Accepted for publication in the WITS-'93 Conference Proceedings, December 1993
Orlando, Florida

An Object-Oriented Implementation of Quality Data Products

October 1993 TDQM-93-14

Richard Y. Wang
M. P. Reddy
Amar Gupta

Total Data Quality Management (TDQM) Research Program
Room E53-320, Sloan School of Management
Massachusetts Institute of Technlogy
Cambridge, MA 02139 USA
Tel: 617-253-2656
Fax: 617-253-3321

Acknowledgments Work reported herein has been supported, in part, by MIT's Total Data Quality Management (TDQM) Research Program, MIT's International Financial Services Research Center (IFSRC), Fujitsu Personal Systems, Inc. and Bull-HN.

1. INTRODUCTION

Data which were viewed as proprietary assets are increasingly being treated as off-the-shelf *data products*. This raises a number of research issues. For example, if the data products procured by data consumers have poor quality, the consequence could be serious, and in some cases disastrous. To avoid problems due to poor data quality, the data consumer must be informed of the quality of data products. The challenge lies in the design and production of off-the-shelf data products that will enable data consumers to make their own judgment about the quality.

To understand issues involved in the development of quality data products, it is useful to observe other off-the-shelf products in the market. Consider a medical product such as Tylenol. It is labeled with a list of its ingredients, possible side effects, expiration date, storage instructions, instructions for its usage, and authorization for its usage, etc. This type of information is typically associated with a medical product. Some other products, such as automobiles, can automatically check the quality of their components like the status of the battery, and if necessary, alert the driver that the power level is below the minimum threshold value. Similarly, one could associate quality information with a data product in such a way that the product could assess its own quality and inform data consumers (or a data administrator) if the quality was below a certain threshold value. Other important issues that need to be addressed include building a more complex data product based on data products that are available in the market, developing a design methodology for quality data products, and creating new techniques for producing quality data products.

This paper is organized as follows: Section 2 reviews related work. Section 3 models quality data products. Section 4 presents a design methodology for quality data products. Section 5 shows how quality data object, which is the building block for a data product, can be implemented using the object-oriented approach. Section 6 concludes the paper.

2. RELATED WORK

The quality of data in a conventional database management system (DBMS) has been treated implicitly through functions such as recovery, concurrency, integrity, and security control [Chen, 1976; Codd, 1979; Bernstein & Goodman, 1981; Fernandez, Summers, & Wood, 1981; Ullman, 1982] . These functions are necessary, but not sufficient, to ensure data quality in the database from the data consumer's perspective [Laudon, 1986; Liepins & Uppuluri, 1990; Redman, 1992; Wang, Kon, & Madnick, 1993] . Integrity constraints and validity checks, for example, are essential to ensuring data quality in a database, but they are often not sufficient to win consumers' confidence on data [Maxwell, 1989] . In general, data in a DBMS are used by a range of different organizational functions with different perceptions of what constitutes quality data in terms of dimensions such as accuracy, completeness, consistency, and timeliness [Ballou & Pazer, 1985] . It is not possible to manage data such that they meet the quality requirements of all their consumers. Data quality must be calibrated in a manner that enable consumers to use their own yardsticks to measure the quality of data. None of the existing DBMSs has the capability to explicitly represent the quality of data or to allow consumers to measure the quality of data.

Several recent research efforts have aimed to address the issue of explicitly representing the quality information. An attribute-based research that facilitates cell-level tagging of data to enable consumers to retrieve data that conforms with their quality requirements has been proposed [Wang & Madnick, 1990; Wang, Kon, & Madnick, 1993; Wang, Reddy, & Kon, 1993] ; this research, however, did not address issues involved in measuring data quality dimension values. In other related research efforts that aim at

annotating data, self-describing data files and meta-data management have been proposed at the schema level [McCarthy, 1982; McCarthy, 1984; McCarthy, 1988] . However, no specific solution has been offered either to manipulate such quality information at the instance level or to measure data quality issues. Still other research efforts [Codd, 1979; Siegel & Madnick, 1991] have dealt with data tagging without a set of quality measures for data quality dimensions.

