Data Bryte:
A Data Warehouse Cleansing Framework

Steven D. Mohan, Doctoral Candidate
University of Phoenix
fusion@rmi.net
Mary Jane Willshire, Ph.D.
Software Engineering Management Associates (SEMA)
Mjwillshir@aol.com

Abstract

Data cleansing consumes 60--80% of the effort required to build a functional and effective data warehouse. A new cleansing methodology is described that offers a coherent and focused approach for attaining improved information quality. Quantitative research results indicate that cleaner data can be obtained.

Introduction

Research has demonstrated that 60-80% of the effort of building a usable data warehouse is consumed in data scrubbing [5]. Yet there is very little published research that addresses the actual scrubbing of the data. As pointed out by Allen [1], this scrubbing phase consumes a lot of effort and is not a "sexy" proposition. The fact remains that the data need to be **systematically** scrubbed (validated) in order to build data warehouses and data marts with an acceptable level of data quality that can be searched coherently.

Though a careful management of the relationship between basic accounts and aggregated accounts has been recommended by Wang [6], no one had undertaken to develop an entire data cleansing engineering program from start to finish. In fact, the warehouse developer has been left to attack the data cleansing issues in an intuitive manner, at best [3]. What we propose to show is that a standards-based, data entity cleansing paradigm integrated with a model-based, combined logical data element quality paradigm (that meets the sufficiency requirement of data cleanliness) can be developed that can be used to define, analyze, and provide guidance to improve data quality.

This paper presents data generated in the research effort to design, build and verify a data cleansing framework, called the Data Bryte Framework, for improving data quality in a data

warehouse for industrial use. The framework is outlined and research results are described below. We invite suggestions from the data quality community on this effort.

Research Method

The research to develop, apply, and prove the viability of Data Bryte, a standards and model-based data cleansing framework consisted of six steps, each of which is described briefly in the following paragraphs. The first step develops and describes the general framework for Data Bryte that is utilized to define, analyze, and provide guidance to improve data cleansing quality. The second step defines measures of success (i.e. generate measurable thresholds) in developing and applying Data Bryte. The third step tailors Data Bryte to an existing and/or new database, then applies the tailored Data Bryte to the subject database. Describing and concisely analyzing the results of applying Data Bryte comprised the fourth step, while drawing conclusions from the application of Data Bryte generated the fifth step. The latter two steps determined the viability of Data Bryte. Recommendations were formulated and presented in step six.

Step 1 Describe the Standards/Model-Based Data Cleansing Framework (Data Bryte)

Concepts, paradigms and ideas were obtained from the data cleansing and data quality community. These were so constructed as to scrub data consistently within the context of specific systems (whether automated or not), their functions, the data at hand, and most importantly, the users of the data. As noted by [2] the cleanliness of the data are fundamentally determined by the users and their analysis methods. The Data Bryte system is explained in detail in a previous paper [4].

Step 2 Define Measures of Success

The success measures for the Data Bryte general framework included:

1. Successful definition of data cleansing attributes specific to a users' needs.
2. Successfully tailored to facilitate the cleansing of a specific set or sets of data.
3. Provided guidance to improve the quality of data in a selected case or cases.

Fundamentally, quantitative evidence was utilized to substantiate the findings. A database was constructed and benchmarked that was free of errors. Though the database schema was based upon proprietary business data the actual data were non-attributable. The constructed database was then intentionally corrupted (within a percent) to match the corruption patterns found in the proprietary business databases. The ratio scale was used for error characterization,

hence permitting the numerical determination of whether or not significant improvement occurred utilizing Data Bryte, or any processes affecting the quality of the cleansing.

Step 3 Verify Data Bryte

Data Bryte was then executed using selected portions of the constructed database as test cases; this step served to verify Data Bryte. The database first consisted of a number of item type or "fact" tables comprised of basic data entities found in a transaction processing system. The tables consisted of generated (as opposed to actual industry) data in the format shown in Figure 1. The fact tables were then rolled up (aggregated) into tables normally mined by corporations for either improved customer service/loyalty or profits. The tables again consisted of generated data in the format shown in Figure 2. Hence the effectiveness of Data Bryte can be evaluated against problems extant in industry.

