Database Schema Elicitation to Modernize Relational Databases

Ricardo Pérez-Castillo[1], Ignacio García Rodríguez de Guzmán[1], Danilo Caivano[2] and Mario Piattini[1]
[1]*Instituto de Tecnologías y Sistemas de Información (ITSI), University of Castilla-La Mancha, Paseo de la Universidad 4, 13071, Ciudad Real, Spain*
[2]*Department of Informatics, University of Bari, Via E. Orabona, 4, 70126 Bari, Italy*

Keywords: Database Modernization, Legacy Systems, ADM, KDM, SQL, Metamodel, Model Transformations, QVT.

Abstract: Legacy enterprise systems mainly consist of two kinds of artefacts: source code and databases. Typically, the maintenance of those artefacts is carried out through re-engineering processes in isolated manners. However, for a more effective maintenance of the whole system both should be analysed and evolved jointly according to ADM (Architecture-Driven Modernization) approach. Thus, the ROI and the lifespan of the legacy system are expected to improve. In this sense, this paper proposes the schema elicitation technique for recovering the relational database schema that is minimally used by the source code. For this purpose, the technique analyses database queries embedded in the legacy source code in order to remove the dead parts of the database schema. Also, this proposal has been validated throughout a real-life case study.

1 INTRODUCTION

Today, many organizations have huge legacy systems supported by relational databases (Blaha, 2001), and these systems are not immune to software ageing (Visaggio, 2001). Nevertheless, the erosion not only affects the source code, but also databases age gradually. For instance, in order to adapt the system to new requirements, new tables and/or columns are added to the schema of the database; other tables are modified and even discarded without erasing them from the database; and so on. These changes over time generate problems related to inconsistency, redundancy and integrity among others.

Therefore, organizations must address maintenance processes in their legacy information systems. The entire replacement of these systems has a great impact in technological and economic terms (Sneed, 2005). So that, maintenance based on evolutionary reengineering processes has typically been carried out (Bianchi et al., 2003). Moreover, the typical re-engineering process has been shifted to the so-called *Architecture-Driven Modernization* (ADM) (Ulrich and Newcomb, 2010) in the last years. ADM advocates carrying out re-engineering process following the principles of model-driven development: taking into account all involved artefacts as models and implementing transformations between them.

The increasing cost of maintaining legacy systems along with the need to preserve business knowledge has turned the modernization of legacy systems into an important research field (Lewis et al., 2010). ADM provides several benefits such as ROI improvement to existing information systems, reducing development and maintenance cost and extending the life cycle of the legacy systems.

This paper proposes the elicitation of database schemas, a reverse engineering technique that follows the model-driven principles to recover a minimal relational schema from the queries embedded in the legacy source code. The reverse engineering technique to elicit relational database schema consists of two key stages:

(i) *Static analysis of source code*, which examines the Legacy source code and looks for SQL (Structure Query Language) statements embedded in the code. In this task, a model of SQL sentences is built according to a metamodel of DML (Data Manipulation Language) of SQL.

(ii) *Model transformation*, which obtains a model of the relational schema from the previous SQL sentences model. This transformation is based on a set of rules that elicits the minimal schema of the underlying database.

The objective of this proposal is to discover the minimal subset of relational elements of the legacy

Pérez-Castillo R., García Rodríguez de Guzmán I., Caivano D. and Piattini M..
Database Schema Elicitation to Modernize Relational Databases.
DOI: 10.5220/0003980801260132
In *Proceedings of the 14th International Conference on Enterprise Information Systems* (ICEIS-2012), pages 126-132
ISBN: 978-989-8565-10-5

database in order to improve the database to use it in the ADM process. Thus, the proposed technique rebuilds the database schema and removes the dead parts based on the embedded SQL sentences. This mechanism eradicates duplicated or unused tables, removes unused columns, and discovers new referential constraints. In this way, the software of legacy systems does not evolve in an isolated manner, but takes into account its relational databases.

The remainder of this paper is organized as follows. Section 2 summarizes related works in the database reengineering field as well as the state-of-the-art about ADM. Section 3 explains the proposed schema induction technique in detail. Section 4 presents a case study to validate the proposal. Finally, Section 5 addresses the conclusions of this work.

