are related only to one invoice. Thus, invoice and invoice lines have one-to-many and many-to-one relationships. (Tillmann 2017)

An invoice must have invoice lines and an invoice line must relate to an invoice, hereby having mandatory-mandatory relationship. A customer might not have created an order yet, but each order must be related to a customer, which is an example of mandatory-optional and optional-mandatory case. Optional-optional means that an occurrence of entity A does not need to be related to any occurrences of entity B, and vice versa. (Tillmann 2017) For example, a customer does not need to buy a product and neither a product needs to be bought by any customer, but the data model still has a relationship between them allowing that transaction to happen.

Relationships can also bind together multiple entities (Tillmann 2017). For example, if there are three entities (Customer, Product, Supplier) and a customer buys a product from a supplier, there will be relationship between all three entities. In data integration context, all these relations within a data are important to understand the structure of the data models that needs to be integrated.

Patig (2006) as well as Tillmann (2017) states that Entity Relationship Diagram (ERD) presents a logical data model, whereas Masri et al. (2008), Zuo & Zhang (2008) and Cagiltay et al. (2013) claims that ERD is a conceptual data model. Entity-Relationship modeling is one approach towards data modeling, which was developed by Peter Chen

in 1976 (Patig 2006). Entity-Relationship modeling is widely accepted method and it presents the real world in easily understandable way (Gogolla & Hohenstein 1991; Masri et al. 2008). An ERD presents entities and their attributes and the relationships between entities (Patig 2006; Halpin & Morgan 2010; Cagiltay et al. 2103).

Especially the correlations between data models might be complex leading to situation where the data integration cannot be done only by mapping one source entity to one target entity. Let's illustrate this with two ERDs presented in figure 5.

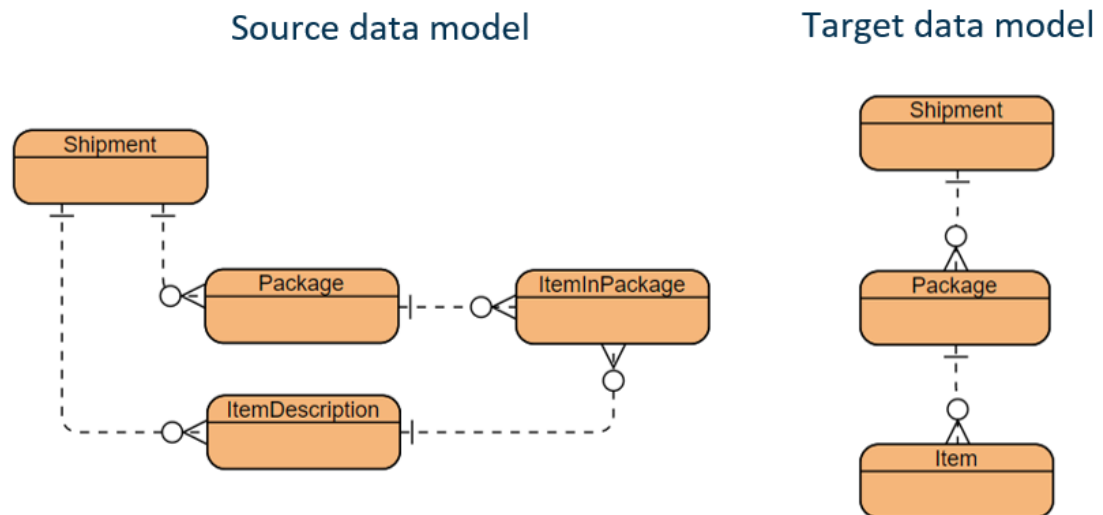


Figure 5: ERDs presenting Shipment in two different data models

The figure 5 illustrates two different data models for a shipment. In the source data model, a shipment contains packages and each package contains a list of items that are packed to the package. Description of those items are under the shipment. "ItemDescription" and "ItemInPackage" entities have one-to-many relationship, meaning that each type of item can be packed into multiple packages, but one item packed into a package refers to certain item description. In practice, this could mean that a package contains only identifiers that are linked to each "ItemDescription" entity which have details about the item that is in the package, such as product names and codes. In the target data model, a shipment contains packages and each package contains items that are in the package. (Patig 2006)

When different data models need to be integrated, there are several challenges. Looping logic needs to be created to map source's "ItemDescription" entity to target's "Item" entity while the number of packages and items in packages should remain same in both source and target models. This kind of challenges are typical in data integrations and solving these requires comprehension of data models. As Yang et al. (2016) paper shows, if the mapper does not know why data needs to be mapped in a certain way, it will lead to

incorrect mappings. It is also worth to mention that if the “ItemDescription” and “Package” entities had one-to-many relationship, the mapping could not be done from target to source, because the target data model does not restrict to have only one type of item in one package. Imagine a business case where the sender would ship water and wine bottles and the package label should define which bottles the package contains. Having a one-to-many relationship between item description and package would limit this, but the target data model still allows one package to have both water and wine.

Physical data model represents view of stored data (Tillmann 2017) and it is system dependent (Zuo & Zhang 2008). In data integration context, according to Joutsenlahti (2017), physical data models are necessary for designing the interfaces used in data integration since they present the physical schema. On the other hand, as Hoberman (2009) states, people have different interpretations about the concepts of logical and physical data model, leading to a situation that people might use different terms about the same thing.

Understanding data models helps to design and create data integrations, because it is necessary to understand data structure within a data model to know, for example, what entities exists only once, what entities occur multiple times and how different entities are linked to each other (Patig 2006; Yang et al. 2016). All different data model types can be utilized in different phases of a data integration solution, as we can see from figure 6.

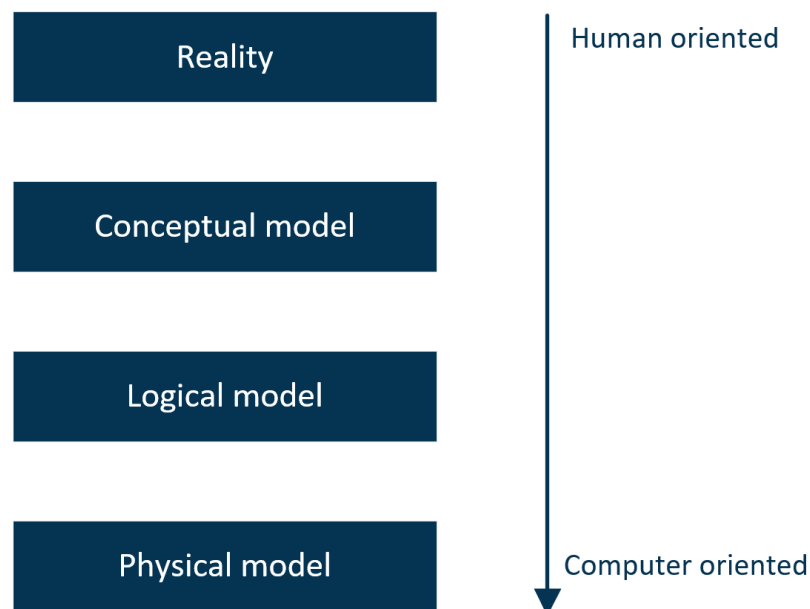


Figure 6: Data model types according to Shahbaz (2015)

In data integration context, when moving from initial solution design to technical development phase, the usefulness of data model types changes. This also means that lev-

eraging different data model types varies between roles (Fan et al. 2012). Whereas technical developer needs to understand the physical data models, such as data types or maximum length of an attribute, as Fan et al. (2012) states, managers and business analysts can utilize conceptual models when clarifying the business requirements.

2.3 Common message formats

As argued in previous chapter, EDI contains set of formats for exchanging information electronically (Kuchibhotla et al. 2009). Different EDI standards are used globally, such as American National Standards Institute (ANSI) X12 and Electronic Data Interchange for Administration, Commerce and Transport (EDIFACT) formats (Kuchibhotla et al. 2009; Schubert & Legner 2011). Lack of understanding the standards is one of the obstacles of data integration success (Jun & Cai 2003). According to Engel et al. (2012), semantic relationships between data elements defines the information content and understanding these needs expertise of the standards.

These EDI standards define also message types, such as order, invoice, or delivery schedule, and the information what each message type can contain. In addition to existing standards, also flat files (Kuchibhotla et al. 2009) and eXtensible Markup Language (XML) formats (Kuchibhotla et al. 2009; Schubert & Legner 2011; Reeve 2013) are used. According to Cardoso & Bussler (2011), B2B integrations have progressed during the years from EDI standards towards XML, but Engel et al. (2016) claims that ANSI X12 and EDIFACT still have nowadays a dominant role. For example, Walmart, one of the biggest companies in the world, demands their suppliers to use EDI and they support both ANSI X12 and EDIFACT standards (Walmart 2014), whereas the Incomes Register, a national electronic database for wages, pensions and benefits in Finland, uses XML format (Incomes Register 2019b). JSON (JavaScript Object Notation) format is also used in data integrations (Reeve 2013; Afsari et al. 2017; Barbaglia et al. 2017).

An EDI format standard contains rules for transforming a business document into an electronic message. A single electronic document is called a transaction set, where a group of information is called a data segment. Data segment contains pieces of data called data elements. (Hill & Ferguson 1989). For example, a purchase order is a transaction that contains party information in data segments, which have party identifiers, street addresses, city, postal code and country code in data elements. EDI standards include also composite data elements, which have two or more pieces of data (known as components), usually containing qualifiers. A composite data element is a data element made of sub-elements. (Engel et al. 2016; Edifactory 2019) A simplified example of a purchase order in EDIFACT D97A message format is presented in figure 7.

```

UNH+11234567+ORDERS:D:97A:UN'
BGM+220+PO12345'
DTM+137:20190412:102'
NAD+BY+FI12345678::92++Buyer Company A+Example street 1+HELSINKI++0100+FI'
NAD+SU+1234::92++Supplier B+Supplier street 4+TAMPERE++33100+FI'
LIN+10'
PIA+5+BuyerCode1:BP'
PIA+5+SupplierCode2:SA'
QTY+21:100:PCE'
DTM+2:20190503:102'

```

Figure 7: Snapshot of an EDIFACT purchase order

In this example, segments are separated with apostrophe (and with row breaks for clarity), elements with plus sign and subelements with colon. Example of composite data element can be seen from last row: The DTM segment has a composite data element “DTM” (Date/Time/Period) which is being qualified by value “2” (delivery date/time, requested). Next data element “20190503” is qualified with a subelement and its value “102”, which defines that the date format is CCYYMMDD. In English, this EDIFACT example is a purchase order message with a message number 11234567 that contains a purchase order number PO12345. The message was created at 12th of April 2019 and in this order, a buyer company with identifier FI12345678 has sent a purchase order to their supplier with identifier 1234 and has ordered the supplier to deliver 100 pieces of certain item to be delivered at 3rd of May 2019. (Edifactory 2019)

Whereas EDI standards and XML formats have named data elements for each data, according to Kuchibhotla et al. (2009), flat files separate the data elements either using fixed width, comma separated value (CSV) or tab-delimited text files. Regardless that the term “CSV” defines to use the comma to separate data fields, the term is commonly used to refer a broader variance of formats, and nowadays the term is used to refer to any file of “character-separated-values” (Mitlöhner et al. 2016).

Typical challenge in data integrations is that in addition to transfer the files between source and target, sender and receiver parties have different systems and therefore their systems process different data formats (Wang & Xu 2008). Thus, the data needs to be transformed from one format to another. All structured data formats, regardless whether they are EDI standards, XML files, flat files or any other format, have elements for each pieces of data, so that computers can process the business document automatically. Figure 8 presents same information than figure 7 but in simple XML format.

```

<?xml version="1.0" encoding="UTF-8"?>
<PurchaseOrder>
  <MessageNumber>11234567</MessageNumber>
  <PurchaseOrderNumber>PO12345</PurchaseOrderNumber>
  <MessageDate>20190412</MessageDate>
  <Parties>
    <Buyer>
      <PartyIdentifier>FI12345678</PartyIdentifier>
      <Name>Buyer Company A</Name>
      <StreetAddress>Example street 1</StreetAddress>
      <PostOffice>Helsinki</PostOffice>
      <PostalCode>00100</PostalCode>
      <CountryCode>FI</CountryCode>
    </Buyer>
    <Supplier>
      <PartyIdentifier>1234</PartyIdentifier>
      <Name>Supplier B</Name>
      <StreetAddress>Supplier street 4</StreetAddress>
      <PostOffice>Tampere</PostOffice>
      <PostalCode>33100</PostalCode>
      <CountryCode>FI</CountryCode>
    </Supplier>
  </Parties>
  <OrderLines>
    <OrderLine>
      <LineNumber>10</LineNumber>
      <SupplierItemCode>SupplierCode2</SupplierItemCode>
      <BuyerItemCode>BuyerCode1</BuyerItemCode>
      <OrderedQuantity>100</OrderedQuantity>
      <QuantityUnit>PCE</QuantityUnit>
      <RequestedDeliveryDate>20190503</RequestedDeliveryDate>
    </OrderLine>
  </OrderLines>
</PurchaseOrder>

```

Figure 8: Purchase order in simple XML format

Understanding EDI standards requires expertise, because for example, EDIFACT and ANSI X12 has certain message structure that is not visible in the file itself. Therefore, the location of a data segment in a file does matter. For example, in EDIFACT, DTM segment grouped to a LIN segment means that the date/time/period refers to the LIN segment, whereas first DTM segment refers to the message level data. (Edifactory 2019) When using flat files, there is no visible structure. This highlights the importance of understanding data models to understand how the data elements are linked to each other (Shahbaz 2015). According to Kuchibhotla et al. (2009), XML provides not only standardized, but also self-describing meaning to the information which results to a human readable format that can be easily verified, transformed, and published. When comparing the examples from figure 7 and 8, it is easy to agree that the latter one gives more information to a human. As Cardoso & Bussler (2011) summarizes, XML provides structure to the data. When an organization receives a message via data integration, they need to map that data to their own internal data representation that is visible only inside the organization

and might not be same than external data representation. This means for example a process that makes the XML file visible in an ERP system for the end user. (Cardoso & Bussler 2011)

2.4 Understanding the data within a message

In order to get the data flowing between endpoints correctly, there must be interfaces that process the messages. For that reason, to be able to create a data integration, technical interface specifications and guidelines are needed, such as a schema file and documents that defines how the schema is used. Depending on the used message formats, different schema files can be available. For example, Document Type Definition (DTD) and XML Schema Document (XSD) specifies what kind of data an XML document can contain (Hoberman 2009). Whereas XSD describes the elements of XML, JSON schema does the same for JSON documents (Barbaglia et al. 2017). Other documents may contain additional information about the data presented in the data model defining how the schema is used and what real-life information they contain, transforming the data to information. As Jun & Cai (2003) states, the benefits of data integrations are highly associated with clear guidelines. Example XSD is presented in the figure 9 (Incomes Register 2019c).

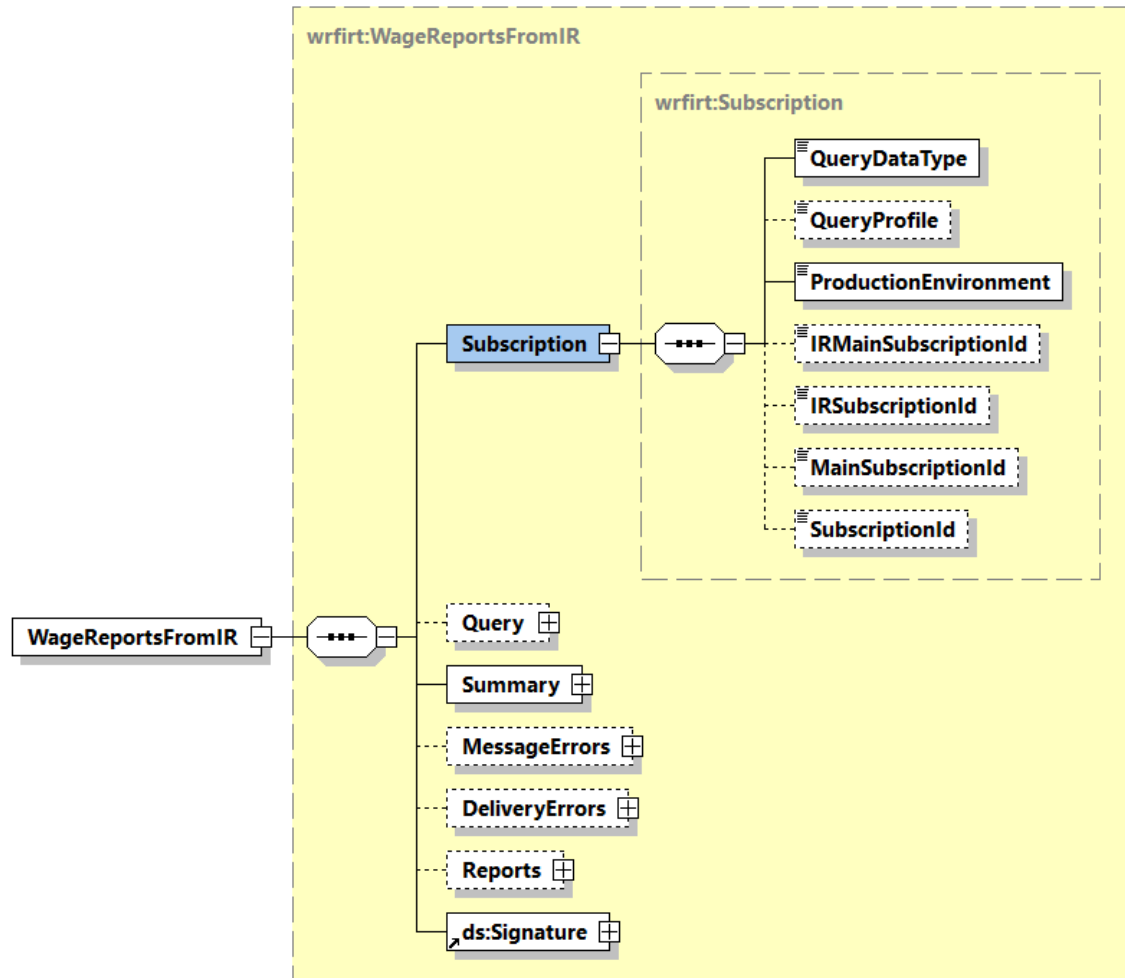
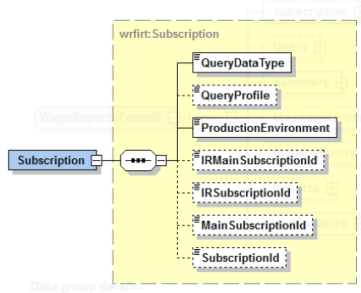


Figure 9: WageReportsFromIR XSD file (Incomes Register 2019c)

The XSD presented in figure 9 describes the data model of an XML document. Each XML document using this schema has “WageReportsFromIR” as a root element and multiple mandatory and optional elements under that element. (Incomes Register 2019c) The XSD does not only give a structure for an XML document, but it also defines what data each element and attribute can contain. For example, XSD can define that an element’s data type is 20-40-character long string, or Boolean type allowing values to be only “true” or “false” (Incomes Register 2019c).

When creating an interface and mapping from such schema, the schema file itself does not give all details needed to understand what information the data really presents. For example, schema does not provide information about the information content of “Query-Profile” entity in that specific case. Therefore, additional information is needed. Understanding the data is crucial for successful data integration (Jun & Cai 2003; Reeve 2013; Blanco et al. 2019). Figure 10 is an example of a document that gives more information how the schema should be used.