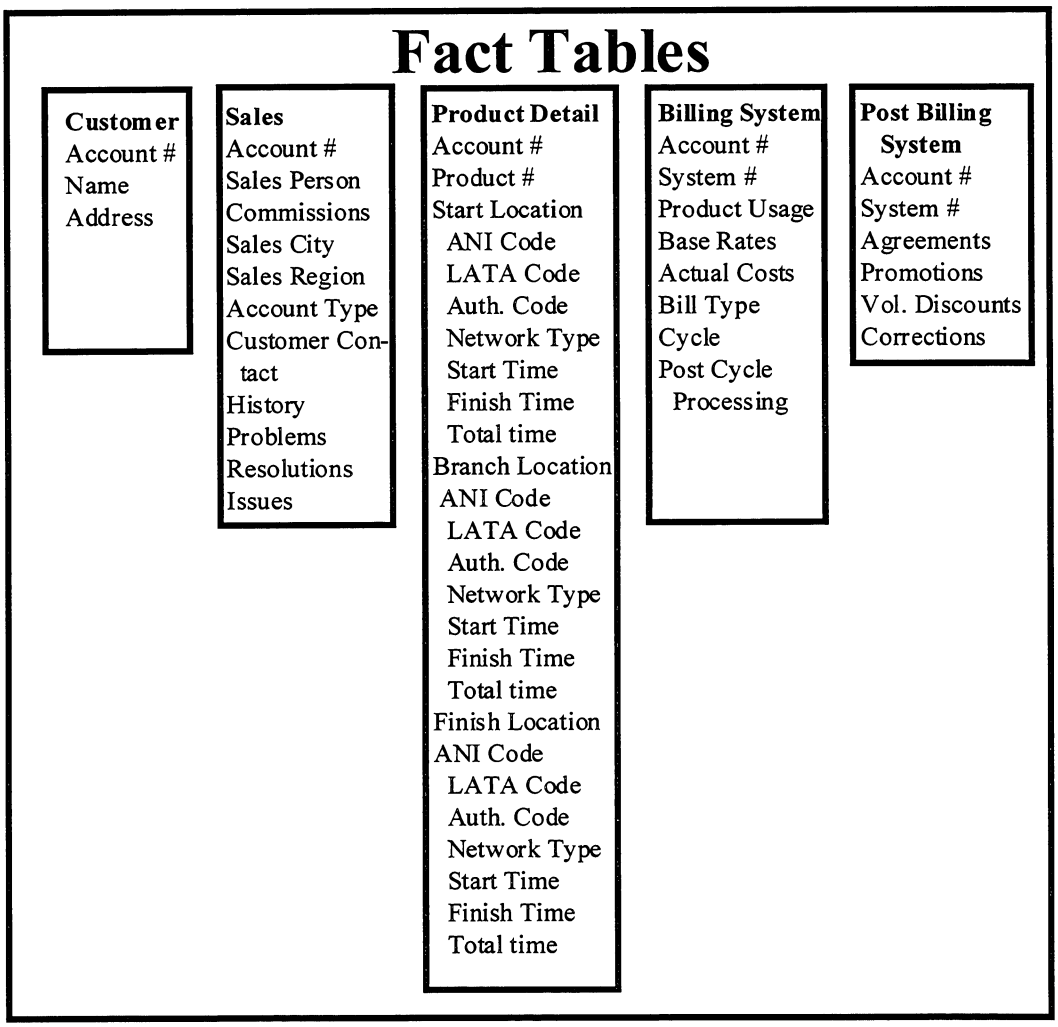

Figure 1 Fact Tables

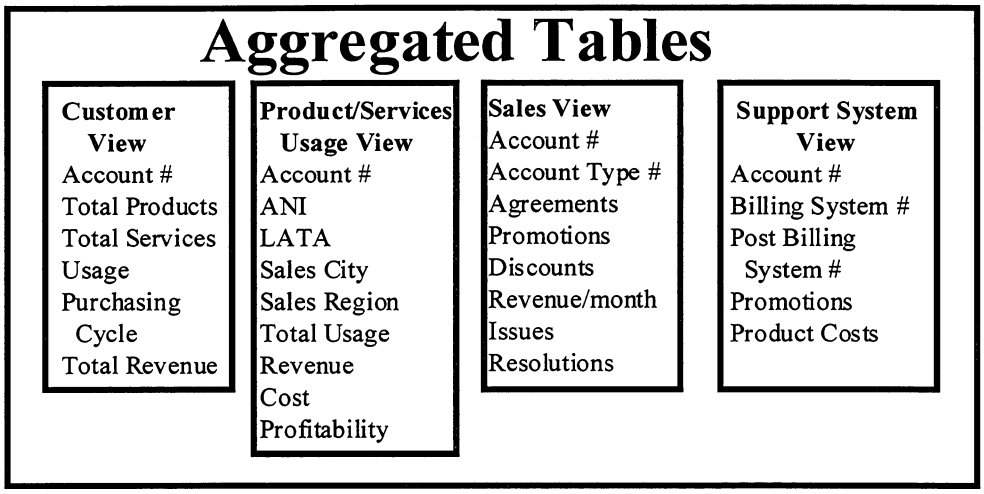

Figure 2 Aggregated Tables

Upon examination and through an extensive literature search, it was found that the attributes of data or item types (for the purpose of identifying their general qualities), fell into four main categories: value, representation, relationships, and behaviors. First, a given item type typically possesses some "value" attribute. This value attribute defines and/or constrains a characteristic of an item type in a particular domain of values for that item type. Commonly, when attribute values are brought up, it immediately brings to mind numeric values. However, the domain values need not be numeric. Indeed, the values could be five different colors or seven different geographic regions, to give just a couple of examples. In addition, these values can be unique to the item type or nonunique, depending upon what is being described. If it is unique, its value will identify one and only one item type instance (e.g. customer "123"). If it is nonunique, the value can identify a set of instances (e.g. phone calls on "Tuesday"). Further, these values can consist of a range or set of values (e.g. dates, weights, money, etc.) that can be either discreet or continuous. Finally, the values are usually predefined as to set membership and range end points.

Second, a given item type typically possesses some "representation" attribute. This representation attribute describes or constrains how an item type can be formatted within the given collection of item types making up the data set. This attribute can include character representation codes (e.g. EBCDIC, ASCII, etc.), data types (e.g. integer, character, money, floating point, time, etc.), basic formatting (e.g. upper/lower case, big endian/little endian, etc.), language (e. g. by country [USA, England, etc.], or by platform [COBOL, UNIX, etc.]), and field

length (e.g. fixed/variable). In some cases this type of formatting is quite minimal (e.g. some item types are "blobs" of data which [by definition] commonly permit any representation with the only [typical] constraint of field length).

Third, a given item type typically possesses some "relationship" attribute. This relationship attribute describes or constrains how an item type relates to some other or collection of other item types. This attribute can include numerical relationships (e.g. one-to-one, one-to-many, many-to-many), as well as cardinality relationships (e.g. mandatory, optional). Hence, this attribute can signal what must or must not exist, given the existence and relationship of a particular item type.

