Developing a generic data translation platform

JP Liebenberg

11730110

Dissertation submitted in fulfilment of the requirements for the degree *Magister* in Computer and Electronic Engineering at the Potchefstroom Campus of the North-West University

Supervisor: Dr JC Vosloo

November 2016

Abstract

Title: Developing a generic data translation platform

Author: Jacobus Petrus Liebenberg

Supervisor: Dr J.C. Vosloo

Keywords: Translation Platform, Design Science, Ontology, Ontology Mapping, System of Systems

The world is becoming increasingly more data orientated. This can be seen across all industries and is evident when looking at the global Big Data drives. This is also the case for the industrial sector. Increasingly more data gets generated in this industry and the value obtained is becoming increasingly more important. The problem, however, is that data comes in different forms and formats and these also tend to change as the underlining technology changes.

To perform analysis on data, with constantly changing formats, is a difficult and time consuming task. It is easy to motivate the development of a platform that translates the data from the different formats into a predefined standard format. However, the effect that the environment plays has in the translator design further presents a number of difficult questions. Questions such as how will the users of this platform dictate what is required from a platform, or how will the type of data influence the design of a platform like this, arise. Answering these questions with the goal of designing a platform that can solve the particular translation needs is the focus of this study.

As part of the study other pre-existing translation platforms were evaluated and found to be inadequate to address the specific requirement. The literature further indicated that large complex systems such as these should be developed by decomposing the complete system into smaller sub-systems thereby modelling the complete system as a system of systems. In order to achieve this, there needs to be a form of interoperability (ability to exchange information and act upon each other) between the smaller systems. This system of systems concept was used in the design of the platform.

In researching how the environment (the one the platform is used in) would influence the design of the platform, a Design Science Research approach has been followed. Design Science Research states that similar to a natural scientist doing research on some phenomenon in nature by studying this phenomenon in the environment it occurs (e.g. Newton researching gravity by

studying the fall of an apple from a tree). So too can the Design Science Researcher do research on some artefact by studying the workings of that artefact in its environment.

By following a Design Science Research approach it was possible to identify how the platform interacted with the rest of the environment wherein the translator was used in as well as what requirements the environment imposes on the platform. It was further possible to study the effect of the nature of the data on the platform and how it was used. By knowing this effect it was clear how the design of the platform should be changed in order to accommodate the nature of the data.

The study indicated that by using Design Science Research it was possible to study the effects that the external environment has on the translation platform design. With this information, the translation platform was designed for a specific industrial application. The implementation of the translation platform automated the manual processes currently used. The result of the study was validated by measuring the reduction in man-hours needed between using the platform and doing the translation manually. Using the platform saved the company enough man-hours that a full-time graduate employee could be freed up to do more important work.

Acknowledgements

I would like to extend my gratitude to:

- Enermanage and HVAC International for funding the research and providing all the data and computational resources.
- Dr. J.C. Vosloo and Dr. S.W. van Heerden for both there guidance and reviewing the document.
- Dr. J.N. du Plessis for his insights and guidance.
- A. Pienaar for proof reading the document.
- Prof. Machdel Matthee for her guidance, support and insight, showing me what it means to do research.
- My wife for her love and support without which this would never have been possible.
- My parents for supporting me throughout my studies and providing me with the best opportunities in life.
- Last, but not least, to all my family and friends for their support and friendship.

Table of Contents

List of Tables

List of Figures

1. Background on Study

1.1. Introduction

As technology advances and gets more complex so do the systems that make up these technologies. One of the more dominating driving forces in technology advancement is the ever-increasing speed at which data can be computed and the ever decreasing cost of performing these computations (Loebbecke & Picot, 2015; Lokers, Knapen, Janssen, van Randen, & Jansen, 2016). It is a classic example of where the demand closely follows that which is physically possible and economically viable. As these two factors change, businesses are forced to adapt or be left behind only to be replaced by another business that took advantage of being able to compute more data faster and cheaper than ever before (Loebbecke & Picot, 2015).

This has led to incredible technologies being capable of performing tasks that were previously thought to be impossible for machines to perform. The Big Data drive that is occupying all major industries out there is evident of this. Everything from relieving traffic through self-driving cars (Zakharenko, 2016), to improving the effect of agricultural practises on the environment (Lokers et al., 2016). Having more data and being capable of processing this data means being capable of deriving more accurate conclusions faster.

This allows businesses to be agile (Larson & Chang, 2016) and respond to changes in the market quicker while also allowing conclusions that would have been impossible to achieve without this new data. All of this gives rise to an increasing need to share data between different systems of increasing complexity (Wang, Xu, Fujita, & Liu, 2016).

As with most other industries, the mining industry is becoming more data oriented (Perrons & McAuley, 2015). The need to acquire more data and integrate that data with more aspects of the mining operations is growing at a rapid pace; doing this allows for more sophisticated machinery, optimised mining and efficient use of energy resources.

The value of being able to incorporate data from across the whole mine and acting on the information extracted from that data is becoming more evident. Being able to draw conclusions from data across multiple systems and feeding that information back to the some or all of the systems allows the systems to function together as a larger system accomplishing a common goal (Laurentis et al., 2007).

This can be explained with the help of the following example. Say there exists an operator on a mine; this operator is in charge of three independent mechanical systems. The main energy source of all three systems is electricity and to ensure that the electricity bill can be paid monthly an energy budget was drawn up for the total energy consumption over the three systems. The operator has a lot of experience with the three systems and he knows in order to maximise the overall production of all of the systems he has to award 50% of the total energy budget to the first system, 30% to the second system and 20% to the third system.

The problem, however, is that the average ambient temperature of the immediate environments of the three systems affects the energy requirements of the systems in different ways. This results in changes to the optimal energy division across the three systems in a manner that is very hard to predict. If the production and energy consumption data from all of the systems are logged daily then conclusions based on the data can be extrapolated in the form of a report. The operator can then make use of this system performance report to adjust for the effect ambient temperature had on the three systems to ensure optimal energy is maintained and energy budgets are adhered.

This is a simple example where the data from different systems are grouped and analysed together thereby helping the systems to act together to accomplish a common goal. They all act together as different parts of a larger system where the goal of the larger system is to optimise the overall production. Systems like this where the overall systems are made up out of smaller systems is called a System of Systems (SoS) (Johnson IV, Tolk, & Sousa-Poza, 2013).

When integrating data across a whole SoS one has to keep in mind that the data will be in different formats (Yaqoob et al., 2016). Different technologies will generate different data. The data generated by a pump will be different to the data generated by a compressor. A temperature sensor from vendor A may provide temperature reading in 5second intervals while a temperature sensor from vendor B only provides temperature readings in 10second intervals.

Differing data formats can create potential problems when conclusions need to be made across the data from different technologies (Chang et al., 2016). The entity (human or machine) that wants to make these conclusions needs to "understand" all of the different data types and formats. When comparing the data from one data format to the data from a different format it needs to know how to interpret the data from both formats to ensure that apples are compared to apples.

This does not seem like such a big problem when the data formats and types are limited but as stated before there is a constant drive to include more and more data from more and more systems and technologies. This leads to an increasing number of different data formats and types that needs to be reckoned with. What makes this even worse is the fact that as technologies change so do the data formats and types. The formats that have already been handled might also change making the whole process of "understanding" the data dynamic in nature.

Therefore automating the whole analysis and drawing conclusions from the data part of this process is challenging. When automating this it means that machines are required to "understand" all of the different data types and formats. Although there have been huge advances in the field of machine learning and artificial intelligence (Cantu-Ortiz, 2014), technology has not yet advanced to the point where the machines can teach themselves all the different data formats. This means we humans still need to give the machines the tools they need to "understand" the different data formats.

A popular way found in literature to do just that (help computers "understand", to some degree, the information that is given to them) is through the use of ontologies (Skjæveland, Giese, Hovland, Lian, & Waaler, 2015). An ontology is a term borrowed from philosophy (Lassiter, 2016) by the computer science community.

In philosophy an ontology is used to try and define the essence of something. It is used to try and capture that which defines the very existence of something to the full (Gruber, 1993). In computer science this term is used for something that gives a high level declarative description of the structure of the information being provided (Orpha Cornelia Lombard, Gerber Co-supervisor, & van der Merwe, 2014; Skjæveland et al., 2015).

If ontologies for two data formats are created (an ontology for each data format) then an ontology map that describes the relation between the two ontologies can be created (Forsati & Shamsfard, 2016). This ontology map can then be used by a computer to "understand" the relation between the different data formats (Mecca, Rull, Santoro, & Teniente, 2015). This is known as ontology mapping or ontology translation and it can be used to help computers draw conclusions from data across multiple systems with multiple data formats.

The current technology used is becoming increasingly complex and the data generated is becoming increasingly diverse. As computation becomes faster and cheaper there is a need to incorporate more and more data into the conclusions drawn from the data. The analysis we

performed becomes increasingly more complex as more data formats are included in the analysis. If the conclusions made from the data are to be automated then the computer doing the analysis requires the tools to "understand" the different data formats and the relationships between them. This can be done by using ontologies and ontology mappings.

1.2. Problem Identification

Esco's (Energy Service Company) are businesses that work on a contractual basis for heavy industry. They provide energy solutions which include but are not limited to the reduction of energy consumption of a business. Esco's have multiple clients from various industries. As part of the services that the Esco's offer, they need to perform analysis on the data from a wide range of systems from clients. This places the Esco's in a position where they inherit the environment as described in 1.1.

Within the internal structures of the Esco, each business can be modelled as a SoS where different sub-sections (modelled as the different subsystems of the SoS) work together to achieve a common goal. The different subsystems usually (but not always) correspond to a different physical branch of the business.

Each of these subsystems can in turn also be modelled as a SoS where the different technologies or groups of technologies (like the pumps, compressors, etc.) can be seen as the different systems of this SoS. A graphical presentation of this can be seen in **Error! Reference source not found.**. Of course, each group of technologies can again be modelled as a SoS but such a level of abstraction is not needed for this study. For more information on the definition of a SoS and the system hierarchy this imposes see section **Error! Reference source not found.**.

Apart from the different subsystems there usually exists entities that accumulate data from the different technologies making up the subsystems. These entities can be anything ranging from a collection of physical meters installed on location, a connection to a database or data from an external metering company a mine uses to store some of its information extracted from the site's Supervisory Control And Data Acquisition (SCADA) system. From here onwards these entities will be referred to as data accumulators.

The data accumulators can be seen as a centralised store for storing and retrieving data. They are a means of gaining access to the data contained within a location and they give all of this information in a single format that is usually unique to that specific accumulator. In this, they act very much as a translator translating all of their information into a single format before relaying that information onto whoever needs it.

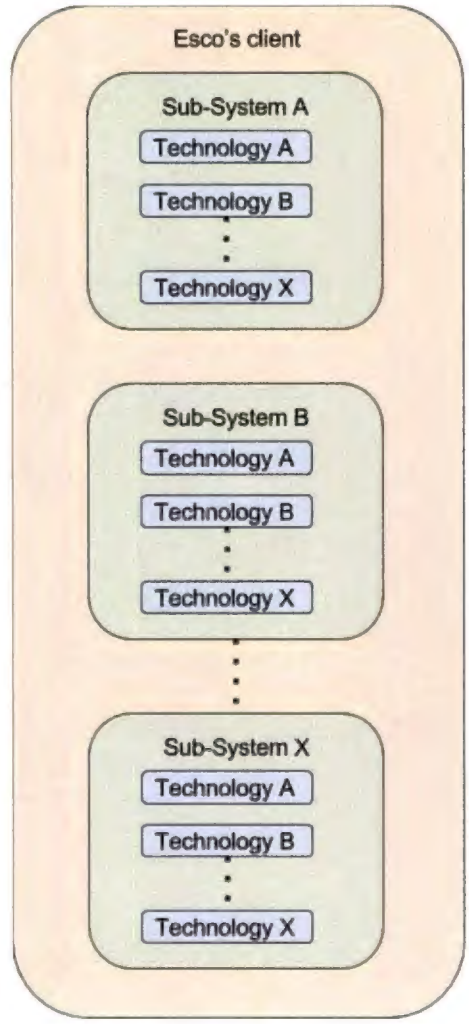

Figure 1: Graphical presentation of the SoS breakdown of a single client

The data accumulators help, to a certain degree, with the problems surrounding the dynamic and diverse nature of the data from the industry. When the type or format of the data from a certain technology changes or more data types from that specific piece of technology needs to be retrieved the data accumulators can absorb this change and still relay the data using the initial format. If however, this was possible in all cases the world would have been a lot less complicated and this study would have been null and void.

The data accumulators can only absorb the change if the change in the data format or type is minor. If the change between the data types is large enough or if the format of the new data that needs to be included differs to an extent that is great enough the format can still be changed or a new data accumulator will be used altogether.

The point being made is that although the data accumulators help to buffer or absorb the effect of the ever-changing nature of the data in the industry, they do not eliminate the issue

completely. The data types and formats received from the industry are still, and will for the foreseeable future be ever-changing, even if the data is only retrieved through the data accumulators. It might not be to the same extent as when the data is accessed directly but it will continue to be a problem that must be addressed.

The analysis that needs to be performed on these data sets are usually system specific which means that the calculation that is needed for the different systems has to be custom designed for each one of the subsystems.

Esco's requires the use of a reporting system that is put in place to do system specific calculations needed for their analysis. This reporting system receives data from the data accumulators and performs analysis. The reporting system will need to do calculations on data from various parts of different systems and as such will have to "understand" the data from numerous file formats.

Setting up calculations over such a wide variety of data formats is a tedious job and when the file formats changes all of the calculations dependent on that format has to be updated to allow for the change. To overcome this a new system will be introduced.

This new system will be tasked with translating all of the data that is received from the data accumulators into a predefined standard format (hereafter referred to as the standard format). Having a system like this allows for the reporting system to only support or "understand" one format. When the format of a file received from one of the data accumulators' changes, only the part of this new translation system that translates that format needs to change.

All of the calculations in the reporting system can then be left unchanged. The design of the standard format is beyond the focus of this study but what is important to know is that the standard format is tag based. A tag defines a grouping of information all originating from the same source, giving information of that source at different time intervals. The temperature readings of a particular piece of technology would be one example of a tag. For more detail on this see section **Error! Reference source not found.**.

It is this system (the system placed in-between the data accumulators and the reporting system) that is the subject of this study. How to design and implement a system that will function well as a platform for translating files that are as diverse and as dynamic as the one's received from the data accumulators. This platform will need to allow for the continuous creation and update

of translators while also allowing the use of these translators to translate the data in an effective way for use by the reporting system.

The platform needs to allow the personnel to monitor and manage the translation processes while also trying to automate the translation processes to the furthest possible degree. The platform must allow for the creation of translators to be able to translate a diverse set of file formats but also take into consideration the available human resources in the company capable of programming. The platform needs to save the translated data to ensure that the reporting system can use the data whenever it needs to.

One thing to keep in mind is that the function of this system can ultimately be performed by human resources alone. To translate from one file type to another can be done by hand. It is a task that is very human intensive and repetitive and would take a lot of time but it is in no way a problem that cannot be solved by a lot of people doing the translations manually. The function of this system, this platform, is then ultimately to reduce the dependence of the translation processes on human resources doing the translations manually. Ultimately this is an automation problem.

Error! Reference source not found. gives a graphical representation of how the systems at the Esco's client, the data accumulation, the translation platform and the reporting system can all be modelled together as a SoS all working together to help integrate all of the different aspects of the mine. The red arrows represent the flow of data before any analysis has been performed on them while the blue arrows represent the conclusions drawn from the analysis performed on the data from the Esco's client.

It (**Error! Reference source not found.**) shows how the data originates at the different technologies used in the subsystems of the Esco's client. How the different data accumulators collects the data from the different technologies. The accumulated data in all of its different file formats is fed to the translation platform to be translated into the standard format. The reporting system now uses the data translated into the standard format to do its analysis and then the conclusions drawn from that analysis is fed back to the subsystems at the Esco's client to ultimately alter the way those technologies are used.

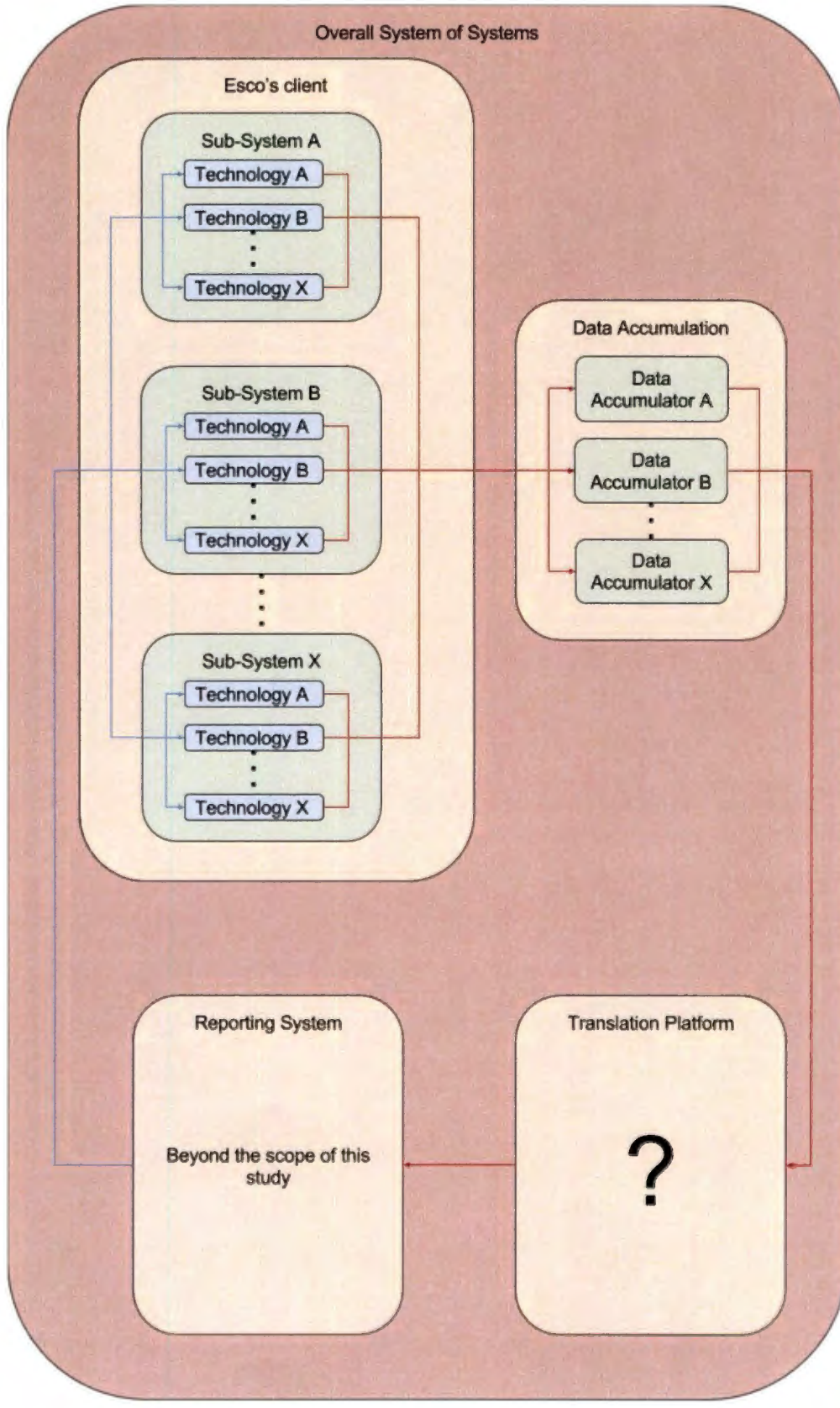

Figure 2: A Systems of Systems breakdown of the flow of data.

What makes this problem more complex and what was unknown at the start of the study is how the environment the platform needs to operate in would affect the design of the platform. Placing this platform in this environment and having this platform interact and play its part within the larger SoS adds a vast degree of complexity to the problem; especially when these interactions are not yet fully understood.

These interactions, the way the platform needs to behave, the way this platform needs to complement the rest of the systems given in **Error! Reference source not found.**, the exact way in which this platform should help solve the overall problem of the larger SoS. This is what this study focusses on. Given all of this, the research questions in the following section were drawn up.

1.3. Research Questions

The rate at which data gets generated is increasing and extracting value from this data by performing analysis on it is becoming increasingly more important. To perform this analysis, it is important to have all of the data in the same format. This is difficult to achieve because the data originates from various sources and because the underlying technologies of these sources often change the formats of the data do so as well. A solution to this problem would thus have to take all of this into consideration. To help in the development of this solution the following research questions have been developed:

Main question: How would the design of such a solution look?

1.1. What role does the nature of the data play on the design of the solution?
1.2. What is the effect of the internal structure of the company on the design of the solution?
1.3. How do the available human resources change the design of the solution?

1.4. Summary of the Research Approach

To find a solution to the problem described in the previous sections research is needed not only into how to develop translators or how to develop a platform that can be used to create different, diverse translators. Research is also needed into what is needed from a platform like the one described, in the environment that was described. What is needed from a platform like this when the function of this platform is not only evaluated by how well files get translated or how effectively different translators get created but how well this platform functions as part of the larger system. Research is needed into what exactly is required from a platform like this when

it needs to function as part of the larger system in the environment described. For this Design Science Research has been chosen as the research approach.

When doing Design Science Research, an artefact is created. Research is done when the artefact is designed, placed in its environment and studied or evaluated for how well it functions within the given environment. The knowledge gained from the design and evaluation can then be used as input for a new design. The new design is then reinserted into its environment to again be studied and so design science has this continues cycles of design, testing and evaluation until the artefact is evaluated to function well enough within its environment or the research questions has been answered.

This can further be explained by the use of an example: People were able to build aeroplanes that can fly long before they knew exactly what was needed to make an aeroplane fly. Not only were they able to build aeroplanes before they knew what was needed but through continuously designing, building and evaluating how well the aeroplanes flew they were able to develop theories of what is needed for an aeroplane to fly. It was for this reason that Design Science Research was chosen as the design approach for this study.

By continuously designing, building and testing a platform, in the environment that the platform needed to function, it was possible to not only test the function of the translators and the creation thereof but also the function of the complete platform as it plays its part in the bigger system. By doing this it was possible to determine, to develop theories if you will, of what is needed or required from a platform like this and to then design the platform accordingly. For more information on Design Science and how it was used to conduct research in this study see section **Error! Reference source not found.**.

The steps that were taken to conduct this study were to first look at what other platforms exist that could be used to create the platform needed and the advantages and disadvantages of these systems were evaluated. This can be found in section **Error! Reference source not found.**. Next, the literature was studied for how to conduct the design of a system that needs to be part of a larger system. How to model a large complex system such as a SoS and by doing that reduce the complexity of the overall system. How to ensure all of the different sub-systems work together to accomplish the overall function of the larger SoS and how to ensure interoperability between the different sub-systems. How to use ontologies to help with the translation between different file types as well as the role ontologies can play to help with data

interoperability. This can be found in Section **Error! Reference source not found.**.

Next, Design Science was used to conduct research into what is needed from the platform using the knowledge gained from the literature as inputs in the design facets of the Design Science cycles. Doing this allowed for the creation of a platform that satisfies the role that it needs to play in the environment it was placed in by firstly discovering what is the exact role it needs to play. It is therefore the exact role the platform needs to play that will be given as the outputs of the Design Science Research. The Design Science cycles can be found in Chapter **Error! Reference source not found.**.

1.5. Overview of Document

Next an overview of this document will be given.

Chapter 2 – Literature and Research Approach

In this chapter literature relevant to this study will be given and the Design Science Research Approach that was followed to conduct the research in this study will be clearly stated.

