A Framework For Addressing Data Quality In Distributed Computing Systems.

Finbar Fletcher - Wesley J. Howe School of Management, Stevens Institute of Technology,Hoboken, NJ; Ministry of Trade and Industry, Port of Spain, Trinidad. email: finfletc@acm.org.

Abstract: *In this paper a framework is proposed for addressing data quality issues in a distributed computing environment. Two models are posited, one for data quality and another for distributed computing, which are then juxtaposed in a matrix whose values represent measures of the data quality risk posed by a distributed computing environment. This measure is called the Data Quality Risk Exposure Level (DQREL) and a method for its operationalization using a qualitative approach based on data from an empirical study by the author is suggested. Several potential contributions of this framework are also identified.*

All Information Systems, regardless of their level of sophistication, are principally geared towards the output of one product, i.e., data. That the production of data is perhaps a major raison d'être of computing systems is often lost on system designers, developers, and users. The result is that minimal attention is paid to the quality of the *data product*. This lack of focus has created a situation where large quantities of 'defective data' exists, as noted by several authors e.g., Laudon[1986], Morey[1982], O'Neill and Visine-Goetz[1988], and more recently by Redman[1998] who reports error rates in the 0.5 percent to 30 percent range. These error estimates, albeit high, understate the true nature of the data quality problem as they for the most part only consider the accuracy dimension of data quality. For example, they do not consider the fact that data might also be inconsistent, incomplete or irrelevant. In today's highly competitive business environment, strategic decision making demands zero-defect data. Anything less would be exposing the enterprise to serious risk of failure. Why then have issues of data quality received so little attention? No answers are readily available, but we can surmise that it has to do with the implicit infallibility that is attributed to the output of computer-based systems, i.e., once there are no discernible flaws in the hardware and or software, we assume that most of the output is perfect. Indeed, the quality revolution that other areas of business and industry have been subjected to over the past decade has left data quality largely untouched, in spite of the existence of solid business justifications for good data quality (e.g., Cronin[1993], Gartner Group[1993]).

Indeed, it has been suggested that since most business process reengineering initiatives require the ability to share data, many will fail through lack of attention to data quality (Gartner Group [1993]).

Although the problems of data quality have not been resolved in the traditional centralized and or stand-alone computing environments, both the popular and academic literature have been alluding to the fact that such problems will be exacerbated in migration to a distributed computing environment (Gray[1986], Redman[1992], Ricciuti[1995]). Some of the reasons quoted are, for example, duplication of data (Redman[1992]), and the general explosion in data storage and communications capacities (Redman[1992]). However, no evidence, empirical or otherwise, has been forwarded to support their comments, and a search of the literature revealed no such studies. Thus the literature has either casually discussed the problem, or not at all. For example, it has been well-documented that telecommunications management systems (of which a Distributed Computing System (DCS) can be considered a subset or variant) require different skills, management styles and generally pose different problems than traditional IS management (see Donavan[1988], Hall & McCauley[1987]), yet *no explicit mention has made been of the data quality issue.* Gray[1986] suggests that "*[distributed systems] will always require more careful design, planning, and management than their centralized counterparts.*" Thus it has long been recognized that distributed computing poses additional problems. However, data quality has not heretofore been explicitly recognized as one of these problems. If we accept the notion that bad data is like a virus and once it gets into the corporate data stream there is no telling where next it will strike, then it would be reasonable to assume that a distributed computing environment would be a very fertile ground for its proliferation. Indeed, the growing importance of data warehousing, distributed databases and client-server architectures in corporate computing environments, gives cause for further concern with respect to the propagation of poor quality data. Neumann[1996] notes that as distributivity increases, the risks to activities such as data and processing also increase. If it is indeed true that migration to a distributed computing environment poses increased data quality risks, then there is a need to develop implementations that alleviate or reduce that risk. Such implementations would *depend on an accurate assessment of the nature and extent of risks involved.*

This paper presents a framework whereby risks to the degradation of data quality based on different distributed computing structures can be assessed. Since the concepts of Data Quality and Distributed Computing are not uni-dimensional, the question essentially being posed is -

How are the various dimensions of Data Quality impacted by the various dimensions of a Distributed Computing System. Models of the two thus need to be identified and or developed. A conceptual and practical framework is presented for looking at data quality problems in distributed systems and in the process we also introduce a concept which we call the Data Quality Risk Exposure Level (DQREL). One possible method of operationalizing this concept is presented based on the results of an empirical study, and while it stops well short of providing the quantitative basis which is ideal, the qualitative values derived can be used initially.

Although the present discussion focuses on the relationship between a DCS and data quality, this relationship can both directly and transitively impact several other organizational variables such as decision-making and organizational performance. Although we recognize its extreme importance, this investigation is left to other efforts. The illustration in Figure 1 shows how DQ fits into a general organizational model.