2.1 Record subscription details (Subscription)



Data designation	Type	Allowed values	V/M	Processing rule
Record subscription details (Subscription)	wrflrt:Subscription		M	This data group is used to distribute information related to a record subscription or a service request.
Record type (QueryDataType)	xs:int	Codes: QueryDataType	M	Data submitted in a record subscription or service request.
Data access profile (QueryProfile)	irct:String40		V	Data submitted in a record subscription or service request.
Production environment (ProductionEnvironment)	irct:trueOrFalse	Codes: ProductionEnvironment	M	Data submitted in a record subscription or service request.
Incomes Register primary subscription reference (IRMainSubscriptionId)	irct:Guid		V	The Incomes Register primary subscription reference for the primary subscription based on which the record was queried. The data is not included in the response message to a real-time WS service invocation compliant with the DataRequestToIR schema.
Incomes Register secondary subscription reference (IRSubscriptionId)	irct:Guid		V	The Incomes Register secondary subscription reference for the secondary subscription based on which the record was queried. The data is not included in the response message to a real-time WS service invocation compliant with the DataRequestToIR schema.

Figure 10: Schema content description (Incomes Register 2019a)

The importance and value of the specification depends on the case and differs between different stakeholders and expertise. For example, the “ProductionEnvironment” element in the schema presented in figures 9 and 10 has either value “true” or “false”, which indicates whether the message is used in a test environment or in a production environment (Incomes Register 2019a; Incomes Register 2019c). From technical developer’s point of view, this is a self-evident field and thus, even if there was no schema description available, they would know what data that field should contain, because the allowed values are defined in the XSD file. On the other hand, for a person that does not know what test and production environment means, the usage of the field could be unclear regardless the fact that there is such schema content description available.

In another example, the element “QueryProfile” contains 40-character long string as a value (Incomes Register 2019a; Incomes Register 2019c) and even the description presented in figure 10 does not give enough information what data that should contain. In situations like this, business domain expertise and additional information is needed, which demonstrates the importance of understanding the business while creating a data integration (Reeve 2013). As Blanco et al. (2019) states, understanding what information an entity represents from real-world is a fundamental problem in data integrations.

Documentation is also needed to define what elements of the schema are used (Edifactory 2019; Incomes Register 2019a). For example, EDIFACT standard has different data segments and elements for large number of business data, but typically only some of

those are used depending on the business case (Engel et al. 2012). As Liegl et al. (2011) states, in practice, only a small percentage of the elements within a schema are used. Thus, it is necessary to define the relevant subsets of elements (Liegl et al. 2011). Parties involved in an order process is typical example as demonstrated in figure 11 with three different use cases.

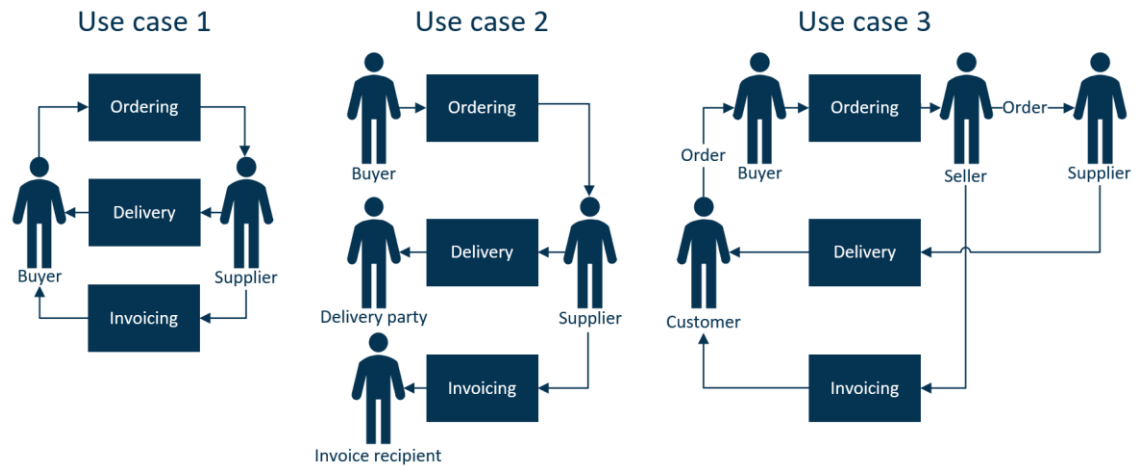


Figure 11: Order process with different use cases (Adopted from Peppol 2019)

In the use case 1 presented in figure 11, an order process is always between a buyer and a supplier, meaning that the supplier will deliver goods and bill the buyer party based on the orders received from the buyer. In the use case 2, even though the buyer orders goods from the supplier, the supplier delivers the goods to another party, like as a specific plant of a buyer organization, and bills third party, such as the financial institution within a buyer organization. Use case 3 could illustrate an ordering process in e-commerce, where an end customer (Customer) creates an order in a buyer's webshop. The buyer sends an EDI order of this event to the retailer (Seller) who orders the supplier to deliver the goods (Peppol 2019). Delivery happens directly from the supplier to the customer, and the retailer bills the customer.

While in the use case 1, the order message could have only buyer and supplier party information, the use case 2 needs four different parties so that the supplier would know how to handle the order. It is also worth to mention that in the use case 3, seller and supplier are not synonyms, which is the case in use cases 1 and 2. There are multiple variances of the involved parties. For example, each party in use case 3 could bill the party who sent the order: The supplier sends an invoice to the seller, the seller invoices the buyer, and the buyer bills the customer. In another example, the delivery could happen from the supplier via consignor (like FedEx or UPS) to ship-to party (such as a post office) from where the customer would pick up the goods. This demonstrates the importance of defining what data segments are used as well as highlights the importance

of understanding the meaning of the data. As Reeve (2013) states, the data integration is not only about implementing technological solutions, but it also requires understanding the business context to identify the meaning of a data.

2.5 Data transformation and mapping

Data mapping is a process of mapping the source schema to a target schema (Dong et al. 2009; Shahbaz 2015). It creates the link between source and target data models (Kalinichenko 1990; Shahbaz 2015). Mapping creates the relations between source and target data entities, and it is one part of data transformation, which converts the source format to a target format (Kalinichenko 1990; Gleghorn 2005; Wang & Xu 2008). Since the objective of a data integration is to get data from source destinations to target destinations in different data formats, the data transformation and mapping are obligatory part of data integration solutions (Lenzerini 2002) and understanding the mapping between schemas is necessary in data integrations (Amano et al. 2014).

According to Shahbaz (2015), the biggest challenge of a mapping process is to understand how the data should flow from source to target. Mappings between data models is a time-consuming task, because understanding the meaning of the data presented in message elements is not always obvious (Cardoso & Bussler 2011). Therefore, the technical developer needs to know what information the data in different data models presents. If the technical developer does not have enough expertise about the industry and therefore about the data, understanding relationships of entities might be hard (Shahbaz 2015), which highlights the importance of understanding data models as well as business. There are different methods how to do the mapping, such as point-to-point, canonical or semantic mapping. This research focuses only to point-to-point and canonical mappings, which are according to Gleghorn (2005) the most common methods.

In a point-to-point approach, data mapping is made between each source and target interface. This means that new connection is created between each interface that needs to be integrated. (Gleghorn 2005) Point-to-point integrations can be created relatively cheaply and fast (Dejan 2007), but they are difficult to maintain (Dejan 2007, Galinec et al. 2012). Point-to-point topology is presented in figure 12.

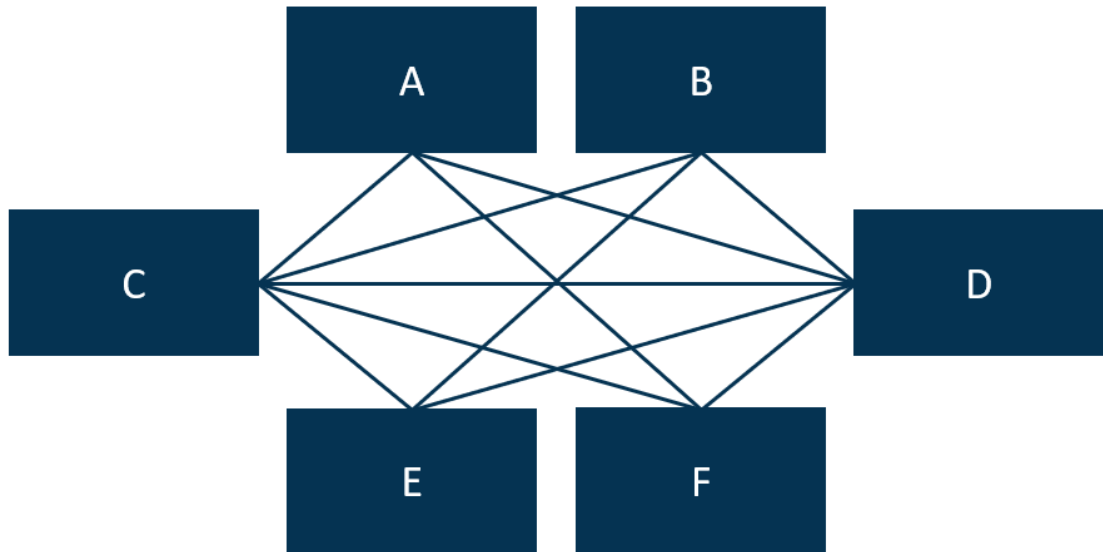


Figure 12: Point-to-point mapping

The number of connections increases nonlinearly compared to the number of endpoints needed to be integrated. For example, to have full-meshed point-to-point topology with 6 interfaces, 15 connections are needed, whereas 12 interfaces require 66 connections. If there is need to do changes to one interface, the same change needs to be done to all connections from and to that interface, which is one of the reasons why the maintenance of point-to-point integrations is challenging (Dejan 2007; Reeve 2013).

When mapping is done using an integration hub, only one connection is needed per interface. Changes done to an interface affects only to a connection between that interface and hub. (Dejan 2007; Reeve 2013) In data model context, this hub can be considered as a canonical model (Reeve 2013; Sarkar 2015). Gartner (2015) summarizes canonical mappings as following *“In practice, every company on the network produces only one set of maps from its internal systems to the B2B network canonicals, eliminating the need for customer-specific maps and substantially reducing integration project implementation efforts”*. According to Sarkar (2015), canonical model uses commonly agreed definitions for data entities, and it presents the data using standardized data formats. This means that each data entity in a canonical model is unambiguous and it represents certain thing from a real-world, such as an entity for invoice number or ordered quantity. Mapping done via canonical is presented in figure 13.

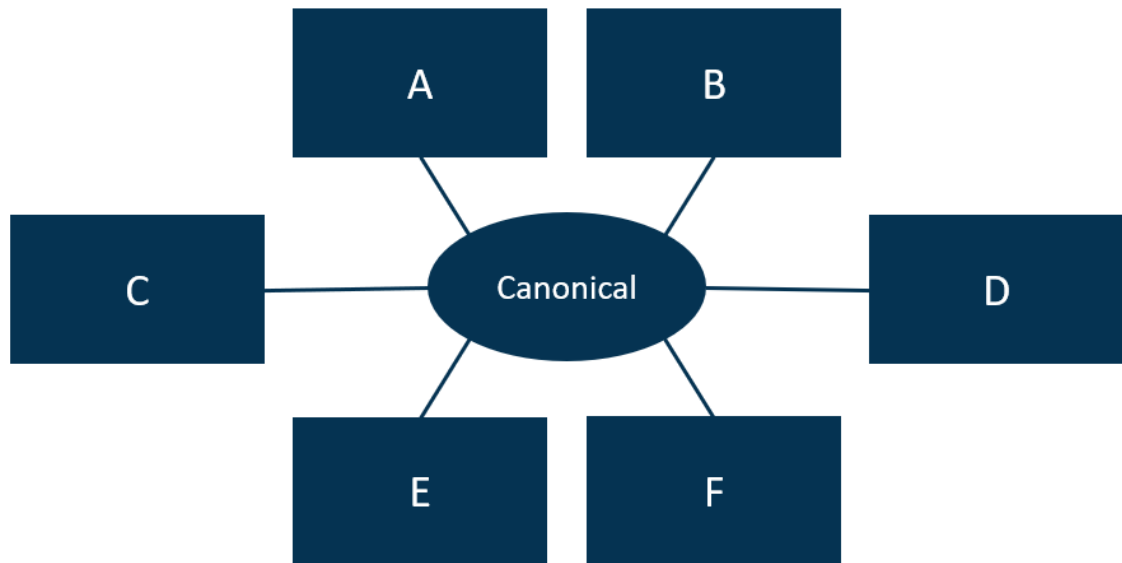


Figure 13: Canonical mapping

In the figure 13, an interface needs to be mapped only to the canonical, regardless the number of other interfaces that interface needs to communicate with. If a new interface G would be added, a new connection between canonical and interface G would allow it to communicate from mapping point of view with all existing interfaces in the topology. (Gleghorn 2005; Dietrich & Lemcke 2011; Reeve 2013; Gartner 2015). When utilizing canonical model in data integrations, overall architecture is less complex compared to point-to-point approach and thus, the maintenance is easier (Gleghorn 2005; Dejan 2007; Gartner 2015). Whereas an ERP migration from current interfaces to new interfaces would require change to all point-to-point mappings, when a canonical is used the change needs to be done only to one connection. However, it must be noted that such change would most likely affect to many other things, too, instead of mapping, such as how business transactions are processed in two different systems. That being said, this still demonstrates why the maintenance of point-to-point mappings needs much more effort compared to canonical approach (Gleghorn 2005; Dejan 2007; Gartner 2015).

As Dong et al. (2009) states, the goal of mapping is to create the relationships between a source and a target schema. When mapping is done point-to-point, there is need to understand only source and target schemas and the relations between them (Gleghorn 2005; Dejan 2007). Simplified example of point-to-point mapping between entities in schemas A and B is presented in figure 14.



Figure 14: Point-to-point mapping of product codes

In point-to-point mapping, the technical developer does not need to understand what the “ProductCode” entity exactly means to create correct mapping, if the correlations of the schemas and their entities are known (Gleghorn 2005; Dejan 2007). This is one of the reasons why point-to-point mapping is faster way to create an integration between two different data formats (Gleghorn 2005). If the mapping is done via canonical data model, the technical developer still needs to understand both and target interfaces, but the canonical data model, too. Otherwise the integration between an interface and canonical might have semantically wrong mappings. In figure 15, same example is presented using simplified canonical data model.

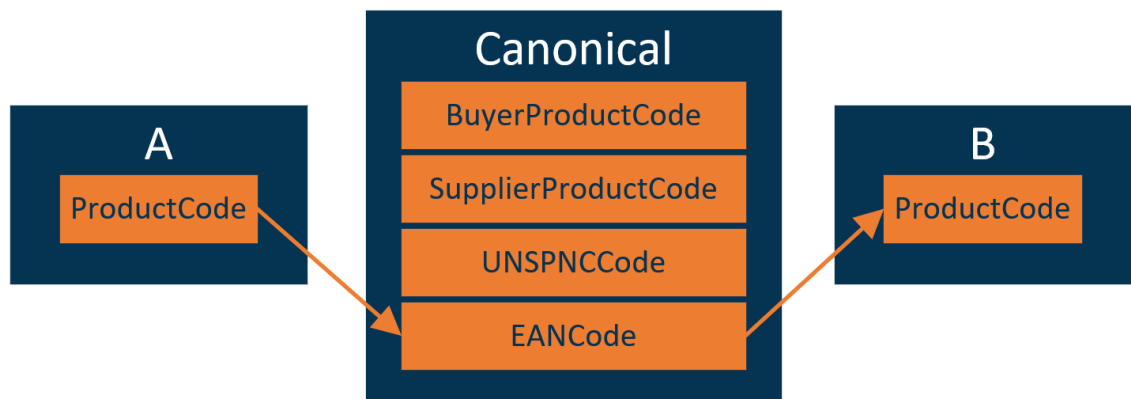


Figure 15: Mapping product codes using a canonical data model

In this case, the technical developer needs to know exactly what kind of product code the schemas A and B contains in the “ProductCode” entity to be able to create the mapping using semantically correct canonical data field. Therefore, additional knowledge is needed to determine the meaning of the data element (Engel et al. 2012). This is one of the reasons why point-to-point integration is usually faster way to implement new integration, since it simply requires less knowledge about the data (Gleghorn 2005; Dejan 2007).

Challenges of canonical data model arises when canonical data model and integrated data models have big differences. According to Gartner (2015), canonicals that are not specifically defined to certain industry and its business processes, should be avoided.

Problems could occur when canonical data model does not have an entity that exists in integrated data models. How to map the product code between interfaces A and B via canonical presented in figure 15, if the product codes are, for example, electrical numbers (product code issued by Finnish Electrotechnical Trade Association, STK)? Mapping cannot be done semantically correctly if that entity is missing from the canonical, whereas point-to-point mapping could be done easily by just directly mapping the data from source entity to target entity. Same kind of issue occurs when the structure of a canonical model is different than the source or target data model, which means that the looping logic of mapping requires extra effort. To tackle these problems, the canonical data model should capture the semantic equivalences of all different data models (Dietrich & Lemcke 2011).

One benefit of a canonical data model is that existing solutions can be used as background knowledge (Dietrich & Lemcke 2011). When canonical mapping method is used, each new map does not need to start from a scratch (Gartner 2016). If a data model needed to be integrated is completely new, existing solutions related to same business process might give information what kind of data there will most likely be flowing, regardless the data model, schema or entity names. Therefore, understanding the canonical data model can lead to improved understanding of other data models (Dietrich & Lemcke 2011). If data needs additional processing, such as databases or custom mapping logic, same implementations can be utilized in multiple integrations with some general solutions. This means that each new implementation does not need to be data model specific, but instead, they can be used in all implementations that use the same canonical data model.

3. DATA INTEGRATION MODELING

3.1 Modeling process

In data integrations, business processes define the requirements for information processing. Data integration technology implementation provides information processing capabilities for these needs. (Schubert & Legner 2011) As Al-Naeem et al. (2004) states, designing data integration solution is a complex process and therefore different level of abstraction is needed. For example, business process interactions are high-level design issues whilst, message routing techniques are considered as low-level aspects (Al-Naeem et al. 2004). High-level processes need to be clear before low-level details can be done, since they affect to each other (Giordano 2010). Giordano (2010) has defined data integration modeling as a *“process modeling technique that is focused on engineering data integration processes into a common data integration architecture”*.

Different modeling methods can be utilized to document the business needs as well as the technical solution. The purpose of modeling methods is to illustrate graphically the processes that data integration solution performs. (Giordano 2010) Different diagrams are commonly used to model systems (Eriksson & Penker 2000; Meier & Meier 2011; Appavuraj et al. 2014; Yang 2014; Tillmann 2017; Baqais & Alshayeb 2018; Zhang et al. 2018), such as flowcharts and structure charts (Tillmann 2017) or other diagrams (Appavuraj et al. 2014). Unified Modeling Language (UML) diagrams are also used to model systems (Eriksson & Penker 2000; Giordano 2010; Baqais & Alshayeb 2018), but as Medvidovic et al. (2002) states, UML is not suitable to illustrate architecture-level components and therefore, other diagrams are needed. Different diagrams can be used to represent a solution from different perspective illustrating, for example, *what* processes are performed, or *when* and *how* they are performed, as well as demonstrate *who* and *where* actions are performed (Kim et al. 2000). As Kim et al. (2000) claims, complex systems need more than one diagram to model the solution from different perspectives. As Li et al. (2018) mentions, all models are not suitable for all stakeholders. Some people want to understand how the solution is linked to business processes, while others want to have a technical description of the components that are part of the solution (Li et al. 2018).