Finally, a given item type typically possesses some "behavioral" attribute. This behavioral attribute describes or constrains how an item type interacts with some other or collection of other item types. For example, if a user tries to edit a particular item type, an interrupt will either permit or allow the editing based upon the allowed or forbidden behavior rules, hence ensuring that certain standards are adhered to.

It was then found, through an extensive literature search, that the types and categories of errors suffered by data item types fell into two categories. These error types surface based upon whether the item types are stand-alone or are aggregated (rolled up) into a larger data structure. For the stand-alone data in the fact tables the errors consisted of omission, duplication, validity, up-to-date, encoding, precision, and error (which included incomplete, mis-fielded, and unreadable). For the aggregated tables the errors consisted of granularity, format, consistency, relevancy, rounding and synchronization. The proprietary corporate database was searched for occurrences of the above-listed types of errors. Where found, they were inserted by type in the constructed database at a frequency that matched (within one percent) of what was present in the corporate database, as delineated by the "Overall Rate" column in the following tables.

The errors were inserted in the database by building spreadsheet tables and color-coding the errors. Later, when the spreadsheet tables were imported into the (constructed) database, the color-coding was not carried into the data fields, essentially "hiding" the flawed fields among the millions of error-free fields.

The Data Bryte cleansing tool was applied against that portion of the constructed (corrupted) database that addressed the representation attribute (described above) for stand alone data. This was done by developing and executing Structured Query Language (SQL) searches, based upon both the standards and model approaches. The resulting data fields that were identified as errors for either approach were then compared with the errors that were identified for the combined approach. This was accomplished by comparing the selected rows against the errors resident in the color-coded spreadsheets. Again, the data in the (resulting) constructed database were then reduced and the errors in given fields characterized.

