Database Integrity in Integrated Systems

José Francisco Zelasco[1] and Judith Donayo[2]

[1]Universidad de Buenos Aires, Facultad de Ingeniería, Departamento de Mecánica
D. Matheu 1222, Piso 7 "A", C1249AAB, C. A. B. A., Argentina
[2]Universidad de Buenos Aires, Facultad de Ingeniería, Departamento de Mecánica
Las Achiras 2279, B1618, El Talar, Buenos Aires, Argentina

Abstract. This article makes a proposal, looking into certain aspects of Unified Modelling Language (UML) and Unified Process (UP) usage, when applied to the database integrity in integrated information systems. The usage of some tools and heuristics based on the gathered experience originated on the MERISE method evolution, gives place to comparative benefits while conceiving this type of informatics applications. The heuristic and employed tools consist of starting from a higher level of abstraction provided with inverse engineering and re-engineering instrumentation methods. The knowledge of a minimal data schema enables a global vision. The data integrity is assured by means of verifications between the minimal data diagram and the persistent data object, reducing the number of modifications done to the initially built subsystem data. Finally, the conceptual and organizational levels of treatment allow the comparison of the organization options, yielding naturally into the user manual and object design.

1 Introduction

Database integrity of an information system from its creation to the end of its life cycle is an important issue of concern among specialists [14], [11], [17], [8], [1] The proposal concerning all the aspects of an information system project which complements the UML method at the level creation [26], is introduced here with the aim of ensuring the integrity of the database throughout the development of the system. The general aim of this paper consists of an interconnected set of heuristics and mechanisms that improve the importance of engineering requirements, facilitating the creation of specifications, suggesting alternatives for the definition of organization in terms of job positions, tasks, and the creation of distributed information systems. Consequently, the information produced is stricter and simpler, and facilitates, in terms of design, the use of tools such as those proposed by the Unified Modeling Language (UML) and the Unified Process (UP). In this presentation we will lay special emphasis on those tools that are related to data structuring and that make their monitoring easier during the optimization and the distribution of data on the physical level, while protecting their consistency.

As an introduction to the process of creation, we will introduce a diagram called sun (Fig. 1) [19] in which we can see that the stage of creation articulated in three levels of abstraction:

Zelasco J. and Donayo J. (2009).
Database Integrity in Integrated Systems.
In *Proceedings of the 7th International Workshop on Modelling, Simulation, Verification and Validation of Enterprise Information Systems*, pages 132-140
DOI: 10.5220/0002200001320140

1. Conceptual level: what the company does, as a whole, before external actors.
2. Organizational or logical level, (namely internal actors), what the company does, where and when.
3. Operational or Physical level, how something is done. There is a distinction here between the tasks performed by men known as men's tasks ,which give rise to the user's manual and the tasks performed by machines known as machine tasks, involved in the information system.

This diagram goes from bottom left to bottom right like the movement of the sun, passing in the middle through the upper level, i.e., the conceptual one. The whole line can be followed iterating twice. The first iteration occurs after the selection of elements that due to their volume (data) and frequency (events) are of greater importance. This selection is, thus, known as a representative subset and its creation corresponds to a preliminary study of the project. The second one comprises the field of study as a whole, so it corresponds to a complete and detailed study.

The line of the figure, from the beginning to the current conceptual level, corresponding to inverse engineering, which involves the scenarios of the system in its current state, is the way the system works. Important aspects, such as management rules, should be stated. Management rules include what the company should do as a response to external requirements and integrity constraints, which allow for the determination of functional dependencies.