Even though graphical diagrams can be useful (Kim et al. 2000; Giordano 2010), Tillmann (2017) highlights that just writing down what the solution should do is however the

best way to document the solution. This is the reason why many technical documentations contain both graphical visualizations as well as text. In conclusion, both graphical diagrams and English can be utilized to model data integration solutions. (Tillmann 2017)

According to Halpin & Morgan (2010), information systems can be viewed from four different levels: conceptual, logical, physical and external. According to Giordano (2010), also data integration modeling should contain conceptual, logical and physical levels. Conceptual level is the high-level definition that describes the business process in a human understandable way (Giordano 2010; Halpin & Morgan 2010). Conceptual modeling can be done before technology and software are selected (Fan et al. 2012), which means that it is system independent presentation (Hüsemann et al. 2000; Giordano 2010). According to Fan et al. (2012), conceptual models are used to capture the business requirements. Logical models define technology specific requirements and physical model is used in the system execution phase (Giordano 2010; Fan et al. 2012). In another words, logical model presents functional capabilities whereas physical model illustrates how the system functions (Tillmann 2017). Figure 16 presents the relationships between these three different models.

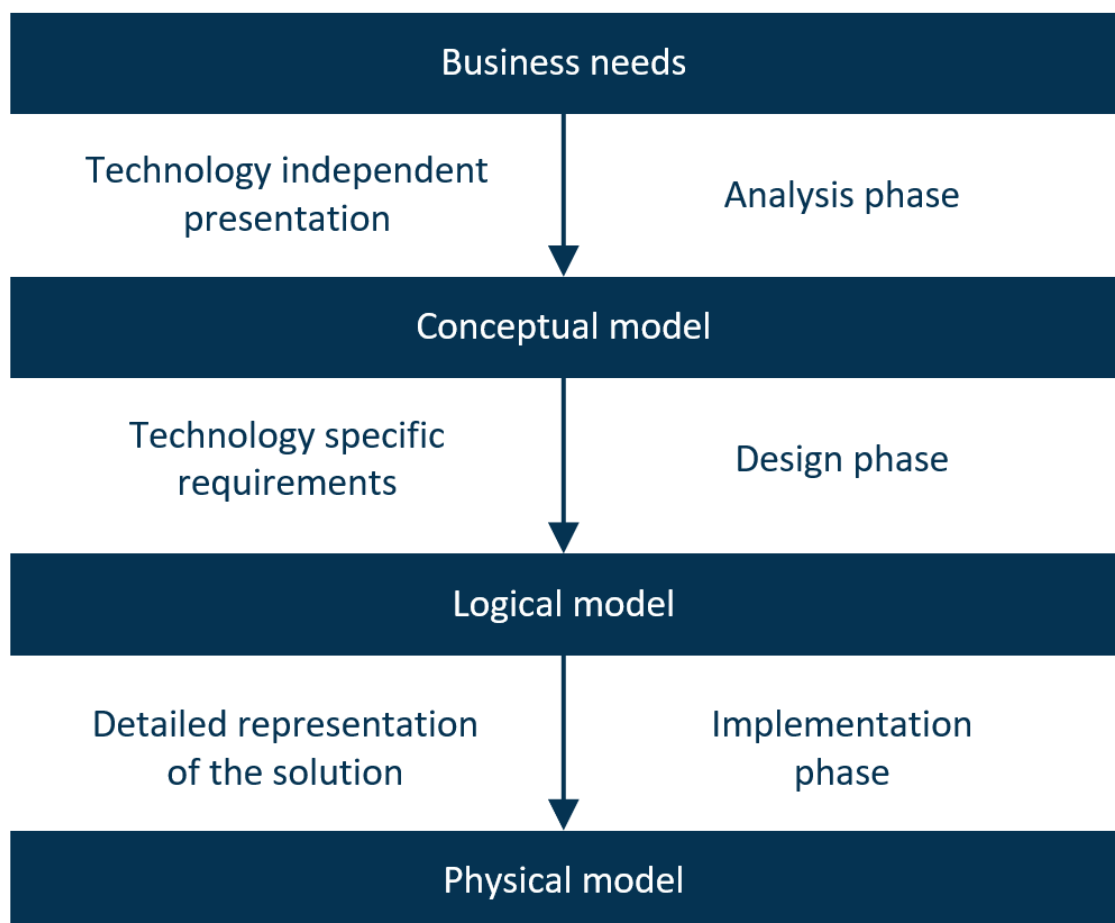


Figure 16: Modeling process (Adopted from Fan et al. 2012)

Conceptual model is the result of business process analysis and it presents overall solution in a system independent way (Hüsemann et al. 2000; Giordano 2010; Fan et al. 2012). After conceptual model is done, next step is to start designing how the conceptual model can be implemented with technology specific requirements and restrictions (Giordano 2010), such as using certain data integration platform. Giordano (2010) states that the conceptual model should provide a visual representation about the scope, such as defining source and target interfaces or systems. According to Tillmann (2017), the most popular methods for modeling logical processes are data flows and English. Physical model contains detail level information about the solution, meaning that it is finalized during the implementation phase. Tillmann (2017) states that most popular methods for physical models are flowcharts, structure charts, pseudocode and English.

3.2 Understanding the business

Understanding business processes behind the data integration solution is crucial to understand what data needs to flow via data integration and what the solution needs to do (Giordano 2010). As Berente et al. (2009) states, understanding the information flows related to business processes should be one of the first steps when creating any integration. The goals of a data integration solution should be clear (Berente et al. 2009). Business objectives defines the data integration design decisions (Al-Naeem et al. 2004; Giordano 2010).

Business process contains flows and activities, where an output of activity is the input of next activity (Berente et al. 2009). Business processes typically contain multiple transactions (Bussler 2002) and understanding how they link to each other is essential to understand the data flow (Giordano 2010; Reeve 2013). Knowing only message types and formats is not enough to capture the business requirements. Especially if there is some data needed from one transaction to another transaction, the sequence of different transactions is important. If a message transaction triggers another message transaction, the sequence matters (Giordano 2010; Reeve 2013). Real-life events should be known to understand what business event triggers a message transaction as well as what business event does a message transaction trigger (Bussler 2002; Berente et al. 2009). As Dorn et al. (2009) and Giordano (2010) states, the design process needs to start from business models and business process models to define what kind of solution needs to be implemented.

An example business case: A buyer goes to supplier's store to get an item. The supplier does not have that item in the store, so the goods will be delivered to the buyer from the

supplier's warehouse. This business process needs to be automatized with a data integration solution between ERP systems. Real-life business events are presented in the figure 17 with orange color and ERP system message transactions with blue color.

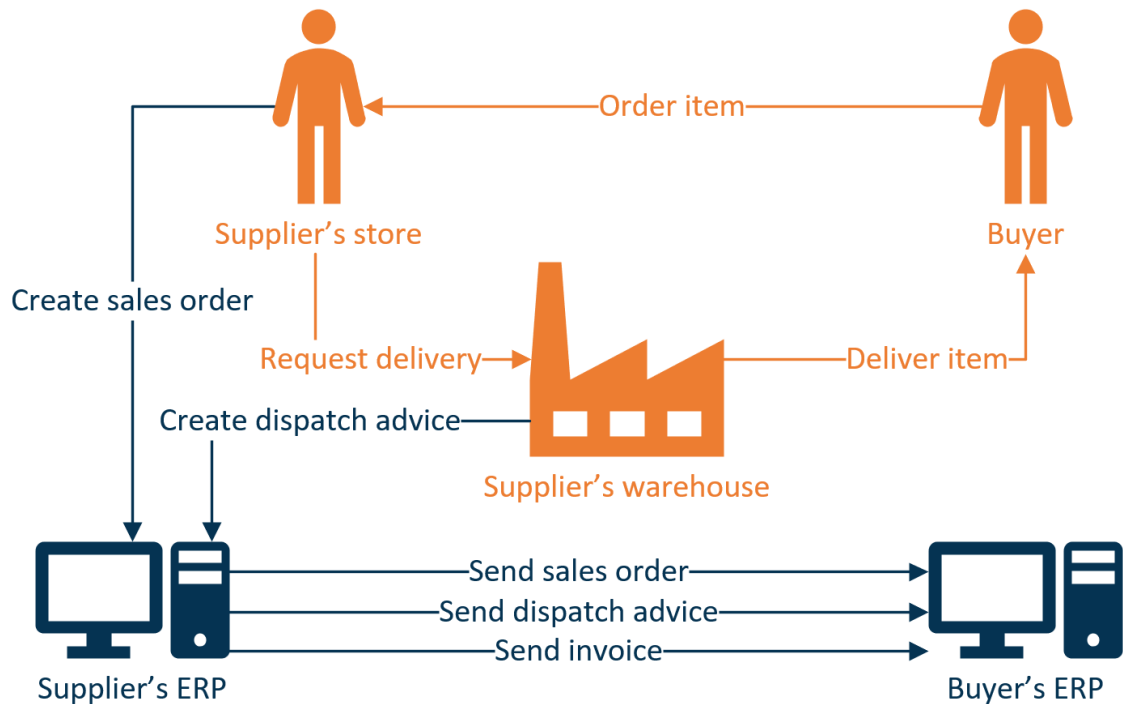


Figure 17: Purchasing process

In this business process, the supplier creates a sales order about the purchasing event to supplier's ERP system and the supplier's warehouse creates a dispatch advice when the item is shipped from the warehouse. Both transactions are sent from the supplier's ERP system to the buyer's ERP system. After the item is delivered, the supplier bills the buyer by sending an invoice. In this example, buyer's ERP system is handling purchase transactions and each invoice must match to a purchase order. Before a payment will happen, the buyer's ERP system needs to know that the item was received. If this business process would be automatized, the purchase order details, such as the purchase order number, would be needed from the buyer's ERP to the supplier's ERP before the dispatch advice and invoice is sent. Figure 18 presents a solution for this business process from ERP systems' point of view.

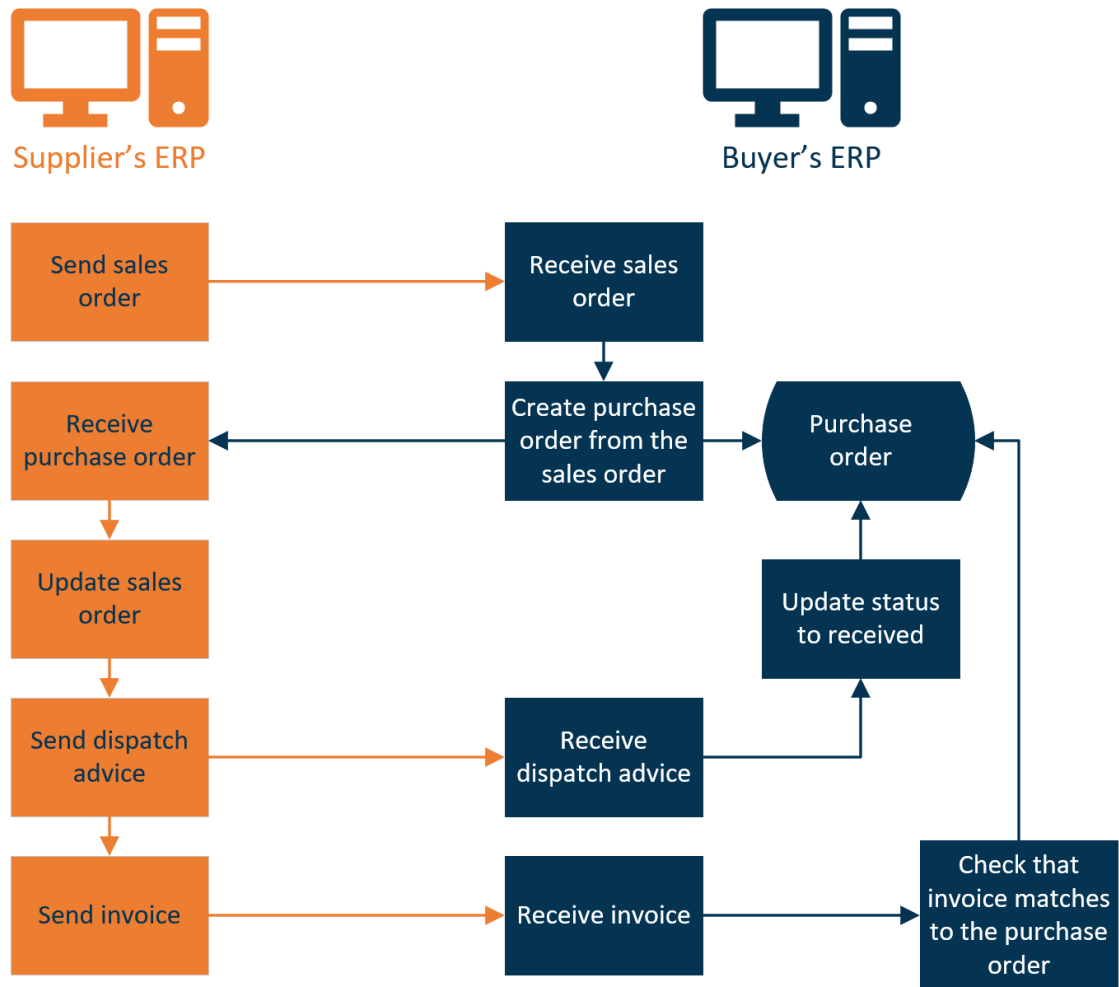


Figure 18: Purchasing process between buyer's and supplier's ERP systems

In this case, the supplier's ERP system triggers the order process by sending a sales order to the buyer's ERP system. That sales order creates a purchase order to the buyer's ERP system. The purchase order updates the sales order's details and therefore necessary data will flow from the sales order to the dispatch advice and lastly to the invoice, so that all following processes can be handled.

The integration design decisions are based on the interactions of systems (Lobaziewicz 2015). For that reason, it is helpful to understand at least in high-level how sender and recipient systems process the messages to model a data integration. Gartner (2016) highlights that companies should seek for an integration brokerage who has industry and ERP experience. Understanding the systems helps to identify the requirements of the data integration solution to support the process presented in figures 17 and 18 (Gartner 2016). For example, if the supplier's ERP system does not support this kind of process, the data integration solution can fulfill the limitations of the systems. This example highlights the importance of understanding the real-world transactions and their sequence,

as well as the sender and recipient systems, in order to be able to support the business requirements with the data integration solution. (Giordano 2010; Reeve 2013)

There are also processes beyond the data integration that still affects to the overall solution. For example, how does the buyer's ERP system know what item has been ordered if the supplier's sales order contains only item codes assigned by the supplier, but the buyer uses their own item codes? This could mean that master data related to item codes is updated and maintained in the buyer's ERP system, linking each supplier item code to corresponding buyer item code. Even though this kind of processes are typically handled outside of the data integration solution, they are nevertheless topics that needs to be covered for the data integration solution to work. Even if the data integration functions as it should, the overall business goal is not achieved if all subprocesses are not completed, regardless whether they are part of the data integration solution or not.

3.3 Data flow diagram

Giordano (2010) mentions data flow diagram (DFD) as one of the data modeling methods. DFD is commonly used visualization method to model data processing (Yang 2014; Tillmann 2017; Zhang et al. 2018) and it is a graphical logical process modeling technique (Yang 2014; Tillmann 2017). DFD illustrates the information flows and their directions (Fountas et al. 2006). In DFD, informal and formal graphical notations are used to explain the process and functions are simplified (Yang 2014; Zhang et al. 2018) and DFD portrays entities and their relationships (Appavuraj et al. 2014).

According to Zhang et al. (2018), DFD contains three main elements: data stream, data entrance, and data processing unit. According to Fountas et al. (2006), Yang (2014) and Tillmann (2017), DFD contains data flows, processes, data stores and external entities. Data streams or flows are pipelines of information flow and data entrances or external entities are objects outside the system with which the system communicates (Fountas et al. 2006; Yang 2014; Zhang et al. 2018; Tillmann 2017). In data integrations, external entity is the source or destination of data. Process or data processing unit transforms the incoming data flow to the outgoing data flow (Fountas et al. 2006; Yang 2014; Tillmann 2017; Zhang et al. 2018). Data stores are repositories of data in a system, illustrating where the data is stored (Fountas et al. 2006; Yang 2014; Tillmann 2017), such as a file in an FTP server.

The highest level of a DFD is called level 0 that describes the entire system (Appavuraj et al. 2014; Yang 2014; Tillmann 2017). According to Yang (2014), the level 0 DFD has only one process that presents the main function of the system, which is in this research

context a data integration solution. A level 0 DFD presenting order and order response data integration between a customer and a supplier is presented in figure 19.

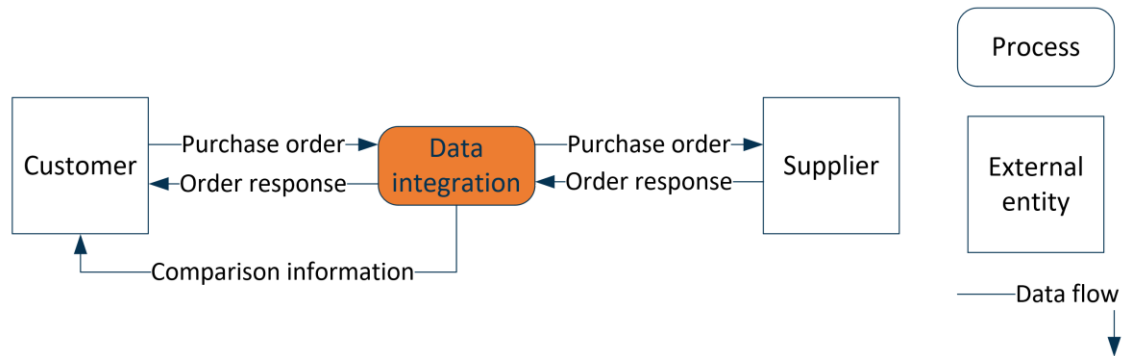


Figure 19: Level 0 DFD of order & order response data integration

This DFD presents a solution, where a customer sends a purchase order to a supplier and the supplier confirms the order by sending an order response back to the customer. The data integration also compares the purchase order and order response and sends comparison information to the customer. As Yang (2014) and Tillmann (2017) states, the level 0 DFD provides only high-level presentation. Therefore, for example, connectivity protocols and message formats are not presented, as well as subprocesses within the data integration solution are abstracted into one process.

Level 1, the next level diagram, describes the main functions of the system (Yang 2014; Tillmann 2017). According to Yang (2014), Level 1 DFD is also called functional-level DFD. It describes the subprocesses of each Level 0 process (Tillmann 2017). Same example from figure 19 is presented in a level 1 DFD in figure 20.