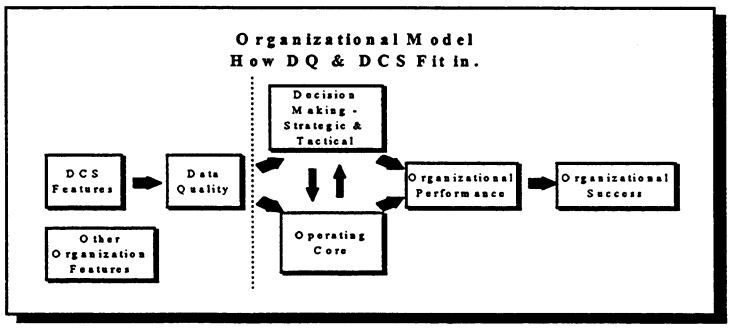

Figure 1

The Data Quality Model:

The discussion in this section centers on those dimensions of data quality which are considered appropriate for the evaluation of distributed computing systems. From a practical standpoint every data quality dimension that appears in the literature cannot be considered. For example, Redman[1992] identifies fifteen dimensions of data quality, while Wang et al.[1994], in the initial part of their study identify more than 120 data quality attributes which they later reduce to twenty dimensions. From a theoretical standpoint also, many of the dimensions quoted

in the literature e.g., objectivity (Wang[1994]), quantity (Zmud[1978]) have little relevance to whether an information system is distributed or not.

While some authors approach data quality as a uni-dimensional concept, e.g., Agmon & Ahituv[1987] considers only accessibility and Morey[1982] sees data quality as being synonymous with accuracy, the literature seems to be in general accord that data quality is a multi-dimensional concept (Wand & Wang[1996], Wang et al[1994], Redman[1992], De Lone & McLean[1992]). However, there appears to be little consensus as to what constitutes a good set of data quality dimensions and accompanying definitions (Wang, Storey & Firth[1993]). Here we have the added burden of zeroing in on those dimensions which we consider particularly useful in discussing the impact of data quality on a distributed computing system. Several authors have addressed the dimensionality of the information concept directly or indirectly e.g. Zmud[1978], Ives et al.[1983], De Lone & McLean[1992], Wang et al.[1994], Redman's[1992], Miller and Boyton[1987], Feltham[1968], Agmon and Ahituv[1985], Orman et al.[1994], Kriebel[1979], Loebl[1990], Wang et al[1993b], Wand et al[1996]. Based on a review of the literature eight dimensions were chosen to compose the data quality model. These dimensions were selected based either on their prominence in the data quality literature and / or their potential relevance to distributed computing systems. The first four dimensions are what can be considered as ubiquitous, since they appear in virtually all discussions of data quality and the quality/effectiveness of the information product. Thes0e are accuracy, relevancy, completeness and a dimension reflecting the temporal aspects of data viz. currency or timeliness, or both. The other four dimensions which will compose our data quality model are consistency, accessibility, availability, privacy/security. A fuller discussion as to the derivation of the data quality model is presented elsewhere (see Fletcher[1997]).

The Distributed Computing System (DCS) Model:

Given the multi-dimensionality of the concept, the state of flux and the relative newness of the DCS field, although several definitions of a DCS have been forwarded by various researchers (e.g., Umar [1992], Khanna[1994], Colyler & Wong[1994]), there is no universally agreed upon definition of a DCS. Khanna[1994], for example, describes a distributed computing environment as one which has evolved from "*one comprised of dumb terminals connected to expensive mainframes in glass houses to one comprised of networked personal work-stations and servers.*"

Notwithstanding the above, some underlying notions can be extracted and most agree that a DCS can be considered as a

> *"collection of **autonomous** computers which **communicate** in order to accomplish **business functions**."*

This is a very generic definition of a DCS, which for example places no restrictions on the location of the computers - they can be contained within a single building, a city, or spread over several countries - or on the type of computers - they can be only micro-computers, or a mixture of micro-computers, mini-computers and mainframes - or on any other differentiating variables.

