A semi-automatic methodology for the design of performance monitoring systems

Claudia Diamantini, Laura Genga, Domenico Potena, Emanuele Storti
Research Paper
Dipartimento di Ingegneria dell'Informazione
Università Politecnica delle Marche
via Brecce Bianche, 60131 Ancona, Italy
{c.diamantini,l.genga,d.potena,e.storti}@univpm.it

Abstract. In the present work, we propose a methodology for the design of a strategic support information system, aimed both at monitoring enterprise daily activities and at supporting decision making by means of Key Performance Indicators (KPIs). In particular, given a set of requested KPIs and the schemas of available data sources, our approach aims at identifying the subset of requested KPIs that can be actually computed over the sources. The KPIs are represented by means of an ontology, over which proper reasoning functionalities have been implemented. Both such automatic functionalities and interactions with experts are required in order to map ontology concepts to schema elements.

Keywords: Performance monitoring system design; Key Performance Indicators; Formula reasoning

1 Introduction

During last years, performance monitoring has gained an increasing importance in enterprise management, due to its role in leading enterprises to achieve strategic goals in a cost-effective way [8]. Identifying (and proper evaluating) suitable key performance indicators (KPIs) with respect to enterprise goals plays a central role both in managing daily activities and in monitoring the degree of achievement of long-term strategies. However, selection and monitoring of the right set of KPIs often turns out to be a non-trivial task, depending on goals to achieve and on expertise of managers. An intensive research effort has being performed in order to define methodologies and best practices to deal with such a topic, as shown by the huge amount of contributions in Literature devoted at the design of performance measuring systems (see e.g. [9] for an overall survey). Within such a context, a well-known issue regards how to match KPI definitions with the enterprise data sources; two main alternative approaches are usually exploited to this end, differing for the relative importance assumed by the ideal KPIs and by the real-world data, namely the “goal-driven” and the “data-driven” approach.

In the former, the most relevant decisions are taken by the manager, whose focus is typically on the selection of KPIs to monitor, while little or no regard
to real data. To this end, the manager needs an overall knowledge of all existing KPIs, from which she selects those most suitable for the applicative domain, mainly supported by her expertise. As one can easily argue, such a selection usually requires strong efforts from the manager, especially with complex analysis tasks. Moreover, a KPI is by definition a synthetic measure, usually defined over a set of other KPIs needed for its computation. Consequently, computing KPIs can likely involve the definition of complex ETL procedures to obtain the needed data, especially when KPI selection is performed without any consideration of data actually produced by the enterprises; in such case, retrieval of needed data can become a very time-consuming and onerous task. In the worst case it's not possible to compute KPIs at all.

The data-driven approach assumes that the KPIs to monitor is defined directly starting from enterprise available data. The IT expert has here a prominent role, since she has the knowledge about enterprise data sources needed to derive indicators. This approach limits the need of complex ETL procedures and guarantees that all selected indicators are actually computable. However, by analyzing the data one can likely derive simple indicators, typically obtained by elementary formulas that provide very little support in complex monitoring tasks. More meaningful and general KPIs usually have a complex structure, difficult to grasp in a bottom-up fashion.

A hybrid approach is also possible, in which the manager’s perspective is merged with that of IT experts. To this end, high-level complex indicators provided by the former have to be mapped to simpler low-level ones identified by the latter; however, such a matching often results very challenging, as it deals with formula definitions, which typically are hard to make explicit and manage.

To overcome the limits of discussed approaches, hereby we propose a semi-automatic methodology for KPIs selection and mapping. More precisely, our proposal firstly involves the definition of the KPIs of interest for a manager, and then their mapping with the enterprise data schema, which represents data produced by available data sources. Such mappings allow to identify, within the set of all requested KPIs, which are actually computable by exploiting data at disposal. The methodology is based on the KPIOnto, an ontology conceptualizing KPI domain and able to handle their formulas. KPIs in KPIOnto are categorized with respect to the application domains, hence supporting the user in the proper selection of KPIs. As a matter of fact, the research space is limited to KPIs useful for monitoring a given domain or goal. Description and other properties can then be used to refine the selection. As regards KPI mappings, users are also provided with a set of powerful reasoning services that deal with KPI formulas and automatically derive the set of lower-level indicators needed to compute them. In such a way, users have no need to manage KPIs computation, as they focus on simpler indicators, looking for their mapping on data schema.