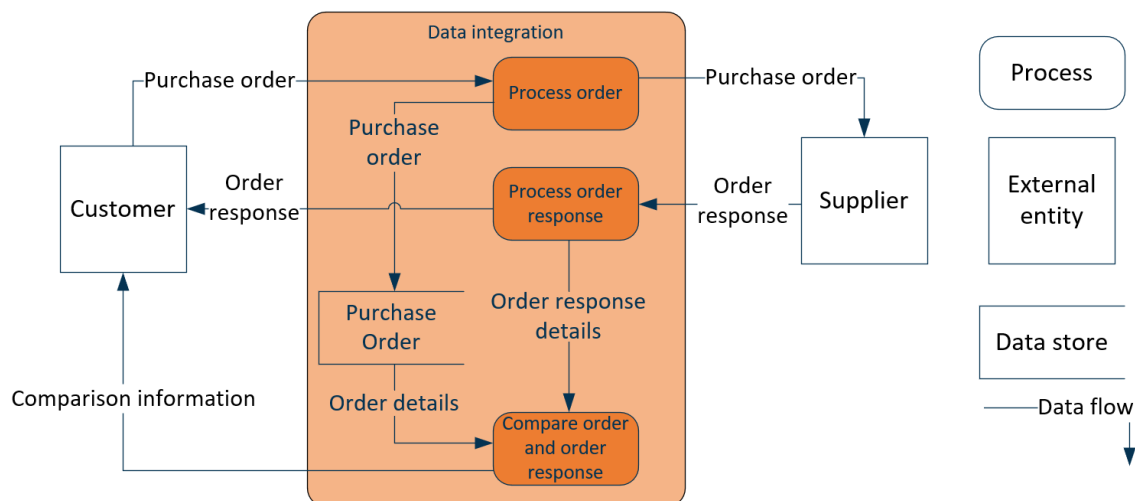


Figure 20: Level 1 DFD of order & order response data integration

The level 1 DFD still has the same external entities and data flows between them, but it presents subprocesses, too (Tillmann 2017). The purchase order data is stored and used

to compare the data between order and order response to be able to send comparison information to the customer. Level 1 DFD can be once again disintegrated to multiple level 2 DFDs, and so on (Appavuraj et al. 2014; Tillmann 2017), where each new level provides more detail level information of certain process presented in previous level.

3.4 Component integration model

While DFD describes only the direction of data flows, it doesn't define how interfaces communicate with each other. DFD does not illustrate whether the recipient fetches files from sender's server or if the sender sends the files to recipient's server. In data integration solutions, it is important to know what components are included in the data integration solution and who is the active party in data transfer (Xia et al. 2000).

UML component diagram is used to describe the architectural model of a solution (Mokarat & Vatanawood 2013). Component-based design allows to identify what separate components are needed to create a solution (Giordano 2010). The component integration model (CIM) represents the communication of interfaces as well as the components that are part of the solution (Xia et al. 2000). Components describe independent units of functionality which communicates with other components (Coulson et al. 2008). CIM is a graphical presentation of involved systems and their interfaces. CIM defines the system components that are integrated, and it illustrates how the systems, or their components communicate. Component modeling method can be also used to illustrate how new components can be added into existing solution (Xia et al. 2000; Coulson et al. 2008).

As Holy et al. (2012) mentions, solutions can easily have hundreds or thousands of components that are connected to each other. Therefore, the CIM becomes quickly too complex to be explored by humans. To address this issue, similar with DFD, CIM can also be presented in different levels. Level 0 CIM is a high-level model where the solution in scope is presented in the center of the diagram and other components integrated are around it. Level 1 CIM reveals the solution in scope, illustrating the internal components of the solution on a high-level. Figure 21 presents a level 0 and level 1 CIM of the data integration solution described in previous chapter 3.3.

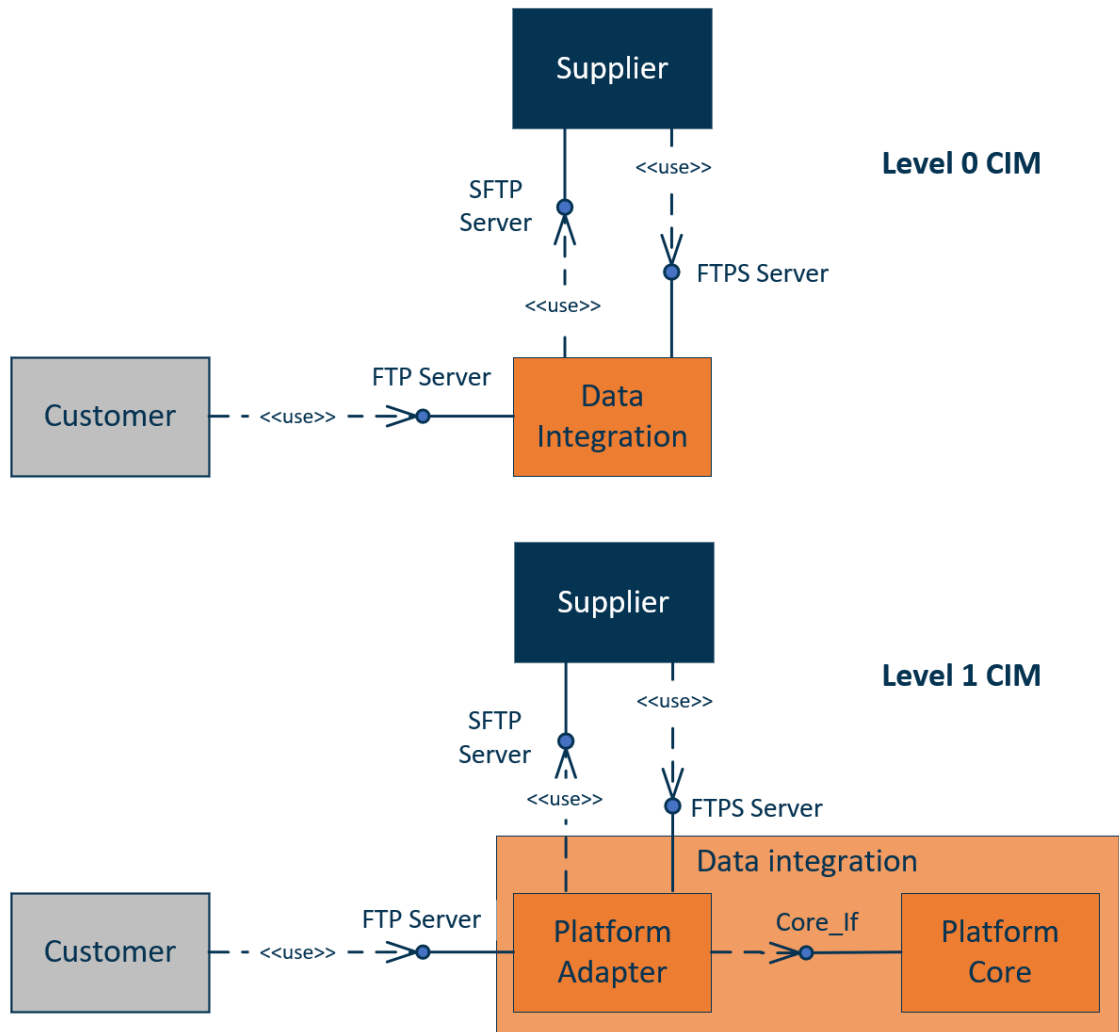


Figure 21: Level 0 & 1 CIM presenting a data integration solution

This solution transfers data between a supplier and its customer. CIM diagrams does not describe the direction of data flows, but instead, it illustrates the interfaces and active parties. Customer connects to FTP server provided by the data integration solution. Supplier and the data integration are connected via SFTP and FTPS connectivity, where the first one is hosted by the supplier and the latter one is hosted by the data integration. In practice this could mean, for example, that the supplier is fetching orders from the FTPS server that is hosted by the data integration solution and sending order responses back to supplier's SFTP server, from where the data integration solution downloads the data.

Level 1 CIM gives more information about the solution, revealing how the solution in scope works. It describes the integration of internal components of the solution. In figure 21, data integration solution has a platform adapter that handles the connectivity between external parties, and it uses the interface provided by the platform core. In practice, this means that there is an adapter component that can provide a connectivity to external networks, while the platform core is available only in internal network. All routing

logics, mapping processors, and other needed components are illustrated as one platform core component, which could be once again revealed in level 2 CIM to illustrate more details how internal components are connected.

It is important to understand that the direction of arrows does not define the direction of data flows, but instead, it just illustrates who is the active party. This is one of the reasons why CIM is not suitable modeling method to illustrate a solution for all stakeholders, especially if there are no other diagrams used. Business might easily confuse the direction of arrows to the direction of data flows, assuming from the figure 21 that data from supplier to data integration is sent via FTPS.

CIM can be used to illustrate what components are involved in the solution and how they are interacting with other components to provide information of how to utilize existing components to plug-in new components. If the supplier and the customer would want to transfer some new data between them in addition to previous solution presented in previous figures, existing connectivity could be used by plugging in new components. For example, if the supplier would want to send shipment information, such as dispatch advices, to the customer, new component could be added to the platform core to support this. This way the supplier would still connect to the data integration solution either via SFTP or FTPS and the data integration solution would transfer the data to the customer via existing FTP, meaning that existing components would be utilized in new projects, too. (Xia et al. 2000; Coulson et al. 2008)

3.5 Sequence diagram

Sequence diagram is also one method to model message flows between objects or components of a system. It can be used to design as well as to document the solution architecture. Sequence diagram illustrates the sequence of processes showing what is happening during the time. It also describes what is happening after one message flow is completed and therefore, it shows the relations of message flows and interaction of processes. (Grgec & Mužar 2007; Baqais & Alshayeb 2018) The diagram illustrates time that progresses from up to down and presents objects in horizontal dimension (Grgec & Mužar 2007; Ravindran et al. 2010).

Sequence diagram contains different notions, such as objects, lifelines, execution specifications, object communication, and interaction frames (Grgec & Mužar 2007). Object is an instance of a concept (Grgec & Mužar 2007), such as a system or a server. It is illustrated with rectangle (Grgec & Mužar 2007). Lifeline is presented as a vertical dashed line and it describes the period of the existence of the participant. Execution specification

is drawn as a tall, thin rectangle and it illustrates the timeframe within the objective is performing some behavior. Communication of objects is drawn as arrows and they describe the interactions (Grgec & Mužar 2007), such as messages transferred via data integration solution. Sequence diagram can also have interaction frames describing different logics. Each interaction frame contains operators, such as alternative choice, optional process, or looping logic. (Grgec & Mužar 2007) Example sequence diagram describing shipping notice data integration is presented in figure 22.

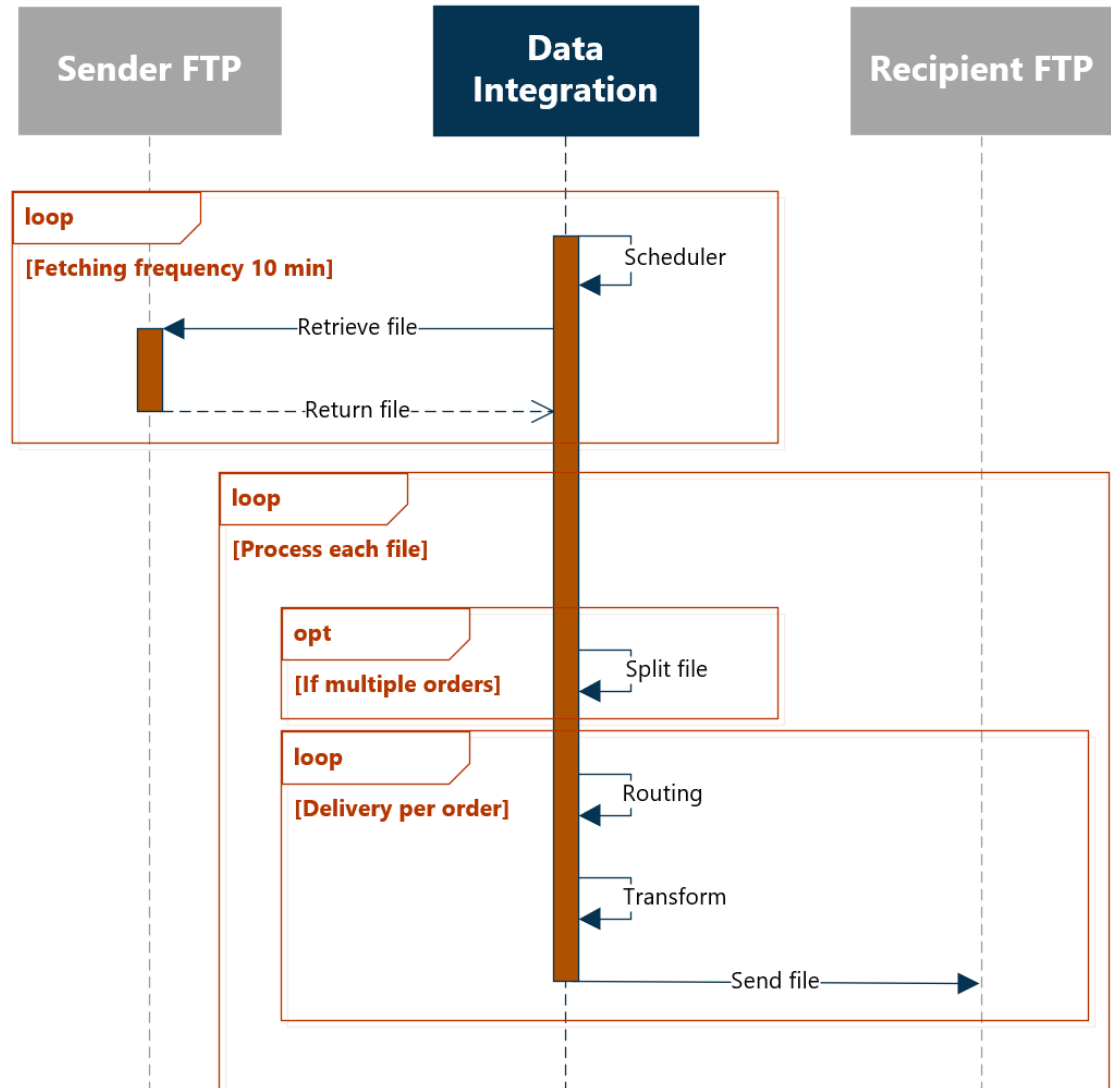


Figure 22: Example sequence diagram

Figure 22 presents a data integration solution, where sender informs the recipient what has been shipped, so that the recipient knows what is delivered and what goods to expect to arrive. Sender is creating output files on delivery basis, meaning that each file can contain multiple orders, but recipient needs to receive order-based delivery, meaning that each file can contain only deliveries of one order. Files are transferred from sender's FTP server to recipient's FTP server. Scheduler is implemented to fetch files from the

sender's FTP server once every ten minutes. Each file that was fetched is processed, so there is a loop operator. Each file can contain multiple orders, so optional logic is needed to split the files in cases where there are multiple orders within a file. This way it is possible to start process files that contain only deliveries of one order. Recipient of order specific delivery message is identified with routing logic and transformed from source format to target format, and finally sent to recipient's FTP server. Processes after that are not presented in the figure, which is why the object does not have an execution specification in the figure 22. (Grgec & Mužar 2007)

Grgec & Mužar (2007) states that sequence diagrams can be used to describe complex solutions, but Baqais & Alshayeb (2018) claims that sequence diagram is not suitable method to describe complex systems. Even though especially the operators are useful to illustrate how the solution works, such as if-else logics or repeated processes, sequence diagrams can be easily become too complex. As we can see from the figure 22, even though the overall process is quite simple, it still has many details. According to Baqais & Alshayeb (2018), sequence diagram is typically small and represents only one scenario to avoid it having too extensive details. Therefore, it is important to decide what to represent in the diagram and what to leave out, if the sequence diagram is selected as a modeling method.

3.6 Flowcharts

Since a data integration solution transfers different message formats from source to target destinations, it is crucial to know what message formats are used and what processing is needed between the source and the target destinations (Lenzerini 2002). The data integration solution needs to identify the inbound document type to process correct transformation and mapping for the file (Gleghorn 2005). Sequence of processes are important to define in which phase certain outcome is expected, such as will the data validator be implemented to validate the inbound or outbound data. Sequence diagram becomes quickly too complex (Baqais & Alshayeb 2018), so other methods are needed to demonstrate the solution overview in abstract level that is easy to understand.

Activity diagram is commonly used UML diagram in software development projects. It can be used to describe what activities are included in the process and who is responsible of each activity. (Eriksson & Penker 2000) UML activity diagram can also be used to model sequence of processes to visualize how the system functions (Chen et al. 2018). According to Li et al. (2018), activity diagram illustrates the behavior of the system in scope and it shows how a combination of actions are processed. Also, dependencies of processes can be visualized in activity diagram, such as decisions and choices (Chen et

al. 2018). Process flow diagram contains many similar elements compared to UML activity diagram. Process flow diagram represents a system and the processes, input and output data, and data flows. (Meier & Meier 2011; Appavuraj et al. 2014) The process flow diagram graphically presents key elements of a process and it can be used in early development of complex processes (Meier & Meier 2011), such as when clarifying requirements (Appavuraj et al. 2014). Like many other models, it does not describe details of the processes, but rather gives an overall view of a solution (Meier & Meier 2011; Appavuraj et al. 2014).

Li et al. (2018) proposes to combine different diagrams, such as UML activity diagram and use-case diagram to extended diagrams. This way it is possible to model the solution in a way that is both easy to understand and contains enough high-level information what the solution is doing. Gleghorn (2005) has illustrated data integrations for managers using this kind of method. Following figure utilizes different modeling techniques to represent a sales invoice solution with control messages as a cross-functional flowchart in figure 23.

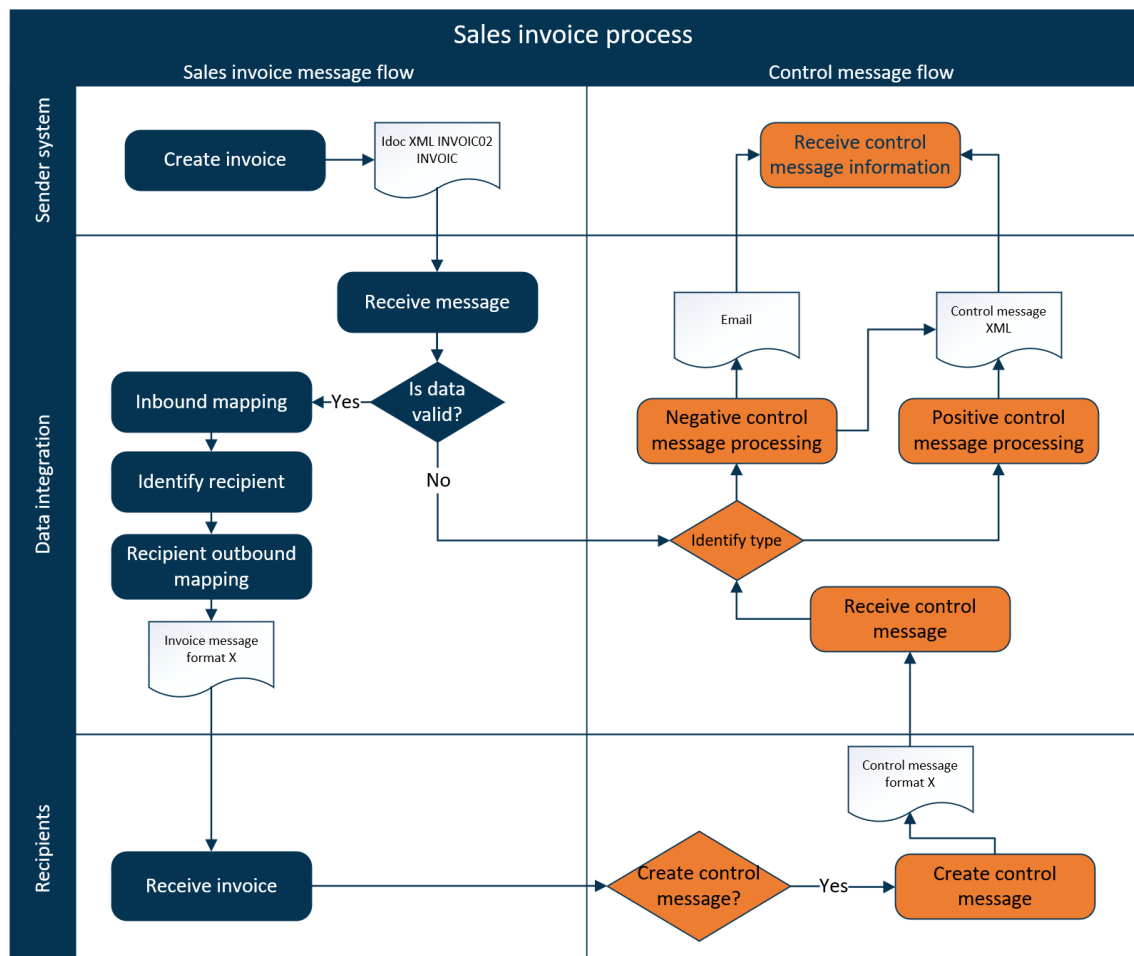


Figure 23: Solution overview of sales invoice process

Figure 23 presents a sales invoice process where invoice is sent from the invoice issuer to the invoice recipient. Invoice is sent as Idoc XML message format and sent to recipients according the message format which the recipient uses. This solution contains also control messages to inform whether the invoice was processed successfully or not. Control messages are optionally created after data validation and by the invoice recipient. Since there are multiple invoice recipients, control messages are implemented only with certain recipients and message formats are party specific. Invoice sender receives control message either only as a control message XML or also via email, depending whether the invoice was successfully processed or not.

