COMPUTING UNCERTAIN KEY INDICATORS FROM UNCERTAIN DATA

(Completed paper)

Carlos Rodríguez, Florian Daniel, Fabio Casati
University of Trento, Italy
{crodriguez, daniel, casati}@disi.unitn.it

Cinzia Cappiello
Politecnico di Milano, Italy
cappiell@elet.polimi.it

Abstract: Key indicators, such as key performance indicators or key compliance indicators are at the heart of modern business intelligence applications. Key indicators are metrics, i.e., numbers, that help an organization to measure and assess how successful it is in reaching predefined goals (e.g., lowering process execution times or increasing compliance with regulations), and typically the people looking at them simply trust the values they see when taking decisions. However, it is important to recognize that in real business environments we cannot always rely on fully trusted or certain data, yet indicators are to be computed.
In this paper, we tackle the problem of computing uncertain indicators from uncertain data, we characterize the problem in a modern business scenario (combining techniques from uncertain and probabilistic data management), and we describe how we addressed and implemented the problem in a European research project.

Key Words: Key Indicators, Uncertain Indicators, Probabilistic Indicators, Uncertain/Probabilistic Data, Data Warehousing, Business Process Intelligence.

INTRODUCTION

Facilitated by the extensive use of Information Technology (IT) in today's companies, business environments have become highly dynamic and responsive. Especially the growing availability of *business data* that are accessible for analysis and interpretation has changed the way business people "read" the performance of their company: increasingly, they base their decisions on summaries, reports, and analyses coming from *Business Intelligence* (BI) applications. In order to gain competitive advantage over their competitors, BI applications allow them to get insight into the changes in the business environment, to rapidly react to changes, and to keep performance under control. With the advent of so-called process-aware information systems (such as Business Process Management systems), we are now also in the presence of large amounts of data regarding the execution of *business processes*, and, hence, business people also have the possibility to access not only business data (e.g., the amount of sales in a particular month) but also execution data (e.g., who executed a given activity in a business process and how long did it take him to complete the task). The analysis of such kind of business process execution data is the focus of so-called *Business Process Intelligence* (BPI) applications and of this paper.
One of the most important instruments used to report on the state of a company's business are *Key Indicators* (KIs), which are metrics that allow a company to measure its performance against specific objectives. Their value mainly lies in their simplicity: with one number they summarize a huge amount of data and, at the same time, intuitively describe a well-specified part of business reality. The use of alarm levels and colours further enhances their readability and (cognitive) accessibility. Typically, indicators like KPIs

(key performance indicators) measure the achievement of business objectives (e.g., the average revenue of a given department), but there are also indicators that rather focus on risk (key risk indicators), compliance with laws or regulations (key compliance indicators), and similar. In the last years, great attention has been paid to the automated computation of KIs over business and process execution data.
The advantages provided by BI and BPI applications and the computation of KIs are possible thanks to advanced technologies used to store large amounts of data reflecting the whole lifecycle of a company's business in a continuous form (typically, we talk about data warehouses). However, the speed at which data are generated, combined, and processed by means of various technologies, software tools, and human actors, the quantity of the available data, plus the fact that today's business scenarios are highly interlinked, i.e., companies do not act in an isolated fashion from an IT point of view (e.g., companies share parts of their business processes with strategic partners or they outsource part of their IT infrastructure and business processes to specialized companies), inevitably leads to data with quality problems: logged data may contain errors or noise, incomplete or inconsistent data flows, etc. For example, if the bus or the logging system suffer from bad configuration, overload, performance problems, or downtimes we might not be able to log all the important messages flowing through an enterprise service bus (e.g., to compute indicators).
Computing KIs from data that are characterized by low quality (i.e., uncertain data) demands for novel and sophisticated algorithms, able to take into account the quality of the data. As a matter of fact, KIs themselves will be uncertain. Not taking into account the uncertainty that characterizes an indicator during its computation could give the people looking at the final value of the indicator a wrong perception of the actual performance of the business and might cause them to take wrong decisions, which eventually could negatively affect their business.
In many situations, the huge amount of potentially uncertain data combined with the need for continuously computing and re-computing KIs, makes the effort of running complex correction procedures (if any) prohibitive and impracticable. Yet, business people need to keep computing KIs in order to keep track of business performance while taking into consideration that indicators are computed on uncertain data. That is, decision makers *must* be aware of the quality of their indicators at the time of taking decisions concerning their business.

Contributions. Computing expressive and meaningful indicators from uncertain data is a challenging and tricky endeavour. In this paper, we approach the problem from both a theoretical and a practical perspective. Specifically, we:

- Characterize the problem of computing key indicators in distributed business environments as a *data quality problem* that is specifically related to uncertain/probabilistic data;
- Propose an approach to compute *values for key indicators* from uncertain/probabilistic data based on techniques from uncertain data management;
- Introduce the concept of *uncertain/probabilistic key indicator* and quantify uncertainties/probabilities starting from the data used in the computation of an indicator;
- Provide a concrete *data warehouse model* for the data needed to compute key indicators in the context of a European project, along with the corresponding extensions to deal with uncertainty;
- Hint at how to *visualize indicators* in order to convey to users the likelihood that an indicator takes a particular value considering the uncertainty in the input data.

Structure of this paper. In the next section we introduce a real-life reference scenario that will accompany us throughout the rest of the paper. Then, we conceptualize the described scenario and its business context and formally define the problem addressed in this paper. Based on this formalization, we describe the theoretical foundation for the computation of key indicators from uncertain data and, next, show how we compute uncertain indicators in practice. Finally, we describe our implementation of the proposed solution in the context of a European project, discuss related works, and draw our conclusions.

REFERENCE SCENARIO

Let's consider a Network Information Center (NIC) that provides Internet domain name registration for a Top-Level Domain (TLD) in the Domain Name System (DNS) via the Web. The NIC is in charge of administrating the (fictitious) domain .sci, which is limited to organizations, offices and programs whose main interest resides in any kind of science. For example, the organization abc that does research in nanoscience could register the domain name abc.sci to provide name resolution for Internet resources, such as mailing services or a web site (e.g., http://www.abc.sci).

For our scenario, we consider two business processes used by the company for administrating the TLD. The first business process consists in the delegation of domain names as shown in Figure 1. The model in this figure is a simple flowchart with swim lanes that show the distribution of tasks among stakeholders. The process can be divided into four parts: (i) insertion of the request (client), (ii) verification of the request (NIC), (iii) payment for the domain name (client/bank), and (iv) activation of the domain name (NIC). The management of the payment is performed in conjunction with a bank that interacts with the NIC through web services, i.e., the NIC and the bank share part of their business processes. When the request for delegation is approved, the client proceeds with the payment via one of the channels offered by the bank. Upon reception of the payment, the bank notifies the NIC, which causes the NIC to automatically activate the domain name requested by the client.

