

A Core Ontology for Business Process Analysis

Carlos Pedrinaci¹, John Domingue¹, and Ana Karla Alves de Medeiros²

¹ Knowledge Media Institute, The Open University, Milton Keynes, MK7 6AA, UK.
E-mails: {c.pedrinaci, j.b.domingue}@open.ac.uk

² Eindhoven University of Technology, P.O. Box 513, 5600MB, Eindhoven,
The Netherlands. E-mail: a.k.medeiros@tue.nl

Abstract. Business Process Management (BPM) aims at supporting the whole life-cycle necessary to deploy and maintain business processes in organisations. An important step of the BPM life-cycle is the analysis of the processes deployed in companies. However, the degree of automation currently achieved cannot support the level of adaptation required by businesses. Initial steps have been performed towards including some sort of automated reasoning within Business Process Analysis (BPA) but this is typically limited to using taxonomies. We present a core ontology aimed at enhancing the state of the art in BPA. The ontology builds upon a Time Ontology and is structured around the process, resource, and object perspectives as typically adopted when analysing business processes. The ontology has been extended and validated by means of an Events Ontology and an Events Analysis Ontology aimed at capturing the audit trails generated by Process-Aware Information Systems and deriving additional knowledge.

1 Introduction

Many companies use information systems to support the execution of their business processes. Examples of such information systems are Enterprise Resource Planning, Customer Relationship Management, and Workflow Management Systems (WFMS). These systems usually generate events while executing business processes [1] and these events are recorded in logs. The competitive world we live in requires companies to adapt their processes in a faster pace. Therefore, continuous and insightful feedback on how business processes are executed becomes essential. Additionally, laws like the Sarbanes-Oxley Act force companies to show their compliance to standards. In short, there is a need for good analysis tools that can provide feedback about how business processes are actually being executed based on the observed (or registered) behaviour in event logs.

BPM results from the limitations exhibited by WFMS which mainly focus on the enactment of processes by generic engines and does not take into account the continuous adaptation and enhancement of existing processes. BPM acknowledges and aims to support the complete life-cycle of business processes which undoubtedly involves post-execution analysis and reengineering of process models. A key aspect for maintaining systems and the processes they support

is the capability to analyse them. BPA is particularly concerned with the behavioural properties of enacted processes may it be at runtime, as in Business Process Monitoring, or post-execution as in Business Process Mining [1] or Reverse Business Engineering.

Due to its cyclic nature, BPM has however made more evident the existing difficulties for obtaining automated solutions from high-level business models, and for analysing the execution of processes from both a technical and a business perspective. The fundamental problem is that moving between the business-level and the IT-level is hardly automated [2]. Deriving an IT implementation from a business model is particularly challenging and requires an important and ephemeral human effort which is expensive and prone to errors. Conversely analysing automated processes from a business perspective, e.g., calculating the economical impact of a process or determining the performance of different departments in an organisation, is again an expensive and difficult procedure which typically requires a human in the loop. *Semantic Business Process Management* (SBPM), that is the combination of Semantic Web and Semantic Web Services technologies with BPM, has been proposed as a solution [2].

In this paper we present results obtained in the context of the European project SUPER (IST-026850) which aims at developing a SBPM framework, based on Semantic Web Services technology, that acquires, organises, shares and uses the knowledge embedded in business processes in order to make companies more adaptive. This semantic framework will support the four phases of the BPM life-cycle and the research presented in this paper provides the foundation for semantic BPA. *In particular we shall describe a core ontology for business process analysis which bridges the gap between low-level monitoring information and high-level business knowledge.* The remainder of the paper is organised as follows. Section 2 reviews existing research that makes use of semantic technologies and present a set of requirements and competency questions that semantic BPA technologies should address. Section 3 presents COBRA, a Core Ontology for Business pRocess Analysis, and Time Ontology which provides the basis for using temporal reasoning within BPA. Section 4 illustrates how COBRA can be applied to BPA. Section 5 presents our conclusions and describes future research to be carried out.

2 Semantics in Business Process Management

In the last years significant efforts have been devoted to integrating automated reasoning with the BPM domain, a field where the application of knowledge-based technologies appears to be the next evolutionary step [3]. These efforts can roughly be divided into top-down and bottom-up approaches. Top-down approaches make use of high-level conceptual models to structure and reason about Business Process Management activities. Among these approaches we find research on enterprise ontologies, models for resources consumption and provision, value flows, service bundling, etc. [2, 4–8]. However, despite the variety of models and tools produced so far there is little uptake within the industry which

is often due to the existing difficulty to provide and maintain good knowledge bases expressed in terms of these conceptual models.