The research question here is how to design and implement data products in such a way that consumers can be equipped with the capabilities to measure the quality of data products they need and to procure data product that conform with the quality requirements of the application at hand. In this paper, we propose a methodology for the design and implementation of data products whose quality can be measured by the consumer of the data product.

3. MODELING QUALITY DATA PRODUCTS

The basic components of a data product are data items. A data item can be as simple as an integer or a string, and as complex as a financial report for a company. A data product is a collection of data items packaged in such a way that it can be readily used. For example, any report generated by any conventional DBMS is a data product. Typically, these data products are not accompanied by their quality information. Here quality means the source, collection method, semantics etc., of each data item. In the absence of such information, it is difficult to understand the meaning, correctness, consistency, and completeness of data in a data product. In general, existing data products have no explicit mechanism for the consumer to evaluate the quality of data.

To overcome this problem, we advocate that the data should always be accompanied with its quality information. Since quality is a dynamic aspect of data, one cannot physically tag the data with a quality value such as high, medium or low. A data product which is of good quality to one consumer may not be good quality to another consumer (e.g., yesterday's stock price of a company may be good enough for a financial analyst but may not be useful to an investor who wants to buy or sell the company's stock). A data product which is good quality today may not be of good quality tomorrow (e.g., a flight schedule which is valid today may be obsolete tomorrow). A data product manufactured by one vendor may not be the same as that produced by another vendor (e.g., one vendor may produce a data product by conducting a survey which covers the entire population of the data product domain, whereas another vendor might survey a small sample of the data product domain and produce a similar data product with extrapolation of results). Therefore each data item should be explicitly tagged with its quality information and should leave the judgment of its quality to the consumer. The quality information should typically consist of the manufacturing process of a data product giving details such as the source of the data, the supplied date and the data collection method adopted by the source, and the semantics of the data item. This will enable a data consumer to judge the

quality of data based on this quality information. In order to evaluate the quality of a data item, the consumer must also be provided with a set of procedures. In general, the quality of a data product is a function of the data consumer, the current state of the data, and its manufacturing process.

We define a quality data item as a data item which is packaged along with a set of quality indicators and a set of procedures which can evaluate whether a data item is of the quality desired by the consumer. Further we define a quality data product as a data product in which each data item is a quality data item. Critical issues in the production of a quality data item, which is a building block for a quality data product, include:

(a) How do we identify the required set of quality indicators, for a given data item ?

(b) How do we develop methods which can evaluate the quality of the data item with respect to the consumer's criteria?

(c) How do we package a data item along with its quality indicators and quality methods as a single unit?

The following two sections address these issues.

4. A DESIGN METHODOLOGY OF QUALITY DATA PRODUCTS

In order to develop a data product, one must first design a data product. Figure 1 shows the steps involved in the analysis and design of a data product [Wang, Kon, & Madnick, 1993] . Step 1 is the traditional data modeling process [Batini, Lenzirini, & Navathe, 1986; Teorey, Yang, & Fry, 1986; Teorey, 1990; Navathe, Batini, & Ceri, 1992] . The data quality requirement analysis begins once the application schema of the data product is designed. It is an effort similar in spirit to traditional data requirements analysis, but focusing on the quality aspects of the data.

