Ontology-based Metrics Computation for Business Process Analysis

Carlos Pedrinaci
Knowledge Media Institute, The Open University
Milton Keynes, MK7 6AA, UK
c.pedrinaci@open.ac.uk

John Domingue
Knowledge Media Institute, The Open University
Milton Keynes, MK7 6AA, UK
j.b.domingue@open.ac.uk

ABSTRACT

Business Process Management (BPM) aims to support the whole life-cycle necessary to deploy and maintain business processes in organisations. Crucial within the BPM life-cycle is the analysis of deployed processes. Analysing business processes requires computing metrics that can help determining the health of business activities and thus the whole enterprise. However, the degree of automation currently achieved cannot support the level of reactivity and adaptation demanded by businesses. In this paper we argue and show how the use of Semantic Web technologies can increase to an important extent the level of automation for analysing business processes. We present a domain-independent ontological framework for Business Process Analysis (BPA) with support for automatically computing metrics. In particular, we define a set of ontologies for specifying metrics. We describe a domain-independent metrics computation engine that can interpret and compute them. Finally we illustrate and evaluate our approach with a set of general purpose metrics.

Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Metrics—*Performance measures, Process metrics*; D.2.9 [**Software Engineering**]: Management—*Life cycle, Productivity*; H.4.1 [**Information Systems Applications**]: Office automation—*Workflow management*; I.2.4 [**Information Systems Applications**]: Knowledge Representation Formalisms and Methods

General Terms

Measurement, Management, Performance

Keywords

Ontology, Metric, Business Activity Monitoring, Business Process Analysis, Semantic Business Process Management

1. INTRODUCTION

In the business world the maxim "*if you can't measure it, you can't manage it*" is often used. Although it is too blunt a statement, it captures an important essence in current management approaches which try to maximise the aspects measured in order to evaluate, compare, and control the evolution of businesses. For instance the Balanced Scorecard is a popular "*set of measures that gives top managers a fast but comprehensive view of the business*" [10]. In a nutshell, the Balanced Scorecard defines four perspectives and suggests for each of them a set of aspects that managers should focus on. Assessing how well a company is doing is then a matter of calculating metrics and contrasting them with respect to pre-established goals for each of these key aspects. In the same vein but in more concrete terms, the Supply-Chain Council defines in the Supply-Chain Operations Reference-model a set of Supply-Chain targeted metrics such as "fill rate by order" or "total order fulfilment lead time" [18], which represent what is often referred to as Key Performance Indicators in the literature [10, 3]. In short, effectively analysing business processes requires, although it is not limited to, computing metrics that can help determining the health of business activities and thus the whole enterprise.

Business Process Analysis (BPA) uses the logs captured by the underlying IT infrastructure such as Enterprise Resource Planning, Customer Relationship Management, and Workflow Management systems to derive information concerning the well-being of business processes [21]. Common practice within the industry is to build a Data Warehouse which consolidates all sorts of corporate information and enriches it with derived statistical data [5]. Not surprisingly one main challenge envisaged by BPA solutions regards gathering and integrating large amounts of heterogeneous yet interrelated data within a coherent whole.

Once a Data Warehouse has been built and populated, Online Analytical Processing (OLAP) and Data Mining tools enable sophisticated data analysis that can help business analysts understand their businesses and even predict future trends. However, the semantics of the data being implicit, both OLAP and Data Mining techniques can hardly benefit from contextual knowledge about the organisation at analysis time, and strictly rely on human interpretation of the results [22]. This not only brings additional manual labour to an already complex and time consuming task, but it also prevents the automation of certain decision making procedures. As a result enterprises often develop expensive

domain-specific solutions which become an additional management overhead when changes within the enterprise need to be implemented.

We have previously argued for the use of semantic technologies, namely ontologies and Problem-Solving Methods [17], as a means to enhance the state of the art in BPA [2]. In the light of this vision, we have defined Core Ontology for Business pRocess Analysis (COBRA) [14] which provides a core terminology where business practitioners can map domain-specific knowledge in order to analyse their business processes. We have also defined additional extensions for capturing semantically the logs produced by IT systems and for deriving knowledge in terms of COBRA. In this paper we present an extension to our framework that allows to process the whole body of knowledge about business processes and their executions in order to compute general purpose as well as domain-specific metrics. In particular, we describe a comprehensive set of ontologies for defining metrics, and a domain-independent engine that is able to interpret these metrics and automatically compute them. The work described herein therefore constitutes an important step forward towards a domain-independent, fully automated, and semantically enhanced Business Process Analysis framework.

The remainder of this paper is organised as follows. We first describe our ontological framework for defining metrics. We then explain our Generic Metrics Computation Engine. Next, we demonstrate how our approach can be applied to analysing business processes by means of Generic Business Metrics Ontology which both validates and enhances our framework with a library of reusable metric definitions. Finally, we contrast our approach with existing work, present some conclusions and introduce future research objectives.

