AUTONOMOUS DATA QUALITY MONITORING AS A PROPERTY OF CODA ENTERPRISE ARCHITECTURE

Tereska Karran
Department of Informatics, University of Westminster, 115 New Cavendish Street, London,W1W 6UW, U.K.

Keywords: Information quality, enterprise architectures, CODA, business intelligence, autonomous data quality monitoring, data mining.

Abstract: Organisations are driven to meet data quality standards by both external pressures and by internal quality processes. However, it is difficult to add these to existing systems as part of enterprise architecture. Moreover, where business intelligence is delivered autonomously, it is difficult to implement efficient data quality processes. We discuss the value of making data quality part of enterprise architecture using CODA and outline how this can be implemented by adding autonomous processing to specific architectural elements.

1 INTRODUCTION

As database technology has developed and matured, it has become possible to extract intelligent information from data and to create self-adaptive systems based on analysis of past and present transactions. As a result, business information systems have become more focused on meeting information objectives at all levels of the organization. This in turn implies managing information flow within a complex architecture in such a way that it mirrors the way that the organization manages decisions. A great deal of management decision-making is based on information which is automatically extracted from data stored within the database via a warehouse or data mart. Therefore it has become increasingly important to monitor and control the flow of data through the organization because detail is lost as it becomes increasingly aggregated and generalized. During this process it is easy to lose track of the ownership and responsibility for errors as well as letting errors creep in. In line with this the auditing and IT governance procedures have changed so that it has become necessary to show the accuracy and the ownership of data behind complex business reports. It is no longer sufficient to audit operational the core transactions within computer systems since errors can occur as information is passed from level to level (Sayana (2002)).

A successful enterprise needs a transparent data architecture which incorporates a strategy for the management of information quality (Rees & Dineschandra. (2005)). However, this architecture should also monitor information at each stage of the data transformation process. Several serious failures within organisations can be pinpointed as originating in the mismanagement of information (Sercu et al. (2006)). It often proves particularly difficult to establish the causes because, in general, flows of information cannot be easily traced due to the lack of transparency in the architecture.

There are two solutions.

The first solution is to ensure that the data is fit for the purpose. This involves establishing objectives and data ownership at a business level and then creating data cleaning processes for each layer of the enterprise. Thus it becomes possible to redefine data for each separate objective.

The second solution is to measure the data quality at each level of the architecture. This involves customised correction of data as it moves through every level of the enterprise. However, it is difficult to do this as a business process since the warehouse aggregation and data gathering process is usually set to operate as an autonomous process.

It would be most efficient to incorporate both solutions within the enterprise architecture. If data quality is part of a layered architecture, it becomes possible to improve system performance globally.

Karran T. (2008).
AUTONOMOUS DATA QUALITY MONITORING AS A PROPERTY OF CODA ENTERPRISE ARCHITECTURE.
In *Proceedings of the Third International Conference on Software and Data Technologies - ISDM/ABF*, pages 112-117
DOI: 10.5220/0001875301120117

Where data ownership is managed within the architecture it is possible to manage data correction in such a way that the data meets specific objectives.

2 A GENERIC ARCHITECTURE FOR THE ENTERPRISE

An enterprise architecture needs to support the mechanisms for turning source data into information and then into knowledge that is useful for decision making. BI architecture can be approached as a number of transactional data sources connected and integrated by an ETL layer feeding a data warehouse that holds aggregated, transformed, multidimensional data which in tern is used to feed different data marts used for analysis such as queries, OLAP, and data mining cubes.

Because of the multiple purposes to which information is put, and because of the way that new information can have multiple impacts, an enterprise architecture can be seen as a type of complex non-deterministic system. This allows us to use complex system theory allows us to model the enterprise architecture as consisting of multiple interacting and semi autonomous databases (Ramanathan (2005)).

