Informing Science | Information Sciences | Volume 4 No 2, 2001

# Salvaging Information Engineering Techniques in the Data Warehouse Environment

Anthony L. Politano, CEO MIS-AG

tpolitano@misag.com

## Abstract

*The art of Information Engineering (IE) continuously evolves and, by today's standards, is considered an integral function in most any organization. Strategic planning teams weave methodologies, which are integrated to process information, the goal being to sort, store, and retrieve useful data.*

*The following article will describe three techniques that can utilize existing information engineering in a data warehouse project. First, the entity relationship diagram and its use in a three phase data model approach. Second, the functional decomposition diagram and its use in segmenting and defining key performance indicators and dimensions. Third, creating a modified CRUD (Create, Read, Update and Delete) matrix that deals with logical entities and current systems.*

Keywords: Entity Relationship Diagram, Data Warehouse, Information Engineering, Functional Decomposition, Interaction (CRUD) Analysis. Data Modeling

## Introduction

During the 1980s and early 90s, Information Engineering (IE) was in its prime. Most major corporations were utilizing some form of system development methodology that could be tied back to IE. The first step in any IE project was the Information Strategy Plan or ISP. The ISP would look at the data, process, organization, technology, and interactions of an enterprise. The ISP was top-down analysis at its best. Three key deliverables of ISP were a data model, functional decomposition, and an interaction (CRUD) matrix. The data model was an entity relationship diagram that encompassed the entire enterprise. The functional decomposition diagram would examine the business functions and decompose to a process level. The CRUD diagram would examine the business functions and decompose to a process level. The CRUD matrix examined the interaction of data and process. These three deliverables provided a basis for top-down analysis.

## The Entity Relationship Diagram

The entity relationship diagram is the standard data technique for creating data models. According to Martin, entity relationship diagrams are composed on entities that are defined as something, real or abstract, about which we can store data. (Martin, 1998 p. 297) The entity relationship diagram enables an analyst to create a graphical view of the data concepts of an organization and their relationships. Tradition system development dictates creation of an Entity Relationship diagram that is converted to a database design of a relational database.

In a data warehouse environment, the traditional normalized Entity Relationship cannot be easily translated into a database design. By nature a normalized Entity Relationship diagram tends to separate the data concepts into separate entities. A traditional approach to Entity Relationship modeling is concerned with three concepts: entities, relationships, and attributes.

### *Components of the Entity Relationship Diagram*

There are many works that describe Entity Relationship diagramming in detail. It is not the intention of this work to present exhaustive details; instead a brief description of the component is presented.

- Entity – A data concept that has relevance to the enterprise. An entity can be a person, place, thing, or concept. Typically an entity consists of a single identifiable concept such as EMPLOYEE, STUDENT, CLASS, PURCHASE ORDER, or SHIPMENT. An entity can consist of subtypes. Subtypes are a decomposition of an entity into its various types. For example an EMPLOYEE entity can be modeled with subtypes FULL-TIME and PART-TIME. Subtyping is necessary when clarity is required about the data (and to some respect the behavior) of the Supertype entity.

- Relationship – A relationship, as the name suggests, is a description about the relationship that exists between two entities. Information about how the entities relate, in particular, the optionality and cardinality of the relationship is modeled. A relationship should only be modeled when the relationship has relevance. If one desired, any entity could loosely be related to any other entity, but this is not the intention of modeling relationships. A special relationship, known as a recursive relationship, exists between an entity and itself, such as an EMPLOYEE to EMPLOYEE related by a REPORTS TO relationship.
- Attributes – Attributes are details about a specific entity. These details provide greater clarification about the data that can or will be captured regarding an entity. One must be careful not to confuse entities and attributes. Entities can exist without attributes, but attributes cannot exist without entities.

There are many nomenclatures for the actual diagrams. For demonstration purposes, this paper will adopt the Crow's Foot diagramming technique.

