An Evaluation of the Challenges of Multilingualism in Data Warehouse Development

Nedim Dedić and Clare Stanier
Faculty of Computing, Engineering and Sciences, Staffordshire University,
Beaconside, Stafford, Staffordshire, ST18 0AD, U.K.

Keywords: Business Intelligence, Data Warehousing, Multilingualism, Star Schema, Semantic Web.

Abstract: In this paper we discuss Business Intelligence and define what is meant by support for Multilingualism in a Business Intelligence reporting context. We identify support for Multilingualism as a challenging issue which has implications for data warehouse design and reporting performance. Data warehouses are a core component of most Business Intelligence systems and the star schema is the approach most widely used to develop data warehouses and dimensional Data Marts. We discuss the way in which Multilingualism can be supported in the Star Schema and identify that current approaches have serious limitations which include data redundancy and data manipulation, performance and maintenance issues. We propose a new approach to enable the optimal application of multilingualism in Business Intelligence. The proposed approach was found to produce satisfactory results when used in a proof-of-concept environment. Future work will include testing the approach in an enterprise environment.

1 INTRODUCTION

Users today expect to access relevant information in the semantic web in their own language (Gracia et al, 2011). This is also the case when accessing analytical data relevant for decision making using Business Intelligence (BI) applications, such as reports or dashboards. In this paper, we discuss the application of BI in an international context focusing the issue of Multilingualism (ML). Multilingualism is defined further in section two but can be understood as data manipulation and reporting in more than one language. Understanding the issues presented by ML requires discussion of the Data Warehouse (DW), which is a core component of BI (Olszak and Ziemba, 2007). We also discuss data marts and focus on the star schema as the most accepted form of dimensional modelling. We evaluate current solutions and approaches to the implementation of ML using the star schema in BI environment and propose an improved approach.

The rest of this paper is structured as follows: section 2 defines BI and ML and the requirement to support ML in a BI context; section 3 discusses ML in a Data Warehouse environment, outlining DW concepts and design approaches, the role of the star schema and the issues presented when implementing ML in a star schema; section 4 presents the conclusions and evaluation, proposing a revised approach to supporting ML.

2 PROBLEM CONTEXT: BUSINESS INTELLIGENCE AND MULTILINGUALISM

To survive in today's business a company has to continuously improve productivity and efficiency, while management and executives have to make decisions almost immediately to ensure competitiveness (Huff, 2013). Information is used to enable improved decision making and efficiency (Yrjö-Koskinen, 2013; Hannula and Pirttimäki, 2003). This process is supported by activities, processes and applications which are collectively known as Business Intelligence. Business Intelligence was initially used to describe activities and tools associated with the reporting and analysis of data stored in data warehouses (Kimball et al, 2008). Dekkers et al., (2007) define BI as a continuous activity of gathering, processing and analysing data. BI helps companies to out-think the

Dedić, N. and Stanier, C.
An Evaluation of the Challenges of Multilingualism in Data Warehouse Development.
In *Proceedings of the 18th International Conference on Enterprise Information Systems (ICEIS 2016) - Volume 1*, pages 196-206
ISBN: 978-989-758-187-8

competition through better understanding of the customer base (Brannon, 2010) and can provide competitive advantage (Marchand and Raymond, 2008), and support for strategic decision-making (Popovič et al., 2010). From the early days of computing, computer technology and software has been associated with development in the English language (Hensch, 2005) and Business Intelligence is no exception.

BI is a fast evolving field (Obeidat et al, 2015; Chaudhuri et al., 2011) and although traditional BI focussed on activities such as data warehousing and reporting, the new generation of BI has an additional focus on data exploration and visualisation (Anadiotis, 2013; Obeidat et al, 2015), which increases the demand for ML. The business and legal context of BI is also evolving. With emerging markets and expanding international cooperation, especially in the case of European Union, there is a requirement to support BI in languages other than English. Users today expect to access information in the semantic web in their own language (Gracia et al, 2011). Based on the online profiles of the biggest European companies (Forbes, 2015), most of these companies are international in their nature. Business users expect to be able to use software and applications, including Business Intelligence, in their own language for the purpose of better productivity (Hau and Aparício, 2008). Thus, when expanding their business to new countries, companies need to extend and adopt their BI infrastructure to support optimal use of local languages. In a European context, the requirement for multilingualism has legal underpinnings in some countries. Most European countries have laws on the official use of their respective languages in public communication (Italian Law No. 482, 1999; Constitution of France, 1958; Constitution of Croatia, 1990; Federation Constitution, 1994; Spanish Constitution, 1978). For international companies, this means a requirement to support the use of several languages to obey local laws. Where there is a need to support multiple languages, there is an imperative to enable transfer and processing of textual accessibilities for localization purposes (Vazquez, 2013). This also applies in a Business Intelligence environment.

Multilingualism is complex phenomenon which can be seen from different perspectives and has many definitions (Cenoz, 2013). The European Commission (2008, p.6.) defines it as "the ability of societies, institutions, groups and individuals to engage, on a regular basis, with more than one language in their day-to-day lives". In the Business Intelligence DW and presentation context, we define multilingualism as the ability to store descriptive information and to use this information in a semantic layer in more than one language. The changing attitudes of business users, the importance of emerging and international markets and ever-growing local data warehousing communities are additional issues that justify the application of multilingualism in Business Intelligence. Multilingualism, however, presents challenges for design and reporting in data warehouses.