Reengineering consists of the transition from the current state to the future state. Some of the factors taken into account for this passage are: the aim of the company, the external companies that affect it (competition, legislation and other factors that could eventually have an incidence in the decisions of the company, etc), all the decisions from the Board (articles of association, policies, strategies, objectives, and available resources), fields of activity, the different duties that arise from the organization chart, etc. From all this information there arise: integrity properties, constraints and restrictions relevant to the data, management rules relevant to the data-treatment (not to be confused with business rules as described by some authors), etc, that bring about the passage to the future state, which corrects and enriches the results of the current conceptual model. The results of the current model known as inverse engineering are the starting point, followed by the gathering of information about the context and definitions of the Board (reengineering) to obtain the future model.

It is also admitted as a hypothesis that there is greater invariance in the data, related to integrity properties and restrictions than in the treatments, concerning events and management rules. From now on, we will try to determine [15], [9]:

1. The minimal data model that includes all the information of a distributed system, in terms of properties, and which will be stored in the physical database to be reached through different transitions, mechanisms and optimizations.

2. The treatments model based on Petri nets [23], [5] are executed by a subsystem. A first scheme describes the Process itself, i.e., what the company is to do as a response to each external requirement, to initial events of processes and to those which derive from them. This brings about a concatenated series of operations, an ongoing activity, per Process. The operations contain tasks expressed in terms of management rules. Next, that same scheme is enlarged in the Procedure replacing each operation in one or many phases, corresponding to the uninterrupted activity

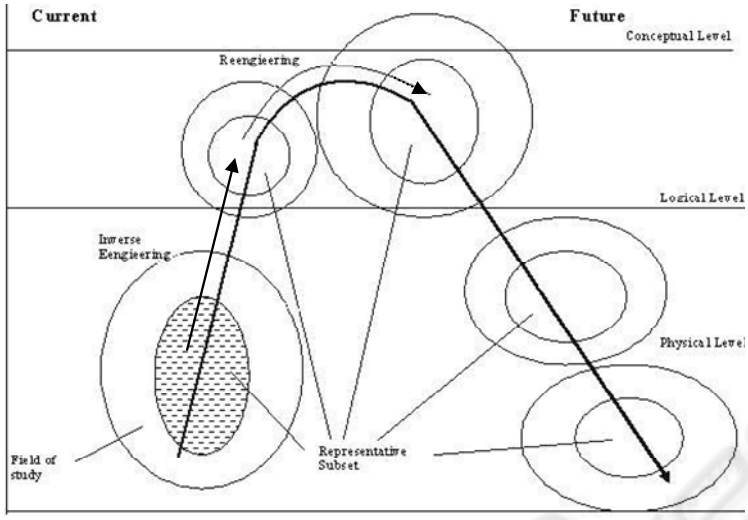

Fig. 1. Diagram of the sun.

of a specific job position. This brings about different organization options i.e., scenario options that, in this level, could be evaluated comparatively by the company's Board, which is to choose the most convenient one. The scheme describes the future scenarios and use cases. The following level is the operational level, in which each job position distributes the tasks according to the person's activity and the information systems activity. The tasks performed by a person correspond to the user's manual and those automated correspond to the system analysis. The tasks derived from management rules are expressed less colloquially and eventually more formally.

3. Integrity verification. From the system analysis we pass on to the design, and once the relevant objects are created, the consistency between the persistent properties and the minimal established scheme is verified. This is to be done by updating and querying each one of the corresponding entities /properties. This heuristic ensures that the minimal data scheme meets the needs of each subsystem, but the main advantage of this mechanism is to ensure that each subsystem provides all the elements required by the other subsystems. In this way the modifications of the previous subsystems are to be minimized when the following ones are developed according to the priorities established by the Board.

2 Conceptual Data Model

The minimal and complete conceptual data model allows an overall view of the information system data. This overall view of the memory shared by all the actors of the system is the mechanism that allows a truly systemic approach of the project. The database administrator transfers this minimal and total conceptual model without redundancy, to its physical form through passages and optimizations.

To create the minimal conceptual data modeling, that allows for a global view of the information system, we do not proceed as in object creation, i.e., the starting point are the fields of activity from which the relevant properties are gathered and classified, trying not to confuse or relate classes to objects. This yields greater objectivity to the integrity verification, as the person who executes the data scheme is not usually the one who creates the objects of each subsystem.

