**Comparison of Conceptual, Logical and Physical models vs. the Data Point Modelling.**

(editor: Ignacio Santos)

**Abstract**

This document analyzes the different phases of the design of the software. Three models for this document are chosen. In the Conceptual Model, the Entity/Relationship model, Multidimensional model, and the Data Point Model (DPM) are compared. For the Logical Model, the Relational Model, MOLAP[[1]](#footnote-1)/ROLAP[[2]](#footnote-2)/HOLAP[[3]](#footnote-3) are compared. This document reviews different models of data design, because the DPM is a model of data.

1. **Introduction.**

Figure . Comparison of models.

When a Project Leader develops an application from the beginning, he or she has to think in the data. Once the designer has created the Conceptual Schema, the designer has to transform the Conceptual to Logic schemas, because the Logic Model is nearest to a physical model, a DBMS (Database Management System) or Taxonomy. The target of every design is the automation of the process, and if it is possible through a tool CASE (Computer Aided Software Engineering). The design starts with the Conceptual Model from UD (Universe of the Discourse) and after from this Conceptual model to Logic model is transformed. At the end, a schema in a DBMS or in XML is created. A scheme has to be verified syntactically and semantically. The syntactical verification has to check the rules of building [1]. The semantic verification tries to find the inconsistency between the semantic constrains and the user. The figure 2 shows the steps of validation and verification of a scheme. A good Conceptual scheme has to have Formal Properties, Quality Factors and Conformance with the user necessities (validation). Formal properties mean that the scheme has to be consistent, complete and irredundant [2] [4]. The completeness of a Conceptual scheme can be defined with respect to the meta-model or the UD represented. The first part the metamodel concerns the mandatory elements that constitute a conceptual schema.



Figure 2. Validation and verification of a schema.

The represented UD means the validation with respect to the user requirements. Checking whether a Conceptual schema represents all the necessary knowledge for a given information system, which refers to conformance of the Conceptual schema to the real world. With respect to quality factors, we have to look at the things the readability and the reusability. Readability is a desirable property, but it is a subjective valuation. The reusability is far away in commercial tools. For the last, the validation of a schema means that if the schema is adapted to the user requirements. In this topic, some tools have developed the Paraphrasing [3]. This technique generates a textual description from a Conceptual schema and the user can validate the model.

1. **Entity Relationship model.**

When a data model from the user is obtained, from the Universe of Discourse the Conceptual model, in this case the Entity Relationship model is gotten. The E/R model can be used as a base for a unified view of the data [5] [6] [7] [8]. An entity is a data object. For Chen, an entity is a thing that can be identify in a clear way. The attributes are properties of the entity. A relationship is the association between objects of one or more different entities. For Chen, a relationship is an entailment between entities. This model has the next elements: Entity, Relationship, Attribute and Domain (for Chen the set of values).

An entity can be defined as any object. So, by example ASSETS can be an entity type that describes the common characteristics of a set of assets. It can have strong and weak entities. An weak entity is an entity who its objects depending on existence of a strong entity. For example, we can have the entities BOOK and EDITION. The entity BOOK is a strong entity, with the tuples “book1”, “book2”, “book3”,… The entity EDITION is a weak entity, the tuples exist, if it exists the book of that edition.

A relationship is an association, entailment, correspondence between entities.

An attribute is the set of properties or characteristics that has an entity or relationship. The attributes can be compounds, as by example the attribute date is consisted of the attributes day, month, and year.

A domain is the possible set of values that can have a characteristic. Then a domain is the set of homogeneous values with a name. For example, it is possible define the domain “Languages” that can have the value “English”, “Spanish”, “German”, “Lithuanian”,…

The model E/R has the constraints inherent between entities that have to have relationships. They can never have "entity-entity" or "relationship-relationship". Also, an entity must have an identifier. It can be considered as inherent constraint. However, this model has integrity and structural constrains. These constraints are about values, an attribute with an only domain is defined.

Two entities or more through a relationship with cardinality are defined. For example, the entities “BOOK” and “EDITION”, the cardinality is 1:n or 1..\*, because a book can have 1 or n editions, but one edition belongs to one book.

In the E/R schemas can have redundancies, and this can imply inconsistency. An element is redundant when can be deleted, without losing semantic. The redundancy can exit in the attributes or in the relationships. An relationship is redundant if this does a cycle. It is possible other type of constraints, as exclusivity, exclusion, inclusiveness, and inclusion.