3 MUTILINGUALISM IN A DATA WAREHOUSE ENVIRONMENT

3.1 Data Warehouse Concepts

Data warehouses are seen as core in the development of BI systems (Olszak and Ziemba, 2007). A widely accepted definition of the Data Warehouse is provided by Inmon (2005) who defined a Data Warehouse as a collection of integrated databases designed to support the DSS (decision support system) function. In this definition, the data warehouse from the architectural point should be almost the same as the source system and may also have data marts, or aggregated tables that are used for reporting and querying purposes. Linstedt et al (2010) propose very similar concept known as the Data Vault approach. Differentiation is only in the context of modelling and storing information inside the Data Warehouse. In the Data Vault approach data is loaded from the source system in its original format (Linstedt et. al, 2010). The Data Vault approach is built around Hubs, Links and Satellites (Jovanović et al., 2014). Hubs represent source system business keys in a master table, links are associations between hubs with validity period, and satellites point to the links containing attributes of transaction with validity period (Orlov, 2014). As the structure of the data is highly normalized (4NF+), this approach for the implementation of a data warehouse is not adequate for direct reporting and requires additional dimensional data marts to enable reporting or querying (Orlov, 2014).

Kimball et al (2008) presented an alternative view of the data warehouse, arguing that a data warehouse should be seen as a collection of data marts, which are used for querying and reporting and connected using conformed dimensions. In this approach, there is no requirement to replicate all the data from the source system, just the data needed by the business.

3.2 The Star Schema

Much of the literature on the development of data warehouses, and particularly the seminal works by Inmon and Kimball, dates from the end of the 20th century/the first decade of the current century. There has been comparatively little recent work on data warehouse design and schema development although there is a significant literature on data warehouse development and optimisation (Cravero and Sepulveda, 2015; Dokeroglu et al., 2014; Sano, 2014; Graefe et al., 2013). Inmon and Kimball are the two seminal figures in this field and although their data warehouse design philosophies, as discussed in section 3.1, are opposed, both propose dimensional modelling and the use of data marts (star or snowflake schema) for reporting (Orlov, 2014). The Data Vault approach introduced by Linstedt (2010) also proposes the use of data marts (in the form of star or snowflake) for reporting. Inmon (1995), Kimball (2008) and Linstedt (2010) all recommend the use of the star schema as the most appropriate design strategy for the development of data marts. Additionally, a survey paper by Sen and Sinha (2005) examined the approaches used by 15 data warehouse vendors and found that 12 of the 15 vendors supported the star schema (alone or in combination with others star schema based approaches).

A star schema is a collection of dimension tables and one or more fact tables (Cios, Pedrycz, Winiarski et al, 2007). Dimensions have a key field and one additional field for every attribute (Kimball et al, 2008; Jensen et al, 2010). The fact table is a central table that contains transactional information and foreign keys to dimensional tables, while dimensional tables contain only master data (Kimball et al, 2008; Jensen et al, 2010; Cios, Pedrycz, Winiarski et al, 2007). In visual model representation, the dimension model resembles a star, thus the name (Jensen, 2010). The main benefits of the star schema design are ease of understanding and a reduction in the number of joins needed to retrieve the data (Cios, Pedrycz, Winiarski et al, 2007).

In dimension tables, the primary key is used to identify the dimensional value, while hierarchy is defined through attributes. Dimension tables do not confirm to the relational model strategy of normalisation and may contain redundancy (Jensen et al, 2010). The Fact table, on the other hand, holds the foreign key to dimensional table values and as there is no redundancy it could be considered to be in 3NF (Jensen et al, 2010). All foreign keys to the dimensional tables build together the primary key for the fact table.

There are other schemas used for the purpose of dimensional modelling, such as snowflake or galaxy. Cios, Pedrycz, Winiarski et al (2007) consider the snowflake and the galaxy schemas as the variations of the star schema, while Inmon (1995), Kimball (2008), Linstedt (2010), Corr and Stagnittno (2014) and Jensen et al (2010) consider it as a separate dimensional modelling philosophy and not as a variation of the star schema.

The star schema is the dimensional modelling concept most used by the industry (Sen and Sinha, 2005). It is recommended as the most appropriate design strategy for development of data marts and is considered as a general dimensional modelling approach in the data warehouse (Inmon, 1995; Kimball, 2008; Linstedt, 2010). Thus, in this paper we focus on the issues of multilingualism exclusively within star schema. Although the accepted design approach for DW development, and regarded as a good fit for business requirements (Purba, 1999), star schemas present issues when handling multilingual systems.

3.3 Multilingualism Issues in Data Warehouse Design

Conventional Business Intelligence uses a process known as ETL [Extract-Transform- Load] (Kimball et al, 2008; Inmon, 2005) in which data is extracted from data source applications, transformed and loaded into a data storage medium, typically a data warehouse or data mart and then analysed to enable meaningful reporting usually via web or desktop applications. However, this process is most effective in a single language environment. Language, including multilingualism, is a difficult issue in software localization (Collins, 2002) especially in a multidimensional context; and the expansion of BI systems to enable reporting in different languages is not trivial. In the context of multilingual websites, several factors have been identified when presenting to different range of audience in different countries

Table 1: Simple Product dimension.

