ELECTRONICS AND ELECTRICAL ENGINEERING
ISSN 1392 – 1215 2011. No. 1(107)
ELEKTRONIKA IR ELEKTROTECHNIKA

SYSTEM ENGINEERING, COMPUTER TECHNOLOGY
T 120
SISTEMŲ INŽINERIJA, KOMPIUTERINĖS TECHNOLOGIJOS

Extracting Conceptual Data Specifications from Legacy Information Systems

B. Paradauskas, A. Laurikaitis
Department of Information Systems, Kaunas University of Technology,
Studentu str. 50, Kaunas, Lithuania, phone: +37061023258, e-mails: parad@soften.ktu.lt, aurimaras@yahoo.com

Introduction

Database reverse engineering (DBRE) is defined as the application of analytical techniques to one or more legacy data sources to elicit structural information (e.g. term definitions, schema definitions) from the legacy information sources in order to improve the database design or produce missing schema documentation [1]. Legacy systems typically contain incredible detailed business rules and form the backbone of the information flow of organization that consolidates information about its business. Legacy information systems are currently posing numerous and important problems to their host organization:

1) Systems usually run on obsolete hardware which is slow and expensive to maintain.

2) Maintenance of software is generally expensive: tracing faults is costly and time consuming due to the lack of documentation and a general lack of understanding of the internal workings of the system.

3) Integration efforts are greatly hampered by the absence of clean interfaces.

4) Legacy information systems are very difficult, if not impossible, to expand.

The database reverse engineering must be accomplished in order to discover and extract as much as possible conceptual data specifications from legacy sources. The obtained conceptual schema must pertain to requirements of DBRE quality [2], i.e. schema must be correct, primitive, minimal and appropriate.

DBRE is a part of system reverse engineering process. Model driven development methods are broadly used in software engineering in order to raise abstraction level of applications (e.g. [3–5]). Reverse engineering is a common technique to recover software models from the application code [6] and reverse engineered models may be used for redevelopment, wrapping or migration as well as for generation of application code on the base of discovered implementation patterns [7].

Since 1980, when research efforts on DBRE increased, a wide range of DBRE methods have been presented. These methods attempt to extract the conceptual schema from an operational legacy system and express the schema in some variant of the entity-relationship model (ER, EER, ECR) or of the object-oriented model (OMT). DB-MAIN, Premerlani, Signore, Petit, Chang, MeRCI, Varlet methods overviewed in [8] use many sources to extract explicit and implicit constructs. All of these methods explore DDL code. The methods of Premerlani and Chang examine data, Signore method analyzes program code. DB-MAIN, Petit, MeRCI and Varlet methods analyze data and application code. Petit method is the only method to extract functional dependencies from DDL code and program code, MeRCI method is the only method to elicit redundancy from DDL code and program code. The methods of DB-MAIN, Premerlani and Signore allow supplementing extracted specification with the knowledge of analyst. A wider range of high level semantics can be recovered in case if more different sources are used and more detailed analysis is done.

All DBRE methods concentrate their research on some area: extracting some constraints and/or analyzing some information sources. No one from above mentioned methods propose tools for computer-assisted extraction of secondary identifiers, enumerated value domains, constraints on value domains, existence constraints, attribute semantics and related code fragments. Comparison of currently existing methods is depicted in Table 1.

The goals of a method of extracting conceptual data specifications

The framework of a process of enterprise knowledge extraction from relational database source code of legacy information systems is presented in [8, 9].

Table 1. Comparison of methods of database reverse engineering by recovered constructs

No.	The name of method	The constructs recovered*												
		Entities	Attributes	Primary identifiers	Secondary identifiers	Optional attributes	Enumerated value domains	Constraints on value domains	Relations	Functional dependencies	Redundancy dependencies	Existence constraints	Attribute semantics	Related code fragments
1.	DB-MAIN	1,A	1,A	1,A	A	1,A	A	A	1,2,3,A	–	A	A	A	–
2.	Premerlani method	1	1	1,2,A	–	1	–	–	1,2,A	–	–	–	–	–
3.	Signore method	1	1	1,3,A	–	1	–	–	1,3,A	–	–	–	–	–
4.	Chiang method	1	1	1	–	1	–	–	1,2	–	–	–	–	–
5.	Petit method	1	1	1	–	1	–	–	1,2,3	1,3	–	–	–	–
6.	MeRCI method	1	1	1	–	1	–	–	1,2,3	–	1,3	–	–	–
7.	Varlet method	1	1	1	–	1,2	–	–	1,2,3	–	–	–	–	–

Note: The analysis of sources: 1 – DDL code, 2 – data, 3 – program code, A – analyst knowledge, "–" – can't recover.

