Ontology Mapping of Business Process Modeling Based on Formal Temporal Logic

Irfan Chishti
Department of Computing and Information Systems
University of Greenwich
London, UK

Jixin Ma[1], Brian Knight[2]
Department of Computing and Information Systems
University of Greenwich,
London, UK

***Abstract*—A business process is the combination of a set of activities with logical order and dependence, whose objective is to produce a desired goal. Business process modeling (BPM) using knowledge of the available process modeling techniques enables a common understanding and analysis of a business process. Industry and academics use informal and formal techniques respectively to represent business processes (BP), having the main objective to support an organization. Despite both are aiming at BPM, the techniques used are quite different in their semantics. While carrying out literature research, it has been found that there is no general representation of business process modeling is available that is expressive than the commercial modeling tools and techniques. Therefore, it is primarily conceived to provide an ontology mapping of modeling terms of Business Process Modeling Notation (BPMN), Unified Modeling Language (UML) Activity Diagrams (AD) and Event Driven Process Chains (EPC) to temporal logic. Being a formal system, first order logic assists in thorough understanding of process modeling and its application. However, our contribution is to devise a versatile conceptual categorization of modeling terms/constructs and also formalizing them, based on well accepted business notions, such as action, event, process, connector and flow. It is demonstrated that the new categorization of modeling terms mapped to formal temporal logic, provides the expressive power to subsume business process modeling techniques i.e. BPMN, UML AD and EPC.**

Keywords*—Business Process Modeling techniques; Ontology; Temporal Logic; Semantics; Mapping

I. INTRODUCTION

Business Process (BP) is defined [16], [24], referring to a structure set of actions designed to show how work is done, rather than what is done. The actions referred to are usually work elements, producing some component or subcomponent of a complete artefact. Actions are structured according to essential logical time ordering of component production. However, there is still a difference between meanings and use of the terms utilized in different modeling techniques. In the business and management field, processes are described mainly for human to human communication, for decision making in production processes, administrative processes, to understand their impact on the organization. In the technical field, processes are considered as a form of high level programming languages, conceived to achieve a better use of web services (and, more generally, e-services), i.e., they represent an executable form of the application logics, as part of a complex software artefact.

Business process models are created "to understand the key mechanisms of an existing business; to orient the creation of suitable information systems that support the business; to implement improvements in the current business; to show the structure of an innovated business; to experiment new business concepts; and to identify business elements not considered part of the core, which could be delegated to an outside supplier" [25]. Hence the job to describe business processes and modeling them is becoming increasingly complex. Both industry experts and academics in the field of business process re-engineering and business process management have concluded that successful systems start with an understanding of the business processes of an organization.

In logic, initial attempts to describe fundamental structures of our world were made by Aristotle. To reason and represent process, temporal systems/frameworks used different objects to represent temporal ontology i.e. interval and moment. In 19th century, Charles Sander Peirce invented classical first order logic, which provides a powerful instrument for representing any factual information. However, it is necessary to ask why it is useful to express processes using first order logic. The reason is that the current literature does not provide a logical foundation of business process modeling and logical knowledge representations are loosely coupled with artificial intelligence.

Formal representations allow for better analysis of the designs to identify the process improvements that can lead to increased profitability and improved productivity. The primary aim of this paper is therefore empirical study of main business process modeling techniques and tools explicitly aimed at modeling business processes with the intent of providing a mapping of modeling terms to temporal logic. To achieve this, a versatile conceptual categorization of commercial modeling terms is proposed which gathers a wide variety of commercial modeling terms into a three distinguishable categories and subsequently formalized them. This will ease the process of ontology mapping and also combine key advantages of commercial modeling techniques by providing rigorous logical basis which is general and expressive enough. Formal representation of the processes also forms the basis for future automation of tasks which make up a business process.

Nevertheless, other techniques might exist that are used or that might be used for modeling business processes, which are not considered in this paper. Such techniques applicable to processes in general are applicable to business processes in particular. However, whether all business process modeling

tools and techniques are applicable or not to process modeling is beyond the scope of this paper. The focus of this paper is BPM tools and techniques such as BPMN, UML AD and EPC and temporal model for business processes [18]. The findings of this paper provide an ontology that is general and expressive enough to subsume commercially accepted modeling terminology and characterized by the following properties:

- Main terms/constructs corresponding to concepts and modeling notions drawn from the BPM techniques.
- Conceptually categorize them based on their dynamic and static properties.
- Formally define the conceptual categories.
- Represent them graphically using logical structure to show its generality, simplification and expressiveness.