2 STATE-OF-THE-ART

2.1 Related Work

Research about re-engineering on applications and databases jointly is usually addressed in literature. *Reus* (Reus et al., 2006) presents a reverse engineering process based on MDA (Model-Driven Architecture) for recovering UML (Unified Modeling Language) models from PL/SQL code. *Fong* (Fong, 1997) and *Ramanathan et al.* (Ramanathan and Hodges, 1997) transform the relational models into object-oriented (OO) models to integrated them with OO applications. Also, *Cohen et al.* (Cohen and Feldman, 2003) convert parts of the application logic from the procedural style of the legacy systems to the declarative style of SQL in order to integrate them with relational databases. *Hainaut et al.* (Hainaut et al., 1996) proposes a reverse engineering process for recovering the design of relational databases and *Wu et al.* (Wu et al., 2008) discover topical structures of relational databases. Moreover, *Pérez-Castillo et al.* (Pérez-Castillo et al., 2009) propose a wrapping technique to extract Web services from relational databases that manage the data access. Finally, *Polo et al.* (Polo et al., 2007) have studied building database-driven applications.

However, related works do not address some key challenges for modernizing relational databases: (1) those works do not follow a model-driven approach in all phases of the reengineering process and (2) recovered database knowledge is not managed in an integrated and standardized manner. ADM is a potential solution for dealing with the first challenge while KDM (Knowledge Discovery Metamodel) enables optimal knowledge management for relational databases within the ADM processes, the second challenge.

2.2 Standards Involved: ADM, KDM and QVT

The reengineering processes and MDA principles converge in ADM, an OMG standard for modernizing legacy systems (Khusidman and Ulrich, 2007). ADM is the concept of modernizing existing systems with a focus on all aspects of the current systems architecture and the ability to transform current architectures to target architectures (Pérez-Castillo et al., 2011).

The *ADM Task Force* in OMG led to several standards (OMG, 2009). The cornerstone of this set of standards is KDM (*Knowledge Discovery Meta*). KDM allows standardized representation of knowledge extracted from legacy systems by means of reverse engineering. In addition, KDM has been recently recognized as the standard ISO 19506 (ISO/IEC, 2009). The KDM metamodel provides a common repository structure that makes it possible to interchange information about software artefacts in legacy systems. KDM makes it possible to represent the PIM models in the horseshoe model. KDM can be compared with the UML standard: while UML is used to generate new code in a *top-down* manner, the ADM processes that involving KDM starts from the existing code and builds a higher level model in a *bottom-up* manner (Moyer, 2009).

The KDM metamodel (ISO/IEC, 2009) is divided into four layers (each one based on a previous layer) representing both physical and logical software assets of information systems at several abstraction levels. This work focuses on (i) *program element layer*, which provide a language-independent intermediate representation for various constructs determined by common programming languages; and (ii) runtime resource layer, which enables representation of high-value knowledge about legacy systems such as databases.

3 SCHEMA ELICITATION

The proposed ADM process is organized into three stages. The first stage of the modernization process aims to obtain, through reverse engineering, a set of PSM models representing each software artefact of

the legacy system. This task involves using a specific metamodel for each artefact. After that, in the second stage, a KDM model is built (using model transformations) from the PSM models recovered from the legacy system. In this case, the KDM model plays the role of a PIM model. Therefore, this model abstracts any technology-specific aspect of the legacy system. It should be borne in mind that obtaining KDM models does not end in the restructuring stage, because it is possible to restructure the KDM model itself. For example, transformations can be tailored between KDM layers. Finally, the forward engineering stage accomplishes building new and improved information systems. This stage involves PSM models to represent specific aspects related to the technological nature of each target system. In this way, the modernization process is completed according to the horseshoe model.

The aim of this modernization process is to modernize legacy systems focused on code and database as exclusive software artefacts. These artefacts undoubtedly determine three KDM models that must be obtained in the reverse engineering stage of this modernization process: (i) *KDM Inventory Model* is based on the *Source Package* of KDM. It enumerates physical artefacts of the legacy system and defines the mechanism of traceability links between all the KDM elements and their original representation in the legacy source code. (ii) *KDM Code Model* supports both *Code Package* and *Action Package*. This model aims to represent program elements and their associations at the implementation level. It includes elements supported by several programming languages such as sentences, operators, conditions, associations, control and data flows. (iii) *KDM Data Model* represents data manipulation in legacy systems. *Data Model* is based on *Data Package* and uses the foundations provided by *Code Model* related to the representation of simple data types. Also, this model can depict the relational databases used by the legacy system.