Key	Description	Code	Category	Subcategory	From_Date	To_Date
123	Apples	FA	Fruits vegetables	Fruits	01.01.2014	31012014
124	Beer	DB	Drinks	Alcoholic	01.01.2014	31012014

(Hillier 2003). The most important are cultural context and accessibility of applications in local language. Creating and maintaining a web environment in a multilingual perspective creates special challenges, both cultural and technical (Huang and Tilley, 2001). Besides the standard issues that arise during translation process from one to other language (such as meaning of words, terminology, phrases, text direction, date formats, etc.) we face additional technical issues when translating texts in computer-based environment (Hillier, 2003).

These problems may range from different application environments to different implementation standards. To optimally apply multilingualism to existing Business Intelligence environment it is necessary to identify the issues in a BI environment. The next section examines three possible workarounds based on amendments to the star schema: including additional attributes, extending the primary key and providing additional dimension tables. All these solutions, as discussed below introduce problems such as extreme data redundancies leading to performance issues, and implementation and maintenance difficulties.

3.3.1 Adding Additional Attributes for New Languages to Dimension

One approach, derived from Kimball's method for delivering country-specific calendars (Kimball and Ross, 2011), recommends that where there are new values for the dimension tables in star schema, we simply add new attributes to the dimension. This approach is also proposed by Imhoff et al (2003) as a solution for simultaneous bilingual reporting. Imhoff et al (2003) state that if we need to provide the ability to report in two or more languages within the same query, we need to publish the data with multiple languages in the same row. When implementing dimensions using this approach, attributes should be descriptive, added in the form of textual labels that consist full words, without missing values, discreetly valued and quality assured (Kimball et al 2008). This is illustrated by the simple Product dimension shown Table 1. As we see, attributes in the Product dimension (Description, Code, Category and Subcategory) are textual fields and at this point, the star schema would be sufficient for the most companies to implement functional and optimal data marts. The limitation of this approach in a multilingual environment would be extremely large dimension tables. For example, if there are ten descriptive attributes for "product" dimension, in the case of the five languages, there would be an additional forty columns.

To demonstrate the problem, we convert the product dimension table (Table 1) to a conceptual view (Table 2a). The sample Product dimension, based on Table 2a, which additionally includes German, Italian and Bosnian language besides English would look like Table 2b.

Table 2a: Conceptual view of the Product dimension.

Key *(Primary Key)*
Description
Code
Category
Subcategory
From_Date
To_Date

Table 2b: Product dimension in English, German, Italian and Bosnian language.

Key *(Primary Key)*
Description
Code
Category
Subcategory
From_Date
To_Date
Description_DE
Code_DE
Category_DE
Subcategory_DE
Description_IT
Code_IT
Category_IT
Subcategory_IT
Description_BA
Code_BA
Category_BA
Subcategory_BA

As we see, new attribute columns for German, Italian and Bosnian language are added for every possible textual description. They have suffix **_DE**, **_IT** and **_BA** (Table 2b).This simplified example does not fully convey the scale of the problem. In implementation practice, the Product dimension might contain more than 20 of textual attributes and the problem would be replicated for all dimensions. A real-world example of a Product dimension would include descriptive attributes to be used as reporting aggregates, for example, description, category, subcategory, assortment, assortment area, buying department, brand, brand origin, country, international categorization, product level, season information, product state, class and type.

As they require large amounts of maintenance

time and CPU (Poolet, 2008), large and wide dimension tables can be problematic, especially for rapidly changing dimensions such as a Customer dimension (Ponniah, 2004). Rapidly changing dimensions are those dimensions where attribute or hierarchical values change frequently (Boakye, 2012). As an example, consider a Customer dimension with several million rows of data intended to be used in five languages. In this example, the Customer dimension has descriptive attributes such as "buying category", "buying frequency" and "monetary value" in all five languages. These categories are intended to be updated on a daily basis. This and similar scenarios could lead to system overhead on a daily basis. In addition, wide dimension tables require duplicate storage for descriptive attributes and make ETL transformation complex as the language-based columns must be taken into account. More complex SQL/MDX statements are required with different language-based columns to change the language of data previews at the semantic level (reports, queries or dashboards). Moreover, queries that return data sets must be re-executed in the required language. There are other external, but related problems caused by using this approach. For example, updating or changing some descriptive hierarchical attributes that are used as basis for tables containing aggregated data. Suppose a specific group of products change their category from non-alcoholic drinks to energy drinks, affecting also subcategories. It is necessary to update the dimension table to change the descriptive records for every language and also to re-aggregate the data in tables holding aggregated data. In this scenario, we would have to delete all data in tables that hold aggregate data according to category and re-aggregate. The process of re-aggregation could take several days if we have billions of records in fact tables, which is not unusual; Wallmart.com sells more than 4.000.000 and Amazon.com more than 350.000.000 different products (Scrapehero.com, 2015). The situation is more critical with wide dimension tables that represent rapidly changing dimensions. The overhead would also increase as the company needs to store more languages.

3.3.2 Extending a Primary Key to Include Language Identifier

The second approach, discussed by Imhoff (2003), proposes extending a primary key to include a language identifier. As we can see from Table 3, the limitation in this case is duplication of the records with every new language. With five languages for the product dimension, which for example holds one million of data, there would be five million records.