The rest of the paper is organized as follows. Section II presents the related work; section III introduces the main BPM techniques i.e. BPMN, UML AD and EPC modeling terms/constructs, and author's contribution; a conceptual categorization of modeling terms and its formal definitions. Section IV discusses about temporal logic/systems and main terms/constructs from temporal model for business processes [18], while section V author's contribution towards a ontology mapping of modeling terms discussed in section III and IV, followed by graphical representation of them using an example and section VI will provide the conclusion and possible future work.

II. Related Work

Logical modeling of business processes is to facilitate the understanding and development of business process model that supports the organization, and to permit the analysis and re-engineering or improvement of them. Even though it was 1960 when Levitt first mentioned the importance of business processes it was not until the 1990s that processes have acquired a real importance in enterprise design [25]. Davenport [24], Hammer and Champy [15], [16], and Harrington [10] have promoted the new perspective. However, increasing popularity of business process orientation has attracted designers and developers in the industry to develop new methodologies, and modeling tools and techniques to support it. The task of describing and modeling business process has become more complex due to increased availability of different modeling techniques with the lack of a guide that explains and describes the logic and concepts involved. In this section we briefly present the major BPM techniques and languages:

A. Business Process Modeling Notation (BPMN)

BPMN is the most recent standard notation proposed by Object Management Group (OMG) to design business processes. The primary goal of *BPMN* is to provide a notation that is readily understandable by all business users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes. Thus, *BPMN* creates a standardized bridge for the gap between the business process design and process implementation[1].

B. Unified Modeling Language (UML)

UML is an Object Management Group (OMG) standard which provides the specification for a graphical, general purpose, object-oriented modeling language. *UML* Activity diagrams(*AD*) are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency.*UML AD*, are intended to model both computational and organizational processes (i.e. workflows) Activity diagrams show the overall flow of control[2].

C. Event Driven Process Chains (EPC)

EPC is a modeling technique for business processes modeling developed in the 1990's. Event-driven process chains are an important notation to model the domain aspects of business processes. The main focus of this rather informal notation is on representing domain concepts and processes rather than their formal aspects or their technical realization. Event-driven process chains are part of a holistic modeling approach, called the ARIS framework; ARIS stands for Architecture of Integrated Information Systems, and it was developed by August-Wilhelm Scheer [2].

D. ICAM Definition method (IDEF)

IDEF's roots began when the US Air Force, in response to the identification of the need to improve manufacturing operations, established the Integrated Computer-Aided Manufacturing (ICAM) program in the mid-1970s. The requirement to model activities, data, and dynamic (behavioral) elements of the manufacturing operations resulted in the initial selection of the Structured Analysis and Design Technique (SADT). The Integrated Definition for Function Modeling (IDEF) is a family of methods that supports a paradigm capable of addressing the modeling needs of an enterprise and its business areas. Among these techniques we mention:

- IDEF0*:* the function modeling method, and
- *IDEF3:* the process description captures method.

IDEF0 is a method designed to model the decisions, actions, and activities of an organization or a system. It allows activities and important relations between them to be represented in a nontemporal fashion. It does not support the complete specification of a process[3]. *IDEF3* provides a mechanism for collecting and documenting processes, by capturing precedence and causality relations between situations and events. There are two IDEF3 description modes[4]:

- *process flow:* capturing knowledge of "how things work" in an organization, and

[1] OMG. Business Process Model and Notation (BPMN), 2011 http://www.omg.org/spec/BPMN/2.0, formal/2011-01-03.

[2] Unified Modeling Language: Superstructure version 2.1.1. (*http://www.omg.org/docs/formal/07-02-03.pdf*).

[3] IDEF Function Modeling Method. [http://www.idef.com/IDEF0.html]

[4] IDEF Process Description Capture Method. [http://www.idef.com/IDEF3.html]

- object-*state transition network:* summarizing allowable transitions an object may undergo throughout a particular process.

E. Business Process Execution Language for Web Services (BPEL4WS, or BPEL:

BPEL/BPEL4WS is a de-facto standard for implementing processes based on web services. According to BPEL, processes can be described as:

- *Executable processes:* modeling the behavior of a participant in a business interaction, or as
- *Abstract processes:* specifying the mutually visible message exchange among the parties involved in the protocol, without revealing their internal behavior.