As Appavuraj et al. (2014) states, process flow diagram presents a clear understanding of all the processes required for the intended outcome. Even though each process has multiple subprocesses, the business stakeholders does not need to understand how the solution performs necessary tasks and subprocesses. For that reason, the processes are abstracted to visualize the overall solution in an easily understandable way (Meier & Meier 2011; Appavuraj et al. 2014). According to Yu et al. (2007), cross-functional flowchart can be used to illustrate how process activities are performed and by who. Each swimlane illustrates an entity, such as participants involved in a data integration. Every activity related to that participant is presented with different symbols. (Yu et al. 2007) This way it is possible to easily model what processes are needed to fulfill the requirements of a data integration.

4. RESEARCH METHOD AND RESEARCH SETUP

4.1 Introducing the case company

Liaison Technologies, that was acquired by OpenText™ in 2018 (Opentext 2018b), provides data integration and data management solutions for several industries (Opentext+Liaison 2019). Liaison Technologies has been listed by Gartner (2015) as one of integration brokerage provider. The case company provides data integration solutions using different platforms and technologies, such as ALLOY™ Platform as a dPaaS and Contivo Analyst™ for data mapping and transformation (Gartner 2016; Opentext+Liaison 2019). The case company operates globally, and Liaison Technologies Oy is the official company registered in Finland (YTJ 2019). The annual revenue of Liaison Technologies Oy in 2017 was roughly 20 million euros and it employed 120 people (Asiakastieto 2019; Finder 2019).

The case company executes several size of projects and therefore, the needed resources and expertise between different stakeholders varies between projects. Different stakeholders are participating to solution planning and development, such as business and IT people from customer's side and project managers, architects and technical developers from the case company. Regardless of the project, typically they have same basic principles and the projects start by capturing the business needs and end when the technical solution has been implemented and the functionality is assured after go-live. This means that projects typically contain project management, solution design, and technical development, testing and implementation. Since the company provides solutions to its customer using the case company's platforms, it also offers maintenance and customer support for existing solutions as Managed Services (Opentext+Liaison 2019).

The case company was a suitable subject of research, because the case company provides data integration solutions for different industries and business processes using various technologies (Opentext+Liaison 2019). Therefore, it was possible to research data integration modeling in broader context and not to limit the research, for example, to study only order-to-cash EDI integrations or eInvoice solutions. The case company has expertise of designing and implementing data integrations and therefore, the people working in the case company are considered as experts of data integration modeling. The people in the case company are also eager to learn more about best practices of designing data integration solutions, which gave a purpose of executing this research.

Following subchapters describes what methods were used and how the empirical research was made.

4.2 Research process

Academic articles, researches and other material related to the research topic were searched from different databases with various keywords. This literature review created the theory background for this thesis. In addition to academic literature, it is also important to notice how organizations in the industry define different terms and explain the processes, which is why information from different organization's reports and websites were also used. The reliability of the results was secured by evaluating the year of publication and number of citations. The purpose of the literature review was to gather knowledge and understanding about the research topic.

The material for the empirical research was collected with structured interviews. The research objective was not to only know how people acts, but also understand the reasons for their decisions. According to Saunders et al. (2016, p. 392), structured interview allows to identify general patterns and it supports a deductive approach to find out how the theory links to practice. In structured interviews, there are questionnaire that the interviewer goes through with the interviewee. The questions are predetermined before the interview and the questions are asked just like they are written. This allows to get answers to same questions from all interviews. Structured interview allows to gain material towards certain phenomenon which was the reason to choose it as a data collection method. (Saunders et al. 2016, pp. 391-392)

The questionnaire contains three types of questions. First set of questions is related to data integration modeling and planning process in general. Second set of questions is about solution design. Solution design is a document type that the case company uses in data integration solutions, which was the reason to use this term in the interviews. Last topic in the interviews was related to data models and mappings, which is one important subtopic in data integration solutions (Lenzerini 2002; Amano et al. 2014).

The purpose of the interview was to understand how the people in the case company see things and what are the reasons behind their actions, to test the theory in the case company's context. This allowed to gain understanding how the data integration modeling could be utilized in the case company. The interviews were held internally in the case company with people who are working with customer projects and solutions. It was not possible to collect data from all persons working in the case company, so samples were selected. As Saunders et al. (2016, p. 274) states, sampling is suitable method when

there are budget and time constraints, which was the case in this research. According to Saunders et al. (2016, p. 275), the target population refers to the population that are wanted to study. Sample was selected from the target population with purposive sampling, because it allowed to ensure that the selected sample enables to meet the research goals (Saunders et al. 2016, p. 301). The persons were selected from different roles to represent the target population. The sample consists of 5 architects, 4 project managers and 4 technical developers. The names of the interviewees are not mentioned and therefore, numbers were assigned to each interviewee when the analyzation was started. Each interviewee is referred as IX to ensure the transparency of the research process and material. It should be noted that the numbers are not sorted either by the name or the sequence of interview events. This ensures the confidentiality since only the researcher knows who said what. The roles are mentioned in direct citations to provide better understanding about the perspective of the interviewee.

After the interviews were done, the collected material was analyzed. After the analyze was completed, the results were compared to existing literature to find out how the theory and the empirical research results fits together in the research context. After this, conclusions were made to answer the research questions and practical solutions are provided.

4.3 Empirical study

Empirical material was collected with structured interviews. Interviews allowed to understand how people working in the case company thinks about the topic. Interview questions were created based on the research objectives, because the objective of interviews was to gain material to answer the research questions. Interview questions were divided into different themes to make sure that the interview will flow fluently, and to ensure that all topics will be covered. The goal of the questions was to get material from the interviewees about their opinions and decisions related to data integration modeling.

After the first version of the interview questions was ready, the questions were reviewed by the research examiner and the research supervisor in the case company. The purpose of the review was to ensure that the answers will give information related to the research questions. Another purpose of the review was to make sure that all questions are neutral to prevent the interviewer to guide the interviewees to certain type of answers, as well as to make sure that the interviews are possible to be held during the time that was reserved for each interview. The interviews were held in Finnish, because all interviewees as well as the interviewer are Finns. For that reason, the questionnaire was made in Finnish, too.

The first interview was made separately from other interviews to test how the questionnaire works in practice and to see how much time is needed to go it through. Another reason was to ensure that there is time to modify the questionnaire towards more suitable version before other interviews in case of any changes are needed. After first interview, two minor modifications were made. First question was changed from “Who are you and what kind of tasks does your role contain?” to “Introduce yourself and your job”, because this was more fluent way to start the interview with interviewees whose name and role was already known. Fifth question was changed from “What kind of methods do you use to model a solution?” to “Describe a typical planning process. Where do you start from and in what order do you perform things?”. This was changed because the “modeling” word was too abstract and unclear, and the latter question gives answers regarding the modeling process that was one important topic to be covered. The final questionnaire is presented in appendix A.

Target population for this research were the people who are involved in customer projects and who participates to the data integration planning process. There are several titles in the case company, which is why the sample was selected based on the role instead of a title. Target population for this research were project managers, architects and technical developers since those are typical roles involved in the case company’s projects and customer solutions. Sample was selected from target population with the managers of each group. The objective was to select a sample that would reflect to the behavior of the entire target population (Saunders et al. 2016, p. 274). For that reason, people with different experience was selected, such as people with very strong knowledge about data integrations as well as people who do not have so long work experience. Since the focus was also to gain general understanding about data integration modeling, people working with different kind of solutions were selected. For that reason, people who are mainly involved with very complex and non-standardized custom A2A integrations as well as people who are doing standardized B2B EDI integrations, were interviewed.

After the potential candidates were selected, each interviewee was contacted first either by phone call, email, or face-to-face conversation. All potential candidates accepted the invitation for the interview. The interview questions were sent to them via email before the interview. The purpose of this was to give them time to go through the topics and think their answers in advance to achieve diverse perspective and better-quality answers during the interview. Another reason for sending the questionnaire in advance to the interviewees was to reduce the time needed for the interview event.

Total amount of interviews was 13 and the interviews were executed either as face-to-face conversations or via teleconference depending on the physical locations of the interviewee and the interviewer. 10 interviews were executed face-to-face and 3 interviews were carried out via teleconference. Interview length varied between 35 and 80 minutes and the average interview length was 51 minutes.

The interview event was started by introducing the research topic and research goals. It was also mentioned to the interviewee that all their answers are treated confidentially. The name of the interviewee, the interview location and date, and the total time of the interview was recorded. All interviews started with an opening question where the interviewee told about him or herself. After the opening question, the interview questions were executed according the listed sequence of interview questions. Additional questions and explanations were made by the interviewer depending on the answers to keep the conversation flowing. Notes were written down during the interview and the answers were transcribed after the interview based on the notes. All notes as well as transcribes were done in Finnish. Following chapter describes how the results were analyzed.

4.4 Analyzing the research results

After all interviews were completed and transcribed in Finnish, the results were analyzed in Finnish. Only direct quotations that were selected for this paper were translated from Finnish to English. Saunders et al. (2016, p. 571) states that the data analysis allows to identify important patterns and themes from the data. Analysis should be started right after the interviews to get most value from the collected data and the data collection moment should also be part of the data analysis (Saunders et al. 2016, pp. 571-572). For that reason, the analyzing process was started during the first interview and the process continued during the time period within the interviews were held.

The results were analyzed using thematic analysis approach. According to Saunders et al. (2016, p. 579), the purpose of thematic analysis is to identify occurring themes or patterns from the collected data. Thematic analysis was suitable method due the deductive approach to easily identify certain patterns from the data and link these to existing theory (Saunders et al. 2016, p. 579).

First the transcriptions were combined and analyzed on a question level. Occurring themes and other markable findings were identified from the answers of a certain question. This allowed to categorize data that had similar meaning (Saunders et al. 2016, p. 580). After that, similar themes were searched from different questions to see if same key results are identified from answers related to different questions (Saunders et al.

2016, p. 589). During this process, the number of categories was expanding continuously. Finally, after all material was categorized, those categories were combined. These categories created the structure for following chapter 5.

After the key results were identified from the interview materials, they were combined to existing theory. Thematic analysis was used also in this phase. The goal was to identify occurring themes or patterns between the existing literature and the empirical research results (Saunders et al. 2016, p. 579) to provide practical solutions according the research philosophy.

5. KEY RESULTS

5.1 Reasons to design and document a solution

When asked about the reasons why a solution should be modeled and documented, few occurring answers were identified. In general, all interviewees saw the planning as a mandatory phase to ensure that everybody has a clear vision what will be done. Everyone thought that a solution design is used to explain to all stakeholders, such as the customer and the project team, what kind of a solution will be done. The documentation should be updated if some changes have been made during the development phase, which means that the documentation should provide information afterwards about what has been done (I1; I2; I5; I6; I7; I8; I10; I11; I12; I13). As one argued, usually it can be seen from the platform what has been done, but the solution design can tell *why* the implementation has been done in certain way (I1). Other stated that the solution design should provide information what has been done without looking the solution from the platform (I12). Sometimes the solution can contain business logics that are necessary even though the technical solution could be against best practices and wrong in another cases. For example, a solution was implemented to match each invoice to corresponding order if both messages had only one row. Even though the information of invoice row is not always same than the order row it refers to, meaning that the solution is not actually working as some would think it should, this was explicit requirement from the business because that solution allowed to automatize some business processes beyond the integration. (I2)

Almost all interviewees mentioned that planning is important to ensure that customer needs are identified correctly. Planning is often made together with customer, which means that planning is needed to capture customer needs together with the customer to get a mutual understanding about the solution. (I1; I2; I3; I5; I6; I7; I8; I9; I10; I11; I12; I13) Some thought that planning is important phase to create a document which the customer can approve before the development is started (I2; I6; I10; I12). That document can be used to make sure that the customer knows what they want, as well as it can be used afterwards to check what has been agreed in case of any disagreement between different stakeholders (I2; I7; I8; I10).

“Planning is needed to make sure that the customer knows what they want and to ensure that the customer will get what they have ordered” – Project manager 2

“The solution design provides general understanding to the customer what the solution is doing” – Architect 2

Almost everyone mentioned that a solution design is needed for smoother development. The plan is used as a guideline for the development, telling the developers what the solution needs to do. (I1; I2; I3; I4; I6; I7; I8; I9; I10; I12) Some argued that the solution design should define in high-level how the solution functions (I2; I3; I5; I9; I10; I11) while others thought that the solution design should have more detail level information about the solution (I1; I4; I6; I7; I8; I12; I13). Some people thought that the solution design should explain *what* the solution is doing and *why* but not give answer towards the question of *how* (I1; I5; I9; I10; I11). Few people thought that the solution design should explain both *what* the solution should do as well as define *how* the solution needs to be implemented, which means that it should contain more technical details (I4; I8). Some interviewees thought that the best detail level is somewhere between those extremities, which means that the solution design should act as a guideline for the developers, but it should leave some questions open for the developer to decide (I6; I7; I12; I13).

“The plan tells developers what needs to be done” – Architect 1

The detail level of a solution design can vary between solutions, since complex solutions need more details compared to standard solutions (I2; I10). As one interviewee highlighted, if the solution is simple and similar with previous solutions that the case company has provided for a long time, there is no need to explain each detail how the solution will be implemented. If it provides the customer an explanation about what the solution is doing, it is enough. (I10)

“The solution design provides a technical documentation for the customer about the solution” – Technical developer 3

Planning phase allows to identify possible problems in the early phase, which means that issues are solved before the development has even started. (I2; I3; I8; I10; I12; I13) Also, questions related to solution are identified and resolved, which means that the developers will have less questions while developing a solution (I4; I8; I9). Most of the project managers highlighted that a plan is needed to ensure that the project is delivered on time and within the allocated budget and resources (I3; I8; I11).

“The planning phase allows to think about how something can be implemented” – Technical developer 4

Some of the interviewees mentioned that planning is important to ensure the quality and maintainability of a solution (I1; I6; I10). Even though the goals can be accomplished with multiple different solutions, some solutions are easier to maintain as well as the overall

architecture is simply less complex compared to other possibilities. it is easier to think at the planning phase how something can be done with good quality and with best practices. (I1; I6; I10)

“Things can be done in many ways, but other ways are clearer throughout the life cycle of integration” – Technical developer 1

Some interviewees mentioned that the solution design should provide necessary information for technical developers who have not been involved in the design phase (I7; I8; I12). If the solution design tells the technical developers what needs to be done, he/she does not need to gather information from other places (I4; I8), such as trying to capture important information from emails, because all important information is already acquired and listed in the solution design (I8). This will make the implementation phase much smoother and faster and allows to split the implementation phase to multiple developers (I8).

“If the first question in the development phase is ‘what do we do?’, then something wrong has happened at the design phase” – Project manager 2

The baseline is that the solution design should not describe internal technical solutions how the platform is working, because that document is delivered to the customer and therefore, it could end up to competitors (I2). Things that will probably change during the lifecycle of a data integration, such as contract related things (I2) or mappings (I1; I4; I10; I12), should be left out from the solution design, even though some (I6; I13) thought that mappings should be mentioned in the solution design. If all details are mentioned in the solution design, it needs to be constantly updated when even minor changes are done (I1; I4) as well as the size of a document expands too much (I10).

5.2 The importance of business processes

When asked about what information is needed to design a solution, eleven out of thirteen interviewees mentioned that it is crucial that involved people, both customer and the people in the case company, knows what will be done and what are the reasons of developing a data integration. Knowing the data integration’s scope itself is not enough, but it is also important to understand business processes behind the technical solution (I1; I2; I3; I5; I6; I7; I8; I9; I10; I12; I13).

“Data integration is part of a business process and the solution needs to support that business process” – Architect 5

“If the developer or the designer does not understand the business process, the solution cannot be planned correctly” – Project manager 2

“Not a single integration can be made well if the business processes are not known” – Architect 1

There are commonly known standard solutions that the case company has expertise about (I2; I7), such as an order process in supply chain or order-to-cash process (I2). In this kind of cases, the information needed from other stakeholders, such as the customer or its trading partners, is less important because the case company is already familiar with that kind of solutions (I2; I7). Therefore, based on the case company’s expertise, possible limitations and needs are known by default at least in high-level (I2; I7). However, even in pretty standardized solutions, it is crucial to know the business processes because they can set restrictions and special needs to the solution (I8; I10; I12).

One interviewee (I10) mentioned an example case where the customer had multiple business units that were receiving messages from their suppliers. The solution was made in a way that all messages were sent from suppliers to the customer via one interface in the case company’s platform. The customer however had to receive messages to different subfolders in the customer’s FTPS server depending on the business unit, so the business process had a clear impact towards the overall solution. Another example was a case where the data integration should have routed messages to certain freight forwarder based on the price list which would have been maintained in the data integration layer. This would have allowed to select cheapest trading partner for certain delivery, which once again highlights how the business processes can affect to data integration’s functions and requirements. (I10)

“Objectives of an integration comes from business processes” – Technical developer 4

“It is crucial to understand the business processes so that right decisions can be made” – Architect 3

“The better we know the business processes, the easier the integration is to implement” – Technical developer 3

It was very common opinion that the data integration needs to support business processes. Thus, understanding business processes is very important to know what is happening in the real life. (I1; I2; I3; I5; I6; I7; I8; I9; I10; I12; I13) It is not enough to know what messages needs to be flowing to which direction (I1; I2; I3; I5; I6; I7; I8; I9; I10; I12; I13), even though few people (I4; I11) thought it is enough.