The rest of this work is organized as follows. In section 2 we introduce a case study used as illustrative example through the paper. Section 3 describes both the main features of KPIOnto and the implemented reasoning functionalities,
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HourlyCost</td>
<td>Hourly personnel costs</td>
</tr>
<tr>
<td>HourRate</td>
<td>Hourly teacher rate</td>
</tr>
<tr>
<td>InvestmentInEmpDev</td>
<td>Money spent on employee training/development</td>
</tr>
<tr>
<td>NumHours</td>
<td>Number of working hours</td>
</tr>
<tr>
<td>NumTrainingHours</td>
<td>Time needed to train personnel to fulfill project tasks</td>
</tr>
<tr>
<td>OverheadRate</td>
<td>Percentage to be added to personnel costs, which takes into account fixed costs, e.g. rewards, taxes, etc.</td>
</tr>
<tr>
<td>PersonnelCosts</td>
<td>Total costs of personnel</td>
</tr>
<tr>
<td>PersonnelTrainingCosts</td>
<td>Expenses to train personnel to fulfill project-related tasks</td>
</tr>
<tr>
<td>TeachCosts</td>
<td>Total costs for teachers</td>
</tr>
<tr>
<td>TravelCosts</td>
<td>Costs related to travels carried out for project tasks</td>
</tr>
</tbody>
</table>

Table 1. List of indicators used in the case study.

which are the basis of the methodology proposed in section 4. The last two sections are devoted to discuss related work and draw conclusions.

2 Case Study

This work is conceived within the EU Project BIVEE\(^1\), which is aimed to develop a platform to support enterprises in pooling their experiences and resources in order to leverage their production and innovation potential, and to optimize costs. The platform gives support in the monitoring of Virtual Enterprises (VE), which are (temporary) aggregations of enterprises that from the outside can be seen as a single enterprise. In particular, for what concerns the present work, the platform helps in defining both the set of indicators to be used for monitoring the enterprise and the formulas to calculate them.

In Table 1 we propose an excerpt of indicators used by one of the end-user partners of the project. In particular, the partner is focused on innovative projects, from metrology to robotics domains, finalized to satisfy specific requests of the customer. To support the methodology proposed in this work, hereafter we refer to a case study based on KPIs described in such a table. Let’s suppose that a certain enterprise \(\text{Ent}\) needs to monitor two main kinds of costs, i.e. those deriving from innovation projects and those related to development of employees skills. To this end, firstly the set of KPIs matching with such analysis requirements are to be identified; this task is carried out by the domain expert that picks out the set of requested KPIs. In particular, in the present case study there are two high-level indicators that are suitable for the enterprise goals, namely \(\text{Costs}\), which stands for the amount of expenses for innovation, and \(\text{InvestmentInEmpDev}\), which represents costs for training of employees. Each of such indicators is computed by a well-defined formula, namely:

\[
\text{Costs} = \text{TravelCosts} + \text{PersonnelCosts}
\]  

\(^1\) http://bivee.eu
InvestmentInEmpDev = PersonnelTrainingCosts + TeachCost \hspace{1cm} (2)

Note that some operands of KPIs formulas are indicators too, that are computed as:

PersonnelCosts = NumHours \times HourlyCost \times (Overhead + 1) \hspace{1cm} (3)

PersonnelTrainingCosts = HourlyCost \times NumTrainingHours \hspace{1cm} (4)

TeachCost = NumTrainingHours \times HourRate \hspace{1cm} (5)

3 Semantic model

3.1 KPI Ontology

KPIOnTO is an ontology aimed to provide a formal reference model for KPIs. The ontology serves as a global shared model capable to define descriptive properties of indicators together with the mathematical formulas needed to calculate them. KPIOnTO arranges the most relevant concepts in classes, namely Indicator, Dimension and Formula. The Indicator is the pivotal class of the KPIOnTO, and its instances (i.e., indicators) describe the metrics enabling performance monitoring. Indicator are described through a priority, a set of dimensions, a formula and a unit of measurement (i.e., both the symbol and its description). KPIs are arranged in a taxonomy, mainly derived from the VRM model [13]. The semantics of an indicator cannot be fully grasped without the representation of its Formula, which is the way the indicator is computed [2]. When the formula is not specified, the indicator is said atomic, and is independent on other indicators (e.g., HourlyCost or NumHours in the case study). Otherwise indicators are said compound, e.g. PersonnelTrainingCosts, and are defined in terms of other indicators producing a lattice of dependencies like in Figure 1. Each formula is characterized by the related indicator, the aggregation function, the way
the formula is presented, the semantics (i.e., the mathematical meaning) of the formula, and references to its dependencies.