To obtain an executable BPEL process, modelers need to specify primitive and structured activities, execution ordering, messages exchanged, and fault and exception handling. Furthermore, a recent proposal, BPEL4People [5], extends BPEL4WS specification to describe scenarios where users are involved in business processes. BPEL is a powerful and a widely adopted standard. Among its major drawbacks are; its inherent complexity, the verbosity of the XML encoding and the lack of a specific graphical representation. Such characteristics make it scarcely accepted by business people.

F. XML Process Definition Language (XPDL)

XPDL is a Workflow management coalition (WfMC) standard for interchanging process models among process definition tools and workflow management systems. It provides the modeling constructs of BPMN and allows a BPMN process to be specified as an XML document. XPDL process models can be run on compliant execution engines, even if has been originally conceived as a process design and interchange format specifically for BPMN. It represents the linear form of the process definition based on BPMN graphics[6].

G. Petri Net or place/transition net

Petri Net is one of several mathematical representations of discrete distributed systems [11]. As a modeling language, it graphically depicts the structure of a distributed system as a directed bipartite graph with annotations. A Petri net consists of *places*, *transitions*, and *directed arcs*, where

- Arcs run between places and transitions,
- Places may contain any number of tokens, and
- Transitions act on input tokens by a process known as *firing*.

Execution of Petri nets is nondeterministic. This means two things: multiple transitions can be enabled at the same time, any one of which can fire, none are *required* to fire — they fire at will, between time 0 and infinity, or not at all. Since firing is nondeterministic, Petri nets are well suited for modeling the concurrent behavior of distributed systems [12].

H. Temporal logic based models/systems

The essential role of time in the modeling of natural processes has given rise in recent years to a body of artificial intelligence research into temporal theory. This research has led to a variety of temporal systems, attempting to capture the primary elements of time. However, as time goes on, the world may change its state from one into another, triggered by some certain events or processes that take place over time. Although different temporal systems show considerable commonality in structure, they also show considerable differences in formalization. In the literature, there are three choices regarding the primitive for the ontology of time: instantaneous points, durative intervals and both points and intervals and problems may arise when one conflates different views of temporal structure. A natural approach to representing and reasoning about the actions, events, processes is to associate them with time elements (i.e., instantaneous points and/or durative intervals) [18].

Many theories, [3], [22] and [23], are based on points as the basic primitive element. In these theories, intervals are defined in terms of points, usually by means of beginning and ending points. However, as Allen has commented [7], modeling intervals by taking their bounding-points can lead to problems: the annoying question of whether bounding-points are in the interval or not must be addressed, seemingly without any satisfactory solution. If intervals are all closed then adjacent intervals have bounding-points in common, which when adjacent intervals correspond to states of truth and falsehood of some property, can lead to situations in which a property is both true and false at an instant. Similarly, if intervals are all open, there will be points at which the truth or falsity of a property will be undefined. The solution, in which intervals are all taken as semi-open, so that they sit conveniently next to one another, seems arbitrary and unsatisfactory. Other theories, predominantly in [7], [8], treat intervals as primitive, and in [13], [14], treat both intervals and points as primitive on an equal footing.

After carefully considering the literature, we found that formal grounding is absent in the main commercial modeling techniques. However, frameworks based on temporal logic are available in the literature that are expressive than others and provides formal semantics if mapped. The objective of our work is to provide an ontology mapping of BPMN, UML AD and EPC to temporal model for business processes [18] which provides formal semantics.

III. Main BPM Techniques and Their Modeling Terms/Constructs

Before describing any technique we define what a business process is. According to Davenport [24], processes are defined as ‘‘structured, measured sets of activities designed to produce a specified output for a particular customer or market’’. There are so many other definitions but in essence all are the same: processes are relationships between inputs and outputs, where

[5] IBM, SAP AG, *WS-BPEL Extension for People–BPEL4People.*Whitepaper,2005 [http://www.ibm.com/developerworks/webservices/library/specification/wsbpel4people/].

[6] WFMC. Process Definition Interface – XML Process Definition Language, version 2.00, October 2005. [http://www.wfmc.org/standards/docs/TC-1025_xpdl_2_2005-10-3.pdf]

inputs are transformed into outputs using a series of activities, which add value to the outputs.

It is beyond the scope of this paper to go into further discussion on the difference between business processes and processes in general. It seems that some authors take them as synonymous. For example, in contrast to [15], [16] defines businesses process as ‘‘a collection of activities that takes one or more kinds of input and creates an output that is of value to the customer’’.