As our referent point we used an adaptation of the Distributed Computing Reference Model (DCRM) suggested by Umar[1992]. The DCRM focuses on how the functionality of the DCS is achieved and incorporates the notions posited in our definition. The DCRM (figure 2) represents "a vision which defines the scope, the structure, and the mechanisms" of a DCS, and consists of three levels:

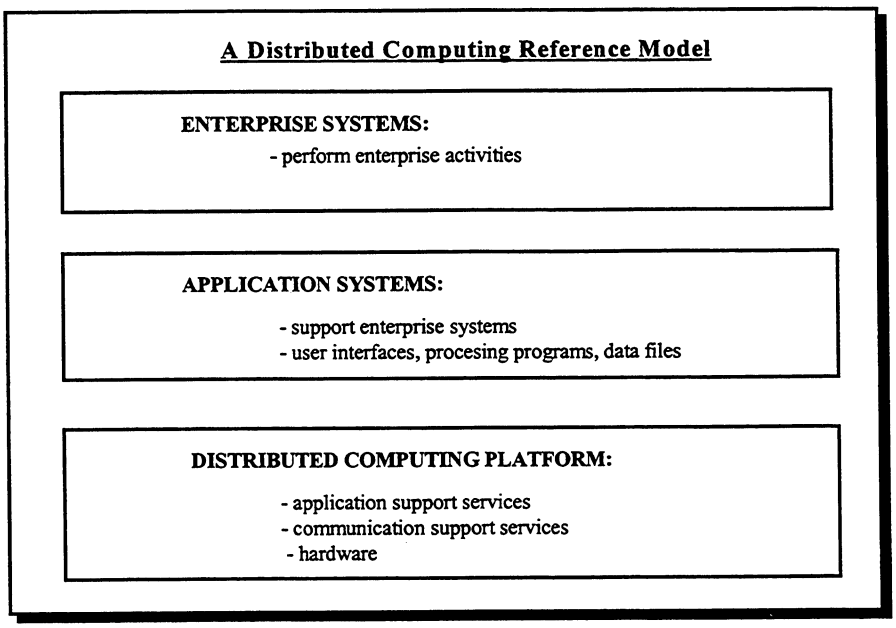

Figure 2.

1. ***Enterprise Systems*** - this is the highest level of the model and represents the general business area - banking, government, education, manufacturing, engineering, medical - and the (macro) structuring of that area, its business processes and activities to satisfy business objectives. Enterprise systems may be organized on a centralized, divisionalized or decentralized basis (Mintzberg[1979]). Thus essentially, enterprise systems may be organized

on a centralized or distributed basis with varying levels of centralization/distributivity in between.

2. ***(Distributed) Application Systems (DAS)*** - these facilitate the enterprise systems by providing support via information technologies. Payroll, HRS, accounting, collaborative computing are all examples of application systems. An AS consists of three components: data, programs, and user-interfaces, each of which can reside on different computers in the DCS. The components can be configured or arranged in an architecture that can range from centralized to various levels of decentralization.
3. ***Distributed Computing Platform (DCP)*** - this consists of a range of technologies and services that provide support for and enable the DCS. It can potentially provide two sets of services - application support services and communication network services. The first includes the interconnectivity options/data access modes - terminal emulation, file transfer, or a client-server system - ; distributed data and transaction management system and distributed operating system. The latter concerns the physical communications infrastructure.

Khanna[1994] also identifies three components of a distributed computing environment, which include distributed computing infrastructure, systems management, and distributed applications. While the first two map into Umar's DCRM distributed computing platform component, and the latter is equivalent to the DCRM's distributed application systems, Khanna's model does not contain anything equivalent to the DCRM's enterprise systems, i.e., the organizational aspect. In the DCRM there are no constraints which say that any of the three levels must be distributed or distributed to the same degree. Thus, it is possible to have a DCP that is distributed and an AS that is not and vice-versa. Indeed, gauging the relationship between the degree of congruence of the three levels of the DCRM in terms of distributivity and performance factors (organizational and technical) appears to be an area for further fruitful research.

The AS, we believe, is the heart of the DCS and has been recognized as "the most critical aspect of Distributed Computing" (Umar[1992]). Further, since it is the focal point, we believe that it is the level of distributivity in the DAS which would largely determine the level of distributivity in the DCS as a whole. *Thus, in this effort the focus is on the distributed application system and how the organization of its components - data, programs and user-interfaces - affects data quality.* A DAS has three major components :-

- the application data
- the application programs

- the application user-Interfaces

In the trend of object-oriented distributed systems models which characterize the set of resources available in a DCS - computers, data, programs, etc., as a collection of objects which can be arbitrarily combined to provide information processing capabilities - we also have recourse to some object-oriented terminology. Thus the three components above can be considered as high-level composite objects which can be broken down or decomposed into smaller objects, i.e., objects which can range in granularity from entire systems to smaller objects such as text strings, images, etc. Each of the three *composite objects* are logically considered whole units, whose components may or may not be physically collocated. We refer to the objects which make up a composite object as *decomposed objects*. Further, a decomposed object itself is considered as a composite if it can be further decomposed.