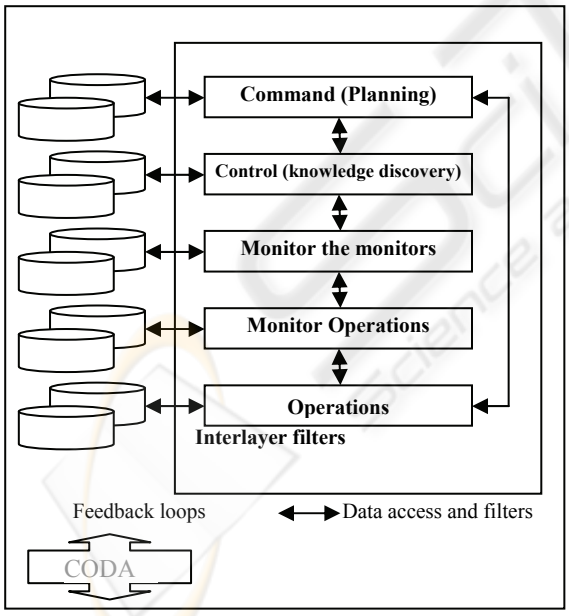

Figure 1: The CODA Architecture showing layers.

At the highest level, an architectural strategy for data quality involves the autonomous and strategic management of data passing through enterprise layers. There are two steps for doing this effectively. Firstly, the automatic filtering of information as it enters the organization. Secondly, the controlled manipulation of information flows within the architecture matching the organizational decision flows. There are several different enterprise architectures on offer

Dill et al (2004) describe the ESA (Enterprise Service Architecture) and show how the addition of a service-oriented BI improves data quality assurance. However, the processes are not viewable and their impact on data flows is not easily managed.

Wu et al (2007) have proposed a Service Oriented Architecture which breaks legacy data components down into service-oriented reusable components communicating with each other using XML messages. IEach of these has a specific functionality such as Data Cleaning, Data Transformation, Data Load and other services. The architecture was used for assessing IT service management performance. However, it does not allow direct communication between data integration and data analysis services except through a centralised data store.

It is advantageous to have a cellular structure with an viewable interface so that it is possible to view the data transformations. Such a structure is available in the CODA enterprise architecture.

CODA is a generic intelligent architecture for use in enterprise applications (Karran et al (2003)). It has been applied in a variety of research contexts (Karran & Wright (2007)) (Karran &Justo (2003)). It is designed to be added to existing systems with a minimum of interference with existing information processing. It also uses autonomous processes for the self management of error. Most enterprise architectures contain three layers, but at the macro level CODA consists of five processing layers (shown in figure 1).

2.1 CODA Layers

Each layer is made up of processing cells. The layer functions can be loosely described as transactional, decision support, executive information, knowledge discovery and planning. These five layers correspond to systemic functions described by Beer and by Espejo (Beer (1998), Espejo & Gill (1999)).

2.2 CODA Cells

At the micro level the smallest atomic component of the CODA architecture is a cell. The generic

structure of a cell is shown in figure 2. The cell grain depends on how the enterprise manages its information processing involving stored data. Hiding the details of the processing within a cell wrapper enables the architect to concentrate on the flows of information and the types of data flowing between the cells with a view to making this process autonomous.

The cell structure is the key to monitoring data quality. Each cell is initiated with processing objectives by an activating role (Megaache, Karran & Justo (2002)). Each cell passes its execution status (a minimum of three conditions: opportunity, threat or normal, with a maximum of five) to a senior cell for intelligent monitoring. As data passes into a cell it goes through the cell filter and is given a quality status (red, amber or green depending on the quality settings defined at the filter). All threat and opportunity data is automatically monitored as a priority, while normal data is placed at a lower priority. At the same time the process is given a condition setting allowing senior cells to know how well the objective is being executed.

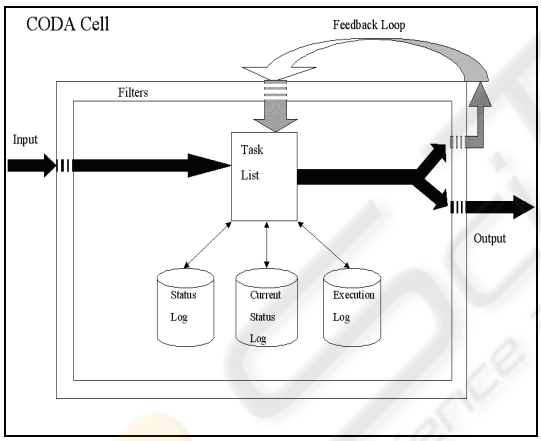

Figure 2: Structure of a CODA Cell.