Larger dimension tables slow the process of query execution and make it harder to manage updates according to the slowly changing dimension rules. Slowly changing dimensions are dimensions whose attribute or hierarchical values change over time, but unlike rapidly changing dimensions, values are changed unpredictably and less frequently (Kimball et al, 2008). This approach to multilingualism is problematic also for rapidly changing dimensions (Ponniah, 2004), and as with the additional attributes approach, makes heavy increased demands in terms of maintenance time and CPU (Poolet, 2008). From a memory perspective this approach is less efficient than that discussed in 3.3.1 as it doubles storage requirements with every additional language. This approach also suffers from the semantic layerproblems previously discussed: to change the language of data preview on semantic layer (reports, dashboards), query statements that return data sets must be re-executed. This approach, unlike the additional attributes approach, does not lead to more complex ETL transformations and SQL/MDX statements but the same problems exist with regard to rapidly changing dimensions and changing the structure of externally aggregated tables. For companies using several languages and holding millions of records in their dimensions, re-executing queries and re-aggregating data according to a specific language can be time, memory and CPU demanding.

3.3.3 Schema and Multiple Dimensional Tables Solution

A third approach discussed by Kimball (2001), Imhoff et al (2003) and Corr and Stagnittno (2014), proposes implementing one fact table and multiple dimensional tables – depending on the number of languages required. Different languages are saved in different database schema and/or in different tables. For example, for five different languages, we would have five "product" dimension tables - one for every language. For the same example, if there are one hundred initial dimensions in data warehouse, five hundred dimension tables would be required to satisfy the MLrequirements for five languages. This approach has numerous limitations. As additional tables and possibly additional schemas are needed in the data warehouse, this approach makes ETL processes more complex as the language-based tables must be planned for. It requires additional transformations to every table for every additional

Table 3: Product dimension with extended primary key.

Key	Lang	Description	Code	Category	Subcategory	From Date	To Date
123	EN	Apples	FA	Fruits vegetables	Fruits	01.01.2014	31.01.2014
124	EN	Beer	DB	Drinks	Alcoholic	01.01.2014	31.01.2014
123	DE	Äpfel	FA	Obst und Gemüse	Obst	01.01.2014	31.01.2014
124	DE	Bier	DB	Getränke	Alcoholisch	01.01.2014	31.01.2014
123	IT	Mele	FA	Frutta e Verdura	Frutta	01.01.2014	31.01.2014
124	IT	Birra	DB	Beve	Alcolico	01.01.2014	31.01.2014
123	SI	Jabloka	FA	Sadje in Zelenjava	Sadje	01.01.2014	31.01.2014
124	SI	Pivo	DB	Pijače	Alkoholna	01.01.2014	31.01.2014
123	BA	Jabuka	FA	Voće i povrće	Voće	01.01.2014	31.01.2014
124	BA	Pivo	DB	Pića	Alkoholna	01.01.2014	31.01.2014

language. The data to be used for aggregation and reporting is doubled and so is the metadata for tables and schemas. This approach requires more complex SQL/MDX statements than two previous approaches, and changing the language of data preview at the semantic level requires the query to be re-executed. Changing any descriptive data in dimensions requires re-aggregation of relevant tables holding aggregated data, which can be critical considering the ETL and SQL/MDX complexity of this approach. If one part of the business (country), for example, changes the ID for a specific dimension value, this could lead to consistency problems. Having different IDs for the same data category in different languages disables consolidated reporting for that aspect at the enterprise level. Other subtle problems that might arise when using this approach as discussed by Kimball (2001) include: the possibility of translating two distinct attributes as the same word in a new language causing ETL and reporting problems; reports cannot preserve sort orders easily across different languages. To overcome these problems, this concept requires additional programming or the application of additional or surrogate keys as actual keys in fact table.

3.3.4 Discussion

Besides data redundancy, sub-optimal memory usage and complex management, the common limitation of the all three approaches is the fact that the business users need to re-execute the query or report to change the interface language. As BI-based reports and queries do complex calculations besides querying huge amounts of data from the data warehouse, this is problematic and introduces performance issues. Further, in the case of translation corrections for master data, corrections have to go through the whole ETL process. For example, to change an attribute description for a specific language, it is necessary to change the value in the source system, then wait for the execution of the process for the respective master data. Other options, such as normalizing star schema dimensions for the purpose of multilingualism could lead to the Star schema developing into a snowflake schema. A snowflake schema is not an optimal design solution for data marts with large amounts of data. Issues include: the execution of the main query needs to consider all joins between tables in the same dimension; multi-level dimension tables inside one dimension have to be joined during query execution; the structure of the snowflake is complex and includes large number of database tables per dimension; harder implementation and maintenance of ETL processes; greater complexity in reorganization than the star schema; changes in the semantic model could lead to the extensive reorganization which would use additional resources.

3.4 Vendor Specific Approaches: SAP Extended Star Schema

We reviewed the data warehouse and BI software market and found that the biggest vendors, such as Oracle, IBM and Microsoft, support one or more of three approaches explained above. However, one of the biggest vendors in Business Intelligence, SAP proposes a specific solution for ML, using the concept of an extended star schema, which also includes language as part of the key. In this approach the dimensions and the fact table are linked to one another using abstract identification numbers (dimension IDs), which are contained in the key part of the respective database table (SAP, 2015). Dimensions are not represented as one table with redundant data as in classical star schema. Values from the tables that hold information about a specific dimension attribute text or value are mapped to an abstract dimension key.