## *Data Modeling for a Data Warehouse*

There has been significant work done on utilizing specialized data modeling techniques for data warehousing. In particular, the dimensional approach has been adapted to model data warehouses for a relational database. As observed by Kimball, the dimensional model used for data warehousing is more asymmetrical with one large dominant table and with other smaller tables joined directly to the centralized table (Kimball, 1996 p. 10) With a dimensional modeling approach, many of the traditional normalization techniques are not utilized. Instead, the model utilizes a mixed approach of highly normalized portions of the model and highly denormalized parts of the model. The model is centered on two types of entities, facts and dimensions. Facts are entities that deal with measurements or indicators. A fact entity for a sales organization could measure revenue per month, or units sold per day. A fact for a manufacturing organization could measure defects per lot per day or units produced per week. Dimensions are entities that represent dimensional information about the facts. Dimensions are ways that the data can be sliced or viewed or segmented. Dimensions typically represent an n-leveled hierarchy such as a product hierarchy or sales organization hierarchy. For example in a sales organization, the SALES TERRITORY can be a dimension that represents the sales territory, its district, the district's region, and the region's division. In a dimensional model, this is one entity. In a traditional Entity Relationship diagram, this would be four entities, TERRITORY, REGION, DISTRICT, DIVISION. The same is true for a product dimension. The product dimension can represent SKU, its brand, the brand's category, the category's division. In a traditional Entity Relationship diagram, this would be represented as four entities, SKU, BRAND, CATEGORY, and DIVISION. Also of note is the absence of data that is purely used in the operational environment. (Inmon, 1993 p. 77)

## *Shortcomings of Dimensional Modeling*

Although the dimensional model is a useful approach for modeling the data needed to create a database, for data warehouses and quick queries there are shortcomings. One of the shortcomings is similar to the problems that traditional Entity Relationship modeling encountered. The data model is designed with an implementation in mind. The dimensional model typically focuses on facts and dimensions for the reporting and analysis needs of the system being designed. The broad-brush approach of traditional Entity Relationship modeling is discounted or ignored. This leads to a myopic view of the data as represented in the dimensional Entity Relationship model.

The dimensional model also pushes business rules and representations to a lower level of abstraction. This can lead to overlooking or discounting business rules that would be much more apparent in a traditional Entity Relationship model. For example, if a sales organization represented a product in the following fashion: A multidivisional company sells some products that are part of consumer brand that is part of category. Some other products, though, do not have a consumer brand, since they are sold through a non-consumer channel. Take a pharmaceutical company that sells prescription and over the counter (OTC) drugs. The OTC may have brands associated with the product and the category may be OTC. The pharmaceutical products may not have a name brand, but are associated with the category or PHARM. Two examples are listed in Table 1.

| SKU | PRODUCT | BRAND | CATEGORY | DIVISION |
|---|---|---|---|---|
| 0001 | Pain-be-gone Ibu-profen Tablets | Pain-be-gone | OTC | Consumer |
| 0002 | Prescription Drug for Headache | N/A | PHARM | Pharmaceutical |

For Division A: SKU aggregates to PRODUCT aggregates to BRAND aggregates to CATEGORY aggregates to DIVISION.

For Division B: SKU aggregates to PRODUCT aggregates to CATEGORY aggregates to DIVISION.

**Table 1: Example of Products**

If this is represented in a traditional Entity Relationship diagram, the diagram could look as shown in Figure 1.

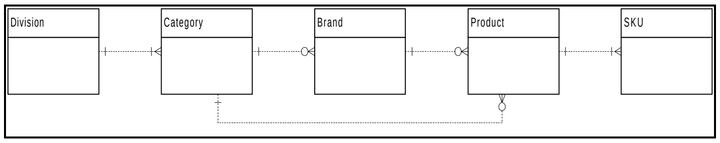

**Figure 1: Traditional Entity Relationship diagram**

While in a dimensional model, the product would be compressed or denormalized into one table. The table would be encoded with attributes that would represent the entities in a hierarchy. The dimensional Entity Relationship would look as shown in Figure 2.

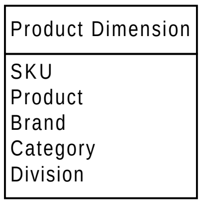

**Figure 2: Dimensional Entity Relationship of Product**

## *The Role of the Two Data Models*

Based on the evaluation of each method, there are uses for each. Yet, each is not complete enough to represent the logical and dimensional data needs. This drives the requirement for a multiphase data model approach. This approach loosely follows a Zachman model (Zachman, 1987 p. 276) of a Conceptual Model and a Logical Model. The conceptual is represented as a traditional Entity Relationship model and the Logical is represented as the dimensional Entity Relationship model. The importance of these separate, but related, models is amplified in a data warehouse environment. In a traditional OLTP environment, normalization is the norm in the conceptual and logical Models. There may be some differences, but generally the models are quite similar in number of entities produced.