When the E/R model is obtained, the next phase is the Relational model (Logical Model) [9] [10]. The Relational Model wants to isolate to the user of the physical structures of the data. But, it is further from user, but closer to DBMS. The RM (Relational Model) is a evolution of the E/R Model. The basic structure is the relation (or commonly named table). The attributes are the properties of the tables (relations). The attributes on domains are defined. Each row of the table tuple is called. The number of rows cardinality is called.

Constraints:

* It can’t have two tuples equals.
* The order of the tuples doesn’t import.
* The order of the attributes is not important.
* Each attribute can have only a value of the simple domain (it isn’t accept repetitive groups)[[4]](#footnote-4).
* If an attribute is an identifier, this cannot be null.
* Constraint of interrelation (or foreign key).

In a way parallel, it is possible to analyze the dependencies in the data. The dependencies are inherent properties with the semantic of the data. They consist of the constraints of the user and they have to be kept. There are a set of mathematical theories, as the Armstrong’s axioms [11], but this topic is out of this document. However, the normal forms more used are explained [12] [13]. It is in the Relational Model where this study is established.

* 1NF. It is said that a relationship is in 1NF when each attribute only takes a single underlying value of the domain.

For example the entity “Student” with the attributes (“Code”, “name”, “Courses”). I we suppose next table with next tuples:

**Student**

|  |  |  |
| --- | --- | --- |
| **Code** | **Name** | **Courses** |
| 10003456 | Johan | ErasmusMetrica |
| 10004567 | Alice | Erasmus |
| 10006678 | Francisco | MetricaDatabases |
| 10007822 | Albert | Databases |

This table, this entity is not in the first normal form because Courses have repetitive groups. In this case, 10003456 studies Erasmus and Merica.

The entity will be in the 1NF:

**Student**

|  |  |  |
| --- | --- | --- |
| **Code** | **Name** | **Courses** |
| 10003456 | Johan | Erasmus |
| 10003456 | Johan | Metrica |
| 10004567 | Alice | Erasmus |
| 10006678 | Francisco | Metrica |
| 10006678 | Francisco | Databases |
| 10007822 | Albert | Databases |

* 2NF. A relation is in the second normal form (2NF) if:
	+ It is in the 1NF, and
	+ Each main attribute has complete dependence on each of the keys.

 For example the entity “Student” has the attributes (“Code”, “Name”, “CourseDate”, “Courses”).

**Student**

|  |  |  |  |
| --- | --- | --- | --- |
| **Code** | **Name** | **CourseDate** | **Courses** |
| 10003456 | Johan | 1/1/2012 | Erasmus |
| 10004567 | Alice | 1/1/2012 | Erasmus |
| 10006678 | Francisco | 1/1/2012 | Metrica |
| 10007822 | Albert | 1/1/2012 | Databases |

This relation is not in the 2NF because Code🡪Name, and Code, CourseDate—Courses.

However, next relations will be in the 2NF:

**Student**

|  |  |
| --- | --- |
| **Code** | **Name** |
| 10003456 | Johan |
| 10004567 | Alice |
| 10006678 | Francisco |
| 10007822 | Albert |

**Student**

|  |  |  |
| --- | --- | --- |
| **Code** | **CourseDate** | **Courses** |
| 10003456 | 1/1/2012 | Erasmus |
| 10004567 | 1/1/2012 | Erasmus |
| 10006678 | 1/1/2012 | Metrica |
| 10007822 | 1/1/2012 | Databases |

* 3NF. A relation is in the third normal form (3NF) if:
	+ It is in the 2NF, and
	+ There is no attribute not main that depends transitively of some key.

For example the relation “Project” with the attributes (“Employee”, “ProjectCode”, “ProjectName”). As example:

**Project**

|  |  |  |
| --- | --- | --- |
| **Employee** | **ProjectCode** | **ProjectName** |
| John | BTS | BizTalk |
| Albert | PTR | Preton Server |
| Ignacio | BTS | BizTalk |
| Alice | PTR | Preton Server |

This relation is not in the 2NF, because Employee🡪ProyectCode, and ProjectCode🡪ProjectName.

However, next relation is in the 3NF:

**EmplProject**

|  |  |
| --- | --- |
| **Employee** | **ProjectCode** |
| John | BTS |
| Albert | PTR |
| Ignacio | BTS |
| Alice | PTR |

**Project**

|  |  |
| --- | --- |
| **ProjectCode** | **ProjectName** |
| BTS | BizTalk |
| PTR | Preton Server |
| BTS | BizTalk |
| PTR | Preton Server |

And the last normal form, and more odd is the Boyce-Codd Normal Form [14].

* BCNF. A relation is in BCNF if and only if, all determinant is a candidate key.

Let be the entity “Studies” with the attributes (“Student”, “Course”, “Teacher”).