The goal of a method of extracting conceptual data specifications is computer-assisted discovery and extraction of knowledge from legacy information systems, i.e. generation of conceptual specification of legacy information system, including entities, atomic, compound and multi-valued attributes, primary identifiers, secondary identifiers, optional attributes, enumerated value domains, constraints on value domains, existence constraints, relations, is-a relations, multi-domain roles, attribute semantics, related code fragments. The obtained conceptual schema must pertain to requirements of DBRE quality.

The steps of MECDS method are computer-assisted; user interaction is required only in these cases:

1) Analyst must make a decision during schema cleaning step if there is a lack of information from legacy system and that would seriously impact the quality of DBRE (attribute constraints supplement, identifier extraction, reference and equality constraints extraction).

2) Analyst must make a decision during conceptual normalisation step if during transformation „Joining entity types with is-a relation" is-a relation with type disjoint or partition was identified, in order to avoid inclusion of incorrect high level semantics.

The features of MECDS method:

1) Can be applied to relational databases only.

2) Can be applied to procedural programming languages only.

3) Omit physical integration sub-step of data structure extraction; therefore legacy information system must include only one database.

4) Omit dead data structures.

5) Can be applied to databases with no data errors.

6) Data are not moved, but are analysed in the database of legacy information system instead.

7) Should be performed to correctly designed systems only. If the system was poorly designed, then the extraction of conceptual data specifications of that system is meaningless.

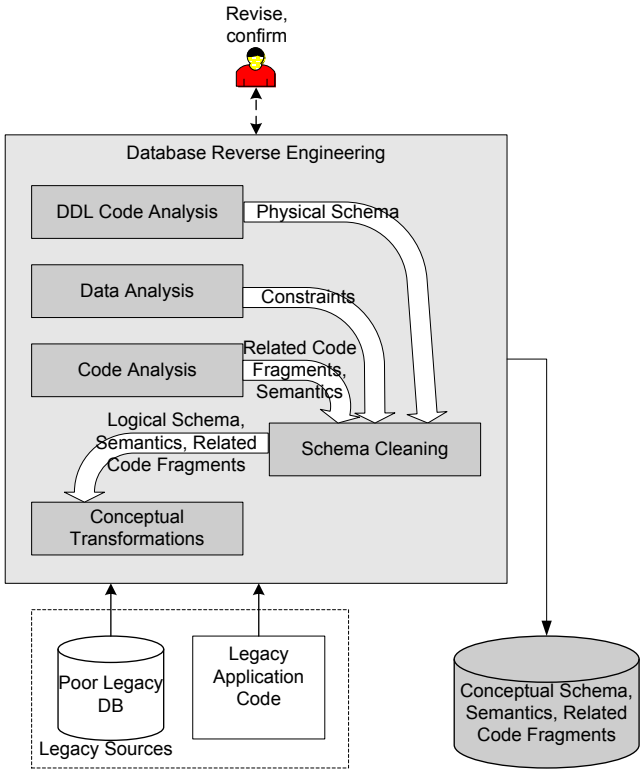

Fig. 1. The schema for method of extracting conceptual data specifications

MECDS method expresses conceptual schema in EER model. This model represents constructs at different levels of abstraction, ranging from physical to conceptual, in graphical form and it is widely used in CAD tools.

MECDS method extracts the basic schema information from legacy system database management system DDL code (Fig. 1). Then this schema information can be semantically enhanced using details from application code and data. Conceptual schema must include high level semantic structures as accurate as possible, so functional dependencies and existence constraints must be elicited and redundancies must be

discovered and removed. The final conceptual schema is generated using the set of conceptual transformations.

The steps of the method

DBRE methodology was widely explored and the main steps were identified. The conducted DBRE research was generalized and integrated into a new solid method. MECDS method supplements the schema refinement step of DBRE with new algorithms to recover implicit constructs from DDL code, program, represented by abstract syntax tree, and data. MECDS method also augments the set of currently existing conceptual transformations in order to evaluate newly extracted structures, defines the order of transformations and enables computer-assisted recovery of conceptual specification.

MECDS method includes two main database reverse engineering processes: data structure extraction and data structure conceptualization. An overview of MECDS method which is comprised of seven steps is shown in Fig. 2.