“We need to know why some message is flowing from place A to place B, we need to know what is happening when that message is sent” – Project manager 4

However, many interviewees also mentioned that it is the customer who should be aware about the business processes. Knowing the business processes helps to design a solution, but after all, the customer should be responsible of defining what are the business needs and exceptions that needs to be taken into consideration in the data integration solution. (I1; I2; I3; I6; I7; I8; I10; I11; I12; I13)

“The customer must know the business requirements” – Project manager 2

One interviewee mentioned an example case, where a financial statement can be removed with a message and even the business stakeholders did not know what the business process is where this scene can occur (I2). This highlights the fact that all business processes might not be known while designing a solution, since even the business might not know all business processes and exceptions. Even though it was thought as a helpful thing to understand the business, all business cases and needs are hard to identify by the people in the case company. Thus, input from customer is needed. (I2; I8; I11)

5.3 Understanding technical boundaries and limitations

Common opinion was that it is not *necessary* to understand the source and target systems that are integrated if the processes and interfaces are known. However, many also highlighted that understanding the source and target systems at least in some level is helpful (I2; I4; I5; I6; I7; I8; I9; I10; I12; I13). One benefit of understanding the systems was related to communication (I2; I4). Therefore, knowing how the systems are working helps to find common terms with other stakeholders, such as with the customer or other trading partners (I1; I2; I4).

“Customer often speaks from their system’s point of view” – Technical developer 2

“If we know e.g. how SAP works, it is much easier to integrate to SAP and communicate with the customer if we know the system that is in use” – Architect 1

People had dissenting opinions about the *importance* of understanding the systems that are integrated. Some thought that it is crucial to understand what kind of limitations the systems have and how it functions (I1; I6; I12; I13) while others thought that it is enough to know in high-level what precondition the systems sets to the data integration (I5; I7; I8; I9; I10).

“It is enough to know the interfaces” – Technical developer 2

“Knowing the business process is more important than knowing the systems” – Technical developer 3

“If the interfaces are well defined, it is not mandatory to understand how the system functions” – Architect 2

“It can be useful to understand the system, because that might affect to decisions where some functionality is wise to be made” – Technical developer 1

“It is important to understand the systems, because there could be technical restrictions that prevents the process from working” – Architect 3

As one interviewee summarized, there are business processes, processes in the systems, and on top of those, the data integration level (I8). For example, if a system cannot handle change orders, it sets restrictions to the data integration solution (I6; I8; I12). The reason why people had different opinions about the importance of understanding the systems that are integrated is highly related to the question of *who* should have the understanding.

“If the customer knows their system, we do not need to know” – Architect 4

Since people had different experience, the opinions varied. People who had worked more with business thought it is more important to know how the systems functions. It might be that the customer’s business people are not aware about the technical limitations that their systems have, which means that the case company is giving needed support (I12; I13). On the other hand, people who had worked with solutions where customer’s IT or customer’s software vendor was strongly involved, the emphasis was clearly moved from understanding the systems to understanding the interfaces (I1; I5; I7; I9; I11).

“If the customer does not know their system, we need to learn it with the customer. My customer knows well what they want, and they have specifications defining what they are expecting from us to support their system” – Architect 5

Some functionalities can be done in different layers, which is one of the reasons why understanding the integrated systems can be useful (I1; I13). It can be architectural decisions or best practices that defines whether something will be done in the system or in the data integration level. The reasons can also be related to costs. (I1)

“If we can do something in 15 minutes and customer can do it in a month, it is most likely that we will do it” – Technical developer 1

“Knowing the system has a significant effect, because if the system is lacking some functions, it requires certain functionalities from the integration” – Architect 5

Some other needed information and limitations that may affect to the solution was also listed. Internal technical capabilities and restrictions needs to be taken into consideration

(I1; I3; I9; I10; I12). Support point of view needs to be considered, which means that the solution should be designed and documented in a way that error cases can be handled (I1; I2). Also, maintenance and possible changes in the future should be taken into consideration (I1; I2; I6). This is highly related to the documentation which should tell afterwards what has been done (I2; I6).

“Often people are searching from the document how the solution should work” – Architect 1

Non-technical topics were also mentioned. Policies, laws, and other regulation, such as GDPR, needs to be covered (I2; I3; I12). For example, invoices need to have some data depending on the country-specific legislation so that it can be used as a legal invoice, which means that the data needs to be mapped in the data integration (I7; I12). Schedule, budget and available resources also affects to the overall solution (I3; I11).

5.4 Understanding the data

Eleven out of thirteen interviewees mentioned that source and target interfaces must be known. Message formats, message specifications and connectivity protocols were listed as a prerequisite of developing a solution. (I1; I3; I4; I5; I6; I7; I8; I9; I10; I11; I13)

Common message formats and standards was EDIFACT (I1; I2; I3; I4; I6; I7; I8; I9; I10; I11; I12; I13), ANSI X12 (I1; I7; I8; I12), XML (I1; I2; I3; I4; I5; I6; I7; I8; I9; I10; I12), different kind of flat files, such as position based or CSV (I2; I6; I10; I12), and einvoice standards, such as Finvoice or Teapps (I3; I6; I11). EDIFACT was mainly used in Europe while ANSI X12 in the USA (I8; I12; I13) and elsewhere outside the Europe (I8). Even though one argued that XML is not a message format but rather a description language (I2), some listed especially Idoc XML as a commonly used message format (I2; I3; I7). JSON was also mentioned (I9).

Message standards, such as EDIFACT, ANSI X12 or Idoc XML was commonly used in supply chain processes, such as ordering process or logistics. Typical message types were orders (I1; I2; I3; I4; I6; I7; I9; I10; I11; I12; I13), order responses (I1; I3; I6; I8; I10; I11; I12; I13), dispatch advices / advanced shipping notices (I1; I4; I6; I10; I11; I12) and invoices (I1; I4; I6; I8; I10; I11; I12; I13). Messages related to logistics, such as different booking (I4; I6) and status messages (I6), or material and balance details and inventory events (I6; I13) were also listed as common messages. Few people highlighted that especially A2A integrations do not have common standards, which means that all solutions are made using custom messages, even though the format could be XML (I5; I9).

Even though it is not necessary to understand the data in entity level while modeling an overall solution, most important data needs to be identified. (I5; I6; I8; I9; I10; I12; I13) If the data integration is just a passthrough setup, it does not matter what data the messages contain (I2; I10). However, if the solution needs to function certain way depending on the data, those data sets needs to be identified (I6; I8; I10; I12). For example, if a change order process is implemented via data integration, it needs to be checked whether the order data contains an element that can be used to separate normal orders from change orders (I6). It is also important to understand in high-level what is the meaning of a data (I5; I6; I8; I9; I10; I12; I13). For example, if a truck is not allowed to enter the factory area before an EDI message is received, it is crucial to understand why that message is time critical (I12). If the message contains timestamp defining when that truck will arrive, it is very important to understand what that data means from business' point of view (I13).

“It is essential to know how the data is linked to business processes” – Architect 5

It is important to identify most important data as well as understand how that data is affecting to business processes (I13). It is necessary to know how message identifiers are linked to other messages (I1; I4; I13). For example, invoices usually contain order number from order message that allows to match the invoice to the purchase order (I4). Order message cannot be implemented without order number (I4; I9; I13) as well as product information must contain material numbers (I13). If data cannot be mapped from one message to another message, but instead it needs to be mapped from multiple sources to one target, that affects to the design. Therefore, especially message identifiers should be identified when modeling a solution to create a link between different messages. (I1)

Data models, especially differences in message structures, might affect to overall solution, which is why those should be identified at the design phase (I1; I8; I12). For example, if a message is a batch and it needs to be splitted, that affects to overall solution design (I1; I4; I8). Even though the information of data is not known in the design phase, the data must be known before mappings can be made (I1; I2; I3; I4; I5; I6; I7; I8; I9; I10; I12; I13).

“Understanding the information content has a strong correlation with how good the implementation is” – Architect 1

It could be the customer that specifies what data messages contain and how those should be mapped, but if the customer cannot specify the data, the information of the data must be understood in the case company (I3). If the customer does not understand

the data, it is also hard for the developers to understand it (I4; I11; I12). It could be a business analyst who defines how mappings should be made, but if nobody has made a data analysis, it is the technical developer who needs to understand the data (I1). If the information of a data is known, mappings are more likely made correctly (I7; I12) as well as possible errors in specifications are more likely identified (I1; I10; I11; I13). Also, if there are no specifications, understanding the data helps to ask correct questions to ensure that mappings are made correctly (I1; I13).

“Someone needs to understand the data” – Technical developer 1

Schemas and interface descriptions for message standards are very accessible which helps to understand the data (I1; I7; I8). At the same time, specifications for custom messages were harder to get (I1; I3; I6; I8; I10; I11; I12). Especially physical data models, such as XML schemas, can be utilized to create a message interface with tools (I2; I7). Utilizing standards varies which makes it sometimes hard to identify what is important data and what is not, such as what elements are used and what needs to be mapped (I4; I10). As one interviewee summarized, messages usually contain crucial data, such as message identifiers and party identifiers, but also optional data, such as free texts (I1). One interviewee highlighted that A2A integrations do not have well established standards compared to EDI integrations, which makes it harder to understand the data as well as to utilize or create a canonical data model that would be suitable in some specific case (I9).

Knowing canonical data model helps to understand the data (I1; I4; I6; I7). Point-to-point mappings could be done without understanding what data is flowing, but when canonical data models are used, it forces to understand the data (I1). Canonical mappings increase the complexity, because then there are two maps compared to point-to-point mapping that would use only one map (I6). However, at the same time, the canonical data model makes it easier to understand the data (I1; I4; I6; I7; I8; I13). Canonical model helps to identify what data is typically flowing in certain type of data integration and therefore, what needs to be most likely mapped in new solution (I4; I6; I7; I8; I13).

5.5 Modeling process and methods

Modeling process varies between different cases, because starting point differs (I1; I3; I5; I12). Some cases are handed over from sales to a project manager, which means that overall picture is already known because the sales has already defined what will be done to be able to provide a budget (I3). Sometimes the request comes straight from customer’s business, which means that the overall scope is not clear at the start (I12). It

could also be that the request for a new solution comes from customer's IT with more clear requirements (I1).

"The modeling process varies depending on what stage we are involved" – Architect 2

Regardless the starting point, first step is to get information about the business needs and overall scope. If these are not known at the beginning, the information is acquired from different stakeholders. (I3; I5; I6; I8; I10; I12; I13) The modeling process starts by communicating with business people or people who understands the business (I5; I12). Before technical details, it is important to know what participants, systems, and business processes are included to the data integration solution (I3; I6; I8; I9; I10; I12; I13).

"I might not even think what messages are flowing and to which direction, but rather I try to understand the business needs" – Project manager 4

After the overall scope has been defined, next step is to start defining how the technical solution can support the business needs (I1; I5; I6; I9; I10; I12; I13). Used message formats and connectivity protocols are identified at the very beginning (I4; I6; I7; I8; I9; I10; I11; I12). If there are several ways to do the implementation, different options are analyzed and evaluated (I1; I6). The goal is to create first design about the solution to ensure that customer needs are identified correctly (I1; I3; I5; I8; I10; I12; I13). As one interviewee mentioned, the objective is to describe in high-level what will be done as well as what will not be done so that every party involved has a clear understanding about the scope (I3). This phase typically consists of modeling what data is flowing from where to where (I1; I3; I6; I7; I8; I9; I10; I11; I12; I13).

"After there is an understanding about the process, the data integration is modeled on top of it" – Architect 3

Most of the interviewees started to model the solution as a process flow defining what processing is needed in what phases in the message flow (I1; I4; I5; I6; I7; I8; I9; I10; I12). UML diagrams and other diagrams were used to create visual representation about the solution. DFD (I2; I10), CIM (I2; I5; I10; I13), sequence diagram (I5; I6; I10; I13) and flowcharts (I1; I5; I6; I7; I8; I9; I10; I11; I12; I13) were mentioned, as well as any kind of figure that illustrates the overall solution (I1; I3; I5; I6; I7; I8; I9; I10; I11; I12).

As all interviewees argued, diagrams are useful to perceive the big picture of a solution, even if the diagram is just a draft drawn to a whiteboard (I1; I2; I6; I7). For example, if there is need to model a solution that consists of 700 trading partners, it is easy to create one figure that demonstrates all trading partners as one entity (I2). Diagrams can also be utilized to minimize misunderstandings between different stakeholders (I12).

“I model high-level architecture with simple diagrams” – Architect 2

“I use UML diagrams because it is an engineering language and I do not need to explain notations to others” – Architect 1

Many of the interviewee’s mentioned that they use flowcharts to model the message flow in the data integration (I1; I5; I6; I7; I8; I9; I10; I11; I12; I13). Flowchart is easy to use which was the reason many chose that method to model the solution (I1; I6; I7; I8; I10; I12; I13). It can be either high-level description of the solution defining only data flow directions and interfaces, or it can have more details to model a complex solution (I6; I10). Flowcharts can be used to define the sequence of processes (I1; I6; I7; I8; I9; I10; I12; I13), such as what data is flowing from where to where and in what phases some processing is made (I7; I9; I10; I11; I12; I13). For example, if the data integration is using canonical mapping, the flowchart can be used to illustrate that first the inbound message is mapped to canonical and afterwards from canonical to outbound message format (I8). Also, if the data integration needs to have if-else functionalities, those are easy to illustrate with a flowchart (I12). Same logic applies to describe other functionalities, such as illustrate if there is need to handle batches by splitting the files (I1; I4). It is also good that the customer knows at least in some level what the data integration solution is doing (I8), even though things can be abstracted (I8; I10). Flowchart can also be used to describe business processes at least in the level that is necessary to understand the overall message flow (I6; I2; I13). Swimlanes can be used to define what parties are involved and what party is performing which tasks (I6; I10), such as who is sending an order, who is confirming it and what is happening between those business events (I6).

UML diagrams received some critique. As one interviewee mentioned, even though the base assumption is that architects and developers understand the notations of UML diagrams, the reality is that many people are lacking the expertise. (I2) For that reason, selected modeling method should be chosen based on the target audience (I1; I5; I13). Few people mentioned explicitly DFD as one of the modeling methods (I2; I10).

“The audience of DFD is mainly the business people, because the business does not understand the notations of other UML diagrams” – Architect 1

DFD was thought as a suitable method to model the direction of data flows in high-level (I2; I10). DFD is easy to use, and it prevents misunderstandings, because the direction of arrows defines the direction of data flows, which is not the case in all UML diagrams (I2).

Some people start to model the solution immediately against the platform’s functionalities (I1; I7; I9) while others try to model what kind processing is needed for messages flowing

via the data integration (I6; I8; I10; I12; I13). However, this differs between the cases. For example, if an architect has already modeled the solution in some level, the technical developer does not need to model that again, but instead, the focus is to model in more detail level e.g. what components are used in the platform (I7; I9). As some interviewees mentioned, it is very rare situation to start to model a solution with zero background information (I3; I7). The platform also sets some boundaries to the implementation, which is why the platform should be taken into consideration while modeling a solution (I1; I3; I11).

Separate components, such as used interfaces and required maps are identified (I4; I6; I7; I8; I9; I12; I13). Other needed components, such as schedulers, pipelines and databases, are also defined (I9). Only architects, four out of five, used CIM diagrams to describe the components (I2; I5; I10; I13).

“CIM is suitable for understanding the environment and to model the interfaces that are important from our point of view” – Architect 5

Level 0 and level 1 CIM was used (I13). The reason to select CIM as a modeling method was because it can be used to define from where the connectivity is made (I2; I10). For example, firewall restrictions and needed actions, such as whitelisting IPs, derives from the direction of connectivity (I10). One thought that solution designs should have a CIM diagram because all other diagrams are lacking the information that CIM provides (I2).

Similar with CIM, sequence diagrams were also used only by architects and four out of five mentioned that they use it sometimes (I5; I6; I10; I13). Sequence diagram can be used to illustrate use cases (I5; I13) and the sequence of processes (I6; I13). One architect stated that sequence diagram is more detail level technical description about the solution, and it illustrates indeed the sequence better than other diagrams. However, same logics can be illustrated with flowcharts in higher level which means that sequence diagram is not so useful. (I6) One mentioned that sequence diagrams are useful in the development phase (I9).

5.6 Common challenges

Most common challenges related to design phase were related to not understanding the requirements and scope (I1; I2; I3; I5; I6; I8; I10; I11; I12; I13), inadequacy technical information (I1; I2; I3; I4; I5; I6; I7; I8; I10; I11; I12; I13) and lack of commitment (I3; I7; I10; I12) from different stakeholders.

It can be hard to understand the customer needs if the customer cannot specify what are the requirements (I1; I2; I8; I10; I11). The customer can describe the requirements in

very high-level (I1; I8) and involved stakeholders might lack the expertise about technical stuff (I8; I10; I12). For example, the customer's business might ask for a solution that is impossible to be done in data integration level, such as fixing problems in customer's own systems (I1).

"Sometimes the customer does not even know what they want, and they assume that we know better what they should want" – Architect 1

"We might discuss with people who do not understand things at good enough level" – Project manager 2

Sometimes the scope has been defined by business or sales and when technical people are involved, the requirements change radically (I12). Often the information received is not true and things need to be ensured multiple times (I8). For example, there was a case where a project was started to automatize ERP ordering process, but during the project it became clear that the trading partner's ERP system was not able to send or receive any data (I12).

It can be hard to define the scope (I1; I3; I7; I10; I13) and lack of information can cause problems and delays (I1; I5; I11; I12). For that reason, sometimes there is no point to design anything but instead pilot something quickly (I1; I3). This can cause problems since the implementation is made without fully understanding the requirements. Therefore, the issues are detected later e.g. during the testing phase which causes iterations because the solution needs to be remodeled and implemented again. This extends the schedule as well as increases costs. (I5; I6; I11; I13)

"Sometimes as the information increases, we discover that previously chosen solutions are not suitable, and we have to return to the starting point and think again what needs to be done" – Architect 2

Lack of commitment creates also challenges (I3; I7; I10; I12). Project is delayed because some stakeholder cannot provide needed information in time (I6; I7; I10; I12; I13), such as message specifications (I7; I13). Sometimes people are not available when needed or does not respond to emails, which delays the design process (I10; I12).

Capabilities of a platform can also create challenges (I1; I2; I9; I10). Even though there are only few technical impossibilities that would prevent the solution performing as wanted, the solution should be modeled with good architecture practices (I1; I2; I9). If technical people are not involved in the design phase, all necessary limitations might not be identified in advance (I10). Even if something is technically possible, bad decisions can increase the complexity of a solution (I1; I9) which increases the implementation time and costs, as well as makes the solution hard to maintain (I1).

When moving towards the development phase, typical mapping challenges were related to message structures. If the source, canonical, or target data model have different looping structures, the mapping could be hard. (I1; I7; I8; I12; I13) For example, if an inbound dispatch advice contains only item rows and outbound message format needs to receive package level and item rows under the packages, the mapping is impossible because the source data model is lacking the necessary information (I8). Similar issues are faced when receiver needs to receive some data that the sender is not able to provide (I4).

Another common challenge in mappings was lack of information about the data and not understanding how the data should be mapped (I3; I5; I6; I7; I9; I10; I11; I12; I13). The customer might deliver only a schema assuming it gives all information that is needed to map the data correctly (I4; I6). Therefore, the necessary information needs to either be acquired or mappings need to be guessed. Both options mean that the development takes more time. If mappings are guessed, tested and corrected, it also means more iterations. (I4; I6; I8; I11; I12) Some argued that the people working in the case company should have expertise about standard solutions (I2; I8), meaning that e.g. data flowing via order message should be known pretty well by default (I2), whereas custom solutions need more information and better specifications from the customer or other stakeholders (I2; I9). However, one interviewee highlighted that even message standards can be utilized many ways and if there are no specifications, the mappings might be made incorrectly (I7).

