ANNALS OF THE ORADEA UNIVERSITY
Fascicle of Management and Technological Engineering
ISSUE #1, MAY 2013, http://www.imtuoradea.ro/auo.fmte/

IMPLEMENTING A RELATIONAL DATABASE TO SURVEY CARS' RELIABILITY EXPLOITED IN ROMANIA

Lucian ROMAN[1], Adrian FLOREA[2], Ileana Ioana COFARU[3]
[1]Industrial Engineering and Management Department, lucian.roman@autohaushuber.ro
[2]Computer Science and Electrical Engineering Department, adrian.florea@ulbsibiu.ro
[3]Industrial Engineering and Management Department, ioanalogistictop@gmail.com

***Abstract*—**In this work, starting from Microsoft Excel documents we developed a crawler module and we designed a relational database that supports time analysis of defects cars. By highlighting common patterns present in car service and parts production failure, and through their intelligent analysis may result important information regarding to cars' reliability and maintainability. Experimental research was to study the behavior in operating a well-determined period of time of Opel cars and data collection from invoice services documents. With the help of this software application it can gather huge history information from any car service indexing files system and, through further investigations, the manager obtains an extremely agile understanding of the common malfunctions that a car system can suffer and even point out to parts producer, patterns that appear in their design.

***Keywords*—**Database, Data mining, Maintainability, Materials reliability.

I. INTRODUCTION

THE main aim of this work is to implement a relational database to survey cars' reliability exploited in Romania. By emphasizing certain patterns (structures / spare parts / maintenance services) that frequently occur in the automotive services and through their intelligent analysis may result important information regarding to cars' reliability and maintainability. This will improve the quality of spare parts production through focusing on specific directions depending on the geographic area, the infrastructure of the region, environmental conditions, characteristics of fuels, etc.

The scientific approach is based on *Data Mining* - an interdisciplinary field of computer science that aims to discover patterns in large data sets. Methods involved are at the intersection of artificial intelligence, machine learning, statistical and database management systems. As we detailed further in section III paragraph C, we developed in Microsoft Visual Studio 2012 (using C# programming language), .NET Framework, using Microsoft SQL Server 2008, a software application that extract the relevant data from archive of orders with maintenance service operations, automotive components and spare parts. Further, these commands will be called briefly invoice services (IS). The application's input consists in information from any car service indexing files system. Currently, these information are office documents of Microsoft Excel and Microsoft Word types. The purpose of the application (output) is to design, implement and populate a relational database with extracted data. As more data are extracted as more accurate will be the analysis. At present, the database is customized to serve at AutoHaus Huber SRL Sibiu, OPEL dealer, but with small modifications it could be extended to any auto service from Romania or outside.

The present application is extremely useful because the processing and interpretation of data extracted obtain a fairly accurate understanding of the common faults that may occur in operating a motor vehicle, may determine the causes of breakdowns, can identify abnormal wear. Also, it can track mode troubleshooting and even it can suggest to manufacturer the templates that appear in spare parts design. This type of analysis is called "*Business intelligence*" and it is becoming one of the key factors in planning marketing strategies. With this pattern recognition ability the management team can easily take decisions that in the past were consider risky and dependent on the manager skills on the subject.

The organization of the rest of this paper is as follows. In section II we shortly review the challenges regarding the cars' reliability exploited in Romania, emphasizing on OPEL vehicles. Section III represents the centerpiece of this work and describes the Database Architecture. It starts with a theoretical background regarding Database design, then it introduces the developed three-tier architecture, it details the business layer concept and finishes with Database mapping, presenting the data structures and relations between tables. Section IV suggests future work directions and concludes the paper.

II. CHALLENGES IN THE VEHICLES RELIABILITY

Reliable system design is concerned with making systems work even in the face of internal or external

problems. In the automotive domain reliability refers to designing a car that is safe. In this sense it should be developed applications that control real-time vehicle. The modern automobiles are complex technical systems which require control systems that are physically distributed around the vehicle (15 to 20 base aggregate subsystems with different functional responsibilities and mechanisms). Networks have been designed specifically to meet the needs of real-time distributed control for automotive components. The basic fact that drives the design of control systems for vehicles is that they are safety-critical systems. Errors of any kind - component failure, design flaws, exceeding the number of kilometers that have made technical revision, etc. - can injure or kill people. Not only must these systems be carefully verified, but they also must be architected to guarantee certain properties [1].