On the other hand, bottom-up approaches integrate some sort of light-weight automated reasoning machinery with existing BPM solutions, see for instance [9–11]. These efforts are mainly dominated by researchers from the BPM area, where knowledge-based technologies have not been widely used so far. The focus has mainly been the annotation of data warehouses or the application of rule engines to control resources and ensure certain business policies are followed. Unfortunately, the information manipulated is mostly in syntactic formats which is hardly amenable to automated reasoning. In fact, most of the budget when applying so-called Business Intelligence solutions is typically devoted to the manual integration of data from BPM systems and this is often an ephemeral effort which has to be repeated over time. As a result the benefits gained by applying these techniques are largely limited. Still, as opposed to top-down approaches, the fact that these research efforts are grounded into deployed BPM systems increases their impact in the industry.

What can be distilled from the current state-of-the-art is that the existent epistemological gap between, on the one hand industry BPM solutions, and on the other hand knowledge-based research, hampers to an important extent the wider application of semantics in BPM. The research presented in this paper aims precisely at reducing this gap when it comes to analysing business process executions. In order to guide and validate our approach we present next a representative set of requirements and competency questions that we have identified based on existing practice within the BPM domain.

2.1 Requirements for Semantic Business Process Analysis

BPA is typically structured around three different views: (i) the *process view*; (ii) the *resource view*; and (iii) the *object view* [12]. The process view is concerned with the enactment of processes and is thus mainly focussed on the compliance of executed processes with respect to prescribed behaviours and Key Performance Indicators that can support business analysts in the examination and eventual optimisation of deployed processes [1]. Relevant information in this respect are (i) “the processes and activities currently running”; (ii) “which ones have been completed and whether they were successful or not”; (iii) “the execution time of the different business activities”; (iv) “which business activities have preceded which others”, etc. The resource view is centred around the usage of resources within processes. In this perspective, the performance at different levels of granularity (individuals, organisational units, etc.), work distribution among the resources, and the optimisation of resources usage are the main aspects analysed. Typical questions in this perspective are for instance (i) “which resources were involved in which business activities”; (ii) “which actor was responsible for a certain process”; (iii) “which external providers appear to work more efficiently”; (iv) “what’s the average number of orders processed by the sales department per month”, etc. Finally, the object view focusses on business objects such as inquiries, orders or claims. This perspective is often adopted in

order to better analyse the life-cycle of so-called Business Objects. In this perspective, business analysts often want answers to questions like (i) “what is the average cost per claim”; (ii) “which is the item we are currently selling the most (or the least)”; (iii) “what’s the overall benefit we are obtaining per item”; (iv) “are critical orders processed in less than two hours”, etc.

These three views are populated with statistical information such as the minimum, the average or the deviation of some parameter of interest, and correlations are typically established across them, e.g., “what is the average process execution time for processing each type of order?”. Common to these scenarios where BPA techniques are applied is the underlying dependency with respect to time manipulation (e.g., “are critical orders processed in less than two hours”), the need to navigate through different levels of abstraction (e.g., “what’s the average number of orders processed by the sales department per month”) and across the different perspectives, and the overall necessity to apply general purpose methods over domain specific data.

Therefore, to enhance the state-of-the-art of BPA we need a comprehensive conceptual model of the BPM domain that supports applying general purpose knowledge-based techniques over domain specific data, coupled with the capacity to navigate through different levels of abstraction across the process, resource, and object perspectives, and the ability to appropriately deal with temporal aspects. The next section is devoted to presenting a core ontology for supporting Business Process Analysis that aims to provide a generic and extensible conceptual model that can support the competency questions exposed above.

3 An Ontology for Business Process Analysis

In order to support the level of automation required by enterprises nowadays we need to enhance BPA with support for applying general purpose analysis techniques over specific domains in a way that allows analysts to use their particular terminology and existing knowledge about their domain. To this end we have defined the Core Ontology for Business pRocess Analysis. COBRA provides a core terminology for supporting BPA where analysts can map knowledge about some particular domain of interest in order to carry out their analyses. It is worth noting that COBRA does not aim to provide a fully-comprehensive conceptualisation for supporting each and every kind of analysis since the scope would simply be too big to be tackled appropriately in one ontology. Instead 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. These extensions are currently been developed in SUPER as part of an ontological framework aimed at providing an extensive conceptualisation of the BPM domain ranging from process modelling to the definition of business strategies. Still, COBRA already provides a good basis for supporting the most typical analysis as described in the previous section.