We resort to OWL2-RL for the representation of the descriptive properties of the ontology, and to mathematical standards for describing the way the formula is presented and its semantic (MathML\(^2\) and OpenMath\(^3\)).

Finally, a dimension is the coordinate/perspective to which the metric refers. Following the multidimensional model, the Dimension class (e.g., TimeDimension, OrganizationDimension) is usually structured into a hierarchy of levels, where each level represents a different way of grouping members of the dimension \[7\]. For instance, means of transportation can be grouped by transport companies, and days are grouped by weeks and years. Each level is instantiated in a set of elements known as members of the level, e.g. the company “ACME”, the weeks “3rd-2012” and “42nd-2011”. A Priority is the goal of optimization for which the indicator is used, e.g. Cost or Velocity. Although the methodology described in the next section focuses only on indicators and formulas, information about Dimension and Priority will be exploited in a future work.

3.2 Reasoning functionalities

On the top of these languages, in order to define KPI reasoning functionalities as a support to integration and analysis, we need to represent both OWL2-RL axioms and MathML formulas in a common logic-based layer. To this end, we refer to the first-order logic and define the functionalities in logic programming (LP), to which both languages have a simple translation preserving expressiveness and (sub)polynomial complexity in reasoning.

While formulas are represented as facts, manipulation of mathematical expressions is performed through specific predicates based on PRESS (PRolog Equation Solving System), which is a formalization of algebra in Logic Programming for solving equations. Such predicates can manipulate an equation through a set of rewriting rules.

To support more advanced functionalities we introduced predicates for checking the consistency of (formulas in) the ontology, for supporting the setup of a virtual enterprise, for enabling multidimensional queries over incomplete data cubes, and so forth \[3, 4\]. In the following, we present predicates useful to determine the set of indicators that can be computed by an enterprise, and to understand how they can be actually calculated. The methodology we propose is based on these predicates.

\(-\ c_1(\varphi, L)\), to calculate the set \(L\) of common indicators from a set \(\varphi\) of indicators. More formally, given a set of indicators \(\varphi = \{I_1, I_2, ..., I_n\}\), common indicators of \(\varphi\) is the minimal set of atomic indicators needed to compute all formulas of \(\varphi\). For instance, by referring to the case study, if \(\varphi = \{\text{PersonnelTrainingCost, TeachCost}\}\), \(L\) is equal to \(\{\text{HourlyCost, Num\-TrainingHours, HourRate}\}\).

\(^2\) http://www.w3.org/Math/

\(^3\) http://www.openmath.org/
Common indicators are useful to determine the possible sets of information that have to be provided by an enterprise in order to calculate the requested set of KPIs or, in a bottom-up fashion, to recognize which indicators can be derived, given the available information at enterprise-level. Hence, they are essential tools at design time.

- $\text{inferableInd}(\varphi, L, F)$, which returns the list $L$ of KPIs that can be derived by a given set of indicators $\varphi$. The set $F$ contains the formulas for indicators in $L$ including as operands only elements in $\varphi$. For instance, if $\varphi = \{\text{NumHours, OverheadRate, PersonnelCosts, NumTrainingHours, HourRate}\}$, the output $L$ is the set $\{\text{PersonnelTrainingCosts, TeachCost}\}$; and $F$ is the set
d
\[
\begin{align*}
\{ & \text{TeachCost} = \text{HourRate} \times \text{NumTrainingHours}, \\
& \text{PersonnelTrainingCosts} = \frac{\text{NumTrainingHours} \times \text{PersonnelCosts}}{\text{NumHours} \times (1 + \text{OverheadRate})} \}\n\end{align*}
\]

It should be noted that the service can return in $F$ both the KPIs formulas stored in the ontology, as in the case of $\text{TeachCost}$, and rewritten formulas, as we can see for $\text{PersonnelCosts}$.

