Enterprise Architecture Analysis for Data Accuracy Assessments

Per Närman, Pontus Johnson, Mathias Ekstedt, Moustafa Chenine, Johan König
Department of Industrial Information and Control Systems
Royal Institute of Technology (KTH)
Stockholm, Sweden
{pern, pj101, mek101, moustafac, johank}@ics.kth.se

Abstract - Poor data in information systems impede the quality of decision-making in many modern organizations. Manual business process activities and application services are never executed flawlessly which results in steadily deteriorating data accuracy, the further away from the source the data gets, the poorer its accuracy becomes. This paper proposes an architecture analysis method based on Bayesian Networks to assess data accuracy deterioration in a quantitative manner. The method is model-based and uses the ArchiMate language to model business processes and the way in which data objects are transformed by various operations. A case study at a Swedish utility demonstrates the approach.

Keywords: Enterprise Architecture, Accuracy, Data Quality

I. INTRODUCTION

In recent years, software, system and Enterprise Architecture (EA) have become established disciplines in both industry and academia. Architecture models can aid the communication between various stakeholders and facilitate comprehension of the properties of the complex systems they represent. One important part of this comprehension is to be able to analyze the models.

However, common EA frameworks such as the Zachman framework [1], TOGAF [2] or DoDAF [3] rarely take architecture analysis into consideration, at least not explicitly. Modeling and metamodeling are often important parts of these frameworks and they commonly use views and viewpoints as a way of implicitly introducing the reasons and purposes of EA models. However, it is hardly ever explicited how the analyses should be carried out from any specific viewpoint.

The present work builds on previous research within the field of architecture analysis where architecture models are analyzed using a formalism based on Bayesian statistics [4][5] this approach allows the analysis of various system properties, such as the interoperability, information security and the availability of software systems. This approach is also explicit on what kind of information architecture models need to contain in order to be useful in an architecture analysis.

One aspect frequently overlooked in the architecture analysis community is the aspect of data accuracy. While of great importance to the business, data accuracy is rarely addressed by any major architecture framework or architecture analysis method.

This paper describes how the Bayesian approach can be used for architectural analyses of data quality. The paper focus primarily on the accuracy of data and how it deteriorates in a business process involving multiple automated and manual processing steps. Based on a process-centric view of data accuracy this paper presents a core model based on ArchiMate [6] for accuracy architecture analysis. A case study performed at a Swedish utility illustrates the approach for one particular process, the outage reporting process.

The next section provides some background to the architecture analysis approach on which this paper is based and touches briefly on ArchiMate. Section 3 presents some related works and gives an overview of common data quality estimation methods and theories. Section 4 proceeds to detail a so called abstract model with which it is possible to model and analyze data accuracy. Section 5 describes a case study where the method is applied followed by Section 6 which concludes the paper and proposes future works.

II. ABSTRACT AND CONCRETE MODELS

An abstract model is an EA metamodel containing entities and entity relations, augmented with attributes and causal relations between attributes. The attributes and attribute relations correspond to nodes and causal relations in Bayesian networks [7], [8]. By “metamodel” we mean a strict set of constructs and rules that can be used to model a specific domain, we equate meta model to the Unified Modeling Language constructs, relations and rules which can be used, for example, to define a model of a specific software system [9].

Classes are fundamental parts found in most metamodels. Classes represent the objects of interest when modeling, e.g. application, services, persons, or processes. Classes in abstract models are similar to classes found in UML.

Class relations connect two entities, e.g. “Interface is provided by Application” or “Person is a resource of a Process”. Class relations also state the multiplicity of the relationship between the Classes, e.g. that one person can be the resource of zero or more processes.
Class relations are similar to class relations found in UML.

Attributes of an abstract model represent variables related to the classes. UML also have attributes related to classes, but the attributes in abstract models differ from the attributes in UML. In abstract models, attributes and attribute relations represent the nodes and relations of a Bayesian network, see below. A richer account of abstract and concrete models is found in Johnson et al. [7].