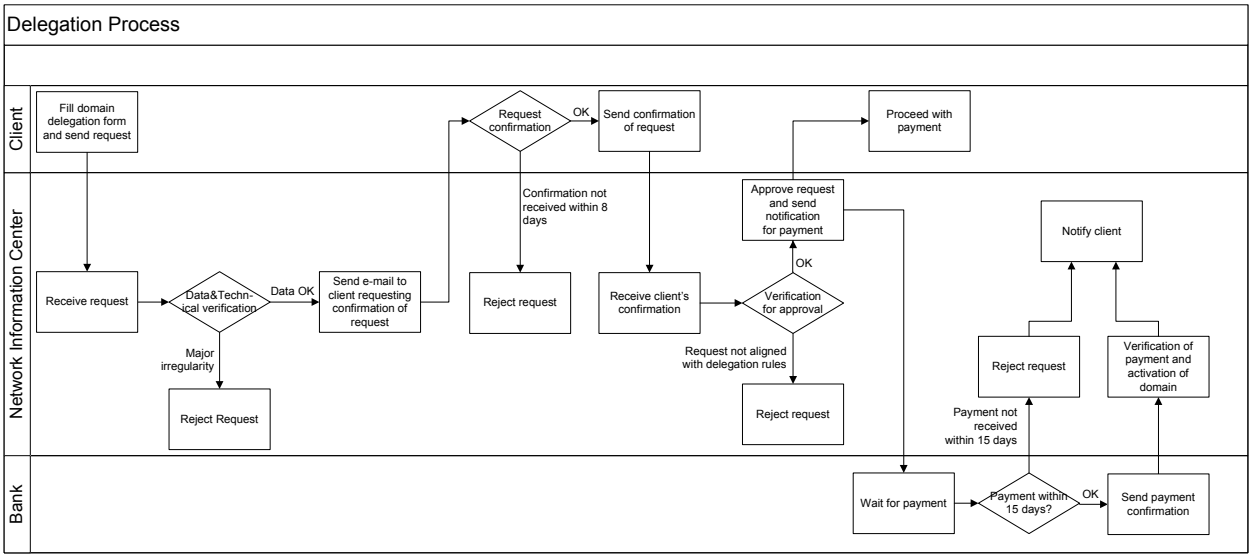


Figure 1 Process for the delegation of domain names. The process is operated by the NIC and the bank; clients are involved through the NIC's web application.

The second business process (see Figure 2) is the procedure for modifying information related to the delegation of domains, such as data of the owner and technical details. The process is part of the customer support that the NIC has outsourced to an external company specialized in user support and providing services like a call center and a support web application. Both the call center and the web application are fully detached from the operational system of the NIC and managed by the Customer Support center. Yet, they provide a reduced set of views and operations on users' data.

The NIC is now interested in studying the performance of its business processes, in order to monitor and improve quality of service. For instance, the NIC is interested in computing the following key indicators:

- *TBRP (time between request and payment)*: With this indicator, the NIC wants to capture how long in average it takes to the client to proceed with the payment for the domain name.
- *SRSE (number of subsequent requests by the same entity that block a specific domain)*: This is an important indicator, since, once a request is inserted, no one else can request that same domain

name, unless the request is cancelled or expired due to missing payment. This indicator helps the NIC to detect when somebody tries to keep a domain name blocked without willing to pay for it, e.g., to prevent others to acquire it.

– *TBAS (time between the activation of a domain name and the first support request)*: This indicator provides an idea of how long it takes in average a user to contact the first time the support center upon the successful registration of a domain name. This allows the NIC, for instance, to assess the quality of the documentation provided in the phase of registration and to estimate the cost of the support service.

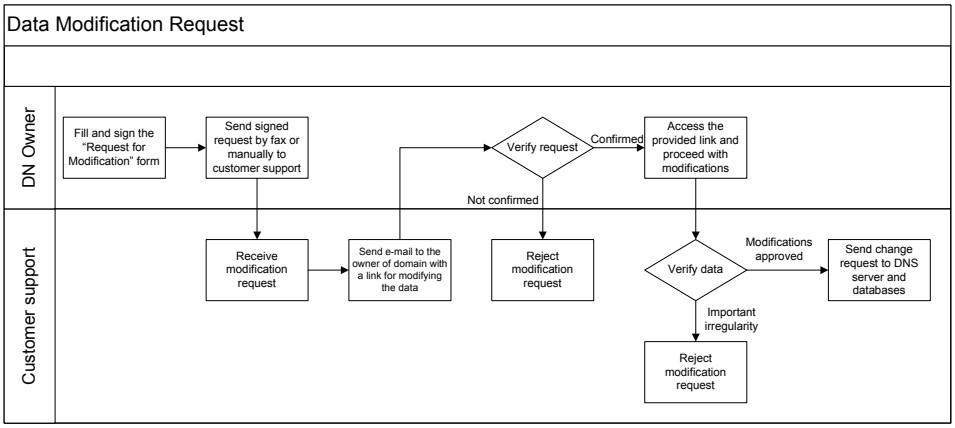


Figure 2 The data modification request process.

For the computation of the above indicators, the NIC instruments its Delegation process, which is mainly based on web services and the client's web application, so that the process generates the necessary information in form of events (we assume each activity in the process models may generate respective events). For instance, the NIC generates a ReceiveRequest event and has already agreed with the bank on the generation of a corresponding PaymentConfirmation event (along with a respective service level agreement ruling the quality of service of the event delivery), which are at the basis of the TBRP indicator. Similarly, the NIC provides for the events necessary to compute the SRSE indicator, which only involves events under the control of the NIC and, therefore, do not require any negotiation or agreement with either the bank or the support center. The computation of the TBAS indicator, instead, is trickier: the time of the activation of a domain name is easy to track (the NIC has control of that), but for the time of the first support request the NIC could only obtain a best effort commitment by the support center, which is already a good achievement. Indeed, in general the support center could also not have been willing at all to provide that information, able to do so, able to do so reliantly, etc. After some days, the NIC looks at the events it could log and sees that, besides the availability or not of events, there are even other problems with the data in the log: events are not always logged correctly; some events seem to be wrong (but the NIC is not fully sure); some data values are not precisely defined, and similar. In short, the event log the NIC would like to use to compute its indicators may present data quality issues, yet the NIC needs to compute its indicators anyway.

UNCERTAINTY AND PROBABILITY IN BUSINESS ENVIRONMENTS

The above scenario demands for the computation of three key indicators related to the two business processes run by the cooperating parties. In this paper, we do not want to pose any restriction on how business processes are executed (e.g., manually vs. semi-automatically vs. automatically). However, in order to be able to automatically compute indicators, we assume that the data necessary for the computation of the

indicators are available in the form of *events* that are generated by the cooperating partner's IT systems and that provide (partial) visibility into the execution of the business processes. We say that the business processes are *instrumented* in order to generate events. For instance, an ActivityStart event could be automatically generated by a business process engine, or a Reject event could be derived from an email sent by a physical person to an archiving system. In order to be able to compute meaningful indicators, events must carry some piece of business data (e.g., the name of the person approving or rejecting a domain name registration).