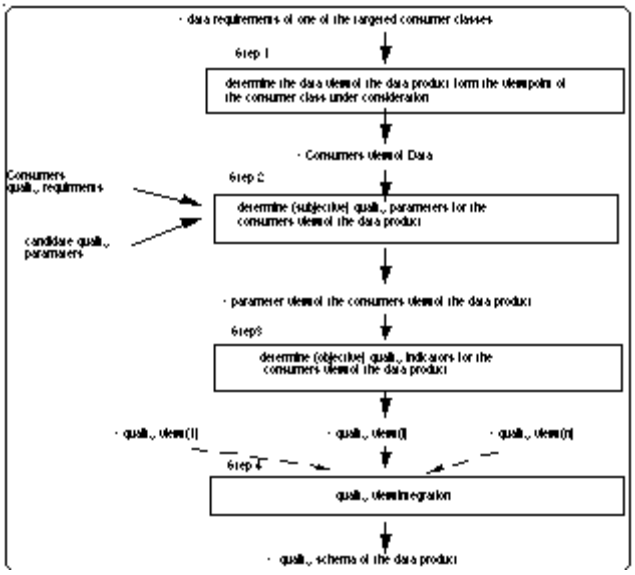

Figure 1: The process of data product design

For illustration purpose, suppose that we are interested in designing a data product STOCK_INFO. A stock has a SHARE_PRICE, a STOCK_EXCHANGE (NYSE, AMS, or OTC), and a TICKER_SYMBOL. An ER diagram that documents the application view of a data product for our running example is shown in Figure 2.

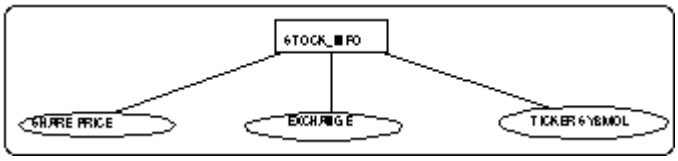

Figure 2: Base data view of the data product (output from Step 1)

The goal of step 2 is to elicit subjective quality parameters [Wang, Kon, & Madnick, 1993] from the consumers of the data product. These parameters need to be gathered from consumers in a systematic way. The application view of the data product must be analyzed with respect to each quality dimension. Figure 3 illustrates the addition of three quality parameters, *interpretability*, *credibility*, and *timeliness* to the base schema of the data product. Each quality parameter identified is shown inside a "cloud" in the diagram. *Timeliness*, in turn, can be defined through *currency* and *volatility*.

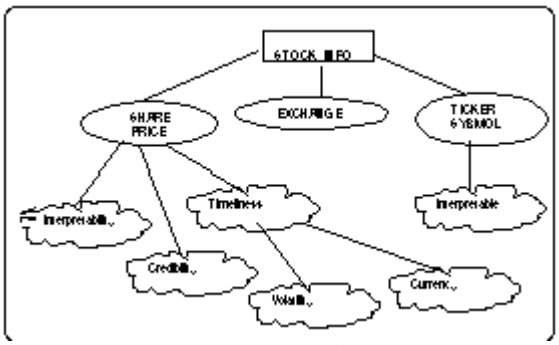

Figure 3: Parameter view of data product

The goal in Step 3 is to operationalize the primarily subjective quality parameters identified in Step 2 into objective quality indicators. Each quality indicator is depicted as a tag (indicated as dotted-rectangle) and is attached to the corresponding quality parameter (from Step 2) to create the quality view. The portion of the quality view for the stock entity in the running example is shown in Figure 4. Corresponding to the quality parameter *interpretability* are the more objective quality indicators, *currency unit* in which SHARE_PRICE is measured (e.g., $ vs. ¥) and *status* which determines whether the SHARE_PRICE is the latest closing price or the latest nominal price.

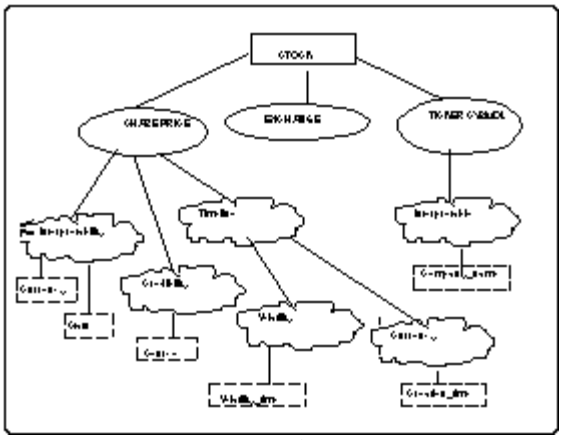

Figure 4: Quality view for STOCK _INFO

All quality views are integrated in Step 4 to generate the quality schema of a data product. When the design is large and more than one set of consumer requirements are involved, multiple quality views may result. To eliminate redundancy and inconsistency, these quality views must be consolidated into a single global view, in a process similar to schema integration [Batini, Lenzirini, & Navathe, 1986] , so that a variety of data quality requirements can be met. The resulting single global view is called the quality schema.

