TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

HOUD BOUCHELOUCHE

REAL-TIME BUSINESS INTELLIGENCE SOLUTION

Master of Science thesis

Examiner: Prof. Kari Systä
Examiner and topic approved by the
Faculty Council of the Faculty of
Computing and Electrical Engineering
on 30th November 2017

ABSTRACT

HOUD BOUCHELOUCHE: Real-Time Business Intelligence Solution
Tampere University of Technology
Master of Science thesis, 54 pages
December 2017
Master's Degree Programme in Information Technology
Major: Pervasive Systems
Examiner: Prof. Kari Systä
Keywords: Business Intelligence, Data Warehouse, Real-time, Data Analytics

Nowadays, terms such as the Internet of Things, Big Data analytic and Business Intelligence are very common among business leaders, managers and analysts. There is no wonder why, acquiring a huge amount of data with the ability to store it, process it and analyze it implies better and more informed business decisions made by corporate decision makers. However, some of these technologies are still maturing. Therefore, a new approach and patterns for designing solutions consisting of such technologies will play an important role in accelerating and improving their implementation.

This thesis work will focus on the study, design and implementation of a real-time BI(Business Intelligence) solution that gets data from operational databases as well as other real-time platforms such as Meter Data Management Systems. The BI will allow end-users to visualize data, view reports and access dashboard for both decision-makers and operational workers.

The developed prototype is able to extract data from disparate data sources and store it into a data warehouse using dimensional modelling. The solution is also able to extract real-time data from an MDMS(Meter Data Management System) to provide real-time analytics for smart meters data. Some tools and technologies that are used in the BI domain were studied, and the selected ones were used for implementing the prototype. Qlik Sense is used for data visualization and reporting. Talend Open Studio is used for the development of ETL process.

PREFACE

The work presented in this thesis has been carried out at the premises of Newelo Oy, where I had the chance to learn more about the world of Business Intelligence and put into practice what I have been learning during my studies.

I would like to start by thanking Newelo Oy for giving me the opportunity to be part of their team and work in a field that I really find interesting. I am also grateful to my examiner Professor Kari Systä for his guidance throughout this thesis work.

I would like to extend my gratitude to my friends who showed their support during my studies and especially the ones who live in Finland.

Finally, I dedicate this work to my family and especially to my parents and my uncle who have been always there and provided me with love and affection which made my objectives become a reality.

Tampere, 17.12.2017

Houd Bouchelouche

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND SYMBOLS

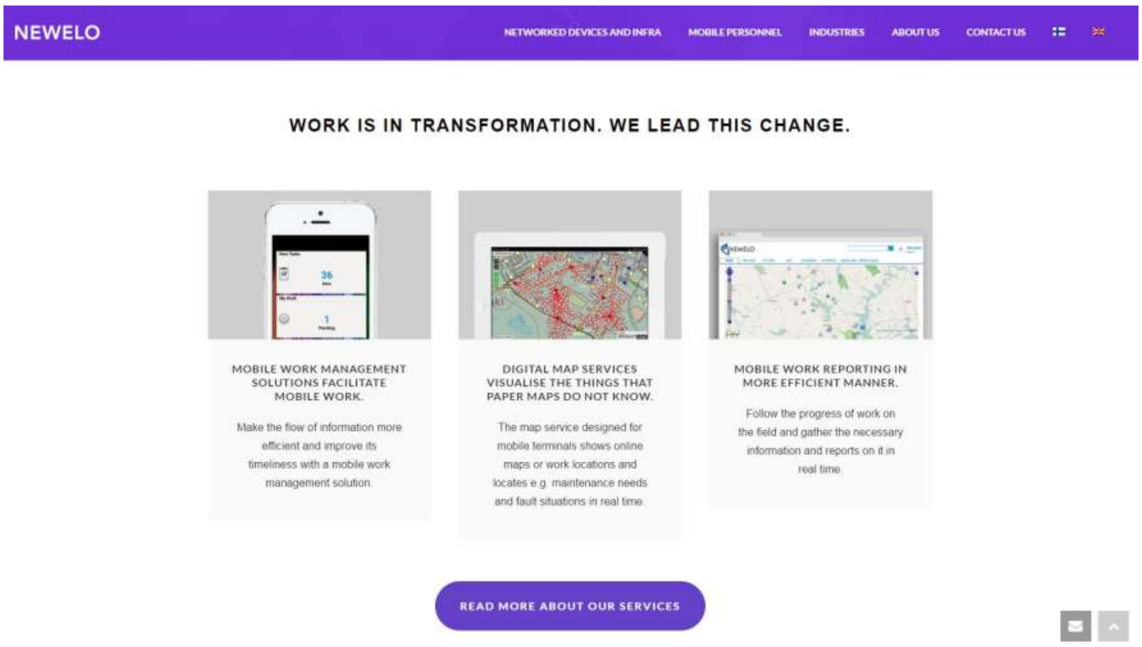
ACID	Atomicity, Consistency, Isolation, Durability
API	Application Programming Interface
BI	Business Intelligence
CDC	Change Data Capture
CRM	Customer Relationship Management
CSV	Comma Seperated Value format
CTO	Chief Technology Officer
DBA	Database Administrator
DBMS	Database Management System
DW	Data Warehouse
EDW	Enterprise Data Warehouse
ELT	Extract-Load-Transform
ETL	Extract-Transform-Load
GPL	General Public License
GPS	Global positioning system
HOLAP	Hybrid OLAP
ICT	Information and Communications Technology
InnoDB	Storage engine for the MySQL relational database management system
IoT	Internet of things
JMS	Java Message Service
KPI	Key Performance Indicator
LDAP	Lightweight Directory Access Protocol
MOLAP	Multidimensional OLAP
MyISAM	Storage engine for the MySQL relational database management system
MySQL	Open-source relational database management system
OLAP	Online Analytical Processing
OLTP	Online Transaction processing
RAM	Random Access Memory
RDBMS	Relational Database Management Systems
REST	Representational State Transfer
ROLAP	Relational OLAP
RTBI	Real-time Business Intelligence
RTDC	External Real-time Data Cache
SaaS	Software as a service

SQL	Standard Query Language
TDWI	The Data Warehousing Institute
UML	Unified Modeling Language
XML	eXtensible Markup Language

1. INTRODUCTION

1.1 Company background

Founded in 2010, Newelo is a Finnish mobile technology company that provides various work management services as SaaS (Software as a Service) for utility companies. The company currently operates in more than a dozen of countries spread throughout different continents with more than a hundred of projects implemented.



WORK IS IN TRANSFORMATION. WE LEAD THIS CHANGE.

MOBILE WORK MANAGEMENT SOLUTIONS FACILITATE MOBILE WORK.

Make the flow of information more efficient and improve its timeliness with a mobile work management solution.

DIGITAL MAP SERVICES VISUALISE THE THINGS THAT PAPER MAPS DO NOT KNOW.

The map service designed for mobile terminals shows online maps or work locations and locates e.g. maintenance needs and fault situations in real time.

MOBILE WORK REPORTING IN MORE EFFICIENT MANNER.

Follow the progress of work on the field and gather the necessary information and reports on it in real time.

Figure 1.1 *Website newelo.com*

The services the company provides make the flow of information between managers and field workers efficient and improve its timeliness with a mobile work management system. Digital map services, for instance, are designed for mobile devices to show information related to work locations, maintenance need and fault situations in real time. Newelo reporting solution is another useful service that comes in a mobile, intelligent and efficient manner. Users can receive all the important reports regarding customers, status and topology directly from their systems to their mobile devices and onto the map, in real time.

Newelo solutions can be easily integrated with companies' existing systems such as other work management systems, customer data systems, warehouse systems, maintenance systems, and device and network information systems, to name a few.

The company's current implementations are related to the energy industry and telecom. However, current services can be easily customized to fit other industries' requirements. Figure 1.2 shows an overview of how the Newelo solution works in the energy industry.

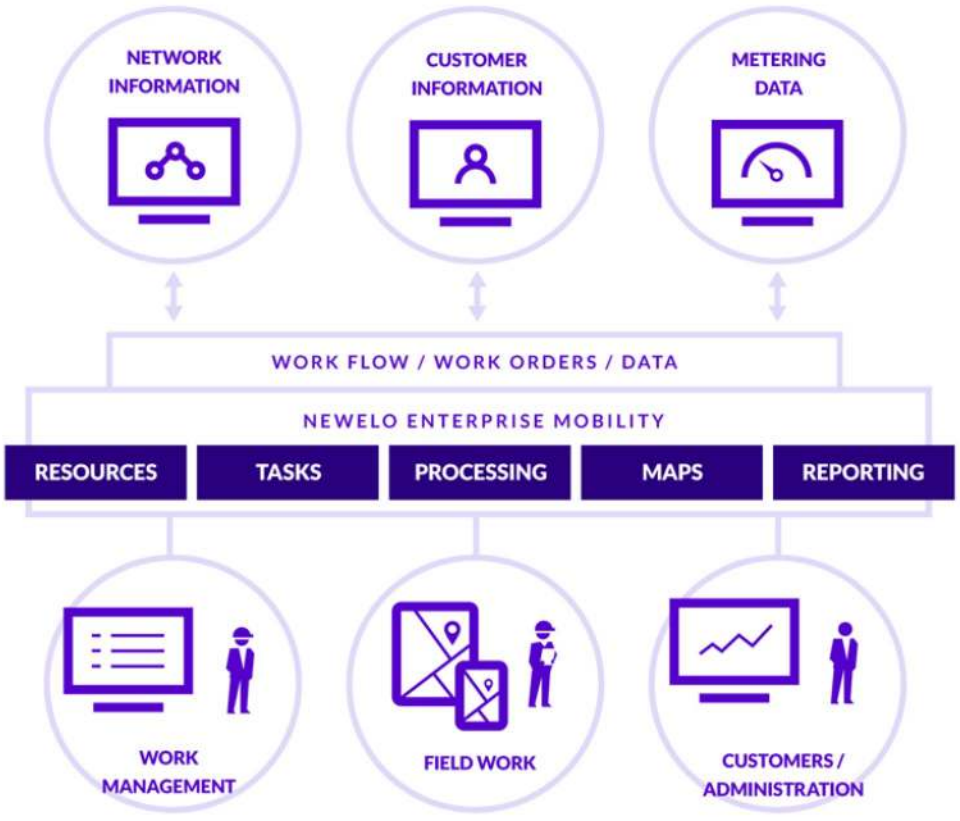


Figure 1.2 *Newelo solution overview.*

Figure 1.2 shows how Newelo solution integrates data from different operational sources such as CRM (Customer Relationship Management), MDM (Meter Data Management), etc. Based on this data, different processes are applied to generate tasks, maps, reports, etc that are delivered to the various users of the system.

1.2 Context of the thesis work

The current plan of Newelo is to expand its services by making use of technologies such as IoT (Internet of Things) and Big Data analytics which are useful for companies specializing in all different sectors. As a first step, Newelo aims to provide a new BI (Business Intelligence) solution that combines data originating from disparate data sources such as work order management systems, smart meter management systems, tracking sensors (GPS), etc.

Even though gathering data is important, processing it and extracting knowledge and information out of it is more crucial. Thus, a real-time BI solution is required in the era of IoT as it can make companies more competitive.

Traditional data warehouses were designed to gather data from operational databases in batches and at a specific period of time, for example, weekly. Nowadays, with the presence of IoT, data can escalate in size dramatically in a very short period of time. In 2013, an article published on Science daily website claimed that 90% of the data that has ever been gathered was collected in the previous two years[2]. In addition to that, BI solutions should be able to process sensor-data in real-time to provide more insights for decision-makers as well as operational managers.

As a result, in order to provide a robust, consistent and long-lasting reporting platform, the company should opt for a new BI solution based on a modern data warehouse.

1.3 Objectives

This thesis work will focus on the design and implementation of a real-time BI solution. The BI will allow end-users to visualize data, view reports and access dashboards for both decision-makers and operational workers. The main objectives of this work are as follows:

- Study the different approaches used for building BI solutions and investigate the various techniques to make them real-time.
- Identify user needs for Newelo's customers.
- Investigate the different tools and technologies that are used in the domain of BI and specify the different implementation alternatives.
- Design and develop a Real-Time BI solution prototype to confirm user needs.