These assumptions and minimum requirements are realistic. Especially in presence of companies that cooperate over the Web and typically base their cooperation on web services and the service-oriented architecture, the generation of such kind of events is no big issue. Also, as we want to compute key indicators periodically for reporting purposes (e.g., each night or once a week) we assume that the events we are interested in are logged in a central event log that can be periodically inspected.

We represent a generic event $\bar{e}$ as a tuple $\bar{e} =< ID, procID, type, ts, src, dest, \bar{d}_1, \dots, \bar{d}_n >$ (note that we use the bar over symbols to indicate that they represent *certain* data; we will use symbols without the bar when instead they represent *uncertain* data), where ID is a unique identifier of the event, $procID$ is the unique identifier of both process instance and process model, $type$ specifies which kind of event we have (e.g., ActivityStart or Reject), ts is the timestamp in which the event has been generated, src and $dest$ are the source and the destination (if any) of the event (e.g., the company or business process instance that is the origin of the event), $\bar{d}_1, \dots, \bar{d}_n$ are the parameters carrying possible business data values (they are the actual body or payload of the event, their number might vary from event type to event type). More specifically, each $\bar{d}_i$ is characterized as follows: $\bar{d}_i =< partype, name, value >$ with $partype$ being the type of the parameter (e.g., integer, enumeration of string values, etc.), $name$ being the name of the parameter, $value$ being the concrete value assigned to the parameter.

Events are generated during the execution of business processes, and each business process in execution (i.e., a process instance) typically generates a multitude of events during its execution. As the only information we have about the executed process instances is the set of events generated by them, we represent a process instance as a *trace* of chronologically ordered events $\bar{t} = \bar{e}_1, \dots, \bar{e}_n$. For instance, the above Delegation process could produce an event trace as follows: $t =< 0, DelegationProcess3, ProcessStart, 20090509144209, NIC, NIC, Customer, RegistrationReq >, \dots, <$
$27, DelegationProcess3, Notification, 20090604050648, 67, NIC, Client, Msg >$
that tells us that there has been an instance of the Delegation process ($procID = DelegationProcess3$), started on May 9, 2009, which sent a notification with content Msg to the client on June 4, 2009. Finally, we represent the event log as a set $E = \{\bar{t}_i\}$.

Since now a KI summarizes execution data of multiple process instances (e.g., all the executions of the Delegation process of the last week), a *key indicator* is a function that is computed over a set of process instances, i.e., a set of events. More precisely, a KI is a function $\overline{KI} = \overline{KI}(\{\bar{t}_i\})$ over a set of event traces that assigns to each subset of $\{\bar{t}_i\}$ a real number (the indicator value), i.e., $\overline{KI}: \mathcal{P}(\{\bar{t}_i\}) \rightarrow \mathbb{R}$.

Data quality in event logs

The problem in practice is that the event log E contains data (or not) that not always are fully aligned with the real world, i.e., with the concretely executed business processes. Inspired by [16], in this paper we distinguish four situations that are characteristic of the described business scenario. In this paper, we address the first three scenarios; we do not explicitly treat the fourth, as it rather represents a design time issue that is out of the scope of this paper:

1. ***Meaningless state*** *= there is an event in the event log, but we are not sure the corresponding real-world event indeed happened*: For instance, in Figure 3a there is an ActivityStart event in the log (row 234) but, as hinted at by the dotted tail of the arrow, we lack the corresponding counterpart in the real world (e.g., an employee sent an email that he started an activity but actually never per-

formed the corresponding task). As a result, there might be events in the log that are uncertain.

2. ***Uncertain data*** *= there is an event in the event log, but we are not sure about the exact data values carried inside the body of the event or, simply, whether values are correct* (Figure 3b): Rows 235 and 236 derive from the same real-world event (the notification of the payment details to the client and the bank), yet we are not sure whether the notification has been sent to Paul or to John. Row 240, instead, presents an uncertainty regarding the exact time in which the event was generated.
3. ***Incomplete representation*** *= we think that a real-world event happened, but there is no corresponding event in the log*: As represented by the empty row in Figure 3c, there might be actions in the real world that should have been logged but that lack a corresponding event in the log. Such a lack could for instance be due to system failures or downtimes, network problems, people forgetting to send an email, or the like. In some cases, however, we might be able to derive that a real-world action must have happened from the business context that can be reconstructed from the log. For instance, if the event log contains a ProcessEnd event, very likely there also must have been a corresponding ProcessStart event.
4. ***Lack of representation*** *= we are not able to log all the events that are necessary to compute an indicator*: If a company, for instance, decides to outsource part of its business, it might lose visibility into the details of how an outsourcing partner actually performs its business, practically losing the events associated with the outsourced part of the business process. As a consequence, the company might no longer be able to compute an indicator, and a re-design of the indicator's computation logic is necessary – if possible; otherwise, the computation of the indicator can simply not be performed any longer.

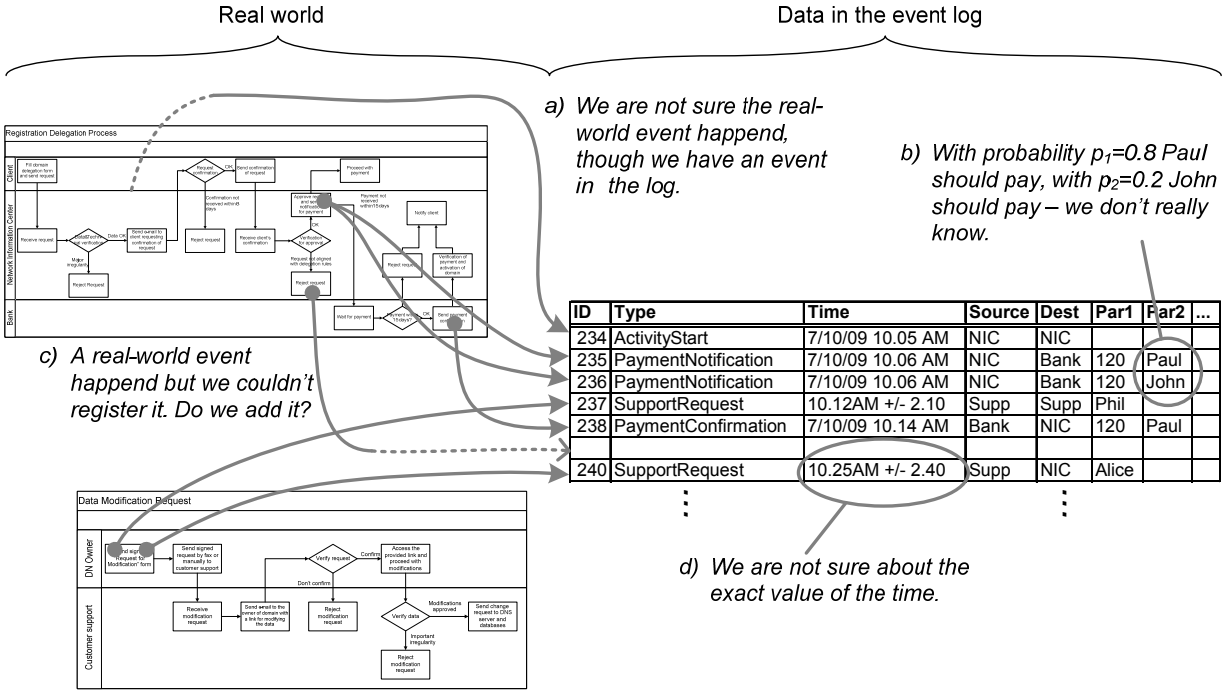


Figure 3 Discrepancy between the real world and the data we might have in the event log.