This is an implementation driven approach which is only supported by SAP BW. This means it cannot be seen as a general design solution as it is a vendor

specific proprietary solution which relies on complex joins to retrieve content for reporting purposes.

4 PROPOSED SOLUTION AND FUTURE WORK

4.1 Conclusions

We have reviewed the problems presented by ML in a BI context and discussed the limitations of current solutions to the design challenge of supporting ML in the star schema. Existing solutions propose design workarounds based on extensions of the star schema but lack theoretical underpinnings and introduce data redundancy and data manipulation and performance and maintenance issues. The challenge of ML in a data warehouse environment is to develop an approach that can support ML without introducing inefficiencies into the star schema.

4.2 Proposed Solution

Imhoff et al (2003) propose that language-based values should be generated during the delivery process. Corr and Stagnittno (2014) suggest a similar approach and propose attribute name translation handled by the BI tool semantic layer. The main challenge to providing support for ML in the context of the Star Schema is that attribute and hierarchy descriptions are saved inside the dimensional tables. We therefore propose an alternative approach that would regard the Star Schema as a higher level entity and save textual descriptions from attributes and hierarchies elsewhere as language files.

Figure 1 shows the concept of proposed approach based on a three-layered BI architecture idea. We describe more detailed technical functionality of the proposed solution in Figure 2. Figure 1 shows that the master and transactional data are extracted from source layer applications into a staging area. From the staging area data can be extracted directly into reporting Data Marts, following the philosophy of Kimball (dashed arrows), or to Data Warehouse and then to reporting Data Marts, following the Data Vault and Inmon philosophies (full line arrows). Before storing data into reporting Data Marts, attributes and hierarchical descriptions are extracted, together with their IDs, to language files elsewhere on the server. As we extract attributes and hierarchical descriptions and their IDs to separate language files, we store only integer values to dimensional tables. Descriptions of attributes and hierarchies would be associated with relevant IDs from the dimensional tables during report or query execution (on the fly), depending on the default language or language selected. This concept has numerous benefits compared with conventional methods of enabling multilingualism in Business Intelligence environment. Working only with integers makes SQL/MDX queries faster, reduces table size and storage requirements (Heflin and Pan, 2010; Smith, 2012).

Initially, the end user sees an *Initial Filter Page* (ivp.file) that enables filtering of the content from data marts to be presented in the *Reporting Page* (rep.file). *Initial Filter Page* uses a default language set-up via *Global Variables* and *Language Content Pages* (steps a, b, c and d). The application sends an SQL/MDX request defined using *Initial Filter Page* to the data mart (step 1). The first results set is than sent to *Check and Define Page* (cdp.file), which takes it and defines attribute and hierarchical numerical values from dimensions as language variables (step 2). The second result with variabilized values is then sent to *Reporting Page* (rep.file) (step 3). *Reporting Page* communicate with *Global Variables* and *Language Content Page* using header (steps 4 and b), thus it reads the relevant configuration settings for the language before taking the second result set. After taking the second result set, *Reporting Page* reads language values for attributes and hierarchies from *Language Content files*, assigns them and displays final Result Set in the form of business readable report (steps 5 and c).

Using this approach, there is no need to re-execute the SQL/MDX query to change the preview language – we need only to read new language values from the *Language Content files*. As there is no need for new query execution, switching between languages would be easy and fast.

The manipulation of the language content would be easier for business users, as this could be supported by additional tools or modulesthat enable manipulation of textual descriptions saved in textual files. In this context, we consider the idea from Arefin, Marimoto and Yasmin (2011) that efficient Content Management System (CMS), or better Multilingual CMS (MCMS), would help us to overcome the technical limitations of multilingual content in semantic web. In the proposed solution, language files are part of the Warehousing layer, but content is manageable by MCSM.

Adding new languages for semantic web interface would not be dependent on the languages that exist in the source system. If MCSM is implemented so that it has a backend and a frontend for example, we could