2. ONTOLOGICAL FRAMEWORK FOR METRICS DEFINITION

Major efforts are being devoted to enhancing the state of the art in Business Process Management (BPM) [21] by using Semantic Web and Semantic Web Services technologies throughout the life-cycle of business processes [13]. The work presented in this paper is part of SENTINEL, a semantic business process monitoring tool [15]. The tool, depicted in Figure 1, connects to a number of repositories populated with semantic information about processes and their executions in order to support advanced BPA techniques. In this paper we shall focus in the Metrics Computation Engine component (see Figure 1), paying particular attention to the Knowledge Level definitions that support specifying and automatically computing domain-specific metrics based on the body of knowledge gathered by IT systems during the enactment of business processes.

Crucial to the work described herein is thus the need for representing both static knowledge, i.e., the metrics, and dynamic knowledge, i.e., how to actually compute them, [16]. The ontologies presented in this paper have therefore been developed using the Operational Conceptual Modelling Language (OCML) [12] which seamlessly supports the integration of both kinds of knowledge paving the way for a rapid prototyping of a fully operational solution[1]. It is worth noting however that OCML provides support for importing and exporting data represented in other languages such as OWL and WSML–the language used within the project–and therefore allows the wider application of our techniques over data represented in Semantic Web and Semantic Web Services formalisms.

2.1 Core Ontology for Business Process Analysis

Although describing COBRA is outside of the scope of this paper we introduce in this section those aspects that are necessary for understanding the rest of the paper. The reader is referred to [14] for further details. COBRA provides a pluggable framework based on the core conceptualisations required for supporting BPA and defines the appropriate hooks for further extensions in order to cope with the wide-range of aspects involved in analysing business processes. COBRA divides the world into *Temporal Entities* and *Persistent Entities* whereby the former are entities that have a temporal extent whereas the latter are essentially independent of time.

Core concepts in COBRA are *Business Activity* and *Business Activity Realisation*. Business Activity represents the specification of a business activity at a high-level where aspects such as the control flow are abstracted away. There are two kinds of Business Activities, namely *Process* and *Activity*. Activity represents atomic Business Activities whereas Processes are *composedOf* at least two Business Activities. Business Activity Realisations are Time Spanning Entities which represent the actual execution of Business Activities. Mirroring Business Activities, Process Instance and Activity Instance are the two kinds of Business Activity Realisations considered. Despite their name, which originates from BPM literature [21], both are concepts which represent the actual executions of Processes and Activities respectively.

Concerning the analysis themselves such as metrics, the previously presented version of COBRA [14] solely captures the concept *Analysis Result*, a Temporal Entity, which has two disjoint sub-concepts: *Qualitative Analysis Result* and *Quantitative Analysis Result*. As part of our work on metrics definition and computation, we have slightly extended COBRA itself. First, COBRA now imports Physical Quantities ontology, in order to include support for units of measure (we shall explain this ontology in more detail in Section 2.2). Secondly, we have introduced the concept *Analysis* a Non-Agentive Non-Physical Entity, which is refined into *Qualitative Analysis* and *Quantitative Analysis* based on the type of Analysis Result they produce. This provides us the means for maintaining a library of Analysis specifications (e.g., metrics, time series, etc.), and it allows us to distinguish between the analysis themselves and the actual results. Indeed, the relationship between Analysis and Analysis Result has also been captured, in such a way that every Analysis Result is a result for a particular Analysis, and every Analysis may have several Analysis Results. Hence we can obtain all the results for a particular analysis, track its evolution over time, apply time series analysis, etc.

[1]All the ontologies described herein can be found at http://www.cpedrinaci.net

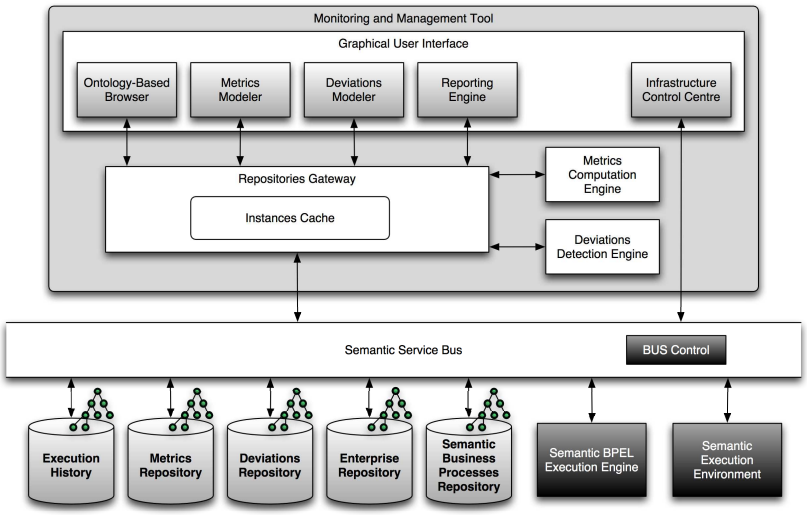

Figure 1: SENTINEL Architecture.