This casuistry shows that in realistic settings it is generally not a good idea to think that indicators can be computed straightaway from the data that can be found in an event log. The log might be incomplete (missing events), it might contain noise (wrong events), it might contain uncertainties regarding the correctness of tuples, or it might contain uncertainties regarding the exact value of data cells. Note that the computation of the degrees of uncertainty in the input data is outside of the scope of this paper.

Expressing uncertainties and probabilities

In order to be able to compute meaningful indicators from a real event log, we must be able to represent the above problems in the data we use to compute the indicators. The metrics that we use in this paper to keep track of data quality depend on the object of the quality problem; specifically, we associate: (i) *reputation* to events in order to express the likelihood that an event in the log corresponds to an event in the real world (covering the cases of meaningless states and incomplete representations); (ii) *probabilities* to data values in order to express alternatives or levels of confidence for discrete values (covering part of the data uncertainty case); and (iii) *confidence intervals* to data values in order to express doubts about the exact value of continuous, numeric values (covering the other part of the data uncertainty case). Taking into account reputation, probabilities, and confidence intervals demands for an extension of our event formalization.

So far, we defined an event as a tuple $\bar{e} =< ID, procID, type, ts, src, dest, \bar{d}_1, \dots, \bar{d}_n >$, in which both the event and its parameter values were fully trusted. In order to associate a reputation value with each event and probabilities/confidence intervals with data values, we define an *uncertain event* as a tuple $e =< ID, procID, type, ts, src, dest, d_1, \dots, d_n, rep >$, where $ID, procID, type, ts, src, dest$ are as defined previously for $\bar{e}$ (note that, rep is the reputation associated to the event, and $d_1, \dots, d_n$ are the business data parameter to which we associate probabilities or uncertainties, as described next. Note that in presence of uncertain date we now omit the bar on top of the symbols.

Modelling reputation. The association of a *reputation level to an event* can, for instance, be done by combing an objective and a subjective measure, i.e., an analysis of the data in the event log and the confidence we have in the correct operation by cooperating partners (i.e., their reputation). The *objective* measure can be derived by looking at how many meaningless state cases and incomplete representation cases we have in the log. The *subjective* measure typically stems from the reputation levels we associate to business partners; Figure 4 conceptualizes our cooperative business scenario and highlights reputation and visibility issues. First of all, the company (e.g., the NIC) runs own processes in-house; the probability that events are correctly registered in own processes is typically high (e.g., $p(e_1) = 0.99$). Next, a company might cooperate with an independent partner by means of a shared business process in which both partners participate and of which both have full visibility (e.g., the NIC cooperates with the bank); as the responsibility of the common process is shared among the two parties, the confidence in correct events is typically lower than the one we have in own events (e.g., $p(e_2) = 0.85$). Finally, if part of a business process is outsourced (e.g., to the Customer Support Center), the company has only a very limited visibility into the outsourced part of the business process and, hence, confidence in events might be lower again (e.g., $p(e_5) = 0.70$). For presentation purposes, we manually assign "reasonable" values to the five types of events; in practice, such values would be derived by a suitable reputation assessment system from historical data about the interacting parties.

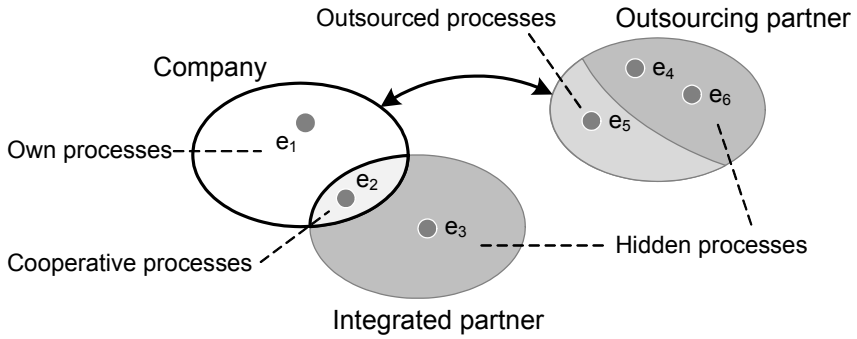


Figure 4 Reputation and visibility into business processes of cooperating partners.

We assume that for each event e, we know its provenance, i.e., the company $c \in C$ that generated the event, where C is the set of companies involved in the business processes over which we want to compute our indicators. The reputation level rep of e can then be seen as a function $rep: C \rightarrow [0;1]$, where $rep(c)$ represents the reputation of company c. Associating reputation levels to events therefore allows us to deal with meaningless states: the level of reputation expresses the likelihood that the events provided by a business partner also have appropriate real-world counterparts. But we can also deal with incomplete representations: if we decide to add an event to the event log because we believe a real-world event is not represented in the log, we can add the presumably missing event to the log, associate it to its respective company, and assign it a low probability (to express that we are anyway not fully sure of our decision).

Modelling uncertainty over data values. As hinted at above, we use two instruments to express uncertainty over data values: confidence intervals and possible worlds. We use *confidence intervals* to refer to measurements performed over business matters, such as the revenue in a time period, for which we are not sure of their exact value. More precisely, here we focus on event parameters expressed as *real numbers* that come with an error or confidence represented as a confidence interval or standard deviation σ. This way, an uncertain value is represented as $value \pm \sigma$, which means that the true value lies somewhere in between $[value - \sigma\,; value + \sigma]$. Instead, we talk about *possible worlds* in order to denote all the possible values (together with their respective probabilities) that a given data field or indicator might assume. For example, in probabilistic databases, a possible world refers to a particular database instantiation that is aligned to a predefined schema. Here, each instantiation is associated with a probability and the sum of the probabilities of all the possible instantiations must be equal to 1 [8]. In our context, we use the possible world model as a base to represent the various values a measurement can take, together with their corresponding probabilities. More precisely, we opt to use the possible world representation to describe the probabilities of occurrence of discrete, countable values. For example, if we have an event parameter *name* for which we are not sure about its true value (e.g., we are not sure if its value is $smit$, $shmit$, or $smith$), the possible worlds for this parameter could be represented by a set of pairs $< value, probability >$ as $\{< smit, 0.2 >, < shmit, 0.2 >, < smith, 0.60 >\}$.

The association of confidence interval with a data value can be done by the source of the event if it knows about the probability distribution of the value to be transmitted (e.g., in the case of the timing information logged by a logging system in a distributed environment), or it can be computed from the log by looking at the probability distribution of the value that derives from past values. The association of a probability can be done directly by the source of the event (a company), which might communicate its doubt regarding the data value, or it can be computed from the log, for example, by means of entity resolution algorithms [4] that are typically able to identify similar tuples and to associate probabilities to the identified options.