Chapter 3 – Design Cycles

This chapter shows how the Design Science principles were implemented to do research into solving the problem identified. The result of this research is given along with the results of the validation and verification of the solution.

Chapter 4 - Conclusion

Here the research questions and how the solution addresses these questions is revisited. A discussion and suggestions for future research is also given.

2. Literature and Research Approach

2.1. Introduction

Given in this chapter is the literature that has been studied to help with the design of a solution to the problem given in section **Error! Reference source not found.**. This section starts by describing some of the other platforms that might have been used to solve the problem described in section **Error! Reference source not found.**. The advantages and disadvantages of each of these platforms will also be detailed.

In section **Error! Reference source not found.**, literature is given where the knowledge gained from that literature was used to help design the platform. The literature shows how to reduce the complexity of large systems by breaking the large system up into a hierarchy of systems in a SoS approach. It shows the need for interoperability between sub-systems and the role that ontologies can play in ensuring data interoperability and data translation.

2.2. Translation Platforms

2.2.1. Background

In this section, existing software packages/platforms will be evaluated on its ability to solve the problem identified in section **Error! Reference source not found.**; they will be evaluated according to how efficiently they can be used to translate files from the diverse file formats (as discussed in section **Error! Reference source not found.**) into the predefined standard format.

The idea is to use these software packages or platforms to create different translators (an entity that perform the actual translation); one translator for the translation of each file format into the standard format. This is illustrated in **Error! Reference source not found.**. The advantages and disadvantages of each of them (as it applies to solving this problem) will also be given.

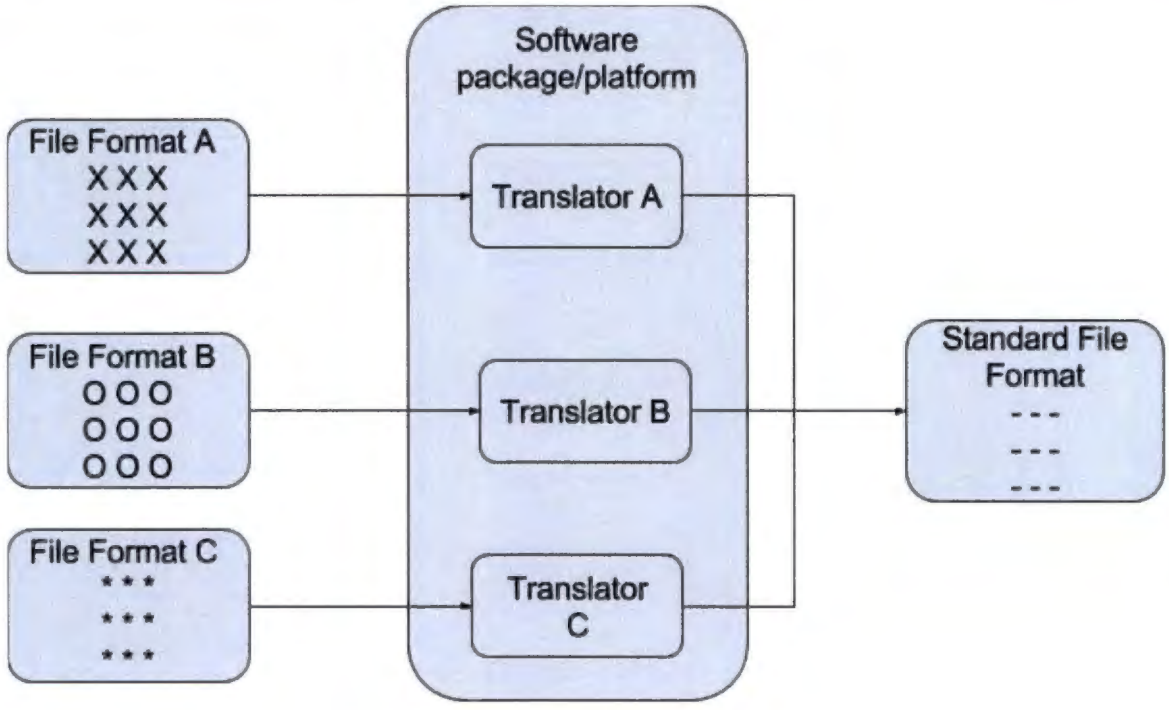

Figure 3: Flow diagram of translation process

Each of the software packages/platforms will be evaluated according to the following criteria:

- The diversity of the file formats that can be translated using the software package/platform. That is to say the diversity of the translators that can be created using this software package/platform. - (Translator diversity)
- The speed and ease of creating these translators using the different software packages/platforms. Here the complexity of creating and using the translators plays a vital role. - (Complexity of creating translators)
- The extent to which the software package/platform can be used to automate the translation processes. To what extent can the process of using these translators, to translate their respective data formats, be automated. - (Degree of automation)
- The technical skill required to monitor and manage the translation processes (using the translators to translate data). - (Managing skills needed)
- The cost of using the software package/platform - (Cost)

2.2.2. Pentaho Data Integration

Pentaho Data Integration is a software package that is used to perform various operations and analytics on data. This makes it very popular in the business intelligence and big data environments. The focus of this study is, however, its capability to build translators.

In Pentaho Data Integration, you have the option to use the various pre-build operations (operations and functions that have been created by the developers of Pentaho Data Integration in order to manipulate data). Each of these operations have a certain degree to which the operation can be configured. This gives each operation some form of customizability but the real power of Pentaho Data Integration comes from the ability to chain these operations together to perform a series of operations on the data. For the use of these operations, no programing skills are required. These chains of the prebuilt operations will then be used to create the translators needed. More information on Pentaho Data Integration can be found at: http://www.pentaho.com/product/data-integration

Advantages:

- Building the translators using the pre-build functions is simple. This makes creating and modifying the translators quick and easy.
- After a translator has been created, it is possible to automate translation process.
- Because of the pre-build functions and the simplified way in which these functions are implemented, the required technical skills to both create the translators and manage the translation processes is minimal.
- The cost to use Pentaho Data Integration is free.

Disadvantages:

- Having to use the pre-build functions limits the diversity that can be achieved in developing translators. If an operation is needed that is not included within the prebuild functions or that cannot be created by using a combination of the pre-build functions, this platform will add no value.
- Although it is possible to automate the translation processes of the different, created translators, the automation is time-based (meaning the translation processes can be programmed to run at set time intervals). Ideally the automation of the platform should be event-based (meaning the translation processes should run when a new file is received by the platform). Time-based automation can be converted into event-based automation by using the time-based automation to check every few seconds or minutes if the event has occurred (in this case it would check if a new file was received) however this makes the whole automation process unnecessarily complicated.

2.2.3. Nucleon BI Studio

As the name suggest, Nucleon BI Studio is a Business Intelligence support package. Although it is capable of some data manipulation the core focus is that of data analytics and visualisation. Nucleon BI Studio is a database-based solution. This means that to use it to manipulate data, the data first needs to be uploaded to any of its supported databases.

Due to the fact that Nucleon BI Studio is focused more on database operations, performing file operations is complex. To translated data from one file format into another using Nucleon BI Studio the file first needs to be uploaded to a database. The required data manipulation then needs to be performed on the data within the database and then only can the data be exported into the standard file format. Nucleon BI Studio has a limited number of file formats into which the data can be exported. This means that if data must be exported into a file format that is not supported by the Nucleon BI Studio, this platform cannot be used. More information on Nucleon BI Studio can be found at: http://nucleonsoftware.com/products/nucleon-bistudio

Advantages:

- Nucleon BI Studio supports both time based and event-based automation of the translation processes.
- There is a free version of Nucleon BI Studio available.

Disadvantages:

- The diversity of the translators that can be created using this software package is limited to the file formats supported. This makes the diversity low.
- Data must first be uploaded to a database before it can be manipulated. This makes the process of creating translators unnecessarily complex and time consuming.
- Because the data must first be uploaded to a database the monitoring and managing of the translation processes requires a high level of technical skills.

2.2.4. FME

FME is a software package designed for data integration and translation. It is focused, but not limited to, spatial data. It can be used to integrate and translate between a vast amount of different data sources and formats.

As with Pentaho Data Integration, FME has a set of pre-built operations from which one can build chains of operations in order to achieve the desired data manipulation.

To perform translations on files with FME, the data source is chosen to be the file type that must be translated. The exact translation that must be performed on the data is then designed using the existing pre-built translations and chaining them together. When the desired translation is achieved the output is then chosen to be the desired output file. Being able to perform translations on the data by chaining the pre-built operations is powerful but because only the pre-built operations can be used to build this chain it still limits the type of translations that can be achieved using FME. More information on FME can be found at: https://www.safe.com/how-it-works/

Advantages:

- As with Pentaho Data Integration, data is manipulated using pre-built functions. These pre-built functions simplify the creation of new translators.
- Event-based automation is possible with this platform.
- The technical skills needed to monitor and maintain the translation process is minimal due to the pre-built functions.

Disadvantages:

- As with Pentaho Data Integration; having to use the pre-built functions to create the translators limits the functionality and diversity of the translators that can be created.
- FME is not free. The basic package starts from a once of $4300.

2.2.5. Summary

There are already very powerful platforms that are used in the corporate environment. These platforms have various pre-built functions available which make it very easy and fast to design new translators on top of the respective platforms. This is arguably their biggest advantages but also their biggest flaw. Having these pre-built functions also limit the type of translations that can be built using these platforms.

Although most of the platforms reviewed can perform most of the translations that are needed in this study, none of the platforms could perform all the required translations. This calls for the design of a new platform where the type of translations that can be built on top of this platform are as diverse as may be required.

Given below is a table summarizing the software packages/platforms and how they were rated against the evaluation criteria set out in section **Error! Reference source not found.**.

Key to **Error! Reference source not found.**:

- X – Unsatisfactory
- * - Can be used but is unnecessarily complex
- O - Satisfactory

	Pentaho data integration	Nucleon BI Studio	FME
Translator diversity	X	X	X
Complexity of creating translators	O	X	O
Degree of automation	*	O	O
Managing skills needed	O	X	O
Cost	O	O	X

Table 1: Comparison between translation platforms

2.3. Design Literature

2.3.1. Introduction

Because the platforms that were evaluated were all found to be lacking in their ability to build all the required translations the decision was made to develop a new platform. The next section discusses what was found in the literature.

2.3.2. A System of System

Many systems today are complex in nature. To try and reduce some of the complexity these systems can be designed as a SoS. This will split the design of the overall system into the design of a series of smaller systems each being less complex than the overall system (Clark, 2009). Once all of the smaller systems have been designed the overall system can be designed by using the smaller systems as building blocks making the design of the overall system less complex.

Defining a System

To define a SoS we first need to understand what a system is. The concept of a system is explained by Johnson IV et al. as follows:

A system is "An ensemble of autonomous elements, achieving a higher level functionality by leveraging their shared information, feedbacks, and interactions while performing their respective roles."

(Johnson IV et al., 2013)

Here Johnson IV et al. explains that a system is made up of different elements. These elements work together in an autonomous manner to achieve a common goal by performing their respective roles. They are dependent on the shared information, feedback and interactions of the other elements and as such will not be able to function on their own. Knowing this we can begin to define a SoS.

Defining a System of Systems

A SoS can be defined in many ways. Listed below are two of the more popular definitions found in literature:

"A System of Systems is integrated, independently operating systems working in a cooperative mode to achieve a higher performance."

(Tannahill & Jamshidi, 2014)

"System of systems applies to a system-of-interest whose system elements are themselves systems; typically these entail large scale inter-disciplinary problems with multiple, heterogeneous, distributed systems."

(Clark, 2009)

The goal of a SoS is achieved by making use of the functionality of the different systems forming part of the overall main system. This is done by creating a system that allows for different systems to function together in a cooperative manner all working towards a common goal. Each system can still function on its own, executing its individual goal independently from the other systems but having an overall system that combines the functionality of all the independent systems which allows for achieving a higher goal that is different but dependent on the individual goals of the different systems (Stary & Wachholder, 2016).

The key concept to understand here is that the goal of a SoS, just like in a normal system, is dependent on the "shared information, feedbacks and interactions" of the "autonomous elements" making up the overall system, but unlike a normal system some or all of these elements can be modelled as a system in themselves acting independently from the other elements (or systems) to achieve their respective goals (Johnson IV et al., 2013).

In a SoS there exists thus a hierarchy of systems where each level in the hierarchy can be deconstructed into another SoS until a point is reached where the elements in that system cannot be deconstructed into further independent systems (Johnson IV et al., 2013).

2.3.3. Interoperability within a SoS

As stated before, a large complex system can be made less complex by breaking the system up into smaller systems and using these smaller systems as building blocks for the overall system (Arasteh, Sepasian, Vahidinasab, & Siano, 2016). Although this is true it brings forth a new dimension of complexity. All of these systems now need to work together to accomplish the larger overall goal. The systems in a SoS need to function in a cooperative manner and for that there needs to be some form of interoperability between the different systems (Jamshidi, 2010; Johnson IV et al., 2013; Tannahill & Jamshidi, 2014; Weichhart, Guédria, & Naudet, 2016).

The different systems need to have some form of communication between them (Ge, Hipel, Yang, & Chen, 2014). They need to share information among themselves (Johnson IV et al., 2013), an event in one of the systems needs to be able to cause an event in another system (Stary & Wachholder, 2016). This communication relies heavily on a standard that has been decided on between the systems (de Farias, Roxin, & Nicolle, 2016; Johnson IV et al., 2013; Stary & Wachholder, 2016). All of this can be addressed by designing each of the smaller systems from the start with the notion that there needs to be some sort of interoperability between the systems (Ge et al., 2014).

Defining interoperability within a System of Systems

There are different definitions for interoperability between systems within a SoS but according to Stary and Wachholder there are two well-known definitions that are widely accepted (Stary & Wachholder, 2016). The first is from the Institute of Electrical and Electronics Engineers (IEEE) and the second is from Tanenbaum and van Steen in:

Interoperability is "The ability of a system or a product to work with other systems or products without special effort on the part of the customer. Interoperability is made possible by the implementation of standards."

(IEEE, 2010)

"Interoperability characterises the extent by which two implementations of systems or components from different manufacturers can co-exist and work together by merely relying on each other's services as specified by a common standard."

(Tanenbaum & Van Steen, 2007)

From both these definitions, it can be seen that heavy emphasis is placed on the fact that there needs to be a common standard between the different systems. If however two systems exist

that do not share a common standard, a different system can also be designed to handle the interoperability between those two systems. By doing this the third system acts very much as a translator between the two systems.

On the one side, the third system will share a common standard with the first system and on the other side it will share a common standard with the second system. The third system then having access to both common standards can translate between the two standards. This is accomplished in the system designed by Stary and Wachholder (Stary & Wachholder, 2016).

2.3.4. The use of Ontologies

One way to approach the design for interoperability is to ensure data interoperability (de Farias et al., 2016). If the only interaction between systems is through the data they use or generate then ensuring data interoperability will ensure interoperability between systems. For this ontologies have been widely used (de Farias et al., 2016), due to the interesting way ontologies provide formal representations of the data. Having these formal representations makes it possible to implement reasoning upon these representations based on the logic embedded in them (de Farias et al., 2016).

Defining Ontologies

The word ontology is borrowed from philosophy. In philosophy, an ontology is used to try and capture the existence of something (Orpha Cornelia Lombard et al., 2014), a systematic account of existence (Gruber, 1993). A complete ontology of something captures the essence of what that something is and defines it completely. In Lombard's study (Orpha Cornelia Lombard et al., 2014) a summarization of somewhat overlapping terms describing the traditional meaning of ontology is given as follows:

- Meaning and nature of things
- Trying to understand the basic structure of the world
- Investigation into being
- Study of nature of being
- Structure of reality
- Study of reality
- An account of existence

Given the fact that the term ontology in computer science is borrowed from philosophy it is not surprising that the definitions for ontology in computer science are very similar to that in philosophy. One such a definition is given by Gruber in (Gruber, 1993):

"An ontology is an explicit specification of a conceptualization."

(Gruber, 1993)

This "explicit specification of a conceptualization" was adapted by the computer science domain out of a need that arose to share knowledge (Orpha Cornelia Lombard et al., 2014). In computer science, it is thus something that defines a specification of a concept with the purpose of using that specification to share knowledge (Cai et al., 2016).

It is important to understand here that although it can be argued that an ontology is within itself a form of knowledge (the knowledge of how to formally represent a conceptualization), it is not the knowledge being shared itself but only a formal representation, a framework, wherein that knowledge can be shared. The specification of a concept is not the concept itself but only a formal representation wherein the knowledge contained in that concept can be shared.

This can be further explained by the use of an example. Say that the knowledge that needs to be shared is that a car of make x and model y is travelling at a speed of 80km/h in a direction of 5 degrees north. To do this the ontology in **Error! Reference source not found.** can then be drawn up. This ontology can then be used to share this knowledge as is done in **Error! Reference source not found.**. Here the ontology (**Error! Reference source not found.**) is not the knowledge that a car of make x and model y is traveling at a speed of 80 km/h in a direction of 5 degrees north but it is a representation/framework/specification for that knowledge and this specification can then be used to share that knowledge (**Error! Reference source not found.**).

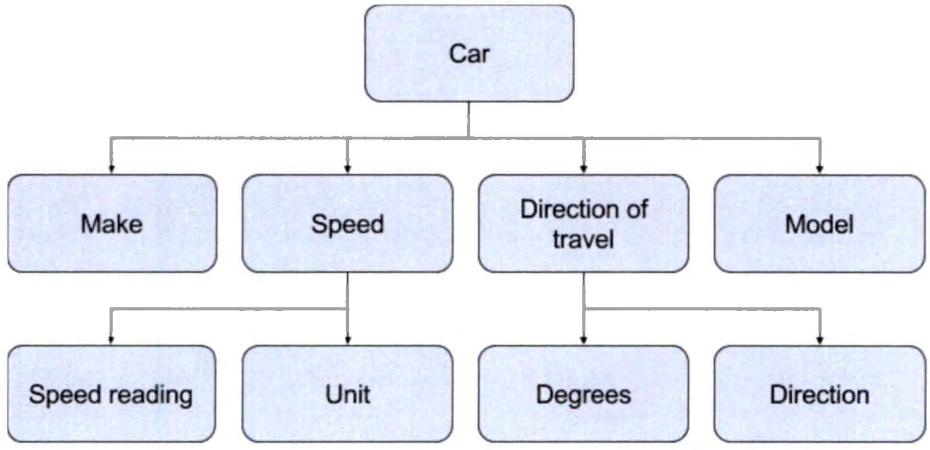

Figure 4: Car ontology

When building ontologies like this it is important to keep in mind that there has to be some sort of balance between how well the ontology represents the concept and how much information needs to be shared (Orpha Cornelia Lombard et al., 2014). It is here where the traditional definition of ontology, as it is found in philosophy, diverges from the definition used in computer science. The traditional definition seeks to capture the existence of an object and represent it to its full state of being. In computer science an ontology will only be developed until all of the knowledge that needs to be shared can be captured within the ontology.

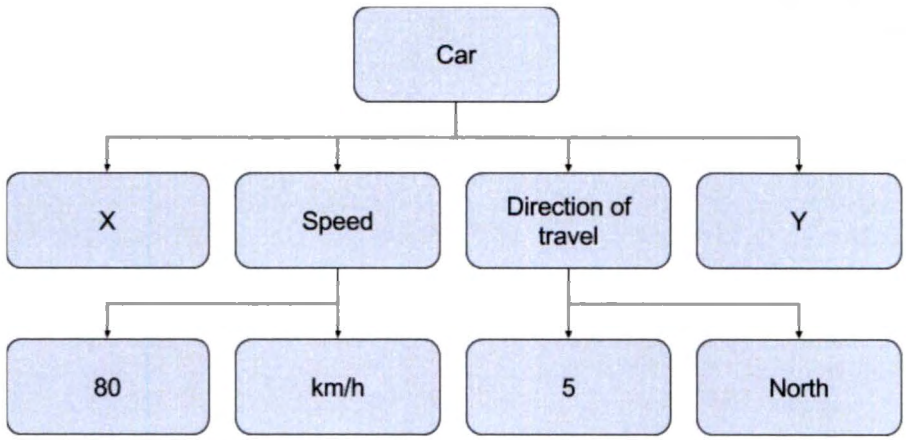

Figure 5: Using the car ontology

In the example of the ontology for a car everything from the status of the fuel tank to the speed and temperature of the engine to the colour of the car could have been included in the ontology but seeing as this information would not have been used it would have carried no value and as such would only be a waste of computer resources. It is however still important to understand the traditional definition. Knowing and understanding the origin of ontologies in philosophy will lead to a better ontology development in computer science.

Ontology Mapping

Ontologies can be used to not only share information within a system but also to share information between systems, this makes it apparent that there might arise a need to share data between different ontologies. Being able to do that will go a long way to ensuring interoperability between systems. De Farias et al. go as far as to say that in his work interoperability can be defined by "the capability to share data between different ontologies" (de Farias et al., 2016).

One way to share data between ontologies is to set up semantic links between the different ontologies (de Farias et al., 2016). This is called ontology mapping, ontology matching or

ontology translation. For example, if there is an ontology for the velocity of an object and that object happened to be in a car we can safely assume that the velocity of the object will be equal to the velocity of the car. We can then do a simple one to one mapping between the car ontology and the ontology for the velocity of the object. This is demonstrated in **Error! Reference source not found.**.

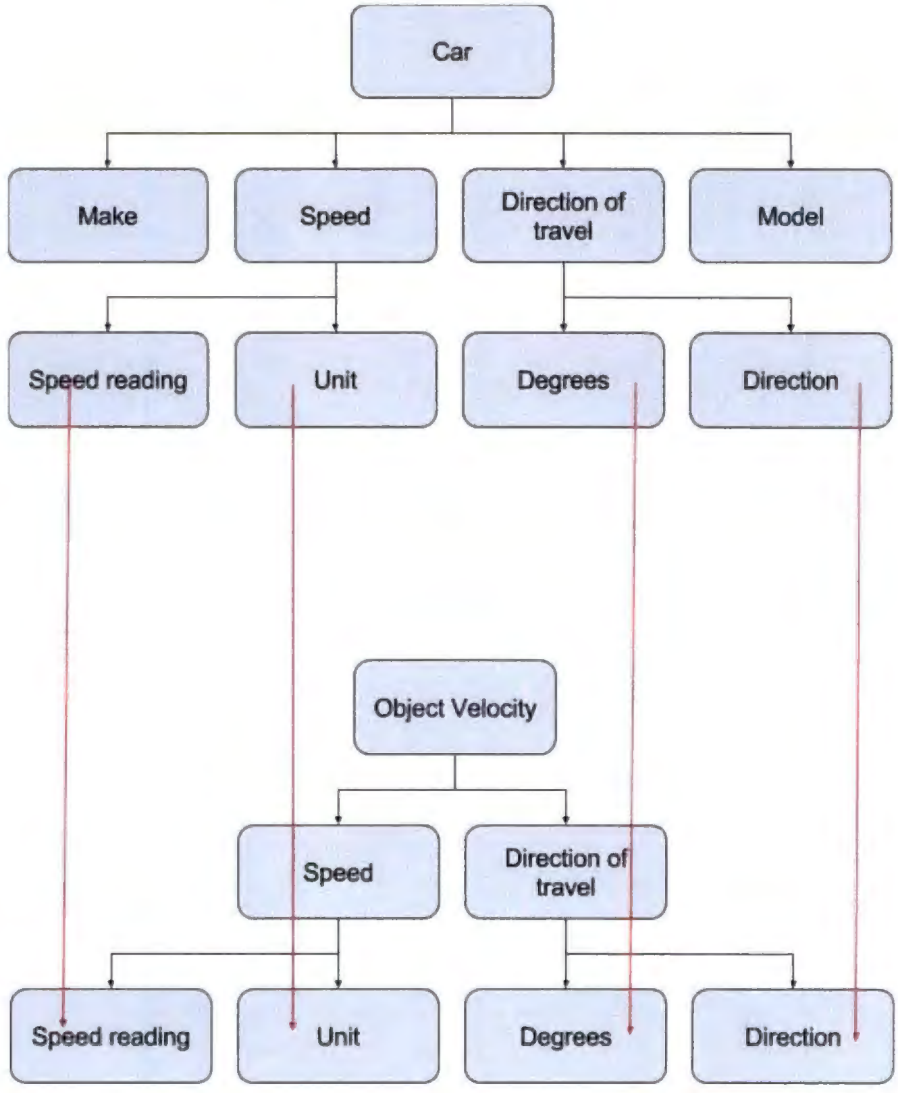

Figure 6: Ontology mapping example

The example in **Error! Reference source not found.** is oversimplified with the object velocity ontology being almost identical to the car ontology but it demonstrates the principle of ontology mapping well. This mapping can either be done automatically (by developing a set of logical rules to govern the mapping process) or manually depending on the software developer to set

up the mapping (de Farias et al., 2016). In his work Skjæveland et al. notes that there are two main things required for ontology mapping. The first is a clear description of both ontologies and the second is a set of rules describing the translation from the one ontology to the other (Skjæveland et al., 2015). In developing interoperability between systems using ontology mapping this is then the two main aspects that must be focused on and developed for a specific ontology map.