The above procedure to specify data quality requirements is a prerequisite for gaining the understanding for the implementation of the quality data product. The enriched schema facilitates the consumer of the data product to assess the quality of data product with respect to each quality dimension and to see whether the quality of data satisfies specific quality requirements. During the production of a data product, both base-data schema and its quality schema must be populated with appropriate data and should procedures to measure the value for each quality dimension for the data product. Now the challenge lies in how to represent and process the quality information.

5. IMPLEMENTATION OF A QUALITY DATA PRODUCT

A quality data product is a set of (aggregated) quality data objects. For example, the data product STOCK_INFO is an aggregation of three quality data objects namely SHARE_PRICE, EXCHANGE, and TICKER_SYMBOL. A quality data object is a composite object constructed from a datum object and its associated quality description object. Each datum is modeled as an object called a datum object. As shown in Figure 5, the quality information corresponding to the datum is modeled as a quality description object. The *is-a-quality-of* link associates a quality description object with its datum object. The composite object constructed from a datum object and its associated quality description object is called a quality data object. Instance variables of a quality

description object include descriptive data (quality_indicatori, i= 1, ..., n) and procedural data (quality_procedurej, j= 1, ..., m).

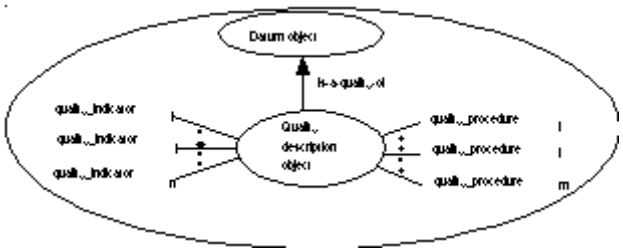

Figure 5: Quality Data Object

A quality data object namely Share_Price is exemplified in Figure 6.

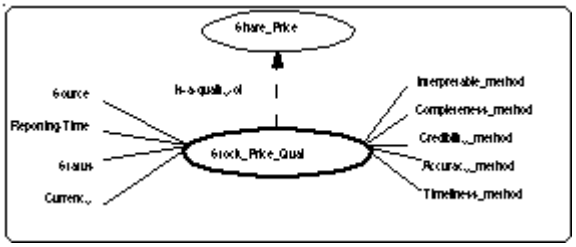

Figure 6: Quality Data Object Share_Price

In the following subsections, the concept of quality data object is presented in terms of its structure and behavior.

5.1. Structure of Quality Data Object

Let I denote the set of system generated identifiers, and B denote the set of base atomic types such as integer, real, or string. Following the object structure defined in the object-oriented paradigm [Maier & Stein, 1987; Khoshafian & Copeland, 1990] , we define two object types for the quality data object.

(a) An object is defined as a primitive object provided that its value belongs to B. The value of a primitive object cannot be further subdivided. In the context of the quality data object, every datum object is a primitive object.

(b) An object is defined as a tuple object if its value is of the form •a1:i1, a2:i2, ..., an:in® where ai's are distinct attribute names and ii's are distinct identifiers from I. In the context of the quality data object, every quality description object is a tuple object.

As shown in Figure 5, the quality description object is associated with its datum object through a *is-a-quality-of* link. The composite object resulting from this association is defined as a quality data object which is a unit of manipulation. As such, every quality data object is a composite object. This composite property can be nested in an arbitrary number of levels.

Note that there is no specific mechanism in the object-oriented paradigm to associate the quality description object with the primitive datum object. More specifically, neither the generalization (*is-a*) nor the aggregation (*is-a-part-of*) construct can be used to capture the semantics of the *is-a-quality-of* link. The *is-a* link is used to associate a subclass object with its super class object; and the *is-a-part-of* link is used to associate an object with its assembly object [Banerjee, 1987] . The *is-a-quality-of* link is conceptually

different from *is-a* because the relationship between a datum object and its quality description object is not a super-class versus subclass relationship and different from *is-a-part-of* because *is-a-quality-of* relationship is not a part and assembly relationship. The quality description object is treated as a weak object and its existence depends on the existence of its corresponding datum object.