6. DISCUSSION

6.1 Modeling process

As literature review shows, the modeling process should be started by defining the business processes and business needs (Berente et al. 2009; Dorn et al. 2009; Giordano 2010; Schubert & Legner 2011; Fan et al. 2012; Reeve 2013). Results of empirical research supports this approach. Business processes defines the objectives of a data integration and therefore, the data integration solution should support the business processes (Al-Naeem et al. 2004; Berente et al. 2009; Giordano 2010; Fan et al. 2012). Empirical research shows that designing a data integration is hard without knowing the business processes beyond the technical solution and highlights that the business perspective should be covered before technical solution can be modeled. It was clear that understanding how the integration is related to business processes affects to the solution design. Therefore, the modeling process should be started by identifying the business goals and reasons why a data integration solution is needed as well as to define the scope of a solution.

Results of empirical research indicates that understanding the business processes and business needs might be hard. Sometimes even the business is not aware about the business processes and requirements. This causes challenges because scope might change during time when more information is received, and this causes more iterations. Lack of commitment was also an issue since it typically delays the project. This verifies the importance of capturing the business requirements as early as possible for smoother design and development process.

According to Bussler (2002), data integration typically consists of multiple message flows flowing between different systems. The interactions of those systems affect to the data integration solution decisions (Lobaziewicz 2015). Gartner (2016) states that understanding the systems is important for a successful data integration. Empirical research partly confirms this perception and shows that it is important to understand how the messages are linked to each other and what is happening in real-life when a message is flowing via data integration from one system into another. However, the empirical research shows that a solution can be done without understanding the systems if the business processes and message interfaces are known. Even though integrated systems can set restrictions and boundaries to the overall solution, empirical research stresses that the understanding should come from the customer, because the case company has

only limited visibility towards the integrated systems. Empirical study emphasizes that good interface specifications and understanding about business processes can be enough and that the systems behind the interfaces do not have so big impact. Results also shows that lack of information, such as bad or missing interface specifications, causes challenges. In situations like this, understanding about the integrated systems can be helpful.

Existing theory proposes top-down modeling process moving from business needs to conceptual model, logical model, and finally to physical model (Giordano 2010; Halpin & Morgan 2010; Fan et al. 2012; Tillmann 2017). Even though top-down modeling process was in general thought as a suitable method in the case company, empirical research shows that integration platform sets boundary conditions to the solution, which means that logical model perspective should be considered quite early. Sometimes the conceptual model can be skipped, which means that after business needs are clear, the solution will be modeled with the restrictions and limitations of the technology – that are deriving from the integration platform in this research context. Complex solutions need more modeling while standard solutions could be made without a conceptual model.

According to Giordano (2010), high-level processes need to be decided before moving forward to low-level details. The research results show that small technical details might affect radically to the overall solution. Therefore, the modeling process needs to be more flexible. Data models, such as a physical schema, can set restrictions to the solution and have big impacts towards the conceptual or logical model. For example, if an order message does not have data indicating whether it is a new order or a change order, it might be that the data integration solution should track the message flow to separate those. If this kind of things are noted only at the later stage, it can lead to remodeling the whole solution after something has already been developed. Therefore, the research results suggest more iterative process between logical and physical model.

According to theory, physical model is used in the development phase and it is detailed representation about the solution (Giordano 2010; Fan et al. 2012). However, existing literature in this paper does not consider information classification. Research shows that it is important to know what can and cannot be shared outside the organization, meaning that what level of details can be shared to a customer. Research shows that internal technical solutions should not be visible for the customer. This means that customer documentation should not contain physical data integration model, such as lines of code or internal components that are used in the platform.

However, since data transformation and mappings are very important topic in a data integration solution (Lenzerini 2002; Amao et al. 2014), physical *data models* should be considered at the early stage. Research shows that message formats and message specifications are prerequisites of developing a solution, which means that physical data models needs to be known even though physical data integration model is not needed. For example, if it is known what data is flowing and how the mappings between different structures are made, it is enough to model the overall solution. Technical solutions, such as technology that is utilized in physical data integration model is not important aspect before the actual development is started. Different data models and especially different looping structures were one of the most common challenges when creating the mappings. Sometimes mappings are impossible, which might reflect to overall solution since alternative solutions are needed, such as getting data from multiple sources instead of mapping one source file to one target file.

Especially B2B EDI integrations utilize common message formats and standards. Existing literature states that EDIFACT and ANSI X12 still have a dominant role (Engel et al. 2016), even though also flat files (Kuchibhotla et al. 2009), XML files (Kuchibhotla et al. 2009; Schubert & Legner 2011; Reeve 2013) and JSON files (Reeve 2013; Afsari et al. 2017; Barbaglia et al. 2017) are used. Empirical research confirms this as all of those were listed as most common message formats and standards. Results shows that JSON files are not so common while EDIFACT and XML were listed to most common formats. Lieg et al. (2011) and Engel et al. (2012) states that standards can be utilized in multiple ways. Empirical research supports this vision and highlights that specifications about the data is needed even with standards. Results also shows that especially custom solutions need more input from the customer or other stakeholders in order to understand the data. Similar results were found from the literature, since Jun & Cai (2003) mentions that message specifications are highly affecting to good solution.

Theory shows that it is important to understand the information content of the data to create correct mappings (Cardoso & Bussler 2011; Reeve 2013; Shahbaz 2015). Empirical research supports this and emphasizes that mappings cannot be made correctly if the meaning of the data is not understood. Literature review shows that point-to-point mappings can be made with limited understanding about the information of the data whereas canonical mappings requires more knowledge (Gleghorn 2005; Dejan 2007; Engel et al. 2012; Reeve 2013). Similar results were gained by the empirical study. Results shows that canonical mapping helps to understand the data as stated in existing theory (Dietrich & Lemcke 2011). As Gartner (2015) mentions, canonical data models should be defined for certain business processes. Empirical study shows that custom

A2A integrations do not have well established standards and practices which causes challenges to utilize and create canonical data models. At the same time, canonical data models created for B2B EDI integrations were very helpful to create correct mappings and to understand what information is flowing via the data integration solution.

Even though all conceptual, logical and physical data integration models could be utilized in different phases of a data integration solution, based on the research results, the logical data integration model is the most important level while modeling a new solution. It can be done within given budget and it provides enough information to understand what the solution is doing without going too deeply to technical details. Therefore, it both provides required information to the customer about the solution that will be developed as well as gives guidelines for the developers to create a solution that will fulfill the requirements.

6.2 Modeling methods

Kim et al. (2000) and Al-Naeem et al. (2004) claims that data integration modeling is complex process and needs different level of abstraction. Giordano (2010) proposes to use figures to model the data integration solution. Empirical research supports the theory and research results shows that diagrams are very often utilized when modeling a solution. Research shows that diagrams are useful to understand the business requirements as well as to illustrate the overall technical solution. Whereas simple solutions can be illustrated with simple figures, complex solutions need multiple diagrams. Kim et al. (2000) supports this view by stating that different diagrams describes the solution from different point of view and highlights that complex solutions need more than one diagram.

Existing literature proposes to use UML diagrams to model a solution (Eriksson & Penker 2000; Giordano 2010; Baqais & Alshayeb 2018). Research shows that mainly architects are using UML diagrams. Some UML notations are not clear for everyone which makes it unsuitable method to communicate with business. Li et al. (2018) supports this point of view stating that all diagrams are not suitable for all stakeholders.

Based on existing literature, DFD is commonly used method to illustrate data processing and the direction of data flows (Fountas et al. 2006; Giordano 2010; Yang 2014; Tillmann 2017; Zhang et al. 2018). Yang (2014) and Tillmann (2017) mentions that DFD can be decomposed to multiple levels, such as level 0 DFD defining the entire system and level 1 DFD illustrating subprocesses in the system. Empirical research shows that DFD was not commonly used method, or at least people are not aware whether they use DFD or not. People might model DFD without knowing that it is indeed a DFD. However, the

research also shows that DFD was thought as a suitable method to model high-level conceptual model utilizing level 0 DFD. Theory background (Yang 2014; Tillmann 2017) claims otherwise stating that DFD is logical model. Yang (2014) and Zhang et al. (2018) mentions that DFD is suitable method to abstract the overall solution. Research supports this and shows that DFD is easy to use and it prevents misunderstanding.

Medvidovic et al. (2002) states that UML is not suitable method to illustrate architecture-level components. Research claims otherwise and especially UML component diagram is useful to create a CIM diagram that illustrates how components are communicating with each other. Mokarat & Vatanawood (2013) supports this view stating that UML component diagrams can be used to illustrate architectural model. Xia et al. (2000) proposes to use CIM to illustrate the communication of interfaces. Existing literature (Xia et al. 2000; Coulson et al. 2008) proposes to use CIM to illustrate how new components can be added into existing architecture. Research shows that CIM is useful to demonstrate the direction of connectivity but other benefits of CIM were not identified.

Theory proposes to use sequence diagrams to model message flows between systems. Sequence diagram can be utilized to illustrate the relations of message flows and sequence of processes. (Grgec & Mužar 2007; Baqais & Alshayeb 2018) According to Baqais & Alshayeb (2018), sequence diagram is not suitable method to describe complex solutions, but research shows that sequence diagram is useful especially in complex solutions. Research shows that use cases can be modeled with sequence diagram and it is typically complex solutions that have multiple use cases that would need separate sequence diagrams. Research points out that sequence diagrams are rarely used since other models can offer same benefits. This is highly related to the fact that UML notations can be complex and are not commonly known. Whereas architects and developers might have the expertise of creating a sequence diagram, many people are lacking the skills. Therefore, sequence diagram is not suitable modeling method to be utilized in simple solutions.

Existing literature shows that also other diagrams than UML diagrams can be utilized to describe the solution (Gleghorn 2005; Meier & Meier 2011; Appavuraj et al. 2014; Tillmann 2017; Li et al. 2018). Empirical research strongly supports this perspective. Even though DFD, CIM and sequence diagrams can be useful in some cases, research shows that most of the people use flowcharts since they are easy to use and understand. Flowcharts can be used to represent simple solutions as well as detail level can be expanded in case of more complex solutions.

Research shows that flowcharts provides same benefits than DFD illustrating the direction of data flows. They are practical to illustrate the sequence of processes, which is why some prefer flowcharts over sequence diagrams. Flowcharts can be used to describe business processes in a level that is needed to understand the business objectives of a solution. Appavuraj et al. (2014) supports this point of view stating that flowcharts can be utilized to clarify requirements.

Things can be abstracted in flowchart which means that internal solutions can be simplified for a documentation that is given to a customer. At the same time, some internal solutions can be revealed at the level that is needed for the customer to understand the solution. For example, if data is mapped via canonical, it can be easily shown in the flowchart. Flowchart was also useful to capture all necessary logics so that it acts as a guideline for developers to implement a solution that fulfills the requirements. Flowchart does not need to consider physical data integration models, such as internal components that are needed for expected outcome. Existing theory supports this by stating that processes can be abstracted to illustrate the solution in an easily understandable way (Meier & Meier 2011; Appavuraj et al. 2014).

7. CONCLUSIONS AND SUMMARY

7.1 Conclusions

The research objective was to answer to the primary research question which was: “How to leverage data integration modeling in data integration solutions?” According the pragmatism research philosophy, the goal was to provide practical solutions that the case company could utilize in the future with new customer projects and solutions. This study was done as a case study that used structured interviews to collect empirical material. This paper studied people who are working with solutions that are delivered to the customers using the case company’s integration platforms. According the deductive approach, existing literature was combined to the key findings of empirical research and analyzed using thematic analysis.

As the research results show, the modeling process starts by understanding the business. Business processes defines the objectives of a technical solution. Conceptual model can be created based on the business needs even though technology can set boundaries to the solution. Therefore, sometimes it can be better to move from the business needs to a logical model. This way it is possible to start modeling a solution with existing restrictions and best practices. The modeling process is iterative, because physical data models can set limitations to the overall solution. The earlier the restrictions and limitations are identified, the less the solution needs to be remodeled. Different diagrams can be utilized to help the communication between different stakeholders.

First step to model a data integration solution should be to understand the business needs and business processes. Since all necessary information is rarely available in the beginning, the first goal should be to understand the scope of a solution, such as what participants are integrated and what data is flowing. Technical details, such as integrated systems, message formats or specifications, or connectivity protocols are not necessary in the beginning if the information is not available. After the overall scope is clear, level 0 DFD should be created. The level 0 DFD should define what data is flowing from where to where and it acts as a conceptual data integration model. Level 0 DFD does not need special skills and therefore, basically anyone can create it. Alongside creating the DFD, the solution design should also describe *why* the solution is needed, such as defining in high-level the business processes that are automatized via data integration solution. Therefore, the conceptual model provides information about the business objectives.

Next step is to identify technical details in high-level. Understanding about the source and target interfaces and message formats should be achieved. If message standards are not used, message specifications should be available as early as possible. Once the source and target messages are known, it should be identified what functionalities and processes are needed from the data integration solution. If customer has multiple business units, it should be defined how those affects to the message flows. Any special logics that is different from transforming the source files to target files should be identified. If the data integration needs to handle batches, schedulers and splitters should be identified as well as databases or any functionality where data needs to be gathered from multiple sources or sent to multiple targets. This might require becoming familiar with the data models, but the goal should be to identify crucial information instead of checking on entity level what data is flowing. Since exceptional looping structures might affect to overall solution, those should be identified. Connectivity protocols and clients/hosts should also be identified to know who the active party is.

During or after the previous phase, flowchart should be made. The flowchart should abstract the solution to provide easily understandable description about the most important process steps. The flowchart should provide logical data integration model that considers source and target interfaces' as well as platform's capabilities and boundaries. The flowchart should be reviewed by the customer and approved in case of approval is needed. It should contain as much information as is needed from overall solution's point of view. Therefore, it should also give enough details for the developers to implement a solution that is functioning as is required. The logical model does not need to take into consideration what components are used in the integration platform to create a certain functionality. The flowchart should define *what* functionalities are expected from the solution, but it should not decide *how* those functionalities are implemented since that question can be left out for the developers to decide in the later phases. Therefore, the same documentation can be delivered to the customer as well as it provides guidelines for the developers.

If it becomes clear that the solution will be complex, sequence diagrams could be utilized. Sequence diagrams should define special use cases that are not feasible to demonstrate via flowcharts. If the assumption is that involved stakeholders do not understand sequence diagrams, the logical flowchart model can be disintegrated to multiple diagrams.

After connectivity details are clear, level 0 CIM should be created. It should define what are the connectivity protocols and from which direction the connectivity is made. It should describe whether the files are sent or fetched by the data integration solution. Level 0

CIM can be modeled quite easily and it is the most suitable figure to illustrate who the active party is.

After all previous phases are done, the solution design should contain enough information for all involved stakeholders to understand what kind of solution will be made. It will tell *why* something is needed as well as it describes *what* process steps are required for desired outcome. The solution design will provide enough information for the customer's business and IT to understand what the solution will do and what functionalities are included to fulfill the business and technical requirements. The solution design also contains enough details for technical developers to start developing the solution that was modeled regardless whether the developers were involved in the design phase or not. If major changes are detected during the development phase, the solution design should be updated. Minor changes, such as what source entity is mapped to target entity, do not affect to the solution design since those are not mentioned in the document, which ensures that the solution design is up to date. The solution design can also be used after the go-live to support the maintenance as well as it can be utilized in the future when new solutions are added on top of existing architecture.

7.2 Evaluation of the research and future study

The research goals were fulfilled, because research questions were answered. This paper also provides practical solutions for modeling data integration solutions. This research succeeded to identify data integration modeling methods as well as identified how those could be utilized in the case company's context. Conclusions provides answers based on existing literature and empirical research and the results of this research can be deployed to operational business in the case company.

In addition to answering to the primary research question, this paper also answers to all secondary research questions. First secondary research question was to identify what the meaning of data integration is. Chapter 2 covered most important aspects from this research's point of view and answered the first secondary question. Second secondary question was to understand how data integrations can be modeled. Chapter 3 meets the objectives of that question, even though the topic is covered quite narrowly. It became clear during the later phases of this research that data integration models are quite broad topic. Therefore, more perspectives could have been covered. Third secondary question was to understand what information is needed to design a data integration solution in the case company and last secondary question was to identify key challenges of a solution process. Empirical research presented in chapter 5 answers these questions and many different perspectives were identified.

Even though the goal was to make objective research, it should however be noted that the researcher might be biased. Structured interview questions were made by the researcher and therefore, some important aspects or topics might have been uncovered. The researcher's own opinions might affect the research results since the researcher is involved in daily operations of the case company and has expertise about certain type of data integrations, business processes, industries and customers. For example, the research results show that complex and custom A2A integrations are not so straightforward compared to standardized B2B EDI integration. Therefore, the research results might not be utilized in all data integration solutions.

Same logic applies to creating the research questions and all aspects related to the research topic were not covered. The researcher might have too architectural and technical viewpoint. For example, project management was covered lightly although projects have always limited resources, budget and schedule.

Though sampling was made, people that were not interviewed might have dissenting opinions which could impact the research results. Wider sample could have led to different results. While longer schedule would have allowed to interview wider sample, this research had limited schedule. However, the sampling was successful since the results of empirical research supported the research objectives.

It should also be noted that this research is a case study and thus, the results are highly linked to the case company's environment. If similar research would have been carried out in another organization, the results might be different. Therefore, there are clear limitations to the generalization of the results of this research. However, at the same time, the results are quite well supported by existing literature which confirms that the research results could be utilized in broader context and in future studies.

Existing theory and empirical research about data integration modeling in integration brokerage's environment is currently very limited. Most of the existing literature is focused to data integration modeling in different environment, such as modeling data integrations in databases or data warehouses. Utilizing cloud services, such as case company's integration platforms, has received little attention. This means that there are multiple opportunities for future studies.

Firstly, longitudinal research could be made to test whether the conclusions of this paper are suitable in practice or not. Secondly, this research did not consider the customer's point of view very widely. Future study could be made to identify what models provides most value for the customer who is not doing the technical implementation. Research could also be made to study more the documentation aspect. For example, if solution

architecture is built within multiple projects during long time period, the documentation could be different compared to a new solution.

While this research studied how to utilize existing integration platforms in customer solutions, this paper did not consider how an integration platform should be modeled when it is developed. This could be a topic for future research. Another research could be made about modeling technical solution focusing more to the physical data integration models. That topic was not covered broadly in this research since it was thought due to information classification as an internal model that should not be published in public release. In addition to previous suggestions, future research could also focus to study how very big and complex integrations, such as big data or IoT ecosystems, should be modeled.