1.4 Methodology

This work will consist of four different phases. The first phase will be devoted to establishing state of the art regarding the different concepts related to BI, with a focus on data warehousing, dimensional data modelling and ETL (extract-transform-load) tools.

The second phase will be about the requirements elicitation. We will focus on interviewing Newelo's staff members and different customers to identify the different needs for the BI solution.

The third phase will consist of studying the different tools that are being used for implementing decision support systems. Comparison between some tools will be elaborated in order to select the ones that best meet the various requirements identified in the previous phase.

In the fourth phase, the design and implementation of the prototype will be realized. For testing purposes, Qlik Sense (a BI dashboard tool) will be used to evaluate the efficiency of the system.

The rest of the thesis is structured as follows:

- Chapter 2 consists of a review of the state of the art about the different notions related to BI and data warehousing.
- Chapter 3 elucidates the different requirements for the real-time BI solution to meet the company's needs.
- Chapter 4 provides an overview of some tools and technologies that are currently used for implementing BI solutions.
- Chapter 5 describes the design and implementation process of the real-time BI solution prototype.
- Chapter 6 discusses the results and the evaluation of the study.
- Chapter 7 provides a conclusion of this thesis work and recommendations to further improve the BI solution for the future.

2. STATE OF THE ART

2.1 Business Intelligence

This chapter is devoted to establishing the state of the art about the different concepts and aspects related to the world of BI. This part is very important and necessary for understanding what methods and techniques are currently being used for designing and implementing such solutions.

2.1.1 Definition

Hancock and Toren [5] define BI as “a set of concepts, methods, and technologies designed to pursue the elusive goal of turning all the widely separated data in an organization into useful information and eventually into knowledge”. Turban, Sharda, Delen and King [12] defined BI in a similar way as “an umbrella term that combines architectures, tools, databases, analytical tools, applications and methodologies”. They stated that such definition is considered as broad and as a result, might lead to various ways of understanding it. Turban et al. [12] also claim that the main objective of BI is to improve the interactivity with business data to get more knowledge for supporting decision-making.

According to [14], “BI is currently the top-most priority of many chief information officers”. This may be due to the benefits of BI which were also mentioned in the same article such as: saving time for data suppliers, becoming aware of what is happening in the business, comprehending the reasons behind certain facts and events, predicting what may happen in the future and all facilitating strategic decision-making. The authors add that the costs related to the hardware and software components of such decision support systems are significant.

Many companies and organizations are evolving in very complex and highly competitive environments. Such circumstances not only make it essential to analyze data to improve enterprise performance but also require that such data analysis needs to be carried out in real-time[1]. Azvine et al. also state that monthly or yearly reports are no longer sufficient for decision-makers and that they demand to have such insightful information delivered to the right people at the right time and in the right place. They also argued that technological advances and modern ICT(Information and Communications Technology) make RTBI (Real-Time Business Intelligence) seemingly achievable[1].

This claim is also supported by Watson and Wixom [14] as they mention that many experienced BI professionals are aware of the demands for fresher data by business management. Furthermore, different information systems and real-time data warehousing are making it possible to provide real-time data to support decision-making[14].

Other articles and papers such as [11] and [15] to reference a few, addressed the concept of RTBI and the possibility to apply it to different businesses. As a result, the need for decision support systems to provide real-time data analysis is quite apparent and requires attention.

According to [1], the definition of RTBI depends on what is meant by the term "real-time" for a given business. They continued by providing some definitions for the term which are:

- the requirement to obtain zero latency within a process,
- that a process provides information whenever it is required by management or other processes,
- the ability to derive key performance measures that relate to the situation at the current point in time and not just to some historical situation.

Based on that, they defined RTBI as an ordinary BI that operates on data which is extracted from operational data sources with zero latency.

Vaisman and Zimányi in their book titled "Data warehouse systems Design and Implementation"[13], mentioned the term "right-time" when they were discussing the concept of real-time data warehouses. They state that depending on the required data latency, the meaning of right-time may vary from right now to several minutes or hours. This definition of right-time supports the sets of definitions mentioned above regarding the term real-time.

2.1.2 BI architecture

In general, a BI solution consists of four different components as shown in figure 2.1. However, it is not always necessary to have them all present in a BI system as it entirely depends on the use case[5].

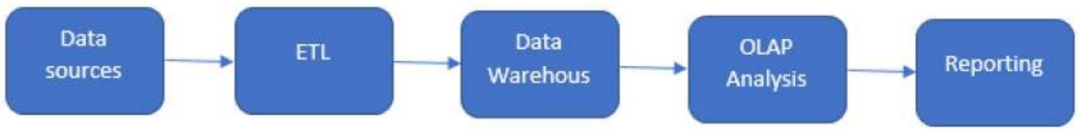


Figure 2.1 *BI solution architecture.*

- **Data transformation (ETL):** This first tier consists of extracting, transforming and loading data from disparate sources to the data warehouse.
- **Data Warehouse (DW):** This is a database containing all the data coming from different sources.
- **Data analysis (OLAP):** This tier provides tools for analyzing data.
- **Reporting:** This tier comes on top. It provides tools such as a dashboard for representing knowledge gathered from the previous tier.

2.2 Data warehouse

2.2.1 Definition

Operational database systems or On-Line Transaction Processing databases (OLTP) are used for handling and storing data originating from different applications that are necessary for the management of daily operations at any enterprise, in real-time. Generally, each department has its own operational database system with well-defined structures and rules which usually differ from one department to another.

Different operational databases in an enterprise usually contain data related to specific domains which are partly useful to decision-makers who are supposed to have a global picture about the enterprise and its operations and make decisions based upon all the different information available in all sources. For example, information coming from customers, Internet, external files, etc. Moreover, data can continuously escalate in size resulting in a huge number of datasets, making it harder to access data quickly and coherently.

The following table summarizes the major differences between operational systems and decision support systems in terms of the data structure as well as the use cases.

Difference	**Operational systems**	**Decision support systems**
Data	Application oriented	Subject-oriented
	Instantaneous	Historical
	Detailed, normalized data	Aggregated, coherent and often redundant
	Data updated frequently	Stable and time variant data
Use	Ensures daily activities	Data analysis
	For operational staff	For decision makers
	Update and simple queries	Read-only and complex queries

Table 2.1 *Operational systems vs decision support systems*

BI solutions provide a set of tools such as data warehousing to tackle those difficulties. Data warehousing has a key role in the BI world as it contains information regarding all the different activities of a given company.

Bill Inmon defines the concept of data warehouse in his book titled "Building the Data Warehouse" which is considered as the bible of data warehousing, [6] as follows:

"A data warehouse is a subject-oriented, integrated, nonvolatile, and time-variant collection of data in support of management's decisions."

In his definition, Bill Inmon deployed a set of characteristics to describe the features of a data warehouse which he explained as [6]:

- **Subject-oriented:** Unlike the transactional approach which is adapted for operational applications such as bank account management, CRM, etc, a data warehouse is designed to handle data related to specific subjects. The relationships between such data are stored and organized in a different way from transactional data.
- **Integrated:** The data warehouse integrates data from multiple disparate sources. Such data need to be properly manipulated to avoid any incoherence.

- **Nonvolatile:** In operational systems data is regularly accessed, updated and deleted. On the other hand, data in a data warehouse is not supposed to be regularly updated. However, when updates are required, data history will be kept.
- **Time-variant:** This characteristic implies that every data entry is accurate for a given moment in time. Hence, every value is time-stamped which makes tracking changes and history easily achievable in data warehouses.
- **Support of management's decision:** data in a data warehouse is organized in a way that allows the execution of analytical processes such as reporting, data mining, etc to support decision-making.

Another concept related to data warehousing is data mart. Data marts are miniature versions of the data warehouse. They are built around the information needs for a specific department inside the enterprise[6].

2.2.2 Data warehouse maturity level

Depending on the maturity level of a data warehouse solution, an organization can respond to more complex needs. Table 2.2 shows the different use cases of a BI solution according to its maturity level based on the TDWI (The Data Warehouse Institute) data warehouse maturity model[4].

TDWI has developed this data warehouse maturity model to make it easy for companies to gauge their current EDW initiative, and where it needs to go next[4].

- **Prenatal stage:** provides a standard set of static reports and is hand-coded which makes it slow to customize reports. Data warehouses are not used at this stage.
- **Infant stage:** spreadmarts are used at this stage. They are low-cost; however, they do not provide a consistent picture of the whole enterprise.
- **Child stage:** at this stage, data marts are used and they focus on a single application area. OLAP or ad hoc query or parametrized reports can be applied; however, it is not possible to have cross-departmental analysis.
- **Teenager stage:** central data warehouses are used at this stage. Standardized definitions, rules and dimensions are applied to allow cross-functional analysis. The BI at this stage provides interactive reporting tools and dashboards.

Stage	**Prenatal**	**Infant**	**Child**	**Teenager**	**Adult**	**Sage**
Executive perception	Cost centred	Inform executives	Empower knowledge workers	Monitor business processes	Drive the business	Drive the market
Structure	Management reports	Spread marts	Data marts	Data Warehouse	Enterprise DW	BI services
Scope	System	Individual	Department	Division	Enterprise	Enterprise
Business value						

Table *2.2 DW maturity level and business value [4]*

- **Adult stage:** Enterprise DWs are used at this stage; they provide a single version of the truth.
- **Sage stage:** at this stage, the enterprise DW can provide BI services such as decision engines embedded in internal and external applications.

According to TDWI, organizations go through these six stages at different rates; however, many DW/BI initiatives struggle at two pivotal points during the evolution of the data warehouse, known as the "Gulf" and the "Chasm". They argued that for organizations to cross the first point (Gulf) which is easier compared to the "Chasm", must not consider the DW/BI as just a management reporting system. While to cross the "Chasm", executives must consider the DW/BI solution as a mission-critical enterprise resource[4].

2.2.3 Data warehousing in the era of big data

According to [3], many people consider that the concept of data warehousing in the era of big data is irrelevant. However, a research paper about augmenting data warehouses with big data illustrated how big data fits with the existing paradigm of feeding the data warehouse using ETL from different data sources[3].

Some research papers discuss the possibility of modernizing data warehousing with big data technology. For instance, Marker Kromer, a big data analyst product manager for Pentaho, states that the use of a mixture of technologies such as relational databases, analytical databases and Hadoop will improve BI solutions [9].

He also adds that the loads of features that have been integrated into enterprise data warehousing over decades, make it difficult for such new technologies to completely replace relational database systems. Although more complex data analytics functions were added to big data platform, many fundamental concepts in RDBMS (Relational Database Management Systems) such as ACID (Atomicity, Consistency, Isolation, Durability) property are still missing[9].

The idea of a hybrid architecture is to benefit from the data processing capabilities of big data technologies and combine them with well-advanced data warehousing techniques that have been developed for a quite some time.

2.3 Data modelling for data warehouse

2.3.1 Dimensional modelling concepts

The concept behind a data warehouse is to store historical data related to certain activities of the enterprise so that it can be analyzed later. DWs are based on the multi-dimensional model, where data is viewed in an n-dimensional space, known as a data cube that is defined by a set of dimensions and facts. The cells of the cube represent the facts that contain different measures, and the dimensions of the cube represent different business perspectives[13].

Dimensions are the different perspectives used to analyze data, and they can have attributes associated with them to describe them. Facts also have measures associated with them which are used for the quantitative evaluation of the various aspects of the analysis. For instance, sale history of different products makes it possible to study the sale evolution of a specific product or all products in general, over time. Thus, decision-makers should be able to access this data in order to come up with new strategies and make better decisions related to that given activity. Figure 2.2 shows a 3-dimensional data cube for the sales of different products; the three different dimensions are “product”, “location” and “time”.

Data cubes can be divided into two categories; “sparse” or “dense”. A data cube is dense if all combinations of the different dimension values have measures associated with them. Otherwise, the data cube is sparse. In real-world applications, cubes are typically sparse.