In order to characterize confidence intervals/probabilities for the parameters $d_1, \ldots, d_n$, we introduce the concept of *uncertain parameter* as $d_i =< partype, name, (v_{conf} \mid v_{prob}) >$ with $partype$ being the type of the parameter, $name$ being the name of the parameter, $v_{conf} =< value, \sigma >$ being the $value$ assigned to the parameter and its standard deviation σ, and $v_{prob} = \{< value_j, prob_j >\}$, being the set of *possible worlds* (in terms of possible values and probabilities) deriving from the probabilistic nature of the data value ($\sum prob_j = 1$,). In order to express confidence intervals for numeric values, we therefore use the values' standard deviation $\sigma \in \mathbb{R}$ (we do not take into account the whole probability distribution of the value), while for each possible world we use a probability $p \in [0;1]$. We assume that from the $name$ of a parameter we can uniquely tell whether the parameter comes with probabilities or a confidence interval.

It is worth noting that if we have both the value and the probability distribution for an uncertain parameter, we could actually compute its probability. This would allow us to assign a probability p to the parameter instead of an uncertainty σ. However, in order to keep the number of possible worlds as small as possible, we express uncertainty over data values as $value \pm \sigma$ whenever possible. This decision further allows us to compute uncertainty levels for KIs independently of the probability distributions of the in-

volved parameters, as shown in the next section.

The problem now is (i) *how to compute an indicator* $KI(\{t_i\})$ *over a set of traces* $t = e_1, \dots, e_n$ *and that is uncertain itself* and (ii) *how to visualize and report on indicators that are characterized by uncertainty*.

COMPUTING UNCERTAIN KEY INDICATORS

In the previous section, we have seen that we characterize events by means of an uncertainty at the *event level* (its reputation) and an uncertainty at the *data level* (the confidence and the possible worlds for the data values). The former is strictly related to the reputation of the company involved in the cooperative process and indicates the probability that a logged event indeed corresponds to an event in the real world. In the computation of a KI, we can use this information to weight the data in the events according to their reputation, so as to give more weight to data with high reputation and less weight to data with low reputation. The uncertainty at the data level, instead, carries over from the data in input to the final value of the indicator in form of either a confidence level associated with the indicator value or a set of possible worlds describing all the possible combination of possible worlds we have in the probabilistic data values of the events used in the computation of a KI.

Therefore, given a set of traces $\{t_i\} = \{e_{i1}, \dots, e_{iJ}\} = \{[e_{ij}]\}$ with $1 \leq i \leq I$, where I is the number of traces in the log, and $1 \leq j \leq J$, where J is the number of events inside a trace i, $e_{ij} =< ID_{ij}, procID_{ij}, type_{ij}, ts_{ij}, src_{ij}, dest_{ij}, [d_{ijk}], rep_{ij} >$ with $1 \leq k \leq K$, where K is the number of parameters inside each event, and $d_{ijk} =< partype_{ijk}, name_{ijk}, (< value_{ijk}, \sigma_{ijk} > | \{< value_{ijkl}, prob_{ijkl} >\}) >$ with $1 \leq l \leq L$, where L is the number of possible worlds characterizing the value of a probabilistic parameter, an *uncertain KI* can be expressed as follows:

$$KI(\{t_i\}) =< \{< f_m, \sigma_m, prob_m >\}, conf >$$

where $1 \leq m \leq M$, being M the number of possible worlds that characterize the indicator, f_m is the indicator value, σ_m is the standard deviation associated to the indicator value, $prob_m$ is the probability that characterizes each possible world, and $conf$ is the overall confidence that we can derive for the indicator from the reputations of the events used in f. The definition of uncertain KI, in general, contains both a confidence interval σ_m for the value, and a probability $prob_m$ for each possible world, as the function f might be computed over events with both, parameter values associated with confidence intervals, and parameter values associated with probabilities (possible worlds).

The number of possible worlds M derives from the combination of the possible worlds inside each event. Specifically, $M = \prod_n L_{ijn}$ with L_{ijn} being the number of possible worlds of each probabilistic data parameter d_{ijk} used in the computation of KI; hence, $0 \leq n \leq K$. An uncertain KI is therefore characterized by a set of possible worlds, where each world can be characterized as follows:

$$f_m = f(\{[< \{(value_{ijk} | value_{ijkl})\}, rep_{ij} >]\})$$

That is, the indicator value of an individual possible world can be computed by means of a function f_m over the data values $(value_{ijk} | value_{ijkl})$ of the parameters d_{ijk} over which the KI is defined; the computation might also take into account the reputation rep_{ij} of the events involved in the computation (e.g., to weight data according to reputation). We use $value_{ijk}$ in case d_{ijk} contains an uncertain data value and $value_{ijkl}$ in case we use the l-th possible value of a probabilistic d_{ijk}. In practice, f_m is given by the designer of the KI. In our reference scenario, it is the NIC who defines the KIs it is interested in. Indeed, we have seen that the NIC wants to compute the three indicators TBRP, SRSE, and TBAS.

$$\sigma_m = \sigma_m^f = g(\{[< \{\sigma_{ijk}\}, rep_{ij} >]\})$$

The standard deviation σ_m can be computed by means of a function g from the standard deviations σ_{ijk} associated with the values $value_{ijk}$; g might take into account the reputation of events. We use the notation σ_{ijk} to uniquely identify the standard deviation of each data value; yet, note that data values of a same parameter in different traces or events are characterized by the same standard deviation. We use the notation $\sigma_m = \sigma_m^f$ to highlight that the computation of the standard deviation (and g) depends on how the data values (i.e., the statistical variables) are used in the formula f_m. We will see this property in the example we discuss next.

$$prob_m = h(\{[< prob_{ijkl}, rep_{ij} >]\})$$

We compute $prob_m$ by means of a function h that is specified over all the probabilities $prob_{ijkl}$ associated with the choices of possible data values that characterize the particular set of possible world of the indicator.

$$conf = conf(\{rep_{ij}\})$$

Finally, we compute the overall confidence $conf$ of the indicator as a function of the reputations associated with the data values considered by f_m. The exact value of $conf$ can be computed by using different aggregation functions (e.g., the minimum of all reputations, the average of them, or similar); in this paper we adopt the minimum, though other functions could be used as well.