In addition to these models, the schema elicitation technique involves other models in the reverse engineering stage of this ADM process (see the shaded part of Figure 1). The database schema is elicited from the SQL embedded in the source code by means of the proposed technique, and thus it generates an *SQL Sentences Model* by means of the static analysis of legacy source code. The static analysis activity also produces the *Inventory Model* and *Code Model*. After that, the *Database Schema Model*, a model that represents the minimal schema of the database, is obtained through the model transformation from the *SQL Sentences Model*. Finally, the needed *KDM Data Model* is obtained from the *Database Schema Model*.

At this point, both the source code and the database are represented according to KDM. Therefore the restructuring and forward engineering stages can be carried out in order to generate the modernized version of the legacy systems (see Figure 1).

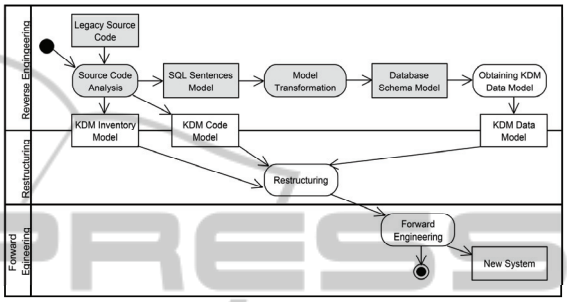

Figure 1: Schema elicitation technique based on ADM.

In order to obtain the *SQL Sentences Model* the technique analyses the legacy source code for embedded SQL sentences by means of a parser. This parser is a syntactical analyser that exhaustively scans source code. When the parser finds an SQL sentence, it translates that sentence into a model according to a metamodel of the DML (Data Manipulation Language) of SQL-92 that has been developed.

The metamodel modeling the syntax of the SQL-92 DML (ISO/IEC, 1992). It can represent the SQL operations such as *Insert*, *Select*, *Update* and *Delete* together with search conditions.

After obtaining the *SQL Sentences Model* through static analysis, the *Database Schema Model* must be obtained by mean of a model transformation. These models of relational database schemas are represented through a metamodel according to the SQL-92 standard (ISO/IEC, 1992). Deductions of the minimal database schema are based on a set of rules developed specifically for this purpose. These rules recover only a subset of the database schema elements that are handled by the SQL sentences embedded in the source code.

Rule 1. The tables that appear in any SQL sentence (Insert, Select, Update or Delete) as either source or target clauses (From, Set, Into, and so on) are created as tables in an induced database scheme. **Rule 2.** The columns that are selected, added, deleted or updated in the SQL sentences are created in the corresponding tables. These tables have

previously been created through the application of Rule 1. **Rule 3.** The columns depicted through the table alias in sentences (As clause) are created in the table related to this alias. This table was created previously by means of Rule 1. **Rule 4.** The data type associated with each column can be deduced through the kind of expressions where these columns appear. For example, Like expressions →'string' data type; arithmetic expressions →'integer, decimal or numeric' data type, and so on. **Rule 5.** The Select sentences structured in Join mode suggest potential primary keys and foreign keys according to the pattern expressed in Figure 2. While the source column(s) of join select is/are related to the column(s) within the foreign key, the target column(s) of join select is/are related to the column(s) within the primary key. **Rule 6.** After applying the previous rules, it is possible that some tables are created without a primary key. In this case, a new column is attached to these tables as its primary key. This column is a sequential number that is generated automatically.

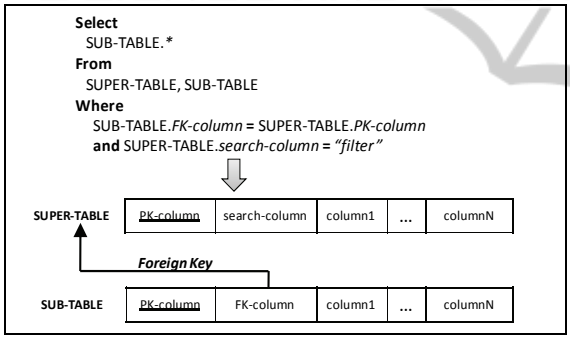

Figure 2: Pattern "join select to foreign key".