In the operation process and even in downtime period of vehicles occurring phenomena leading to worsening functional indicators and partial or total loss of functional capacities and performance. Preservation in time the whole system performance leads to vehicle reliability. An important parameter that defines cars reliability is the Reliability Index [2], a score that is calculated as a combination of the number of times a car fails, the cost of repairing it, the average amount of time it spends off the road due to repairs, the age of the vehicles. Influencing factors resulted in operating conditions can be related on the one hand the quality of the various materials used to maintain the functionality of the vehicle such as lubricants, fuels, liquid cooling system, brakes, actuators, etc., and on another part of the travel and transport conditions highlighted by road and climatic qualities, operating system, quality management and others. During vehicle operation are continuous interaction between parts, mechanisms and materials and operating units. Depending on their properties, it speeds up or slows down the degradation of parts and depositing the material residues, with secondary influence on the aggregates' functioning, it changes the consumption of operating material and vehicles productivity. Adopting of operating materials must comply constructive and technological peculiarities of motor vehicles, their technical status and operating conditions. Basic factors of operating conditions that influence reliability and durability are the road conditions, weather conditions, operating mode, quality management, quality maintenance and quality auto repair. Due to operating conditions, the car manufacturers differentiate the period in which they perform maintenance revision, depending by country, geographic area, etc. [3].

III. DATABASE ARCHITECTURE

A. Explaining the basic concepts

Because our paper has an interdisciplinary character applying domain-specific tools computer science in industrial engineering, specifically road vehicles and transportation engineering, we start this section by explaining some main concepts regarding databases. The first step in order to implement a relational database that supports time analysis of defects cars is data collection and analysis and implementation of a conceptual model. At this stage are considered nature and the use of data. Identified data will be stored and processed, and are divided into logical groups and establishes relationships between groups. This stage should be according with client's requests (in this case OPEL). The most important feature in database design is the normalization [4]. This technique eliminates / avoids certain anomalies and data inconsistencies. Thus, the data should not be redundant and data manipulation operations (update / insert / delete) must ensure database integrity. A design tool of conceptual model of relational databases is Entity Relationship Diagram (ERD). An ERD developed during the conceptual data modeling phase of the database development process is generally transformed and enhanced through normalization principles during the logical database design phase. The next design step is to transform conceptual model into logical schema of data model. The database model is not just a way of structuring data, but also defines a set of operations that can be performed with the data and also defines a set of rules in order to keep database integrity. One of the most common data models, and which we used in our application is the relational (due to the simplicity of data types, text only). The data is organized as tables, there are relationships between them. Relational model based on relational algebra made possible development of relational languages as software that assists the implementation of databases. One such language used in this application is SQL (Structured Query Languages).

B. The proposed three-tier architecture

We use 3-tier architecture in our software application. Commonly, 3-tier architecture consists of the following three tiers (levels) [5, 6]:

- *Presentation tier* as known as *Frontend*

  This is the topmost level of the application. It is the layer which users can access directly such as a web page or application GUI (graphical user interface). Through the presentation tier the user / client ask for information (arrow 1 in fig. 1) such “Get the most replaced component?” or “Which is the most performed operation to a certain car?” It communicates with other architectural tiers in order to output, finally, the text or graphical results (arrow 6 in fig. 1).
- *Logical (Application) tier* that includes *Business layer* and *Data access layer*, as known as *middleware*

  The logical tier is pulled out from the presentation tier and, it controls an application’s functionality by performing detailed processing based on set of rules. The Business layer includes the *Crawler* module that is responsible for data mining in office documents, parsing, computing, and synchronizing. The extracted data are collected and indexed. With their help, in the Data access layer (software implemented in the *Database* module), are conceived and designed the

conceptual scheme of database, tables and relationships (arrow 2 in fig. 1). This layer manages accesses to database by means of SQL, transforming customer requirements from the Presentation layer in database queries (arrow 3 in fig. 1). After Database tier provide the requested data these are forwarded back to the Presentation layer (arrow 5 in fig. 1).