Finally, as part of this extension, COBRA also includes *Query*, the Qualitative Analysis *par excellence*. The concept Query supports capturing ontological queries in a similar vein to stored procedures in Databases. Capturing Queries ontologically provides the means for maintaining a library of useful queries parametrised in a way such that business analysts can directly reuse and apply them over their specific domain. Hence, this addition is of particular relevance for better supporting business practitioners–typically not highly-skilled technically–in analysing processes. Queries are characterised by a set of input roles [12], a list of output types, and a body which captures in a unary procedure, the ontological query parametrized in terms of the input roles. We shall see concrete examples and uses of queries in Section 4.

2.2 Units Manipulation

Metrics quantify some aspect of interest. Their results are basically quantities, numbers, but they are often expressed in some unit of measure. For instance the execution time of a process is indicated using some unit of time such as seconds. An appropriate manipulation of metrics thus depends on a proper support for units of measure. This includes support for characterising quantities using units of measure, but also requires adapting the algebraic operations for dealing with them. We have therefore defined a set of ontologies that provide this capability to our framework. In particular, based on EngMath [8], we have defined the following ontologies: Physical Quantities, Standard Prefixes, Standard Dimensions, International System of Units, and Units Manipulation. It is worth noting however that, as opposed to EngMath which is explicitly designed for sharing mathematical formulae, our ontologies focus on the actual support for mathematical manipulation.

Physical Quantities ontology, captures the basic definitions necessary for what in physics, chemistry and engineering is referred to as dimensional analysis. Central to the ontology are the notions of *Physical Quantity* and *Physical Dimension*. A Physical Quantity is a measure of some quantifiable aspect of the world, for example your height. We characterise Physical Quantities by their *Unit of Measure*, like "kilogram". In fact, the main distinction between Physical Quantities and plain numbers is the fact that they are expressed in some unit of measure which determines how it should be treated within algebraic operations. A Unit of Measure provides a standard measurement for some particular Physical Dimension. For example, "gram" and "stone" are units of measure for the "weight" dimension. Units of Measure are further characterised by an abbreviation (e.g., "s" for "second"), and a conversion rate. A Physical Dimension is thus a property that we associate indirectly to Physical Quantities, but which allows us to establish whether any two Physical Quantities are compatible, and therefore determines which are the mathematical operations that can be performed between the two. For example, we cannot add the quantities "200 meters" and "1 second" because these are units of measure for different dimensions.

Physical Quantity is further refined into the disjoint concepts *Constant Quantity* and *Function Quantity*. Function Quantities are those whose value is defined as a function like for example "the speed of a particle in free-fall as a function of time". Conversely, as the name implies, Constant Quantities are those whose value is constant and they are therefore characterised by a magnitude, i.e., a numeric value. A special kind of Constant Quantity are so-called *Dimensionless Quantities*, i.e., Constant Quantities whose unit of measure is a special one which is the *Identity Unit*[2]. This special kind of unit of measure allows us to have an homogeneous treatment for Physical Quantities, independently of whether they are expressed in some unit of measure or not.

The final concept introduced within Physical Quantities ontology, captures the notion of *System of Units*. A System of Units is a reference system of measurement which is defined by a set of base units that can be used for specifying anything which is measured. The most commonly used system of units is the "Système International d'Unités" (SI) [1] or

[2]As a consequence we also have a particular dimension called the *Identity Dimension*

International System of Units which we have partly captured by means of the ontologies Standard Prefixes, Standard Dimensions, and International System of Units. The former captures typical prefixes such as *Kilo* or *Mega* as conversion rates that can be used for manipulating units. Standard Dimensions defines the base dimensions defined in SI, plus *Currency*, *Amount of Information*, and *Signal Transmission Rate* as additional base dimensions, and some other derived ones such as *Volume* or *Speed* for convenience. On the basis of Physical Quantities, Standard Dimensions, and Standard Prefixes, International System of Units ontology implements SI using the standard units [1].

Finally, Units Manipulation ontology redefines the basic mathematical algebraic operators to support the use of Physical Quantities. Additionally, the ontology also redefines the basic relations for comparing numbers in order to support Physical Quantities (e.g., '>', '<'), defines functions for converting between any two compatible units (using the base one as reference), and provides a set of basic statistic operations, i.e., count, sum, prod, max, min, mean, range, standard-deviation, mode, variance. The latter are indeed of most use for metrics computation as described next.