During the preliminary study, (first iteration) the most important data are chosen, taking into account the volume and the most frequent processes, so as to determine a representative subset which, facing the Conceptual data model, can be articulated with all and each of the subsystems. This can be done iteratively, since at the stage of detailed study (second iteration) such model will have all the properties with no redundancy that are to be stored in the physical base.

A list of integrity constraints and restrictions should be created to give rise to functional dependencies. From the current model and taking into consideration the modifications that result from reengineering, we pass on to the future model, as we have mentioned above.

To avoid property redundancy, this scheme requires for existing unary relationships to be expressed without being transformed into binaries. Although the look here approach (MERISE proposal) [19] [20] [16] and not the look across one [4] could simplify the representation of options in ternary relationships, it is useless to discuss their advantages. In fact, an approach should be developed, which in a simple way allows the representation of all the functional dependencies that both complementary approaches represent.

There are certain points to take into account to reach this minimal model. Some are rather elementary, others are subtle:

1. Each property appears only once in the scheme, as a class or relation attribute.
2. The scheme should respect the second normal rule [2] [21] [24].
3. If a property depends on the identifiers of two or more classes, it is a property of the relation that links such classes (third normal rule.) [2] [21] [24] [22].
4. Each property can be assigned only one value.
5. An arc that links a class with a relation cannot be optional, i.e., when there is a relation occurrence, it should be linked with an occurrence of each class. If this is not the case, it is because there are two different relations that have to be separated because conceptually it is not the same, even though in the physical level this optimization may not be admitted.
6. Only one occurrence of each class converges to each relation occurrence and reciprocally only one relation occurrence converges to each set of occurrences. If this does not happen it is because the indispensable class that avoids this ambiguity has been omitted.
7. It should be verified that each class has a set of occurrences, so as not to confuse unique objects (company's Board) with data classes, which is a frequent mistake among beginners.

It should be pointed out that this minimal scheme can be the basis for a conceptual object-oriented database model scheme using simple generalization [25], and evaluating multiple inheritance.

With this minimal model you can verify all the updating and queries of each and every persistent data of each application or subsystem object; this is the "verification of the integrity".

From there on, the database administrator may choose between an object-oriented database and a relational database [3]. He will progressively simplify and include optimizations with the corresponding redundancy that will be documented to allow inverse engineering to be done without difficulty whenever necessary. This permits to control redundancy since reference is always made to the same minimal model. Indeed, whenever a modification to the database is required, (inverse engineering) that modification must be expressed in the minimum scheme so that it affects the entire distributed database to ensure data consistency. Thus, it will be enough to go down the optimizations tree and transformations from the minimum model to the database. It should be taken into account that the optimizations must respect, if possible, the search for the minimum storage cost function and process time cost (holding cost) and it must also be well documented; thus facilitating such drop . Besides, the information will be distributed and correctly documented at the corresponding level. Relying on the documented tracing of the simplification, the optimization, and the distribution at its different levels are not irrelevant factors to guarantee integrity and the consequent minimization of the cost of development and maintenance of the project.

3 Processing Model

We shall briefly mention some aspects of the mechanisms used to describe treatments at different levels, with the sole intention of showing an overall view of the complete proposal, since this presentation focuses on the aspect of data integrity in information systems.

We suggest tools derived from Petri's nets to develop the Conceptual Model of Treatments as well as the Organizational Model of Treatments [27]; [5]. The richness of the proposed diagrams is superior to that of the sequence or activity diagrams suggested by UML at lower levels, and thus more appropriate to model at this level of abstraction as the conceptual level.

3.1 Conceptual Model of Treatments

The Conceptual Model of Treatments describes the processes that initiate from primary events which are generally external or assimilable to external ones.