In order to better explain the concepts introduced in this section, let us consider the case in which the NIC is interested in monitoring the *TBAS* indicator that calculates the average time between (i) e_{i1} = registration of a new domain name and (ii) e_{i2} = first time that the customer contacts the customer support center. Let's assume $e_{i1}[TReg] =< time, TReg, < TReg.value, TReg.\sigma >>$ and $e_{i2}[TSup] =< time, TSup, < TSup.value, TSup.\sigma >>$, that is, both data values come with a confidence interval. Formally, the $TBAS$ indicator is then characterized by the value obtained by summing the time intervals calculated in each trace and weighed based on the companies' reputation:

$$TBAS(\{t_i\})[f] = \frac{\sum_{i=1}^{I}(e_{i2}[TSup.value] - e_{i1}[TReg.value])}{I}$$

In this case, the indicator $TBAS(\{t_i\})$ is also characterized by an aggregate σ that can be obtained by applying the rules used for the computation of error propagation from uncertain values to functions computed on these values, such as the one presented below [15]. In this formula, f is the function we want to compute, $TSup.value = TS$ and $TReg.value = TR$ are statistical variables, and, $\sigma(TS)$ and $\sigma(TR)$ are the standard deviations of TS and TR, respectively:

$$f = \frac{\sum_{i=1}^{I}(TS - TR)}{I} = \frac{I(TS - TR)}{I} = TS - TR$$

$$\sigma(f) = \sqrt{[\sigma(TS)]^2 + [\sigma(TR)]^2 \pm 2COV(TS, TR)}$$

In our example, we can assume that the time of the support is independent of the time of the registration.

Therefore, the events that are completely independent, and the covariance between the two different values $cov(e_{ij}[value_{ijk}], e_k[value_{ijk}])$ is equal to 0. This simplification, together with the assumption that, on one hand, all $TReg$ values are associated with roughly the same standard deviation throughout all traces and, on the other hand, all $Tsup$ are associated with roughly the same standard deviation as well, allows us to define the standard deviation for the $TBAS$ indicator as:

$$TBAS(\{t_i\})[\sigma] = \sqrt{e_{i1}[\,TReg.\sigma]^2 + e_{i2}[\,TSup.\sigma]^2}$$

Finally, each parameter discussed above (i.e., f, σ) that characterize the indicator should be weighted by the person looking at the indicator with the confidence that we associated to the overall indicator (in our convention, we use the minimum; other conventions could be used as well):

$$TBAS(\{t_i\})[conf] = Min(e_{i1}[rep]; e_{i2}[rep])$$

In conclusion, our uncertain indicator $TBAS$ is given by:

$$TBAS(\{t_i\}) =<< TBAS(\{t_i\})[f], TBAS(\{t_i\})[\sigma] >, TBAS(\{t_i\})[conf] >$$

If instead of having uncertain data values, the two parameters used in the computation of $TBAS$ were of probabilistic nature, e.g., $e_{i1}[TReg]$ having three different alternatives and $e_{i2}[TSup]$ having two different alternatives, we would be in presence of six possible worlds for the final indicator. We would therefore need to apply the above procedure to each of the possible worlds of the indicator, and we would need to compute the combined probability for each of the possible worlds as follows (note that the two events e_{i1} and e_{i2} are independent):

$$TBAS(\{t_i\})[prob] = \{e_{i1}[prob_l] * e_{i2}[prob_l]\}$$

with l being the index of the possible worlds of the indicator. The final indicator would therefore look like the following:

$$TBAS(\{t_i\}) =< \{< TBAS(\{t_i\})[f]_l, TBAS(\{t_i\})[\sigma]_l, e_{i1}[prob_l] * e_{i2}[prob_l] >\}, TBAS(\{t_i\})[conf] >$$

IMPLEMENTATION: KEY INDICATORS IN PRACTICE

The described approach to the computation of uncertain indicators has been developed in the context of a European FP7 research project (MASTER – Managing Assurance, Security and Trust for sERvices[1]), which focuses on methodologies and infrastructures to manage security and compliance of service-based business processes. We in particular focus on the assessment of and reporting on compliance, starting from a log of events generated by the MASTER infrastructure, and the computation of uncertain key compliance indicators is one of our main contributions, along with an analysis of correlations among indicators and process model discovery.

Figure 5 shows a simplified architecture of the MASTER infrastructure; specifically, we focus on the diagnostic infrastructure with its data warehouse and analysis algorithms. The main input to the infrastructure is the (uncertain) *Event log*, which contains events generated by the operational system (a service-oriented architecture). During ETL (Extract-Transform-Load), events are extracted from the log and stored in *Staging area* for transformation. Of the overall transformation process, we highlight the *Process instance reconstructor* and the *Key indicator tables creator*, which reconstruct from the events in the log

[1] For the details, the reader is referred to http://www.master-fp7.eu

which process instances have been executed and create auxiliary tables for indicator computation, respectively. Then, we load (*Warehouse loader*) the data into the *Data warehouse*, upon which we then run our analysis algorithms to (i) compute uncertain indicators (*Indicator calculator*), (ii) correlate indicators (*Indicator correlation analyzer*), (iii) analyze compliance of processes with regulations and laws (*Compliance analyzer*), and (iv) discover process models from the log (*Process model discoverer*). All analysis results are stored back into the warehouse and rendered to the user (compliance expert or business analyst) via a *Reporting dashboard*. In this paper, we focus on the computation of the indicators, which are at the heart of the *Reporting dashboard*; the respective components are highlighted in the architecture.

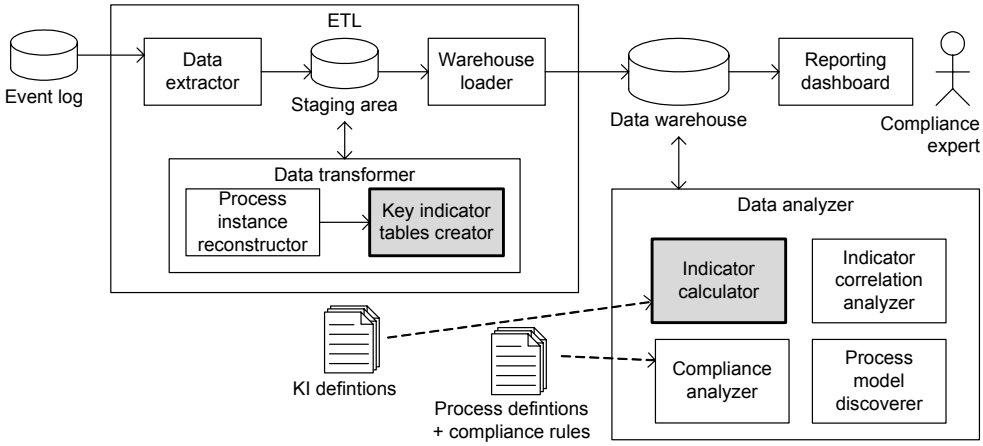


Figure 5 Functional architecture of the diagnostic infrastructure in the MASTER project

In Figure 6 we hint at the conceptual model of the data warehouse underlying the analysis algorithms, yet, for lack of space, we do not describe its details here. For the sake of this paper, it suffices to know that we store all the events in the warehouse, along with the data quality metadata associated to them and to the individual data parameters inside the events (reputation levels, uncertainties, probabilities). We also keep track if an event is a deduced event that we added to the log during ETL to solve incomplete representation cases. Therefore, the warehouse contains a complete historical view over the performance of a company.