To have efficient analysis, it is important to be able to view the cube from different perspectives and at several levels of details. Hierarchies allow viewing of the data

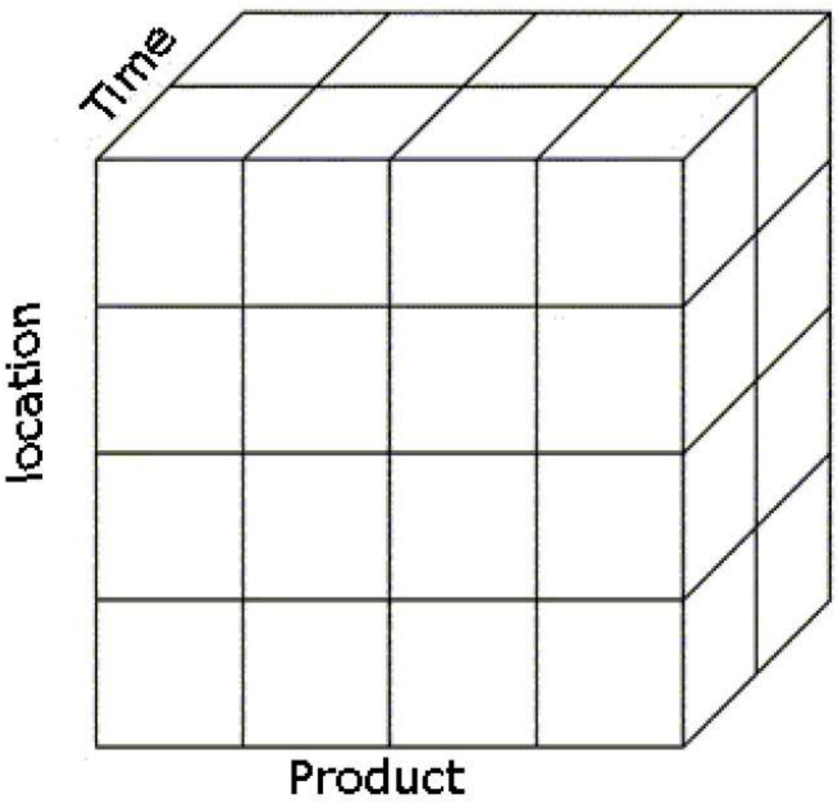


Figure 2.2 *Data cube representing product sales per location and time.*

cube at different levels of granularity by applying different aggregate functions to go from lower-level detailed concepts to higher-level ones.

A dimension schema is used for defining the hierarchical structure of a data cube; it shows all the different levels that are available in a given dimension and gives more details about its structure. The dimension instance, on the other hand, comprises all the members of all the different levels of a dimension. It is very common to name the top level of a hierarchy by “ALL”.

Measures in a cube are associated with aggregation functions that combine several values of a specific measure into a single one. Aggregations are performed when end-users change the level of details at a which a cube is viewed. This is achieved by traversing the hierarchies of certain dimensions, for example by using operations like addition[13].

according to [13], “Summarizability refers to the correct aggregation of cube measures along dimension hierarchies in order to obtain consistent aggregation results”. To ensure summarizability, a set of conditions must hold according to the same reference:

- **Disjointness of instances:** the grouping of instances in a level with respect to their parent in the next level must result in a disjoint subset.
- **Completeness:** all instances must be included in the hierarchy and must be related to one parent.
- **Correctness:** refers to the correct use of the aggregation functions.

DWs must allow decision-makers to easily, interactively and coherently run queries related to their specific area of work. Therefore, they must be able to respond to queries quickly and guarantee the integrity and the accessibility of data by the right personnel.

The design and implementation of an Enterprise Data Warehouse require a particular procedure which is different from the one used for relational database systems. Processes for extracting data from a variety of sources (operational database, files, big data platforms) and integrating it into the data warehouse are very crucial. In most cases, it is sufficient to integrate the data as an entire set or batches. The extraction and integration can be scheduled daily, weekly or according to the need. With the apparition of many new technologies, this approach can be inefficient, and real-time integration would be required.

In the current literatures, two approaches exist for building a data warehouse[13]:

- **Bottom-up:** In this approach, small data marts are implemented according to the need. After that, those data marts are merged together to get a data warehouse. This approach is suitable for companies who want fast results and less risks as DW implementation usually takes time.
- **Top-Down:** The DW is built right from the beginning. Data marts are obtained from it later. Sometimes, in this approach data marts are just logical views.

2.3.2 Concept of OLAP (Online Analytical Processing)

Online Transaction Processing (OLTP) is an approach for designing information systems that are transaction-oriented. It corresponds to transactional databases with a relational schema that can support a significant amount of simple transactions (create, delete, update, etc.). The efficiency of such systems can be measured in terms of transactions per second. Such systems were proven to be inefficient to meet the new requirements of decision support systems. Thus, a new paradigm was introduced, oriented for data analysis called OLAP (Online Analytical processing)[13].

The number of transactions in an Online Analytical Processing (OLAP) is relatively low compared to those in OLTP. However, queries are usually complex and involve aggregate functions. The efficiency can be measured in terms of response time and, the data is stored in a multi-dimensional schema.

See table 2.3 for comparison between OLTP and OLAP.

Characteristics	**OLTP**	**OLAP**
Operations	Updating	Analysis
Data	Recent	Old
Access type	Read/Write	Read
Orientation	Line	Multi-dimension
Size	Small	Large

Table 2.3 *OLAP vs OLTP*

OLAP servers architectures

OLAP systems are based on the multi-dimensional model. However, they can be implemented using relational databases that support SQL extension for special access methods. There are several approaches [13] for implementing a multi-dimensional model which are:

- ROLAP (Relational OLAP) stores data into relational tables where relational joints are used. Queries are somehow complex and might take some time to be executed. The results of the different queries are never stored and therefore the same query will be executed multiple times. As mentioned before, the main drawback is the response time. However, the cost is relatively low as this approach makes use of existing resources such as hardware resources, licenses, etc.
- MOLAP (Multi-dimensional OLAP) stores data in a cube, which is, in fact, a multi-dimensional database. With this approach, relational databases are no longer used. In MOLAP, information is processed before indexing it into

the dimensional database and data retrieval can be achieved instantly and without any delay. The main drawback is the costs which is significant as it is required to purchase licenses for dimensional databases in addition to the cost of designing the different cubes. The advantage is a very short response time.

- HOLAP (Hybrid OLAP) is the combination of ROLAP and MOLAP. Part of the data is stored in MOLAP cubes and another part in relational tables depending on the preferred type of processing. For heavy data processing, the data is more efficiently stored in relational tables, whereas multi-dimensional databases are used for speculative processing.

OLAP operations

Once the multi-dimensional cube is created on the OLAP server, different operations are possible to be performed on it. Such operations allow the exploration of data inside the cube at different levels of details. Few operations are mentioned next:

- **Slice & Dice:** "Slicing" and "Dicing" are two techniques offering the possibility to select and filter data subsets according to different criteria of each dimension. Slicing consist of selecting one slice from the data cube which filters a dimension according to one value. Dicing, on the other hand, can be viewed as an operation of extracting a sub-cube that filters data according to a set of conditions related to the different dimensions.
- **Drill-down & Roll-up:** These two methods consist of representing the data cube at different granularity level, lower level in the case of "Drill-down" and higher one for "Roll-up". They are basically used to control the level of details of the cube's data.

2.4 Data warehouse design

There is still no consensus on the phases that should be followed for the design and implementation of a data warehouse project. Most of the books related to data warehouse design follow a bottom-up approach based on the relational schema[13].

The design approach applied in this thesis is mainly taken from [13]. The approach is similar to the one used in database design and it consists of four different phases. See Figure 2.3.

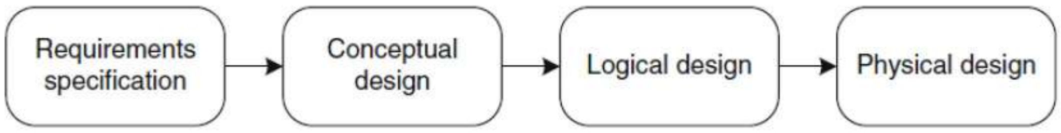


Figure 2.3 *Phases in data warehouse design.*

Three approaches can be followed for conducting the requirements elicitation phase:

- **Analysis-driven approach:** Needs and requirements are identified from the different key users of the system.
- **Source-driven approach:** this is done by analyzing the data sources. Key users are required later to confirm the correctness of the data structure.
- **Analysis/Source-driven approach:** where the combination of the two previous approaches is used.

2.4.1 Conceptual data warehouse design

Conventional databases are generally designed at the conceptual level using conceptual models such as entity relationship or object-oriented models using UML. These models have been proven by the database community to be effective in improving communication between designers and users and are more stable than logical schemas that are dependent on the technologies used for the deployment. On the other hand, there is no well-established conceptual model for the multi-dimensional data structure. As a result, Data Warehouses are designed at the logical level based on some relational schema such as star-schema, snowflake-schema, etc. These schemas are difficult to be understood by typical users[13].

The multi-dimensional model is used for designing conceptual models for data warehouses. This model is able to represent all conceptual elements required in data warehousing and OLAP systems.

The main components of the model are:

- **Schema:** is the set of all the dimensions and facts.
- **Dimension:** is a structure representing a specific business perspective used to answer business questions.

- **Fact:** relates several levels and dimensions. The same dimension can be assigned to a specific fact several times with different roles.
- **Hierarchies:** a key element in dimensional modelling as they are used to represent data at different abstraction levels according to the selected dimension.

2.4.2 Logical data warehouse design

Several approaches exist for implementing a multi-dimensional model. The selection of the right approach depends on the technique used for storing the data cube logically. In this section, we will give more details about the ROLAP approach.

ROLAP stores data in RDBMS (Relational Database Management System) with an extension of SQL (Standard Query Language) in order to make it able to support analytical operations. The advantage of using such systems is their strength in storing a large amount of data.

Different schemas can be used to design a data warehouse. The two most famous approaches are:

- **Star-schema:** This schema consists of one central fact table connected to the different dimensions surrounding it. The dimensions contain redundant data and are not normalized.
- **Snowflake-schema:** tries to avoid the redundancy of star-schema by normalizing dimension tables which are represented by several tables.

Other approaches exist such as:

- **Starflake-schema:** which is the combination of the two previously mentioned approaches. Some dimension tables are normalized and others are not.
- **Constellation-schema:** in this approach, multiple fact tables share same dimension tables.

The relational model is not well adapted for decision support systems as it contains normalized data to avoid redundancy, improve query performance and guarantee data coherence. The goal of a data warehouse in the first place is to offer an understandable data structure for end-users and enhance the performance of decisional queries. Dimensional modelling relies on the concepts of fact and dimension. In this study, we will focus on the most common model, which is star-schema.

The principle of star-schema is to describe the object to be analyzed through a fact. A fact is defined by a name and a set of attributes and foreign keys to other dimensional tables where descriptive information is kept. See figure 2.4 for a star-schema example.