4 CASE STUDY

The case study addresses a modernization project that is currently being carried out. The subject legacy system of this project is the intranet of the *Computer Science Faculty* at the *University of Castilla-La Mancha*. This intranet was developed five years ago by several people.

The structure of the intranet consists of five modules: (i) the *Main* module is the major module of the intranet and has the standard functionalities; (ii) the *Administration* module is in charge of administrative and office tasks; (iii) the *Old Students* module manages information related to students who were members of the faculty; (iv) the *Management* module is a module for the setup of the intranet; and finally, (v) the *Quality* module measures and reports on the quality of the faculty.

The intranet has a typical Web architecture separated into three layers: presentation, business and persistence. The technology used to develop the presentation layer was *JSP* (*Java Server Pages*), *JAVA* for business layer and *ORACLE* together with *JDBD-ODBC* for the persistence layer. The total size of this legacy system is 72.68 KLOC.

The case study establishes two research questions to analyse the *Database Schema Models* obtained through the proposal:

Q1. Are the output models complete?

Q2. Are the output models minimal with regards to the original database schema?

The question *Q1* aims to assess the completeness of the obtained database schema. A specific schema is complete when: (i) all tables in this schema model have a primary key; (ii) there are no tables without columns; and (iii) the schema model does not have any duplicated elements. Moreover, the question *Q2* takes into account the minimization of the database schema. The minimization is measured by means of the size of the obtained schema regarding the size of the source database schema. In order to measure the gain between the previous and current sizes, we use two variables: the gain related to the number of tables (1) and the gain related to the number of columns in each table (2). In these formulas, T_0 is the number of tables in the legacy database schema and $C_{0\{Ti\}}$ represents the number of columns of Table i in the legacy database. Moreover, T represents the number of tables in the improved database schema and $C_{\{Ti\}}$ is the number of columns in Table i in the obtained database schema.

$$G_T = \frac{T_0 - T}{T_0} \quad (1)$$

$$G_{C\{T_i\}} = \frac{C_{0\{T_i\}} - C_{\{T_i\}}}{C_{0\{T_i\}}} \quad (2)$$

The execution of the case study was carried out by means of a tool based on the *Eclipse* platform that was developed to support the elicitation schema technique. A QVT (Queries / Views / Transformations) (OMG, 2008) model transformation is tailored in the tool from the proposed rules. The tool accomplishes several *SQL Statements Models*, a model for each source code file. Table 1 summarizes the models obtained from the legacy source code.

After that, the QVT transformations are executed through the tool using these models as input models

to obtain the output model that represents the new and minimal database schema.

Table 1: Input SQL statement models.

Module	Source Files	SQL Statements	SQL Statement Models
Main	18	23	18
Administration	8	14	8
Old Students	4	4	4
Management	1	1	1
Quality	44	60	44
Total	75	102	84

After the execution of the QVT transformations, a set of output models that depicts database segments (used for each module of the intranet) was obtained. Table 2 summarizes the results obtained for each output model. The legacy database had 140 tables and 25 tables were recovered. Table 2 shows the tables recovered for each intranet module. In addition, it presents the gain related to the tables (G_T) as well as the gain regarding the columns (G_C).

The analysis of the results obtained for these models presents several conclusions that should be considered as a response to the question *Q1*: (i) tables are usually obtained without primary keys unless Rule 6 is launch after other QVT relations; (ii) obtaining tables without columns is not usual, because any column that appears in a SQL statement is normally associated with its table.

In this case study, the QVT relations do not infer enough foreign keys, because the only QVT-implemented mechanism for inferring foreign keys is Rule 5. Indeed, the intranet source code has only two *join select* sentences due to the bad design of the legacy database.

In order to respond to the question *Q2*, the gain of the obtained database schema was also assessed. In total, 18% of the tables were recovered (25 tables) and the G_T value was 82%. With respect to the columns, the box diagram in Figure 3 shows the distributions of G_C for each intranet module. The mean per table of the G_C values was 27%, although in some modules this mean was higher. In this study, the G_C mean is lower than the G_T. However, the total gain related to the size minimization of the new database schema is significant.

5 CONCLUSIONS

This paper proposes a modernization process based on KDM. The objective of this process is the modernization of legacy source code together with legacy relational databases. For this reason, this proposal considers two complementary sources of knowledge: (i) the schema of the legacy database, and (ii) the SQL sentences embedded in the legacy source code.