2.3.5. Conclusion

The development of large complex systems can be made less complex by designing the overall system as a SoS. Doing this requires a form of interoperability between the different smaller systems. This interoperability can be facilitated by ensuring data interoperability between the data being shared by the different systems.

Ontologies have been found to be a good means of ensuring data interoperability both within a system and between two or more systems sharing data. This can lead to a case where there is a need to share data between different ontologies. For this ontology mapping is proposed. All of this will be used to help design the platform that must be created and to help to ensure that the platform adheres to the requirements placed on it by the rest of the systems in the larger overall SoS.

2.4. Design Science

2.4.1. Why Design Science

A need existed to develop a platform that can broadly be described as a translation platform for creating and using translators. These translators need to translate files that are diverse and dynamic in nature. This platform also needs to exist as part of a larger system and play its part to enhance the interoperability between different systems on the mine by allowing analysis on the data from different systems on the mine.

It was unclear what the exact role of this platform was and how it needed to fit into the larger context of the company it would be used in. Research was thus needed into, not only how to design a platform for the design and use of translators, but also into what is needed from a platform like this in the environment as the one described in section **Error! Reference source not found.**. To solve this Design Science was used. A description of what Design Science is and how it works is given below.

2.4.2. Background

According to (Kuhn, 1970), research can be any activity that leads to a new or better understanding of the phenomenon being studied. In Design Science this phenomenon is the behaviour and interaction of an artefact in and with its environment.

In information systems and IT there are mainly two types of research activities that lead to the advancement of human understanding and knowledge. The one is the study of Behavioural Science and the other is the study of Design Science (March & Smith, 1995). The two approaches are closely linked and definitely build and interact with each other. In fact, Design Science is dependent on Behavioural Science for core theories and principles to build new and innovative artefacts. As such Design Science is reliant on Behavioural Science for its existence (Hevner, March, Park, & Ram, 2004) but there is a distinct difference.

In Behavioural Science the focus is on developing and justifying theories for the behaviour and interaction of naturally occurring phenomena between humans, organisations and technology (Hevner et al., 2004).

In Design Science the focus is on developing and creating interesting artefacts that address a distinct problem (Hevner et al., 2004; Reinecke & Bernstein, 2013). This approach has its roots in engineering (Simon, 1997) and without a problem to solve Design Science would not be applicable (Kuechler & Vaishnavi, 2008). In fact, it is through designing artefacts to solve problems that Design Science contributes to the creation of new knowledge (Owen, 1998).

In **Error! Reference source not found.** (Owen, 1998) describes the two types of research activities (Behavioural and Design Science) as operating in two realms. The Behavioural Science activity operates in the realm of theory and the Design Science activity operates in the realm of practice. In **Error! Reference source not found.** it is clear how Behavioural Science and Design Science both build upon the same knowledge base and how this links the two activities. Mark and Storey even go as far as to say that these two (knowledge accumulated through Behavioural Science and knowledge accumulated through Design Science) are two sides to the same coin and that the one cannot exist without the other (Hevner et al., 2004).

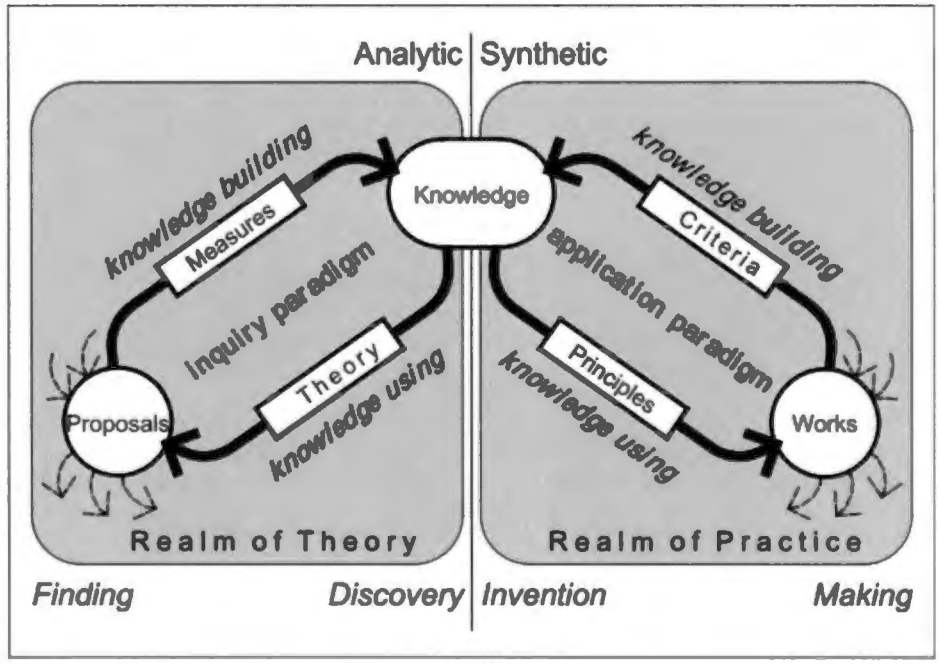

Figure 7: Behavioural and Design Science Knowledge accumulating (Owen, 1998)

Focusing more on the right part of **Error! Reference source not found.** it can be seen that works (artefact(s) designed to solve a specific problem) are built based upon some sort of knowledge base where it takes that knowledge and uses it in the design of the artefact(s). It can also be seen that the design and implementation of the artefact(s) in turn builds more knowledge.

It is this (building artefacts based upon prior knowledge and building knowledge through designing and implementing artefacts) that justifies research through design (Hevner, 2007). If the design and implementation of the artefact(s) does not add additional knowledge to the knowledge base, no research has taken place (Hevner et al., 2004).

How do building new and innovative artefacts add more knowledge to the knowledge base? March & Storey states that:

"As field studies enable Behavioural Science researchers to understand organisational phenomena in context, the process of constructing and exercising innovative IT artefacts enables Design Science researchers to understand the problem addressed by the artefact and the feasibility of their approach to its solution."

This can further be explained by the following example. People were able to build working aircrafts long before they fully understood why said aircrafts were able to fly. In fact, it was

the designing, building and studying of the working aircrafts as they flew that enabled theories of why they are able to fly to be properly formulated (Vaishnavi & Kuechler, 2004).

This is further emphasised by (March & Smith, 1995) when they state that "an instantiation sometimes precedes a complete articulation of the conceptual vocabulary and the models (or theories) that it embodies". Again it should be emphasised that if new knowledge (usually in the form of new models and/or theories) is not gained by designing and exercising the artefact, no research has taken place and the whole process was simply a routine design exercise.

In (Hevner, 2007) Hevner states that Design Science is performed by iterating through three cycles. These cycles should be clearly definable throughout the Design Science Research Project:

- Relevance Cycle
- Rigor Cycle
- Design Cycle

In the Relevance Cycle, the requirements for the design of the artefact are determined and the artefact is tested in the environment for which it was built. This will not only determine whether or not the artefact adheres to the requirements or not but it will also determine whether or not the requirements that were set up is sufficient for the problem the artefact was meant to solve (such as in the case where the artefact satisfies the requirements but only partially addresses the problem it was supposed to solve) (Hevner, 2007).

It is in the Rigor Cycle that the Design Science Research project engages with the scientific and academic body of knowledge. Here the Design Science draws on the previous knowledge as well as area expertise and practises in the application domain of the Design Science Project. The Rigor Cycle is also responsible for investing back into the knowledge base by adding new insights, theories, methods and knowledge gained by performing the Design Science Research (Hevner, 2007).

The Design Cycle is the essence of any Design Science Research project. As stated above it is reliant on the Relevance and Rigor Cycles but once they have played their role there is a form of independence in the Design Cycle. It is here where the knowledge gained through the Relevance and Rigor Cycles is used to construct the artefact.

These three cycles are illustrated in **Error! Reference source not found.**.

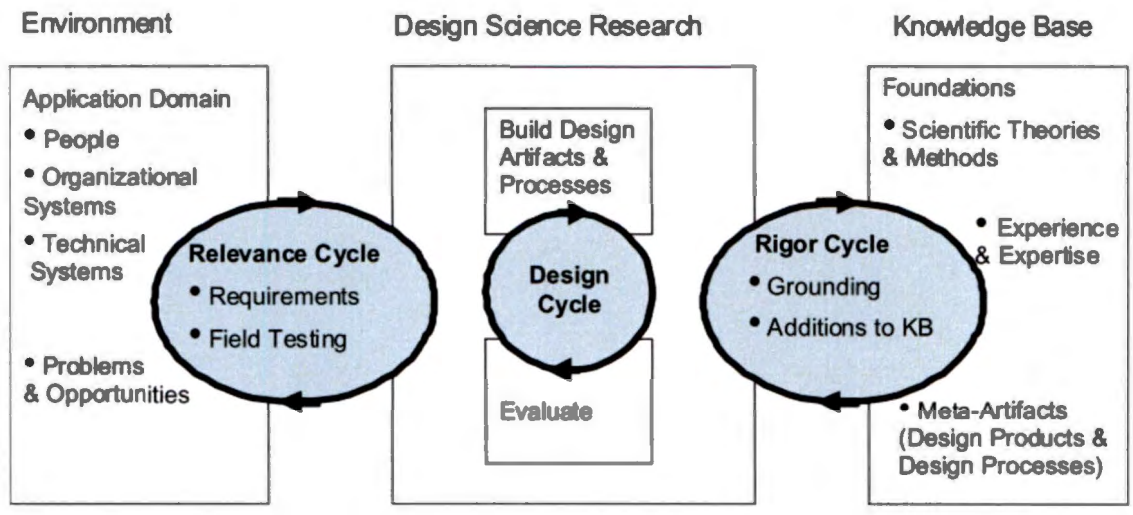

Figure 8: Design Science Research Cycles (Hevner, 2007)

This was taken further in (Peffers, Tuunanen, Rothenberger, & Chatterjee, 2007) where they propose a methodology for performing Design Science Research. According to them, there are six activities that can be performed to make up Design Science Research. These six activities are discussed below:

Activity 1: Problem identification and motivation – Here the research problem is defined and reasoning for the resolution of the problem is motivated. It is important to define every aspect of the problem since the results of this activity will be used to develop objectives and specifications for the artefact (Walls, Widmeyer, El Sawy, & Sawy, 1992). It is also worth noting that the problem identified here is not yet finalised. It might very well be that the result from evaluating the design cycles helps to better define the problem being addressed. This is especially true for cases where the problem that needs to solve is vague and hard to define (Peffers et al., 2007).

Activity 2: Define the objectives for a solution – Here the objectives and specifications for the solution is derived using the problem identification and prior knowledge about what is possible and what has been achieved for similar problems (Hevner et al., 2004; Peffers et al., 2007).

Activity 3: Design and Development – This is the heart of any Design Science Research Project (Hevner, 2007). In this activity an artefact to solve part or the entire problem is designed and developed. Prior knowledge gained through the Rigor cycles usually plays a big role in this activity (Hevner et al., 2004). It is how the Design Science Research Project draws from prior knowledge, area expertise and common practices as a starting point for coming up with a solution for the problem (Peffers et al., 2007).

Activity 4: Demonstration – After the artefact is designed and developed the operation of the artefact should be demonstrated. This could be through simulation, experimentation or any activity showing the working of the activity (Peffers et al., 2007).

Activity 5: Evaluation – Here the feasibility of the artefact to solve the problem is determined by rigorously and thoroughly evaluating the demonstration of the workings of the artefact. It is with this activity that the researcher must decide if more iterations are required. Analysis of why the artefact is or is not a feasible solution to the problem will be performed at this point (Nunamaker, Chen, & Purdin, 1991). The knowledge gained through that analysis will then either be taken back to one of the previous activities for another iteration or be used going forward in the communication activity (Peffers et al., 2007). Both validation and verification are performed with this activity.

Activity 6: Communication – All aspects should be communicated; everything, from the problem specification to the designed artefact, why and how this artefact solved the problem, knowledge and rigor gained and the effectiveness of the artefact, Communication should be directed to other researchers and relevant audiences like general practitioners and professionals that are in the same field (Peffers et al., 2007).

These activities and the interaction between them are demonstrated in **Error! Reference source not found.**.

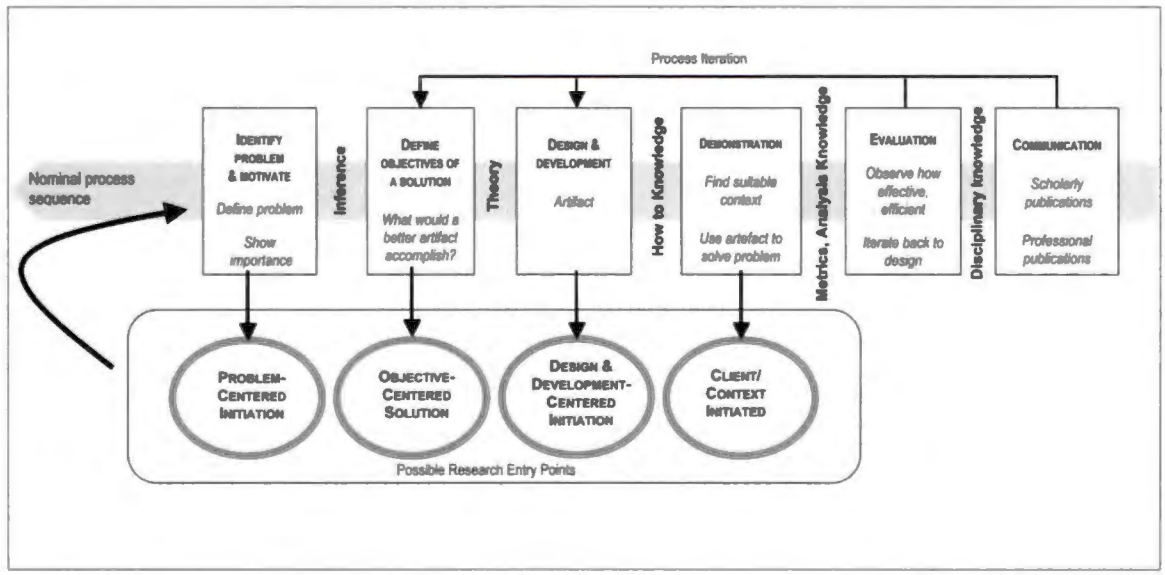

Figure 9: A proposal to a Design Science Research Methodology in Information Systems (Peffers et al., 2007)

Lastly, the work of Myers and Venable where they call for conversation around a set of ethical principles for Design Science Research in information systems (Myers & Venable, 2014) is

referenced. It was known from the start of the information age that information technology has the potential to improve the lives of the human race to a large extent. However, parallel to that, the potential of information technology to do significant damage to the very human lives it was meant to improve has also been known for just as long. This is clear when literature as far back as that of Mason is referenced (Mason, 1986).

Using Design Science Research to do research in information science is growing in popularity but it is the view of Myers and Venable that the guidelines found in literature on how to conduct this research lacks any ethical guidelines. This view is shared by the author of this dissertation and seeing that Design Science was extensively used throughout this study it was thought of to be irresponsible to not at least mention the work of Myers and Venable thereby increasing the call for further discussion around this subject.

2.4.3. Approach

As stated earlier, Design Science was used to study the interaction between the platform (the artefact) and the rest of the SoS in order to determine what is needed from a platform like this in the environment that it will be used in. This environment is described in Chapter **Error! Reference source not found.**. As part of the problem identification, the whole environment was modelled as a SoS with the platform being an independent system within this larger SoS. This helps with defining where the platform fits in with the larger SoS and how it interacts with the rest of its environment. This model along with the literature found in Chapter **Error! Reference source not found.** serves as the inputs for the initial design of the first design iteration.

It is important to remember that the artefact (the actual platform) should allow for the design and implementation of different translators. This means that the action of designing different translators is not part of the design of the artefact but that by using the platform to design different translators, the workings of the artefact is demonstrated. It is thus logical to test how well the artefact allows for the development of different translators by designing various translators on the platform. This aspect is vital to the understanding of how Design Science is used to conduct this study.

For each design iteration, a design (or alteration to the previous design) of the platform is performed, which is then implemented. To demonstrate the working of the platform a translator is designed and implemented on the platform. What is meant by the implementation of the translator is that the newly created translator is used to translate files. Along with testing how

well the platform works it is also important to test how well the translators that are created on the platform works.

Having translators implemented on the platform that do not function correctly does not necessarily mean that the platform does not fulfil its purpose. It might only mean that the translator that was designed using the platform may have been poorly designed. However to prove that the platform can be used to design and implement well-functioning translators it is of importance that fully-functioning translators are implemented on top of the platform.

Due to the artefact being the platform and not the translator itself the design iterations will be centred on improving the design of the platform and not necessarily the design and implementation of the various translators built on top of the platform. It is important to demonstrate that the platform has the ability to implement well-functioning translators; therefore a new design iteration might be undertaken with the goal of improving and redesigning an implementation of a specific translator on the platform.

In this study, there will be main design cycles and sub design cycles. In each main design cycle a new translator will be introduced. Each sub-design cycle will focus on fine tuning the current implementation of the platform or improving the design or implementation of the translator introduced in that design cycle (if necessary). The design cycles will be conducted using the activities set out by Peffers et al. as guide (Peffers et al., 2007). Not all the design cycles will consist of all the activities (especially relating to the sub design cycles) however all of the cycles will have at least involve both the demonstration and evaluation activity.

The relevance cycle Hevner spoke about (see Section **Error! Reference source not found.**) can be seen in how each of the evaluation activities (see each of the evaluation sections in Chapter **Error! Reference source not found.**) ensures relevance by steering the design of the platform in a direction that will make the platform more useful in the environment it is used in. The design cycles Hevner spoke about are given in Chapter **Error! Reference source not found.** and shows how the Design Science process was followed to conduct research into solving the problem identified in Section **Error! Reference source not found.**. The output of this process is a platform that solves this problem.

The Rigor cycles that Hevner spoke about can be seen in how the knowledge gained through studying the literature (Chapter **Error! Reference source not found.**) is used in the design cycles and how the knowledge gained by conducting the Design Science Research is communicated back to the scientific community (using this document).

Each design cycle will be evaluated on the basis of whether or not the required data was translated and stored correctly. If this is true it means that the platform not only allowed for the creation of the translators needed to translate the data but also allowed for these translators to be used to translate and save data.

Seeing as this problem is ultimately an automation problem the platform will further be evaluated on the number of man-hours required to operate the platform. This will include aspects such as how much time is needed to create the different translators and how much time is needed each day to manage and maintain the platform.

2.4.4. The environment of the study

The study will be conducted at a company that has clients from five industrial groups which translates to 82 subsystems. As part of the services that the company offers, they need to perform analysis on the data from a wide range of systems from within these industrial groups. This places the company in a position where they inherit the environment described in Section **Error! Reference source not found.**.

The analysis that needs to be performed on this data is usually system-specific, this means that the calculations required for the different systems (as they are defined within the company) has to be custom designed for each one of the 82 subsystems within the company. To handle this, the company has a dedicated team of personnel that design, implement and maintain the different analysis for each of the different subsystems. From here on this personnel will be referred to as Non-Technical Human Resources (NTHR). The non-technical part referring to the fact that they do not necessarily have (nor do they need) programming skills.

There exists a system (from here on referred to as the reporting system) put in place to help the NTHR to perform and automate their system specific calculations required for analysis. The reporting system will receive its data from the data accumulators and then perform its analysis that has been designed and implemented by the NTHR. It will need to do calculations on data from various parts of the mine and as such will have to "understand" the data from numerous data file types.

2.4.5. Summation of Design Iterations

Given below is a short summary of the main Design Iterations and what to expect in these iterations.

Iteration	Translator that is introduced	Description of what to expect
Error! Reference source not found.	Export Translator	In this iteration, the first design of the platform is introduced. This design is then tested and evaluated by developing the Export translator and using that translator to translate Export files.
Error! Reference source not found.	Energy Management (EM) Translator	In this iteration, the EM translator was introduced. When the design of the EM translator was compared to the design of the Export translator it was found that there were a lot of similarities between the two designs.
Error! Reference source not found.	Standard Translator	Here the standard translator was introduced. For the design of this iteration, a solution was designed that allowed the similarities within the translators to be migrated to the platform.
Error! Reference source not found.	PDI Translator	In this iteration the Pentaho Data Integration (PDI) translator was introduced. This iteration was used to validate the solution.

2.5. Conclusion

A literature review into other translation platforms has been conducted and these platforms have been evaluated according to the requirements of this study. The literature was also studied to determine how a translation platform such as the one required should be designed. Design Science was implemented as the chosen methodology for this study. The literature was consulted into what Design Science is and how it worked and this was then used as the base with which the rest of the research was conducted.

3. Design Cycles

3.1. Introduction

In the section that follows (Chapter **Error! Reference source not found.**) the Design Science Cycles are given. This section has been sub-divided where each sub-section (**Error! Reference source not found., Error! Reference source not found., Error! Reference source not found.**) represents a new main design iteration. Each one of the main design iterations introduces the design and implementation of a new translator. Each main design iteration is also sub-divided into the design activities set out by Peffers et al.

Each one of the main design iterations is verified using two test projects referred to as Project A and B. At the end of the last iteration the artefact is evaluated to be satisfactory using Project A and B. A final iteration is then given in Section **Error! Reference source not found.** where the artefact is not evaluated using Project A and B but is evaluated within the full environment the artefact is intended to function in. This will validate the solution reached at the end of the design iterations.

3.2. First iteration

In the first design iteration, the initial design of the platform was conducted. The problem identification (Section **Error! Reference source not found.**) and knowledge gained through the literature (Chapter **Error! Reference source not found.**) were used as inputs to the design. To demonstrate the operation of the platform the design and implementation of the export translator were used.

3.2.1. Main Design Cycle

Initial Problem Identification

As stated earlier, the need exists for a data translation platform. These translators should translate files from different file types into a predefined standard format. An example of this standard file format is given in **Error! Reference source not found.**. The file types that need to be translated to this standard format differ to a great extent from file type to file type. This makes the files that must be translated diverse in nature.