Step 4 Data Bryte Analysis

In step four of the research, the results of applying each methodology (standards-based, model-based) is presented and analyzed in detail; the impacts and results of the methods selected for each task are determined and discussed. Matrices were developed (see Figures 3 – 4) wherein each of the constructed databases' five data types (e.g. character, integer, money, alphanumeric, and floating point) were arrayed against each of the applicable error sources.

Two of the error types (duplication and precision) did not apply to the character and the alphanumeric data types, as first, duplicates of the data were permitted for each of the fields, and second, precision was not relevant for non-floating point or non-money fields. A single error type (precision) did not apply to the integer data type, as precision was not relevant for non-floating point or non-money fields. Two of the error types (duplication and encoding) did not apply to the money data type, as first, duplicates of the data were permitted for each of the fields, and second, the field definitions are "rigid" enough to provide strong encoding error identification. A single error type (encoding) did not apply to the floating point data type, as the field definitions are "rigid" enough to provide strong error identification.

The standards-based approach to omission identified all blank fields (for character, integer, money, alphanumeric, and floating point) as omission errors. Unfortunately, not all omissions were errors, as not all empty fields constituted an error. For example, not all accounts possessed a sales person (or city or region) "B." Finally, in the case of compound errors (e.g. where a "misfielded" value covers up an omission) the standards-based approach did not identify an omission.

	FIELD NAME	Bytes	TYPE	Overall Rate (%)	Omission	Duplication	Validity	Up to Date	Encoding	Precision	Errors		
											Incomplete	Mis-fielded	Unreadable
1	Surname	15	character	1	X		X	X			X	X	X
2	Given Name	15	character	8	X		X	X			X	X	X
3	County	20	character	3	X		X	X	X		X	X	X
4	State	4	character	<1	X		X	X	X		X	X	X
5	Country	4	character	12	X		X	X	X		X	X	X
6	Sales Person A	15	character	<1	X		X	X			X	X	X
7	Sales City A	14	character	3	X		X	X	X		X	X	X
8	Sales Region A	12	character	4	X		X	X			X	X	X
9	Sales Person B	15	character	<1	X		X	X			X	X	X
10	Sales City B	14	character	3	X		X	X	X		X	X	X
11	Sales Region B	12	character	4	X		X	X			X	X	X
12	Usage Cost Unit	9	character	3	X		X	X			X	X	X
1	Account	10	integer	<1	X	X	X	X	X		X	X	X
2	Account Type	3	integer	<2	X		X	X	X		X	X	X
3	Area Code	7	integer	<1	X	X	X	X			X	X	X
4	Phone #	8	integer	<1	X	X	X	X	X		X	X	X
5	PrimaryZip	10	integer	<1	X	X	X	X			X	X	X
6	Last Four	6	integer	33	X	X	X	X	X		X	X	X
7	Sale #	12	integer	2	X	X	X	X			X	X	X
8	Product/Service #	8	integer	5	X		X	X			X	X	X
9	Start LATA	10	integer	<1	X		X	X			X	X	X
10	ANI A	10	integer	<1	X		X	X	X		X	X	X
11	Net Type A	8	integer	3	X		X	X			X	X	X
12	Fork LATA	10	integer	22	X		X	X			X	X	X
13	ANI B	10	integer	22	X		X	X			X	X	X
14	Net Type B	8	integer	3	X		X	X			X	X	X
15	End LATA	10	integer	4	X		X	X			X	X	X
16	ANI C	10	integer	6	X		X	X			X	X	X
17	Net Type C	8	integer	3	X		X	X			X	X	X
18	BillingSystem #	7	integer	1	X		X	X			X	X	X
19	Cycle	4	integer	2	X		X	X			X	X	X
20	Post System #	7	integer	<1	X		X	X			X	X	X
21	Discount	3	integer	<1	X		X	X			X	X	X
22	Agreement	3	integer	<1	X		X	X			X	X	X
23	Promotion	3	integer	<1	X		X	X			X	X	X
24	Issues	3	integer	<1	X		X	X			X	X	X

Figure 3 Character And Integer Data Type Error Matrix

For duplication, the standards-based approach found all errors (no more, no less), likely due to the "rigid" field definitions, which provide strong error identification. For validity, the

standards-based approach identified all invalid data types (in a given field) as errors. However, it could not consistently identify when the data itself was invalid. For up-to-date, the standards-based approach found all errors (no more, no less), due to the small range of allowed values.