The main contribution of this paper is a mechanism named 'schema elicitation' to rebuild the database schema from the SQL sentences embedded in the source code.

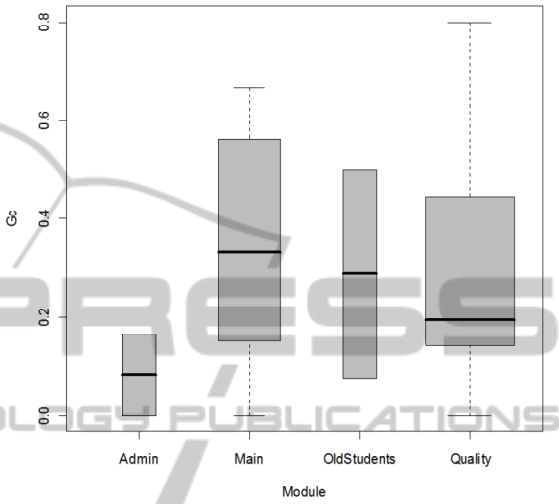

Figure 3: Box diagram of GC per module.

This mechanism removes the dead parts of the database schema such as duplicated or unused tables and unused columns. Also, this mechanism can discover new referential constraints implicit in the source code. Therefore, this proposal obtains improved and minimal database schemas used in the latter stages of the modernization process. In order to support the schema elicitation mechanism, two SQL metamodels were developed: a metamodel for representing SQL sentences embedded in source code as well as a metamodel for modelling schemas for relational databases. In addition, a set of QVT relations was tailored to transform a SQL sentences model into another database schema model.

Finally, a case study with a legacy intranet reports the main advantages of this proposal. Firstly, the obtained database schema had the adequate completeness level, thus that schema can be used as new database schema in the modernized system. Secondly, the dead parts were removed. In our case study the minimization of the size of the obtained schema was around 80% with regard to the original size.

The future extension of this research focuses on the improvement of the completeness level of the database schema models and on the detection more dead parts by means of more refined patterns. For

Table 2: The schema elements recovered through the case study and the gain of the minimal schema.

	Total Tables = 140	ALUMNOSCEP	ANTIGUOSALUMNOS	ASIGNATURA	CEP	CLAVEALUMNOSOLD	CONFERENCIA	CONFIGURACION	CURSOS_CERO	CURSOS_ESP_POSTGR	CURSOS_TECNICAS_ET	DOCUMENTOS	EMPRESA	FECHAS_VISITAS	FORMACION_REGLADA_ASIG	GRAINTS	INFORMES_CALIDAD	MATRICULASCEP	NUMERO_ALUMNOS	PRESUPUESTO_INVESTIGACION	PROYECTO_INVESTIGACION	SEMINARIOS_DI	SUBCURSO_EP	TAREAS	USERS	VISITAS	Primary Keys	Foreign Keys	Tables	G_T
Main	x			x			x				x		x				x							x	x	x	1	1	9	94%
Administration																x									x		0	0	2	99%
Old Students		x			x																						0	0	2	99%
Management			x																								0	0	1	99%
Quality				x		x		x	x	x		x		x		x		x	x	x	x	x					1	1	13	91%
TOTAL	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**2**	**2**	**25**	**82%**
Recovered Cols	4	12	3	9	1	6	1	4	5	6	7	5	4	14	5	6	3	6	4	4	5	6	11	3	1					135
All Columns	4	13	15	28	2	19	2	7	6	7	9	9	6	18	6	6	8	6	4	7	6	7	12	3	3					213
G_C	0	0,1	0,8	0,7	0,5	0,7	0,5	0,4	0,2	0,1	0,2	0,4	0,3	0,2	0,2	0	0,6	0	0	0,4	0,2	0,1	0,1	0	0,7					27% (Mean)

this purpose, more case studies will be carried out in order to detect more information needs in the target database schema that must be obtained. In addition, the future work will address the next stages of the proposed modernization process such as the transformation from database models to KDM models, and then, the restructuring and forward engineering stages, which will use the previous knowledge.

ACKNOWLEDGEMENTS

This work has been supported by the *FPU Spanish Program*; by the R&D projects funded by *JCCM*: ALTAMIRA (PII2I09-0106-2463), INGENIO (PAC08-0154-9262) and PRALIN (PAC08-0121-1374).