2.3 Structure of a CODA Layer

Figure 3 shows the structure of a layer, managing information for decision support objectives. Each layer consists of multiple interacting component cells which manage the information needed by a specified user. The user is defined by the user profile which consists of security protocols and permissions as well as a history of accesses. Within each layer the component cells continuously adapt their behaviour based on critical tolerances which are managed by cells at 'senior' layers.

An important advantage of layering the information flows is that it becomes possible to add new component 'cells' and test their effects on the enterprise with the minimum of side effects (Bedau (1997)). Once this has been established, it is possible to design and test both individual elements and the whole system in a variable environment (Bedau (1996)).

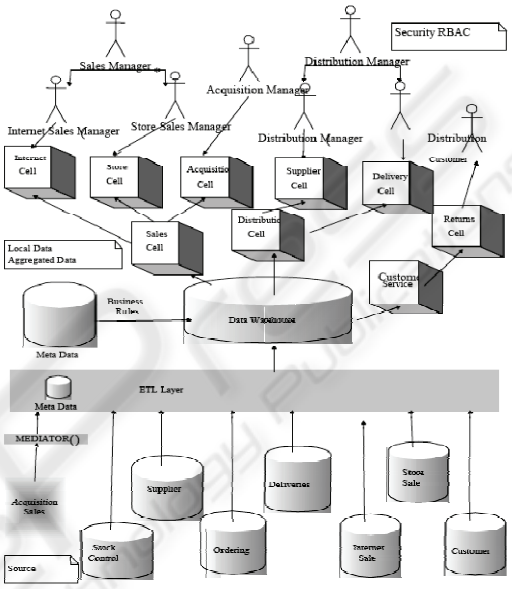

Figure 3: Data Warehouse Architecture showing a Decision support Layer composed of multiple cells.

3 ADDING AUTONOMOUS DATA QUALITY MONITORING TO AN ENTERPRISE ARCHITECTURE

We have outlined how the management of information quality is a key property of complex enterprise architecture. The complete information quality management process involves designing and managing multiple views of the original input data each with their own data quality requirements. Each of these may be modified as a result of the intelligence gained by cells at the higher layers. Quality has to be strictly managed at each cell and each layer because although the source input data is likely to be highly detailed, it becomes highly generalised as it moves through the decision layers[1].

[1] We assume that information is logically structured so that the steps from the detail to the general are logical.

The first step is to wrap existing database systems and data marts using a cell structure (Karran & Wright (2007)). Each cell manages its data-flow autonomously providing it has been initiated by a role with the required access permissions. Each cell can assess the data quality which passes through its 'cell membrane' and returns a quality value for data it uses. Data which does not meet predefined quality settings is transferred to holding bay for further cleaning and at predefined intervals (depending on user settings), the dirty data is passed to a higher level cell for further cleaning. As well as dealing with the error and the number of errors, the higher level cell is able to assess and count the type of errors reaching that layer. If necessary, a new data cleaning filter can be added to the lower level cell to prevent this type of error from occurring in future.

The layered cells allow the organization to develop a strategy for managing day to day transactions. At the same time the internal monitoring procedures of the cell membranes allow the organization to respond to unforeseen situations, and to abnormal or incorrect data. At the higher layers, the intelligence cells may base their decisions on generalised versions of core information, but it is also possible to drill through the underlying layers to obtain the source data on which these decisions were made. The rules for decision making are refined at the control layer, depending on what is needed for extrapolating rules about the environment, monitoring patterns of behaviour, and adapting to circumstances.

Data quality is a fundamental property of good intelligence and is managed in a layered way so that the source of the generalized rules can be retrieved and tested if necessary. Since data quality is also vital to ensure a holistic and speedy adaptive response once new information has been discovered, there are efficient feedback loops linking the layers. The whole process is dynamic since decisions made must result in improvement and the intelligence cells can test this. The system will become increasingly autonomous because it is possible for the top layer to test the rules and predictions made by the lower layers and to replace poorly performing strategies with better adaptations based on analysis. This data quality cycle holds true for any complex information system and not only business enterprises.

4 ADDING AUTONOMOUS DATA QUALITY MONITORING TO AN ENTERPRISE ARCHITECTURE

Testing of the architecture has so far been confined to specific architectural elements. It is important to note that there are several different strands to the automation process for data quality. For example, there is a distinction between architectural elements which deal with the intrinsic quality and those which deal with the data requirements of the user/role. For example, in figure 3 above, each user has a security profile detailing accesses and permission to cells. At the same time, the enterprise layer has it own data quality requirements. It is anticipated that each layer will include a data mining tool which analyses user accesses and transactions. It is envisaged that this may become part of the meta data catalogue which stores details about the quality of the data. It should help to ensure that users receive the right data in a timely manner. At the same time, it is possible to monitor for potential unauthorised access.