1) DDL code analysis is the simplest step of data structure extraction. It produces raw physical schema, which consists of declared data structures and constraints.

2) The schema refinement is the most important step of data structure extraction. It identifies and extracts structures and constraints that were implemented implicitly or were discarded during application development. Many implicit constraints exist: primary and secondary identifiers, reference and equality constraints, functional dependencies, meaningful names, etc. Schema refinement process enriches physical schema with implicit constructs and produces complete physical schema. It consists of six sub-steps:

a) The attribute constraints supplement augments raw physical schema with constraints that were not discovered during DDL code analysis: optional attributes, enumerated value domains, constraints on value domains.

b) The abstract syntax tree generation constructs abstract syntax tree for the legacy application code in order to support various techniques of program understanding for discovering implicit structures and constraints.

c) The identifier extraction obtains primary identifiers, if primary key information was not retrieved in DDL code analysis step, and discovers secondary identifiers.

d) The reference and equality constraints extraction identifies constraints to help classify the extracted entities, which represent both the real-world entities and the relationships among them.

e) The other constraints extraction obtains implicit constructs, i.e. constraints and dependencies: coex, exact-1, disjoint(R1.A1,R2.A1), R1∪R2:A1-->A2, R1.A1in(R2.A1∪R3.A1), R:BA1==>(BA1=A1) and redundancy constraint rd.

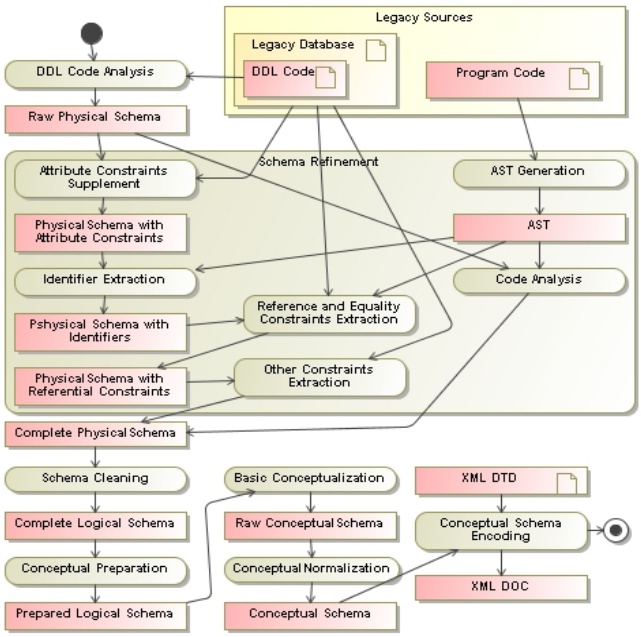

Fig. 2. The steps of method of extracting conceptual data specifications

f) The code analysis: (1) augments entities with domain semantics, and (2) identifies related code fragments and constraints not explicitly stored in the database, but which may be important to the process of reverse engineering. This method to code analysis is based on program understanding, which includes system dependency, program slicing and pattern matching.

3) The schema cleaning removes or replaces physical constructs into logical ones and transforms complete physical schema into complete logical schema.

4) The conceptual preparation discards technical data structures, i.e. prepares the schema that only contains structures and constraints that are necessary to understand the semantics of the schema.

5) The basic conceptualization extracts a raw conceptual schema without worrying about esthetical aspects of the result. The main transformations are:

a) Removing redundancies.

b) Transforming foreign keys with constraints into relationship types and internal constraints.

c) Transforming remaining foreign keys into relationship types.

d) Transforming list of attributes into a multi-valued attribute.

e) Aggregating attributes.

6) The conceptual normalization transforms various constructs and gives expressiveness, simplicity, minimality and extensibility of conceptual schema. It tries to make higher level semantic constructs explicit (e.g. is-a relation). The semantic value of schema is not changed, however the syntax is normalized. The main transformations are:

a) Merging entity types.

b) Transforming entity types into attributes.

c) Transforming entity types into relationship types.

d) Transforming relationship types into multi-domain roles.

e) Transforming relationship types into is-a relations.

f) Connecting entity types with is-a relations.

g) Name processing.

7) The conceptual schema encoding is a technical step that extracts schema in the form of an XML document.