REFERENCES

Bianchi, A., Caivano, D., Marengo, V. and Visaggio, G. 2003. Iterative Reengineering of Legacy Systems. IEEE Trans. Softw. Eng., 29, 225-241.

Blaha, M. Year. A Retrospective On Industrial Database Reverse Engineering Projects-Part 1. In: *Proceedings of The 8th Working Conference on Reverse Engineering (WCRE'01)*, 2001 Suttgart, Germany. Ieee Computer Society, 136-147.

Cohen, Y. and Feldman, Y. A. 2003. Automatic High-Quality Reengineering of Database Programs By Abstraction, Transformation and Reimplementation. *ACM Trans. Softw. Eng. Methodol.*, 12, 285-316.

Fong, J., 1997. Converting Relational to Object-oriented Databases. *ACM SIGMOD Record*, 26, 53–58.

Hainaut, J.-L., Henrard, J., Hick, J.-M., Roland, D. and Englebert, V. 1996. Database Design Recovery. *Proceedings of The 8th International Conference on Advances Information System Engineering*. Springer-Verlag.

ISO/IEC. 1992. ISO/IEC 9075:1992, Database Language Sql.

ISO/IEC. 2009. ISO/IEC Dis 19506. *Knowledge Discovery Meta-Model (KDM)*, V1.1 (Architecture-Driven Modernization).

Khusidman, V. And Ulrich, W. 2007. Architecture-Driven Modernization: Transforming the Enterprise. Draft V.5. andHttp://Www.Omg.Org/Docs/Admtf/07-12-01. Pdf. Omg.

Lewis, G. A., Smith, D. B. and Kontogiannis, K., 2010. A Research Agenda for Service-Oriented Architecture (Soa): Maintenance and Evolution of Service-oriented Systems. *Software Engineering Institute*.

Moyer, B., 2009. Software Archeology. Modernizing Old Systems. *Embedded Technology Journal* [Online], 1. Available: Http://Adm.Omg.Org/Docs/Software_ ArcheoLogy_4-Mar-2009.Pdf [Accessed Junio 2009].

Omg 2008. Qvt. Meta Object Facility (Mof) 2.0 Query/View/Transformation Specification. Http:// Www.Omg.Org/Spec/Qvt/1.0/Pdf. Omg.

Omg. 2009. Architecture-Driven Modernization Standards Roadmap. [Online]. Omg. Available: Http://Adm. Omg.Org/Admtf%20roadmap.Pdf [Accessed 29/10/2009].

Pérez-Castillo, R., García-Rodríguez De Guzmán, I., Caballero, I., Polo, M. And Piattini, M. 2009. Preciso: A Reengineering Process and a Tool for Database Modernisation Through Web Services *24th ACM Symposium on Applied Computing*. P. 2126-2133.

Pérez-Castillo, R., García Rodríguez De Guzmán, I. and Piattini, M. 2011. Architecture-Driven Modernization. In: Dogru, A. H. And Bier, V. (Eds.) Modern Software Engineering Concepts And Practices: Advanced Approaches. Igi Global.

Polo, M., García-Rodríguez De Guzmán, I. And Piattini, M. 2007. An Mda-based Approach For Database Re-Engineering. *J. Softw. Maint. Evol.*, 19, 383-417.

Ramanathan, S. And Hodges, J. 1997. Extraction of

Object-oriented Structures From Existing Relational Databases. *ACM SIGMOD Record*, 26, 59–64.

Reus, T., Geers, H. And Deursen, A. V. Year. Harvesting Software for MDA-based Recovering. In: European Conference on Model Driven Architecture - Foundations and Applications, 2006 Bilbao (Spain). Springer-Verlag Berlin Heidelberg.

Sneed, H. M., 2005. Estimating The Costs of a Reengineering Project, *IEEE Computer Society*.

Ulrich, W. M. and Newcomb, P. H., 2010. Information Systems Transformation. Architecture Driven Modernization Case Studies, Burlington, MA, Morgan Kauffman.

Visaggio, G., 2001. Ageing of a Data-intensive Legacy System: Symptoms and Remedies. *Journal Of Software Maintenance*, 13, 281-308.

Wu, W., Reinwald, B., Sismanis, Y. and Manjrekar, R. Year. Discovering Topical Structures of Databases. In: *ACM SIGMOD International Conference on Management of Data*, 2008 Vancouver, Canada. Acm, 1019-1030.