**Studies**

|  |  |  |
| --- | --- | --- |
| **Student** | **Course** | **Teacher** |
| Jones | Physics | Newton |
| Walker | Physics | Maxwell |
| Jones | Maths | Hilbert |
| Smith | Physics | Newton |

This relation is in the 3NF, because the dependencies are Student, Course🡪Teacher, and Teacher🡪Course. Because, there is no attribute not main that depends transitively of some key. However is not in the BCNF, because all determinant is a candidate key. This means that we can have semantic anomalies, as in the example, where teacher is not candidate key.

It is advisable that the design is in the 3NF (especially de BCNF is odd). We can be and exception when an entity has little tuples. For example:

**Employee**

|  |  |  |
| --- | --- | --- |
| **Name** | **Department** | **City** |
| Jones | Physics | 30111 |
| Walker | Physics | 30111 |
| Jones | Maths | 22222 |
| Smith | Physics | 30111 |

We suppose 15,000 employees and city only two:

**Cities**

|  |  |
| --- | --- |
| **Code** | **Name** |
| 30111 | Sevilla |
| 22222 | Frankfurt |

Then we will be in the 2NF:

**Employee**

|  |  |  |
| --- | --- | --- |
| **Name** | **Department** | **City** |
| Jones | Physics | Sevilla |
| Walker | Physics | Sevilla |
| Jones | Maths | Frankfurt |
| Smith | Physics | Sevilla |

And, at last, we have to map from the Relational model to Physical Model in the DBMS chosen.

1. **Multidimensional Data Model.**

OLAP is a term invented to describe a dimensional approach to interactive decision support (analysis from the perspective of its components or dimensions, considering also the different levels or hierarchies that they have) [15] [17].

As it is possible to see in the figure 1, in the multidimensional design there are three phases. The first, is the conceptual model, or Multidimensional Data Model (MDM). The second is the logical model or MOLAP/ROLAP/HOLAP. And the third is the physical model, the implementation in a specific DBMS.

In the MDM the static elements are:

* The fact schema (the objects to analyze),
* Attributes of fact or of synthesis or measures,
* Summary functions or hash (Statistical type functions that apply to the attributes of fact),
* Dimensions,
* Attributes of Dimension or classification,
* Hierarchies (several dimension attributes linked by a hierarchical relationship, as day->week->month->year).

In figure 3 can be saw these definitions a graphic format.

Figure 3.- Multidimensional data model.

Another example in the figure 4 is shown (Is a report XBRL).

An example in figure 5 is shown. This example a snowflake model is shown [16]. This example belongs to an XBRL report (the same figure 4).



Example of a snowflake model.

Figure 4. Example of an analysis of a cube.

The Logical model is equivalent to Relational model, but not the same.

 The multidimensional data model is not associated with a physical representation of data-> can be supported over a relational DBMS (ROLAP).

The models can be in:

* Star model, or
* Model snowflake, or
* Constellation model

It is possible to generalize the normal form, because if not it is possible to have aggregation anomalies (similar to anomalies in the Relational model) and if not, it is possible to have problems in the queries.

In summary the normal form must to be until 3NF in the logical model, including the exception of the before section [18] [19].

At the end the physical model.

1. **Data Point Model.**

The typical phases are the Universe of the Discourse, Conceptual Model, Logical Model, and Physical Model.

The end users, the regulators create from the Universe of the Discourse (UD) the templates. IT analyses the requirements of this set of templates.

Functional analysts prepare, analyze the conceptual model, in our case the Data Point Model (DPM). In others environments instead of the DPM is use the Multidimensional Data Model (MDM), or Entity Relationship Model (E/R Model). In a conceptual model isn’t referenced the software or the hardware. It is conceptual because the concepts are obtained. By example, *Total assets per quarter*, the designer don’t know the precision, if it is numeric, real, or is per cent, or a string. However, the designer knows that the total per quarter is a monetary quantity. And moreover, he knows, by example, through the UD that *Total assets per quarter* is the sum of the asset *A2* and the asset *A3*.

How is it implemented in the software? It is the same. However, through a CAD tool is possible to do a feedback, more or less in an automatic way. That is to say, it is necessary to do feedback from the DPM to templates and vice versa, but almost automatically in theory. On the other hand, *Total assets per quarter (A1),* *Total assets per semester (A2),* and *Total assets by removing the revenue account (A3)* from the point of view of the business are assets, but three assets have different meaning. For the users of the business, the regulators are three different concepts. These three concepts in some way are related, according to the specifications of the users. In this level, everything related with the implementation of the application disturbs the design, and it doesn’t add anything to this design. Then, when more clear is the design conceptual is easier the feedback with the UD.