The standard format is defined in terms of time-based readings of specific tags. Each tag must have a Tag Name, Tag ID and Tag Unit. The Standard format, then further, consists of values linked to specific dates (aggregating horizontally) and time intervals (aggregating vertically).

These time intervals can be half-hourly, hourly, daily totals, daily average, monthly total or monthly average values.

Tag Name:	Tag A			
Tag ID:	0001			
Tag Unit:	A Unit			
Date	Date 1	Date 2	...	Date N
00_30	Value	Value	...	Value
01_00	Value	Value	...	Value
01_30	Value	Value	...	Value
:	:	:		:
24_00	Value	Value	...	Value

Table 2: Example of the predefined standard format.

It is also important to consider the dynamic nature of the internal structure of the files that must be translated. The way the files are created and defined is not always final and as the requirements for the specific files may change the internal structure of these files may be adapted as well. This is particularly important when consideration needs to be made for the limited amount of human resources that are available to design, modify and implement the different translators required.

Vast amounts of data from different sub-systems will be translated on a daily basis. This process must be managed and maintained for each sub-system and will differ from sub-system to sub-system. The translated data will repeatedly be used by many different calculations during the analysis performed in the reporting system.

The platform also needs to function as a subsystem within a larger SoS. To help conceptualise where the platform fits in and what its interaction with the rest of the SoS is, the SoS breakdown given in **Error! Reference source not found.** is also used as input. For convenience **Error! Reference source not found.** is repeated here as **Error! Reference source not found.** as well.

Initial Objectives

The initial objectives of this study were to develop a translation platform that will allow for the creation of translators for the purpose of translating files into the predefined standard format given in **Error! Reference source not found.**. The translators that are developed for this platform should be able to handle the diverse nature of the files that must be translated.

It needs to be possible to design, modify and implement translators on this platform to allow for the dynamic and versatile nature of the internal structures of the files. The platform should allow for an already created translator to be adapted should the internal structure of a file type it is translating change.

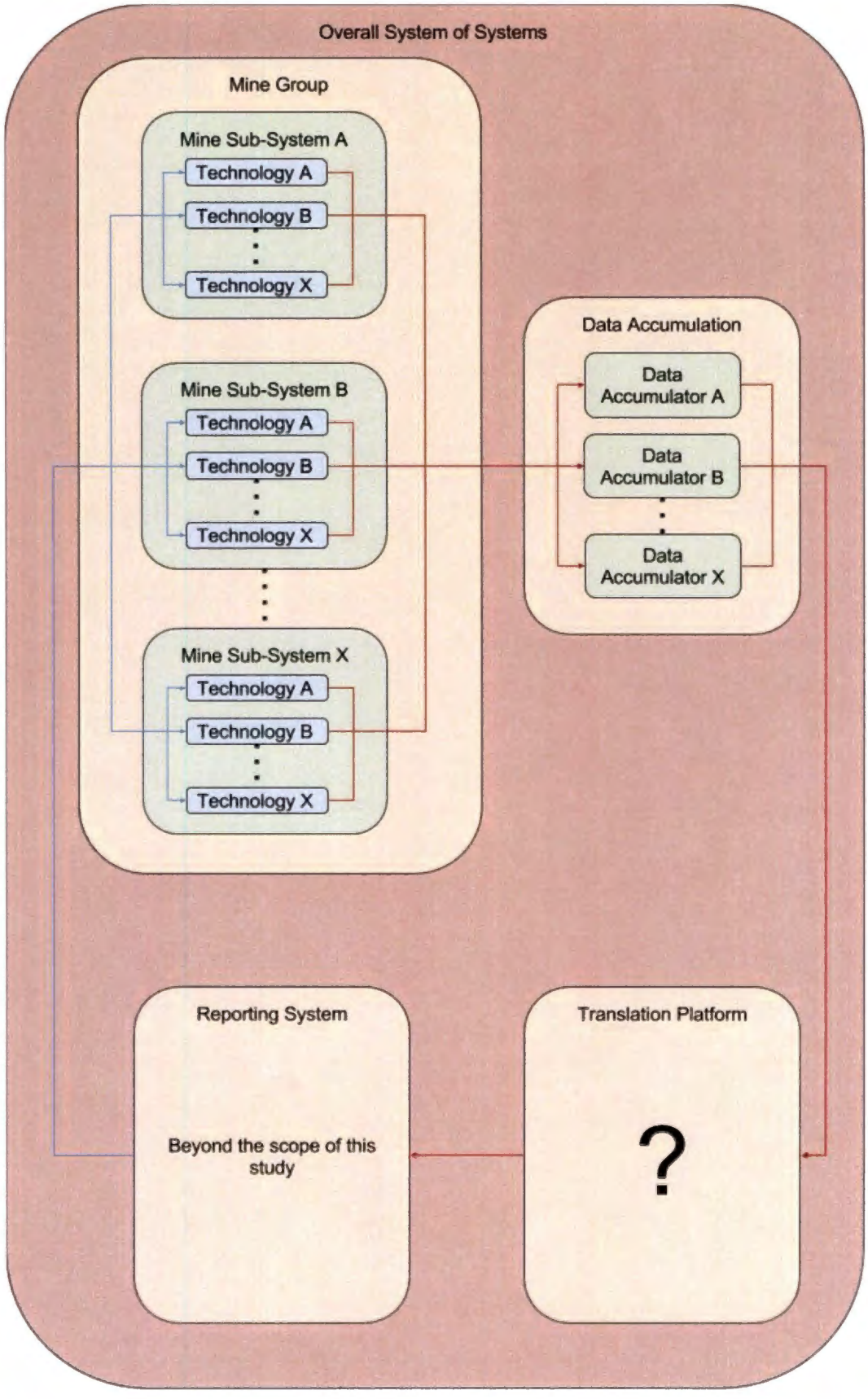

Figure 10: Repeat of **Error! Reference source not found.**

Artificial intelligence has come a long way in the past couple of years but we are not yet there where a computer can perceive enough from a simple file it has never seen before to know how to translate that file into a predefined standard format. Therefore this responsibility still remains with the human resources element within a company. Due to the diverse nature of the files that must be translated it has been decided to let the different translators be designed and created using a programming language instead of the more simplified graphical but also limited manner in which some of the other platforms (such as the platforms listed in Section **Error! Reference source not found.**) allows the creation of translators.

This, unfortunately, has the consequence that the human resources doing the actual design and creation of the translators are required to have some programming knowledge. In the company, the number of staff members that meet this requirement (in this document they are referred to as THR) is limited. Thus the creation and modification of translators needs to be efficient enough to ensure that the time that the THR have available to create and modify translators, matches the demand for the creation and modification of translators.

It has been decided to use C3 as the default programing language in which the translators must be developed. This has the added benefit of allowing the use of some of the libraries that have already been developed for the company, in the development of the platform and the translators.

The translated data must be successfully stored in order for the data to be retrieved at a later stage. The process of managing and maintaining the translated data must be efficient enough in order to ensure that the data is ready retrieval. For this, it has been decided to use a blended solution between human resources and systems built into the platform.

The best human resources to use for this will be the human resources that are responsible for creating the custom calculations in the reporting system (in this document referred to as NTHR). This staff will be the best at determining when data needs to be received and where in the larger SoS to investigate for any problem when data stops being received by the translation platform. The NTHR do not, however, possess any programming skills and as such coding should not be needed in managing and maintaining the translated data.

Initial Design

From the literature it has been decided to design the platform in such a way that ontology-based translations are used for the design of the translators. The way this works is that an ontology is designed for the standard format and for the format of the file that must be translated.

Afterwards, an ontology translation or ontology map is designed to translate the new file ontology into the standard ontology. Data gets pulled from the file that must be translated, inserted into the newly developed ontology, translated into the standard ontology after which it then gets saved into a standard file format. This is shown in **Error! Reference source not found.**.

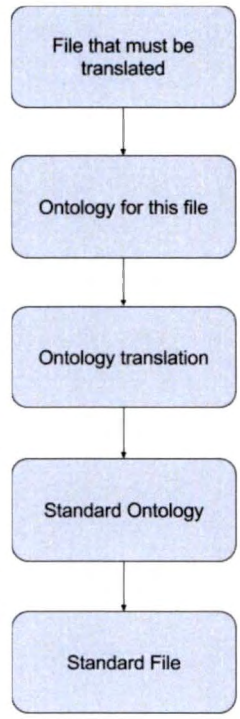

Figure 11: Data Flow for the translation process

Next, the standard ontology will be discussed. In the design of the standard ontology, an effort was made to try and capture the essence of the standard file. The function of any file format is to capture and relay information to the reader (human or computer) of the file. For the standard file format, this information is time-based values for different tags (in this case tags define an entity of information).

The time intervals for these values can be any one of the following: half-hourly values, hourly values, day total values, day average values, month total values or month average values. Along with the time-based values and the corresponding times for which these values are valid, the standard format also relays the tag name, tag ID and the tag unit in which those values are captured.

The ontology that was designed for the standard format is given in **Error! Reference source not found.**. For each tag in the standard format this ontology captures the Tag Name and the

Tag Unit. It then also captures, for each tag, a date array with individual date elements for each date contained in that particular tag.

Each date element then contains a value array where each element in the value array corresponds to one of the time intervals mentioned above. For the day total, day average, month total and month average time intervals the value array will only contain one element. For the hourly time intervals the value array will contain 24 elements and for the half-hourly time intervals the value array will contain 48 elements. Each time interval element also contains the corresponding value for that time interval.

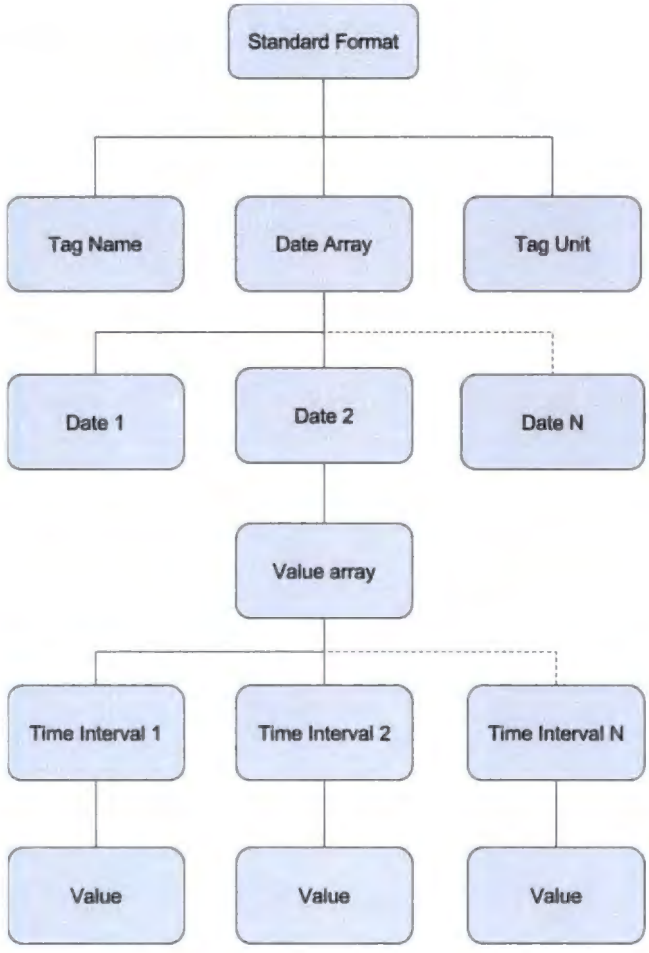

Figure 12: Standard Format Ontology

Using ontologies in this manner for the translation of files helps with the rapid nature in which the different translators need to be developed. Having the ontology already in place when the structure of an input file is changed means that only that specific change needs to be applied to the ontology and ontology translation for the system as a whole to accept the new changes.

To help lessen the workload for the NTHR that are allocated to manage the flow of data through the system (ensuring that the data comes in, monitoring the translation process, etc.), an automated process has been set in place. This automated process relies on an email system for acquiring the data from the data accumulators.

It was decided to make use of emails for acquiring data due to the fact that the NTHR will not necessarily have a very advanced level of technical skills when it comes to the field of computer science, but will know how to use and send emails. In this way, they can use an already familiar technology from the start to initiate the translation process on the data that they need translated. Automatic email forwarding can then also be used to further automate the translation process.

A simple folder structure for the flow of data has been decided on to allow for easy monitoring by the NTHR. When files are emailed they are extracted from the email and placed in the inbox folder. From there the data in the files undergoes the translation process. After the translation, the translated data gets tested for any merge conflicts with the data already in the data folder.

If any such merge conflicts occur, the original files get moved to the unprocessed folder and the translated data gets deleted. It is then the responsibility of the NTHR to handle the merge conflicts. The NTHR will do this by either deleting the part of data in the data folder that experiences the merge conflict or by removing it from the original file now residing in the unprocessed folder. Once the merge conflicts have been resolved the original file (the one that is now residing in the unprocessed folder) gets reinserted into the inbox folder by the NTHR to get translated.

In the case where no merge conflicts occurred, the translated data gets stored in the data folder. Here the NTHR have also been considered when the decision was made to store the data in the data folder as normal Comma Separated Values (CSV) files. In this way they can easily open up the files to check for data. It has also been decided that the data in the data folder should be split up into monthly files (meaning that the data for each month gets stored in its own CSV). This will further help with the monitoring of data as only data from that month needs to be searched when looking for a specific piece of information. The flow of data described in this paragraph is summed up in **Error! Reference source not found.**.

When the platform was designed the speed at which new translators could be designed and the speed at which already designed translators could be altered was one of the main focus points of the design. With that in mind, it was decided to try and move as much of the processes required for the system out of the design of the actual translators and into the design of the

platform without moving processes unique to specific translators into the platform. Identifying these processes (processes that are not unique to specific translators and that can be moved to the platform) was harder than originally thought but has since become more apparent with the design of each Design Science iteration.

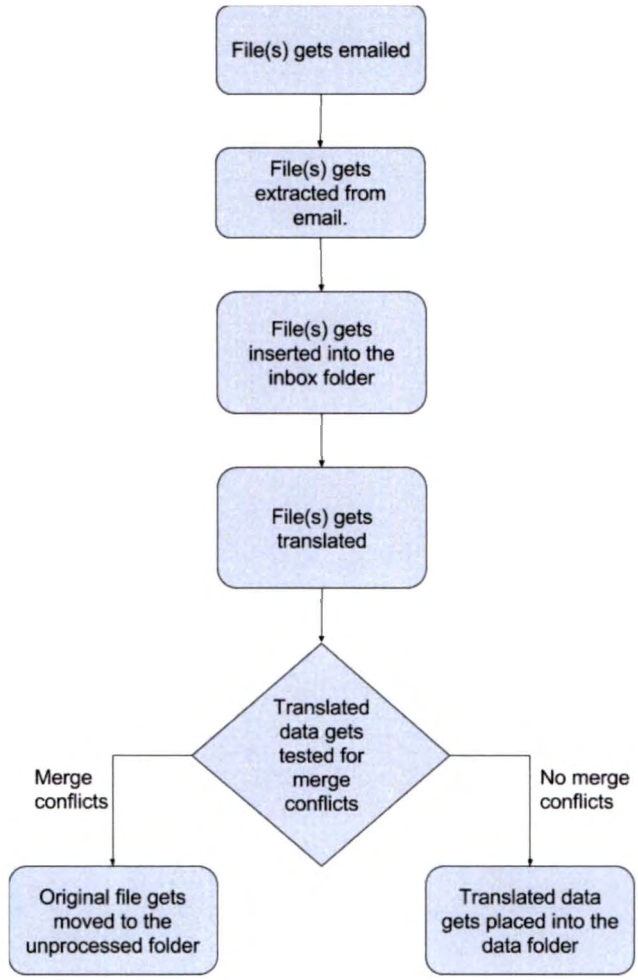

Figure 13: Flow of data through the designed system.

Given below in **Error! Reference source not found.** is the initial design for these processes and the entities that are responsible for executing them (platform or translator). The platform is responsible for monitoring the inbox folder for new files. As soon as new files are detected the platform accumulates a list of the new files and starts an instance of each translator created on the platform. The platform then gives the list of files to each translator.

Each translator then takes the list of files given to it by the platform and identifies which of the files are compatible with that specific translator; the translator then translates each of those

files. After the translation of a file, the translator then checks for merge conflicts between the newly translated data and the data already in the data folder. If merge conflicts are present, the original file is moved to the unprocessed folder and the original file is deleted.

If no merge conflicts occurred the newly translated data then gets added to the data already in the data folder and the original file is deleted.

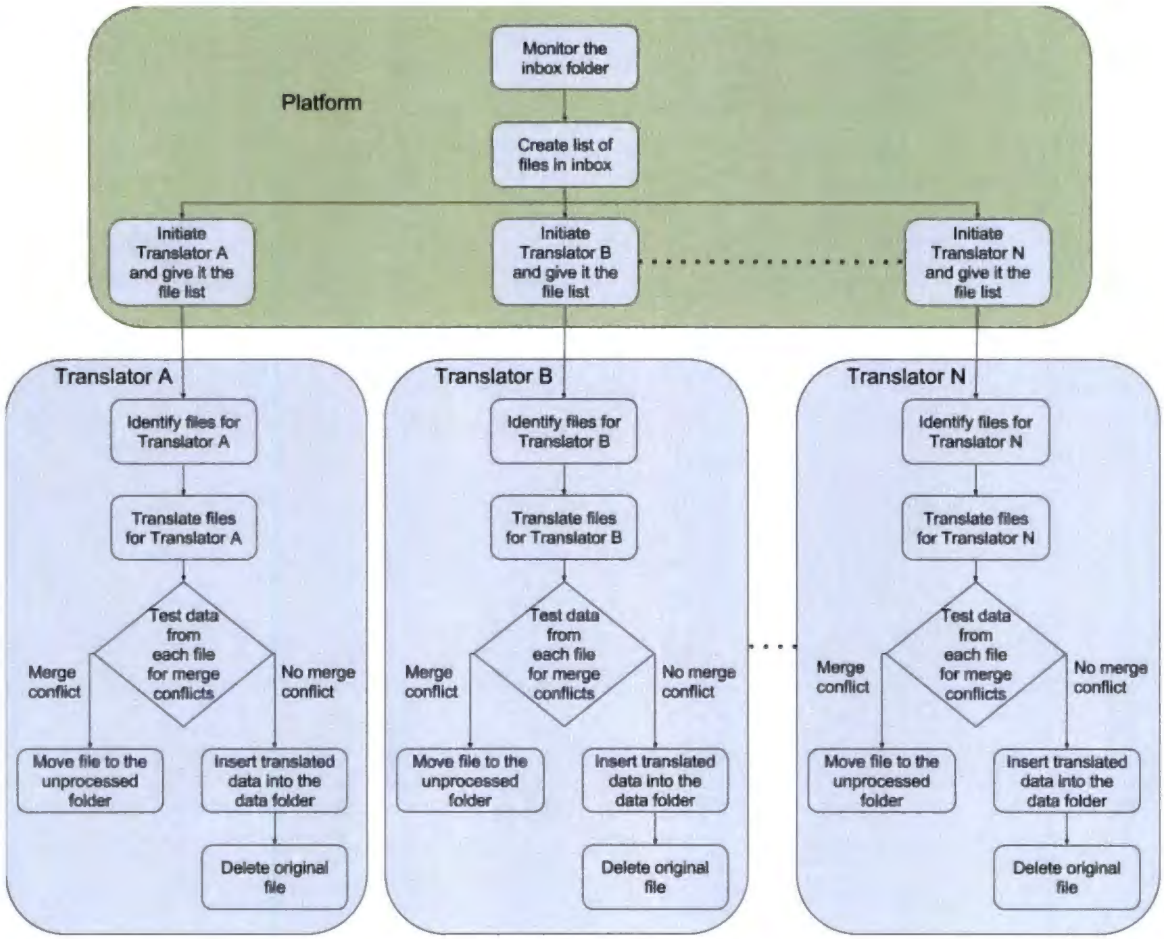

Figure 14: Initial platform and translator design

To help conceptualise how the THR and the NTHR act upon and influence the workings of the platform, a SoS modulation of the current design of the platform is given in **Error! Reference source not found.**. The red arrows in **Error! Reference source not found.** represent the flow of data between systems while the black arrows represent interactions of a particular system upon another. The file acquisition system represents the part of the platform that receives and extracts the different files from the emails received by the platform.

The translation system represents the part of the system that does the actual translation of the different files into the standard format. The different translators that are created on the platform

form part of this system. The data storage system in **Error! Reference source not found.** represents the part of the platform that stores the already translated data. Both the THR and the NTHR interact with the platform at different stages. The THR interact with the translation system in the sense that it creates and modifies the translators within the translation system that do the actual translation.

The NTHR interact with the file acquisition, the translation system and the data storage system. They act upon the file acquisition system by setting up and ensuring that the files that must be translated get sent to the platform. They decide what data it is that they want translated and then either emails the files to the platform themselves or sets up some sort of automated email forwarding process to automatically email the files to the platform.

The NTHR also interact with the translation system by monitoring the actual translation of the files and making sure that they get translated and that no merge conflicts occur with any of the files being translated. Lastly, the NTHR interact with the data storage system when it fixes any merge conflicts that might have occurred.

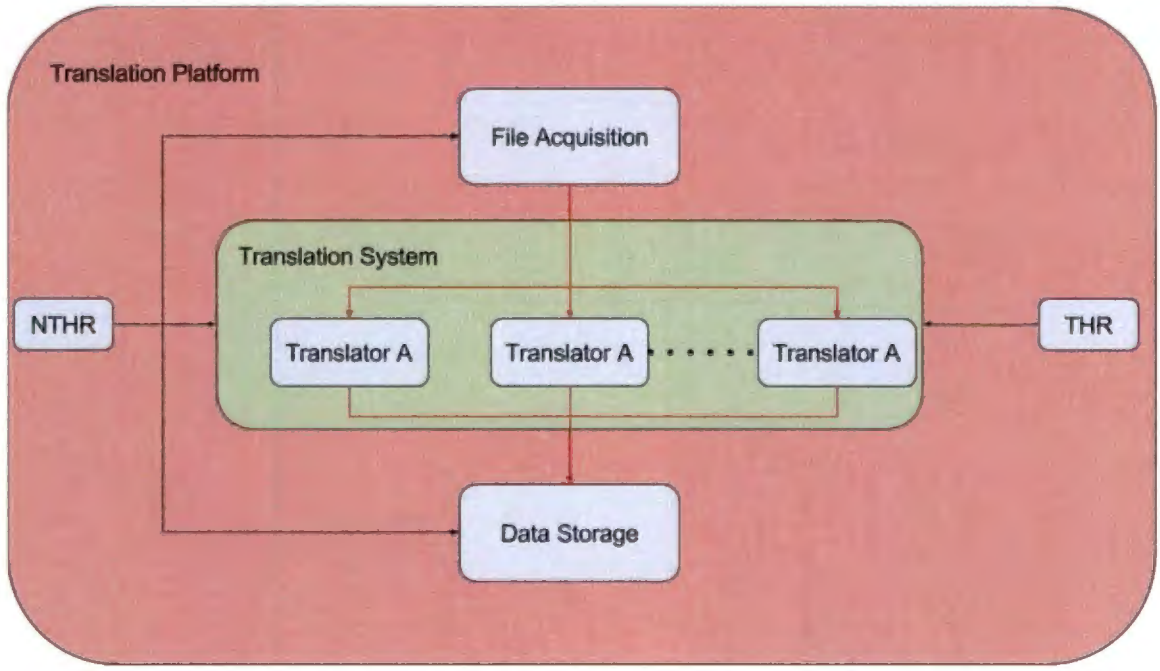

Figure 15: A System of Systems representation of the initial design.