- $\text{reqCompInd}(\varphi, C, L)$, returns the collection $C$ of minimal sets of indicators needed to compute the set $\varphi$. The predicate is satisfied for each set of indicators $L$ provided by the enterprise, such that a formula for $\varphi$ can be derived from the ontology and it is defined on the basis of this set. Duplicates and atomic indicators are discarded.

- $\text{isMandatory}(C, L_1, L_2)$ is used to determine the set $L_1$ of indicators that are common to all the sets of the collection $C$. The collection $C$ is the one returned by the predicate $\text{reqCompInd}(\varphi, C, L)$. The set $L_2$ is formed by indicators for which those in $L_1$ are mandatory.

4 Methodology

In this Section, we propose a methodology for identifying indicators needed to compute a set of requested KPIs on the basis of available data. Let’s define $\varphi$ as the set of requested KPIs and $S$ as the data schema of the enterprise data sources. The pseudo-code of the proposed methodology is shown in the following.

1. Compute the Common Indicators $\text{ci}(\varphi, L_{ci})$
2. Define $ci^+$ as the subset of the $L_{ci}$ that maps to $S$
3. Compute $\text{reqCompInd}(\varphi, L_{comp}, ci^+)$
4. Compute $\text{isMandatory}(L_{comp}, L_{mand}, L_{ind})$
5. If not exists a map for some element of $L_{mand}$ then set $\varphi = \varphi - L_{ind}$
6. Set $ci^+ = ci^+ \cup L_{mand}$
7. Compute $\text{reqCompInd}(\varphi, L_{comp}, ci^+)$
8. while ($\emptyset \notin L_{comp}$ AND $L_{comp} \neq \emptyset$)
   (a) Add to $ci^+$ elements of $L_{comp}$ for which a mapping exists
   (b) Compute $\text{reqCompInd}(\varphi, L_{comp}, ci^+)$
9. Compute \( \text{inferableInd}(\varphi, \varphi^+, \emptyset) \)
10. Set \( \varphi^- = \varphi - \varphi^+ \)
11. Return \{\( \varphi^+, \emptyset, \varphi^- \)\}

Within our case study, the procedure begins with \( \varphi = \{\text{Costs, InvestmentInEmpDev}\} \). Starting from such a set, the procedure has to check if the available data allow to derive the requested KPIs. To this end, at first the predicate \( c_i \) is invoked; the output is the set \( L_{c_i} = \{\text{OverheadRate, NumHours, HourlyCost, NumTrainingHours, HourRate, TravelCosts}\} \). Such set is proposed to the user, which has to verify if a direct map with some elements of \( S \) exists; if so, such indicators are added to \( c_i^+ \) set. It is noteworthy that such a mapping often turns out to be a non-trivial task. Let’s suppose that the enterprise data schema is the one represented in Figure 2. As one can easily argue, the user is often not able to detect all possible mappings at once; many heterogeneities typically exist between schema elements and indicators, e.g. they can be represented with different naming convention, or some indicators cannot be obtained by a one-to-one mapping, but they require proper elaboration upon schema elements. Therefore, the user has to exploit her knowledge to identify correct associations between schema elements and indicators. However, she can find a valuable support for such a task in the ontology, which provides the semantic interpretation of KPIs, thus simplifying the research of corresponding schema elements. The main knowledge required to the user is about the data schema. In our case study, the user is able to derive the following mappings set \{\( \text{NumHours:Partecipation.WorkingTime, NumTrainingHours:Course.Duration, TravelCosts:Mission.Amount}\)\}. Note that, for instance, the second map is possible only by exploiting knowledge about the schema (the duration of a course is measured in hours). We can also note that some common indicators are missing in such a set, i.e. \{\( \text{OverheadRate, HourlyCost, HourRate}\)\}. As regards the first one, by its description in KPIOnto one derives that it refers to those generic costs, not measurable with precision, associated to the participation of the enterprise to a generic project. Since a quite common practice in project reports consists in taking into account such costs simply by means of a fixed share, the indicator will have the same value for all
the enterprise projects. Therefore, the user can easily obtain it, despite the fact that it cannot be derived from the data schema. However, such an assumption does not hold for the last two missing indicators, which actually represent a lack of information. Indeed, the schema reveals that the enterprise does not store the hourly cost of its employees, considering instead their monthly salary. Moreover, as for the training courses, the total fee of a course is represented, without any specification about the hour rate of a single teacher.