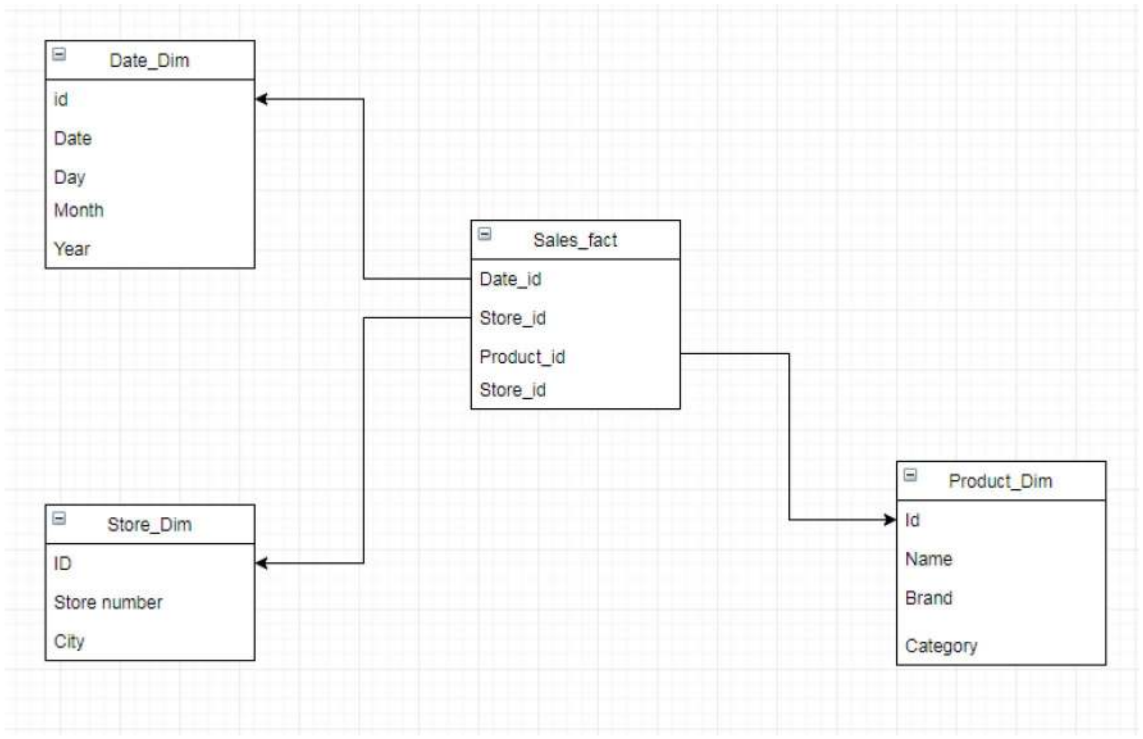


Figure *2.4 Example of a star-schema*

Dimensions may consist of a large set of attributes for describing the fact data. Some of the common dimensions are time, location data, employee data, etc. Every element of a dimensional table may be associated with zero, one or multiple facts, whereas, a fact is associated with only one dimensional-entry.

On the other hand, the snowflake schema differs from star-schema on the dimension side. Dimension tables in star schema are not normalized, i.e. each dimension is represented by one table, whereas in the snowflake schema, all tables are normalized. This schema decrease in normalization may cause performance issues as well as make queries more complex for end-users. However, the snowflake schema is recommended in certain cases. For instance, in a case of a large dimensional table with a sub-dimension containing many attributes. To illustrate, let's take a case of a customer dimension with a sub-dimension dealing with demographic data for each customer.

It is recommended to store each combination of the different attributes in the sub-dimensions and reference those combinations with a foreign key.

When loading data into the data warehouse, it is important that dimensional tables are already populated even before loading data into fact tables. Dimensional tables contain all the primary keys that would be referenced by fact tables. It is necessary to match facts with dimensions during the process of loading fact data. Joints or other equivalent ways proposed by some ETL tools can be used for this purpose.

Primary keys used in operational systems might be reused after few years, whereas, data warehouses store information for a longer period thus being independent from operational systems is very important. Therefore, primary keys used in OLAP must be different from the ones used in operational systems and it is necessary to use substitution keys known as surrogate keys.

2.5 Extract-Transform-Load layer

The objective of ETL phase is to ensure the continuous loading of data from the different data sources to the data warehouse. This is achieved by extracting the data from sources, applying some rules set by the enterprise and then loading the data at the end. This process constitutes the hidden part of the iceberg for a data warehouse project. It occupies around 80% of the effort estimated for the implementation [7]. Preprocessing of data is a very critical process. It is recommended to apply this process in a preparation zone separated from the data warehouse.

Many techniques have been used for collecting data from operational sources to load it into the data warehouse. The major difference between these techniques is the data latency[8]. For instance, some techniques use daily-batch processes while others require real-time extraction of data. In addition to that, capturing data can be done using two methods; the first method uses incremental queries which filter data according to their timestamp or flag, while the second one makes use of a Change Data Capture(CDC) mechanism which detects all the changes as they happen. It is also possible to distinguish all of these methods by the type of operations they perform for collecting data. Two types of operations exist known as "pull" and "push" operation[8]. Pull operation looks for new data at fixed intervals, push operation loads data when only a change has occurred.

ETL(Extract, Transforming and Loading) is a tool used for integrating data. It is a middleware layer that aims to synchronize massive amount of data originating from different sources.

	Batch	Mini-Batch	Micro-Batch	Real-Time
Description	Data is loaded in full or incrementally using a off-peak window.	Data is loaded incrementally using intra-day loads.	Source changes are captured and accumulated to be loaded in intervals.	Source changes are captured and immediately applied to the DW.
Latency	Daily or higher	Hourly or higher	15min & higher	sub-second
Capture	Filter Query	Filter Query	CDC (Change Data Capture)	CDC
Initialization	Pull	Pull	Push, then Pull	Push
Target Load	High Impact	Low Impact, load frequency is tuneable		
Source Load	High Impact	Queries at peak times necessary	Some to none, depending on CDC technique	

Table 2.4 *Different ETL frequency processes[8]*

2.5.1 Extract process

Extracting data is the first step in an ETL process. It aims to fetch and read data from different operational databases or other files. It uses different connectors to interact with various sources.

Different architectures are used for collecting data from different operational sources and loading it into the target data warehouse. The extraction of data can be executed with incremental queries that filter data according to its timestamp, flag or the use of a mechanism that helps detect occurring changes.

Imported data must be selected carefully without worrying much about exhausting the system and importing less information that cannot reflect reality nor answer questions made by decision-makers. It is necessary to extract only useful data and study the possibility of collecting data from sources using an incremental approach.

The extraction process may be very critical in some cases where one or more data sources must operate seven days a week and 24 hours a day. The process is heavy and thus should be executed in a manner that does not cause any interruptions or failures to operational systems. Typically, the process is executed at a given time intervals where there is less load on the operational systems.

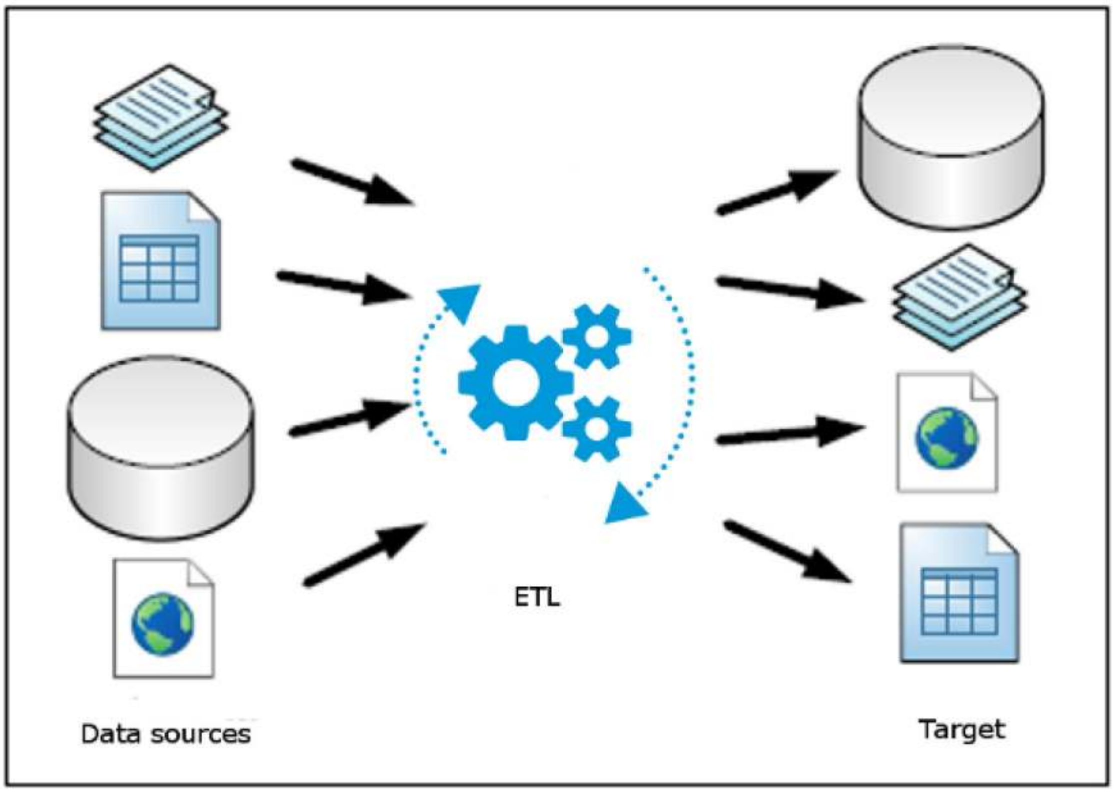


***Figure* 2.5** *ETL process*

2.5.2 Transform process

At this stage, certain rules are applied to the extracted data. The aim is to load a cleansed, filtered and well-structured data into the DW and this is due to the fact that data collected from different sources do not conform to the same rules.

For example, if we have two data sources "A" and "B". In "A" we have dates stored in "dd/mm/yyyy" format wherein source "B" dates are written in "mm/dd/yyyy". It is during this stage that we convert the dates to a specified format. Other transformations are also applied such as:

- **Cleaning:** for example, transforming Female into "F" and vice versa.
- **Filtering:** Selecting only specific columns.
- **Splitting:** Splitting one column into multiple ones.
- **Joining:** Merging data that originate from different sources.

There are cases where transformation is not required. In this case, data is called "rich data", "direct move" or "pass through" data.

Here are some examples of the transform processes that can take place during this phase:

- Substituting primary keys with new ones that have numerical values. This will improve the performance and the independence of the data warehouse.
- Another example, find a textual equivalent that matches names or addresses to certain values that are already saved in the system but may be spelt differently in the data source. In this case, differences must be detected and altered in order to achieve the matching.

At the end, dimension and fact tables management also take part in this phase, as well as some calculations.

2.5.3 Load process

This is the last phase of the whole ETL process. At this stage, the extracted and transformed data will be loaded into the destination database. To optimize the process, it is recommended to disable all data constraints as well as index the database.

The three phases of ETL (Extract-Transform-Load) can be executed simultaneously. The extraction of data takes some time and so the second phase is executed in parallel and prepares the data for the third phase. When some of the data is ready, they will be directly loaded into the destination database without waiting for the previous phases to end.

2.6 Real-time Data warehousing

ETL processes are the most challenging part when implementing any data warehouse, and it gets more complicated when it comes to processing real-time data [10]. One of the simple solutions for real-time data warehousing is to increase the refresh frequency for the ETL process which is known as "near real-time ETL". However, ETL processes involve downtime of the data warehouse, i.e. no user is able to access data when the process is taking place, and as a result, the heaviest periods of incoming data might occur at the peak times of the data warehouse; thus, other techniques exist for tackling this issue.

Another solution for real-time data warehouse applications consists of continuously feeding the data warehouse with new data from the source systems. This technique is called “Direct trickle feed”. The problem with this technique is that it does not scale very well and it leads to the same problem why opting for a data warehouse in the first place, as continuous updates and inserts do not mix well with complex queries [10].

Defining real-time partitions is another classic solution for reducing data latency[13] known as “Trickle & Flip”. In this approach, real-time data is stored in separate partitions that have the same structure as of the fact tables. These partitions are subject to special rules for updating and querying, and their data will be swapped to fact tables on a periodic basis. These partitions must[13]:

- Have same granularity as the fact table.
- Contain all data that was created after the last refresh cycle.
- Support high-performance queries.
- Be lightly indexed.

One last solution is known as External Real-time Data Cache(RTDC) which consists of using an external database server that takes care of managing real-time data to avoid any potential performance problems. By using RTDC, most of the scalability and performance issues can be avoided and queries requiring real-time data will be executed very fast as they run on their own dedicated system[10].

2.7 BI reporting & data visualization

BI reporting is the process of presenting information stored in a data warehouse to end-users. It is the top layer on the whole BI solution and is responsible for delivering summarized and structured reports that aim to help different users understand the significance of data by placing it in a visual context such as charts or dashboards.

There are a variety of tools and techniques that can be used for data visualization and BI reporting, namely:

- **BI dashboards:** visual displays that consist of multiple reports or graphs, that usually fit in one computer screen to make it easy to get an overall picture of various aspects of a given business at a single glimpse. They are used by different levels of management and their data might be coming from different sources.