Seven steps of MECDS enable the extraction of the following schema information from the legacy database:

1) Entities;

2) Compound, multi-valued and atomic attributes;

3) Primary and secondary identifiers;

4) Existence constraints;

5) Relationships;

6) Is-a relations;

7) Multi-domain roles;

8) Related code fragments.

The conceptual specification extracted for order information system is presented in EER diagram in Fig. 3 [9].

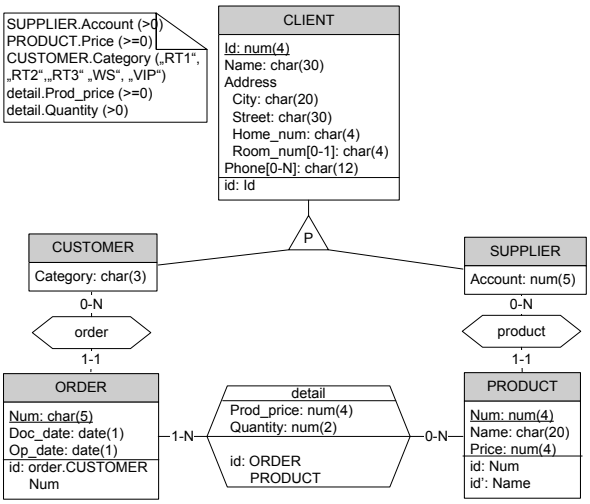

Fig. 3. The extracted conceptual data specification

Table 2. Comparison of MECDS method with other methods of DBRE by recovered constructs

No.	The name of method	The constructs recovered*											
		Entities	Attributes	Primary identifiers	Secondary identifiers	Optional attributes	Enumerated value domains	Constraints on value domains	Relations	Functional dependencies	Redundancy dependencies	Existence constraints	Attribute semantics	Related code fragments
1.	DB-MAIN	1,A	1,A	1,A	A	1,A	A	A	1,2,3,A	–	A	A	A	–
2.	Premerlani method	1	1	1,2,A	–	1	–	–	1,2,A	–	–	–	–	–
3.	Signore method	1	1	1,3,A	–	1	–	–	1,3,A	–	–	–	–	–
4.	Chiang method	1	1	1	–	1	–	–	1,2	–	–	–	–	–
5.	Petit method	1	1	1	–	1	–	–	1,2,3	1,3	–	–	–	–
6.	MeRCI method	1	1	1	–	1	–	–	1,2,3	–	1,3	–	–	–
7.	Varlet method	1	1	1	–	1,2	–	–	1,2,3	–	–	–	–	–
8.	MECDS method	1	1	1,3,A	1,3	1,2,A	1,2	1,2	1,2,3,A	1,2	1,2	1,2	1,3	1,3

Note: The analysis of sources: 1 – DDL code, 2 – data, 3 – program code, A – analyst knowledge, "–" – can't recover.

The evaluation of the method

Comparison of MECDS method with other currently existing DBRE methods is depicted in Table 2.

MECDS method explores DDL code, program code and data. As other analysed methods MECDS method automatically obtains these constructs: entities, attributes, primary identifiers, optional attributes, relationships. Proposed method analyses various information sources more deeply than other methods: primary identifiers are elicited from DDL code and program code (like Signore method), optional attributes obtained from DDL code and data (like Varlet method). From DDL code and data MECDS method extracts functional dependencies (Petit

method obtains it from DDL code and program code) and redundancy dependencies (MeRCI method obtains it from DDL code and program code). Presented method obtains enumerated value domains, constraints on value domains and existence constraints from DDL code and data and secondary identifiers, attribute semantics and related code fragments from DDL code and program code. Other currently existing DBRE methods can't recover these constructs.

MECDS method evaluates more sources and extracts more accurate physical constructs, so it better recovers high level semantic structures. MECDS method performs computer-assisted recovery of all constructs that are extracted by other currently existing DBRE methods; moreover, it obtains secondary identifiers, enumerated value domains, constraints on value domains, existence constraints, attribute semantics and related code fragments from DDL code, program code and data.

Conclusions

All currently existing database reverse engineering methods analyse legacy sources formalised by database logical schemas and extract only a limited range of implicit structures. None method propose tools for computer-assisted extraction of secondary identifiers, enumerated value domains, constraints on value domains, existence constraints, attribute semantics and related code fragments.

Performed research validated hypothesis, that currently existing methodology of database reverse engineering can be improved and more accurate conceptual specification of legacy system can be recovered by computer-assisted by using data analysis and program understanding techniques.