We have presented the quality data object in terms of its structure. Though the *is-a-quality-of* construct is unique to the quality data object, other constructs in the object-oriented paradigm such as generalization/specialization and aggregation can be used to construct a quality data product schema. The next subsection presents the behavior of the quality data object that will address the issues of how to measure the quality of data.

5.2. Behavior of quality data object

In general, the behavior of an object is encapsulated in its methods and messages. In the context of the quality data object, both datum objects and quality description objects need methods for their creation, deletion, and update, just like objects in the object-oriented paradigm. Only methods and messages relevant to the data quality aspect are presented in this paper. Below we define key methods that measure data quality. It is important to note that the following methods constitute a subset of the many ways to evaluate data quality dimensions**.** These methods presented below must be modified based on the nature of the data and its application.

5.2.1. Interpretability

Mis-interpretation of data causes serious data quality errors. Providing universal semantic interpretability for a data item is difficult and this problem has been studied at the schema level [McCarthy, 1982; McCarthy, 1984; McCarthy, 1988]. In this paper, we provide semantic knowledge that is sufficient for the set of consumers of the data product to understand the meaning of the base-data of the quality data object. We choose to represent this knowledge as quality_indicator values. The meaning of each datum is captured by a set of quality indicators called semantic_quality_indicators. If the value of any quality_indicator is not self-explanatory then it will be characterized by its own set of semantic_quality_indicators. These semantic quality indicators facilitate the consumer to use the data in more meaningful ways which is very important from the data quality's viewpoint. The interpretability method described here presents value of the semantic_quality_indicators of the base_data of the quality data object upon the request of the data consumer. For example, the interpretability method of SHARE_PRICE object returns its exchange and its currency units (see Figure 4).

5.2.2. Currency

Currency is a measure which gives the current age of data. Data_origination_time should be one of the quality indicators identified during quality requirement analysis for every datum object whose quality is based on its currency. This time stamp reflects the time at which the data came into existence in the real world. Currency method calculates the age of data from this quality indicator. Let to be the data_origination_time of the datum and let tc be the current_time. The age of the datum is given by tc - to. We propose to measure currency on a continuous scale from 0 to 1. State 0 would be assigned to data that are as current as possible, state 1 to the oldest stored data. Let C represent the measure for currency (0 ² C ² 1). The value of C is computed dynamically using the

data_origination_time of the instance. Depending on the message, the currency method can determine the currency of an individual instance, or the average currency of the instances of the class.

5.2.3. Volatility

The volatility of data is an intrinsic property of the data which is unrelated to its storage time. For example, the fact that George Washington was the first president of the United States remains true no matter how long ago that fact was recorded. On the other hand, yesterday's stock quote may be woefully out of date. We propose to measure volatility on a continuous scale from 0 to 1 where state 0 refers to data that are not volatile at all (they do not change over time) and 1 refers to data that are constantly in flux. Volatility may be static or dynamic. In a static case, a quality indicator is created to give volatility of the datum object. Whenever consumer checks the volatility of the datum object the volatility method returns 1 if the datum object is valid and otherwise returns zero. In the dynamic case, volatility is measured as a mean time between successive updates. The volatility method monitors updates to the value of an instance variable and computes the mean time between successive updates. This time is used to judge the volatility of a data product.

5.2.4. Timeliness

Timeliness is defined as a function of currency and volatility of a data value. The most stable situation is to have data for which the currency is 0 (entered very recently) or the volatility of 0 (unchanging) or both. For such data, there is no timeliness problem. The worst situation arises when data are old (currency = 1) and highly volatile (volatility = 1). As such, the timeliness is a function of currency and volatility. This function should be defined based on application at hand. One measure of timeliness is obtained by

combining currency and volatility via their root-mean square:  where 0 ² T ² 1 with 0 representing as the best and 1 representing as the worst case. The Timeliness method computes the timeliness of a datum object from its currency and volatility values.