In a data warehouse environment, a conceptual view of 25 entities could yield a logical model of 7 entities. The example above compressed the five-entity structure of a product into one entity with many attributes. This compression is a double-edged sword. The concise logical model is much easier to convert to a database structure that is geared toward data warehousing. This logical model, though, hides the business rules in attributes and their optionality and cardinality. It is only through the full examination of the attributes and their allowable values, optionality, and cardinality that the rules are uncovered. Even then, there is not a simple graphical way to represent the model short of representing it as the conceptual model.

This leads to the proposal that in a data warehouse environment, a traditional Entity Relationship model is a business-modeling tool while a logical model is a technology tool.

One may state that this is obvious, and in fact was the practice in traditional Information Engineering SDLC projects. The current problem is that much of the industry has shunned the traditional model as unnecessary. Instead the industry and methodologies are proposing to start with the dimensional model. It is true that the system development starts with the dimensional model, but business data understanding is facilitated through the traditional model.

For many dimensional modelers, this is not enough to justify creating a traditional model. There are other uses, though.

First, the Entity Relationship model can be used as a contextual map of the data connecting the multiple dimensional modeling subject areas. Second, the Entity Relationship model can represent divergent hierarchies in the data. Third, the Entity Relationship model can feed the CRUD matrix, explained below, for a technical scoping and increment management tool.

## The Functional Decomposition

Another major component of information engineering is the functional decomposition diagramming technique. As noted by Martin, in this technique, high-level functions are decomposed to lower level functions; these are decomposed further. (Martin, 1998 p. 165). In this practice, an organization is modeled in terms of the functions for which it is responsible. This follows a strict top-down approach. The enterprise is modeled as one box, which is then decomposed to the next level. This continues until the decomposition reaches a process level. It is at this point, that the diagram ceases being a functional decomposition and instead becomes a process decomposition. It is important at this time to differentiate between a function and a process.

- A function is a set of business activities that does not have a finite start and end point. The function is generally an ongoing effort within a company.
- A process is a business activity that does have a finite start and end point. Processes are generally a lower level of abstraction than a process.

Some examples are:

- Class Registration is a function and Register for a Class is a process.
- Accounts Payable is a function and Create a Check is a process.
- Flight Reservations is a function and Reserve a Seat is a process.

A functional decomposition diagram was useful in an information engineering environment. The functions were decomposed to processes. The processes were converted to program specifications. The program specifications were converted to program code. This was a logical progression.

If done properly, a functional decomposition diagram should represent the entire organization. This representation should categorize functional needs. As data warehousing has shown, the design is user centric and is based on the needs of a particular set of users. By examining the functional decomposition diagram at the leaf (lowest) level, an inquiry should be made as to the measurements of that function. This measurement, usually a fact in a data warehouse, should represent how this function analyzes or measures their function. One should guard against analyzing what data is captured by that function, but instead focus on the measurement of fact.

By organizing at a fact to function level, an organization has a high level understanding of the measurements across the organization. This is done by function also, as compared with by organizational unit.

Suppose an organization has the functional decomposition shown in Figure 3 (greatly simplified for discussion purposes).

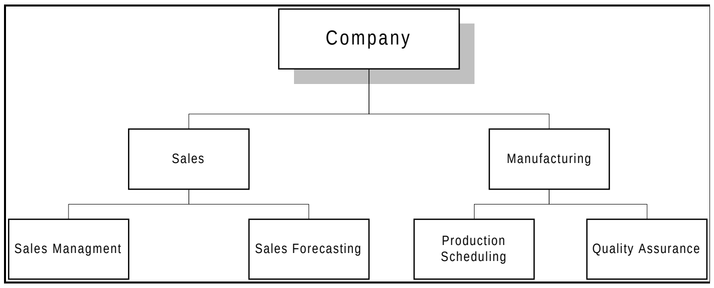

**Figure 3: Functional Decomposition of a Company**

Upon interviewing the respective employees in the functional areas, it is determined that:

- **Sales Management** measures Revenue per Week per Sales Representative.
- **Sales Forecasting** measures Revenue per Month per Region.
- **Production Scheduling** measures Units Produced per Hour per Plant.
- **Quality Assurance** measures Defective Units Percentage per Lot per Plant.

By arranging the captured information in a matrix format where the functions are rows and the measurements are columns, one for each measure and one for each dimension, a concise graphical representation of enterprise high-level reporting needs can be demonstrated, as shown in Table 2.