In the conceptual model are created:

1. DPM: Dimensions, Primary items, members,…
2. MDM: Facts, attributes of the facts, dimensions, hierarchies,…
3. ERM: Entities, attributes, interrelationship, constraints.

When the conceptual level is designed, the analyst has to approach to the implementation, that is to say, the logical model is implemented.

1. XBRL: Types of data, precisión, …
2. MOLAP/ROLAP/…: Organization, primary keys, foreign keys, …
3. Relational: Tables, references, constrainsts.

The idea is the same, it is to do feedback between the logical model and the conceptual model, but not the UD.

And at the end the implementation in the taxonomy, that is to say the physical model.

1. **Conclusions.**

The DPM is a data model, and at the end it defines a structure of metadata. And the design of the data model must have the same structure, because both the multidimensional model, as the E / R have long been developing and universities.

1. **Bibliography.**

[1] Bouzeghoub, M., Kedad, Z., Métais, E. “CASE Tools: Computer Support for Conceptual Modeling”, chapter 13 of “Advanced Database Technology and Design” of Piattini, M. and Díaz Oscar. Ed. Artech House, Inc. 2000.

[2] Santos, I.J., Martínez, P., Cuadra, M. D. On the semi-automatic validation and decomposition of ternary relationships with optional elements. In Proceedings of the 9th International Conference on Enterprise Information Systems, June 12th -16th, 2007, Funchal, Madeira - Portugal. Pages: 465-472.

[3] NLDB’2000 5th International conference on Applications of Natural Language to Information Systems. Versailles (France), June 28th -30th, 2000.

[4] Santos Forner, I.J., Martínez, P., Cuadra, D. On the Semi-Automatic Validation and Decomposition of Ternary Relationship. IADIS International conference, Applied Computing 2006. February, 25th -28th, 2006, San Sebastian, Spain.

[5] Thalheim, Bernhard 2000. Entity-relationship modeling: foundations of database technology. Publishing: Springer

[6] Teorey, Toby J. Data Base Modeling and Design, Third Edition. San Francisco, Mogan Kaufmann, 1999.

[7] Robert J. Muller. Database Design for Smarties (Using UML for Data Modeling). San Francisco, Mogan Kaufmann, 1999.

[8] Chen, Peter Pin-Shan. The entity-relationship model-Toward a unified view of data. ACM Transactions on Database systems 1 (1):9-36, 1976.

[9] Date, C.J. An Introduction to Database Systems, Second Edition. 1977. Reading. MA: Addisson-Wesley.

[10] Teorey, Toby J., 1999. Database Modeling and Design, Third Edition. Mogan Kaufmann Publishers.

[11] Armstrong, W.W. Dependency Structures of Data Base Relationships. IFIP 74,

North-Holland Pub. Co. Amsterdam, 1974, pp. 580-583.

[12] Millist W. Vincent. 1994. PH. D. Thesis, “Semantic Justification for Normal Forms in Relation Database Design”, Department of Computer Science, Monasch University.

[13] Codd, E. F. (1970). A Relational Model of Data for Large Shared Data Banks. Comunications of ACM. Volumen13, number 6, June 1970.

[14] Codd, E. F. "Recent Investigations into Relational Data Base Systems." IBM Research Report RJ1385 (April 23, 1974). Republished in Proc. 1974 Congress (Stockholm, Sweden, 1974). New York, N.Y.: North-Holland (1974).

[15] Kimball, Ralph C. The data warehouse toolkit : the complete guide to dimensional modelling. New York [etc.] : John Wiley and Sons, 2002.

[16] Ignacio Santos and Elena Castro. XBRL Interoperability through a Multidimensional Data Model. December 8th-10th, 2011.. IADIS International Conference on Internet Technologies & Society (ITS 2011) Shanghai, China.

[17] W. H. Inmon. Building the Data Warehouse. 4th Edition. John Wiley & Sons 2005.

[18] Bodo Hüsemann, Jens Lechtenbörger, and Gottfried Vossen.Conceptual. Data Warehouse Design. June 5th, 2000. Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW’2000). Stockholm, Sweden, June 5-6, 2000.

[19] W. Lehner, J. Albrecht, y H. Wedekind. Normal forms for multidimensional databases. Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW’2000), Stockholm, Sweden, June 5-6, 2000.

1. Multidimensional Online Analytical Processing. [↑](#footnote-ref-1)
2. Relational Online Analytical Processing. [↑](#footnote-ref-2)
3. Hybrid Online Analytical Process. [↑](#footnote-ref-3)
4. First normal form. [↑](#footnote-ref-4)