- *Database tier* as known as *Backend*

  This tier is the physical storage layer, responsible for data persistence and consists of database servers. Here information that comes from processed data during the Logical layer is stored and retrieved and forwarded back to Logical layer (arrow 4 in fig. 1). This tier keeps data neutral and independent from previously two tiers. Giving data its own tier also improves scalability and performance.

By splitting the application's architecture into 3 logical layers - presentation, application and database access, are improved the development efforts, allowing re-usability and extensibility as much as possible, increase maintainability and by division of work ensure a faster development. The idea is to reduce the overall maintenance and increase the scalability of your application.

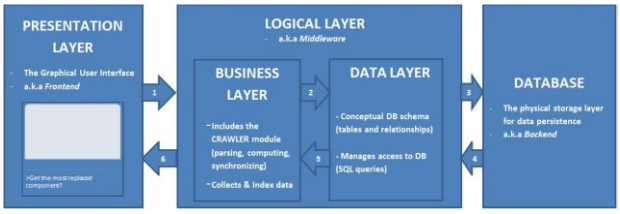


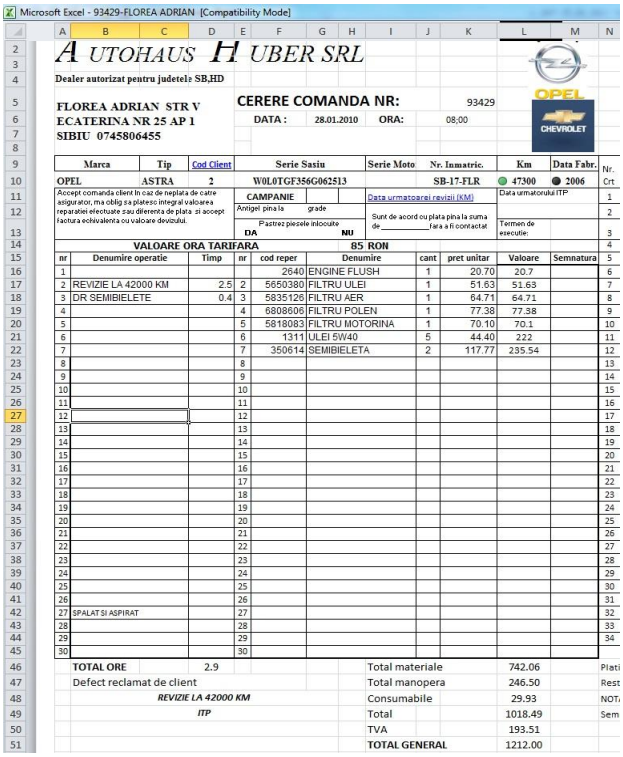
Fig. 1. The three-tier architecture

C. Description of business layer concept

Starting from the guidelines found in [7] we developed a 3-tier database architecture. The application's input are office documents of Microsoft Excel type that represent the invoice services collected for last 5 years from AutoHaus Huber SRL Sibiu, OPEL dealer.

The key aspect of this software implementation is the ability to gather information as easy as it can. This is due to the fact that larger history means higher accuracy rate. So to be able to gather information from all possible car services providers means we need an intelligent document template interpreter. This points our first software module *Crawling Module*.

"Crawler" is a generic term for any program (hardware - such as a robot or software - web application), used to automatically discover and scan new locations / sites by following links stepper or links from a Web page to another. In this case, the application requires such a module because every day, at every vehicle checked / repaired, new data appear in the system and even new pattern of faults. As shown in fig. 1, the Crawling module provides the interface with actual data present on client computers (in this case, OPEL Company). This module is responsible for collecting these data, index them (assigning indexes for different spare parts / operations, etc.) and then forward them to the next module, Database. Some of these data are field names in the developed tables and others will be field values.



Microsoft Excel - 93429-FLOREA ADRIAN [Compatibility Mode]

AUTOHAUS HUBER SRL

Dealer autorizat pentru județele SB,HD

FLOREA ADRIAN STR V ECATERINA NR 25 AP 1
SIBIU 0745806455

CERERE COMANDA NR: 93429
DATA: 28.01.2010 ORA: 08:00

Marca	Tip	Cod Client	Serie Sasiu	Serie Moto	Nr. Inmatric.	Km	Data Fabr.
OPEL	ASTRA	2	W0L0TGF356G062513		SB-17-FLR	47300	2006

Accept comanda client in caz de neplata de catre asigurator, ma oblig sa platesc integral valoarea reparatiei efectuate sau diferenta de plata si accept factura echivalenta cu valoare devizului.