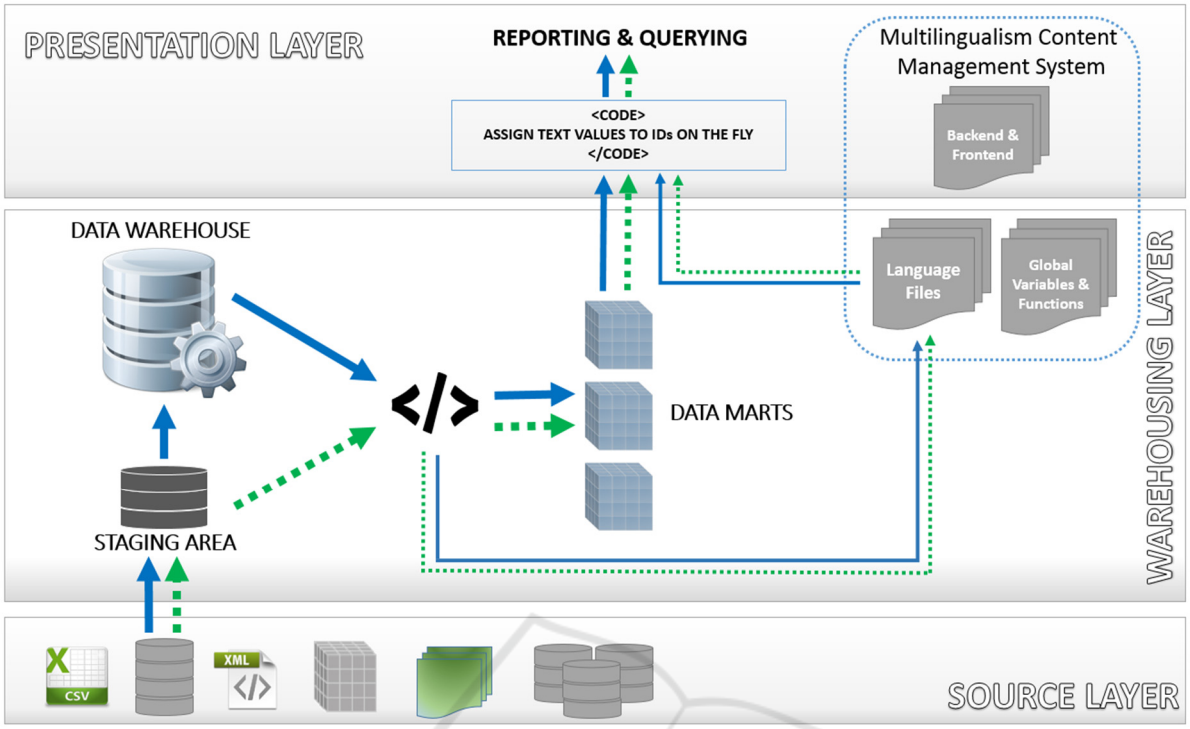

Figure 1: The concept underlying the proposed approach based on three-layered BI architecture.

define new languages through the MCSM backend. This would allow the business user to change descriptions for languages as required.

As it would be possible for business users to change descriptive content directly and by themselves, the ETL process required to perform language changes would be simplified or in some cases eliminated. In conventional solutions, the whole ETL process may be performed just to change a small descriptive value for the specified master data. This is unnecessary in our proposed solution although it would be highly recommended that values should be changed in source systems as well to reflect corrections made in language files. Otherwise, if there is a future need to load whole master data for specific dimension, it would overwrite the corrections made. We recommend using language files only for smaller language corrections and executing standard ETL process when dealing with three or more corrections at the same time.

This approach simplifies the manipulation of textual changes which would be performed outside the values stored in dimensions. This is especially important with dimensions which are typically large such as Article, Product or Customer as there would be no need to re-aggregate existing data according to textual descriptions of attributes or hierarchies. This would allow a more optimal use of database memory as there would be no duplicated data descriptions in dimensions.

Our proposed approach was implemented in a proof-of-concept (PoC) artefact. We had one fact table for Sales data, one dimension table for Product Categories data, and one language file holding Product Categories data.The fact table had Product Category ID as primary key and Price and Amount as measures. Product Category dimension had Product Category ID used for relation with fact table and a Description field holding descriptions of categories. The language file had the same descriptive data as the Product Category dimension. The PoC was tested with 107.768 records in the fact table, 97 records in Product Category dimension table and 97 records for Product categories as language file. For testing purposes, we wanted to show total price per category in our report. In our first PoC method, we multiplied Amount and Price from fact table per Product Category ID and assigned descriptions from language file during report execution - to present the names (descriptions) of categories shown. Using a second method we made the same multiplication, however, we used the names (descriptions) of categories from Product Category dimension and assigned them result set using SQL JOIN.

Results from the PoC showed enviable improvement in performance when using language

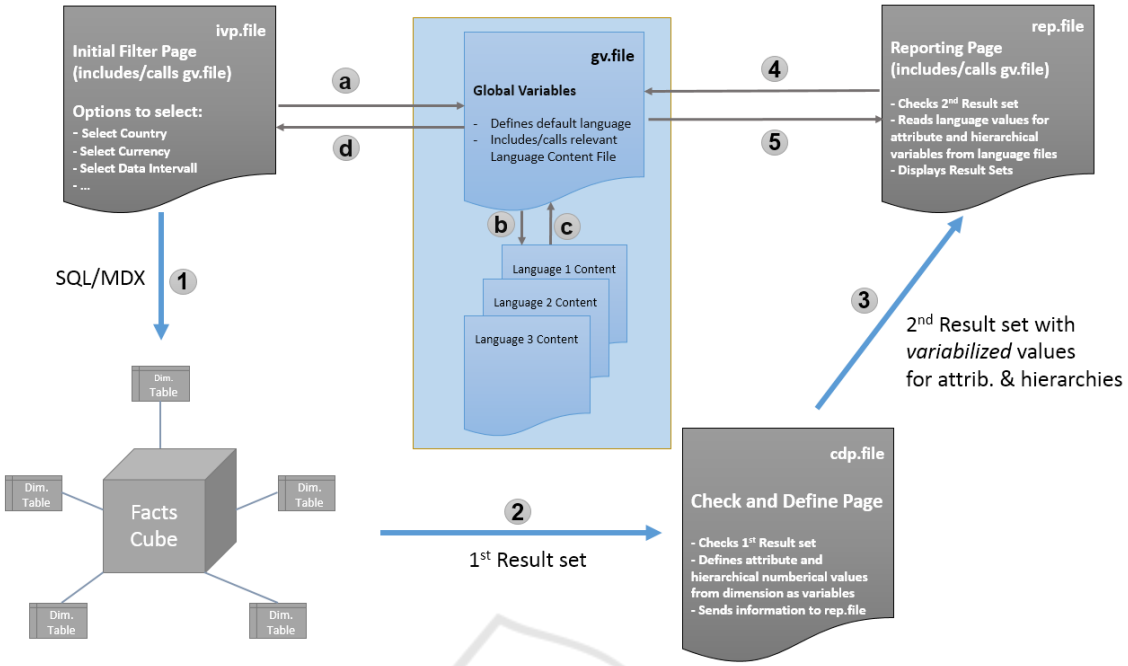

Figure 2: Detailed description of technical functionality of the proposed approach.