These processes are divided into operations which have a set of management rules. The events as well as such management rules (not business rules) [3] come from the criteria mentioned in the introduction. Management rules describe each action that the company should accomplish to satisfy events, not only the initial ones but also those arising during the process. The operations involve a series of ongoing or uninterrupted actions. The necessity of a new operation arises from an interruption, as a result of a response event from an external actor. The organization thus waits for another external event to restart the following operation. In terms of events, there is no distinction between response and events, that is to say the response is, in itself, a new event.

It is worth noting that at this level (the conceptual level) no account is taken of who does what, when and where in the company. The organization as a whole responds to the events; this permits, in the following level, to propose, analyze and evaluate different management options in terms of job positions having previously stated "what the company does" when facing events, essentially external ones.

The synchronization prior to each operation can be seen at this conceptual level. This synchronization uses logical operators of conjunction and disjunction. This allows representing a single event or the different sets of events that trigger the operation. In the case of response events, there can also be an indication of the output conditions in which they are produced. The acronyms of the corresponding management rules are inscribed in the operation which must have a name. These acronyms are in correspondence with the list of rules where the actions involved are described

3.2 Organizational Model of Treatments

The organizational model of treatments describes the characteristics of the treatments that have not been expressed in the conceptual model of treatment, expanding, in terms of job positions, the conceptual model of the company's organization, that is to say, timing, resources and places.

The conceptual model of treatments describes the flow of events, mainly those between the organization of the company and the environment.

The organizational model of treatments adds to the conceptual model the company's flow of events among the different job positions [5]

The study of a company's organizational problem belongs to other experts, so computer specialists are frequently restricted to taking that model as the working base for the system development [6]. Different organization options, other than the one imposed, show improvements in the information system. This methodological proposal not only prevents this inconvenience and the waste of money and time but also proposes a study of the company's organization options. When a computer specialist takes part in contributing with this formal tool based on Petri nets [12] the advantages of one option over the other one become obvious, particularly from the automatization point of view. The decision about the company's organization results then in an agreement and so the computer specialist faces a more solid and robust option. Besides, this organizational model of treatments contributes to a more precise elaboration of the user's manual and to the analysis prior to design.

The organizational model of treatments consists of expanding each process of the Conceptual Model of Treatments into the respective procedures.

A chart is used to describe the procedure. The first column or the first set of consecutive columns corresponds to the external actor/actors related to such process. The subsequent columns correspond to the job positions to which the procedures are expanded. The uninterrupted or ongoing activity at the procedure level is called phase. The phase structure is identical to the operation structure of the conceptual model of treatments. The phase has the possibility of synchronizing events by means of logical operators of conjunction and disjunction. It has an identification name, the acronyms of the management rules which correspond to the actions that a job position must carry out during that phase and the output conditions of the response events. It is

important to highlight that response events may be directed to both external actors and internal actors (other job positions). From the foregoing, it can be said that one or more phases correspond to each operation and that in the case the operation is divided into various phases, the management rules contained in it will be distributed among the respective phases.

Finally, in case the phase has rules capable of automatization, the operations that will be done by the person in that job position and those operations to be automated will be established. These actions are called “men’s tasks” and “machine tasks”. This classification is called Operational Treatments Level and gives rise to the user’s manual and the systems analysis at the level of each use case.

The progressive detail grade of these three levels shows that they are three different levels of abstraction and that they are useful for a better information system definition. In this way the representation of use cases and scenarios options is facilitated.

4 Integrity Verification or Consistency Validation

Integrity Verification or consistency validation (Fig. 2) allows us to ensure data integrity and the harmonic development of subsystems by reducing costs and the redundancies derived from the modifications of previous subsystems developed in order to satisfy the subsequent ones. The verification process consists of contrasting the object persistent properties of each system with the minimal and complete conceptual data model.