### A. Bayesian networks

Friedman et al. [10][12][14] describes a Bayesian network, $B=(G, P)$, as a representation of a joint probability distribution, where $G=(V, E)$ is a directed acyclic graph consisting of vertices, $V$, and edges, $E$. The vertices denote a domain of random variables $X_1,\ldots, X_n$, also called chance nodes. In the context of abstract models, each chance node corresponds to an attribute. Each chance node, $X_i$, may assume a value $x_i$ from the finite domain $\text{Val}(X_i)$. The edges denote causal dependencies between the nodes, i.e. the causal relations between the nodes. The second component, $P$, of the network $B$, describes a conditional probability distribution for each chance node, $P(X_i)$, given its parents $\text{Pa}(X_i)$ in $G$. It is possible to write the joint probability distribution of the domain $X_1,\ldots, X_n$ using the chain rule of probability, in the product form

$$P(X_1,\ldots, X_n) = \prod_{i=1}^{n} P(X_i | \text{Pa}(X_i)).$$

In order to specify the joint distribution, the respective conditional probabilities that appear in the product form must be defined. The second component $P$ describes distributions for each possible value $x_i$ of $X_i$, and $\text{pa}(X_i)$ of $\text{Pa}(X_i)$, where $\text{pa}(X_i)$ is the set of values of $\text{Pa}(X_i)$. These conditional probabilities are represented in matrices, here forth called Conditional Probability Matrices (CPMs). Using a Bayesian network, it is possible to answer questions such as what is the probability of variable $X$ being in state $x_1$ given that $Y = y_2$ and $Z = z_1$.

In the general case, the relations between variables described by the CPMs can be arbitrarily complicated conditional probabilities. Table 1 contains a general description of a CPM for the three nodes $X$, $Y$ and $Z$ where the first is causally affected by the other two.

<table>
<thead>
<tr>
<th>$Z$</th>
<th>$z_1$</th>
<th>$z_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Y$</td>
<td>$y_1$</td>
<td>$y_2$</td>
</tr>
<tr>
<td>$X_1$</td>
<td>$P(x_1</td>
<td>y_1,z_1)$</td>
</tr>
<tr>
<td>$X_2$</td>
<td>$P(x_2</td>
<td>y_1,z_1)$</td>
</tr>
</tbody>
</table>

More comprehensive treatments on Bayesian networks can be found in e.g. Neapolitan [11], Jensen [12], Shachter [13] and Pearl [14].

### B. Creating concrete models

The abstract model tells us what information we need to find in order to conduct analyses of different variables. Once this information is collected it is specified in the model, thus creating an instantiation of the abstract model. Once a concrete model has been created we can employ Bayesian inference to calculate the values of the attributes of the model.

In Fig.1 below we see a Bayesian network stating that the property $X$ of Applications is affected by property $Y$ of Servers and property $Z$ of Operating Systems. This can be transformed into an abstract model where Application, Server, and Operating System are modeled as classes and the properties as class attributes. Finally, when instantiating the abstract model into a concrete model, the class attributes assume values either through data collection or through inference using conditional probabilities.

![Figure 1](image_url)
C. The core of the metamodel: ArchiMate

The ArchiMate EA metamodel has been used as a basis for the development of the abstract models. ArchiMate is an EA framework [15] developed by the ArchiMate Foundation consisting of primarily Dutch organizations. ArchiMate is also about to be published as a Technical Standard by The Open Group. ArchiMate is a mature modeling language and its applicability as a core language for abstract models has been tested in previous projects [16].

The ArchiMate metamodel distinguishes between three layers; the business, application and technology layer, where the technology supports the applications and the applications support the business.

The entities of the layers are in turn categorized according to three aspects of EA; the passive structure modeling informational objects, the behavior structure modeling the dynamic events of the EA and finally the active structure which models the components in the architecture that perform the behavior aspects.

The service concept has been used extensively in the ArchiMate language. Services encapsulate and hide the internal behavior of underlying layers and give the overlying layers access to functionality from underlying layers through well-defined interfaces. For instance, an Application Service is seen to encapsulate a number of internal Application Functions to make them useful to the business actors of the business processes. By using services the business actors neither have to consider the internal behavior of the information systems nor the technology used to realize the behavior. Fig. 2 shows the original ArchiMate metamodel [6].

III. DATA QUALITY AND ACCURACY

Data quality is a multi-faceted concept. The most common dimensions of data quality are completeness, consistency, currency relevance and accuracy [18] [19][20][21][23]. In this paper the accuracy dimension is the main focus.