A. The Main Modeling Terms/Constructs of BPMN

The main goal of BPMN is to standardize a business process modeling notation in order to provide a simple means of communicating process information among business users, customers, suppliers, and process implementers. The basic BPMN constructs are *activity*, *event*, *gateway and sequence flow.*

An *activity* is a generic term for work performed within a company. It can be atomic or non-atomic (compound). The types of activity are: *task* and *process/sub-process*. A *task* is an atomic activity included within a *process/sub-process*. *Process/Sub-Process* is a sequence or flow of activities in an organization with the objective of carrying out work. There are two basic types of *processes*; *Private (internal),* and *Public Processes*. The *Private Processes* are those internal to a specific organization and further divided into two types *Executable* and *Non- executable*. An *executable process* is modeled for the purpose of being executed according to the semantics defined and there will be stages in lifecycle of a process where not enough detail available to execute it. A *nonexecutable process* is modeled for the purpose of documenting process behavior at a modeler-defined level of detail. A *public process* represents the interactions between a private business process and another process or participant. In addition to *process*, there are two types of *sub-processes*: *embedded* and *independent*. The *embedded sub-processes* are further divided into two views of *sub-processes*; *collapsed and expanded*. *Collapsed* view hides its details or *expanded* view that shows its details within the view of the *process* in which it is contained. An *event* is something that “happens”, like a trigger or a result, during the execution of a business process affecting the flow of the process. Since an event can start, suspend, or end the flow, we can distinguish the above between *start events*, *intermediate events*, and *end events* respectively. A *gateway* is a modeling element used to represent the interaction of different sequence flows, as they diverge and converge within a process. When the sequence flows arrive at a *gateway*, they can be merged together on input and/or split apart on output. There are different types of g*ateway* according to the types of behavior they define in the sequence flow. Decisions and branching are represented by *OR-Split*, *exclusive-XOR*, *inclusive-OR*, and *complex*, merging is represented by the *OR-Join gateway*, and forking is represented by the *AND-Split gateway*, and joining by the *AND-Join gateway*.

However, there are other constructs used to model any relevant entity that is able to activate or perform a process i.e. *pool and lane*. They are representing more aggregate organization units and more specific ones, respectively. They allow for a partitioning of activities according to the performers.

Critical Evaluation BPMN: BPMN elements are hard to sketch on paper unlike UML AD or flowcharts [21]. In [4], numerous ambiguities in the descriptions and under specifications of semantically relevant concepts pervade the standard document and leave space for incompatible (but, due to the lack of precision, standard `conforming') interpretations in design, analysis and use of BPs. Another under specification concerns *expression evaluation*. When should expressions (particularly event expressions) be evaluated? Depending on the type, it could (probably should) be either before or at process start, or upon state change or when a token becomes available. BPMN provides only poor conceptual support for numerous features which are characteristic of the design and management of business processes. One of BPMN's main shortcomings is that an increase in graphical notation yields an increased complexity in the meta-model and makes transparent faithful implementation more and more impractical. BPMN comes with a plethora of interdefinable constructs. Instead of defining a core of independent constructs in terms of which other constructs can be defined, as suggested in [5] for a previous version of the standard (it was also suggested to the standardization committee). The fuzzy overlapping of different constructs prevents `closed' descriptions of individual constructs in one place and makes their comprehension unnecessarily complex by forcing the reader to simultaneously and repeatedly consider multiple sections of the standard document. It also creates the problem that where the definitions overlap they have to be consistent; this problem is not considered in the standard document. Furthermore a statistical evaluation (of BPMN 1.1) shows that `the average BPMN model uses less than 20% of the available vocabulary' and that, out of the more than 50 graphical elements in BPMN, `Only five elements (normal flow, task, end event, start event, and pool) were used in more than 50% of the models we analyzed.

B. The Main Modeling Terms/Constructs of UML (AD)

UML(AD) is the object-oriented equivalent of flow charts and data-flow diagrams from structured development and describes the workflow behavior of a system. The process flows in the system are captured in the activity diagram and also illustrates the dynamic nature of a system by modeling the flow of control from activity to activity. The main constructs of UML AD are *activity, action, initial node, final activity* node, *connecting* nodes, *and* control flow.

An a*ctivity* is used to represent a set of actions and an *action* represents a single step within an activity. An *Initial Node* is the entry point to an activity diagram and an *Activity Final Node* is the final node in an activity that terminates the actions in that activity. However, an *Object Node* is used to represent an object that is connected to a set of Object Flows. A *Decision Node* is used to represent a test condition to ensure that the control flow or object flow only goes down one path. A *Merge Node* is used to bring back together different decision paths that were created using a decision-node. A *Fork Node* is used to split behavior into a set of parallel or concurrent flows of activities (or actions). A *Join Node* is used to bring back together a set of parallel or concurrent flows of activities (or

actions). *Control flow* shows the flow of control from one action to the next and also shows the sequence of execution.