2.3 Metrics Ontology

Metrics Ontology, depicted in Figure 2, aims to support business analysts in defining and computing metrics over the low-level audit trail information generated by the IT infrastructure. It extends COBRA in order to capture more detailed information about Analyses to fulfil two main purposes. Firstly, the metric definitions are such that our Generic Metrics Computation Engine described in Section 3 can interpret and compute them automatically in a domain independent manner. Secondly, by virtue of our ontological framework, metric results enhance the overall body of knowledge about business processes execution better supporting subsequent analyses for identifying potential improvements, detecting deviations, etc. In short, Metrics Ontology provides us with the capacity for specifying and computing metrics, as necessary for analysing and managing business processes, in a domain independent way.

On the basis of our conceptualisation we can capture kinds of metrics, e.g,. "process instance execution time", as well as specific metrics to be computed, e.g., "process instance X execution time". The former are defined as concepts, whereas the latter are modelled as instances. In this way we can provide libraries of metrics such as general purpose ones, or specific for some domain like Supply-Chain, and at analysis time the analyst can specify which of these metrics should be computed over which entities by instantiating them. This provides a convenient way for organising metric definitions and seamlessly supports the comparison of results by kind of metric, e.g., "which is the process which takes longer", as well as it allows tracking their evolution over time.

Central to Metrics Ontology is the concept *Metric* which is defined as a Quantitative Analysis (see Section 2.1). Metrics are specified by a set of input roles that point to domain-specific knowledge [12]. We refine Metrics into two disjoint kinds, Function Metrics and Aggregation Metrics. A Function Metric is a metric that can be evaluated over a fixed number of inputs. For example, the Metric *Process Instance Execution Time* is a Function Metric which takes as input one Process Instance. In order to support Function Metrics computation, which is indeed metric dependent, each metric has a computation expression which is defined as a unary procedure[3], the argument of the procedure being the metric itself. Finally, we have included the Function Metric *Ratio* since it is a commonly used kind of metric. In Section 4 we include concrete examples showing how these metrics and their computation expressions can be defined.

Conversely, Aggregation Metrics (e.g., "average process execution time") take an arbitrary number of individuals of the same kind (e.g., a set of Process Instances) as input. Therefore, Aggregation Metrics are computed over a population in order to obtain an overall perception of some aspect of interest such as the average execution time of some particular process. In a nutshell, Aggregation Metrics are defined in terms of an aggregation construct that is applied over the population obtained by applying a *Population Filter*. A Population Filter is defined intensionally as a kind of Query that has only one kind of output type (see Section 2.1). Thus Population Filters simply capture a Query that filters a particular kind of individuals that meet certain criteria. The capacity for capturing Queries ontologically thus plays here a very important role, allowing to completely specify and compute Metrics even when the population to be analysed varies over time. For instance the definition of the metric for computing the average execution time of process instances remains unchanged even when new process instances are created over time. Aggregation constructs are unary functions that take a list as input and return a Constant Quantity as a result. Thus, most of the statistical functions previously introduced (e.g., count, min, max, mean, etc) are indeed valid aggregation functions.

Additionally, in order to provide a convenient way for business analysts to define metrics, Metrics Ontology allows Aggregation Metrics to have a nested definition whereby, a given Function Metric will be evaluated over each individual of the population prior to the computation of the aggregation function. The reason for this is that we want to support defining metrics such as the "average execution time of process X", or even the "maximum of the average cost per process". To do so Aggregation Metrics include the *hasFunctionMetric* and *hasUnboundRole* slots. The former indicates the Function Metric that will be applied whereas the latter determines which slot of the Function Metric should be bound to each individual of the population being analysed. Indeed, the output type of the Population Filter should coincide with the type of the unbound role of the given Function Metric. Metrics Ontology includes several kinds of Aggregation Metrics defined on the basis of the implemented statistical functions we introduced in Section 2.2. The current version of the ontology includes *Count*, *Maximum*, *Minimum*, *Average*, *Standard Deviation*, *Variance*, *Sum*, and *Prod* that specify their corresponding aggregation construct.

Within Business Process Analysis, the concept of Key Performance Indicator and its relevance is well-known. It can therefore come as a surprise that it is not a central concept within Metrics Ontology. The reason for this is simply the

[3]Note that OCML is operational and thus provides means for including procedural definitions within the ontologies.