In Figure 3 the mapping step of atomics is shown: the gray and white circles respectively represent indicators that cannot be derived and those available. These latter, in general, are both indicators directly mapped on \( S \) and indicators that can be derived from the mapped ones. In our case the only available indicators are those directly associated to some attributes of \( S \); however, starting from these latter, anything else can be derived. In step 3 of the procedure, the output of \( \text{reqCompInd} \) is:

\[
L_{\text{comp}} = \{ \{ \text{PersonnelCosts}, \text{TeachCost} \}, \{ \text{TeachCost, PersonnelTrainingCosts} \} \}.
\]

In step 4, the predicate \( \text{isMandatory} \) extracts all mandatory indicators, i.e. \( L_{\text{mand}} = \{ \text{TeachCost} \} \) and \( L_{\text{ind}} = \{ \text{InvestmentInEmpDev} \} \). In our schema, a mapping for \( \text{TeachCost} \) exists, i.e. \( \{ \text{TeachCost} = \text{Teaching.Fee} \} \) (assuming that the \( \text{Teaching.fee} \) element is the fee required by a teacher for a certain course). Note that if the user cannot map some mandatory elements, the procedure cannot derive all requested KPIs; therefore, the \( \varphi \) has to be updated, by removing the elements of \( L_{\text{ind set}} \) (step 5 in the procedure). At this point \( \text{reqCompInd} \) is again invoked, obtaining \( L_{\text{comp}} = \{ \{ \text{PersonnelCosts} \}, \{ \text{PersonnelTrainingCosts} \} \} \).
From the schema, it can be noted that the information about personnel cost is actually represented; however, some ambiguities exist. The cost of an employee can be represented both from her salary and from her cost in a specific project. Hence, in order to select the correct schema element, the user has to know the meaning of the indicator PersonnelCost; in such situation, the support provided by the ontology plays a key role, since it is needed to detect the correct interpretation. Therefore, a new mapping is created, i.e. \{PersonnelCosts : Participation.TotalCost\} and PersonnelCosts is added to \(\varphi^+\). After this new mapping, by invoking reqCompInd we finally found that one of the solutions is the empty set, meaning that there is no need to add any other indicators to find all \(\varphi\) elements. Therefore, by invoking inferableInd service (step 9) we obtain that \(\varphi^+ = \{\text{Costs, InvestmentInEmpDev}\}\) (i.e. it corresponds to \(\varphi\)) and, hence, \(\varphi^-\) is the empty set. Consequently, in \(\vartheta\) we find all the computation formulas for the set of requested KPIs.

Such a situation is represented by Figure 4. It’s noteworthy that through the final mapping all indicators turn out to be available, even HourlyCost and HourRate, for which the user cannot found a direct mapping in previous steps. This is due to the fact that the used predicates are able to move in all directions through the lattice of formulas; hence, it takes into account both the higher-level and lower-level indicators with respect to those provided, thus deriving new formulas for HourlyCost and HourRate on the basis of available indicators. Note

![Mapping of compound indicators](image)

**Fig. 4.** Mapping of compound indicators

that if there is some missing information we cannot derive all requested KPIs; for example, if the user cannot map PersonnelCosts or PersonnelTrainingCosts, the requested KPIs cannot be computed. In such case, in step 9, \(\varphi^+\) is just a subset of \(\varphi\). Therefore, in \(\vartheta\) there are only the formulas related to computable KPIs, and in \(\varphi^-\) we have the set of KPIs that cannot be derived.

Summarizing, by running our methodology on this case study we have highlighted some important benefits for the user with respect to other typical design methodologies. These latter, in fact, usually require that the user makes an in-
tensive use of her personal knowledge during the requirement analysis phase, to match the set of high-level goals with the real data. In our methodology, some of needed knowledge is actually provided by the system, by means of the KPIOnto; indeed, the KPIs computation formulas are managed by the system, thus allowing the user focus only on the single operands in order to perform the mapping. In other words, the user can deal not directly with high-level goals (i.e. KPIs) but with lower-levels ones, which in most of the cases result simpler to map onto data schema. Moreover, several mapping alternatives, involving both atomic and compound indicators, are presented to the user, that hence can choose those which better fit with her data schema.