A. Accuracy

According to [18][23], accuracy is defined as the degree of proximity of a value $V$ to $V'$, where $V'$ is an actual concept in the domain of reference and $V$ is a datum that represents it. So if the datum $V$ has a value of 200 and its frame of reference $V'$, the “truth” so to speak V is 199, V is inaccurate. When reasoning about classes of data values, accuracy instead becomes an issue of probability: the probability that a value selected...
from a class of values is accurate according to the above definition.

B. Models for data quality analysis

There are several modeling techniques proposed for data quality analysis. The Quality Entity Relationship (QER) model is an extension of the ER models to accommodate several data quality dimensions [19]. With QER the ER models are extended to incorporate data quality related information on attributes in the relational model. The attributes in relations are associated with quality indicators (e.g., accuracy) and quality rankings (e.g., excellent).

The QER model lacks the ability to incorporate information on the origin of the data. Tracing the origin of data is known as data provenance [18] and is important in most systems where data is collected from various distributed sources with a varying degree of data quality.

The earliest attempt in tracing data provenance is the Polygen model [19] that is more geared to analyze data quality in distributed heterogeneous data sources. The Polygen model is a relational model that defines a set of operators (e.g., union, Cartesian product, difference, etc) based on relational algebra that can semantically annotate the propagation of data.

While the QER and Polygen Models are expressive and applicable to the database relational domain, they are less applicable to distributed enterprise information systems in general. In such systems the data structure is not necessarily defined in relational algebra and could be more the result of an aggregation of data elements in a process which could in turn also be a result of several processes.

Information Product Maps (IP Maps) [22] accommodate such process based modeling of data. IP Maps are graphical models that treat data as a product from one or several processes where the input was raw data. This is analogous to the manufacturing process where raw materials are inserted and a finalized product exits the manufacturing line after a series or manipulation and modification. Such manipulations can be represented in the IP Maps model with the use of specific constructs to represent the process and actions on the data, e.g., processing, quality check, data source or data receiver. Each construct in term has associated metadata which can be used in a model to specify the construct.

The strength of the IP Maps is the ability to portray the data provenance as well as the components and elements in the process that manipulate the data. The IP Maps model has been extended into a IP UML profile [20] to make use of UML’s richer semantics and to conform to the de-facto standard that UML is.

In the UML model the data units are associated through a quality association with a stereotyped quality dimension class e.g., timeliness. By using UML the IP UML provides the opportunity to model data units in interaction diagrams, to observe the data flow between object calls and in activity diagrams. Both IP Map and IP UML lack the ability to perform quantitative analysis of data accuracy in a process. Instead, there function is mostly to visualize data quality problems and requirements and aid software engineers in designing better information systems.

The assessment method that is presented in this paper is similar to the IP Maps and the IP UML model in the sense that it is aimed at the management or Enterprise Information systems by visualizing processes and data quality.

The main contribution of the method presented here is to offer a way to quantitatively assess data accuracy across a business process. Unlike the Polygen model which also features quantitative data quality assessments using its relational algebra but only on the database level the method presented here captures the impact on data accuracy from various behavioral elements in a business process. Furthermore, this method incorporates constructs from the ArchiMate language, and is as such a first attempt to use EA models for data accuracy assessments.

IV. AN ABSTRACT MODEL FOR ACCURACY ASSESSMENTS

This chapter introduces an abstract model with which it is possible to describe how the manipulation of data across a business process gradually deteriorates the data accuracy.

A. Abstract model structure

The abstract model consists of six modeling entities, each with an “Accuracy” attribute. Of the six, three belong to the passive, or informational structure of ArchiMate, and the other three to the behavior structure [6], see Fig. 3.

1) The passive objects.

The “main” passive objects in the abstract model are the Business Objects which are classes of pieces of information that are important to the business.

In the Application layer, Business Objects are realized by Data Objects, which are
elements stored in databases. In the business layer, the Business Objects are realized by Representations, which represent regular documents used by the business.

Both Representations and Data Objects model classes of information objects. Both Data Objects and Representations may be composed of other Data Objects or Representations respectively. We will denote the lowest level of abstraction for this modeling as the “atomic” passive objects. ArchiMate is not able to represent the instance level, i.e. the physical instances of the classes that are represented as Data Objects (see Fig. 4).