When analyzing the object persistent properties in previous subsystems, it is assumed that some anomalies may occur. However, in the minimal conceptual model these anomalies would be considered as a sub product of the verification process. Since verification mainly aims to ensure that each subsystem yields the essential information to the proper functioning of other subsystems. This interaction is fundamental when the information is distributed.

Some applications are developed before others for some priority reasons. In this case, the verification process guarantees that the cost of the modifications of the previously designed applications is minimum or inappreciable.

This objective is achieved by verifying that the persistent properties of each subsystem can be updated and accessed. It is neither necessary nor possible that data structure from the minimal conceptual model resembles that from the object models. However, there must be identifiers (or access paths) that allow for the updating of classes and relation occurrences as well as of particular properties.

The overall view through the minimal model permits us also to control redundancies and to avoid problems thus contributing to diminish the system entropy whose cost may be inappreciable mainly at the beginning of the maintenance phase. Applying this mechanism and consequently reducing the entropy leads to a more balanced and tolerant system to the changes in the context in which the organization is immersed and to the requirement modifications.

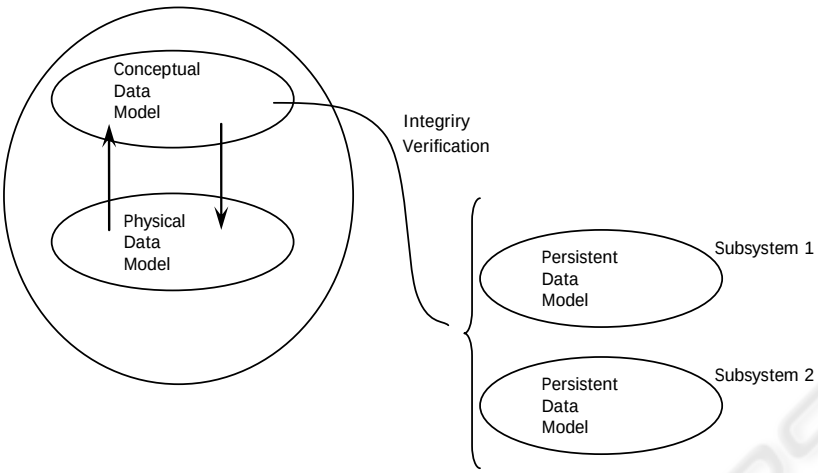

Fig. 2. Integrity verification.

5 Conclusions

Preserving the minimal conceptual model along the system life preserves from inconsistencies

The minimal conceptual data model provides a systemic global view.

The current model of data and treatments, achieved by inverse engineering, guarantees a higher stability of the system.

And finally, the consistency verification assures minimal modification in previous developed subsystems.

The application of these tools and heuristics ensures the stability of the database, provides a reduction of the number of iterations in the development and control of the systems entropy that result in a substantial reduction of development and maintenance costs.

References

1. An Lu, Wilfred Ng, 2009: Maintaining consistency of vague databases using data dependencies. Data & Knowledge Engineering, In Press, Corrected Proof, Available online 28 February 2009.
2. Batini C., S. Ceri y S. B. Navathe, 1991, “Database Design: An Entity-Relationship Approach”, Prentice-Hall.
3. Ceri, S., Fraternali, P., 1997, “Designing Database Applications with Objects and Rules: The IDEA Methodology” Addison Wesley; 1st edition, ISBN: 0201403692.
4. Chen, P. S., 1976: “The entity relationship model: to-ward a unified view of data”, ACM Transactions on Data-base Systems, (1), 9-36.
5. Chu, f., Proth, J-M., Savi, V. M., 1993: “Ordonnancement base sur les reseaux de Petri“. Raport de recher-che No 1960 INRIA Francia.