- **Ad-hoc analysis:** a process of performing a query that aims to answer one question related to a specific need. The result can be in form of an analytic report, data summary or some other forms of data visualization.
- **Performance scorecards:** graphical representations used for evaluating some entities such as an enterprise, a business unit, against some specific goals. They use KPIs (Key Performance Indicators), which are metrics used for evaluating the progress of a business or an organization.

Reporting tools are very crucial as they are the interface connecting end-users to all the data available in a data warehouse. They allow users to:

- Select data related to a specific sector, period, geographical area, etc.
- Sort, group or divide data according to criteria of their choices.
- Perform different calculations and aggregations as well as performing comparisons according to specific dimensions.
- Present results in a detailed or a summarized manner using different graphical representations according to their needs or the decision-makers expectations.

Many BI reporting tools are available in the market today. Some of them are commercial while others are open source developed by a variety of communities. These tools can be easily connected to different data warehouses making it efficient to analyze their data. More details regarding specific tools will be provided in the next chapters.

3. REQUIREMENTS SPECIFICATION

In this chapter, system requirements of the real-time data warehouse will be discussed in details. At the end of the analysis, a specification will be established.

3.1 Work order management system

Work order management systems enable utility companies to keep track of customers and work information. Such systems help companies to overcome the different limitations of paper-based work orders and thus, improve productivity and revenue. In addition to that, they help in preventing revenue loss that is due to lost or incomplete work orders.

Newelo's work order management system comes with a reporting solution that allows managers and users to receive all the important information regarding customers, work status, etc. However, Newelo's current service is not based on a dedicated BI solution and docs usc any data warchousc systcms that arc capablc of providing more advanced analysis. An analysis that would make it possible to respond to more complex queries, queries that are costly to be run on an operational database system. Therefore, the need to improve the reporting service by implementing a data warehouse or a whole BI solution is self-evident.

3.2 Requirements elicitation

Every DW solution must be capable of responding to the different user needs for a given company. This cannot be achieved without conducting a thorough study to gather those needs and requirements. Three approaches for data warehouse requirements elicitation have been described in the first part of this thesis. The approaches are "Analysis-driven" and "source-driven" approaches as well as the third one which is mixed approach of the two, known as "analysis/source-driven" approach.

Adopting one of the two approaches exclusively limits the final results and many requirements may be ignored or unnoticed. From a practical point of view, it is much easier to apply the mixed approach to reveal information and gather more requirements from both approaches and make use of all the possibilities and opportunities from both users and data sources.

During the requirements gathering phase, it was spontaneous not to limit ourselves to only gather information from customers, but also involve other personnel from Newelo such as CTO (Chief Technology Officer) and database administrators. According to [7], DBAs (Database Administrators) are the main experts of the existing applications and interviewing them will serve to gather facts about topics that will arise during the interviews with end-users.

In order to accomplish a thorough study and identify as many user needs as possible, the whole procedure was divided into a subset of stages which can be summarized as follow:

- Preliminary study of source systems, data sources and conducting interviews with the CTO and the DBA,
- Identification of users for the interviews,
- Planning, preparing and conducting the interviews,
- Use of other means to identify requirements,
- Elaboration and validation of the identified requirements.

1. **Preliminary study of source systems, data sources and conducting interviews with the CTO and the DBA**

   This first phase consisted of a deep study of the operational systems, their environment and how they are used by different users. Understanding the operational systems as a whole facilitates the comprehension of how different customers are interacting with Newelo services and serves as a starting point for brainstorming about the analytical aspect of such systems.

   Secondly, databases' schemas for the different operational systems were analyzed in detail to initially identify what kind of information could be extracted and which one may be useful for decision-makers. This phase was based on interviewing the CTO of the company and the DBA as well as analyzing the structure of the different DBMS.

2. **Identification of users for the interviews**

   Given the large number of the customers Newelo has all over the world, it was clearly impossible to meet and interview all of them. Therefore, this study focused on customers from Finland working in the electricity industry only. It was very crucial to identify Newelo's customers whose use cases can represent the majority of other potential customers for the BI solution.

   In compliance with the company's instruction, information about the different customers is regarded as confidential; therefore, any details are not discussed in this thesis.

3. **Planning, preparing and conducting the interviews**

   Before diving into details, it is important to understand why interviews were selected as a technique for gathering user needs even though many other techniques exist such as "Requirement gathering workshop" or "Observations". Interviews have the advantage of being more or less easy to plan compared to the other techniques as they do not take very long periods to be conducted.

   The planning phase usually takes place few days before the interviews. It consists of setting a date for the interviews with the identified customers. Afterwards, the preparation phase consists of identifying the questions for the interviewee and selecting the different dashboard prototypes that were developed from the preliminary study of data sources.

   On the day of the interview, the prototype is presented to the interviewee alongside the different use case scenarios behind each visualization. The interviewee gets the chance to ask questions and give feedback about the prototype. The attendees from Newelo side will ask follow-up questions as well as the questions that were elaborated during the preparation phase. Once the interview is finished, a short meeting with the attendees from newelo is held to further discuss the different issues risen during the interview.

4. **Use of other means to identify requirements**

   Studying and analyzing the different reports (existing as well as the requested ones by different customers) is a good source for gathering user needs. It reveals more information and requirements that the interviewees did not mention due to the influence of the questions or other reasons such as thinking that some information is so evident and is not worth mentioning.

   Analyzing data sources and studying the operational systems were performed before the interviews in order to design prototypes for the attendees and get the chance to confirm the findings of the previous phase with them.

The requirements elicitation phase allowed us to better understand customer needs and their business processes. In compliance with the company's instruction, the results of this phase is regarded as confidential; therefore, only two business domains are discussed in this thesis which are:

- Work orders for fixing smart meter communication faults.
- Smart meters' consumption data that requires real-time processing.

3.2.1 Work orders for fixing smart meter communication faults

Smart meter communication faults occur when a smart meter fails to communicate with the MDM (Meter Data Management). The MDM notifies the WOMS (Work Order Management System) about the different devices that are experiencing the fault, and work orders will be generated accordingly by the latter system.

Four main fixes be performed to resolve a meter communication fault:

1. **Meter change:** the problem is caused by the smart meter. It is required to change the whole meter to fix the fault.
2. **Communication module change:** consists of changing the communication module that causes the fault.
3. **Antenna change:** the antenna is broken and needs to be replaced.
4. **Antenna transfer:** consists of changing the position of the antenna.

The requirement elicitation allowed us to identify the need for analyzing the work orders for fixing meter communication faults. The analysis will provide the following benefits to managers:

- Understanding the correlation between the faults and the causes (a broken meter, a broken communication module, etc).
- Understanding the correlation between the faults, causes, electricity meter types and communication modules.
- Finding any connections between the faults, locations of the electricity meters and the different periods of time.

3.2.2 Smart meters' consumption data

Electricity consumption data can provide a useful preventive analytics for operation managers as it helps in detecting electricity consumption overloads. The analytics should be performed with data that originates from different consumption points in real-time. In practice, the MDM gets consumption data periodically every 15 minutes making the implementation of such a real-time BI solution easier compared to other systems where data is created at random intervals. In addition to that, the consumption data stored in the MDM is never updated, and only new rows are inserted. The objectives of this analysis are to:

- Analyze the different consumption points by time, location and client type (individual or company).
- Understand the behaviour of electricity consumption according to various natural events such as cold weather or hot weather.
- Provide preventive analytics for detecting electricity overloads for each electricity transformer.

3.3 Scope of the project

The initial need is to have a real-time data warehouse solution capable of extracting data from diverse sources that provide historical or real-time data, allowing different users to access and analyze data in a more efficient way through a variety of reporting tools on different platforms such as desktop or mobile applications, web browsers. The solution will be extended iteratively according to the various user needs by extracting more data from their diverse data sources.

The BI solution will be composed of three main components: ETL process, Data warehouse and data visualization layer. As for this thesis, the work will be all about these three main components. For data visualization, Qlik Sense will be used to test and demonstrate how the different OLAP queries will respond to the user needs.

As for the queries, it will only be related to the Work order management system as well as the meter data management system. The latter system will be simulated using a script to send electricity consumption data to the BI solution.

3.4 System requirements

The following list will offer a brief outline and a description of the primary requirements of Newelo BI solution. The requirements will be split into two categories: Data warehouse requirements and ETL requirements.

3.4.1 Data warehouse requirements

The data warehouse for the solution must be able to respond to all the different needs that were identified during the requirement elicitation phase. As a result, the DW should provide real-time data analysis for electricity consumption data. All other business domains do not require such a constraint, and their data need to be updated weekly at maximum.

For the prototype sake, the "Trickle & Flip" approach was judged to satisfy the initial requirements. Therefore, real-time partitions will be developed for the analysis subject requiring the real-time feature.

Constellation-schema will be selected for the logical design of the prototype. The idea behind constellation-schema is to have multiple fact tables sharing same dimension tables to avoid data duplication. The complexity of the final constellation-schema may increase which will make it more difficult for designers to analyze and understand the schema. However, this issue is not very critical at this stage of the development, and the schema developed for the prototype can be easily converted to star-schema for production.

3.4.2 ETL requirement

As discussed in the previous sections, ETL development takes a big part in the success of data warehouse projects as well as the development time. Therefore, the data integration solution to be selected for a given project will have a significant influence in the evolution of this latter, and if chosen correctly, will facilitate the development and maintenance of such ETL processes.

A tool that provides a graphical user interface for designing ETL jobs can be useful and helps in decreasing the development and maintenance time. It is essential that the selected tool must be compatible with the disparate data sources and able to scale up with the company's needs for an extended period. The cost of acquiring the tool must also be low especially at the beginning of the project. Therefore, an open-source solution with a very active community is highly preferred. The open-source product should be based on a robust development and updated regularly, and the size of the community must not be neglected as it reflects the well being of the developed solution.

Finally, it will be useful if the ETL tool offers some advanced CDC (Change Data Capture) mechanisms that are adapted for real-time data integration. Nonetheless, such CDC mechanisms are not very critical at this stage because the current need for real-time support consists of dealing with electricity consumption data only, which is a simple use case as described earlier. The current real-time feature requires that the ETL process retrieves consumption data from the MDM every 15 minutes(when it becomes available in the MDM). As a result, the use of near real-time ETL is considered to be the simplest solution as it consists of only increasing the ETL frequency.

4. IMPLEMENTATION ALTERNATIVES

In this part, we will discuss the different tools and technologies that can be used for implementing the different component of the BI solution. The tools must meet the requirements specified in the previous section.

4.1 ETL solution

Although there are a lot of commercial ETL solutions nowadays, only the alternatives are considered in this study to avoid all of the extra expenses that would incur due to licenses and training.

Two of the most well-known open source solutions have been studied and tested which are a real alternative to the commercial ones:

- Pentaho Data Integration
- Talend Open studio

4.1.1 Pentaho Data Integration

"Pentaho Data Integration" previously known as "Kettle" is now part of the BI solution "Pentaho Business Analytics" developed by Pentaho Organization. The tool can be used independently from the BI suite and is of type ETL engine. The product is available in both commercial and community version.

The tool is composed of three modules:

- The first one is a graphical interface called "Spoon" used to develop transformation jobs.
- The second module, "Pan" is used to launch the execution of tasks.
- The third one called "Kitchen" used for executing tasks.