CAMPANIE
Antigel pina la grade
Plastrez piesele inlocuite
DA NU

Data urmatoarei revizii (KM)

Sunt de acord cu plata pina la suma de _____ fara a fi contactat

Data urmatorului ITP

Termen de executie:

VALOARE ORA TARIFARA 85 RON

nr	Denumire operatie	Timp	nr	cod reper	Denumire	cant	pret unitar	Valoare	Semnatura
1			1	2640	ENGINE FLUSH	1	20.70	20.7	
2	REVIZIE LA 42000 KM	2.5	2	5650380	FILTRU ULEI	1	51.63	51.63	
3	DR SEMIBIELETE	0.4	3	5835126	FILTRU AER	1	64.71	64.71	
4			4	6808606	FILTRU POLEN	1	77.38	77.38	
5			5	5818083	FILTRU MOTORINA	1	70.10	70.1	
6			6	1311	ULEI 5W40	5	44.40	222	
7			7	350614	SEMIBIELETA	2	117.77	235.54	
27	SPALAT SI ASPIRAT								

TOTAL ORE	2.9	Total materiale	742.06
Defect reclamat de client		Total manopera	246.50
REVIZIE LA 42000 KM		Consumabile	29.93
ITP		Total	1018.49
		TVA	193.51
		TOTAL GENERAL	1212.00

Fig. 2. The invoice services document

In such real-time application, the synchronization and data consistency are very important. An issue consists in dynamically configuring of moment when the files are parsed and extracted dates, and then are inserted into database. Crawling module creates a hierarchy of files and ads in each folder a file with the extension *.nrd* used for indexing and synchronization.

In the process of normalization of data from Business layer, we face some integrity problems. The first error was due to car brand - "Marca" field from "Autoturism" table that does not correspond with the car driven series chassis [8]. The second was due to incomplete synchronization of data. Some spare parts were assigned to "*orders*" in "ListaPiese" table that were not registered (yet) on invoice services (in "Comanda" table). The first error was eliminated by Crawler module implementation in a chassis code checker which automatically corrects and updates the "Marca" field. With our implemented solution for auto correction, we reduced by 70% the erroneous data. The second error was treated by resynchronization of data (after short period of time) and, if the error still persisted, these records were removed from the database. Another problem that has required attention in order to preserve the integrity of database consisted in aligning data text variables in *varchar* fields of databases tables and removing empty spaces from field names. Such situations may be encountered due to

negligence of human operators who place wrong values or, leave empty spaces on invoice services.

D.Database mapping

After having data crawled and ready for storage there was needed a mapping structure in the database taking into account the obtained data and their type. The following tables are currently mapped in the database: *Autoturism, Comanda, ListaOperatii, ListaPiese, Piesa.* Figure 3 shows the implemented database structure, the component tables and main fields, primary keys, etc.

Database module is responsible for designing the conceptual scheme of database and mapping, performing database queries, being able to handle huge bulks of data. It is interfaced with the Crawling module and adds data in safe location data storage to be accessed later by all users. This way we connect our local extracted data to the global database (Database tier) thus creating a data share point.