COBRA has been developed using the Operational Conceptual Modelling Language (OCML) [13], which provides support for executing the definitions in

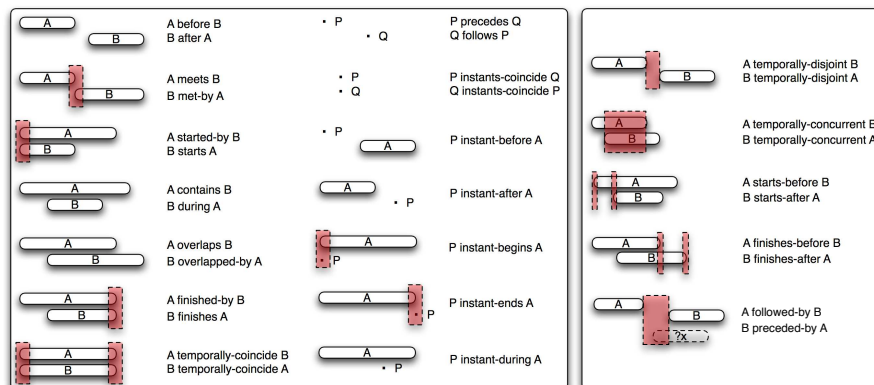


Fig. 1. Instants and Interval relations.

the ontology as well as export mechanisms to other representations including OWL and WSM. COBRA builds upon two ontologies, namely Base Ontology and Time Ontology, and is currently enhanced with Events Ontology for capturing audit trails, and Events Analysis Ontology which provides a set of generic reusable rules and relations³. Base Ontology provides the definitions for basic modelling concepts such as tasks, relations, functions, roles, numbers, etc. The interested reader is referred to [13] for further information. The other ontologies will be described in the remainder of this section.

3.1 Time Ontology

COBRA builds upon Time Ontology that provides a temporal reference by means of which one can determine temporal relations between elements. The ontology defines three top-level concepts, namely *Time Instant*, *Time Interval*, and *Temporal Entity*. Time Instant is the main primitive element and it provides the means for identifying a point in time with precision up to the microsecond for we aim to support monitoring automated systems. Time Intervals are defined by means of the start and end instants and have therefore an associated duration which can be computed by means of a function that subtracts the limiting instants. Temporal Entity, as opposed to the conceptualisation proposed in [14], represents entities that have a temporal occurrence, and are therefore different from Time Instant and Time Interval which are the base constructs that represent a particular point or period in time.

Using these core concepts we have implemented the interval relations defined by Allen [15], the additional instant-interval relations defined by Vilain [16], and useful functions for computing the duration of intervals or for obtaining the current Time Instant. The left hand-side of Figure 1 illustrates these relations, whereby *A* and *B* represent Time Intervals, whereas *P* and *Q* represent Time Instants. The relations are self-explanatory, the interested reader is referred to [15] and [16] for further details. It is worth noting that we have renamed the equality

³ The ontologies can be found at <http://kmi.open.ac.uk/people/carlos>

relations for Time Intervals and Time Instants to *Temporally Coincide* and *Instants Coincide* respectively, for we believe it is counterintuitive to use the term “equal” for referring to different things that occur at the same time.

In addition to these relations we have also included for convenience a few typical disjunctions of Allen’s algebra, e.g., *Before-Or-Meets*, and further relations which are relevant for BPA. The latter are depicted in the right-hand side of Figure 1. The new relations we have implemented are namely *Temporally Disjoint*, *Temporally Concurrent*, *Starts-Before*, *Starts-After*, *Finishes-Before*, *Finishes-After*. Two intervals are considered to be Temporally Disjoint if there is no interval shared between the two, which in Allen’s interval algebra is equivalent to a disjunction between Before, After, Meets and Met-By. Temporally Concurrent is the inverse relation of Temporally Disjoint and it therefore holds when there exists some interval shared between the two concurrent intervals. Starts-Before, Starts-After, Finishes-Before and Finishes-After, which we believe are self-explanatory, are based on the numerical comparison between the start instant or end instant of the intervals.

Our Time Ontology considers two kinds of *Temporal Entities*, namely *Instantaneous Entity* and *Time Spanning Entity*. Instantaneous Entities are phenomena that occur at a specific point on time and whose duration can be neglected. By contrast, Time Spanning Entities are those that last over a period of time indicated by the *spansInterval* slot. The distinction between, on the one hand, Temporal Entities and, on the other hand, Time Instant and Time Interval allows us to apply the previous relations over a plethora of entities, i.e., every Temporal Entity. In addition to the previously mentioned relations we have included two which are specific to Time Spanning entities and are particularly useful for BPA, namely *Followed-By* and *Preceded-By*. A Time Spanning Entity *I* is *Followed-By* by another Time Spanning Entity *J* of kind *C*, if *J* is After *I* and there is no other Time Spanning Entity *X* of kind *C* which is After *I* and Starts-Before *J*. *Preceded-By* is the inverse relation of *Followed-By*.