However, we can note that the effectiveness of the methodology is constrained, on a certain extent, on the ontology granularity. Generally speaking, although some indicators are atomic in the ontology, they can require data manipulation to be derived from the data schema, thus meaning that the ontology has not the right granularity level with respect to that schema. In this case, to derive such indicators the user’s knowledge is needed; since the ontology does not store the required formula, clearly no reasoning service can be exploited to support the user. Anyway, we can likely figure out that such a situation will originate an ontology refinement process, aimed to better match the user needs.

5 Related Work

Some similarities there exist between with our mapping approach and design methodologies for data warehouses, and in particular w.r.t. the information requirement analysis phase, aimed at matching user requirements with available data sources. Such mappings are usually performed under a goal-driven approach, mainly aimed at deriving DW requirements from user’s goals, or under a data-driven approach, centered on available data[14]. Several works also take into account an hybrid approach; for instance, in [5], the authors describe a methodology, named GRAnD, which can be exploited both in a goal-driven approach and in a hybrid one. More precisely, GRAnD firstly identifies the main system actors with their goals, representing these latter through $i^*$ diagrams [15]. Such elements are then exploited to model the set of required DW facts, dimensions, and measures. Then, a mapping is performed between the modeled DW entities and the data schema, by means of a set of guidelines regarding the selection of proper data schema elements. Note that modeling of user requirements and their mapping on schema elements are mainly based on experience of the system designer. In fact, although the $i^*$ diagrams result simpler to obtain and manage than an ontology, they cannot provide the user with any kind of automatic support. An example of the application of ontology formalisms in mapping between high-level business goals and enterprise data schemas is presented in [6]. Here the authors address the exchange, between different organizations, of their business calculation formulas, usually defined over their own DWs. To simplify such exchange, the authors propose to define three ontologies, i.e. one dedicated to the interesting business calculation, one to DW elements and, fi-
nally, a mapping ontology, aimed at linking entities of previous ontologies. Both expertise knowledge and automatic inference rules are used to define mappings. Note that, although useful to knowledge transferring, a dedicated mapping ontology requires a significant effort to be set up, and to be maintained. Moreover, although the authors describe formulas by means of a dependencies lattice, they can use this latter only in a top-down fashion. More precisely, they can automatically derive the low-level components needed to compute the high-level business calculation definition; however, they cannot move from low to high level, i.e. they cannot manipulate or rewrite formulas (e.g. inverting them), thus limiting the set of mapping links they can exploit.

For what concerns a systematization of KPI for enterprise environments, there is a plethora of Performance Indicators definitions and/or Glossaries provided by researchers and research groups, international and national public bodies, e.g. in reference models like the Supply-Chain Reference Model (SCOR) [11], the Value Chain Reference Model (VRM) [13] and Six-Sigma [12] (see also [3] for a more comprehensive list of performance measurement approaches and KPI description). Although homogeneous in the kinds of information provided, they mainly differ on the specific domain of interest and on the degree of precision and formality of the description. Indeed, for what concerns formal definition of indicators, several approaches to semantic modeling have been proposed (see [10] for a survey). However, almost all of them are specifically targeted to support more advanced monitoring within the context of business processes, and hence focusing only on the operational levels with little or no regard to strategic and managerial perspectives, or to innovation KPIs. To the best of our knowledge, the only work dealing with the explicit representation of KPI formulas is by Barone et al. [1], in which indicators are formally represented and arranged in hierarchies, with the aim to exploit run-time monitoring and what-if analysis of business processes.

6 Conclusion

In the present work we described a semi-automatic methodology devoted at checking if a set of requested KPIs can be computed from available data of the enterprise. Such an approach is based on the KPIOnto, that we introduced together with its reasoning functionalities. By applying our proposal to an illustrative case study, we have shown that it provides some important benefits to the user, especially with respect to a manual mapping. Nevertheless, effectiveness of the methodology in supporting users depends also on the granularity level of KPIOnto with respect to the given data schema.

In future works, we plan to improve the mapping support by extending the proposed approach with state of the art results in the Semantic Web field, especially for what concerns the semantic annotation of database schemas. Indeed, having at disposal an ontology describing the enterprise databases one can exploit more sophisticated techniques to combine this latter with the KPI-Onto. Among these, we are particularly interested in the application of similarity
matching techniques, as the detection of semantic similarities between a database and KPI\textsc{Onto} entities will help to obtain more meaningful mappings.

References