It must be acknowledged that each functional area will most likely utilize multiple measures and the dimensions should be

| | Measurement | Time | Dimensions |
|---|---|---|---|
| Sales Management | Revenue | Week | Sales Representative |
| Sales Forecasting | Revenue | Month | Region |
| Production Scheduling | Units | Hour | Plant |
| Quality Assurance | Defective Percentage | N/A | Lot, Plant |

**Table 2: Reporting Needs**

more exhaustive than the above example.

Once this analysis is complete and the matrix is validated, it should be examined and analyzed for commonalties. These commonalties can be exploited in system design and scoping as well as having impact on organizational and business process refinement.

## The CRUD Matrix

A CRUD matrix by nature is enterprise wide. It deals with both data and process, but more importantly with their interactions. It is in this interaction that the true relation to data warehousing disciplines becomes relevant.

### *The CRUD for Current Systems*

The CRUD matrix in its traditional ISP use compare functions to entities. For use with a data warehouse, the CRUD should compare current systems to entities. By taking this more physical slant to the analysis, implementation and data redundancy issues are more easily identified.

### *Identification of Systems*

A system should be a logical grouping of programs that support at least one business function. This could be as small as one program or could be an entire purchased systems. For example, an SAP ERP system may have Accounts Payable, Order Entry, and Accounts Receivable. The system should be represented at the Accounts Payable level not the SAP level. This system inventory will most likely be part of the plan since it is required prior to estimating or designing a back end effort.

### *Identification of Entities*

The entities should be representative of data concepts that have meaning to the organization. For most organizations, this can be derived from an enterprise data model, as described above. Usually this model is at the level of CUSTOMER,

| Area | Explanation | Actions |
|---|---|---|
| Multiple creates | If the complete matrix shows that more than one system can create the data, this may be a data quality problem. See Example 1. | The first step is to determine if both the systems are accessing the same physical file(s) or table(s). If they are not, this must be further analyzed to determine the business rules and prioritization rules. If they are accessing the same physical file, the purpose, method and user community must be determined. In particular the aspect of timeliness and consistency must be examined. |
| Creates related to Reads | Most entities will have at least one C and more than one R. See Example 1. | If there is more than one R, each R, must be analyzed to determine: • Is this the same file? • If not what is the replication reason and method, and what is the synchronization method? • If it is the same file what is the business usage? |
| Clustering | By performing an affinity analysis on the rows and columns, groupings of data and process can be found. See Example 2. | Based on the affinity analysis, physical increments of the data warehouse can be defined. A small set of data sources and systems can be outlined to be the increment from a technical perspective. Obviously, this must be balanced with the actual data needs outlined across the organization as defined by a technique such as User Community Segmentation or Measure to Dimension Matrix. |

**Table 3: CRUD Analysis Checks**

ORDER, SHIPMENT, etc. Anything below this level is probably too detailed for this analysis.

## Types of Analysis

By completing a CRUD matrix, data redundancy and synchronization issues are highlighted.

First, standard CRUD analysis (see Table 3) must be done to determine that the checks and balance are present. This analysis includes the following checks:

- *Is there at least one Create?* If not, one must question how the data are being populated.
- *Is there at least one Delete?* If not, one must determine if the business really does require this.
- *Does each system have interaction with at least one entity?* If not, this could indicate that the system's decomposure is too granual. Or, it may indicate that the analysis is not complete. Or, it may indicate that the system is just a big calculator.
- *Does each entity interact with at least one system?* If not, this could indicate that all systems are not listed. Or, it could indicate that the data may be manually stored, such as a file cabinet or index cards.

### Example 1

In the following example, the Customer entity can be created by two systems. Let us assume that the Customer Information System uses a file for Customers, and the Order Entry System uses the Customer Master Table. Since the two systems use different physical resources for the data, a yellow flag must be raised. The data must be examined on a field-by-field basis. The metadata associated with each field must be gathered. If the data sources are similar or the same, a combination strategy must be defined.

The matrix does not define this combination; instead the matrix can be used as an estimating and impact analysis tool. The number of creates/updates that utilize separate data sources assists in determining the amount of effort required when performing the cleansing and loading of the data warehouse. One must caution from drawing a direct relationship between data sources and effort. Instead it is possible that the combination of two data sources may require two units of effort whereas the combination of three data sources may require four or more units of effort. With each added data source comes the increased risk that the fields will be incongruent.

In the example 1, let as also assume that the Sales Tracking and the Billing Systems read from the Customer Master Table and create a local copy. In this case, one must examine this duplication or replication strategy. Some key issues are:

- What is the timing of the replication (daily, hourly, monthly)?
- Is the timing the same for both the Sales Tracking and Billing Systems?
- Is the any filtering or aggregation of the replicated data/
- What is the physical mechanism for replication?