One of the main characteristics of our Time Ontology is the use of *polymorphism*. Our ontology supports determining temporal relations about the primitive elements Time Instant and Time Interval, between these and Temporal Entities, and between any two Temporal Entities. To do so, the relations have been implemented on the basis of backward-chaining rules that perform the appropriate transformations between Temporal Entities and their primitive counterpart and then invoke the primitive relation. This polymorphism is a convenient feature of our conceptualisation in order to support BPA where temporal relations need to be evaluated between executions of activities, e.g., “was Activity A executed after Activity B?”, executions of processes, e.g., “has Process A been run concurrently with Process B”, but also with respect to reference intervals or instants, e.g., “retrieve all the Processes executed in the last month”.

3.2 Core Ontology for Business Process Analysis

We previously introduced that BPA is concerned with the analysis of the execution of business processes from several perspectives. In particular, we identified

the *process view*, the *resource view*, and the *object view*. COBRA has therefore been structured around these very views in an attempt to enhance BPA with support for the automated reasoning, querying, and browsing of audit trails from different perspectives and at different levels of abstraction. The ontology is depicted in Figure 2 using an extended UML notation where arrows represent the *isA* relation, dashed arrows denote the *instanceOf* relation, and lines represent custom relations. Further notation extensions will be explained as the need arises during the description of the ontology.

The development of COBRA has been guided to an important extent by existing ontologies like the Enterprise Ontology [5], DOLCE [17], TOVE [4] and CIDOC [18]. COBRA distinguishes between Temporal Entities (see Section 3.1) and *Persistent Entities* which are disjoint. This terminology is borrowed from CIDOC [18] but is conceptually inline with DOLCE [17], whereby Endurant corresponds to Persistent Entity and Perdurant to Temporal Entity. In short, Temporal Entities are entities that have a temporal extent whereas Persistent Entities are essentially independent of time. COBRA uses this high-level categorisation as a foundational basis but it doesn't go however much further in the reuse of existing foundational ontologies for it aims at supporting analysis of processes and a complete grounding into this kind of ontologies would carry an important computational overhead. Instead, we provide a simple categorisation of Persistent Entities specifically tailored to our needs, though informed by DOLCE, whereby we contemplate Physical and Non-Physical Entities which are disjoint. Physical entities are those that have a mass.

Physical and Non-physical Entities are further refined into *Agentive* and *Non-Agentive*. The distinction between these classes which are obviously disjoint, is that Agentive Entities are those that can take an active part within some specific activity. Finally, we define *Agent* as the union of both Physical and Agentive Non-Physical Entities. We include for reference and self-containment a few concepts widely used within BPM. For instance, we include *Object*, *Person*, *Organisation*, *Software Agent*, and *Role*. The latter will be dealt with in more detail later on. COBRA, for its purpose is to provide core definitions for supporting business analysis, does not refine these classes any further. Instead they serve as placeholders for including additional conceptualisations such as Organisational Ontologies or domain-specific master data. By doing so we aim at reducing the ontological commitment, while we support the seamless integration of further specific conceptualisations. Finally, since sometimes one needs not specify a concrete instance but rather the type, e.g. "you require a computer", we have defined the meta-class Persistent Entity Type such that all the sub-classes of Persistent Entity are instances of Persistent Entity Type. This is depicted in Figure 2 by means of a double-headed arrow.

Core concepts in COBRA are *Business Activity* and *Business Activity Realisation*. A Business Activity is a Non-Agentive Non-Physical Entity (the *isA* relation is not depicted in the figure for the sake of clarity) that represents the specification of any business activity at a high-level where aspects such as the control flow are abstracted away. We contemplate two kinds of Business Activ-

Resource—which are of most use when analysing business processes. Again, Roles categorisation is to be extended for specific domains. Finally, we include the *Role Type* meta-class in order to support describing things like “we require an engineer”. Persistent Entities are further characterised by a set of Role Types they can play within Business Activity Realisations. This allows to model for example that “Directors can play the Role Type Supervisor”.