The figure also contains the so-called *Key indicator tables* supporting the computation of KIs, in that they contain in a concise fashion, process instance by process instance (trace by trace), all the data values and their confidence levels and probabilities that we need to compute the indicators specified in the *KI definitions* document shown in Figure 5. The tables we highlight in Figure 6 are the ones we need to compute the TBAS indicator of our NIC: the RegDel table contains all the parameters regarding the Delegation process, and the DataMod table contains all the parameter regarding the Data Modification Request process. We only show the parameters necessary for the computation of the TBAS indicator: TReg is the registration time of the domain name, TRdev is the standard deviation, TRrep is the reputation, and ClientID is the identifier of the client; the parameters in DataMod are defined analogously. These tables are the output of the *Key indicator tables creator* in Figure 5. Next, we explain the logic of the *Indicator calculator*, that is, we show how we concretely compute uncertain indicators from the data warehouse.

In order to compute the TBAS indicator from the auxiliary tables in Figure 6, we translate its mathematical formula into SQL statements that we can issue to the data warehouse. For example, the following statements compute TBAS for all customers who successfully registered a domain name in May 2009 (note that we TBAS does not contain probabilistic parameters, so we only compute its value and confidence interval):

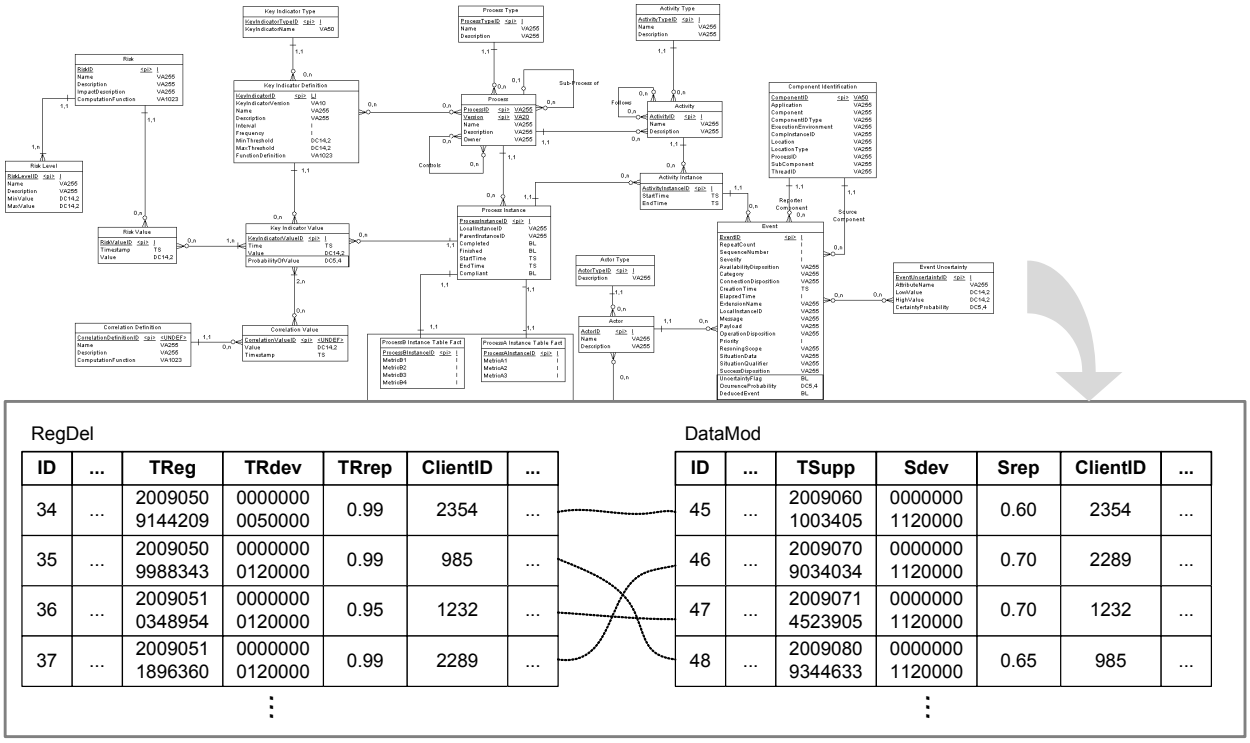
```
IntervalSum = select sum(TSupp-TReg)
              from Rendell join DataMod on RegDel.ClientID=DataMod.ClientID
              where TReg>=20090501000000 and TReg<=20090531999999;
```

```
RegDelCount = select count(RegDel.ClientID)
        from RegDel join DataMod on RegDel.ClientID=DataMod.ClientID
        where TReg>=20090501000000 and TReg<=20090531999999;

TBAS_value = IntervalSum/RegDelCount;

TBAS_sigma = select sqrt(sum(power(TRrep,2) + power(Srep,2)))
         from RegDel join DataMod on RegDel.ClientID=DataMod.ClientID
         where TReg>=20090501000000 and TReg<=20090531999999;

TBAS_conf = select min(case when TRrep<=SRep then TRrep else SRep end)
          from RegDel join DataMod on RegDel.ClientID=DataMod.ClientID
          where TReg>=20090501000000 and TReg<=20090531999999;
```



RegDel

ID	...	TReg	TRdev	TRrep	ClientID	...
34	...	2009050 9144209	0000000 0050000	0.99	2354	...
35	...	2009050 9988343	0000000 0120000	0.99	985	...
36	...	2009051 0348954	0000000 0120000	0.95	1232	...
37	...	2009051 1896360	0000000 0120000	0.99	2289	...
⋮						

DataMod

ID	...	TSupp	Sdev	Srep	ClientID	...
45	...	2009060 1003405	0000000 1120000	0.60	2354	...
46	...	2009070 9034034	0000000 1120000	0.70	2289	...
47	...	2009071 4523905	0000000 1120000	0.70	1232	...
48	...	2009080 9344633	0000000 1120000	0.65	985	...
⋮						

Figure 6 Conceptual model of data warehouse with uncertain data, plus – in the foreground – the key indicator tables supporting the computation of the indicators.

The final indicator is therefore given by: $TBAS =< TBAS\_value, TBAS\_sigma, TBAS\_conf >$.
Although not shown in this paper, it is important to observe that the presence of probabilistic parameters in the computation of an indicator can be handled by suitably applying the group by SQL statement to the columns that contain multiple possible values for a given parameter. In this way, the output of the computation is no longer a single value, but a set of tuples describing all the legal alternatives for the indicator in terms of value and confidence.
Finally, as already discussed in [6], it is very important to convey the uncertainty that characterizes the indicator to the user of the *Reporting dashboard*, in order to create the necessary awareness of the data quality problem underlying the computation of the indicator and to enable better informed decisions. We are still working on this aspect, but we propose to provide first a very high-level view of the indicator through an intuitive, graphical visualization of all the indicators (and their alternatives) in the system and to explicitly mark those indicators that are uncertain. If a user wants to inspect the nature of the uncertainty, we will support a drill-down mechanism allowing the user to explore, for instance, the indicators' alternatives, their probability distribution, and the confidence we have in the indicator.