By following a set of questions similar to this, one can determine if this is truly a replication and what impact could be made by combining the reporting capabilities of these "downstream" systems with the envisioned data warehouse. (See Table 4.)

### Example 2

In example 2 there are four technical groups that could relate

| | Customer | Order | Invoice | Sales Person | Check |
|---|---|---|---|---|---|
| Customer Information System | CRUD | R | R | R | |
| Order Entry System | CRU | CRUD | R | R | |
| Sales Tracking | R | R | R | R | |
| Billing System | R | R | R | | |
| Check Writer | | | | | CRUD |

**Table 4: Example 1 - Multiple Reads and Creates**

to the data warehouse increments:

1. Customer/Order
2. Vendor/AP
3. Production Mgt.
4. Inventory/Shipping

Within each of the increments, a mini-analysis can be done to determine redundancy and other impacts. This mini-analysis should be coordinated across the whole matrix to ensure conformity. Note that some of the interactions, such as the Customer to Shipment Tracking Read, do not fall within a box or increment. Even though this is true, the analysis must still include this interaction point when examining the rows or columns. (See Table 5.)

## Integrative Framework

Information Engineering by nature is a data centric methodology. The same is true for current data warehouse methods and projects. Some of the techniques from information engineering can be reused and modified for a pragmatic approach to data warehousing. Three of these techniques are Entity Relationship Diagramming, Functional Decomposition, and Interaction (CRUD) Analysis. These three techniques can be used disjointedly, but are best used in concert. A simple flow chart is shown in Figure 4.

| | Customer | Order | Invoice | Vendor | PO | Check | Employee | Plant | Warehouse | Shipment | Shipper |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Customer Information System | CRUD | R | R | | | | | | | R | |
| Order Entry | CRU | CRUD | R | | | | | | | | |
| Invoicing | R | R | CRUD | | | | | | | | |
| Purchasing | | | | RU | CRUD | R | | | | | |
| Vendor Mgt. | | | | CRUD | R | R | | | | | |
| A/P | | | | R | RU | CRUD | | | | | |
| Warehouse Mgt | | | | | | | R | R | CRUD | R | |
| Production Mgt. | | | | | | | R | CRUD | R | R | |
| Inventory Replenishment | | | | | | | | R | CRUD | | |
| Shipment Tracking | R | | R | | | | | | R | CRUD | R |