Application data:- The application data consist of all the data used in the distributed application and stored on the computers of the DCS. The application data can be decomposed into one or more smaller objects which we refer to as *data objects*. If we consider the application data as a logical whole then a data object results from any feasible division. By feasibility we mean a division or decomposition scheme that is both possible and makes sense to the designer. *Thus a data object can be a database, a set of tables (relations), a set of rows(tuples), or a set of columns of a relational database; a set of objects in an OODB, or a set of rules in a KBS.* [Note that a set can contain one or more objects].

Application Programs: The application programs consist of all the programs used in the DAS and stored on the computers of the DCS. Similar to the application data, it is a logical whole which can be decomposed into smaller objects which we refer to as *program objects. Thus a program object can be a program, a sub-program, a module, a function or any other feasible division unit.*

Application User-Interface: This facilitates user interaction with the rest of the DAS and consists of all the user-interfaces and components thereof used in the DAS and stored on the computers of the DCS. It is also a logical whole which can be decomposed into feasible units called *interface objects*. For example, an order-entry application user-interface may contain fifty screens or forms. These forms can be taken as a whole or divided up into feasible sets for different users. Any of these feasible division can be considered an interface object.

Different placement or distribution strategies can also be applied to the components of the DAS resulting in different configurations. Two such generic strategies are **replication** and

partitioning (see Fletcher [1997] for a fuller discussion and derivation of these placement strategies). Based on these generic strategies and the DAS components identified above, six attributes which may be used to differentiate distributed computing environments can be identified as follows:

- **data replication**: copies of the same data residing on different nodes in your system.
- **data partitioning**: division of the data set among several nodes in the system.
- **program replication**: refers to the extent to which the copies of the programs reside on several nodes.
- **program partitioning**: the extent to which the component programs are divided and shared between different nodes, i.e., the extent to which program modules are divided up between different nodes.
- **user-interface replication**: use of the same user-interface on all nodes in a system.
- **user-interface partitioning**: the extent to which different user-interfaces are used on different nodes as part of the same application system

The Horizontal-Vertical Dimension of Distributed Computing Resource Distribution: The discussions above have focused only on the fact that data and other resources can either be replicated or partitioned. However, resources can be distributed in terms of at least one other dimension, i.e., horizontal vs. vertical. Horizontal distribution refers to the movement of resources (D, P, U) horizontally across different nodes. Vertical distribution, however, refers to the movement of resources between machines of different computing capacities. For this purpose we can identify three generic classes of machines - mainframes, mini-computers, and work-stations. The previous discussion has perhaps implicitly assumed that replication and partition was taking place along the horizontal dimension only. Indeed vertical distribution can be considered a special case of horizontal distribution, having only three nodes. In that case all that has been developed thus far can be equally applied, i.e., resource replication and partitioning takes place both vertically and horizontally. Figure 3 below illustrates. It should be noted that it is vertical distribution which in the main characterizes the move to client-server systems.

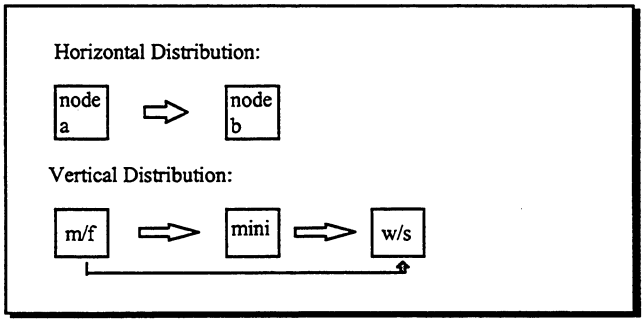

Figure 3.

It should be noted that vertical distribution can also be looked at in two ways. In the first instance it can be considered as the extent to which resources are apportioned between the levels of computing capacity. In the second instance it can be considered as the extent of downward migration from mainframe to mini and work-stations in which case the application of weights would be necessary. For example, resources residing on lower capacity machines would be given higher weights in terms of distributivity.

Bringing it all Together : A Framework for Addressing Data Quality in Distributed Computing Systems.

In the discussions above we have said that Data Quality (DQ) in an organization can be described by a set of eight dimensions - *accuracy, completeness, relevancy, consistency, timeliness, accessibility, availability, and privacy*. A distributed computing system (DCS), in particular a distributed application system (DAS) can also be identified by a set of attributes along which various DCS differ, and which also distinguish them from other computing environments - data replication, program replication, user-interface replication, data partitioning, program partitioning, and user-interface partitioning along both the horizontal and vertical dimensions. These attributes both distinguish DAS from non-DAS, and from other types of DAS. Note that these two sets do not define the possible complete set of DQ dimensions and DCS attributes, but only those that we have previously identified. In order to better examine the relationship between DQ and DCSs we introduce a matrix called the Attributes/Dimensions (AD) matrix, with DCS attributes as columns and DQ dimensions as rows of the matrix. An X in a cell (i.e., intersection of row and column) identifies a possible or potential relationship between the