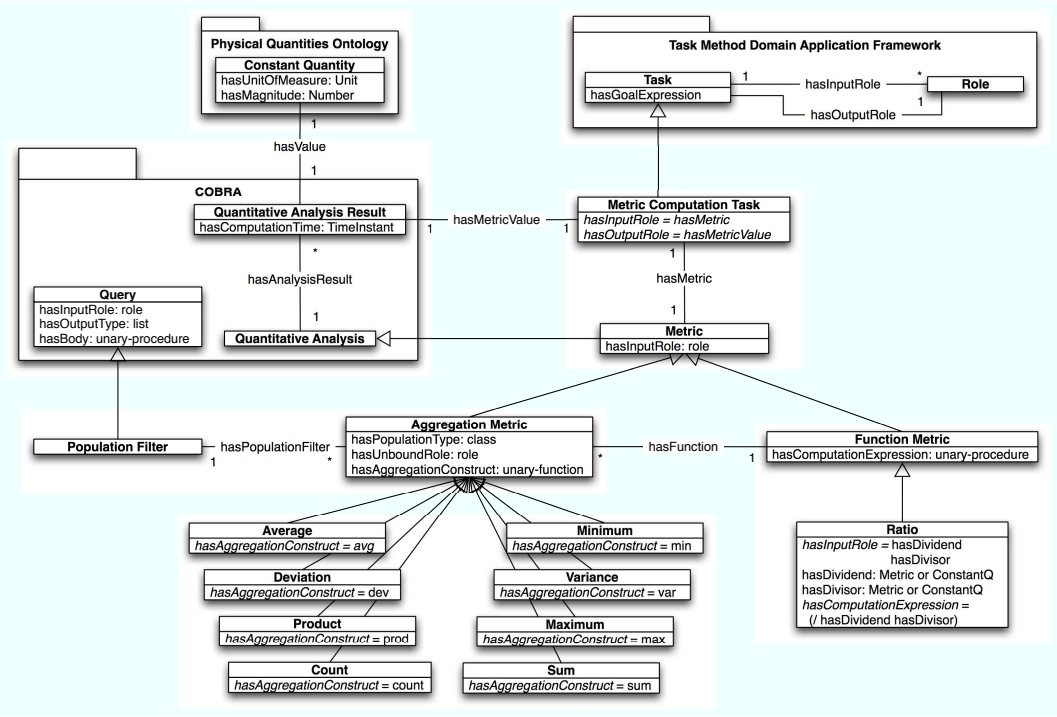

Figure 2: Metrics Ontology.

characteristic contextual nature of KPIs: what can be considered as a KPI within a particular domain might not be as relevant for another. For instance, the smallest deviation with respect to the agreed deadline is not as relevant for a software company as it is within the logistics domain. Furthermore, even within the same domain, different departments, business processes, or even particular process instances might be driven by diverse KPIs. A simple example can be the so-called Return of Investment which is certainly crucial for most businesses but can hardly be used to measure the success of the research department within a company. Due to this context dependence we have opted to capture the concept KPI using the *Role* design pattern included in COBRA, see [14]. In this way we can model that a given Metric *plays the role* KPI within a particular Business Activity or Business Activity Realisation.

3. GENERIC METRICS COMPUTATION ENGINE

In this section, we describe our Generic Metrics Computation Engine that takes Metric definitions as input, interprets them, and computes their value taking into account the overall body of knowledge about process executions obtained from audit trails and captured in terms of COBRA.

3.1 Task Method Domain Application

Our approach is based on previous research on Problem-Solving Methods [16, 12]. In particular, we build upon the Task Method Domain Application (TMDA) framework [12] for Knowledge-Based Systems reuse and development. In a nutshell, TMDA prescribes constructing Knowledge-Based Systems based on the definition of task ontologies that define classes of applications (e.g., diagnosis, classification), method ontologies that capture the knowledge requirements for specific methods (e.g., heuristic classification), domain ontologies which provide reusable task-independent models, and application ontologies for integrating domain models with domain-independent problem solvers.

TMDA has been applied to the construction of systems that tackle diverse knowledge-intensive tasks (e.g., parametric design, planning). Despite the relative simplicity of the metrics computation endeavour with respect to more complex knowledge-intensive tasks [12, 16], this approach gives us the appropriate genericity and support for interchanging methods as the need arises. Furthermore, this represents a step towards the creation of library of tasks for Business Process Analysis and their corresponding methods which we plan to base on previous research in Problem-Solving Methods such as diagnosis, classification and configuration design [2].

3.2 Metric Computation Task

Metric computation is defined within the TMDA framework as a kind of task that takes a Metric as input and returns a Quantitative Analysis Result with the actual value for the Metric at that particular point in time, see Figure 2. Metric Computation task is decomposed as illustrated in Figure 3, whereby white boxes represent tasks and grey ones represent methods. Compute Metric is the top-level task for metrics computation and it is currently delegated to the sub-tasks Compute Aggregation Metric or Compute Function Metric depending on the kind of Metric being calculated.