												Errors	
	FIELD NAME	Bytes	TYPE	Overall Rate (%)	Omission	Duplication	Validity	Up to Date	Encoding	Precision	Incomplete	Mis-fielded	Unreadable
1	Commission A	10	money	<1	X		X	X		X	X	X	X
2	Commission B	10	money	<1	X		X	X		X	X	X	X
3	Revenue/month	12	money	<1	X		X	X		X	X	X	X
1	Promotion End Date	8	alphanumeric	2	X		X	X	X		X	X	X
2	Account name	35	alphanumeric	<1	X		X	X			X	X	X
3	Department	15	alphanumeric	2	X		X	X			X	X	X
4	Number	8	alphanumeric	<1	X		X	X			X	X	X
5	Street Name	20	alphanumeric	<1	X		X	X			X	X	X
6	Address #2	20	alphanumeric	3	X		X	X			X	X	X
7	Contract 1	13	alphanumeric	<1	X		X	X			X	X	X
8	Contract 2	13	alphanumeric	<1	X		X	X			X	X	X
9	Start Date A	11	alphanumeric	<1	X		X	X			X	X	X
10	Start Date B	11	alphanumeric	<1	X		X	X			X	X	X
11	Start Date C	11	alphanumeric	<1	X		X	X			X	X	X
12	Product Name	18	alphanumeric	2	X		X	X			X	X	X
13	Bill Type	18	alphanumeric	<1	X		X	X			X	X	X
14	Resolutions	8	alphanumeric	4	X		X	X			X	X	X
1	Start Time A	16	floating point	2	X	X	X	X		X	X	X	X
2	End Time A	16	floating point	2	X	X	X	X		X	X	X	X
3	Total Time A	8	floating point	3	X		X	X		X	X	X	X
4	Start Time B	16	floating point	2	X	X	X	X		X	X	X	X
5	End Time B	16	floating point	2	X	X	X	X		X	X	X	X
6	Total Time B	8	floating point	3	X		X	X		X	X	X	X
7	Start Time C	16	floating point	4	X	X	X	X		X	X	X	X
8	End Time C	16	floating point	4	X	X	X	X		X	X	X	X
9	Total Time C	8	floating point	6	X		X	X		X	X	X	X
10	Base Rate	8	floating point	<1	X		X	X		X	X	X	X
11	Usage Cost	9	floating point	<1	X		X	X		X	X	X	X
12	Actual Cost	9	floating point	<1	X		X	X		X	X	X	X

Figure 4 Money, Alphanumeric, and Floating Point Data Type Error Matrix

For encoding, the standards-based approach found all errors (no more, no less), again due to the rigidity of the definitional rules. For precision, the standards-based approach found all errors that exceeded or were less than the required number of decimal places. For errors (e.g.

incomplete, mis-fielded, and unreadable), the standards-based approach did not find all errors. As explained above, this was due to the impact of compound errors.

The model-based approach to omission identified fewer errors than existed. This was a result of compound errors where "misfielded" data would eliminate an empty data field. For duplication, the model-based approach found all errors. The model-based approach to validity identified all invalid data types (in a given field) as errors and could identify data validity itself given there was a) sufficient rules defining what constituted valid data and b) the membership set was something less than an amorphous "blob" of data. Hence, the model-based approach, due to the broader reach across more metadata, could identify more errors than the standards-based approach.

For up-to-date and for encoding, the model-based approach found all errors (no more, no less), due to the small range of allowed values. For precision, the model-based approach found all errors. The model-based approach to errors (e.g. incomplete, mis-fielded, and unreadable) due to the broader reach across more metadata, could identify more errors than the standards-based approach. As explained above, this was due to the impact of compound errors, and the ability of the model-based approach to provide greater data definitional constraints than could the standards-based approach.

More introduced errors have been (correctly) found by combining the two approaches (Data Bryte) for at least the omission and compound type of errors, when researching the representation attribute in a stand-alone data structure. Hence, the research validated that Data Bryte is a viable framework for defining, analyzing, and improving data quality through an integrated, standards/model-based cleansing process. A significance level of 5% was utilized to reject the null hypothesis, which the changes in the identified errors were due to truly random processes rather than the effects of the cleansing process. As the other attributes (value, relationship, and behavior) and the other data structures are researched, it is likely more types of error improvement can be identified.