5.2.5. Accuracy

Accuracy is a measure that is most desired and difficult to quantify. Accuracy is defined as "the recorded value in conformance with its true value in the real world". The data object cannot track down the true values in the real world. Therefore the above definition cannot be directly used to compute the accuracy of a data item. As such, the accuracy method expects from the consumer either the surrogates of true values in the real world or general behavior/rules that the true values obey in the real world. Taking this information from the consumer, the accuracy method either compares the recorded instances of the object with that of consumer supplied instances and returns the percentage of match, or returns the percentage of recorded instances that obey the rules given by the consumer. In general, a consumer can test the accuracy of the data supplied by a data product with a set of sample data considered to be accurate by the consumer. For example, a consumer who wants to check the accuracy of a payroll data product can

first check whether his or her salary (known data) is recorded correctly by the data product or not. On the basis of this test, the consumer can make judgment about the accuracy of data supplied by a data product.

5.2.6. Completeness

Completeness, involves two levels: data product level and data instance level. Data product level completeness gives the ratio of the number of instances that can be supplied by the data product to the total number of instances the data product is expected to supply. It is very difficult to quantify this measure. If the data product manufacturer possesses knowledge about the number of missing instances of the data product, the manufacturer could tag it as a quality indicator at the data product level. Instance level completeness can be quantified as the ratio of the number of instances in which at least one of the components is missing to the total number of instances of the object.

5.2.7. Credibility

The credibility of a datum object is computed based on: (i) the quality indicator values present in the quality description object of the datum; and (ii) the set of specifications given by the consumer. Let x be an instance. Let qi be the ith quality indicator of x and let `J' be the number of quality indictors the consumer wants to use to compute the credibility of x. Let uvi be the consumer's specified value for qi and let rvi be the recorded value of the quality indicator qi for x in the quality data object. Let wi be the credibility weight assigned to the quality indicator qi by the consumer such that $\sum_{i=1}^{J} w_i = 1$ Let δi be a binary variable defined as follows: δi =1 if uvi =rvi else δi =0. The credibility of x is computed by the following expression:

$$\sum_{i=1}^{J} w_i * \delta$$

.

The method credibility returns an instance value and its associated credibility.

In the above paragraphs, we have presented methods measuring the key dimensions of data quality. They define an important part of the behavior of the quality data object. The other critical behavioral component of the quality data object is the capability of self quality assessment which is discussed in the following subsection.

5.3. Data Quality Demons

One of the difficulties with the existing data products is the task of separating bad quality data from good quality data. Data products should encapsulate demons which monitor the quality of the base data of the quality data object. In other words, it should be the responsibility of the quality data object to evaluate its current status with respect to the pre defined set of quality methods. If the quality of its state is below the expected value then it should request the data quality administrator to update its status to reflect its real world counter-part. In the existing data products, the data quality administrator needs to constantly monitor the entire set of data associated with the data product to ensure its quality. One good example for such a demon is a consistency demon. If a set of quality objects has functional relationships and if the state of any one of the objects in the set changes, then the demon verifies the functional relation. If the functional

relationship is violated, it informs the data quality administrator to check the status of all objects involved in the functional relation or the correctness of the relation itself.

6. CONCLUSIONS

In this paper, we have investigated how to associate data with quality information that can help consumers make judgments of the quality of data for the specific application at hand. Our research question was how to structure and manage data in such a way that consumers could be equipped with the capabilities to measure the quality of data they need and to retrieve the data that conforms with their quality requirements.

Toward this goal, we have proposed the concept of quality data object in which each datum object is associated with appropriate data and procedures used to indicate the quality of the datum object. Specifically, the *is-a-quality-of* link is proposed to associate a datum object with its corresponding quality description object. The composite object constructed from a datum object and its associated quality description object provides methods which can access object instances which matches consumers' quality requirements. It also provides a set of quality measure methods that compute quality dimension values including currency, volatility, timeliness, accuracy, consistency, and completeness.

The concept of quality data object presented in the paper is a first step toward the design and manufacture of data products. We envision that the quality data objects proposed in this paper can be used as basic building blocks for the design, manufacture, and delivery of quality data products. This will enable consumers to measure the quality of data products according to their chosen criteria; and to procure data products based on their quality requirements, hopefully enhancing overall data quality and data reusability. We are currently working to provide a more concrete definition for a data product and to crystallize its characteristics in further detail.