![Figure 3: An abstract model for accuracy assessments.](image)

Although the instances will not be modeled in this paper either, they are crucial for the definition of accuracy. Instantiated Business Objects, i.e. concrete pieces of information, are the “true” pieces of information that constitute the frame of reference against which the accuracy of the other passive components will be judged. An instantiated Data Object is a physical piece of data stored in a database which realizes a Business Object instance. An instantiated Representation is a concrete document realizing a Business Object instance. See Fig. 4 for an example of how instances relate to the ArchiMate model.

2) **Accuracy for the passive objects.**

On an instance-level, “Accuracy” is an attribute of the passive components which can assume the states “Accurate” and “Inaccurate”.

In the abstract model, which deals exclusively on the class level, the “accuracy” of a Data Object or a Representation is the probability that a Data Object or Representation belonging to the class is in the state “Accurate”. This can be estimated through the frequency of errors encountered [22]:

\[
P(\text{Passive Object Accuracy = Accurate}) = \frac{\text{Number of erroneous Passive Object instances}}{\text{Total number of Passive Object instances}}
\]

3) **The behavior components.**

The passive components of ArchiMate are read or written by behavioral components. The entities considered by this abstract model are Business Processes, Application Services and the
Application Functions. A Business Process is a set of activities performed by users that transforms input into some kind of output.

Business Processes use Application Services. Application Services describe the behavior of an application from an external perspective and is defined as a unit of functionality that is meaningful to (business) users. Application Services are realized by one or several Application Functions. An Application Function describes a well-defined piece of internal behavior of an application.

All of the behavior components can access the passive components somehow. Business Processes can read or write Representations. Application Functions and Services can read and write Data Objects, and sometimes (in the case of scanning or printing functionality) read and write Representations as well.

4) Accuracy and behavior components.

Behavior components influence the accuracy of the information entities through “write” relations, (which are specializations of “access” relations in ArchiMate). Changes in accuracy happen for instance when Application Functions fail to read certain Data Objects or when someone in a Business Process makes a mistake. In order to model this dependency, all of the active components contain “execution accuracy”.

On an instance level, i.e. for a single execution, behavior components can be either “accurate”, meaning that they transform input data to output data according to what is specified, or “inaccurate”, meaning that the behavior component fails to meet the requirements.

On the class level, i.e. what is shown in this abstract model, the execution accuracy attribute describes the probability that, given correct input data, the Business Process, Application Function or Application Service yields accurate output data. The probabilities are calculated as frequencies analogous with the procedures used for the passive components.

Taking the view that data is consumed and produced in Business Processes we assume that the execution accuracy of a Business Process can be measured by how much output data is deviating from the correct value. This is often done within the process management community [24]. The execution accuracy of an Application Function is defined in a similar way, which makes it similar to the “Computational accuracy” of ISO 9126-2 [17] standard. This measures accuracy in terms of number of correct executions divided by the total number of executions. “Correct” is here defined according to what the application’s specification considers to be correct given the input data. Application Services are merely bundles of Application Functions, and their accuracy is defined in terms of Application Function accuracy.

All the classes and relations in this abstract model are consistent with the ArchiMate metamodel as presented in [6], except for the access relation between the Application Service and the Representation which has been added here.

B. Calculation framework

As discussed in chapter 2, attribute relationships are defined by Bayesian network logic. In addition to this there are several rules for how to perform the accuracy assessments.

To be able to show how the accuracy of data deteriorates in a process it is necessary to show how passive components are transformed by active, behavioral components in a process. We therefore introduce the notion of time in our models. Whenever a passive component at process step t is modified (written to) by a behavior component, a new instance of the passive component will be modeled again at the subsequent process step t+1.

Although this paper represents the passive objects at different discrete process steps as new Bayesian nodes, an alternative approach could use Dynamic Bayesian Networks [25]. Whenever there is a “write” entity relation between an active component such as a Business Process and a passive component such as a Representation, this is accompanied by an attribute relation from the active components execution accuracy to the accuracy of the passive component.

As has been stated already, passive components sometimes are composed by other passive components. Whenever the composition relation exists, it is the accuracy of the “atomic” passive components on the lowest level of abstraction which will be transmitted to the next generation of passive components, see Fig 5.