Comparison of data quality both before and after was performed. The generated histograms depicting both central and variance tendencies for selected data fields were compared and

contrasted to identify significant changes. The errors in given fields were identified by first applying a model-based cleansing process against the entity item data sets, then analyzed against the baseline. Second, the errors identified by applying a standards-based system against the entity item data sets were analyzed against the baseline. Third, errors identified by applying the combined standards and model-based process against the representation attribute portion of the constructed database were analyzed against the baseline. As a result, it was found that applying the combination of a standards and model-based cleansing paradigm was most successful (in excess of 5% improvement) in the representational attribute utilizing a stand-alone data structure, where compound error sources (omission and misfielded) were involved.

Those elements of the data cleansing process that were affected poorly (exhibited the least amount of improvement) fell into two categories. Either they possessed extremely strong or rigid definitions; or they possessed very inconsequential definitions, which permitted almost any data to qualify as clean. The former category included the data with encoding and precision errors. The latter included the alphanumeric data type. The ability to include a more constraining or defining set of metadata appeared to be the most significant external data influence. Currently a stronger set of metatdata is being evaluated.

Step 5 Draw Conclusions

The success criteria for developing Data Bryte (greater than 5% improvement in error detection) were met for the omission and compound error sources. Generally, the standards-based approach was effective only insofar as a) the standard itself was clean and b) the definition of the item type was rigid. As expected, as the standard itself or the item type definition deteriorated, so did the standard-based approache's error-finding ability. When the results of identifying bad data from all of the errors sources were combined, errors missed by either or both the standards and model-based approaches were illuminated.

The identification of errors overall was poorest among the alphanumeric fields. It appears that this was due to the low "constraining power" of a data type that permits either integers and/or characters in (almost) any ordering.

It is important to note that the quality of the validity error source could not be improved greatly (in excess of 5%) in the representation attribute and stand-alone data structure. The underlying problem is that bad data often look and smell just like good data. The significance of the poor improvement rate is this. Though the amounts may be minor, the sheer volumes of modern transactions create a situation where the validity errors are no longer minor. This problem deserves further research.

Step 6 Make Recommendations

If unclean atomic data are the root of all evil, the aggregated data are worse, due to the compounding effect of multiple error sources. It is crucial that a coherent set of strongly defining or constraining metadata be generated for a given data repository, if any serious attempt will be made to cleanse the data. Due to the immense size of todays' data repositories (approaching the Petabyte size), the acquisition of a strong metadata tool will be mandatory. Issues such as data volumes, interfaces, data import/export services, generation of code, and finally the ability to reverse engineer existing code/interfaces, will be key criteria for deciding which tool to select.

Data Bryte can be utilized against any data repository, as long as the tool is tailored to the data. This will require analysis of the data, analysis of the metadata, and the building of a suite of structured queries that strongly drill through the data.

Conclusion

Telecommunications call records and product/service information provides a rich source of data with which to evaluate a coherent data standards/data model cleansing approach. Data errors range from a low of one percent for the financial data, five-percent plus for the customer data, and an (initially) undefined large number of errors for the aggregation data. Data analysis strongly suggests that Data Bryte is a viable data cleansing methodology, especially where compound error sources are at work. Additional research in the value, relationship, and behavior attribute domains will be undertaken to further optimize the methodology.

References

[1] Allen, S. "Name & Address Data Quality," Proceedings of the 1996 Conference On Information Quality, Massachusetts Institute of Technology, (1996). pp. 242-255.

[2] Ballou, D. P., and Tayi, G. K., "Managerial Issues in Data Quality," Proceedings of the 1996 Conference On Information Quality, Massachusetts Institute of Technology, (1996). pp. 186-206.

[3] Jarke, M., and Vassiliou, Y., "Data Warehouse Quality: A review of the DWQ Project," Proceedings of the 1996 Conference On Information Quality, Massachusetts Institute of Technology, (1996). pp. 299-313.

[4] Mohan, S. D., Schroeder, C., and Willshire, M. J., Data Bryte: "A Proposed Warehouse Cleaning Framework," Proceedings of the 1998 Conference On Information Quality, Massachusetts Institute of Technology, (1996). pp. 283-291.

[5] Rosenthal, A., and Dell, P. "Propagating Integrity Information in Multi-Tiered Database Systems," Proceedings of the 1997 Conference On Information Quality, Massachusetts Institute of Technology, (1997). pp. 339-351.

[6] Wang, R. Y., "A Product Perspective on Total Data Quality Management," *Communications of the ACM*, vol. 41 #2, pp. 58-65, February 1998