Function Metrics computation is performed in a two-steps process. First, the input roles of the Function Metric are evaluated, and then the Function Metric is evaluated. In

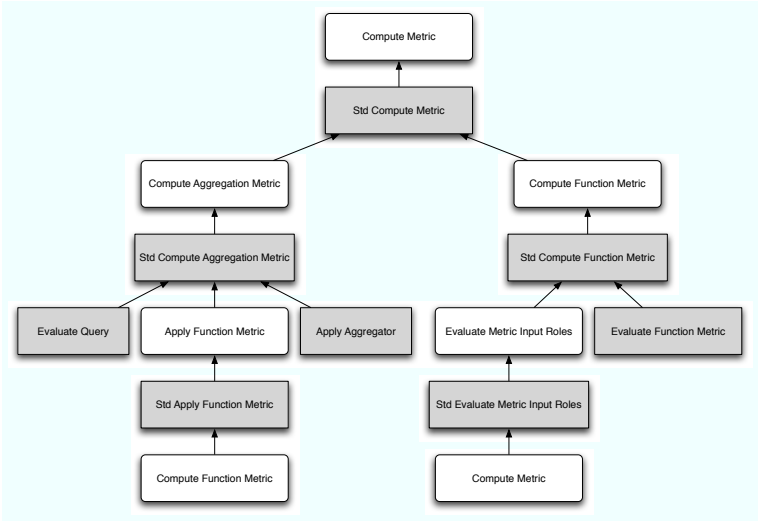

Figure 3: Task-method decomposition for metrics computation.

this way our framework supports nested definitions of Function Metrics, which provides important benefits. On the one hand, the business analyst is provided with more convenient means for defining metrics by reusing pre-existing ones. For instance, the "ratio success-failure for a Business Activity" can just be defined on a one-off basis using two other metrics for computing respectively the "number of failures" and the "number of successes" for the given Business Activity. On the other hand, this provides the required level of tracking over metric values. Retaking the previous example, for computing the ratio, using concrete values for the input roles (e.g., 40 and 7) would hide important information concerning the origin for the role values (i.e., "number of successes" and "number of failures" respectively).

Our metrics computation engine currently provides an eager implementation for the Compute Function Metric task which, independently from whether the metric value might have changed or not, always calculates it. It would, however, be possible to include mechanisms for avoiding the computation of metrics which are known unchanged (e.g., execution time of a finished process) and we could derive new values based on previous computations and newly received information which could be an important performance improvement for computing average values for example. In this respect, the TMDA framework provides us with the necessary support for seamlessly including new methods, and using them as the need arises.

As we previously introduced, Aggregation Metrics are basically Metrics which are computed by applying an aggregation construct such as *average*, over a particular population. Aggregation Metrics computation, as illustrated in Figure 3, is a three-steps process. First, the population is selected by evaluating the Population Filter part of the specification of the Aggregation Metric. Secondly, Aggregation Metrics computation supports, as previously introduced, the automated computation of a Function Metric over each individual prior to applying the aggregation construct. This second step is achieved by taking every individual obtained after evaluating the Population Filter, binding it to the unbound role specified by the Aggregation Metric, and finally computing the Function Metric in the same way as explained previously. Finally, the aggregation construct, e.g., average, min, max, is applied over the resulting population and the corresponding Quantitative Analysis Result is returned.

4. GENERIC BUSINESS METRICS AND QUERIES

In this section we present specific metric definitions that give concrete details to the reader and fulfil two important roles. First, they allow us to evaluate our framework from a representation and computation perspective, so that we can better assess the appropriateness and correctness of our approach. Secondly, the metrics that we present here are themselves important contributions to our framework since they are generally applicable.

Metrics Ontology contemplates two kinds of Metrics. While there exist many Aggregation Metrics that can be defined in a domain-independent manner, the typical Function Metrics one could think of such as "Process Instance cost" and "Return of Investment" are most often domain dependent. We therefore include here mainly Aggregation Metrics, although some generic Function Metrics could also be defined. In particular, in Generic Business Metrics ontology we have captured more than 40 Metric definitions, out of which 4 are Function Metrics. As previously explained, Aggregation Metrics definitions depend on Population Filters. The Aggregation Metrics we have captured in Generic Business Metrics ontology are therefore specified using an extensive set of Population Filters. These approximately 50 Population Filters have been captured within Generic Business Queries ontology so that they can be seamlessly reused by business analysts for appropriately filtering the information. Among the Population Filters defined, we have queries for obtaining Business Activity Realisations based on their current execution state (e.g., Completed, Running); based on their execution state at some particular point on time; based on the Business Activity they perform, etc.