A passive object on the instance level is accurate if and only if (i) it is composed by other passive objects, all of which are accurate, (ii) the behavior component that write the passive components must be accurate while doing so. So, a behavior component that performs an inaccurate “write” operation on a passive component makes the passive component inaccurate, and (iii) it was not inaccurate previous to being read and written by either one of the behavior components.
Stated more formally using the Object Constraint Language (OCL), the rules of computations are the following for a passive object at process step $t$:

\[
\text{context } \text{object}[t]: \text{PassiveObject at process step } t \text{ inv:}
\]

\[
\begin{align*}
& \text{if } (\text{object}[t].\text{composedOf} \rightarrow \text{forAll}(\text{object}[t]: \text{PassiveObject at process step } t | \text{object}[t].\text{accuracy} = \text{true}) \\
& \text{and } \text{object}[t].\text{writtenBy} \rightarrow \text{forAll}(\text{object} | \text{BehaviorObject}(\text{object}, \text{accuracy} = \text{true}) \text{ and } \text{object}[t-1].\text{accuracy} = \text{true}) \text{ then } \\
& \text{object}[t].\text{accuracy} = \text{true} \end{align*}
\]

Endif

These "all-or-nothing" rules have implications on the underlying Bayesian network used for the computations of the concrete model. Specifically, the CPM of the accuracy nodes will consistently describe an AND-relation; the node is accurate if and only if its predecessor nodes are accurate as well.

Table 2: A CPT describing an AND relation as used in Figure 5. All predecessor nodes need to be in the "Accurate" state in order for the passive component "XY" to be accurate at process step $t=2$.

<table>
<thead>
<tr>
<th>Accuracy of behavior component X (t=2)</th>
<th>Accurate</th>
<th>Inaccurate</th>
<th>Accurate</th>
<th>Inaccurate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy of passive component X (t=2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accurate</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Inaccurate</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

The concrete model models classes of passive objects which contain information about the probability of the See Table 2 for an example of an AND function. The AND relations can be used on the class levels as well as on the instance levels; an AND CPM can be used to multiply probabilities, which is the same as calculating the intersection of independent events.

V. THE CASE STUDY

The case study was performed at a medium-sized electrical utility in Sweden. The process of interest was the reporting process of power outage information to the authorities. Power outage information is primarily concerned with assessing the quality of service offered by the power grid operator to its customer and uses a set of predefined reliability indices to express this. These indices show the frequency and duration of outages as well as the number of customers affected which are based on compilations of all known outages within a predefined period of time. Below follows a brief description of the relevant Business Objects followed by a description of the process itself.

A. The Information Architecture

The information architecture describes the structure of the passive objects in the case study. Every unit of information that is of interest to the business is represented as a Business Object. An Outage Ticket is composed of an Outage Time and data on the affected Customer. Initially, the Outage Time only includes the time the power outage was reported. In order to conduct maintenance to resolve the outage a Work Order is created. The Work Order includes the Outage Ticket as well as the specifics on the Maintenance Task that is required to resolve the outage. When the maintenance has been completed and the power is restored, the Outage Ticket is updated with the time of power restoration, and the Maintenance Task is updated with details concerning
the maintenance that was performed. The information contained in multiple Outage Tickets is subsequently aggregated to the indices that go into the Outage Report.

B. The outage reporting process.

The process (Fig 6.) begins when the Create Work Order Business Process is initiated. This business process here uses the Create Work Order Application Service that is one of the services offered by the Maintenance Management System Application Component (Application Components are not part of the abstract model, but are modeled here for easier process comprehension). The Create Work Order Application Service reads an Outage Ticket Data Object (at process step t=1, see Fig. 6) and in turn creates a new Work Order Data Object (process step t=2), containing data from the Outage Ticket as well as new data from the Create Work Order Business Process, specifically the Maintenance Task Data Object (process step t=2).

Once the Work Order has been created, the Resolve Outage Business Process is initiated when a field crew reads a Work Order Data Object (process step t=2) through their Maintenance Field Support Application Service which is another service offered by the Maintenance Management System. When the outage is resolved, the Resolve Outage Business Process updates the Work Order Data Object (process step t=3) through the services provided by the Maintenance Field Support Application Service. The data updated here is the details on the completion of the Maintenance Task, which is updated in the Maintenance Task Data Object (process step t=3). The Outage Ticket Data Object (process step t=3) is also updated with the time of power restoration.