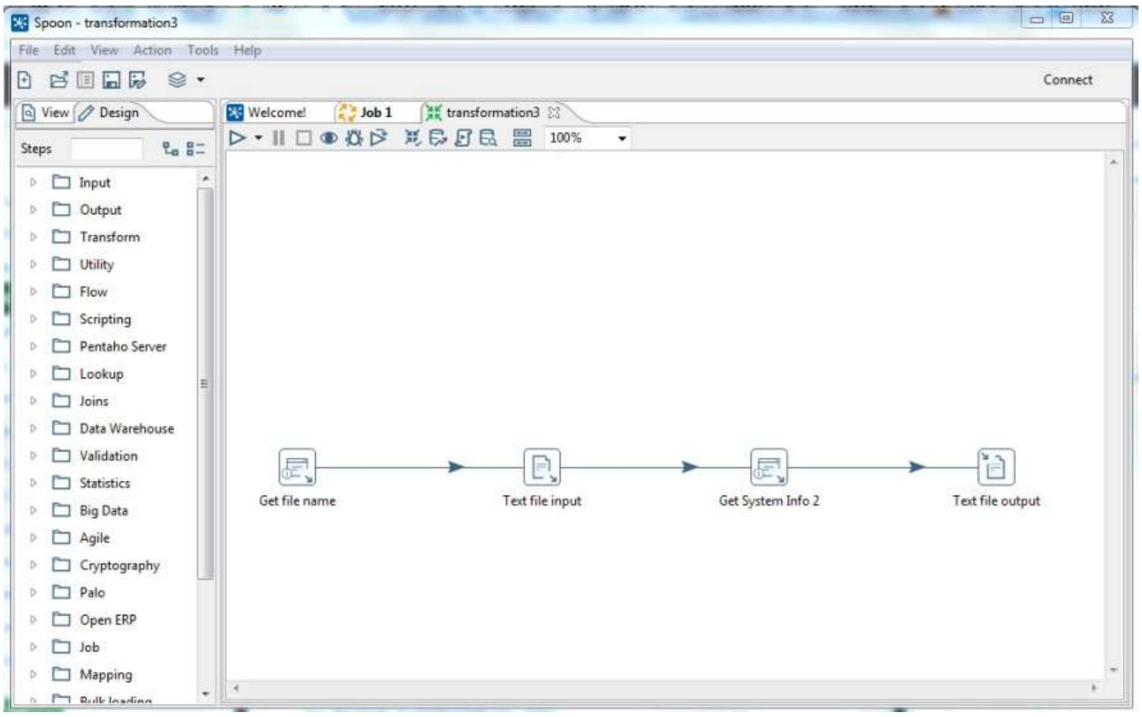


Figure 4.1 *Pentaho Data Integration tool*

In Pentaho:

- Jobs correspond to the set of transformations that constitute the base of all data integration processes including all the different steps required for extraction, transformation and data loading.
- Jobs make it possible to sequence multiple transformations related to different functionalities such as file management, script execution or error logging, etc.
- Each job is saved as an XML file and can be hosted on a database system to facilitate collaborative work.

Given that this tool is of type ETL engine, this component needs to be installed on all the different machines that run the Data Integration processes. The execution of these processes can take place on a local machine, remote server or a grid of servers. It is possible to test and visualize the flow of data for a specific step. The tool supports the connectivity with many DBMS and a variety of file types (CSV, XML, Excel...). It is also equipped with a mechanism for data quality.

On the community version, the administration panel makes it possible to manage data sources, users, launch planned jobs and monitor the different refreshment mechanisms.

On the commercial version, support is available; it is also possible to manage access rights with control over the content and the actions that can be performed by different developers. It also supports collaborative work with a version control system and tasks lock in case of modifications occur during execution. The planner is advanced and, other additional components make it possible to support other technologies such as JMS (Java Message Service), Hadoop, etc.

4.1.2 Talend Open Studio

Talend Open Studio was developed by Talend, a company mainly based in France with branches in the US and China.

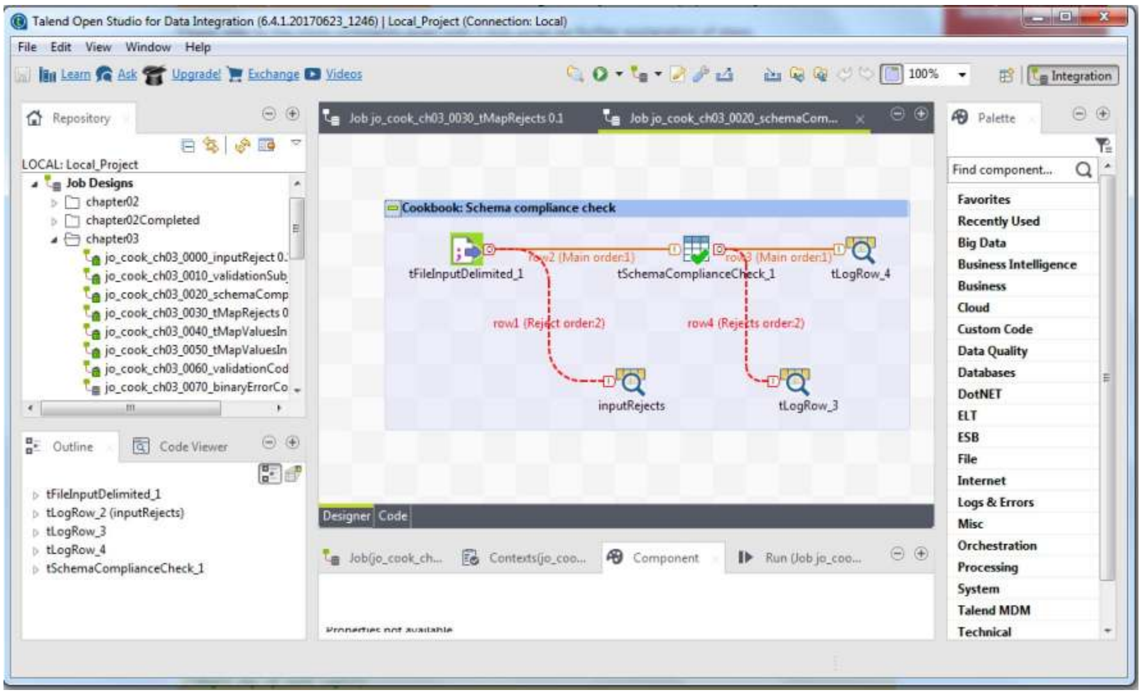


***Figure 4.2** Talend Open Studio*

Talend is an ETL of type Code Generator. A specific program in Java language is generated for each transformation job. The execution of the integration jobs consists of executing the compiled Java programs. The server or any other machine that runs the jobs do not require the installation of the ETL component but merely having the appropriate environment for running Java programs. It is also not required to run all the processes on a central server. It is possible to distribute the different tasks on different servers and also limit the usage of resources by the server (through parameterizing the execution of each script or via the additional functionalities available on the commercial version).

Talend Open Studio provides through an intermediate toolbox more than 200 components that offer basic functionalities such as transforming, data mapping or advanced ones like filtering or multiplexing. The tool also supports most of the DBMS, different file formats, LDAP (Lightweight Directory Access Protocol), web services, etc. Moreover, the components can be augmented using Java language or SQL.

The graphical interface "Job Designer" that is based on Eclipse makes it possible to create jobs with graphical and functional views of the integration processes. The processes are formed by placing the components and linking them with each other in the workplace using different connectors. Java code is generated in real-time and is accessible directly from the development environment. Modification of the generated code is not possible and can be done only through the graphical interface.

It is also possible to execute jobs using the "Job Designer" and test and debug the processes. This functionality makes it possible to monitor the execution of data flows; statistics are displayed in real-time showing the number of integrated records, rejected records as well as the processing speed. In addition to that, trace mode can be triggered to visualize the behaviour of the process record by record and analyze the result of each transformation.

Another graphical interface "Business Modeler" exist for designing data flow in a non-technical way. This interface allows other stakeholders to take part of the design of the data flow by modelling and documenting the different integration processes in the form of diagrams, which will help in orienting the development.

Talend also takes care of the meta-data related to the current data integration project and ensures the coherence between all jobs. The meta-data of the source and the target systems are easily loaded due to the available feature of database introspection (fetches the structure of all tables). Properties defined in the meta-data referential are also inherited by different jobs.

Talend studio also comes with a version control system as well as a context execution manager which facilitate maintenance and deployment of data integration processes. This solution introduces an advantage compared to other solutions since it allows to combine ETL approach with ELT (Extract-Load-Transform) approach.

The product is available in two versions: a community version under GPL license named "Talend Open Studio" and a commercial version titled"Talend Integration suite".

The commercial version provides additional functionalities such as a scheduler, collaborative work, automatic deployment, monitor of jobs execution as well as a professional support with a different service level according to the subscription.

4.1.3 Comparison and choice

The table below lists the differences between the two products:

	Pentaho Data Integration	**Talend Open Studio**
Documentation	++	+++
Support	Yes	Yes
Type	Engine	Code generator
Decrease of development effort	Yes	Yes
Collaborative work	Yes but limited on the community version	No (only on the commercial version)
Scheduler	Yes but limited on the community version	No(only on the commercial version)
Support ELT	No	Yes
Compatibility	++	+++
Performance	++	+++
Available components	++	+++
Dedicated CDC functions	No	No(only on the commercial version)

Table 4.1 *Comparison of ETL tools*

The two studied tools are proven data integration tools which are supported by large communities of users. The differences between the commercial versions and the open source ones of the two tools are more or less the same.

To conclude the comparison, we recommend Talend Open Studio as it has some advantages such as a broad set of useful components and a dedicated CDC functions on the commercial version. In addition to that, Talend integration jobs are Java programs that do not require the installation of the ETL tool on the environment of the execution. The need for the CDC functionality is not crucial at this stage, and the selected tool can meet the real-time requirements identified in the previous chapter.

We have chosen the community version of Talend Open Studio to respect the costs factor. The functionalities provided by the community version are sufficient compared to the actual needs, requirements and the development team size which is currently one person. Moreover, the missing functionalities do not handicap any requirements:

- For the collaborative work, in case of a team size increase, it is still possible to work in a team as long as best practices are respected.
- The missing scheduler to manage different jobs was not an issue as it was possible to use another open source schedulers such as "Job Scheduler" which can execute Talend jobs, report status and trace executions.
- The missing support might seem challenging; however, the available documentation is quite large, and the community is very active.

Finally, if it becomes necessary to go for a commercial version, the company can migrate easily to the new version as it will be compatible with what was developed using the community version.

Talend Open Studio was selected as an ETL tool, an additional component has been added to the system, "Job Scheduler" to ensure the planning and execution of data integration jobs.

4.2 Database Management Systems

Nowadays different structures and techniques are being used for storing data by different DBMS such as row and column-oriented databases. Each technique has its own advantages and drawbacks.

In the row-oriented structure, data is stored as complete records. Therefore, selecting any field would require reading the entire row and changing a single value means reading and rewriting the whole record. On the other hand, columnar databases store data in columns i.e. all the different values for a given column are stored in a contiguous data set. The columnar orientation improves the reading performance because only the required parts of the record are fetched from the database. It also allows data compression by removing duplicates of values belonging to the same column.

The insertion in column-oriented databases is relatively slow; however, some analytical queries are faster compared to the row-oriented systems, and this is due to the better performance in reading.

Many BI/analytical applications take in use an in-memory database engine. Data residing in the data warehouse will be transferred to such reporting tools for querying. Therefore, the difference between record-oriented and column-oriented performances can be ignored. Also, the fact that Newelo is currently using MySQL database for the development environment makes sure that there is no need to switch to a different technology. As a result, it was judged better to stick to the current DBMS to save time and avoid any extra expenses that might incur from obtaining a license for a commercial product just for the development of a prototype.

MySQL is an Oracle open source database management system available under two versions, a community version free to use under the GPL license, and a commercial enterprise version which offers advanced functionalities such as better scalability in case of high loads and customer support. This DBMS was developed for personal use by a company called MySQL AB founded in 1995. In January 2008, MySQL AB was sold to Sun Microsystems and later to Oracle in April 2009.

MySQL offers a variety of storing engines that can be assigned to different tables of the database. The two widely used engines are MyISAM and InnoDB. MyIsam is an easy to deploy engine which offers a good performance for reading. However, the performance for writing is relatively slow when there is a concurrent usage due to table-level locking. For example, the whole table is locked when there is an update query even if it only concerns one row. From a physical point of view, each table is defined by two files: one containing data, the other containing indexes and this on top of the file that defines the structure. The main drawback of MyISAM is that it does not support transactions and integrity constraints. It is frequently used when there are high demands for accessing data in read-only from different external systems.