files. The average execution time for a report that sums the product sales for 97 different categories using language files method was seven times faster than using conventional methods of implementing star schema. We executed PoC report 228 times: 139 times for method based on language files and 89 times for other method. Using the language files approach we could perform smaller descriptive language changes quickly (simultaneously in data mart and language files). The approach was also less expensive in terms of memory. As this approach can be implemented in the form of add-on to existing monolingual DW structure, it would be optimal way to enable multilingualism in existing BI environment.

4.3 Future Work

The findings from the PoC artefact were encouraging and provide the basis for our future work which will test our solution in a real world environment, as part of a MCSM. Future challenges include integrating multilingualism into the Business Intelligence framework and addressing migration issues as businesses move from single language to multilingual business intelligence systems.

REFERENCES

Anadiotis, G., 2013. Agile business intelligence: reshaping the landscape. , p.3.

Arefin, M.S., Morimoto, Y. & Yasmin, A., 2011. Multilingual Content Management in Web Environment. In *2011 International Conference on Information Science and Applications*. pp. 1–9.

Boakye, E.A., 2012. From Design and Build to Implementation and Validation. In C. McKinney, ed. *Implementing Business Intelligence in Your Healthcare Organization*. Chicago: HIMSS, pp. 73–86.

Brannon, N., 2010. Business Intelligence and E-Discovery. *Intellectual Property & Technology Law Journal*, 22(7), pp.1–5.

Cenoz, J., 2013. Defining Multilingualism. *Annual Review of Applied Linguistics*, 33, pp.3–18.

Chaudhuri, S., Dayal, U. & Narasayya, V., 2011. An overview of business intelligence technology. *Communications of the ACM*, 55(8), p. 88–98 .

Cios, K.J. et al., 2007. *Data Mining: A Knowledge Discovery Approach* 1st ed., New York, USA: Springer Science+Business Media, LLC.

Collins, R.W., 2002. Software localization for internet software: Issues and methods. *IEEE Software*.

Corr, L. & Stagnittno, J., 2014. *Agile Data Warehouse Design: Collaborative Dimensional Modeling, from Whiteboard to Star Schema*, Leeds: DecisionOne Press.

Cravero, A. & Sepúlveda, S., 2015. Using GORE in Data Warehouse: A Systematic Mapping Study . *Latin America Transactions, IEEE (Revista IEEE America Latina)* , 13(5), pp.1654 – 1660.

Croatian Government, 1990. The Constitution of the Republic of Croatia. , p.Article 12.1. .

Dekkers, J., Versendaal, J. & Batenburg, R., 2007. Organising for Business Intelligence: A framework for

aligning the use and development of information. In *BLED 2007 Proceedings*. Bled, pp. 625 – 636.

Dokeroglu, T., Sert, S.A. & Cinar, M.S., 2014. Evolutionary Multiobjective Query Workload Optimization of Cloud Data Warehouses . *The Scientific World*, 2014(14), pp.1 – 16.

European Comission, 2008. *Final Report: Commission of the European Communities High Level Group on Multilingualism*, Luxembourg.

Forbes, 2015. The World's Biggest Public Companies. *Forbes.com*. Available at: http://www.forbes.com/global2000/list/ (Accessed November 24, 2015).

Gouvernement de la République française, 1958. Constitution of France. , p.Article 2.1. .

Government of Federation of Bosnia and Herzegovina, 1995. Federation Constitution. , p.Article 6.1.

Gracia, J. et al., 2012. Challenges for the multilingual Web of Data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 11, pp.63–71.

Graefe, G. et al., 2013. Elasticity in cloud databases and their query processing. *International Journal of Data Warehousing and Mining*, 9(2), p.1.

Hannula, M. & Pirttimäki, V., 2003. Business intelligence empirical study on the top 50 Finnish companies. *Journal of American Academy of Business*, 2, pp.593–599.

Hau, E. & Aparício, M., 2008. Software Internationalization and Localization in Web Based ERP. In *SIGDOC '08 Proceedings of the 26th annual ACM international conference on Design of communication*. New York: ACM , pp. 175 – 180.

Heflin, J. & Pan, Z., 2010. Semantic Integration: The Hawkeye Approach. In *Semantic Computing*. Haboken, New Jersey, US: IEEE Press; John Wiley & Sons, pp. 199–227.

Hensch, K., 2005. IBM History of Far Eastern Languages in Computing, Part 1: Requirements and Initial Phonetic Product Solutions in the 1960s. *IEEE Annals of the History of Computing*, 27(1), pp.17–26. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1401743.