DQ dimension (DQD) in that row and the DCS attribute in that column. *(Note also that it would be possible to construct an AD matrix using the points along the distributivity continuum as forwarded by Fletcher[1997]).* Using the AD matrix and what we know of data quality and DCSs, we can identify either quantitatively or qualitatively or both how the attributes and the dimensions interrelate as shown in Table 1 below. Each 'x' would in fact represent some quantitative or qualitative value and indicates some possible relationship between the DQ dimension and the DCS attribute. For example, an 'x' in the Accuracy-Data Replication cell says that accuracy would be impacted by data replication to an extent indicated by the 'x'.

Table 1.

AD MATRIX (Partial Representation)

Attributes → Dimensions ↓	**Data Replication**	**Program Replication**	**U-I Replication**
Accuracy	x	x	X
Timeliness	x	x	X
Completeness	x	x	X
Consistency	x	x	X
Relevancy	x	x	X
Accessibility	x	x	X
Availability	x	x	X
Privacy	x	x	X

The above matrix of 24 cells is just a partial representation with the partition attributes omitted. The full matrix should indicate 48 (96 if the vertical-horizontal distributed computing dimension is considered) possible relationships. Both the validity and the extent of these relationships need to be ascertained. Each of the attributes will also affect the particular DQD in varying degrees. Put in another way, this is saying that *each DCS attribute will expose DQ to different levels of degradation via its effect on the DQD*. Thus what a DCS does (via its attributes) is to increase or reduce the risk of defective data being propagated in the organization.

We thus propose a concept (or measure) called the ***Data Quality Risk Exposure Level (DQREL)*** which is intended to capture the magnitude of the risk exposure that a DCS attribute poses to a DQD. For each attribute identified we would like to say how much danger it poses to the DQD expressed in terms of the DQREL. For example, we would like to say that given the existence of an attribute in a DCS, we would like to express the risk (either an increase or

reduction) to DQ posed, i.e., DQREL in terms of some quantitative (say 10%) or some qualitative (say high, low or moderate) measure. The ideal would be to have a quantitative measure, however, given the difficulties involved, a qualitative measure may suffice in the interim. But let us assume for the moment that we can indeed assign quantitative values to the DQREL. We can then construct another version of the AD matrix - say the AD-DQREL matrix, and replace the Xs with the values obtained for the DQRELs. This matrix will then be rich in information. Based on this matrix we

* can define a compound measure of DQREL for each dimension, which aggregates the DQREL of all distributivity attributes (in same row) with respect to the DQD - call this $DQREL_{dim}$
* can define another composite measure of (total) DQREL by combining all the $DQREL_{dim}$
* can also define a measure which aggregates the DQREL for a distribution attribute over all the quality dimensions - call this $DQREL_{att}$

This can be summarized thus -

given n DCS attributes $ATT_i ... ATT_n$ and m DQ dimensions $DQD_j ..... DQD_m$

$DQREL_{j,i}$ = risk exposure level of DQD_j associated with ATT_i

$$DQREL_{DIMj} = \sum_{i=1}^{n} DQREL_{j,i} = \text{total risk exposure to } DQD_j \text{ associated with } ATT_i ... ATT_n$$

$$DQREL_{ATT_i} = \sum_{j=1}^{m} DQREL_{j,i} = \text{total risk exposure level over } DQD_j ... DQD_m \text{ by } ATT_i$$

$$DQREL_{TOT} = \sum_{j=1}^{m} DQREL_{DIM_j} = \sum_{i=1}^{n} DQREL_{ATT_i} = \text{Total Risk Exposure Level}$$

If a DCS configuration is defined as any combination of DCS attributes then a DQREL could be evaluated for each such configuration. Some of the assumptions and shortcomings of the quantitative DQREL model include the following:

- the above equations assume that the DQREL is quantitatively measurable and that the summation of its different components would give some meaningful total measure. This may not be the case. (see below).

- the model is additive, i.e., it assumes the DCS attributes are orthogonal. It is likely that this may not be the case as an attribute may moderate or exacerbate another's effect on the DQD, for example with the introduction of program replication, the effect of data replication on say the accuracy dimension may increase. Thus the underlying relationship may be other than additive, but for exposition and simplicity, it may be necessary to assume orthorgonality.
- there is an implicit assumption that the DQREL of a DCS configuration will be constant across organizations. For example, we are not sure whether a given configuration applied to the banking sector and a university administration will result in similar DQRELs.