Given the notion of Role and how these relate to Persistent Entities, we can now fully describe Business Activity and Business Activity Realisation. Business Activities may *use*, *consume*, *produce*, and *provide* a set of Persistent Entity Types. The relationship *uses* may also be defined over specific Persistent Entities, e.g., “this Activity requires this specific machine”, reason why we actually include two relations *usesPersistentEntity* and *usesPersistentEntityType*. Usage, like in the Enterprise Ontology, concerns Persistent Entities that can play a Resource Role, and which are not consumed during the execution of the business activity. In other words, the availability of the Resource will decrease during the execution of the Business Activity and will be restored to the original level at the end. For example, we use a screw-driver for screwing but as soon as we are done the screw-driver is available for others to use. Resource consumption is captured by means of the relationship *consumes*. This relationship is only applicable to those Persistent Entity Types which are not Agents. For situations where some things are required but not used or consumed, we provide the relation *requires*. Business Activities may *require* a set of Persistent Entities (e.g. “a particular document is required”), Persistent Entity Types (e.g. “one license is required to use the software”), and Role Types (e.g. “a coordinator is required”) in order to be performed. The three scenarios are modelled as separate relations. The relationship *produces* captures the outcomes of a Business Activity and is applicable to Persistent Entity Types excepting Non-Agentive Non-Physical Entities for which we have devoted instead the relationship *provides*. These two relationships allow us to capture things like “this production process produces a robot” and “market analysis provides knowledge”. These, excepting the relationship *provides*, are all ternary relationships that can be characterised by the quantity involved, see dashed line in Figure 2.

When it comes to Business Activity Realisations we capture their relation with Persistent Entities in a very similar manner. We do so by means of five relations—*involves*, *uses*, *consumes*, *produces*, and *provides*. Whereby *involves* is a super-relation of the others. We finally provide a ternary relation between Business Activity Realisation, Persistent Entity, and Role which allows us to capture the Role a Persistent Entity *plays in* a Business Activity Realisation (see *playsRoleIn* in Figure 2). Business Activity Realisations are the bridge between the high-level conceptualisation of the BPM domain and the low-level monitoring information captured at runtime by the IT infrastructure. Thus, Business Activity Realisations are further characterised by an *execution history*, a *life-cycle*, and the *current state* of the execution.

The execution history is a set of *Monitoring Events* relevant for monitoring the life-cycle of a Business Activity, see Figure 2. Monitoring Events are

Instantaneous Entities generated by Agents. They are characterised by a reception timestamp which is to be filled by the logging infrastructure upon reception of an event. The main goal of this attribute is to support monitoring even in environments where clock synchronisation mechanisms are hardly applicable. Additionally, Monitoring Events can have a causality vector, i.e., the set of Monitoring Events that caused that particular event. This supports capturing the actual derivation of events by the monitoring infrastructure as necessary for Complex Event Processing. Finally, Monitoring Events might be characterised by additional associated data which is expressed as *Data Value* instances. These instances identify a particular parameter and the value associated to it.

Monitoring Events are further refined into *Message Events* and *Business Activity Monitoring Events*. The former accommodates Event-Based environments so that their execution can also be traced. The latter supports monitoring the life-cycle of Business Activity Realisations in Process-Aware Information Systems. Business Activity Monitoring Events therefore *concern* a specific Process Instance and, depending on the granularity of the event occurred, may also concern an Activity Instance. Similarly to the proposals in [20, 12, 19], Business Activity Monitoring Events are centred around the notion of state model. Every event identifies a particular transition within the state model, the transition being indicated by means of the *leadsToState* attribute. Conversely the *canOccurInState* attribute allows to ensure that the transitions are consistent with the prescribed state model or to detect anomalies within the execution history possibly due to missing events.

COBRA supports the definition of specific state models in a simple ontological form by means of the *Business Activity State* concept which has a set of *possibleNextStates*. Business Activity States are used to further characterise Business Activity Realisations with the *hasLifeCycle* and *hasCurrentState* slots. The former captures the overall life-cycle of Business Activity Realisations as a set of Life-Cycle Periods which are Time Spanning Entities whereby the executed business activity was in a particular state. The latter is a shortcut for avoiding heavy usage of temporal reasoning in order to obtain the current state. On the basis of these Life-Cycle Periods it is possible to revisit the complete life-cycle of a Business Activity Realisation in a suitable manner for interval-based temporal reasoning. Instead of prescribing a particular state model and the corresponding events COBRA remains agnostic from the domain-specific details. Still, we provide an extension, i.e., Events Analysis Ontology, with a set of generic event processing forward-chaining rules that can derive information based Business Activity Monitoring Events. These rules will be detailed in the next section.