4.1 Managing Database Autonomy

The concept of managing databases autonomously is still in its early stages. This is because the data held by the organization is a primary resource and any change is potentially dangerous to the integrity of the existing systems. However, we have explored how it is possible for a remote agent, initially the database administrator, but potentially an autonomous cell, to administer the database logs in terms of simple errors, and other log conditions (Karran & Wright (2007)). We have shown that it is possible to design and build a CODA type cell which is able to provide advice for the database administrator away from the console.

4.2 Creating Autonomous Filters for Managing Data Quality at ETL Layer

The methods and principles of autonomously filtering data have been applied using Oracle and Excel formats for project metrics in a large software house maintaining third party software(unpublished research Ragaven (2005)).

The original source data was huge, error prone and duplicated. Each project had a separate metric

spreadsheet. The template used as a starting point for each new software maintenance project was not accurate. The result was that time was spent in re-defining MS Excel macros to generate derived data for analytical purposes. Designing errorless macros to do this work required expensive training, and, because of high staff turnover, calculated management reports contained inaccurate figures. Lack of integration of data in multiple spreadsheets made it difficult to spot patterns and trends to estimate future project resource use, such as cost and manpower. There was no security so that sensitive data such as daily/hourly cost for staff were not protected. Finally, there was no data integrity due to the lack of efficient in-built validation in the spreadsheet application. Backup and restore proved difficult due to the vast number of individual spreadsheets and where spreadsheets had shared access, the system crashed regularly.

The solution was implemented in three stages. The first step was to ensure quality of collected and stored data. The second step was to filter and consolidate data in a meaningful way for use at decision support level. The third step was to add role profiles to ensure that the right data should be available in a secure way.

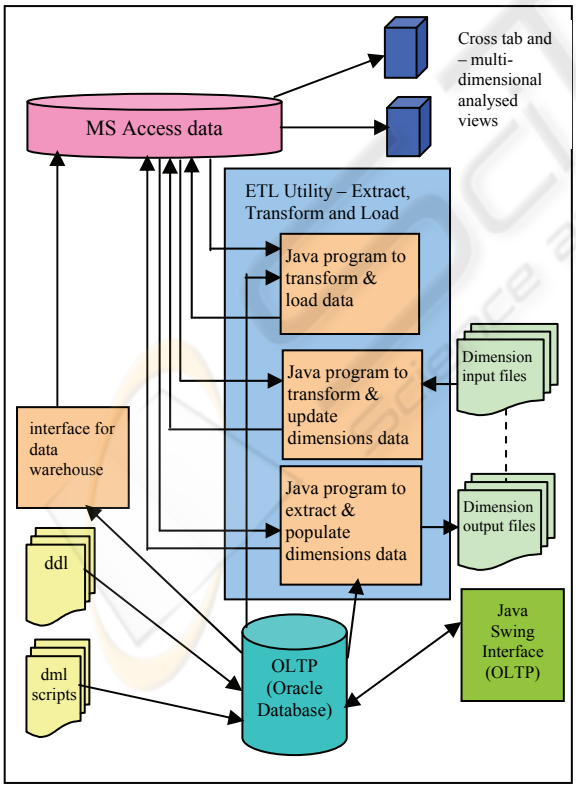

Figure 4: Autonomous Transformation of Data for the Decision Support Layer (Ragaven 2005).

It was possible to ensure the quality of OLTP data by creating a user-friendly graphical interface as front-end filter for entering metrics together with an appropriate database as back-end for storing data. The front end, designed in Java-swing, allowed a single point data entry system to record metric data and the data was autonomously transformed to meet the needs of the users at the decision support layer (as shown in figure 4.). This data was moved to the data warehouse using java processes to add missing data and dimensions. The cleaned data was then loaded to spreadsheets customised for specific decision support roles.

The project showed that it was useful to apply an autonomous data filtering program to a layer and to tie data access to specified roles.