Demonstration of initial design – The design of the export translator

For the demonstration of how the initial design of the translation platform works a translator was designed to translate data received in the Export format into the standard format. An

example of the Export format can be found in **Error! Reference source not found.**. The Export File is a normal CSV file consisting of different export tags. The tag names for these tags can be found in the first row of the file, from the second column onwards.

The first column is reserved for time information stating both the date and the half hour value for which the corresponding values given are valid. This format allows for a practically unlimited amount of tags and time intervals but the time intervals do not have to be in order and some of the time intervals can be left out. This will happen if none of the tags listed in the file has values for that particular time interval.

Date Time	Tag Name A	Tag Name B	…	Tag Name N
Date1 00:30	Value	Value	…	Value
Date1 01:00	Value	Value	…	Value
Date1 01:30	Value	Value	…	Value
:	:	:		:
Date1 24:00	Value	Value	…	Value
Date2 00:30	Value	Value	…	Value
:	:	:		:
DateN 24:00	Value	Value	…	Value

Table 3: Export file format.

The ontology designed for the Export format given in **Error! Reference source not found.** contains a tag array where each element in the tag array corresponds to a tag in the Export file. Due to the fact that the number of tags in the Export file format can be infinite, the number of elements in the tag array can go up to a very large number. Each tag element then contains a tag name and a value array where the elements in the value array correspond to the actual values in the Export file. Each value element then contains a time interval as well as the actual value.

For the ontology translation of the export format to the standard format, different areas from the export format ontology were mapped to different areas of the standard format. For each tag in the Tag array in the Export Format ontology, a new standard tag is created. The export tag name is directly mapped to the standard tag name. Then by looking at the dates of the time intervals of all of the values in the value array of each export tag in the export array it is determined how many standard tag dates should be created for each standard tag.

Seeing as the Export Format contains half-hourly values the values in the export file will be mapped to half-hourly values in the standard tag format. In the cases where the Export Format

file has missing half-hourly values, the standard tag counterparts will be filled with blank values (“”).

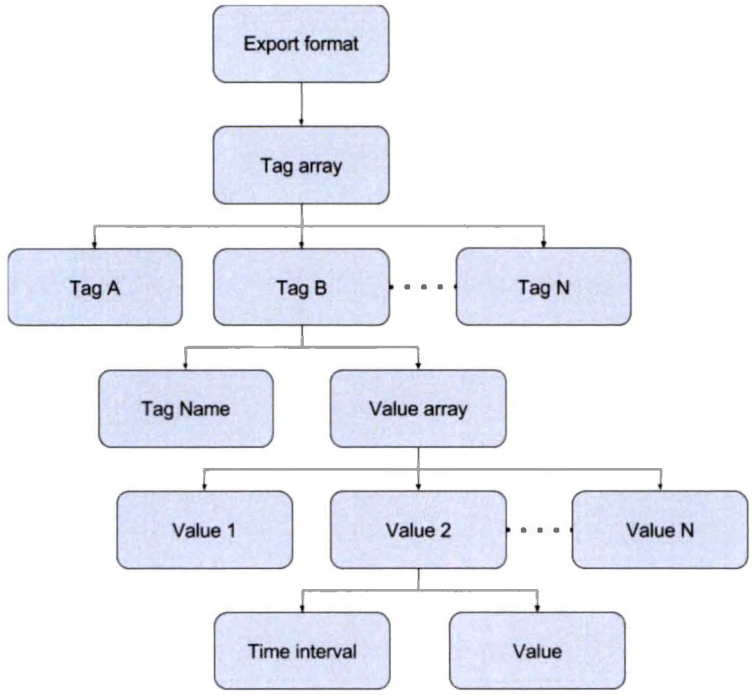

Figure 16: Export format ontology

Export file identification – a list of conditions has been drawn up for an Export file to adhere to in order for that file to be identified as an Export file:

- The file must be a CSV file with the file extension .csv
- The first row (excluding the first element in the first row) must contain no values
- The first column (excluding the first element in the first column) must contain both a date and a half-hourly time value

The export translator will take the file list passed to it by the platform and evaluate each file in the file list against the criteria listed above to identify all of the Export Format files. For each export file it will then extract the data in that file into the export ontology. Next, it will create a standard tag array where the number of elements in the standard tag array equal that of the number of export tags in the export tag file. It will then use the export tag ontology translation, designed above, to translate the export tags in the export ontology into the standard tags in the standard tag array.

Before merging the new standard data with the data already in the data folder, it will check for any merge conflicts between the two sets of data. This will be done by checking that no tag in

the standard tag array has a different value for the same day and time of said tag in the data in the data folder. If such a conflict exists, the original export file is moved to the unprocessed folder and the data in the standard tag array is discarded. If such a merge conflict did not occur the data in the standard tag array is appended to the data in the data folder and the original export file is deleted. After this is completed the translator will move on to the next export file. This whole process is illustrated in **Error! Reference source not found.**.

The demonstration of the initial design of the platform was shown by designing the export translator on top of the platform. The export translator was used to translate export data for two projects over the course of two weeks. The time spent on each project to manage and maintain the flow of data through the system can be seen in **Error! Reference source not found.**. Each project received eight export files to translate per day.

The human resources did not work over the weekends and thus the maintenance work on the projects for Saturday and Sunday was done on the following Monday. As such the time spent on Monday (by the NTHR) was divided by three to find the average time spent on the projects over the three days. For the first project, the time spent by the human resources used to manage and maintain the flow of data through the system was an average of five minutes and ten seconds per day while the average for the second project was ten minutes and three seconds.

The main reason for the big difference in the time spent between the two projects was due to the difference in the number of merge conflicts between the two projects. Where Project A only had merge conflicts to deal with on the Wednesday of the first week, Project B had merge conflicts to deal with on the Thursday, Friday and Sunday of the first week and on the Tuesday, Saturday and Sunday of the second week.

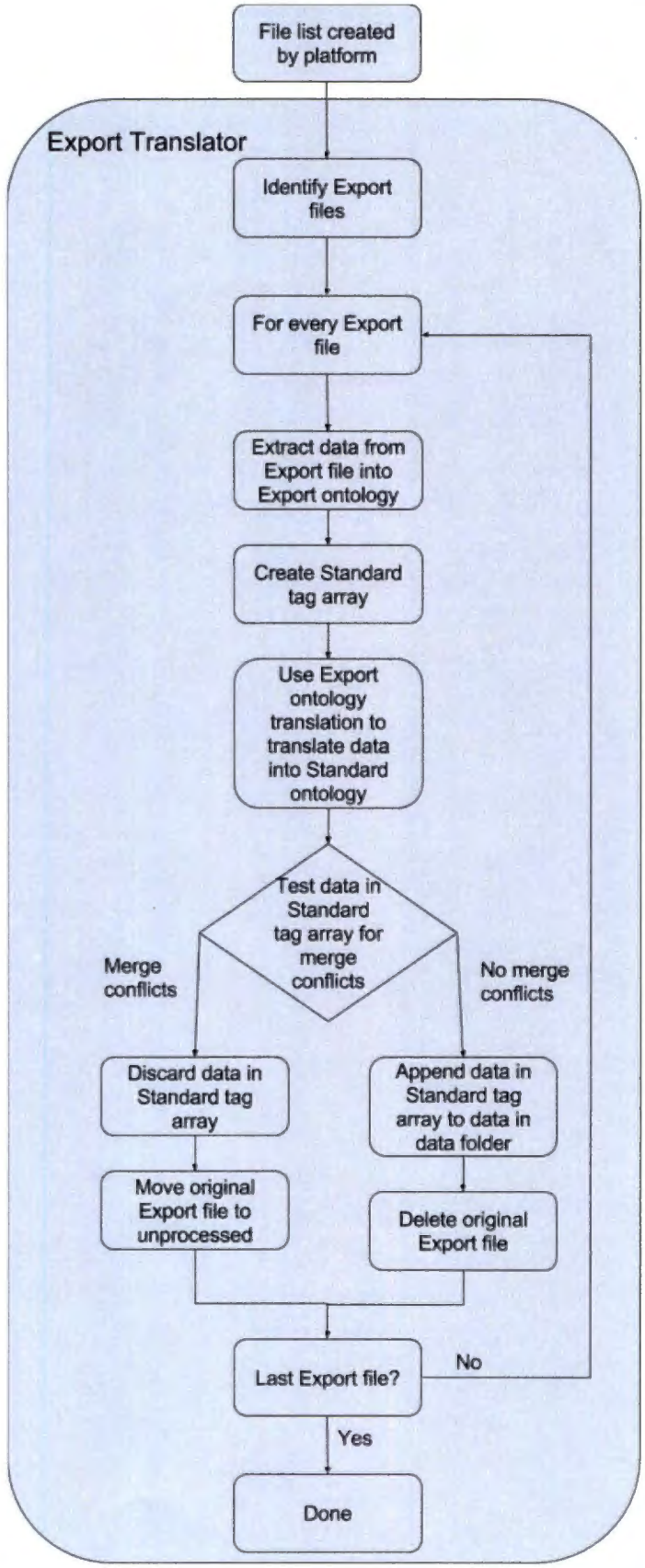

Figure 17: Export Translator

	Day	Time spent on managing and maintaining Project A	Time spent on managing and maintaining Project B
Week 1	Monday	4 Minutes	6 Minutes
	Tuesday	7 Minutes	4 Minutes
	Wednesday	9 Minutes	6 Minutes
	Thursday	5 Minutes	15 Minutes
	Friday	6 Minutes	26 Minutes
	Saturday	-	-
	Sunday	-	-
Week 2	Monday	13 Minutes	24 Minutes
	Tuesday	5 Minutes	14 Minutes
	Wednesday	3 Minutes	3 Minutes
	Thursday	3 Minutes	7 Minutes
	Friday	6 Minutes	3 Minutes
	Saturday	-	-
	Sunday	-	-
	Average	**5 Minutes 10 Seconds**	**10 Minutes 3 Seconds**

Table 4: Time spent on managing and maintaining two projects in the first design cycle.

Evaluation of the first Design Iteration

Evaluating the platform by looking at the data in the data folder after the two weeks it is clear that the data was successfully translated from the Export format into the Standard format and then successfully saved in the data folder. This means that the platform successfully allowed the creation of a translator for the translation of Export data into Standard data. The design of the Export translator took 18 man-hours.

Due to the internal structure of the company it has been decided the NTHR are the best capable of monitoring and maintaining the flow of data through the platform. This is because the NTHR are also responsible for setting up the calculations. The NTHR will best be capable of determining when what data should be translated and will be best capable at spotting errors in the data. These responsibilities include the handling of possible merge conflicts when it comes to storing the data in the data folder. To design the platform in such a way was therefore a good decision but in the current form of the platform the NTHR do this by hand and it turned out to be quite a tedious process.

This alongside the fact that the nature of the data being received is such that these conflicts occur more often than originally thought, requires some default software solution to be built into the platform. By studying these conflicts, it was found that the first data points most often need to be kept while ignoring the second data points.

The way in which the data accumulators interact with the platform is to email the files that must be translated to the platform; this also turned out to be a good design decision. Some of the data accumulators are protected by firewalls and having them email the data that must be translated is an easy way to get access to the data while still maintaining a large level of security on the data.

Sometimes files are emailed to the platform that either should not be translated or for which a translator has not yet been designed. These files are ignored by the export translator (or any other translator that might exist on the platform) and as such never gets deleted after the translation but only fills the inbox leading to the platform initiating another translation process. This is repeated until the NTHR manually deletes these files from the inbox.

Looking more broadly at the platform and evaluating its interaction with the rest of the systems in the overall SoS, the platform (as it is designed in this design iteration) is modelled with the rest of the systems in the overall SoS. **Error! Reference source not found.** shows the result of this.

In evaluating whether the creation of new translators will be efficient enough to keep up with the demand for new translators the source for the demand was evaluated more closely. The demand for new translators comes from the new formats the data comes in when it arrives at the translation system inside the translation platform. Tracing the path of the data back it is clear that the Data Accumulators and ultimately the different technologies play a big part in demand for new translators.

This is all to be expected, as stated before the need for the different translators is a direct result of the different technologies. It would therefore make sense that the different technologies play a big role in determining the demand for new translators. Only the data that needs to be analysed by the reporting system needs to be translated. There is no need to translate data that will not be used.

As a result, the demand for new translators is also influenced by the analysis needs of the reporting system. This, in turn, is influenced by the action that results from the conclusions of that analysis. As stated before there is no value in performing analysis on data if the conclusions of that analysis do not lead to some action. Even if the action is to carry on as before because the conclusion of the analysis was confirmation that nothing should be changed.

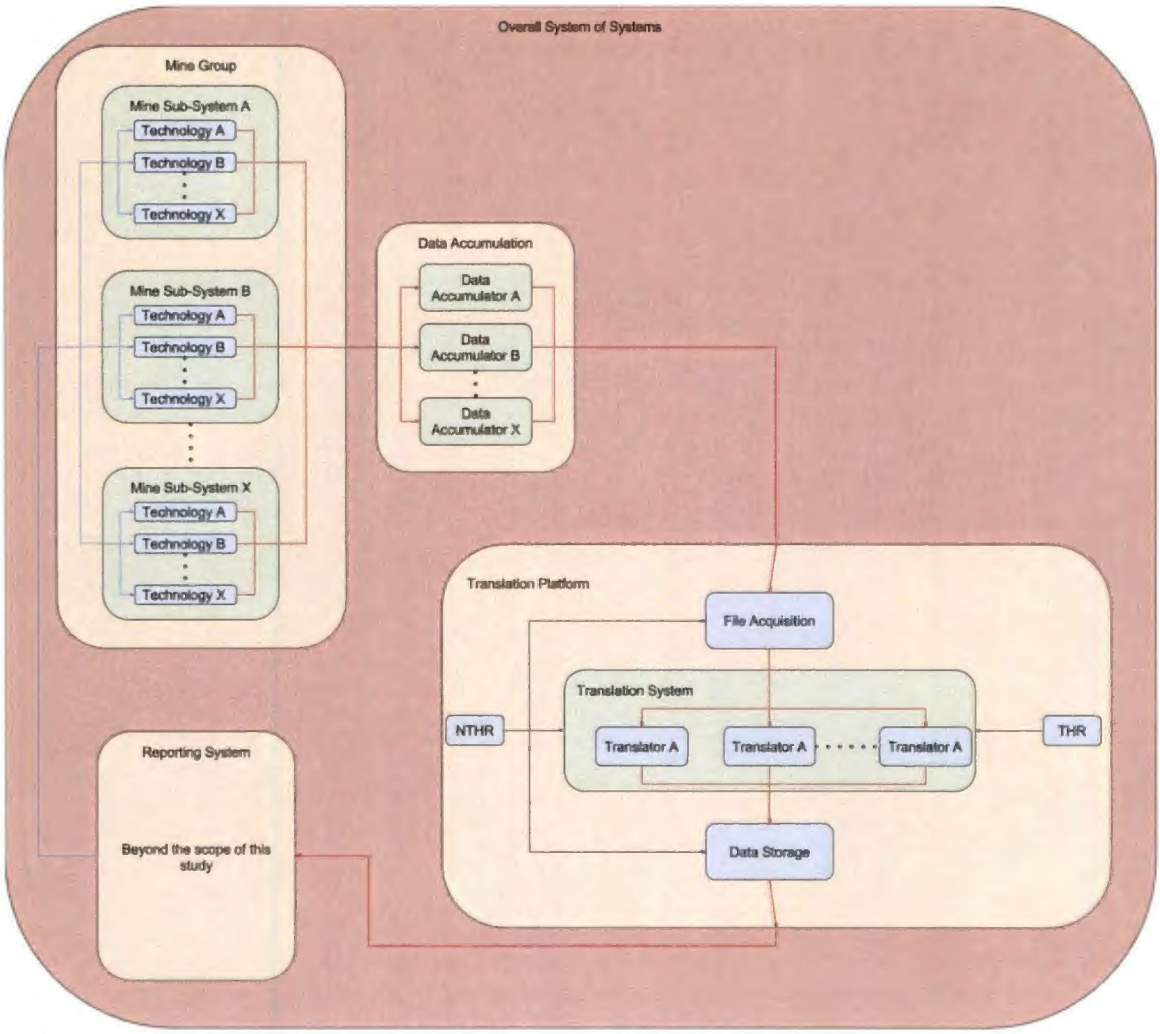

Figure 18: First iteration overall System of Systems Design

3.2.2. Sub Design Cycle

Design Objectives for the Sub Design Cycle

The objective of this iteration is to modify the solution in such a way that the work done by the NTHR is less human intensive. As such two key areas have been identified that require optimisation to accomplish this. The first is to handle merge conflicts in such a way that it requires less human resources. The second key area is to handle the files that should not be translated differently thereby ensuring that these files do not trigger another translation process.

Design for the Sub Design Cycle

To accomplish these objectives the design of the platform has been adapted as depicted in **Error! Reference source not found.**. The platform will still monitor the inbox folder for new files and once the presence of new files are detected in the inbox it will create an instance of each translator and pass the list of the new files to each translator instance. Each translator will

then take that list and identify each file that is compatible with that particular translator. The translator will then translate each of the identified files and determine if any merge conflicts exists between the data in the data folder and the new, freshly translated data.

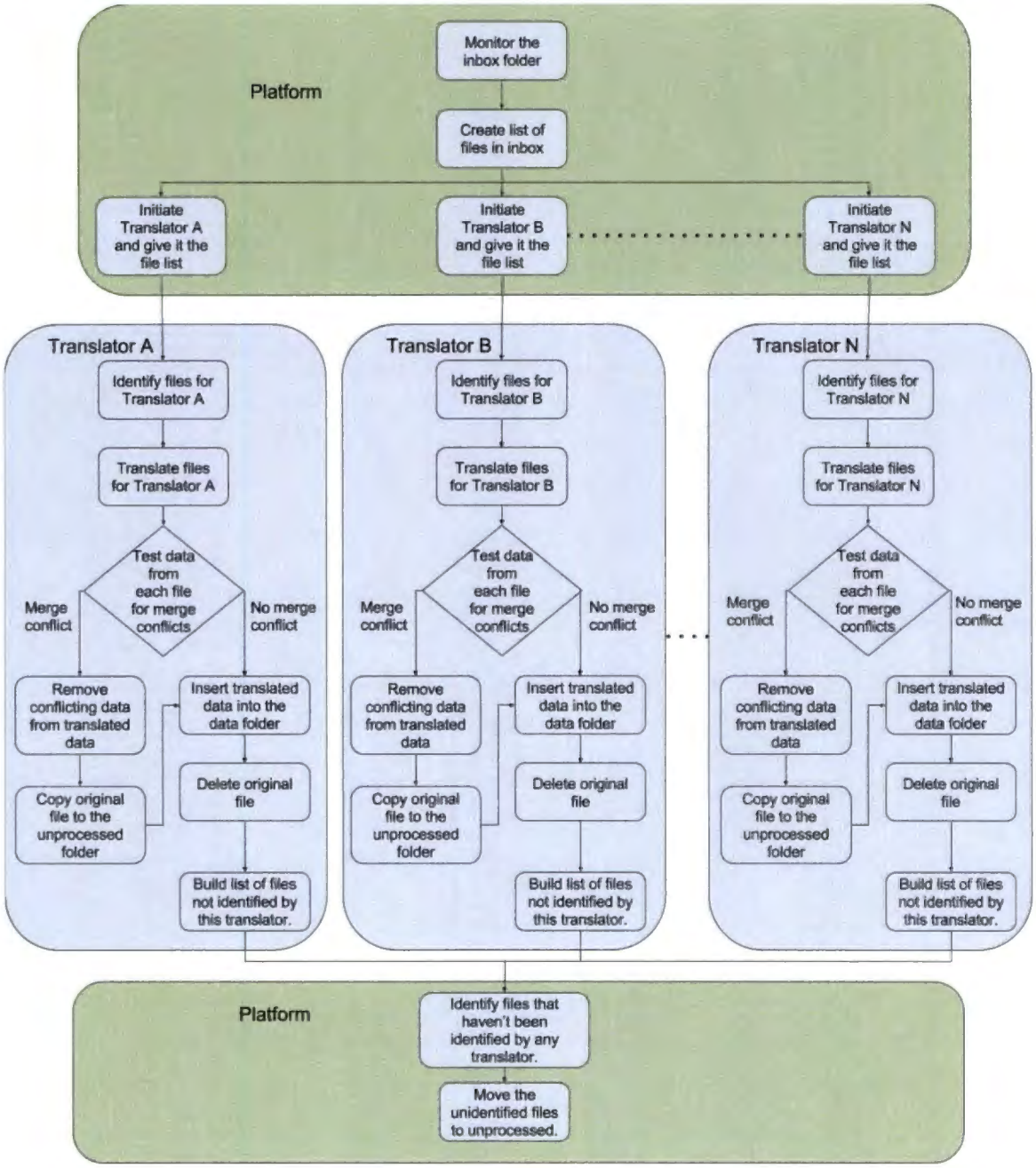

Figure 19: Second iteration Platform and translator design.

If any such merge conflicts occur, only the conflicting data points in the translated data will be removed so that the data that is left is only the translated data that does not contain merge conflicts. The original file (the file containing the conflicting data) will be copied to the unprocessed folder and the translated data now containing no conflicts will be appended to the

data in the data folder. The translator will then delete the original file. Lastly, the translator will build a list of the files it did not translate from the list of files handed to it by the platform and the translator will pass that list back to the platform.

In the case where no conflict occurred the translated data will be append to the data in the data folder and the original file will be deleted. Lastly, the translator will build a list of the files it did not translate from the list of files handed to it by the platform and the translator will pass that list back to the platform.

The platform will take the lists of files that were not translated by the specific translators and determine which files did not undergo translation by any translator and move these files from the inbox to the unprocessed folder.

Demonstration of the Sub Design Cycle

To demonstrate the design of the platform in the second design iteration the design of the Export translator has been adapted for this platform design. The design of the Export translator stayed exactly the same till just after it has been determined whether or not the file currently being translated contains merge conflicts. This can be seen in **Error! Reference source not found.**.

In the case that only the specific data points in the Standard tag array that do conflict with the data in the data folder are removed from the standard tag array. The original file (the original file containing the data that conflicts with the data in the data folder) is then copied to the unprocessed folder in case further analysis by the NTHR is needed.

The data in the Standard tag array (with the conflicting data removed if needed) is then appended to the data in the Data folder and the original file is deleted. Lastly, the file is removed from the list originally passed to the export translator by the platform. After all the export files have been translated, this list (now containing no more export files) will then be passed back to the platform.

For the demonstration of the second design iteration, the Export translator designed in Project A and B has again been used and again the time spent on managing and maintaining the flow of data through the system has been recorded. Just like the first design iteration the projects ran for a total of two weeks receiving eight files each to translate every day. The results can be seen in **Error! Reference source not found.**. Again the NTHR did not work over weekends so the time spent on the following Monday has been divided by three to get the average per

day for the previous Saturday, Sunday and that Monday. This average value has then been used for those days.