Finally, given that COBRA aims to support Business Process Analysis, both Persistent Entities and Business Activity Realisations are characterised by a set of *Analysis Results*. Thus one can capture results of previous analysis for all the relevant perspectives for BPA. Analysis Results are Instantaneous Entities of a *Quantitative* or *Qualitative* nature⁵. Being part of the core ontology for analysing business process, this allows us to reuse results across different types

⁵ Note that we have used slot renaming for *occursAt*

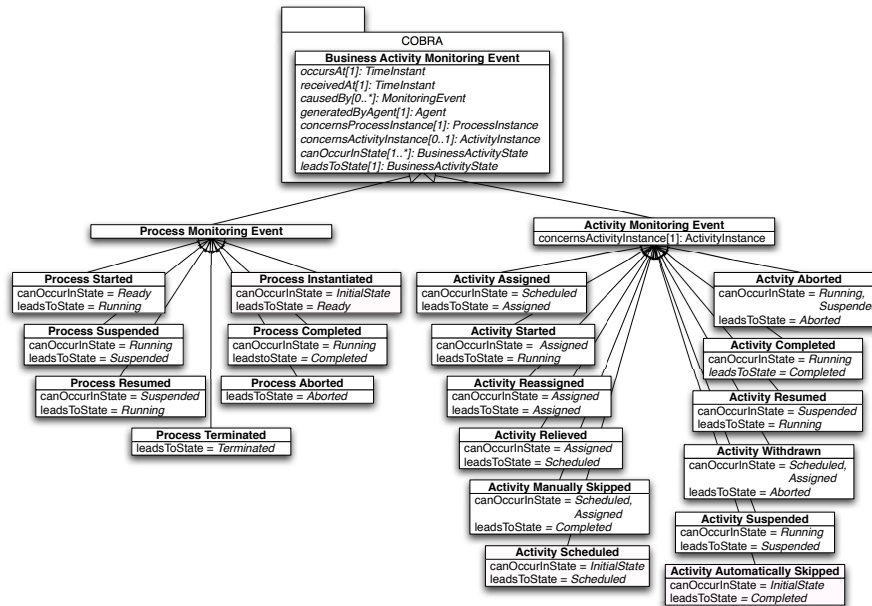


Fig. 3. Events Ontology.

of BPA which paves the way for enhancing current analysis techniques [11]. For instance, metrics computed at runtime can be reused when performing RBE, mining results can be applied during monitoring, etc.

4 Events Processing

COBRA aims at providing a conceptual and extensible framework for supporting BPA. Thus, it purposely leaves many aspects, such as domain-specific data or infrastructure specific monitoring events unspecified. In order to apply COBRA to specific domains these particular details have to be modelled and integrated. As part of the overall BPA conceptual framework but also in order to validate and test our conceptualisation we have developed an ontology for capturing monitoring events from a plethora of BPM systems and a general purpose Events Analysis Ontology that allows to derive information in terms of COBRA from monitoring information. In the remainder of this section we shall describe first the Events Ontology and next the Events Analysis Ontology.

4.1 Events Ontology

BPA takes the audit trails generated by the supporting IT infrastructure as a starting point, and attempts to derive information from the business perspective. Each of the supporting systems provides its own level of detail, in heterogeneous formats making it particularly difficult to integrate the audit trails generated as well as it complicates the creation of general purpose solutions. Common formats have been proposed as a solution to overcome this problem, e.g., MXML [20]



Fig. 4. Events Ontology State Model.

or the Audit Trail Format by the Workflow Management Coalition (WFMC). Although these formats have proven their benefits, they are supported by technologies that are not suitable for automated reasoning. In order to overcome this, we have extended COBRA with a reference Events Ontology (EVO) that provides a set of definitions suitable to a large variety of systems and ready to be integrated within our core ontology for analysing business processes. EVO is however an optional module which can be replaced by other models if required.

EVO is based on the previously mentioned formats since they provide general purpose solutions that have shown to be suitable to capture logs generated by a plethora of systems. As prescribed by COBRA, EVO is centred around a state model that accounts for the status of processes and activities, see Figure 4. The figure shows the different states and possible transitions contemplated for both Process Instances and Activity Instances which we believe are self-explaining. Note that process abortion differs from process termination in that in the former any ongoing activity is allowed to finish [12]. The dark dot represents the initial state, arrows represent transitions, the smaller boxes depict states, and bigger boxes encapsulate (conceptual) families of states. The state model has been captured ontologically as shown in Figure 3, an enhanced with additional relations. For instance it is possible to determine whether an Activity Instance has been allocated—*isAllocated*—which is true for those that are either in state Running, Suspended, or Assigned. It is also possible to determine whether a Business Activity Realisation is active—*isActive*—which is equivalent to Running, or inactive—*isInactive*—which is true for the rest of the states.