Listing 1 Function and Aggregation Metric Example.

```
(def-class #_BARExecutionTime (#_metrics:FunctionMetric)
  ((has-input-role :value #_hasBusinessActivityRealisation)
   (#_hasBusinessActivityRealisation :type
    #_cobra:BusinessActivityRealisation)
   (#_metrics:hasComputationExpression :value
     '(lambda (?fm)
       (in-environment
        ((?bar .
         (the-slot-value ?fm
                         #_hasBusinessActivityRealisation))
         (?result . (#_time:duration ?bar))
           ?result))))))

(def-instance #_DefaultBARExecTimeInst #_BARExecutionTime)

(def-instance #_AvgClosedBARsExecutionTime #_metrics:Average
  ((#_metrics:hasFunctionMetric #_DefaultBARExecTimeInst)
   (#_metrics:hasUnboundRole #_hasBusinessActivityRealisation)
   (#_metrics:hasPopulationFilter #_gbq:whichBARsClosed)))
```

On the basis of the Population Filters represented in Generic Business Queries ontology, we have defined a set of Aggregation Metrics, that count the number of individuals that share some criteria, like being a Process Instance currently Running. The current version of Generic Business Metrics, includes Aggregation Metrics for counting all the Business Activity Realisations in some particular state, the Process Instances performing a particular Process, etc. Additionally, we have included a set of Aggregation Metrics which are based on more advanced features of our framework. We have defined a few Aggregation Metrics that make use of the capacity for computing Function Metrics automatically for the individuals of the population. In particular we have defined the average, maximum, minimum "execution time for Business Activity Realisations".

Listing 1 shows the definition of a Function Metric for computing the execution time of Business Activity Realisations, and an Aggregation Metric for computing the average of the execution time for all the Business Activity Realisations that are closed (i.e., those that have finished)[4]. The Function Metric takes as input a Business Activity Realisation and, based on Time Ontology [14], computes the duration. The Aggregation Metric on the other hand computes the average of the execution time for each of the existing Business Activity Realisations that are closed. The reader is referred to the ontologies for further examples and details.

5. RELATED WORK

Significant efforts have been devoted to integrating automated reasoning within the BPM domain, see for instance [20, 6, 4, 7, 9, 19, 3]. Among these approaches we find research on enterprise ontologies, value flows, etc. Indeed BPA is no exception in this respect and it has been claimed that the application of knowledge-based technologies is the next evolutionary step [22]. Some have focussed on the definition an implementation of distributed architectures for Business Process Monitoring [3, 19], whereas others have enhanced existing analysis techniques with lightweight semantics [4, 7]. However, despite the efforts devoted to enhancing BPA with semantics, the degree of automation currently achieved is largely insufficient. With respect to metrics formalisation, [11] is perhaps of most relevance to us, however their measurement ontology is focussed on providing a foundational basis for domain-specific measurement ontologies rather than on the computation itself.

Our approach to BPA [2, 14] is based on an extensive conceptualisation of the BPM domain spanning from low-level monitoring details to high-level business aspects so as to bring this vital information to the business-level as required by business practitioners. The research presented in this paper aims at automating BPA to a greater extent by extending our existing framework to support the seamless definition and computation of metrics. In this respect, our work is, to the best of our knowledge, the most complete generic framework for the definition and computation of business metrics in an automated manner. Furthermore, we do so completely based on semantic technologies thus providing a solid basis for the development of more advanced knowledge-based techniques for BPA.

6. CONCLUSIONS AND FUTURE WORK

Business Process Analysis is a branch of Business Process Management which is concerned with analysing deployed processes in order to assess their well-being, identify potential improvements and even ensure that these processes meet certain criteria. Crucial within BPA, is the computation and analysis of metrics concerning business activities, departments, resources, etc. These metrics include punctual details like the "cost of a particular process instance", statistical aspects like the "average execution time of a process", or even the detection of trends and tendencies over time, like the "evolution of customer satisfaction in the last year".

A fully-fledged framework for BPA must therefore support defining and computing metrics in a seamless manner. We have defined Metrics Ontology, a domain independent ontology that supports the seamless definition of business met-

[4]'#_metrics:' is an abbreviation for the namespace of Metrics Ontology, whereas '#_' identifies the default namespace.

rics, and the corresponding engine which can interpret and automatically compute these metrics over domain-specific data. Finally, we have defined two further extensions to our framework, i.e., Generic Business Metrics and Generic Business Queries ontologies, which help us validate the genericity and correctness of our approach while, at the same time, they provide us with a set of reusable definitions thus contributing to our BPA framework.

Despite the expressivity and relative simplicity provided by our framework for defining metrics, we have identified the need for supporting users in the definition of queries. In fact, although we have seen that defining metric computation expressions is not particularly complex or different from current practices within custom-tailored software, defining ontological queries seems to be beyond the tasks a business practitioner would undertake. Future work will therefore be devoted to simplifying the definition of queries both by means of our conceptual model for specifying queries and by means of an ontology-based user interface as part of SENTINEL [15]. In more general terms, our future work will pursue the vision previously outlined in [2] towards the greater automation of BPA using knowledge-based techniques.

7. ACKNOWLEDGMENTS

This work was funded by the European projects SUPER (FP6-026850) and SOA4All (FP7-215219). We are also grateful to Alessio Gugliotta for his valuable feedback.