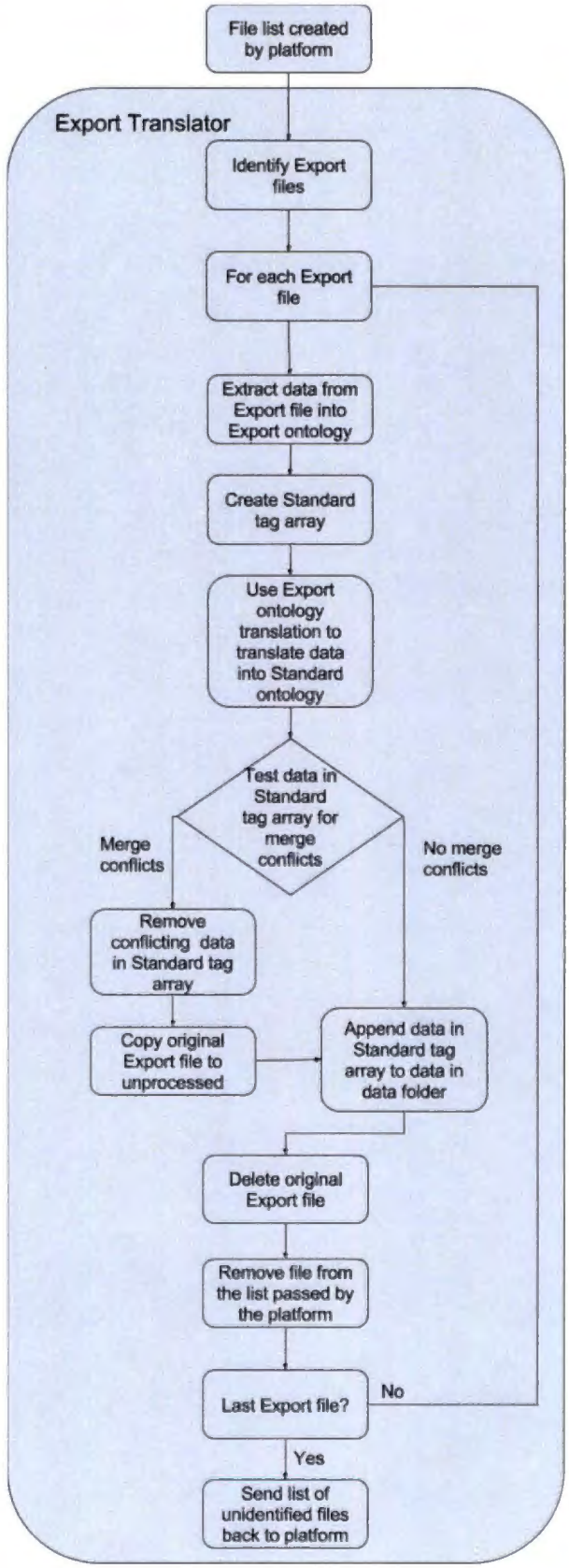

Figure 20: Second iteration Export Translator design

	Day	Time spent on managing and maintaining Project A	Time spent on managing and maintaining Project B
Week 1	Monday	4 Minutes	7 Minutes
	Tuesday	8 Minutes	8 Minutes
	Wednesday	5 Minutes	5 Minutes
	Thursday	3 Minutes	4 Minutes
	Friday	6 Minutes	5 Minutes
	Saturday	-	-
	Sunday	-	-
Week 2	Monday	17 Minutes	20 Minutes
	Tuesday	2 Minutes	6 Minutes
	Wednesday	8 Minutes	4 Minutes
	Thursday	4 Minutes	5 Minutes
	Friday	5 Minutes	8 Minutes
	Saturday	-	-
	Sunday	-	-
	Average	**5 Minutes 8 Seconds**	**6 Minutes 14 Seconds**

Table 5: Time spent on managing and maintaining two projects in the second design cycle.

Both projects spent less time managing and maintaining their respective projects with Project A spending an average of five minutes and ten seconds and Project B spending an average of six minutes and fourteen seconds on the management and maintenance of their respective Projects. Again Project B had more merge conflicts to deal with than Project A. Project A only had to deal with merge conflicts on the Tuesday of the first week and the Wednesday of the second week. Project B had merge conflicts on the Monday, Tuesday, Saturday and Sunday of the first week and the Friday, Saturday and Sunday of the second week.

Evaluation of the sub-Design Cycle

The data in the export files that were sent for translation by the platform for project A and B has been successfully translated and saved in the data folders of the respective Projects. This (the fact that the data was successfully translated and saved) is proof that the Export translator works, which in turn is proof that the platform successfully allows for the creation of a translator to translate export data into standard data.

Handling the large number of conflicts now happens mostly automatically and human intervention with regards to conflict handling is now only needed when data must be overwritten. In these cases, the NTHR must delete the specific data that must be overwritten from the files in the data folder and reinsert the original data files (the files containing the data that must be used to overwrite the old data) into the inbox folder for that Project. This has reduced the time NTHR spends on managing and maintaining the projects by a significant amount.

The evaluation of this design iteration found that using normal CSVs to store the data was a good idea. As stated before the best suitable human resources to handle the conflicts is the NTHR. Having the data stored in normal CSVs helps them to easily find, delete and modify any conflicts that exist in the data. Using normal CSVs to store the data in the data folder also helps with the monitoring and maintenance done on the systems by the NTHR by making it easy to look at and search for the data already saved in the data folder.

Files that must not be translated or files for which no translators were created yet now get redirected to the unprocessed folder and no longer cause new translation processes to be triggered when no new files were received.

3.3. Second Iteration

Having now designed a platform that handles both the files that must not be translated and the files that contain merge conflicts, it has been decided to further test and evaluate the platform by implementing a new translator. This is shown in this design cycle by implementing a translator for the EM file.

Demonstration of the Main Design Cycle – The design of the EM Translator

An example of the file structure of the EM Format can be seen in **Error! Reference source not found.**. The file name of the EM file format is as follows: "Date Time ComponentGroup.csv". The "Date" part of the EM file name is the date (in the format of yyyy-mm-dd) for which the values in the EM file is valid. A single EM file is not allowed to have values for more than one day. The "Time" part of the EM file name is the first time interval (in the format of hh-mm-ss) for which there is values in the EM file. Usually, this will be 00-00-00. The "ComponentGroup" part of the file name is the group name for all of the components in the EM file.

The EM file contains values in a resolution of two-minute intervals and as such the EM file will have a maximum of 724 lines. The platform mode part of the EM file is of no use in the

Standard format and as such it will be ignored when translating the EM file into the Standard format. All of the components in an EM file will be of the same group and as such will have the exact same component properties.

CP: Component Property CN: Component Name

Date	Time	Platform mode		Name	CP A	CP B	...	CP N		Name	CP A	CP B	...	CP N	...	Name	CP A	CP B	...	CP N
															...					
															...					
															...					
Date 1	00:00:00	Manual/Auto		CN A	Value	Value	...	Value		CN B	Value	Value	...	Value	...	CN M	Value	Value	...	Value
Date 1	00:02:00	Manual/Auto		CN A	Value	Value	...	Value		CN B	Value	Value	...	Value	...	CN M	Value	Value	...	Value
:	:	:		:	:	:		:		:	:	:		:	...	:	:	:		:
Date 1	23:58:00	Manual/Auto		CN A	Value	Value	...	Value		CN B	Value	Value	...	Value	...	CN M	Value	Value	...	Value

Table 6: The EM file format.

The ontology that was designed for the EM format can be seen in **Error! Reference source not found.**. The ontology exists out of a component group which contains a component group name, a file date and a component array. Each item in the component array consists of a component name and a component property array. The elements in the component array will each correspond to the different components in the EM file with the component name of each element corresponding to the values in the name column of that particular component. The file date will correspond to the dates given in the date column of the EM file.

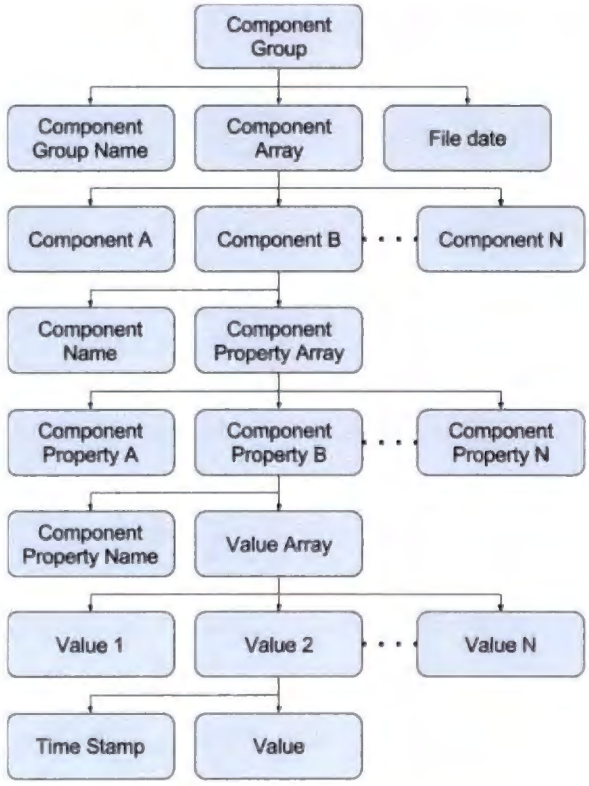

Figure 21: EM format ontology

Each element in the component property array consists of a component property name and a value array. The component property array corresponds to the values in the first row of the EM file next to the component name column and will repeat for every component. The value array is used to store the actual values of the specific component property. Every element in the value array consists of a time stamp and an actual value.

Ontology Translation – Because the Standard format has a resolution of 30 minutes instead of 2 minutes like the EM file format the translation from the EM format to the Standard format needs to aggregate the EM values to 30-minute interval values. This can be done by taking the average over a 30-minute interval for the Standard value.

To keep some of the information that will be lost when taking the average it has been decided to also use the minimum, maximum and the number of times the value has changed over the course of the 30 minutes. This means that when translating from the EM file format into the Standard format each component property in the EM file format will result in four Standard tags in the Standard format.

Whet translating from the EM file format to the Standard format the tag name is made up out of a combination of the project name, the EM component group name, the EM component name, the EM component property name and a tag tail. The tag tail is defined by the type of calculation performed to aggregate the 2-minute data to 30-minute data (average, min., max. and the number of status changes). **Error! Reference source not found.** shows the specific tails for the different calculation types. For an example of a Standard tag name build from an EM file, see **Error! Reference source not found.**. All of the different parts of the Tag name are separated by an underscore ("_").

Calculation Type	Tag tail
Average	""
Minimum	"min"
Maximum	"max"
Number of status changes	"StatusChange"

Table 7: EM file translation tag tails

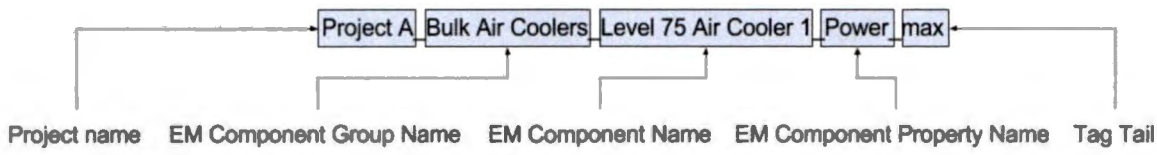

Figure 22: EM tag name example.

The Standard tag unit value will always be -1 and the date array will always contain only one date for the tags that are created from an EM file because the EM file may only contain one date. The value array is made up of the aggregate 30-minute values as mentioned above where the time intervals are the corresponding 30-minute time stamps.

EM file identification – a list of conditions has been drawn up that will be used to identify an EM file. They are given below:

- The file must be a CSV file with the file extension .csv
- The first three items in the first row must be "Date", "Time" and "Platform mode" in that order.
- The second, third and fourth line must be left completely blank.

Actual EM translator design – The first action that the EM translator will perform when it receives the list of files from the platform is that it will identify all of the EM files from that list. For each of the identified files, it will extract the data from the EM file into the EM ontology and determine how many standard tags will come from the data in the ontology (four Standard tags for each component property of each component). It will create the Standard tag array with one date in the date array of all the Standard tags and then translate the data in the EM ontology into these standard tags using the EM ontology translation given above.

The data in the Standard tag array will then be tested for conflicts with the data in the data folder and if such conflicts do exist the conflicting data will be removed from the Standard tag array and the original file will be copied from the inbox to the unprocessed folder. This is done so that a copy of the original file is kept for the NTHR should they need it to help resolve the merge conflicts.

After the original file has been copied or if no merge conflict have occurred the data in the Standard tag array are then appended to the data in the data folder and the original file in the inbox is deleted. Once this has been completed the file is removed from the list passed from the platform and the translator moves on to the next EM file. When all the EM files have been translated the list containing now only the files that were not identified as EM files get passed

back to the platform. A flow diagram demonstrating this process can be seen in **Error! Reference source not found.**.

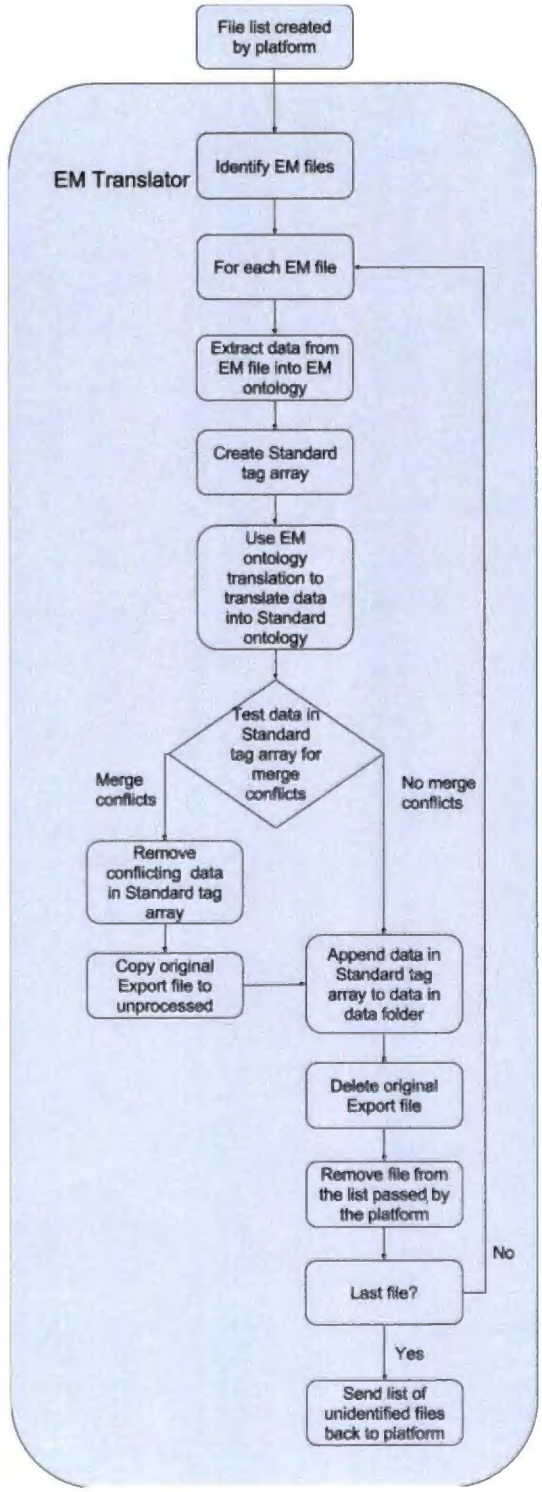

Figure 23: EM Translator

Evaluation of the Main Design Cycle

The same two projects that were used previously to test both the platform and the Export Translator were also used to test the EM translator. In addition to the eight Export files, four EM files have been added to each project to translate daily. The test was conducted over the course of two weeks. The results for the maintenance hours spent on managing and maintaining the two projects, with the EM files included, can be seen in **Error! Reference source not found.**.

	Day	Time spent on managing and maintaining Project A	Time spent on managing and maintaining Project B
Week 1	Monday	8 Minutes	12 Minutes
	Tuesday	10 Minutes	6 Minutes
	Wednesday	8 Minutes	8 Minutes
	Thursday	6 Minutes	6 Minutes
	Friday	8 Minutes	9 Minutes
	Saturday	-	-
	Sunday	-	-
Week 2	Monday	23 Minutes	31 Minutes
	Tuesday	10 Minutes	11 Minutes
	Wednesday	4 Minutes	11 Minutes
	Thursday	5 Minutes	5 Minutes
	Friday	6 Minutes	9 Minutes
	Saturday	-	-
	Sunday	-	-
	Average	**7 Minutes 4 Seconds**	**8 Minutes 43 Seconds**

Table 8: Time spent on maintenance for the two projects with the EM files included.

The average time spent on maintenance of Project A increased from 5 minutes and 8 seconds to 7 minutes and 4 seconds while the average for project B increased from 6 minutes and 14 seconds to 8 minutes and 43 seconds. Upon questioning the NTHR, they contributed this increase not only to the fact that more files were translated daily but also to the fact that more data existed in the data folder which made the maintenance of the projects harder to do.

The manner in which the platform appends data to the data in the data folder is challenging in that it leads to data for the same tag being scattered within the file. This has made determining

which data is already saved more time-consuming. Having scattered data as described also made it more complex and time-consuming to find the possible merge conflicts. All of this had a negative effect on the time it takes to do the maintenance on these projects.

The time spent on creating the EM translator was 23 man-hours which is 5 hours more than the time spent developing the Export translator for the first iteration of the platform. This can be contributed to the extra development required to create the different calculations needed to aggregate the 2-minute data intervals into 30-minute data intervals. It has also been noted that the part of the structure of the EM translator relating to the handling of merge conflicts are comparable to a large extent to that of the Export translator.

3.4. Third Iteration

Design Objectives for the Main Design Cycle

In this design iteration, there are two objectives. The first objective is to reduce the time needed for maintenance by the NTHR by designing a solution in which the data is not saved in the scattered manner as per the previous iterations. The second objective is to move the handling of the merge conflicts to the platform.

Design for the main Design Cycle

To only append data to a data file is easy and quick enough that is has been decided to leave it to the individual translators to perform. The realisation that there is a need for merge conflict handling and that the translators already handled appending the translated data to the data folder, determined that the logical place for the merge conflict handling was then to put it just before appending the data, in the translators themselves. This has led to duplicated stages across the translators. These duplicated stages unnecessarily prolong the design process of the translators and it was decided to move the merge conflict handling out of the designs of the translators and into the design of the platform.

The platform will still monitor the inbox folder for any new files. If new files have been detected the platform will build a list containing all these files, create an instance for each translator and then give each instance the file list. Each translator will then take the list given to it by the platform and identify which files are compatible with the translator. The compatible files will then be translated and the data along with a list containing the files not identified by the translator will then be given back to the platform.

After the platform has established which files have not been identified by any translator it will move those files to the unprocessed folder. It will also take the translated data and test it for any merge conflicts. If any merge conflicts occurred, it will remove those specific conflicts from the translated data and copy the original file from the inbox folder to the unprocessed folder. It will then insert the newly translated data into the data folder by merging rather than appending it to the data already in the data folder. After all this is done, the original data files in the inbox will be deleted by the platform. This can be seen in **Error! Reference source not found.**.

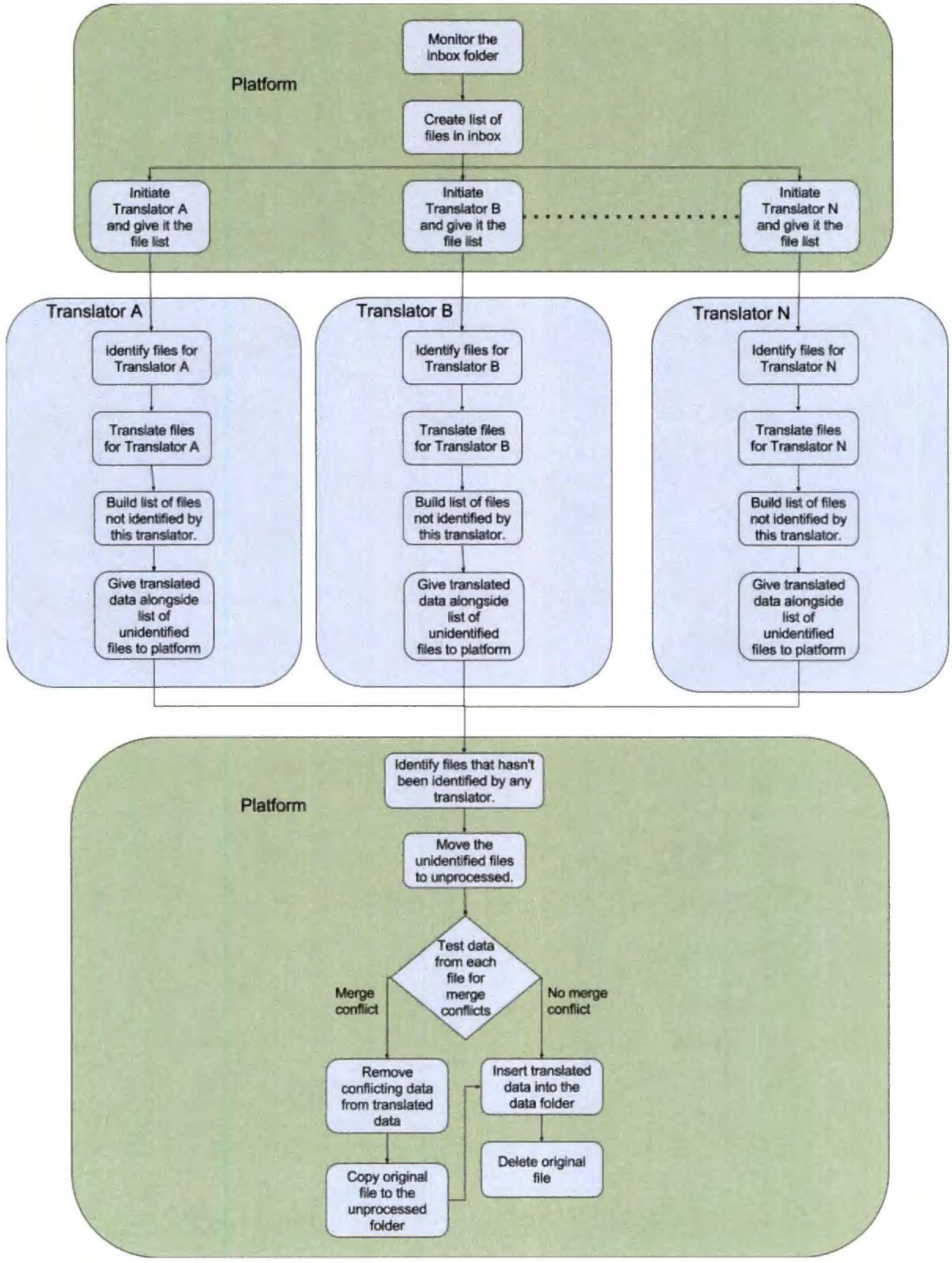

Figure 24: Fourth iteration platform and translator design.

Demonstration of the Main Design Cycle – The design of the Standard Translator

To demonstrate the workings of the platform designed in the fourth design iteration the Standard translator has been designed. There exists a need to save data already in Standard format to the data in the data folder. To save this type of data the Standard translator has been designed to take the Standard file and perform a one to one translation on the file, mapping the Standard ontology onto itself. The standard ontology can be seen in **Error! Reference source not found.**.

Standard file identification – A list of conditions has been drawn up which must be satisfied to successfully identify a file as a Standard file:

- The file must be a CSV file with a file extension of ".csv".
- The item in the first row of the first column must contain the words "Tag Name:"
- The item in the second row of the first column must contain the words "Tag ID:"
- The item in the third row of the first column must contain the words "Tag Unit:"

The Standard translator will take the file list given to it by the platform and identify which of these files are standard files according to the Standard file identification given above. For each standard file, it will then extract the data from the file into the Standard ontology. Due to the fact that the Standard ontology translation is an ontology map onto itself, no ontology translation is needed. The file that is now translated is then removed from the file list given to the Standard translator by the platform and the process is repeated for each of the following Standard files in the list.