The state model does not distinguish between Process Instances and Activity Instance. The reason for this is mainly to simplify some tasks, e.g. monitoring of active Business Activity Realisations. Still, this necessary distinction is preserved within the logs by means of the Business Activity Monitoring Events defined, see Figure 3. EVO includes two subclasses, namely *Process Monitoring Event* and *Activity Monitoring Event*. EVO currently captures seven Process Monitoring Events and twelve Activity Monitoring Events based on the state model in Figure 4. Process Monitoring Events capture the different transitions which are possible for Process Instances. A Process Instance can be *Instantiated*, *Started*, *Suspended*, *Resumed*, *Completed*, *Aborted* and *Terminated*. Activity Monitoring Events, in addition to the typical execution events, contemplate the distribution of work to Agents. Thus, there are events that capture the scheduling of activ-

ities, the *Assignment*, *ReAssignment*, or *Relief* of activities to specific agents. Additionally like MXML, EVO contemplates the possibility for skipping activities either manually or automatically, which lead to a correct completion. Finally, EVO captures the abortion of activities by means of two events *Activity Aborted* and *Activity Withdrawn*. The distinction between the two lays in the fact that only started activities can be aborted.

4.2 Event Analysis Ontology

So far we have focussed on the conceptual models that capture the BPM domain spanning from the low-level details concerning audit trail information, to higher-level aspects such the roles played by agents in certain processes. In this section we focus on how, on the basis of this conceptual model and by capturing monitoring information ontologically, we derive knowledge about the enterprise that can then support business practitioners or even Knowledge-Based Systems in the analysis and eventual decision-making process.

OCML provides support for defining both backward and forward-chaining rules. In order to derive information upon reception of monitoring events we have defined a set of generic forward-chaining rules which are independent from the domain and the specific Monitoring Events defined. The goal is to provide reusable rules which can then be enhanced with domain specific ones to derive a richer knowledge-base. Additionally we have implemented a set of relations which are of most use when analysing processes. Some of these relations have been defined for COBRA in a generic manner, whereas others have been bundled with EVO for they are EVO-specific. The rules currently implemented support (i) deriving and checking the consistency of life-cycle information about Business Activity Realisations; (ii) updating the execution history of Business Activity Realisations; (iii) updating the relations between Process Instances and Activity Instances; (iv) tracking the Agents involved and; (v) updating the Roles played by Actors within Business Activities.

The first set of rules uses Business Activity Monitoring Events to update the current state of activity realisations, generate Life-Cycle Period instances, and contrast the transitions with the given state model. Basically, every event defines the end of a period and the beginning of a new one⁶. In this way, by simple updates over the life-cycle and with temporal reasoning we can support many of the monitoring competency questions previously exposed. To this end we provide a general purpose relation that holds when, given a Business Activity Realisation, a Time Instant, and a Business Activity State, the activity realisation was in the state given at that particular instant.

The second set of rules aim at correctly tracking the execution history for specific Business Activities so that they can later on be used within Business Process Mining algorithms. The third aspect is supported by a rule that tracks the coincidence of Process Instances and Activity instances within the same Business Activity Monitoring Event and derives the appropriate *composedOf*

⁶ The initial state is a special one which is predefined in COBRA

relation. Agents involvement is derived from the *generatedBy* slot in the events. Finally, whenever one of the Actors involved is the only one taking part in a Business Activity Realisation that can play a certain Role Type that was required, we can derive the role this Actor played. This last rule is bundled with EVO since it is necessary to know whether the business activity was completed before deriving this. The interested reader is referred to the ontologies for further details about the rules and relations currently implemented.

5 Conclusions and Future Work

BPM systems aim at supporting the whole life-cycle of business processes. However, BPM has made more evident the current lack of automation that would support a smooth transition between the business world and the IT world. Yet, moving back and forth between these two aspects is a bottleneck that reduces the capability of enterprise to adapt to ever changing business scenarios. As a consequence there is a growing need for integrating semantics within the BPM domain. A crucial branch of BPM where semantics have a clear and direct impact is Business Process Analysis, where in fact so-called Business Intelligence is appearing as a key enabler for increasing value and performance [3]. Important efforts but with limited success have been devoted to integrating semantics within BPA. The reason for this appears to be the fundamental gap between semantics technologies and the ones currently deployed within BPM solutions.

To reduce this gap we have defined COBRA, a core ontology for business process analysis. The research carried has been guided on a set of competency questions extracted from existing needs with the BPA domain. Our approach is based on a conceptualisation that links low-level monitoring details with high-level business aspects so as to bring this vital information to the business-level as required by business practitioners. This conceptual model is based on a Time Ontology and has been enhanced and validated by means of two extensions for logging monitoring information in a semantic manner, and eventually processing this information.