Hillier, M., 2003. The role of cultural context in multilingual website usability. *Electronic Commerce Research and Applications*, 2(1), pp.2–14. Available at: www.elsevier.com.

Huang, S. & Tilley, S., 2001. ssues of content and structure for a multilingual web site. In *Proceedings of the 19th annual international conference on computer documentation*. ACM, pp. 103–110.

Huff, A., 2013. Big Data II: Business Intelligence - Fleets Analyze Information from several sources to improve overall performance. *Commercial Carrier Journal*, (April), p.53.

Imhoff, C., Galemmo, N. & Geiger, J.G., 2003. *Mastering Data Warehouse Design: Relational and Dimensional Techniques*, Indianapolis: Wiley Publishing, Inc.

Inmon, B.W., 2005. *Building the Data Warehouse* 4th ed., Indianapolis: John Wiley & Sons.

Inmon, B.W., 1995. *Building the Data Warehouse* 1st ed., New York: Wiley.

Jensen, C.S., Pedersen, T.B. & Thomsen, C., 2010. *Multidimensional Databases and Data Warehousing* 1st ed., Morgan & Claypool Publishers.

Jovanovic, V., Subotic, D. & Mrdalj, S., 2014. Data Modeling Styles in Data Warehousing. In *37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. Opatija, Croatia: MIPRO, pp. 1458 – 1463.

Kimball, R., 2001. Design Tip #24: Designing Dimensional In A Multinational Data Warehouse. *Kimball Group*. Available at: http://www.kimballgroup.com/2001/06/ design-tip-24-designing-dimensional-in-a-multinational-data-warehouse/ Accessed December 1, 2015).

Kimball, R. et al., 2008. *The Data Warehouse Lifecycle Toolkit* 2nd ed., Indianapolis: John Wiley & Sons.

Kimball, R. & Ross, M., 2011. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling* 2nd ed., John Wiley & Sons.

Kingdom of Spain, 1978. Spanish Constitution. , p. Article 3.1.

Law No. 482, 1999. Norme in materia di tutela delle minoranze linguistiche storiche. *Gazzetta Ufficiale*, 297 (Article 1.1).

Linstedt, D., Graziano, K. & Hultgren, H., 2010. *The Business of Data Vault Modeling* 2nd ., Lulu.com.

Marchand, M. & Raymond, L., 2008. Researching performance measurement systems: An information systems perspective. *International Journal of Operations & Production Management*, 28(7), pp.663 – 686.

Obeidat, M. et al., 2015. Business Intelligence Technology, Applications, and Trends. *International Management Review*, 11(2), pp.47–56.

Olszak, C. & Ziemba, E., 2007. Approach to Building and Implementing Business Intelligence Systems. *Interdisciplinary Journal of Information, Knowledge & Management*, 2, pp.135–148.

Orlov, V., 2014. Data Warehouse Architecture: Inmon CIF, Kimball Dimensional or Linstedt Data Vault? *WMP Blog*. Available at: http://blog.westmonroepartners.com/data-warehouse-architecture-inmon-cif-kimball-dimensional-or-linstedt-data-vault/ (Accessed February 20, 2015).

Ponniah, P., 2004. *Data Warehousing Fundamentals: A Comprehensive Guide for IT Professionals*, New Yor: John Wiley & Sons.

Poolet, M.A., 2008. Data Warehousing: Rapidly Changing Monster Dimensions. *SQL Server Pro*. Available at: http:// sqlmag.com/database-administration/data-warehousing-rapidly-changing-monster-dimensions (Accessed November 24, 2015).

Popovič, A., Turk, T. & Jaklič, J., 2010. Conceptual Model of Business Value of Business Intelligence Systems. *Management: Journal of Contemporary Management*, 15(1), pp.5–29.

Purba, S., 1999. *Handbook of Data Management* 3rd ed., Boca Raton, FL, US: Auerbach, CRC Press.

Sano, M. Di, 2014. Business Intelligence as a Service : a new approach to manage business processes in the

Cloud. In *2014 IEEE 23rd International WETICE Conference*. Parma, pp. 155–160.

SAP AG, 2015. SAP Library - XML: BW - Data Warehousing - Modeling. Available at: http://help.sap.com/saphelp_snc70/helpdata/en/4c/89dc37c7f2d67ae10000009b38f889/frameset.htm (Accessed March 2, 2015).

Scrapehero, 2015. How many products does Walmart.com sell in comparison to Amazon.com? *Scrapehero.com*. Available at: http://learn.scrapehero.com/how-many-products-does-walmart-com-sell-vs-amazon-com/ [Accessed November 24, 2015].

Sen, A. & Sinha, A.P., 2005. A comparison of data warehousing methodologies. *Communications of the ACM*, 48(3), pp.79–84.

Smith, P., 2012. *Professional Website Performance: Optimizing the Front-End and Back-End*, Indianapolis, USA: John Wiley & Sons.

Vázquez, S.R., 2013. Localizing Accessibility of Text Alternatives for Visual Content in Multilingual Websites. In *ACM SIGACCESS Accessibility and Computing*. ACM, pp. 34–37.

Yrjö-Koskinen, P., 1973. Johto – Tiedontarve – Tiedonhankinta. Miten informaatiopalvelu voi palvel la yrityksen johtoa? *Publications of Insinöörijärjestö*, No.77 - 73.