RELATED WORK

Recently, there has been lots of interest in databases specifically designed to manage uncertain data [1][3][5][14]. In this case, data are coupled with a probability value indicating the degree of confidence to the accuracy of the data. These probabilities are then taken into account by the database management system when processing the data to produce answers to user queries. Most of the contributions deal with simple queries while only a few deals with the aggregation of uncertain data to produce the results of queries in which the aggregation functions (e.g., sum, count, avg) are used [9][13]. Anyway, the proposed systems however do not deal with the problems of deriving probabilities in more complex cases, such as when computing reports, and of extracting uncertain data from not trustable sources Furthermore, past contributions relate the value uncertainty only to the value correctness (i.e., accuracy) and do not consider the case in which meaningless and incomplete representation affect the databases.

In our case, we need to reason about data uncertainty caused by all the possible poor quality problems that could affect objects of different granularities (i.e., values, events), in order to characterize the reliability of the reports within their indicators. In an organization, KIs represent the links between available sources of performance data and organizational objectives [2]. KI measurement issues have been largely analyzed in the literature since sometimes decision support systems are not efficient as expected since they are based on erroneous KI values calculated on not reliable performance data. In fact, practically, any input data have uncertainty that can be caused by different issues such as inaccuracy of measuring and inaccuracy of rounding-up, scale restrictions, impossibility of measuring or definition of values with needed precision, hidden semantic uncertainty of qualitative data [11]. The number of data sources required to support the most common KIs measurement is large and thus the uncertainty issues cannot be neglected [10].

Previous work (e.g., [11]) focus on the evaluation of the KI measure starting from the assumption that the confidence of the obtained value depends on the assessment process. Here, the validity of the KI is defined as a property of a KI that makes it suitable as a basis for performance assessment. The generic attributes of valid performance measurements are straightforward: relevance, accuracy, timeliness, completeness, and clarity. In these contributions, the unavailability and the reliability of some data is not discussed since they assume that the internal operational systems provide all the needed information. In our approach, KI measurement relies on the available data obtainable from the companies involved in the cooperative process and takes into account the trustworthiness of all these sources. Thus, data availability and reputation are considered as variables to consider in data uncertainty evaluation. The evaluation of the quality of data received by other companies involved in a cooperative process is an issue that has been also analyzed in [12]. In [12] authors propose an architecture that evaluates the reputation of the different companies involved in the cooperative process on the basis of the quality of the information that they provide. In their evaluation, they did not consider uncertainty in data values but in order to evaluate data correctness they assume that is always possible to retrieve a certificate and correct value to assess the data provided by the different companies. This is an assumption that is difficult to validate in the real world since the availability of certificated values is scarcely guaranteed.

CONCLUSION AND FUTURE WORK

Uncertainty is a real issue in modern data management. Indeed, the database community (both academia and industry) has already started investing huge efforts into research on uncertain and probabilistic databases, yet there is a lack of business intelligence applications that are able to profit from the results of such research. In this paper, we discuss how to compute uncertain key indicators from uncertain data and we provide a contribution toward uncertain business intelligence, that is, business intelligence that runs atop uncertain data and whose data analysis algorithms take into account uncertainty.

We characterized the problem of computing uncertain key indicators in the context of distributed, cooperative business scenarios that are characterized by different levels of reputation and different levels of visibility into the partners' business practices. We discussed and classified the typical data quality prob-

lems of that scenario and proposed both a conceptual and practical solution to the computation of key indicators, which, in general, we describe as a set of values, their standard deviations, their probabilities and an overall level of confidence (taking into account the reputation of the cooperating partners).
Next, we will apply the concepts and practices discussed in this paper to the case of compliance assessment. We will work on the correlation of uncertain key indicators, so as to identify correlations in the dynamics of two indicators over a predefined time span (e.g., KI_1 drops in average one business day after KI_2 drops). This will allow us to perform root-causes analyses or, if used to look into the future, to help in the prediction of future behaviours. In parallel, we will also work on the visualization of uncertain indicators inside reporting dashboards and test the solutions in the context of compliance assessment.

Acknowledgements: This work was supported by funds from the European Commission (contract N° 216917 for the FP7-ICT-2007-1 project MASTER).

REFERENCES

[1] Antova, L., Koch, C., Olteanu, D. 10^(10^6) Worlds and Beyond: Efficient Representation and Processing of Incomplete Information. *Proceedings of ICDE'07*, pp. 606-615.
[2] Baird, H. Validating performance measurements. Call Center Management Review (2005)
[3] Benjelloun, O., Das Sarma, A., Halevy, A., Thobald, M. and Widom, J. Databases with uncertainty and lineage. *VLDB Journal*, 17(2), pp. 243-264, 2008.
[4] Benjelloun, O., Garcia-Molina, Gong, H., Kawai, H., Larson, T., Menestrina, D., and Thavisomboon, S. D-Swoosh: A Family of Algorithms for Generic, Distributed Entity Resolution. *Proceedings of the 27th IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2007.
[5] Dalvi, N. and Suciu, D. Efficient query evaluation on probabilistic databases. *VLDB Journal*, 16(4), pp. 523-544, 2007.
[6] Daniel, F., Casati, F., Palpanas, T., Chayka, O., Cappiello, C. Enabling Better Decisions through Quality-Aware Reports in Business Intelligence Applications. *ICIQ'08*, November 2008, Boston, pp. 310-324.
[7] Deshpande, A., Guestrin, C., Madden, S., Hellerstein, J and Hong, W. Model-driven data acquisition in sensor networks. *Proceedings of VLDB'04*, 2004.
[8] Green, T., and Tannen, V. Models for Incomplete and Probabilistic Information. *Data Engineering Bulletin,* vol. 29, no. 1, 1996.
[9] Jayram, T.S., Kale, S., Vee, E. Efficient Aggregation Algorithms for Probabilistic Data. *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 2007, pp. 346 – 355.
[10] Kechagioglou, I. Uncertainty and confidence in measurement. *Proceedings of the 18th Panhellenic Conference on Statistics*, 2005.
[11] Krissilov, D.S. Towards the problems of an evaluation of data uncertainty in decision support systems. *Information Theories & Applications*, 13, 2006, pp. 376-379.
[12] Mecella, M., Scannapieco, M., Virgillito, A., Baldoni, R., Catarci, T., Batini, C. The DaQuinCIS Broker: Querying Data and Their Quality in Cooperative Information Systems. *J. Data Semantics,* 1: 208-232 (2003).
[13] Ré, C. and Suciu, D. Efficient Evaluation of HAVING queries on a Probabilistic Database. *Proceedings of the 11th International Symposium on Database Programming Languages*, DBPL 2007
[14] Singh, S., Mayfield, C., Shah, R., Prabhakar, S., Hambrusch, S., Neville, J. and Cheng, R. Database Support for Probabilistic Attributes and Tuples. *Proceedings of ICDE'08*, pp. 1053-1061.
[15] Taylor, J. An Introduction to Error Analysis: The Study of Uncertainties in Physical Measurements. 2nd ed., ISBN 0-935702-42-3.
[16] Wand, Y. and Wang, R.Y. Anchoring Data Quality Dimensions in Ontological Foundations. *Communications of the ACM*, Vol. 39, No. 11, November 1996, pp. 86-95.