InnoDB which comes by default in MySQL distribution presents the advantage of supporting transactions conforming to the ACID properties(Atomicity, Continuity, Isolation and Durability), supporting integrity constraints and row-level locking. In addition to that, the engine comes with an automatic recovery system. On the other hand, InnoDB management is more complex and consumes a lot of system resources. From a physical point of view, table storage consists of one file defining the structure and containing data and indexes at the same time. InnoDB is recommended for applications that require transaction management and for cases where data integrity must be guaranteed. It has become the default storage engine from version 5.5 of MySQL.

4.3 Data visualization tool

For data visualization, many commercial tools such as Qlik Sense, Tableau and Chartio are used in the BI domain. Tableau and Chartio offer a 14-day free trial for testing, whereas Qlik Sense comes with a variety of licensing options including free ones.

The main reason for using a data visualization tool at this prototyping stage is to be able to demonstrate and test the analytical features of the designed multi-dimensional models. Therefore, the selected tool should be as efficient as possible and does not incur any extra costs to be acquired. Qlik Sense desktop version was chosen as it meets the current requirement. For future use, different data visualization tools can be used for different customers.

Qlik Sense is a self-service data visualization and discovery application designed for individuals, groups and organizations. With Qlik Sense it is possible to analyze data and make data discoveries according to the needs that may raise unexpectedly. Enterprise version makes it possible to share knowledge and analyze data in groups and across organizations.

Qlik Sense comes with an integrated ETL module which allows the creation of analytical dashboards directly from disparate data sources which tends to be useful when there is no data warehouse implemented. It allows the realization of more complex transformations when compared with MySQL and provides a set of data connectors to some specific data sources such as ODBC, REST API, etc. It also consists of an in-memory database system that relies on the main memory of the computer which is faster than DBMS employing disk storage mechanism. The tool offers a data modeller module useful for designing and specifying associations between the different datasets making it easier to analyze data and run more complex queries.

Qlik Sense Desktop is only compatible with Windows platforms. However, the enterprise version can be installed on a dedicated server making it easier and more efficient to share and access dashboard via web-browsers between different users.

The different licensing options for Qlik Sense tool are divided into two main categories which can be classified as cloud and non-cloud options.

- **Qlik Sense Cloud**: based on a subscription license model offering two editions, one of them is free, named as "Cloud Basic". Both editions provide the possibility to create, edit and share interactive applications. However, the Basic edition allows the sharing feature with up to 5 users only, whereas the business edition does not have this limitation and offers group-level governance, coauthoring workspace and automated data refreshes which the basic edition does not provide.

- **Qlik Sense Non-cloud**: offered in two formats. The desktop edition is free with unlimited access, but sharing applications is not possible. With the enterprise edition, it is possible to have the tool running on an on-premises server. It offers the same features as the Qlik Sense cloud business edition.

In Qlik Sense desktop, refreshing data is performed manually. This may be considered as a disadvantage when having a real-time data analysis. However, the purpose of using a data visualization tool in this work was just to test and confirm the efficiency of the designed dimensional models. Other licensing options of Qlik Sense are not concerned with this limitation as they provide a feature for setting a refresh cycle for the data.

5. DESIGN & IMPLEMENTATION OF THE PROTOTYPE

This part focuses on the design and implementation of the prototype and its different components that are necessary to make it fully-functional and respond to the different requirements that were identified in chapter 3.

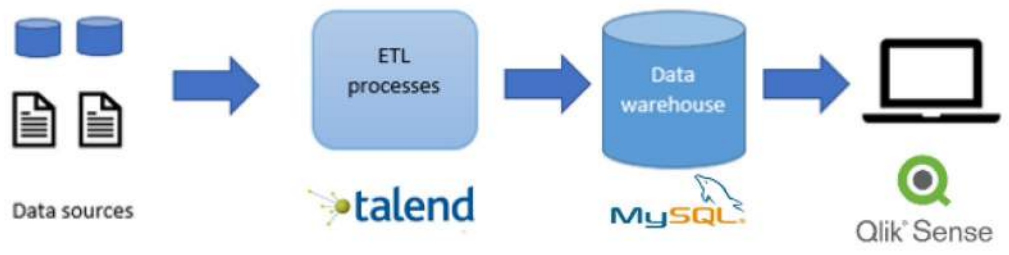


Figure 5.1 *Technical architecture of the prototype*

Figure 5.1 shows the general architecture of the prototype with its components which are: ETL layer, data warehouse and data visualization layer.

5.1 Development environment

The following hardware and software tools were selected for the development of the prototype:

5.1.1 Software tools

The tools and technologies that were studied and selected in Chapter 4 of this thesis were used for the development of the prototype.

For the ETL layer, Talend Open Studio version 6.4.0 was used for the design of ETL processes. MySQL 5.5.58 was adopted as a DBMS for hosting the data warehouse and Qlik Sense desktop edition was selected for testing the designed dimensional models. As for automating and scheduling the ETL processes, JobScheduler 1.11.5 was selected and used.

5.1.2 Hardware

Three machines have been used for the development and testing of the solution. The purpose was to be able to use different environments to ensure the compatibility and possibility of deploying the solution for production smoothly and successfully.

The first machine is a server computer used for hosting the data warehouse with the following specification:

- **Operating system:** Ubuntu server 16.04 LTS.
- **Processor:** Intel Xeon E5-2620V2 (6 cores).
- **RAM:** 48 GB.
- **Storage:** 3.3 TB.

The second machine is a laptop computer used for developing ETL processes and deploying them with the scheduler. The laptop has the following characteristics:

- **Operating system:** Windows 7 professional.
- **Processor:** Intel core i5 5200u 2.2GHZ (4 cores).
- **RAM:** 8 GB.
- **Storage:** 500 GB.

The third machine is also a laptop used for data visualization having Qlik Sense desktop installed on it. It has the following characteristics:

- **Operating system:** Windows 7 professional.
- **Processor:** Intel core i3 7300u 2.4GHZ (4 cores).
- **RAM:** 8 GB.
- **Storage:** 500 GB.

5.2 Design & implementation procedure

To make the design and implementation phase as smooth as possible, the development was divided into three different stages complementing each other. These were:

1. Specifying in details the different business activities that require analytics by defining the different facts, dimensions, hierarchies, etc.
2. Elaborating the logical design of the data warehouse using constellation schema since many fact tables share same dimensions. Once the logical design is completed, the physical structure of the relational tables is defined and implemented on the DBMS.
3. The next phase consists of designing and developing the different ETL jobs required for loading data into the data warehouse.
4. The last phase was about designing the different data visualizations for each business activity. It consists of connecting Qlik Sense to the data warehouse using its default connectors, and creating the different visualizations using the drag and drop features that Qlik sense offers.

5.3 Design of the Data Warehouse

After defining the user needs, we can start working on the design of the Data Warehouse as well as the dimensional modelling. The two business domains discussed in the requirements specification part share almost the same dimensions. The electricity consumption analytics needs to be performed in real-time. As a result, an extra dimension for representing time is required.

5.3.1 Dimensions participating in the constellation schema

Dimensions are used to describe facts. It is important to identify all the information that describes work orders related fixing to electricity meter communication faults and electricity consumption data.

1. **Date dimension**

   Typically, a date dimension is always available in any data warehouse. The lowest granularity level of this dimension is a single day. In our design, the date dimension is defined as follows:

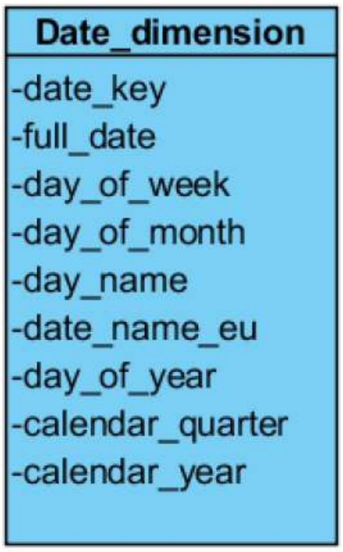


***Figure* 5.2** *Date dimension.*

Table 5.2 gives a description of the different attributes of the date dimension.

Attribute	Description
date_key	Primary key for date dimension.
full_date	Date in a complete format.
day_of_week	Number of the day of the week.
day_of_month	Number of the day of month.
day_name	Name of the day of the week.
date_name_eu	Date in European format.
day_of_year	Number of the day of the year.
calendar_quarter	Quarter of the date.
calendar_year	Year of the date.

***Table* 5.1** *Description of date dimension attributes*

2. **Time dimension**

   The time dimension is only used with the real-time partition for electricity consumption data fact. The lowest granularity for this dimension is a minute.

This dimension is not used with the non-real-time partition as it is not important to keep a detailed historical data for the past days. The real-time feature is only required for the current day. Table 5.2 provides a description of the attributes of the time dimension.

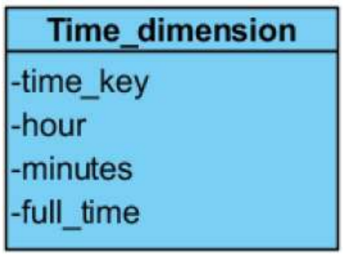


***Figure* 5.3** *Time Dimension.*

Attribute	**Description**
time_key	Primary key for time dimension.
hour	The hour of the day.
minute	Number of minutes of the hour.
full_time	Time in full format.

***Table* 5.2** *Description of time dimension attributes*

3. **Metering point dimension** This dimension is the biggest among the other ones as it contains information about the metering points, electricity meters' models, transformers' information, location data, etc. The lowest granularity of this dimension is the metering point. The different metering points are the location of every smart meter that is connected to the grid. Table 5.3 provides a description of the attributes of the time dimension.

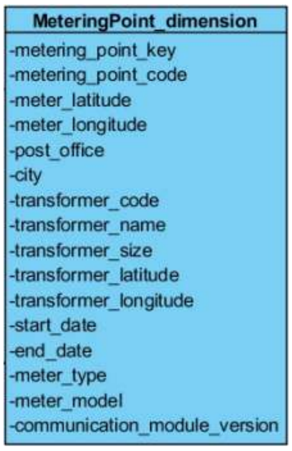


Figure *5.4 Metering point dimension.*

Attribute	**Description**
metering_point_key	Primary key for metering point dimension.
metering_point_code	The code of the metering point.
meter_latitude	The latitude of the metering point.
meter_longitude	The longitude of the metering point.
post_office	Post office of the metering point.
city	City of the metering point.
transformer_code	Transformer code that the metering point is connected to.
transformer_name	Transformer name.
transformer_size	Transformer size.
transformer_latitude	Transformer latitude.
transformer_longitude	Transformer longitude.
start_date	Installation date of the meter.
end_date	Removal date of the meter.
meter_type	Smart meter type.
meter_model	Smart meter model.
communication_module_version	Communication module version

Table *5.3 Description of metering point dimension attributes*

5.3.2 Facts participating in the constellation schema

The two business domains studied in the requirements specification part represent the two facts that participate in the constellation schema which are:

- **Work order fact (for fixing meter communication faults):** The measures of this fact table concern the work order itself and the main fixes that were performed to solve the fault. The measures are the work order ID and four booleans representing the nature of the fixes. These measures are aggregated using the count function.
- **Electricity consumption fact:** The only measure of this fact is the consumption value. The lowest granularity for the non-real-time partition is a day and therefore, the consumption values for a given day are calculated from the daily real-time partition before being swapped.

The different dimensions make it possible to perform various OLAP operations. For instance, the electricity consumption values can be summed up according to the transformer code and used to compare the calculated value with the transformer size to detect any electricity overload.

Figure 5.5 represents the constellation schema of the current solution. Other business domains and activities that were identified in this work are not discussed as they are regarded as confidential.