**Table 5: Example 2 - Clustering**

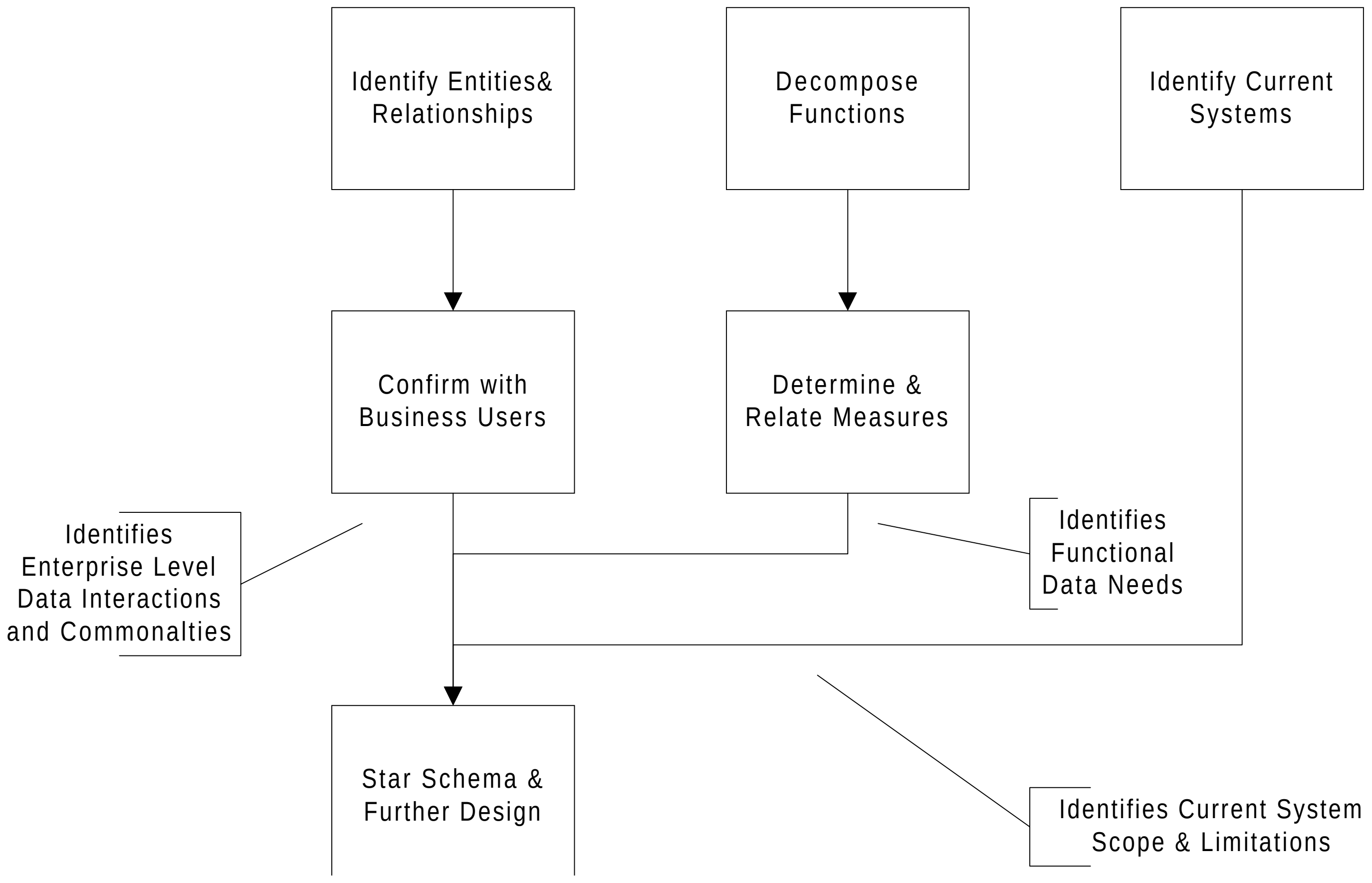
Identify Entities& Relationships
Decompose Functions
Identify Current Systems
Confirm with Business Users
Determine & Relate Measures
Identifies Enterprise Level Data Interactions and Commonalties
Identifies Functional Data Needs
Star Schema & Further Design
Identifies Current System Scope & Limitations

**Figure 4: Flowchart**

In the flowchart, each of the information engineering techniques plays a role in building the star schema. The Entity Relationship Diagram identifies the enterprise level data interactions. It also identifies commonalties and shared data. The Functional Decomposition identifies the functional data needs. One or more of these needs must then become the subject of the iteration of the data warehouse being designed. The CRUD matrix identifies the system limitations and redundancies required to source the data. Based on these three prerequisite tasks, a star schema design can concentrate on the functional information needs without losing sight of the contextual data view. The schema, and subsequent design, can also be scoped to a manageable level of technical complexity based on the star schema.

This is not meant to be an exhaustive methodology. Instead, this article has presented a method for salvaging techniques that are readily available in the current work force. These techniques are also design techniques to be applied in the current data warehousing environment. An analysis at a higher level of abstraction may be necessary and could be done prior to this work.

## Conclusion

When we discuss creating a data warehouse environment, we do so, taking the finest attributes of several of the evolving information engineering methodologies. This article demonstrates that the entity relationship and functional decomposition diagrams, coupled with a modified CRUD matrix, are fortified techniques that complement each other. We have also illustrated by examining two data models, the traditional entity relationship diagram versus dimensional modeling, that each model is justified, yet neither complete to delineate both logistical and dimensional data needs.

Further research on the subject would utilize case studies to examine the maturation of information engineering techniques of organizations. Compare and contrast the methodologies of both successful and not-so practitioners. Interview IT architects for insight into the demands of data warehouse design and challenges being faced. Undoubtedly, progress in technology and methodology will consistently drive data warehousing to unforeseen levels of hierarchy within organizations. Those able to make practical use of the information will emerge.

## References

Inmon, W. (1993). *Building the Data Warehouse*, New York: J. Wiley & Sons.

Kimball R. (1996). *The Data Warehouse Toolkit*. New York: J. Wiley & Sons.

Martin, J. McClure, C. (1988). *Structured Techniques, The Basis for CASE*, Englewood Cliffs, NJ: Prentice Hall.

Zachman, J. (1987). A framework for enterprise architecture, *IBM Systems Journal* 26(3), 276-292.