6. Dey D., Storey V. C. and Barron T. M., 1999, “Im-proving Database Design trough the Analysis of Relation-ship“, ACM Transactions on Database Systems, 24 (4), 453-486.
7. Dullea J. and I. Y. Song, 1998, "An Analysis of Struc-tural Validity of Ternary Relationships in Entity Relation-ship Modeling", Proceed. 7º Int. Conf. on Information and Knowledge Management, 331-339.
8. Eastman C, Stott Parker D. and Tay-Sheng Jeng 1997: “Managing the integrity of design data generated by multiple applications: The principle of patching.” Research in Engineering Design, Volume 9, Number 3 / septiembre de 1997
9. FAQ Merise et modelisation de donnees - Club d’entraides developpeurs francophones. http://uml.developpez.com/faq/merise/
10. González, J. A., Dankel, D., 1993: “The Engineering of Knowledge-Based Systems“. Prentice Hall.
11. Guoqi Feng, Dongliang Cui, Chengen Wang, Jiapeng Yu 2009: “Integrated data management in complex product collaborative design” Computers in Industry, Volume 60, Issue 1, January 2009, Pages 48-63
12. He, X., Chu, W., Yang, H., 2003: “A New Approach to Verify Rule-Based Systems Using Petri Nets“. Information and Software Technology 45(10).
13. Larman Craig, 2001: “Applying UML and Patterns: An Introduction to Object-Oriented. Analysis and Design and the Unified Process” Prentice Hall PTR; 2d edition July 13, ISBN: 0130925691.
14. Melton J, Simon A R, 2002 “Constraints, Assertions, and Referential Integrity” SQL: 1999, 2002, Pages 355-394
15. MERISE - Introduction à la conception de systèmes d’ınformation. http://www.commentcamarche.net/merise/concintro.php3
16. Mounyol, R. “MERISE étendue: Cas professionnels de synthèse“. ISBN : 2729895574. Rojer Mounyol ed. Elipses
17. Post G., Kagan A., 2001:“Database Management systems: design considerations and attribute facilities” Journal of Systems and Software, Volume 56, Issue 2, 1 March 2001, Pages 183-193
18. Song, I. Y., Evans, M. and Park, E. K., 1995: “A comparative Análisis of Entity-Relationship Diagrams”, Journal of Computer and Software Engineering, 3 (4), 427-459.
19. Tardieu, H., Rochfeld, A., Colletti, R., 1985 “La Méthode MERISE“. Tome 1. ISBN: 2-7081-1106-X. Ed. Les Edition d’organization.
20. Tardieu, H., Rochfeld, A., Colletti, R., Panet, G., Va-hée, G., 1987 “La Méthode MERISE“. Tome 2. ISBN: 2-7081-0703-8. Ed. Les Edition d’organization.
21. Ullman J. D. and J. Windom, 1997, “A FirstCourse in Database Systems”, Prentice-Hall.
22. Ullman Jeffrey D. & Freeman W. H., 1988, “Principles of Database and Knowledge-Base Systems” Vol. 1, 1a edition, ISBN: 0716781581.
23. Wu, Q., Zhou, C., Wu, J., Wang, C., 2005: “Study on Knowledge Base Verification Based on Petri Nets. Inter-national Conference on Control and Automation“ (ICCA 2005) Budapest, Hungry, June 27-29.
24. Yourdon, Inc., 1993: “Yourdon Method“, Prentice-Hall.
25. Zelasco J. F, Alvano, C. E., Diorio, G., Berrueta, N., O’Neill, P., Gonzalez, C., 1998: “Criterios Metódicos Básicos para Concepción de Bases de Datos Orientadas a Objetos“. Info-Net, III Congreso Internacional y Exposición de Informática e Internet. Proceding en CD. Mendoza, Argentina.
26. Zelasco J. F., Donayo J., Merayo G., 2007: Complementary Utilities for UML and UP in Information Systems. EATIS 2007 (ACM DL). El Faro, Portugal.
27. Zhang, D., Nguyen, D., 1994: “PREPARE: A Tool for Knowledge Base Verification“. IEEE Trans. on Knowledge and Data Engineering (6).