8. REFERENCES

[1] 11th Conférence Générale des Poids et Mesures. Système International d'Unités. http://www.bipm.org/en/CGPM/db/11/12/, 1960.

[2] A. K. Alves de Medeiros, C. Pedrinaci, W. van der Aalst, J. Domingue, M. Song, A. Rozinat, B. Norton, and L. Cabral. An Outlook on Semantic Business Process Mining and Monitoring. In *Proceedings of International IFIP Workshop On Semantic Web & Web Semantics (SWWS 2007)*, 2007.

[3] B. Azvine, Z. Cui, D. D. Nauck, and B. Majeed. Real Time Business Intelligence for the Adaptive Enterprise. In *The 8th IEEE International Conference on E-Commerce Technology*, Los Alamitos, CA, USA, 2006. IEEE Computer Society.

[4] M. Castellanos, F. Casati, U. Dayal, and M.-C. Shan. A comprehensive and automated approach to intelligent business processes execution analysis. *Distributed and Parallel Databases*, 16(3):239–273, 2004.

[5] S. Chaudhuri, U. Dayal, and V. Ganti. Database Technology for Decision Support Systems. *Computer*, 34(12):48–55, December 2001.

[6] J. Gordijn and H. Akkermans. Designing and evaluating e-business models. *IEEE Intelligent Systems*, 16(4):11–17, 2001.

[7] D. Grigori, F. Casati, M. Castellanos, U. Dayal, M. Sayal, and M.-C. Shan. Business Process Intelligence. *Computers in Industry*, 53(3):321–343, Apr. 2004.

[8] T. R. Gruber and G. R. Olsen. An Ontology for Engineering Mathematics. In J. Doyle, P. Torasso, and E. Sandewall, editors, *Fourth International Conference on Principles of Knowledge Representation and Reasoning*, pages 258–269, Bonn, Germany, 1994. Morgan Kaufmann.

[9] M. Hepp, F. Leymann, J. Domingue, A. Wahler, and D. Fensel. Semantic Business Process Management: A Vision Towards Using Semantic Web Services for Business Process Management. In F. C. M. Lau, H. Lei, X. Meng, and M. Wang, editors, *ICEBE*, pages 535–540. IEEE Computer Society, 2005.

[10] R. S. Kaplan and D. P. Norton. The Balanced Scorecard - Measures that Drive Performance. *Harvard Business Review*, January/February 1992.

[11] H. M. Kim, A. Sengupta, M. S. Fox, and M. Dalkilic. A measurement ontology generalizable for emerging domain applications on the semantic web. *Journal of Database Management*, 18(1):20–42, January-March 2007.

[12] E. Motta. *Reusable Components for Knowledge Modelling. Case Studies in Parametric Design Problem Solving*, volume 53 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 1999.

[13] C. Pedrinaci, C. Brelage, T. van Lessen, J. Domingue, D. Karastoyanova, and F. Leymann. Semantic business process management: Scaling up the management of business processes. In *Proceedings of the 2nd IEEE International Conference on Semantic Computing (ICSC) 2008*, Santa Clara, CA, USA, Aug. 2008. IEEE Computer Society.

[14] C. Pedrinaci, J. Domingue, and A. K. Alves de Medeiros. A Core Ontology for Business Process Analysis. In *5th European Semantic Web Conference*, 2008.

[15] C. Pedrinaci, D. Lambert, B. Wetzstein, T. van Lessen, L. Cekov, and M. Dimitrov. SENTINEL: A Semantic Business Process Monitoring Tool. In *Ontology-supported Business Intelligence (OBI2008) at 7th International Semantic Web Conference (ISWC2008)*, Karlsruhe, Germany, 2008.

[16] G. Schreiber, H. Akkermans, A. Anjewierden, R. de Hoog, N. Shadbolt, W. V. de Velde, and B. Wielinga. *Knowledge Engineering and Management: The CommonKADS Methodology*. MIT Press, 1999.

[17] R. Studer, R. Benjamins, and D. Fensel. Knowledge Engineering: Principles and Methods. *Data Knowledge Engineering*, 25(1-2):161–197, 1998.

[18] Supply-Chain Council. Supply-Chain Operations Reference-model. http://www.supply-chain.org, 2008.

[19] M. Thomas, R. Redmond, V. Yoon, and R. Singh. A Semantic Approach to Monitor Business Process Performance. *Communications of the ACM*, 48(12):55–59, 2005.

[20] M. Uschold, M. King, S. Moralee, and Y. Zorgios. The Enterprise Ontology. *Knowledge Engineering Review*, 13(1):31–89, 1998.

[21] W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske. Business Process Management: A Survey. In W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske, editors, *Business Process Management*, volume 2678 of *LNCS*, pages 1–12. Springer, 2003.

[22] H. J. Watson and B. H. Wixom. The Current State of Business Intelligence. *Computer*, 40(9):96–99, 2007.