The proposed method enables the recovery of conceptual data specifications that includes a wide range of conceptual constructs: entities, atomic, compound and multi-valued attributes, primary identifiers, secondary identifiers, optional attributes, enumerated value domains, constraints on value domains, existence constraints, relations, is-a relations, multi-domain roles, attribute semantics, related code fragments. Elicited conceptual schema can be used in forward engineering in order to specify or adjust functional requirements.

The advantage of recovered conceptual elements of schema:

The increased number of domain elements enables the representation of business rules as simple sentences of natural language.

Intuitive normalization procedure [10] could be applied to legacy logical schema that is optimised in expert manner, transitive functional dependencies are not included in recovered schema and it is in 3NF. Therefore the requirements of DB structural fullness could be fulfilled in forward development of reengineering process.

Elimination of transitive functional dependencies during structural normalization enables the representation of conceptual schema semantic description and functional requirements as simple sentences of natural language.

References

1. **Hammer J., et al**. Knowledge Extraction in the SEEK Project Part I: Data Reverse Engineering // Technical Report TR-0214, 2002.
2. **Paradauskas B. Nemuraitė L**. Duomenų bazių semantiniai modeliai. – Kaunas: Technologija, 2008. – 337 p.
3. **Kalnius R., Eidukas D.** Probability Model Quality of Informations Systems // Electronics and Electrical Engineering. – Kaunas: Technologija, 2009. – No. 4(92). – P. 13–18.
4. **Pavalkis S., Nemuraitė L., Tarvydas P., Noreika A**. Specification of finite element model of electronic device using model-driven Wizard-based guidance // Electronics and Electrical Engineering. – Kaunas: Technologija, 2010. – No. 2(98). – P. 59–62.
5. **Šilingas D., Butleris R**. Towards Implementing a Framework for Modeling Software Requirements in MagicDraw UML // Information Technology and Control. – Kaunas: Technologija, 2009. – No. 38(2). – P. 153–164.
6. **Canfora G., Penta M. D.** New frontiers of reverse engineering // Proceedings of Future of Software Engineering (FOSE'07), 2007. – P. 326–341.
7. **Ablonskis L., Nemuraitė L.** Discovery of model implementation patterns in source code // Information Technology and Control. – Kaunas: Technologija, 2010. – No. 39(1). – P. 70–76.
8. **Paradauskas B., Laurikaitis A.** Business knowledge extraction from legacy information systems // Information Technology and Control. – Kaunas: Technologija, 2006. – No. 35(3). – P. 70–76.
9. **Paradauskas B., Laurikaitis A**. Business knowledge extraction using program understanding and data analysis techniques // Proceedings of the 15th International Conference on Information and Software Technologies, April, 2009. – P. 337–354.
10. **Nummenmaa J., Seppi A., Thanisch P.** Automating Support for Refactoring SQL Databases // Proceedings of the 16th International Conference on Information and Software Technologies (IT 2010), April, 2010. – P. 343–349.

Received 2010 10 18

B. Paradauskas, A. Laurikaitis. Extracting Conceptual Data Specifications from Legacy Information Systems // Electronics and Electrical Engineering. – Kaunas: Technologija, 2011. – No. 1(107). – P. 46–50.

This article discusses the process of extraction of conceptual data specifications from relational database DDL code and data and source code of legacy information systems. Problems of legacy information systems and main solutions for them are briefly described here. Method of extracting conceptual data specifications from legacy information systems is provided. Comparison with other currently existing database reverse engineering methods is presented. Ill. 3, bibl. 10, tabl. 2 (in English; abstracts in English and Lithuanian).

B. Paradauskas, A. Laurikaitis. Koncepcinių duomenų specifikacijų išgavimas iš liktinių informacijos sistemų // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2011. – Nr. 1(107). – P. 46–50.

Straipsnyje nagrinėjamas koncepcinių duomenų specifikacijų išgavimas iš liktinės informacijos sistemų reliacinės duomenų bazės DDL kodo ir duomenų bei pradinio kodo. Aptariamos liktinių sistemų problemos ir pagrindinės jų sprendimų alternatyvos. Pateikiamas metodas koncepcinėms duomenų specifikacijoms išgauti iš liktinių sistemų. Jis lyginamas su kitais literatūroje aprašytais atvirkštinės duomenų bazių inžinerijos metodais. Il. 3, bibl. 10, lent. 2 (anglų kalba; santraukos anglų ir lietuvių k.).

DOI: 10.5755/j02.eie.9079