Whilst the above approach may be the way to ultimately proceed, we believe that further investigation is needed to resolve the above and other issues related to a quantitative model. However, whilst a qualitative approach may not yield as rich an information content as the quantitative approach outlined above, it can nonetheless provide valuable information as to the nature of the DCS-DQ relationships. In the discussion that follows, a qualitative operationalization of the DQREL is presented.

Operationalizing the DQREL:

Recall the Xs in the original AD matrix. Each X can in fact be considered as an hypothesis about the relationship between a DCS attribute and a DQ dimension. A set of hypotheses can thus be established and tested and the results translated into a DQREL measure. One possible way to do this use correlation measures from an empirical analysis and translate those to DQREL qualitative risk measures - e.g., high (H), moderate (M), low (L), for both positive and negative significant correlation values and none (N) for insignificant values. Thus a cell in the DQREL matrix would contain one of the following values: H^+ for high positive correlations, M^+ for moderately positive correlations, L^+ for low positive correlations, N for insignificant correlations, H^- for high negative correlations, M^- for moderately negative correlations, and L^- for low negative correlations. One of the principal objectives of an empirical study conducted by the author (see Fletcher[1997]), was to ascertain whether any association existed between data quality and the extent of distributivity in an application system. Utilizing the models of DQ and DCS outlined earlier, tests of individual hypotheses of relationship between a given data quality dimension and a given DCS attribute were conducted using correlation analysis. The data quality dimensions were measured at the level of the user (data consumer) of the application system with

the instrument used being based on a modification of the short form of the User Information Satisfaction (UIS) measure as suggested by Baroudi and Orlikowski [1988]. Data quality responses were elicited from a prinicipal user of each application system, e.g., a manager or other user. The distributivity attributes were measured at the level of the (application) system, based on responses from technical managers as to the extent of distributivity (replication and partitioning) of computing resources i.e. data, programs and user-interfaces in a given application system. A total of 41 application systems drawn from nine organizations were utilized in the study. Application of the results of this study yields the DQREL matrix in Table 2 below.

Table 2.

Data Quality Risk Exposure (DQREL) Matrix.

DAS Attribute ---> DQ Dimension	DR	DP	PR	PP	UR	UP
Accuracy	H-	N	L-	N	N	M+
Completeness	M+	N	H-	N	L+	M-
Consistency	M+	M-	N	N	N	N
Accessibility	M+	N	L+	L-	M+	L-
Availability	M+	N	N	L-	M+	M-
Relevancy	M-	L+	L+	M-	M+	M+
Privacy	M-	M-	N	N	N	M+
Timeliness	H+	N	L+	N	M+	M-

Values in the DQREL matrix above were assigned as follows:

- High (H): >= .30 for significant correlations (positive/negative).
- Moderate (M): >= .15 and < .30 for significant correlations (positive/negative).
- Low (L): < .15 for significant correlations (positive/negative).
- None (N) for insignificant correlations.

In summary, the results, translated to the values in the DQREL matrix above, indicate a cause for concern with regard to the increased risk in the degradation of data quality in the move towards distributed and more distributed systems. This is particularly so in the case of data replication. Some caveats however should be advanced with respect to the use of the actual results of this study in the derivation of DQREL values. First, the size of the sample used was small. Second, the sample was not at all representative as cases were drawn from a limited area. However, we believe that this does not invalidate the framework presented.

Concluding Remarks:

Although we caution against its general applicability, the results of the Fletcher[1997] study as shown in the DQREL matrix in Table 2 indicate that information managers should at least 'put their guard up' if they are considering moves towards distributed computing and wish to maintain high levels of data quality. The DQREL matrix can give an idea of some of the data quality pitfalls which may lurk along the path to (more) distributivity. Also, oftentimes in distributed systems IS managers have a very hard time justifying in advance the provision of defensive measures, whose benefits are not readily evident (Neumann[1996]). A framework such as the one presented can help in the justification of these defensive measures. Specifically, we believe that the ideas presented here has the potential to contribute to both theoretical and practical endeavors and overall lead to the better management of a distributed computing environment in terms of data quality.

From a theoretical standpoint:

- it can form a basis for the discussion of data quality problems in distributed computing systems.
- it may also form a foundation for the development of a theoretical model with respect to DQ in DCS, e.g., a cause and effect model as suggested by Wang et al.[1993b].