To be able to generate the final Outage Report there is a need to import Work Order information, and in particular the Outage Tickets into the Network Information System (NIS), which is a Geographical Information System with extra functionality tailored to display distribution grids on maps, perform network calculations etc. Currently the only method available for importing the work orders to the NIS is by compiling them, printing them and then manually inserting them into the NIS database.

Before printing the work orders, they are aggregated and compiled into a more suitable format. The documents are compiled via the Word File Generator - a collaboration between MS Word and the MMS Application Components, offering the Application Service Generate Work Order Compilation to MS Word.

An Application Service Print, a service assigned to MS Word, prints out the compiled Work Orders Data Object (process step t=4) onto paper. In Fig. 6 this is illustrated as the Compiled Work Order Representation at process step t=5.

The final Business Process Create Outage Report includes the steps of importing Outage Ticket information into the NIS and thereafter generate the Outage Report Data Object (process step t=7).

The information on the printed documents is manually typed into the NIS, here represented as a read relation between the Create Outage Report Business Process and the compiled Outage Ticket Representation (process step t=5)(which in turn is a part of the compiled Work Orders.

The Create Outage Report Business Process then uses the services provided by the Register Outage Information Application Service which in turn writes the information to the Outage Ticket Data Object (process step t=6). The compiled outage tickets are now available in NIS ready to be used for creating the outage report. The Generation of Outage Report Application Service is then invoked to the NIS to generate the final Outage Report Data Object (process step t=7).
C. Results

To perform the accuracy analysis some data collection was made regarding the accuracy of the input Data Objects, and the execution accuracy of the Business Processes and Application Services. The estimates were made by the process participants during interviews and yielded the rough approximation that the Business Process execution accuracy, i.e. the probability that the Business Process performs according to specification, is around 99%, whereas the Application Services were thought to be accurate at all times, i.e. having a 100% correct execution time. The reason for not having 100% accuracy in the Business Processes had to do with the repetitious work often involving manually transcribing data from paper into a computer. The accuracy of the input data to the process, i.e. the Outage Tickets, was estimated to be 99% accurate on average. The error in the input data mainly had to do with erroneous estimations of the Outage Times by customers calling in to report their power outages.

The Bayesian network part of the concrete model and the estimates collected where then inserted into an identical model in GeNiie, a Bayesian network inference engine [26]. The Bayesian network in GeNie implemented the AND CPTs shown in Section 4.

The results from the computations in GeNie showed the gradual degradation of accuracy. Fig. 7 illustrates this degradation graphically by showing how the various representations of the Business Object Outage Ticket were affected. It can be clearly seen how the initial accuracy of 99% in the input Outage Ticket slowly deteriorates across the Business Process and lowers the accuracy of the Outage Report to a mere 95%. Referring to Fig. 7 it can be seen, that the accuracy is not influenced during the time steps t=5 and t=7, this due to the lack of influence from the inaccurate Business Processes. While data is only processed by applications its accuracy is kept intact leading to a situation where the human error has the largest impact.

Figure 7: The gradual deterioration of accuracy in the Outage Tickets.

This suggests that one way to improve process accuracy could be to integrate the Maintenance Management Systems with NIS, leading to a reduction of manual activities when generating the outage report. Notwithstanding the fact that the estimates on the accuracy of the processes and input data may be wrong, the case study nevertheless illustrates how the method proposed in this paper can be employed to quantitatively analyze data accuracy deterioration across a business process.
VI. CONCLUSIONS AND FUTURE WORKS

In this paper a Bayesian network based architecture analysis method is proposed for the assessment of data accuracy. The method caters for information and data transformation as a result of modification from software or due to business process. Use of Bayesian networks makes it possible to trace data provenance and observe data deterioration in a quantitative manner. The adoption of the ArchiMate language makes it possible to analyse the accuracy of enterprise system data and illustrate the impact of the accuracy of the business processes and software functions on this data.

Future works include making more use of the Dynamic Bayesian Network formalism, to collect more empirical data to validate the method, and to incorporate other data quality dimensions into the assessment framework.

VII. REFERENCES