A key requirement underlying our work has been the need to produce a generic yet comprehensive conceptual infrastructure where additional extensions can be seamlessly plugged-in in order to better support BPA techniques. Future work will thus be devoted to extending our work along the vision previously presented in [11]. In particular, next steps will be devoted to the definition of a metrics computation engine that will support the computation of both generic and user defined metrics, and the implementation of a classification Problem-Solving Method for detecting process deviations. In parallel, we are working on an ontology-based user interface to the reasoning infrastructure as part of WSMO Studio (www.wsmostudio.org).

References

1. van der Aalst, W.M.P., Reijers, H.A., Weijters, A.J.M.M., van Dongen, B.F., de Medeiros, A.K.A., Song, M., Verbeek, H.M.W.: Business process mining: An

- industrial application. *Information Systems* **32**(5) (2007) 713–732
2. Hepp, M., Leymann, F., Domingue, J., Wahler, A., Fensel, D.: Semantic business process management: A vision towards using semantic web services for business process management. In Lau, F.C.M., Lei, H., Meng, X., Wang, M., eds.: ICEBE, IEEE Computer Society (2005) 535–540
 3. Watson, H.J., Wixom, B.H.: The current state of business intelligence. *Computer* **40**(9) (2007) 96–99
 4. Fox, M.S.: The tove project towards a common-sense model of the enterprise. In: IEA/AIE '92: Proceedings of the 5th international conference on Industrial and engineering applications of artificial intelligence and expert systems, London, UK, Springer-Verlag (1992) 25–34
 5. Uschold, M., King, M., Moralee, S., Zorgios, Y.: The enterprise ontology. *Knowledge Engineering Review* **13**(1) (1998) 31–89
 6. Geerts, G.L., McCarthy, W.E.: An accounting object infrastructure for knowledge-based enterprise models. *IEEE Intelligent Systems* **14**(4) (1999) 89–94
 7. Gordijn, J., Akkermans, H.: Designing and evaluating e-business models. *IEEE Intelligent Systems* **16**(4) (2001) 11–17
 8. Malone, T.W., Crowston, K., Herman, G.A.: *Organizing Business Knowledge: The MIT Process Handbook*. MIT Press, Cambridge, MA, USA (2003)
 9. Castellanos, M., Casati, F., Dayal, U., Shan, M.C.: A comprehensive and automated approach to intelligent business processes execution analysis. *Distributed and Parallel Databases* **16**(3) (2004) 239–273
 10. Grigori, D., Casati, F., Castellanos, M., Dayal, U., Sayal, M., Shan, M.C.: Business process intelligence. *Computers in Industry* **53**(3) (2004) 321–343
 11. de Medeiros, A.K.A., Pedrinaci, C., van der Aalst, W.M.P., Domingue, J., Song, M., Rozinat, A., Norton, B., Cabral, L.: An Outlook on Semantic Business Process Mining and Monitoring. In: Proceedings of International IFIP Workshop On Semantic Web & Web Semantics (SWWS 2007). (2007)
 12. zur Muehlen, M.: *Workflow-based Process Controlling. Foundation, Design, and Implementation of Workflow-driven Process Information Systems*. Volume 6 of *Advances in Information Systems and Management Science*. Logos, Berlin (2004)
 13. Motta, E.: *Reusable Components for Knowledge Modelling. Case Studies in Parametric Design Problem Solving*. Volume 53 of *Frontiers in Artificial Intelligence and Applications*. IOS Press (1999)
 14. Hobbs, J.R., Pan, F.: Time ontology in owl. Available at <http://www.w3.org/TR/owl-time/> (2006)
 15. Allen, J.F.: Maintaining knowledge about temporal intervals. *Communications of the ACM* **26**(11) (1983) 832–843
 16. Vilain, M.B.: A system for reasoning about time. In: AAI. (1982) 197–201
 17. Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A., Schneider, L.: WonderWeb Deliverable D17. The WonderWeb Library of Foundational Ontologies and the DOLCE ontology. <http://www.loa-cnr.it/Papers/DOLCE2.1-FOL.pdf> (2003)
 18. ICOM/CIDOC CRM Special Interest Group: CIDOC Conceptual Reference Model. http://cidoc.ics.forth.gr/docs/cidoc_crm_version_4.2.2.pdf (2007)
 19. Gangemi, A., Borgo, S., Catenacci, C., Lehmann, J.: Task taxonomies for knowledge content. Technical report, EU 6FP METOKIS Project D07 (2004)
 20. van Dongen, B., de Medeiros, A., Verbeek, H., Weijters, A., van der Aalst, W.: The prom framework: A new era in process mining tool support. In: *Applications and Theory of Petri Nets 2005*. 26th International Conference, ICATPN 2005, Miami, USA, Springer-Verlag (2005) 444–454