Once all of the Standard files have been translated the list now containing only files that are not standard files are given back to the platform along with the translated data. This can all be seen in **Error! Reference source not found.**. The Export and EM translators have also been changed in a similar manner to work with the platform designed in the fourth iteration.

Evaluation of the Main Design Cycle

Along with the eight Export files and the four EM files, five Standard files have been added to the daily translations for Project A and B. The test was again run for two weeks and again the time it took to do the maintenance by the NTHR on the two projects was measured. The results can be seen in **Error! Reference source not found.**.

The average time spent on maintenance for project A was 6 minutes and 13 seconds while the average time spent on maintenance for project B was 6 minutes and 56 seconds. In both cases,

the average time spent is less than with the previous iteration although additional standard files were also translated. When the NTHR were questioned about this decrease, they said that the maintenance was now easier and quicker because all the data belonging to a particular tag was now located in the same place in the file.

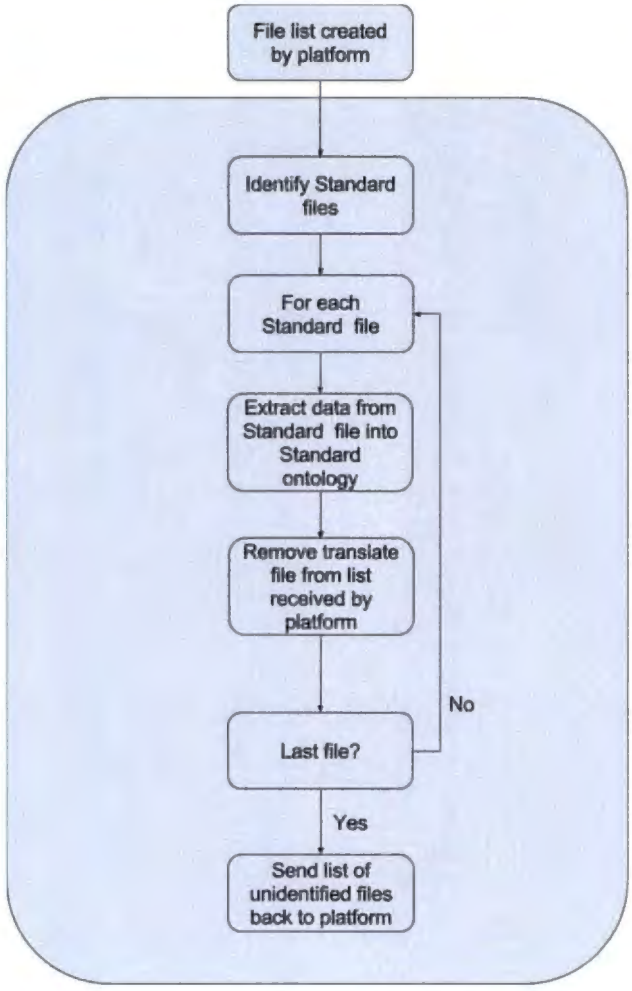

Figure 25: Design of the Standard translator

The development of the Standard translator itself took 10 man-hours to design and implement which is much lower than the time it took to design and implement the Export and EM translators. This can be contributed to the fact that the Standard ontology has already been designed and that because of the one to one mapping the ontology translation was not needed. It can also be contributed to the fact that the merge-conflict handling was moved to the platform itself and once the data has been translated the responsibilities of the translator has ended. This made the development of the standard translator faster.

In this design iteration, the performance of the platform using Project A and B was evaluated to be satisfactory. Reaching this point one is tempted to say that a solution to the problem has

been designed and that the research is complete. Although it might be true that the research questions have been answered, the validation of those answers has only been proven in the context and environment of project A and B.

To truly say that a solution has been found for the problem, the solution needs to be verified by testing and evaluating it within the complete environment. Although an effort has been made to keep the environment of Project A and B as close to that of the true environment; the reality is that the environment is not the true environment.

	Day	Time spent on managing and maintaining Project A	Time spent on managing and maintaining Project B
Week 1	Monday	7 Minutes	9 Minutes
	Tuesday	11 Minutes	5 Minutes
	Wednesday	4 Minutes	7 Minutes
	Thursday	5 Minutes	3 Minutes
	Friday	9 Minutes	5 Minutes
	Saturday	-	-
	Sunday	-	-
Week 2	Monday	25 Minutes	27 Minutes
	Tuesday	6 Minutes	8 Minutes
	Wednesday	5 Minutes	9 Minutes
	Thursday	3 Minutes	3 Minutes
	Friday	4 Minutes	7 Minutes
	Saturday	-	-
	Sunday	-	-
	Average	**6 Minutes 13 Seconds**	**6 Minutes 56 Seconds**

Table 9: Time spent on maintenance with the Standard files included

Therefore to verify that the solution satisfies the problem the solution needs to be implemented and tested within the complete environment. To do this a final iteration will be conducted where the solution will be implemented within the full context of the problem. That iteration is given in Section **Error! Reference source not found.**.

3.5. Final Iteration (Validation)

3.5.1. Introduction

In the following section, the solution, within the complete environment, will be demonstrated and evaluated. To do this, one more translator has been developed and evaluated alongside the other translators. This evaluation was done not within project A and B but within the complete environment. The translator that was developed is the PDI translator. At the end of this section, the overall performance of the platform is given and evaluated.

3.5.2. Development of the Final Translator

The PDI file is a CSV file with the file extension of ".pdi". It is generate by using the Pentaho Data Integration software package to extract data from one of the data accumulators. It will always have four columns and the values in the first row of the file will always be "Date", "Time", "TagName" and "Value" in that order. The time interval in the PDI file will always be 30 minutes but the values in the file do not necessarily have to be in a specific order. All of the values of a specific tag are usually grouped together but this is also not always the case. An example of such a file can be seen in **Error! Reference source not found.**.

Date	Time	TagName	Value
Date 1	00:30:00	Tag A	value
Date 1	01:00:00	Tag A	value
:	:	:	:
Date 1	24:00:00	Tag A	value
Date 2	00:30:00	Tag A	value
:	:	:	:
Date N	24:00:00	Tag A	value
Date 1	00:30:00	Tag B	value
:	:	:	:
Date 1	24:00:00	Tag B	value
Date 2	00:30:00	Tag B	value

Table 10: Example of a PDI file

The ontology designed for the PDI format looks similar to that of the Standard ontology. It consists of a tag array where the number of items in the tag array corresponds to the number of

different tags found in the TagName column of the PDI file. Each tag item consists of a tag name (as it is found in the TagName column) and a date array. The items in the date array correspond to the dates found in the Date column of the PDI file for that particular tag.

Every date item consists of a date and a value array where each item in the value array corresponds to a time-value. The value item then consists of a time-value as it is found in the Time column of the PDI file and an actual value as it is found in the Value column of the PDI file. A visual representation of the PDI ontology can be found in **Error! Reference source not found.**.

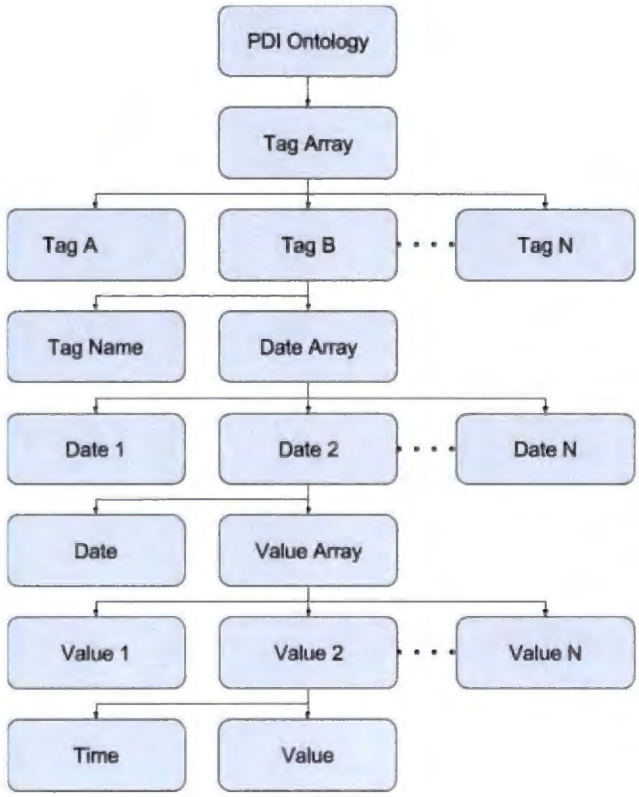

Figure 26: PDI Ontology

A list of conditions has been drawn up for a PDI file to adhere to in order for that file to be identified as a PDI file:

- The file must be a CSV file with the file extension ".pdi".
- The file must only consist of four columns.
- The values of the items in the first row must be "Date", "Time", "TagName" and "Value" in that order.

PDI ontology translation – Due to fact that the PDI ontology and the Standard ontology are so similar; the mapping between the two ontologies is very straight forward. Each PDI tag in the PDI tag array will map to a Standard tag in the Standard tag array. The Standard tag name will

be mapped to the PDI tag name. The Standard tag ID and tag unit will both be given a value of for all the tags as these values cannot be derived from the PDI file itself.

Each item in the date array of the PDI ontology will be mapped to an item in the date array of the Standard ontology. The same goes for the items in the value arrays of the respective ontologies. Each item in the value array of the PDI value array will be mapped to an item in the Standard value array. Seeing as the time intervals between the values in the PDI file are 30-minute intervals, each time value in the PDI ontology will be mapped to a 30-minute profile in the Standard ontology.

The PDI translator will start off by taking the list given to it by the platform and identifying which of the files are PDI files. For each of the PDI files, it will then extract the data in the file into the PDI ontology. The PDI ontology translation will be used to translate the data in the PDI ontology to Standard data and the original file will be removed from the list passed to the translator by the platform. This will be repeated for each PDI file in the list. Once all the PDI files have been translated the translated data along with the list of files (which no longer contains any PDI files) will be passed back to the platform. This can be seen in **Error! Reference source not found.**.

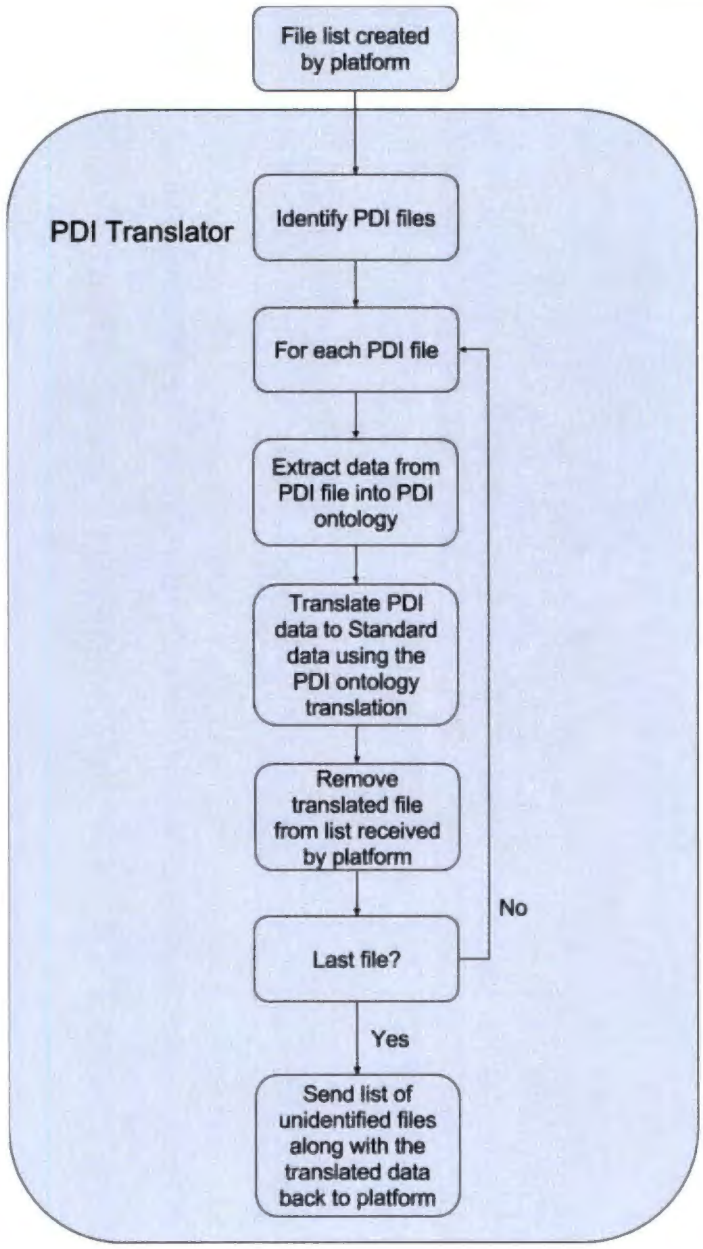

Figure 27: Design of the PDI translator

3.5.3. Evaluation of Solution in the complete environment

The platform was rolled out to handle the translation of eight industrial groups and the performance of the platform was validated over a period of one month. Each client group is made up of different systems where each system usually corresponds to a different division within the client group. The systems from across all of the industrial groups; is comprise of 82 sub-systems.

These systems each have between 20 and 40 files that must be translated daily. From all of the files across all of the systems a total of 5808 tags were extracted and translated and 4458 of those tags were daily tags (tags that contain values for a single day) while the rest of the tags are monthly tags. This means that 4458 of the tags get translated daily while the rest of the tags only get translated once a month. The division of tags originating from the different file types can be seen in **Error! Reference source not found.**.

File type	Number of Standard tags
EM	1088
Standard	1631
Export	1048
PDI	2041

Table 11: Standard tag division across originating files

Having all these tags successfully translated and saved within the data folders of the respected systems depicts the success of the platform. The fact that all of these tags were successfully saved implies that the respected translators successfully translated the data from their respective original file formats. The fact that the translators successfully translated the data implies that the platform successfully allows for the creation of translators.

The NTHR were asked to record the time they spent on managing and maintaining the flow of data throughout the platform. This included ensuring and monitoring that the platform received the data that must be translated on a daily basis and the handling of any merge conflicts. Given in **Error! Reference source not found.** is the total number of man-hours spent on monitoring and maintaining each file type.

File Type	Average man-hours per day per tag (min)	Average time spent per tag (sec)
EM	32	1.76
Standard	46	1.69
Export	41	2.35
PDI	53	1.56
Total	172	

Table 12: Time spent by NTHR on managing and maintaining the systems

The THR were responsible for designing and implementing the different translators on top of the platform. The time it took them to do this is given in **Error! Reference source not found.**.

File type	The time it took to develop the translator. (hours)
EM	23
Standard	10
Export	18
PDI	15

Table 13: Time it took to develop translators

As stated before this problem is ultimately an automation problem. It is by no means a problem that cannot be solved by a lot of people doing the translations by hand. As such the true value of the solution can be measured by measuring the time difference between the man-hours it took to do the translation by hand in the past versus the time it takes to create, maintain and manage the translation within the platform.

To do this the time it takes to do a translation by hand and merge that data with data already in the data folder has been measured for each of the file types. The time it took for that translation was then divided by the number of standard tags resulting from that translation to get a per standard tag time and compared to the per standard tag time it takes to create, manage and maintain the translation with the platform. This can be seen in **Error! Reference source not found.**.

File type	Time for file (sec)	Number of tags (sec)	Time per tag (sec)
Standard	374	10	37.4
Export	532	10	53.2
EM	2278	8	284.75

PDI	750	10	75

Table 14: Manual Tag Conversion Time

Knowing this it is possible to estimate the time it would have taken to manually translate the data used to test the translators and to then compare that with the time it took to create the translator and to then manage and maintain that translator. A graphical representation comparing the manual time versus the platform time it takes to translate the tags from the four file types can be seen in **Error! Reference source not found.** through to **Error! Reference source not found.**. Included in the amount of time human resources (indicated by the "Platform" line) needed to translate the first tag of a given file type is the time spent to create the translator.

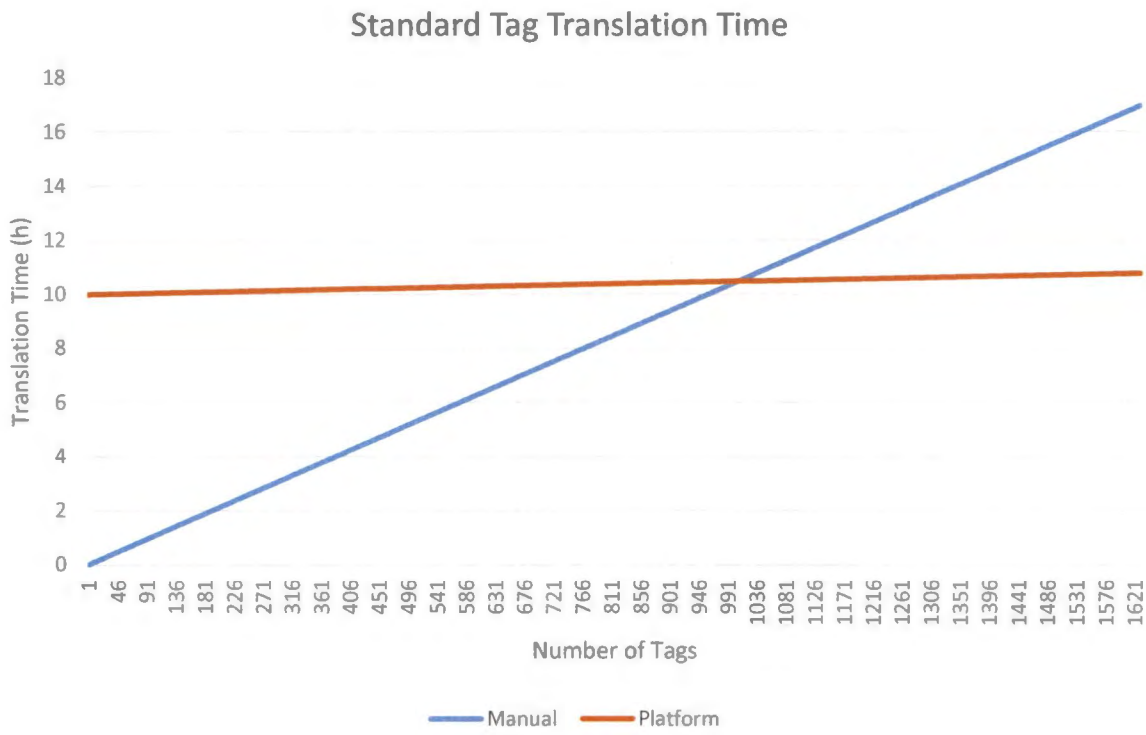

Figure 28: Standard Tag Translation Time Comparison

In **Error! Reference source not found.** the time (man-hours) it would take to translate files in the standard format manually (without using the platform) is compared to the time it would take to translate the files using the platform. For this comparison the number of standard tags that would typically be received in one month is used (typically this would be 1631 tags).

The data form a Standard file is already in the standard format so one would think that the manual translation time needed would be small but the data from the standard file still needs

to be saved in the monthly files. In the monthly files the data must still be sorted and grouped together so the data is not scattered (see Section **Error! Reference source not found.**); this takes time to perform manually.

That time is shown by the blue line in **Error! Reference source not found.**. To use the platform to perform the translation a standard translator must first be designed. This means that the time before the first tag is translated is long (10 hours) but after that the time needed per standard tag is much less than doing the translation manually. This means that from the 1009th tag it would require less man-hours to undertake the translation using the platform rather than doing the translation manually.

After translating 1631 Standard tags (the typical amount of Standard tags that must be translated in one month) a total of 6 hours and 10 minutes will be saved using the platform. As time goes on and more tags get translated the amount of time saved will only increase. Especially seeing as the Standard translator only has to be designed once; this means that for the following months the total number of man-hours saved each month by using the platform would be an estimated 16 hours and 10 minutes. In the first year this leads to an estimated 184 hours and 8 minutes saved by using the platform.

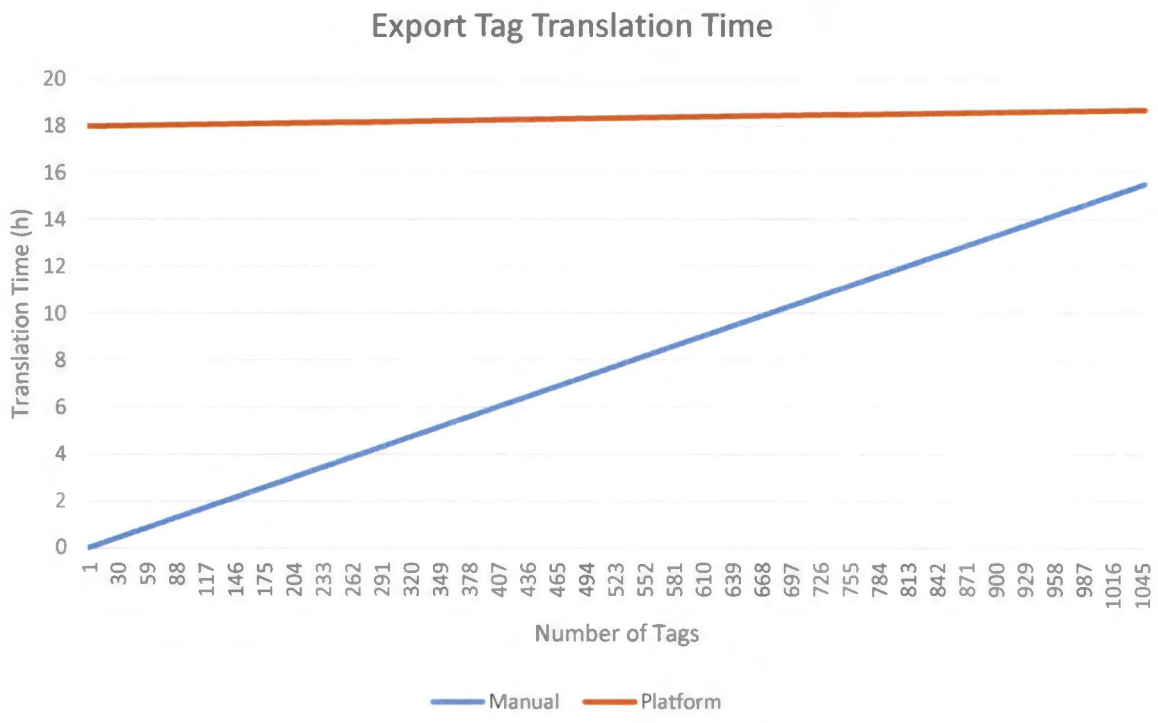

Figure 29: Export Tag Translation Time Comparison

In this figure (**Error! Reference source not found.**) the time in man-hours to translate Export tags manually is compared to having the platform translate them. In this comparison the number of tags that would typically be received in one month is used. Typically this would be around 1048 tags.

The blue line represents the time needed to translate the tags manually while the red line represents the time needed to do the translation using the platform. It takes 18 man-hours to design the translator so the first tag that is translated using the platform is only translated after 18 hours. After that the time needed to translate the tags using the platform is much faster than doing the translation manually.

In one month a total of 1048 tags must typically be translated. The export translator took a long time to design and because of this the benefit of using the platform to translate data from export format will typically not be realised in the first month of using the platform. But studying **Error! Reference source not found.** it is clear that by including more tags one would start to benefit from using the platform from month two onwards. In the first year this benefit is estimated to be around 159 hours and 38 minutes by using the platform.