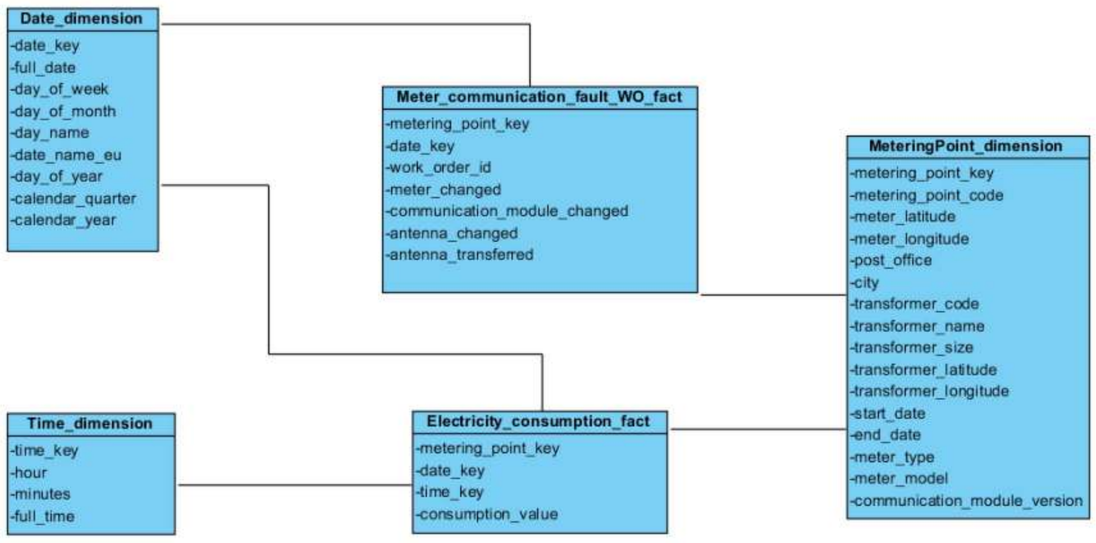


Figure 5.5 *Constellation schema.*

5.4 Design of ETL processes

In this section, we describe the process of designing the different ETL jobs. Some specific data integration jobs are described in details to provide an overview of how the ETL development was performed in this work.

One of the key concepts of using Talend Open Studio is to use metadata. This concept is important as it concerns every data source or target. Metadata makes it easier to reuse and maintain the different jobs when changing data sources or targets. For instance, changing one data source in the meta-data will propagate all the necessary changes to jobs using that data source. It is beneficial as it reduces the development and maintenance time.

5.4.1 Data loading strategy

To follow best practices[7] two possible options for data loading can be adopted for data warehousing. For tables with a size not exceeding a million of rows, data loading is performed using a standard SQL insert command. On the other hand, bulk-loading should be used for large tables. Bulk-loading consists of storing ordered data in a file and then loading it into the database using a specific command.

From the requirement elicitation chapter, we identified two different intervals for loading data for the different fact tables. A weekly loading schedule was judged best for most of the facts; however, the electricity consumption fact requires data to be loaded every 15 minutes into the real-time partition.

5.4.2 ETL processes for handling dimensions

Case of simple dimensions, time and date

Time and date are simple dimensions as they do not require any extraction of data from any data sources. SQL scripts were used to generate dates from January 1st, 2013 to December 31st, 2029 as well as the different time interval for a single day consisting of minutes and hours. For facts that require precise time indication, both dimensions are used, and their combination will specify the date and time. The time dimension is used for the electricity consumption fact. All of the other facts were only linked to the date dimension.

Case of slowly changing dimensions

Some attributes for specific dimensions such as the "metering point dimension" could change values over time. Therefore, such dimensions must be able to handle these updates to keep the data analytics coherent.

Many methods are used for tackling this issue that, are categorized into different types:

- **Type 1:** This method is straightforward as it consists of just overwriting the old value with the new one making it impossible to track the historical data.
- **Type 2:** Consists of creating multiple records for a given natural key and adding some columns to indicate the version, effective date, status flag, etc.
- **Type 3:** Consists of adding new columns to preserve old values. The issue with this method is, that it is only possible to truck a limited history according to the number of columns.

Broken or old electricity meters are replaced by new ones. This means that one metering point can have more than one meter installed at different periods. In our solution, we opted for the second method (type 2) to tackle the issue of this slowly changing dimension as we wanted to keep a detailed history of all the smart meters.

5.4.3 ETL processes for facts

Once all the dimensions are ready and updated, loading facts into the data warehouse becomes straightforward. It consists of extracting data from operational systems by pulling the entries respecting the time-stamp conditions. We can differentiate two cases for loading data into the data warehouse which are: the first loading and the incremental loading. The first loading takes place when deploying the data warehouse. It consists of extracting all the data entries that are available in the source systems and meet certain conditions. The incremental loading is done periodically (weekly or every 15 minutes). It consists of loading the new data entries that started to exist after the last refresh cycle. After the extraction, the extracted data is transformed and loaded into the data warehouse.

5.4.4 Real-time ETL process

As discussed earlier, the refresh cycle that was set for meeting the requirements for a real-time Data Warehouse was defined as 15 minutes. The real-time capability is only needed for electricity consumption data, and the solution consists of increasing the refresh cycle of the ETL process. The data is inserted into the real-time partition and swapped into the historical one at the end of the day.

5.5 Deployment

To better describe the deployment of the prototype, UML deployment diagram has been used to present the deployment architecture. The prototype consists of an extraction layer, data storage and a data visualization layer which are presented in figure 5.6

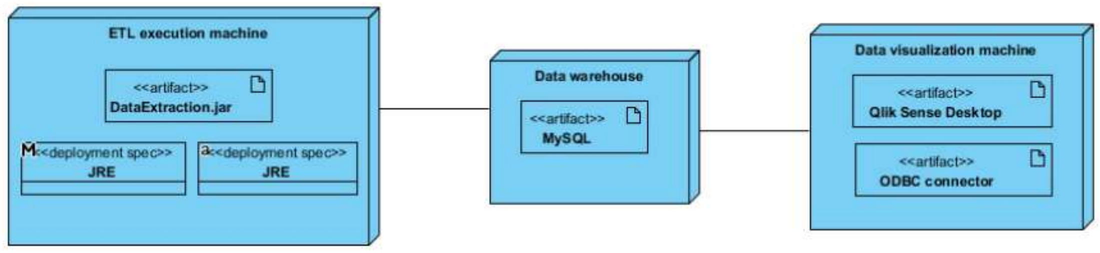


Figure 5.6 *Deployment diagram*

6. EVALUATION

In this section, a brief evaluation of the study is presented together with a discussion about the developed prototype.

6.1 Evaluation of the study

The presented study focused on the various concepts related to BI as well as the different approaches used for designing and implementing traditional and modern real-time BI solutions. Elaborating the state-of-the-art allowed us to understand the notions related to RTBI and revealed a lot of information about the techniques that are currently used for developing such systems. We have concentrated on the selection of the right approaches, technologies and tools for designing and implementing a prototype that would meet the specified requirements.

The most challenging part of this work was the requirements specification phase. A Hybrid approach (analysis/source-driven approach) was adopted for eliciting the different requirements. However, many obstacles were encountered such as the difficulty to arrange interviews with different customers, data access restrictions and the limited knowledge of some interviewees about the basic notions of BI.

The real-time requirement was identified for a single subject which was electricity consumption data. The near real-time ETL technique was combined with the "Trickle and Flip" approach in order to reduce the time of data extraction and decrease the load on some specific fact tables of the data warehouse.

Finally, the study of the different notions and concepts related to BI in general and RTBI in specific allowed us to gain an in-depth knowledge about decision support systems and provided us with the required skills to design and implement a functional prototype.

6.2 Evaluation of the prototype

The implementation of the prototype allowed us to verify the feasibility of developing a BI solution as a service for Newelo's customers. The developed prototype was tested and proven to conform to the diverse needs identified during the requirements elicitation phase. It was also proven to be fully-functional with the hardware and software configuration selected in the study.

The real-time capability, which was set as a requirement from the start of the project has not been tested with a huge set of data due to the access-rights restrictions. Therefore, randomly generated data was used to simulate the use case with a relatively small amount of data. The real-time feature was taken into consideration in all phases of the study. For instance, software tool selection was based on the fact that the final solution would respond to the real-time needs.

The use of Qlik Sense to analyze and visualize the data cubes demonstrated the benefits of using such tools that come with in-memory DBMS that are capable of generating queries for OLAP operations automatically by detecting the data warehouse schema.

Although the selected tools were proven to be effective for the prototype, the possibility of using cloud service for hosting the data warehouse can offer an advantage to avoid any scalability issues especially with real-time data that grow in size quickly. The separation between the data warehouse and the visualization layer makes it possible to switch to other visualizations tools in order to respond to future customer needs.

Overall, the implementation of the data warehouse and the use of the ETL tool to manipulate raw data, demonstrated the strength of calculating new attributes thus, providing more knowledge that was not possible to achieve using OLTP systems.

7. CONCLUSION

In this work, we have studied the different concepts related to BI as well as the different techniques related to designing and implementing BI solutions. The aim of the study was to define the different requirements for Newelo's customers, and the tools to be used for implementing a solution capable of handling real-time data. Different methods and approaches for implementing real-time data warehouses were also discussed in this thesis.

During the study, we have followed all of the approaches for designing and implementing a BI solution that were specified in the second chapter. For defining user requirements, we opted for the mixed approach known as "analysis/data-driven approach". We also studied some tools and technologies that are currently used by the BI community where a priority was given to free and open source ones to reduce the maximum of costs that might incur from developing the prototype.

At the end, we were able to provide a fully-functioning prototype and demonstrate it to the customers that were involved in the requirements specification phase. The customers were interviewed at the end of each demonstration to better evaluate the solution. The need for future development can be foreseen and many perspectives can be suggested in what follows:

- Test the performance of the real-time capability with larger datasets and try some "Change Data Capture" mechanisms if necessary.
- Study the possibility of using a REST API for feeding the data warehouse with real-time data.
- Iterate through the requirements elicitation phase to identify more needs and develop the conceptual model further.
- Study the possibility of embedding data analysis and reports in operational systems.

BIBLIOGRAPHY

[1] B. Azvine, Z. Cui, D. D. Nauck, and B. Majeed, “Real time business intelligence for the adaptive enterprise,” in *E-Commerce Technology, 2006. The 8th IEEE International Conference on and Enterprise Computing, E-Commerce, and E-Services, The 3rd IEEE International Conference on.* IEEE, 2006, pp. 29–29.

[2] B. Data, “for better or worse: 90% of world’s data generated over last two years,” *SCIENCE DAILY, May*, vol. 22, 2013.

[3] B. Devlin, “Will data warehousing survive the advent of big data?” *O’Reilly Radar.*, 2011. [Online]. Available: http://radar.oreilly.com/2011/01/data-warehouse-big-data.html

[4] W. Eckerson, “Gauge your data warehouse maturity,” *Information management*, vol. 14, no. 11, p. 34, 2004.

[5] J. C. Hancock and R. Toren, *Practical Business Intelligence with SQL Server 2005.* Pearson Education, 2006.

[6] W. H. Inmon, *Building the Data Warehouse,3rd Edition*, 3rd ed. New York, NY, USA: John Wiley & Sons, Inc., 2002.

[7] R. Kimball, *Le data warehouse: Guide de conduite de projet.* Eyrolles, 2005.

[8] A. Kotopoulis, “Best practices for real-time data warehousing(white paper),” *An Oracle White Paper*, 2014.

[9] M. Kromer, “Modern hybrid big data warehouse architectures,” *Business Intelligence Journal VOL. 19, NO. 4*, 2015.

[10] J. Langseth, “Real-time data warehousing: Challenges and solutions,” *DSSResources. com*, vol. 2, no. 08, p. 2004, 2004.

[11] C. R. Powers, K. C. Gardner, T. J. Beauchamp, T. C. Netsch, and G. D. O. Nicholls, “Architecture for general purpose near real-time business intelligence system and methods therefor,” Sep. 18 2007, uS Patent 7,272,660.

[12] E. Turban, R. Sharda, J. E. Aronson, and D. King, *Business intelligence: A managerial approach.* Pearson Prentice Hall Upper Saddle River, NJ, 2008.

[13] A. Vaisman and E. Zimányi, *Data Warehouse Systems: Design and Implementation.* Springer, 2014.

[14] H. J. Watson and B. H. Wixom, "The current state of business intelligence," *Computer*, vol. 40, no. 9, 2007.

[15] H. J. Watson, B. H. Wixom, J. A. Hoffer, R. Anderson-Lehman, and A. M. Reynolds, "Real-time business intelligence: Best practices at continental airlines," *Information Systems Management*, vol. 23, no. 1, p. 7, 2006.