From a practical standpoint (both managerial and technical) properly derived DQREL can provide the following benefits:

- it can be used as a practical basis for the assessment of organizational risk with respect to data quality, given the organization's DCS configuration. Neumann[1995] for example, identifies the need for a method to assess risk in system development and operation. The magnitude of risk present for areas of data quality concern can be determined. For example, in a distributed warehouse control system, one must know that the values being read are true, or at the very least the degree of confidence one can attach to the values (Suomi[1994]). Our research can help in this. Our framework can also be applied to parts of or an entire organization.
- it can be used to guide the IS department as to where technical resources should be allocated towards data quality improvement.
- organizations have to make choices when allocating scarce resources; e.g., they may need to know whether to put resources into maintaining accuracy or consistency. Specifically, it can form the basis for the allocation of resources to Data Clean-Up and Validation,

given finite resource levels and having identified the areas most at risk. The data quality risk parameter can be an additional parameter in an optimal resource allocation model or used to determine an optimal distribution of data, programs and user-interfaces given certain data quality requirements.

- it can be used as an aid in the design of DCS by paying attention to those areas most at risk. Tradeoffs may be necessary between the risk of data quality degradation and the need for certain attributes.
- help in the design of (Data Quality) Disaster Recovery Plans by identifying where risk level is highest; e.g., if a high consistency risk is indicated, and the cost of maintaining complete consistency is high, one may wish to design on a basis of planned inconsistency.
- Umar[1992] notes that "*as distributed systems become more intelligent, application system components will be automatically moved to appropriate locations for improvements in availability and performance.*" Our research results and model can be used as part of the decision-making framework for this.
- Application downsizing sometimes takes place gradually, beginning first with the distribution of user-interfaces, then the distribution of programs, and so on. Our framework should indicate to managers what data quality problems to expect at each transition stage.
- Can be used as a management tool for examining technical versus organizational tradeoffs. For example, an organizational objective may the creation of an informated organization (Zuboff[1985]). This may be partly achieved by the dispersion of data throughout the organization. This may not only pose problems for data quality dimensions such as consistency, but may also impact on the efficiency with which the data is processed.

Finally, the ideas presented here can open up other areas for fruitful research. One immediate area is the derivation and validation of the DQREL values based on a study utilizing a much larger sample size and one which is more representative in terms of the broad spectrum of DCS types and wider range of national and international organizational units.

References.

Agmon, N., Ahituv, N. (1987). Assessing Data Reliability in an Information System. Journal of Management Information Systems, Fall, 4, 2, 34-44.

Ballou, D.P, Kumar.Tayi, G. (1989). Methodology for Allocating Resources for Data Quality Enhancement. Communications of the ACM, March, 32, 3, 320-329.

Baroudi, J.J., Orlikowski, W.J. (1988). A Short-Form Measure of User Information Satisfaction: A Psychometric Evaluation and Notes on Use. Journal of Management Information Systems, Spring, 4, 4, 44-59.

Cronin, P. (1993). Closing the Data quality Gap through Total Data Quality Management. (TDQM). MIT Management, June.

De Lone, W.H., McLean, E.R. (1992). Information Systems Success: The Quest for the Dependent Variable. Information Systems Research, March, 3, 1, 60-95.

Donavan, J. (1988). Beyond Chief Information Officer to Network Manager, Harvard Business Review, 43, 4, 684-696.

Feltham, G. (1968). The value of Information. The Accounting Review, 43, 4, 684-96.

Fletcher, F. (1997). The Effect of Distributed Application Systems Resources on Organizational Data Quality. An Empirical Analysis from the Data Consumer's Perspective. Ph.D Dissertation, Stevens Institute of Technology.

Fox, C., Levitin, A., & Redman, T. (1994). The Notion of Data and Its Quality Dimensions. Information & Processing Management, 30, 1, 9-19.

Gartner Group. (1993). Data Pollution Can Choke Business Process Re-Engineering. Gartner Group Inside Industry Services, 1.

Gray, J.N. (1986). An Approach to Decentralized Computer Systems. IEEE Transactions on Software Engineering, June, 12, 6, 684-698.

Hansen, J.V.(1983). Audit Considerations in Distributed Processing Systems. Communications of the ACM, August, 26, 8, 562-570.

Hansen, M., Wang, R. (1990). Managing Data Quality: A Critical Issue for the Decade to come. (No. CISL-91-05). Composite Information Systems Laboratory, MIT, Sloan School of Management.

Honeyman, P (1994). Distributed File Systems. In Distributed Computing. Implematation and Management Strategies. Prentice Hall.

Huh, Y.U., Keller, F.R., Redman, T.C., & Watkins, A.R. (1990). Data Quality. Information & Software Technology, October, 32, 8, 559-565.

Juran, J.M.(1980). Quality Planning & Analysis: From Product Development Through Use. New York, Mc Graw Hill.

Khanna, R. (1994a). Introduction. Distributed Computing : Implementation and Management Strategies. Prentice Hall.

Khanna, R. ed. (1994b). Distributed Computing: Implementation and Management Strategies, Prentice Hall.