REFERENCES

- Afsari, K., Eastman, C.M. & Castro-Lacouture, D. (2017). JavaScript Object Notation (JSON) data serialization for IFC schema in web-based BIM data exchange, *Automation in Construction*, Vol. 77 pp. 24-51.
- Al-Naeem, T., Rabhi, F.A., Benatallah, B. & Ray, P.K. (2004). Systematic Approaches for Designing B2B Applications, *International Journal of Electronic Commerce*, Vol. 9(2), pp. 41-70.
- Amano, S., David, C., Libkin, L. & Murlak, F. (2014). XML Schema Mappings: Data Exchange and Metadata Management, *Journal of the ACM (JACM)*, Vol. 61(2), pp. 1-48.
- Appavuraj, R., Gupta, P.K.D., Ghosh, P., Samal, P.B., Saha, A. & Proof and Experimental Establishment, DRDO, Chandipur-756 025, India (2014). System Analysis and Design of Armament Integrated Management System, *Defence Science Journal*, Vol. 64(6), pp. 524-529.
- Asiakastieto (2019). Liaison Technologies Oy, taloustiedot. Available (Accessed at 11.5.2019): <https://www.asiakastieto.fi/yriytykset/fi/liaison-technologies-oy/08599126/taloustiedot>
- Baqais, A.A.B. & Alshayeb, M. (2018). Sequence diagram refactoring using single and hybridized algorithms, *PloS one*, Vol. 13(8), pp. e0202629.
- Barbaglia, G., Murzilli, S. & Cudini, S. (2017). Definition of REST web services with JSON schema, *Software: Practice and Experience*, Vol. 47(6), pp. 907-920.
- Batra, D., & Marakas, G. M. (1995). Conceptual data modelling in theory and practice. *European Journal of Information Systems*, 4(3), 185-193.
- Berente, N., Vandenbosch, B. & Aubert, B. (2009). Information flows and business process integration, *Business Process Management Journal*, Vol. 15(1), pp. 119-141.
- Blanco, R., Enriquez, J.G., Dominguez-Mayo, F.J., Escalona, M.J. & Tuya, J. (2018). Early Integration Testing for Entity Reconciliation in the Context of Heterogeneous Data Sources, *IEEE Transactions on Reliability*, Vol. 67(2), pp. 538-556.
- Bussler, C. (2002). The Application of Workflow Technology in Semantic B2B Integration, *Distributed and Parallel Databases*, Vol. 12(2), pp. 163-191.
- Cagiltay, N.E., Tokdemir, G., Kilic, O. & Topalli, D. (2013). Performing and analyzing non-formal inspections of entity relationship diagram (ERD), *The Journal of Systems & Software*, Vol. 86(8), pp. 2184-2195.
- Cardoso, J. & Bussler, C. (2011). Mapping between heterogeneous XML and OWL transaction representations in B2B integration, *Data & Knowledge Engineering*, Vol. 70(12), pp. 1046-1069.
- Chen, H., Jiang, J., Hong, Z. & Lin, L. (2018). Decomposition of UML activity diagrams, *Software: Practice and Experience*, Vol. 48(1), pp. 105-122.

- Coulson, G., Blair, G., Grace, P., Taiani, F., Joolia, A., Lee, K., Ueyama, J. & Sivaharan, T. (2008). A generic component model for building systems software, *ACM Transactions on Computer Systems (TOCS)*, Vol. 26(1), pp. 1-42.
- Debicki, T. & Kolinski, A. (2018). INFLUENCE OF EDI APPROACH FOR COMPLEXITY OF INFORMATION FLOW IN GLOBAL SUPPLY CHAINS, *Business Logistics in Modern Management*, pp. 683-694.
- Dejan, R. (2007). An integration strategy for large enterprises, *Yugoslav Journal of Operations Research*, Vol. 17(2), pp. 209-222.
- Dietrich, M. & Lemcke, J. (2011). A Refined Canonical Data Model for Multi-schema Integration and Mapping, 2011 IEEE 8th International Conference on e-Business Engineering, IEEE, pp. 105-110.
- Dong, X.L., Halevy, A. & Yu, C. (2009). Data integration with uncertainty, *The VLDB Journal*, Vol. 18(2), pp. 469-500.
- Dorn, J., Grün, C., Werthner, H. & Zapletal, M. (2009). From business to software: a B2B survey, *Information Systems and e-Business Management*, Vol. 7(2), pp. 123-142.
- Edifactory (2019). Available (accessed at 14.4.2019): <https://www.edifactory.de/>
- Engel, R., Krathu, W., Zapletal, M., Pichler, C., Bose, R. P. Jagadeesh Chandra, Aalst, W., Werthner, H. & Huemer, C. (2016). Analyzing inter-organizational business processes, *Information Systems and e-Business Management*, Vol. 14(3), pp. 577.
- Engel, R., Pichler, C., Zapletal, M., Krathu, W. & Werthner, H. (2012). From Encoded EDIFACT Messages to Business Concepts Using Semantic Annotations, 2012 IEEE 14th International Conference on Commerce and Enterprise Computing, IEEE, pp. 17-25.
- Eriksson, H. & Penker, M. (2000). *Business modeling with UML*, New York, pp. 1-12.
- Fan, S., Zhao, J.L., Dou, W. & Liu, M. (2012). A framework for transformation from conceptual to logical workflow models, *Decision Support Systems*, Vol. 54(1), pp. 781-794.
- Finder (2019). Liaison Technologies Oy, taloustiedot. Available (accessed at 11.5.2019): <https://www.finder.fi/IT-konsultointi+IT-palvelut/Liaison+Technologies+Oy/Tampere/yhteystiedot/113881>
- Fountas, S., Wulfsohn, D., Blackmore, B.S., Jacobsen, H.L. & Pedersen, S.M. (2006). A model of decision-making and information flows for information-intensive agriculture, *Agricultural Systems*, Vol. 87(2), pp. 192-210.
- Galinec, D., Steingartner, W. & Macanga, D. (2012). Command and control information systems semantic interoperability using a canonical messaging approach, *Open Computer Science*, Vol. 2(3), pp. 316-330.
- Gartner (2015). Negotiate Better Integration Brokerage Agreements With These Essential Best Practices. Available (accessed at 9.5.2019): <https://www.gartner.com/en/documents/2998119>
- Gartner (2016). Market Guide for Integration Brokerage. Available (accessed at 9.5.2019): <https://www.gartner.com/en/documents/3551218>

- Gartner (2017). B2B/EDI Integration: Choosing Between In-House or Outsourced Delivery. Available (accessed at 9.5.2019): <https://www.gartner.com/en/documents/3621325>
- Giordano, A.D. (2010). Data Integration Blueprint and Modeling: Techniques for a Scalable and Sustainable Architecture, Pearson Education.
- Gleghorn, R. (2005). Enterprise application integration: a manager's perspective, IT Professional, Vol. 7(6), pp. 17-23.
- Gogolla, M. & Hohenstein, U. (1991). Towards a semantic view of an extended entity-relationship model, ACM Transactions on Database Systems (TODS), Vol. 16(3), pp. 369-416.
- Grgec, M. & Mužar, R. (2007). ROLE OF UML SEQUENCE DIAGRAM CONSTRUCTS IN OBJECT LIFECYCLE CONCEPT, Journal of Information and Organizational Sciences, Vol. 31(1).
- Gupta, M. (2015). DPaaS: A Prerequisite to Effective BI in Today's Dynamic Data Environment, Business Intelligence Journal, Vol. 20(3), pp. 38.
- Halpin, T. & Morgan, T. (2010). Information modeling and relational databases, Morgan Kaufmann.
- Hill, N.C. & Ferguson, D.M. (1989). Electronic data interchange: a definition and perspective, Citeseer, pp. 5-12.
- Hoberman, S. (2009). Where Does XML Fit?: Is an XML document a logical model or a physical data model? Information Management, Vol. 19(8), pp. 38.
- Holy, L., Snajberk, J. & Brada, P. (2012). Visual clutter reduction for UML component diagrams: A tool presentation, 2012 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), IEEE, pp. 253-254.
- Hüsemann, B., Lechtenbörger, J. & Vossen, G. (2000). Conceptual data warehouse design, Universität Münster. Angewandte Mathematik und Informatik.
- IBM (2019a). What is B2B integration? Available (accessed at 15.4.2019): <https://www.ibm.com/supply-chain/b2b-integration>
- IBM (2019b). What is data integration? Available (accessed at 12.4.2019): <https://www.ibm.com/analytics/data-integration>
- IDC (2019). Data Integration and Integrity Software. Available (accessed at 12.4.2019): https://www.idc.com/getdoc.jsp?containerId=IDC_P31631
- Incomes Register (2019a). Data distribution – Schemas – Earnings payment reports. Project to establish the National Incomes Register. Available (accessed at 20.4.2019): <https://www.vero.fi/globalassets/tulorekisteri/data-distribution-schemas-earnings-payment-reports.pdf>
- Incomes Register (2019b). Software developers. Available (accessed at 20.4.2019): <https://www.vero.fi/en/incomes-register/software-developers/>

Incomes Register (2019c). Technical interface – ZIP file for application developers. Available (accessed at 19.3.2019)

Informatica (2019). What is Application Integration? Available (accessed at 12.4.2019): <https://www.informatica.com/services-and-training/glossary-of-terms/application-integration-definition.html#fbid=Nj0OGI9PNKc>

Joutsenlahti, J-P (2017). Data management platform architecture – case study, Master of Science Thesis, Tampere University of Technology, 62p.

Jun, M. & Cai, S. (2003). Key obstacles to EDI success: from the US small manufacturing companies' perspective, *Industrial Management & Data Systems*, Vol. 103(3), pp. 192-203.

Kalinichenko, L.A. (1990). Methods and tools for equivalent data model mapping construction, Springer, pp. 92-119.

Karimpour, J., Isazadeh, A. & Izadkhah, H. (2013). Early performance assessment in component-based software systems, *IET Software*, Vol. 7(2), pp. 118-128.

Kim, J., Hahn, J. & Hahn, H. (2000). How Do We Understand a System with (So) Many Diagrams? Cognitive Integration Processes in Diagrammatic Reasoning, *Information Systems Research*, Vol. 11(3), pp. 284-303.

Kuchibhotla, H.N., Dunn, D. & Brown, D. (2009). Data integration issues in IT organizations and a need to map different data formats to store them in relational databases, 2009 41st Southeastern Symposium on System Theory, IEEE, pp. 1-6.

Lenzerini, M. (2002). Data integration: A theoretical perspective, *ACM*, pp. 233-246.

Li, H., Zhao, A., Zhang, D. & Zhang, J. (2018). Research on building software usage model based on UML model, *International Journal of System Assurance Engineering and Management*, Vol. 9(3), pp. 675-683.

Liegl, P., Huemer, C. & Pichler, C. (2011). Registry support for core component-based business document models, *Service Oriented Computing and Applications*, Vol. 5(3), pp. 183-202.

Lobaziewicz, M. (2015). Integration of B2B system that supports the management of construction processes with ERP systems, 2015 Federated Conference on Computer Science and Information Systems (FedCSIS), Polish Information Processing Society (PIPS), pp. 1461-1466.

Manouvrier, B., Ménard, L. & ebrary, I. (2008). Application integration: EAI, B2B, BPM and SOA, ISTE, Hoboken, NJ; London; John Wiley & Sons.

Masri, K., Parker, D. & Gemino, A. (2008). Using Iconic Graphics in Entity-Relationship Diagrams: The Impact on Understanding, *Journal of Database Management (JDM)*, Vol. 19(3), pp. 22-41.

Medvidovic, N., Rosenblum, D.S., Redmiles, D.F. & Robbins, J.E. (2002). Modeling software architectures in the Unified Modeling Language, *ACM Transactions on Software Engineering and Methodology (TOSEM)*, Vol. 11(1), pp. 2-57.

- Meier, F.A. & Meier, C.A. (2011). The Process Flow Diagram, Instrumentation and Control Systems Documentation (2nd Edition), ISA, pp. 15-25.
- Mitlöhner, J., Neumaier, S., Umbrich, J. & Polleres, A. (2016). Characteristics of open data CSV files, IEEE, pp. 72-79.
- Mokarat, C. & Vatanawood, W. (2013). UML Component Diagram to Acme Compiler, 2013 International Conference on Information Science and Applications (ICISA), IEEE, pp. 1-4.
- Mukhopadhyay, T. & Kekre, S. (2002). Strategic and Operational Benefits of Electronic Integration in B2B Procurement Processes, Management Science, Vol. 48(10), pp. 1301-1313.
- Opentext (2018a). What is Electronic Data Interchange (EDI)? Available (accessed at 12.4.2019): <https://blogs.opentext.com/electronic-data-interchange-edi/>
- Opentext (2018b). OpenText Buys Liaison Technologies, Inc. Available (accessed at 2.5.2019): <https://www.opentext.com/about/press-releases?id=521E2DCCF95A46448B307DB004BEF90E>
- Opentext (2019). B2B integration. Available (accessed at 12.4.2019): <https://www.opentext.com/products-and-solutions/products/business-network/b2b-integration-services>
- Opentext+Liaison (2019). Company's web sites. Available (accessed at 11.5.2019): <https://liaison.opentext.com/>
- Patig, S. (2006). Evolution of entity–relationship modelling, Data & Knowledge Engineering, Vol. 56(2), pp. 122-138.
- Peppol (2019). BIS Ordering 3.0. Available (accessed at 11.5.2019): <http://docs.peppol.eu/poacc/upgrade-3/profiles/28-ordering>
- Ravindran, N., Liang, X. & Liang, Y. (2010). A labeled-tree approach to semantic and structural data interoperability applied in hydrology domain, Information Sciences, Vol. 180(24), pp. 5008-5028.
- Reeve, A. (2013). Managing data in motion: data integration best practice techniques and technologies, Newnes.
- Saeed, K.A., Malhotra, M.K. & Grover, V. (2005). Examining the Impact of Interorganizational Systems on Process Efficiency and Sourcing Leverage in Buyer-Supplier Dyads, Decision Sciences, Vol. 36(3), pp. 365-396.
- Sarkar, P. (2015). Data as a Service: A Framework for Providing Reusable Enterprise Data Services, 1st ed. Wiley-IEEE Computer Society Pr, US.
- Saunders, M., Lewis, P., & Thornhill, A. (2016). Research methods for business students, 7. painos, Pearson Education Ltd, 741p.
- Schubert, P. & Legner, C. (2011). B2B integration in global supply chains: An identification of technical integration scenarios, Journal of Strategic Information Systems, Vol. 20(3), pp. 250-267.

- Shahbaz, Q. (2015). *Data Mapping for Data Warehouse Design*, Morgan Kaufmann Publishers Inc, US.
- Tillmann, G. (2017). *Usage-Driven Database Design : From Logical Data Modeling Through Physical Schema Definition*, 1st ed. Apress L. P, Berkeley, CA.
- Veselá, L. (2017). Factors Affecting the Adoption of Electronic Data Interchange, *Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis*, Vol. 65(6), pp. 2123-2130.
- Walmart (2014). *Getting Started with EDI – Implementation Guide*. Available (Accessed at 20.4.2019): <https://cdn.corporate.walmart.com/5d/8d/897b4bb84a95bb05214bf897cee3/edi-getting-started-guide.pdf>
- Wang, C. & Xu, L. (2008). Parameter mapping and data transformation for engineering application integration, *Information Systems Frontiers*, Vol. 10(5), pp. 589-600.
- Xia, Y., Ho, A.T.S. & Zhang, Y. (2000). CIMO - Component Integration MOdel, *Proceedings Seventh Asia-Pacific Software Engineering Conference. APSEC 2000*, IEEE, pp. 344-348.
- Yang, J. (2014). Construction on Data Flow Diagram and Data Dictionary of Chinese Online Examination System, *Applied Mechanics and Materials*, Vol. 687-691 pp. 2335-2338.
- Yang, L., Cao, L. & University of West Georgia/ Department of Computer Science, Carrollton, GA, USA (2016). The Effect of MySQL Workbench in Teaching Entity-Relationship Diagram (ERD) to Relational Schema Mapping, *International Journal of Modern Education and Computer Science*, Vol. 8(7), pp. 1-12.
- YTJ (2019). *Liaison Technologies Oy*. Available (accessed at 11.5.2019): <https://tietopalvelu.ytj.fi/yritystiedot.aspx?yavain=700826&tar-kiste=A41E579F76B2EFFF256F3F10FD9A6CB909DA4A0>
- Yu, H., Al-Hussein, M. & Nasser, R. (2007). Process flowcharting and simulation of house structure components production process, *Proceedings of the 39th conference on winter simulation*, IEEE Press, pp. 2066-2072.
- Zhang, H., Liu, W., Xiong, H. & Dong, X. (2018). Analyzing data flow diagrams by combination of formal methods and visualization techniques, *Journal of Visual Languages and Computing*, Vol. 48 pp. 41-51.
- Zuo, X. & Zhang, S. (2008). The Design of Optimization Strategy for Physical Data Model, *2008 IEEE International Symposium on Knowledge Acquisition and Modeling Workshop*, IEEE, pp. 336-339.

APPENDIX A: INTERVIEW QUESTIONNAIRE

Haastattelun tarkoituksena on kerätä aineistoa diplomityössä tehtävän tutkimuksen toteutusta varten. Haastattelussa keskitytään dataintegraatioiden mallinnukseen ja suunnitteluun kohdeyrityksessä. Haastattelukysymyksiin vastataan kohdeyrityksen projektien ja toteutusten näkökulmasta. Haastattelun vastaukset käsitellään luottamuksellisesti.

Haastateltava:

Haastattelun kesto:

Haastattelu-aika ja -paikka:

1. Kerro itsestäsi ja työtehtävistäsi.

Toteutusten suunnittelu

2. Osallistutko toteutusten suunnitteluun tai toteutukseen?
3. Minkä takia toteutuksia mielestäsi suunnitellaan?
4. Mitä tietoa eri sidosryhmiltä tarvitaan toteutuksen suunnitteluun?
 - a. Miten liiketoimintaprosessit vaikuttavat mielestäsi toteutuksen suunnitteluun?
 - b. Millainen merkitys integroitavien järjestelmien toiminnallisuuden ymmärtämisellä on mielestäsi toteutuksen suunnitteluun?
5. Kuvaile tyypillistä suunnitteluprosessia. Mistä lähdet liikkeelle ja missä järjestyksessä teet asioita?
 - a. Käytätkö visuaalisia menetelmiä toteutuksen mallintamiseen? Jos kyllä, millaisia ja minkä takia juuri niitä?
6. Millaisia haasteita suunnitteluvaiheessa tulee vastaan? Kerro esimerkkejä.

Solution design

1. Arvioi meidän nykyisiä solution design dokumentteja
2. Mikä on mielestäsi solution designin tarkoitus?
 - a. Mitä sidosryhmiä varten solution design mielestäsi tehdään ja mitä hyötyä siitä on heille?
 - b. Millä tasolla solution designin tulisi mielestäsi kuvata ratkaisu?

- c. Tulisiko solution designin ottaa sinun mielestäsi kantaa tietomalleihin ja mappauksiin?
3. Missä vaiheessa toteutusta solution design kannattaa mielestäsi tehdä, jotta siitä saadaan suurin hyöty? Miksi?
4. Mikäli uusi toteutus liittyy olemassa olevaan toteutukseen, tulisiko solution designin ottaa sinun mielestäsi kantaa kokonaisuuteen vai pelkästään projektiin? Miksi?

Tietomallit, sanomaformaatit ja mappaukset

1. Mitkä ovat mielestäsi yleisimmät sanomaformaatit ja sanomatyyppit?
2. Kuinka merkittävänä pidät sanomien tietosisällön ymmärtämistä?
 - a. Kuinka oleellista sinun mielestäsi on ymmärtää, miten integraatioissa kulkeva data linkittyy liiketoimintaprosesseihin?
3. Kuinka helposti sanomarakenteeseen liittyvät tiedot, kuten skeemat ja sanomakuvaukset ovat sinun mielestäsi saatavilla?
 - a. Jos eivät ole saatavilla, miltä pohjalta rakenne ja mappaukset tulisi mielestäsi tehdä?
4. Mitkä ovat sinun mielestäsi keskeisimmät haasteet mappauksissa? Kerro esimerkkejä.
5. Mitä hyötyä tietomalleista voi olla mappauksien luomisessa?