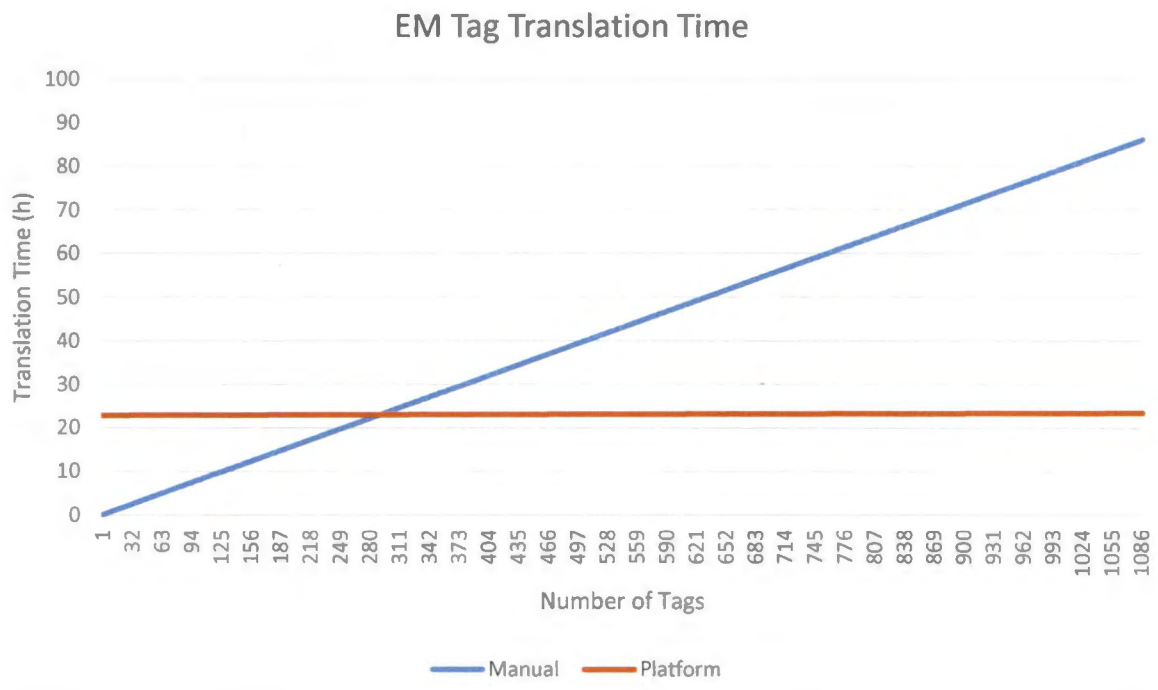

Figure 30: EM Tag Translation Time Comparison

Error! Reference source not found. shows the comparison (translating tags manually versus translating tags using the platform) for the EM translator. In one month the platform would typically need to translate around 1086 tags.

As with the other figures, the blue line represents the time it takes to translate the tags manually while the red line represents the time it take to translate the tags using the platform. The time it took to create the EM translator is 23 hours. As with the others, after the translator has been created it takes a lot less time to translate the tags with the platform than doing the translation by hand. This means that from the 293rd tag it would require less man-hours doing the translation using the platform than doing the translation manually.

The time saved in one month by using the platform rather than doing the translation manually is typically around 62 hours and 31 minutes. In the following months (because the translator only has to be created once) a total of 85 hours and 31 minutes will be saved each month. This adds up to an estimated 1003 hours and 18 minutes saved by using the platform in the first year.

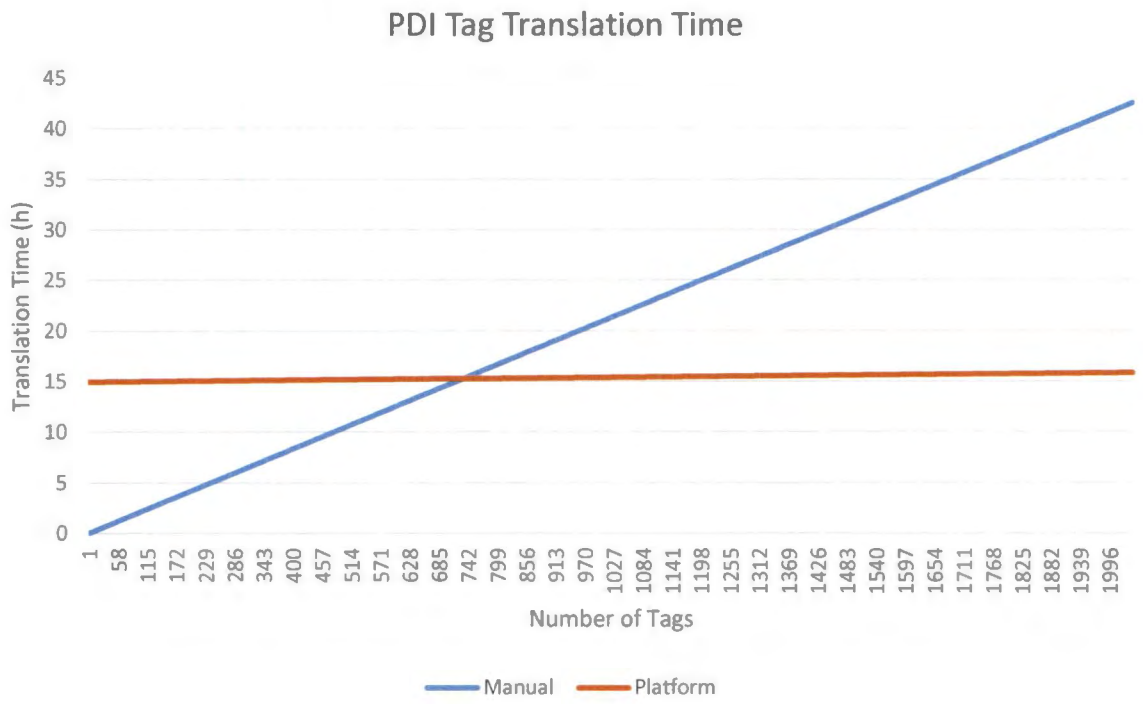

Figure 31: PDI Tag Translation Time Comparison.

In **Error! Reference source not found.** the time comparison for translating the PDI file manually versus using the platform is given. The blue line represents the manual time and the red line represents the platform time. In one month there typically needs to be 2041 tags translated.

It took 15 man-hours to create the PDI translator and the benefit of using the translator (when it becomes less time consuming to use the platform to do the translation rather than doing the translation manually) to translate files is realized after the 736th tag is translated.

In the first month 26 hours and 38 minutes will be saved using the platform. From the second month onward (because the translator only needs to be created once) a total of 44 hours and 38 minutes will be saved each month. This translates to an estimated 517 hours and 38 minutes saved in the first year by using the platform.

3.6. Conclusion

Studying the graphs in **Error! Reference source not found.** through to **Error! Reference source not found.** the value the platform adds (in terms of the man-hours needed to translate the data into the standard format) is evident. It takes a long time to create a translator on the platform but once the translator is created the time needed to manage and maintain the

translation processes (per tag that is translated) is a lot less compared to the time needed to translate that tag by hand.

The results shown in the graphs are only for the first month of implementation. The longer a specific translator is used, the more tags it will translate and the more value (in terms of man-hours saved) will be gained from using the platform. The estimated man-hours saved for the first year for all four file types by using the platform is 1864 hours and 43 minutes.

4. Conclusion

4.1. Revisit research questions

As properly defined in Chapter **Error! Reference source not found.**, a need exists for a translation platform. This need arises from the fact that data across different formats must be analysed together and to satisfy that need the platform must be able to translate the data from the different data formats into a predefined standard format. These formats that must be translated changes frequently and due to that fact the platform must allow for the creation and modification of new translators. In researching a solution to this, the following research questions were developed:

Main Question: How would the design of such a solution look?

1.1. What role does the nature of the data play on the design of the solution?
1.2. What is the effect of the internal structure of the company on the design of the solution?
1.3. How do the available human resources change the design of the solution?

In the previous sections, it has been shown that a translation platform has successfully been designed and the value of this platform has been shown. This proves that the design reached at the end of the last design cycle, successfully allowed for the creation of a platform that solves the problem identified. This design and how it relates and answers the research questions will now be discussed further.

It has been shown that this platform allows for the creation of translators to translate files from diverse file formats into the standard format. It has also been shown that by using these translators to translate the data into the standard format, man-hours will be saved by reducing the amount of time needed to translate the data. All of this proves that a solution has been reached that answers the main research question.

It is a common problem that the platform receives conflicting data. This was handled by designing the platform in such a way that the NTHR are able to resolve conflicts and by designing the platform to, by default, accept the first value as correct. This forms part of the answer to research questions 1.1, 1.2 and 1.3. The nature of the data is such that merge conflicts occur and that they occur more often that originally predicted. To this end the nature of the data influences the design of the platform and as such answers research question 1.1 by dictating that the platform specifically be designed to handle these conflicts.

There is no way for the platform to know which conflicting value is the correct value. This states that the design of the platform should allow for interaction with humans to resolve any conflicts that may arise. The internal structure of the company dictates that the NTHR are best positioned to know which of the conflicting values would be the correct value. This shows how the internal structure of the company influences the design of the platform and as such forms part of the answer to research question 1.2.

It is thus necessary to design the platform in such a way that the NTHR can resolve the conflicts and it is important that the platform takes into account the technical skill level of the NTHR. This shows how the interaction of the platform with the human resources influences the design of the platform and as such forms part of the answer to research question 1.3.

The data that must be translated by the platform is of the nature that it is typically stored behind firewalls. It has been decided that the best way to securely gain access to this data was to email it to the platform. This is a direct response to how the nature of the data (that of typically being stored behind a firewall) affects the design of the platform and as such is a direct answer to research question 1.1.

Emailing the data to the platform was also found to be a good design decision due to the internal structure of the company dictating that the NTHR be responsible for setting up and ensuring that the platform receives the data that requires translation. This way the NTHR can use something that is already familiar to them.

The fact that the platform must be designed to allow the NTHR to set up and ensure the platform receives the data is a direct response to the internal structure of the company and as such answers research question 1.2. Using emails (a technology that is already known to the NTHR) shows how the interaction between the available human resources and the platform affected the design of the platform and as such is a direct answer to research question 1.3.

As previously stated, the internal structure of the company dictates that it should be the NTHR that monitors and maintains the flow of data throughout the platform and designing the platform to allow this is a direct answer to research question 1.2. Knowing this, it is therefore a practical design decision to save the data using normal CSV's ; something that is commonly used by programs like Excel. It is one of the areas where the platform interacts with human resources and using normal CSV's means that there is a simple and easy manner for humans to look through and search through the data. As such this forms part of the answer to research question 1.3.

To further help in this regard, it has been decided that the tags should be sorted alphabetically and the data for a single tag should not be scattered throughout the file. This further reduces the time and effort needed by the NTHR to monitor and manage the translation process and as such also forms part of the answer to research question 1.3. This speaks to how the platform eases the interaction between the human resources and the platform.

Likewise, storing the data should also be split up into monthly files so that a file only contains data for a single month. Doing this ensures that the file sizes remains small thereby also reducing the maintenance time needed by the NTHR. This also eases the interaction between the human resources and the platform and as such also forms part of the answer to research question 1.3.

4.2. Discussion

The world is becoming ever more data orientated and the need for the value that can be derived from analysing more of the data together is also increasing. The need for translating data into some sort of standard format before performing such analysis will become increasingly important. This is true for all industries that follow the current Big Data explosion.

Clearly, the available human resources, the nature of the data and the internal structure of the company played a big role in dictating the exact design of the platform. At the beginning of the study, it was unclear how these factors would influence the design of the platform. By using design science principles it was possible to discover how and to what extent the design of the platform should be adapted, modified and developed to accommodate these factors.

The fact that these factors had such a large influence over the design of the platform implies that a generic design that caters for all possible variances of these parameters is most likely unachievable. This is echoed by the relevance that was forced onto the design of the artefact by following the design science principles.

As stated earlier Design Science Research requires relevance (see Section **Error! Reference source not found.**). The way Design Science Research enforces relevance is by evaluating each of the Design cycles for how well it addresses the problem at hand. By using the results of these evaluations as input to the next design cycle it is ensured that the artefact in the next design cycle addresses the problem to a higher degree, thereby ensuring relevance. The relevance in this study ultimately boils down to the amount of man-hours saved by using the platform. In this study, it added up to 1864 hours and 43 minutes that was saved, enough that a full-time graduate employee would have been needed if the platform was not used.

The areas where the artefact lacked or inhibited relevance in the various forms of the artefact, as it went through the design cycles were not areas that are generic in nature. It is areas that are specific to the exact environment the platform is used in:

- The nature of the specific data that needs to be translated.
- The specific internal structure of the company in which the platform will be used in.
- The specific human resources that are available to the platform.

It means that not designing for these specific areas will make the overall design lose some of its relevance, ultimately leading to fewer man-hours being saved.

A more realistic approach to finding a generic solution (a solution that caters for all possible variances) would be to rather develop design guidelines that can help in guiding specific designs to cater for the specific requirements that the specific variance on these parameters places on that design. This way these guidelines can be used to create multiple designs that are custom made, if you will, to the specific situations these variances on the platform will be used in.

4.3. Future Research

Managing and maintaining the flow of data throughout the platform is one of the areas of the platform that takes up the most man-hours. This has been addressed by saving the data in monthly files and by ensuring that the data gets saved in alphabetical order (see Section **Error! Reference source not found.**). Although this has greatly reduced the man-hours it can still take a long time when the monthly files grow larger in size. One suggestion for future research would be to find a better way in which to manage and save the translated data in order to reduce the man-hours required for this.

Another suggestion for future work would be to incorporate a validation mechanism into the platform. It happens that the data received by the platform is sometimes incorrect. This can lead to conflicts within the data however the platform was adapted to handle these (see Section **Error! Reference source not found.**). In the case where the incorrect data does not lead to conflicts within the data, it is still the responsibilities of the NTHR to identify and correct them. Having a mechanism in place that can be used to set up validation criteria for different tags that can be used to identify some or all of the incorrect data will to a large extent decrease the man-hours spent by the NTHR.

Lastly, following on the discussion in Section **Error! Reference source not found.**, a suggestion for future research into design guidelines to govern the design of a platform like this is made. Knowing that these factors (the nature of the data, the internal structure of the company and the human interaction with the platform) play such a large role in the relevance of the design, it might be wise to search for design guidelines rather than a design for a more general platform. These design guidelines need to assist in designing multiple platforms where they (the guidelines) govern the design of each of the platforms to better accommodate the exact environment each platform will be used in.

References

Arasteh, H., Sepasian, M. S., Vahidinasab, V., & Siano, P. (2016). SoS-based multiobjective distribution system expansion planning. *Electric Power Systems Research*, *141*, 392–406. http://doi.org/10.1016/j.epsr.2016.08.016

Cai, Y., Chen, W. H., Leung, H. F., Li, Q., Xie, H., Lau, R. Y. K., … Wang, F. L. (2016). Context-aware ontologies generation with basic level concepts from collaborative tags. *Neurocomputing*, *208*, 25–38. http://doi.org/10.1016/j.neucom.2016.02.070

Cantu-Ortiz, F. J. (2014). Advancing artificial intelligence research and dissemination through conference series: Benchmark, scientific impact and the MICAI experience. *Expert Systems with Applications*, *41*(3), 781–785. http://doi.org/10.1016/j.eswa.2013.08.008

Chang, V., Ramachandran, M., Wills, G., Walters, R. J., Li, C. S., & Watters, P. (2016). Editorial for FGCS special issue: Big Data in the cloud. *Future Generation Computer Systems*, *65*, 73–75. http://doi.org/10.1016/j.future.2016.04.007

Clark, J. O. (2009). System of Systems Engineering and Family of Systems Engineering from a standards, V-Model, and Dual-V Model perspective. In *3rd Annual IEEE Systems Conference* (pp. 381–387). http://doi.org/10.1109/SYSTEMS.2009.4815831

de Farias, T. M., Roxin, A., & Nicolle, C. (2016). SWRL rule-selection methodology for ontology interoperability. *Data & Knowledge Engineering*, *105*, 53–72. http://doi.org/10.1016/j.datak.2015.09.001

Forsati, R., & Shamsfard, M. (2016). Symbiosis of evolutionary and combinatorial ontology mapping approaches. *Information Sciences*, *342*, 53–80. http://doi.org/10.1016/j.ins.2016.01.025

Ge, B., Hipel, K. W., Yang, K., & Chen, Y. (2014). A novel executable modeling approach for system-of-systems architecture. *IEEE Systems Journal*, *8*(1), 4–13. http://doi.org/10.1109/JSYST.2013.2270573

Gruber, T. R. (1993). Translation Approach to Portable Ontology Specification. *Knowledge Acquisition*, *5*(2), 199–220. http://doi.org/http://dx.doi.org/10.1006/knac.1993.1008

Hevner, A. R. (2007). A Three Cycle View of Design Science Research. *Scandinavian Journal of Information Systems*, *19*(2), 87–92.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in the Information Systems Research. *MIS Quaterly*, *21*(1), 75–105.

IEEE. (2010). Interoperability (Deffinition). Retrieved August 9, 2016, from http://www.ieee.org/education_careers/education/standards/standards_glossary.html

Jamshidi, M. (2010). *From large-scale systems to system of systems—Control challenges for the 21st century. Large Scale Complex Systems Theory and Applications, Vol.9 | Part.1,France* (Vol. 43). IFAC. http://doi.org/10.3182/20100712-3-FR-2020.00004

Johnson IV, J. J., Tolk, A., & Sousa-Poza, A. (2013). A theory of emergence and entropy in systems of systems. *Procedia Computer Science*, *20*, 283–289. http://doi.org/10.1016/j.procs.2013.09.274

Kuechler, W. L., & Vaishnavi, V. K. (2008). The Emergence of Design Research in Information Systems in North America. *Journal of Design Research*, *7*(1), 1 – 16.

Kuhn, T. S. (1970). *The Structure of Scientific Revolutions. Philosophical Review* (Vol. II). http://doi.org/10.1119/1.1969660

Larson, D., & Chang, V. (2016). A review and future direction of agile, business intelligence, analytics and data science. *International Journal of Information Management*, *36*(5), 700–710. http://doi.org/10.1016/j.ijinfomgt.2016.04.013

Lassiter, C. (2016). Aristotle and distributed language: Capacity, matter, structure, and languaging. *Language Sciences*, *53*, 8–20. http://doi.org/10.1016/j.langsci.2015.05.011

Laurentis, D. De, Dickerson, C., DiMario, M., Gartz, P., Jamshidi, M. M., Nahavandi, S., … Walker, D. R. (2007). A Case for an International Consortium on System-of-Systems Engineering. *IEEE Systems Journal*, *1*(1), 68–73. http://doi.org/10.1109/JSYST.2007.904242

Loebbecke, C., & Picot, A. (2015). Reflections on societal and business model transformation arising from digitization and big data analytics: A research agenda. *Journal of Strategic Information Systems*, *24*(3), 149–157. http://doi.org/10.1016/j.jsis.2015.08.002

Lokers, R., Knapen, R., Janssen, S., van Randen, Y., & Jansen, J. (2016). Analysis of Big Data technologies for use in agro-environmental science. *Environmental Modelling and Software*, *84*, 494–504. http://doi.org/10.1016/j.envsoft.2016.07.017

March, S. T., & Smith, G. F. (1995). Design and natural science research on information technology. *Decision Support Systems*, *15*(4), 251–266. http://doi.org/10.1016/0167-9236(94)00041-2

Mason, R. O. (1986). Four Ethical Issues of the Information Age. *MIS Quarterly*, *10*(1), 5–13. http://doi.org/10.2307/248873

Mecca, G., Rull, G., Santoro, D., & Teniente, E. (2015). Ontology-based mappings. *Data and Knowledge Engineering*, *98*, 8–29. http://doi.org/10.1016/j.datak.2015.07.003

Myers, M. D., & Venable, J. R. (2014). A set of ethical principles for design science research in information systems. *Information Management*, *51*(6), 801–809. http://doi.org/10.1016/j.im.2014.01.002

Nunamaker, J., Chen, M., & Purdin, T. (1991). Systems development in Information Systems research. *Journal of Management Information Systems*, *7*(3), 89–106. http://doi.org/10.1109/ISIE.1992.279627

Orpha Cornelia Lombard, by, Gerber Co-supervisor, A., & van der Merwe, A. (2014). *The Construction and Use of an Ontology to Support a Simulation Environment Performing Countermeasure Evaluation for Military Aircraft*. University of South Arica.

Owen, C. L. (1998). Design research: building the knowledge base. *Design Studies*, *19*(1), 9–20. http://doi.org/10.1111/j.1365-2354.2009.01132.x

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, *24*(3), 45–77. http://doi.org/10.2307/40398896

Perrons, R. K., & McAuley, D. (2015). The case for "n??all": Why the Big Data revolution will probably happen differently in the mining sector. *Resources Policy*, *46*, 234–238. http://doi.org/10.1016/j.resourpol.2015.10.007

Reinecke, K., & Bernstein, A. (2013). Knowing What a User Likes: A Design Science Approach to Interfaces that Adapt to Culture. *MIS Quarterly*, *37*(2), 427–453.

Simon, H. a. (1997). *The sciences of the artificial, (third edition). Computers & Mathematics with Applications* (Vol. 33). http://doi.org/10.1016/S0898-1221(97)82941-0

Skjæveland, M. G., Giese, M., Hovland, D., Lian, E. H., & Waaler, A. (2015). Engineering ontology-based access to real-world data sources. *Journal of Web Semantics*, *33*, 112–140. http://doi.org/10.1016/j.websem.2015.03.002

Stary, C., & Wachholder, D. (2016). System-of-systems support - A bigraph approach to interoperability and emergent behavior. *Data and Knowledge Engineering*, *105*, 155–172. http://doi.org/10.1016/j.datak.2015.12.001

Tanenbaum, A. S., & Van Steen, M. (2007). *Distributed Systems: Principles and Paradigms, 2/E. Communication.* http://doi.org/10.1002/1521-3773(20010316)40:6<9823::AID-ANIE9823>3.3.CO;2-C

Tannahill, B. K., & Jamshidi, M. (2014). System of Systems and Big Data analytics – Bridging the gap. *Computers & Electrical Engineering*, *40*(1), 2–15. http://doi.org/10.1016/j.compeleceng.2013.11.016

Vaishnavi, V., & Kuechler, B. (2004). Design Science Research in Information Systems Overview of Design Science Research. http://doi.org/10.1007/978-1-4419-5653-8

Walls, J. G., Widmeyer, G. R., El Sawy, O. A., & Sawy, O. A. E. (1992). Building an information system design theory for vigilant EIS. *Information Systems Research*, *3*(1), 36–59. http://doi.org/10.1287/isre.3.1.36

Wang, H., Xu, Z., Fujita, H., & Liu, S. (2016). Towards felicitous decision making: An overview on challenges and trends of Big Data. *Information Sciences*, *367–368*, 747–765. http://doi.org/10.1016/j.ins.2016.07.007

Weichhart, G., Guédria, W., & Naudet, Y. (2016). Data & Knowledge Engineering Supporting interoperability in complex adaptive enterprise systems : A domain speci fi c language approach. *Data & Knowledge Engineering*, *105*, 90–106. http://doi.org/10.1016/j.datak.2016.04.001

Yaqoob, I., Hashem, I. A. T., Gani, A., Mokhtar, S., Ahmed, E., Anuar, N. B., & Vasilakos, A. V. (2016). Big data: From beginning to future. *International Journal of Information Management*, *36*(6), 1231–1247. http://doi.org/10.1016/j.ijinfomgt.2016.07.009

Zakharenko, R. (2016). Self-Driving Cars Will Change Cities. *Regional Science and Urban Economics*, *61*(October 2015), 26–37. http://doi.org/10.2139/ssrn.2683312