7. REFERENCES

[1] Ballou, D. P. & Pazer, H. L. (1985). Modeling Data and Process Quality in Multi-input, Multi-output Information Systems. *Management Science*, 31(2), 150-162.

[2] Banerjee, J., et al, (1987). Data Model Issues for Object-Oriented Applications. *ACM Transactions on Office Information Systems*, 5(1).

[3] Batini, C., Lenzirini, M., & Navathe, S. (1986). A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys*, 18(4), 323 - 364.

[4] Bernstein, P. A. & Goodman, N. (1981). Concurrency Control in Distributed Database Systems. *Computing Surveys*, 13(2), 185-221.

[5] Chen, P. P. (1976). The Entity-Relationship Model - Toward a Unified View of Data. *ACM Transactions on Database Systems*, 1, 166-193.

[6] Codd, E. F. (1979). Extending the relational database model to capture more meaning. *ACM Transactions on Database Systems*, 4(4), 397-434.

[7] Fernandez, E. B., Summers, R. C., & Wood, C. (1981). *Database Security and Integrity.* Reading: Addison-Wesley.

[8] Khoshafian, S. N. & Copeland, G. P. (1990). Object Identity, Readings in Object Oriented Database Systems, Morgan. Kaufmann Publisher, San Mateo CA.

[9] Laudon, K. C. (1986). Data Quality and Due Process in Large Interorganizational Record Systems. *Communications of the ACM*, 29(1), 4-11.

[10] Liepins, G. E. & Uppuluri, V. R. R. (Ed.). (1990). *Data Quality Control: Theory and Pragmatics*. New York: Marcel Dekker, Inc.

[11] Maier, D. & Stein, J. (1987). *Development and Implementation of an Object-Oriented DBMS.* Cambridge, MA: MIT Press.

[12] Maxwell, B. S. (1989). Beyond "Data Validity": Improving the Quality of HRIS Data. *Personnel*, 66(4), 48-58.

[13] McCarthy, J. L. (1982). Metadata Management for Large Statistical Databases. In *Proceedings of the 8th International Conference on Very Large Data bases (VLDB)*, (pp. 234-243) Mexico City.

[14] McCarthy, J. L. (1984). *Scientific Information = Data + Meta-data.* U.S. Naval Postgraduate School.

[15] McCarthy, J. L. (1988). The Automated Data Thesaurus: A New Tool for Scientific Information. In *the Proceedings of the 11th International Codata Conference*, Karlsruhe, Germany.

[16] Navathe, S., Batini, C., & Ceri, S. (1992). *The Entity Relationship Approach.* New York: Wiley and Sons.

[17] Redman, T. C. (1992). *Data Quality: Management and Technology.* New York: Bantam Books.

[18] Siegel, M. & Madnick, S. (1991). Context Interchange: Sharing the Meaning of Data. *SIGMOD Record, ACM Press*, 20(4), 77-79 (December).

[19] Teorey, T. J. (1990). *Database Modeling and Design: The Entity-Relationship Approach.* San Mateo, CA : Morgan Kaufman Publisher.

[20] Teorey, T. J., Yang, D., & Fry, J. P. (1986). A Logical Design Methodology for Relational Databases using the Extended Entity-Relationship Model. *ACM Computing Surveys*, 18(2), 197-222.

[21] Ullman, J. D. (1982). *Principles of Database Systems.* Rockville, Maryland, Computer Science Press.

[22] Wang, R. Y., Kon, H. B., & Madnick, S. E. (1993). Data Quality Requirements Analysis and Modeling. In *Proceedings of the 9th International Conference on Data Engineering*, Vienna: IEEE Computer Society Press.

[23] Wang, R. Y., Reddy, M. P., & Kon, H. B. (1993). Toward Quality Data: An Attribute-based Approach. *To appear in Journal of Decision Support Systems (DSS)*.

[24] Wang, Y. R. & Madnick, S. E. (1990). A Source Tagging Theory for Heterogeneous Database Systems. In *International Conference on Information Systems*, (pp. 243-256). Copenhagen, Denmark.