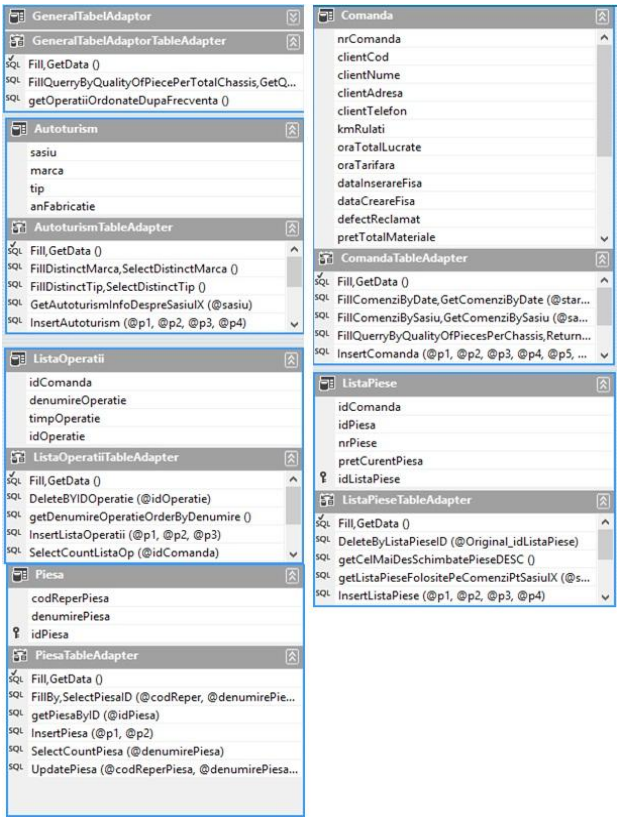


Fig. 3. Database tables – C# view

These data mappings tables are created for a single file template, currently an Excel file. With the use of these tables, a data visualization module that we will implement as further work, will have enough available data for further management and marketing analytics.

As a short explanation of fig.4, basically there is a unique code (primary key) of car chassis ("sasiu" field in "Autoturism" table) but in "Comanda" table can find many records with the same car chassis (as the same car was repaired on several times using different invoice services). The link is named *1 to many* relationship.

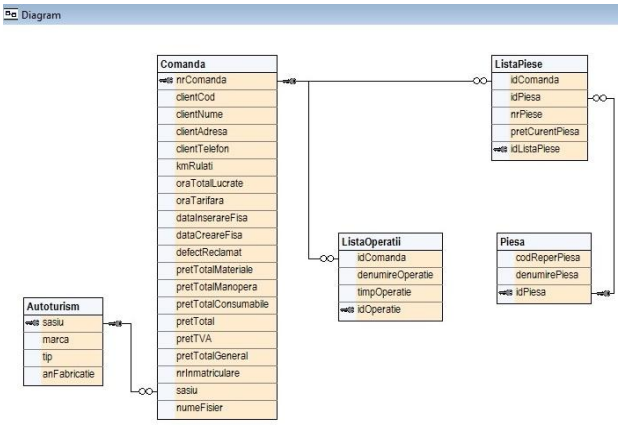


Fig. 4. Relationships between database tables

IV. CONCLUSIONS AND FURTHER WORK

In this work, starting from Microsoft Excel documents we developed a C# crawler module and we designed and populate a relational database that supports time analysis of defects cars. We removed some database integrity problems which were caused by employees that completed wrong some invoice services or due to later or no synchronization. The usefulness of application developed consists in efficiency interpreting of extracted data. These information could provide a fairly accurate understanding of the common faults that may occur in operating a motor vehicle, may determine the causes of breakdowns, can identify abnormal wear.

We intend to continue working on this application bringing new features, and to exploit the created database, in order to identify the main faults and the possibility of optimizing the operating conditions in Romania. Also, experimental validation proposals for improving technical quality using some computer-aided design tools we plan.

REFERENCES

[1] W. Wolf, "High-Performance Embedded Computing: Architectures, Applications, and Methodologies", Morgan Kaufman, 2006, ch. 1.

[2] www.reliabilityindex.com/ (accessed 15 March 2013)

[3] T. Nagy, M.A. Stănescu, N. Turea, D. Dima. "Reliability and tero-technology of vehicles" (in Romanian), Vol I, "Transilvania" University of Brasov, 1999.

[4] T.M. Connolly, C.E. Begg, "Database Systems: A Practical Approach to Design, Implementation and Management", 5th Edition, Addison-Wesley, 2009.

[5] A. Tarhini, "Concepts of Three-Tier Architecture", 2011, http://alitarhini.wordpress.com/2011/01/22/concepts-of-three-tier-architecture/ (accessed 15 March 2013).

[6] M. Fowler, D. Rice; M. Foemmel; E. Hieatt; R. Mee; R. Stafford "Patterns of Enterprise Application Architecture", Addison-Wesley Professional, 2002, ch. 1.

[7] M. Sarvade, "Implementing 3-tier Architecture in C# .net", Indian Streams Research Journal, Vol. 1, Issue 1 / February 2011, pp.173-176.

[8] http://www.mitchellsupport.com/ondemand5/VIN/VIN.pdf (accessed 10 April 2013)