5 RESULTS

The project developed a single set of cells with viewable and autonomous data cleaning interfaces. The application involved a single role to manage an ETL utilities workflow metrics system. The process cells were designed using JAVA and extracted transformed and loaded project metrics data into a business intelligence data warehouse adding multidimensional data structures. The data in the business intelligence warehouse was used to support management decisions involving project work flows, risks and costing. The overall results obtained from the system were meaningful because they mirrored the workflow of the business decision processes exactly. This was possible because the Java based cells mirrored the role actions. The ETL utility programs were fast, reliable and automatic and transformed the OLTP core data in a way which was intuitive and clear to the management roles using it. The data moving between the cells is viewable at all stages so that data quality can be assured. For example, the date is defaulted to the project inception date, and the system automatically prompts for data quality checks. The information available allows the user to check and confirm, not only current system releases under test, but also past system releases, which were under test.

By making the system both autonomous and close to the business workflow test manager were able to combine information drawn from multiple sources and analyse metric data without complex external checking of sources. For example, chargeable, non-chargeable and overall cost for each

staff can be viewed in terms of staff ids for each system release and this improves the understanding of project actual costs. This provides useful information to monitor performance of staff within the organization because the data quality is reliable, timely and up to date.

We were able to test the principles for cell design and role activation for managing data quality autonomously. The application showed that it is possible to obtain good results for managing data by separating OLTP and OLAP filters and attaching data to a specified role. So far we have only developed a single role view of the system. The next stages will involve attaching quality status to data and passing the data to a further layer. It will also involve adding further roles and showing the interactions between them.

6 CONCLUSIONS

The application of Beer's model to an enterprise architecture in the CODA application has so far produced a number of interesting results. Initial research suggests that the model has some bearing on the problems facing large organizations in adding data quality processes to existing enterprise architectures. Initial trials have proved promising and have been enthusiastically adopted by the organizations involved.

REFERENCES

Bedau M.A., (1997) Emergent Models of Supple Dynamics in Life and Mind. *Brain and Cognition* 34: 5-27.

Bedau, M. A., (1996) The Extent to which Organisms Construct their Environments Adaptive Behaviour 4: 476-482.

Beer, S. (1989) *The Evolution of a Cybernetic Management Process* in R. Espejo and R. Harnden (Eds) Interpretations and Applications of Stafford Beer's VSM : 77-100. Wiley.

Cochran, W G (2001) *Sampling Techniques*, Wiley

Chatfield, C (2001) Time-series Forecasting, Chapman and Hall/CRC Press

Dill et al., (2004). Service-oriented BI. SIGMOD. (online) Availablefrom: http://portal.acm.org/citation.cfm?id=1007568.1007683&coll=ACM&dl=ACM&type=series&idx=SERIES473&part=series&WantType=Proceedings&title=SIGMOD (Accessed 29 January 2008).

Espejo, R., Gill, A. (1997) The Viable System Model as a Framework for Understanding Organisations. Phrontis .com.

Karran, T., Wright, S., (2007) Towards Database Autonomy: An Approach Using Complex Organic Distributed Architecture (CODA) *EEE'07 World Comp 07*

Karran, T., Madani K., Justo, G. R., (2003) CODA - Self Learning and Adaptive Systems in Dillinger M. (ed) Software Defined Radio, Wiley.

Megaache, S., Karran, T., Justo, G. (2000) *A Role-based Security Architecture for Business Intelligenc,* Conference Proceedings Tools 2000.

Moen, R D; Nolan, T W & Provost, L P (1991) Improving Quality Through Planned Experimentation, McGraw-Hill

Ragaven A. (2005) Developing Workflow Metrics for a Workflow Metrics system (unpublished MSc project)

Rees, M. and Dineschandra, J. (2005) Monitoring Clinical Performance: The role of software architecture. *Journal of Healthcare Management Science.* Vol 8, p197 – 203.

Sayana (2002) Auditing General and Application Controls. *Information Systems Control Journal*. Vol 5.

Saaty, T (2001) Decision Making for Leaders: The Analytic Hierarchy Process for Decisions in a Complex World, *Analytic Hierarchy Process Series* Vol. 2

Wu et al., (2007). A service-oriented architecture for Business Intelligence. IEEE International Conference on Service –oriented computing and applications. (online) Available from: ieeexplore.ieee.org/iel5/4273393/4273394/04273437.pdf (Accessed 19 January 2008).