Kriebel, C.H. (1979). Evaluating The Quality of Information Systems. In Design and Implementation of Computer Based Information Systems. Germantown: Sijthoff & Noordhoff.

Laudon, K.C. (1986). Data Quality & Due Process In large Interorganizational Record Systems. Communications of the ACM, January, 29, 1, 4-18.

Loebl, A.S. (1990). Accuracy and Relevance and the Quality of Data. in Data Quality Control:Theory & Pragmatics, Liepins, G.E., Uppuluri, V.R.R eds.
Malone, T.W. et al.(1987). Intelligent Information Sharing Systems. Communications of the ACM, 30, 5, 391- 402.
McCue, D, Little,M(1992). Computing Replica Placement in Distributed Systems. IEEE Proceedings.
Menou, M.J. (1995). The Impact of Information - 1. Toward a Research Agenda for Its Definition and Measurement. Information Processing & Management, June, 31, 4, 455-477.
Morey, R.C. (1982). Estimating and Improving The Quality of Information in an MIS. Communications of the ACM, May, 25, 5, 337-342.
Morse, S.(1993). Rightsizing : Tailoring the Applications and the Platforms. Network Computing, Feb., 63-77.
Neumann, P.G. (1995). Computer Related Risks, ACM Press.
Neumann, P.G. (1996). Distributed Systems Have Distributed Risks, Communications of The ACM, November, 39, 11, 130.
Nutt, G. (1992). Open Systems. Prentice Hall.
Oman, R ., & Ayers, T. (1988). Improving Data Quality. Journal of Systems Management, May, 31-35.
O'Neill, E.T., & Vizine-Goetz, D. (1988). Quality Control In On-line Databases. Annual Review of Information Science & Technology (ARIST) M.E. Williams editor, 23, 125-156.
Orman, L., Storey, V., Wang, R. (1994). Systems Approaches to Improving Data Quality.Total Data Quality Management Research Program, MIT Sloan School of Management, Aug.
Parsaye, K, Chignell, M, (1993). Intelligent Database Tools & Applications: Hyper Information Access, Data Quality, Visualization, Automatic Discovery. New York: Wiley.
Raab, L.(1992). Bounds on the Effects of Replication on Availability, IEEE Proceedings.
Ricciuti, M. (1993). How to Clean Up your Dirty Data, Datamation, August 15, 51-52.
Ricciuti, M. (1995). Data Replication: A risky Rx, Infoworld, July 3. 1,16.
Redman, T.C. (1992). Data Quality Management & Technology, Bantam Books.
Redman, T.C. (1998). The Impact of Poor Data Quality on the Typical Enterprise. Communications of The ACM, February, 41, 2, 79.
Sack, J. (1994). Organizational Issues in Distributed Computing. In Distributed Computing. Implementation and Management Strategies. Prentice Hall..
Strong, D., Lee, Y., Wang, R. (1994). Beyond Accuracy: How Organizations are Redefining Data Quality. Total Data Quality Management Research Program, MIT Sloan School of Management, September.
Suomi, R. (1994). What To Take Into Account When Building An Inter-Organizational Information System. Information Processing & Management, 30, 1, 151-159.
Te'eni, D. (1993). Behavioral Aspects of Data Production and Their Impact on Data Quality. Journal of Database Management, Spring, 4, 2, 30-38.
Umar, A. (1992). Distributed Computing: A Practical Synthesis. Prentice Hall.
Wand, Y., Wang, R. (1996). Anchoring Data Quality Dimensions in Ontological Foundations. Communications of the ACM, 39,11, 86-95.
Wang, R., & Kon, B. (1993). Towards Total Data Quality Management (TQDM). Total Data Quality Management Research Program, MIT Sloan School of Management.
Wang, R., Kon, B., & Madnick, S. (1993). Data Quality Requirements Analysis & Modeling. Proceedings of the IEEE 9th International Conference on Data Engineering,

Wang, R., Storey, V., Firth, C. (1993). Data Quality Research: A Framework, Survey, and Analysis. Total Data Quality Management Research Program, MIT Sloan School of Management, July.

Wang, R., Storey, V., Firth, C. (1995). Data Quality Research: A Framework for analysis of Data Quality Research, IEEE Transactions on Knowledge and Data Engineering, 7, 7, 623-640..

Wang, R., Strong, D. M., Guarascio L.M. (1994). Data Consumers' Perspectives of Data Quality. TDQM Working Paper . Total Data Quality Management Research Program. MIT Sloan School of Management.

Zmud, R.W. (1978). An Empirical Investigation of The Dimensionality of the Concept of Information. Decision Sciences, April, 9, 2, 187-196.

Zuboff, S. (1985). Automate/Infomate: The Two Faces of Intelligent Technology. Organizational Dynamics, NY: American Management Association.