Critical Evaluation of UML AD: Weaknesses of UML AD are given below:

- Some of the UML AD constructs lack a precise syntax and semantics. For instance, the "well-formedness" rules linking forks with joints are not fully defined, nor are the concepts of dynamic invocation and deferred events, among others.
- UML ADs are extremely limited in modeling resource-related or organizational aspects of business processes. It is interesting to note that UML ADs cannot capture many of the natural constructs encountered in business processes such as cases and the notion of interaction with the operational environment in which the process functions.

These limitations observed by [19] and [20] are common to many other business process modeling formalisms and reflect the overwhelming emphasis that has been placed on the control-flow and data perspectives in contemporary modeling notations. While UML ADs are functional, business analysts somehow cannot use them without prior technical knowledge in [1]. A business analyst cannot model a business process and its sub-processes from the highest level to the lowest level of detail in an UML AD. It is increasingly losing favor with practitioners (although there are currently several projects working on UML-to-BPEL translations by IBM and OMG) [17]. This is mainly due to industry's growing consolidation of BPMN as the de facto standard for BPM.

C. The Main Modeling Terms/Constructs of EPC

Event-driven process chains (EPCs) are part of a holistic modeling approach, called the ARIS framework. Process modeling uses event-driven process chains. The main building blocks of event-driven process chains are *events, functions, connectors, and control flow edges*.

Functions describe transformations from an initial state to a resulting state. They represent units of work and granularity of these functions depends on the modeling purpose. The entering of a business relevant state is represented by an *event*. *Events* trigger *functions* and passive elements in EPC. They describe under what circumstances a *function* works or which state a *function* results in. Examples of *events* are "requirement captured", "material in stock", etc. EPC diagram must start with an *event* and end with an *event*. Unlike events, functions are active elements that take input and transform it to output. Functions can also make decisions that influence the behavior of the process through connector nodes associated with the function. They are triggered by events, and on the completion of a function, an event occurs. There are three kinds of logical relationships exists between events and functions and are *branch/merge, fork/join* and *or*. C*ontrol flow* connects events with functions, process paths, or logical connectors creating chronological sequence and logical interdependencies between them.

Critical evaluation of EPC: It works as an ordered graph of events and functions and supports parallel execution of processes. However, the semantics and syntax of the EPC are apparently not well defined [6] [26]. Because of these limitations and the absence of a standardization process, the EPC will not be classified as a graphical standard.

Formal systems are used for the clarity and unambiguous reasoning and representation. However the commercial modeling tools and techniques are ambiguous in representation and lack formal foundation. To provide a clear and precise meaning to different modeling terms discussed above, will provide a mapping of ontology to formal logic. For this, we categorize and group modeling terms/constructs used by the modeling tools and techniques discussed above into three distinguished conceptual terms and formally defining them. This categorization will serve the purpose of the ontology mapping to a formal system.

D. Conceptual Categorization of Modeling Terms/Constructs

We have seen that the modeling terms discussed in previous sub-sections have somewhat similarities and differences. Main modeling techniques such as BPMN, UML AD and EPC provide no formal foundation and leads to ambiguity in the design. To overcome this problem and fill this gap, we propose conceptual categorizations of modeling terms used by BPMN, UML AD and EPC. Modeling terms of these techniques are grouped into three conceptual categories: *process*, *connector* and *constraint*. These categories refer to the modeling terms/constructs used in modeling the dynamic and static aspects of the domain. This categorization will assist in ontology mapping of commercial modeling terms to formal notion given in section V. We will take conceptual category's first letter of every word to form a title of our categorizations which would be ***P**rocess*, ***C**onnector* and ***C**onstraint* (**PCC)** and formally defined below:

Definition (Process): In our proposed conceptual category of *process*, we take every activity as a non-instantaneous *process* and define *process* as nonempty set containing processes, or it also could be a singleton set which may be considered as a special case of a *process* and above can be represented as

$$Process\ \text{`P'} \neq \emptyset \text{ or } Process\ \text{`P'} = \{p\}$$

In addition, an event may also be considered as a *special process* that starts (p_S) and ends (p_E) a *process* P and is shown below:

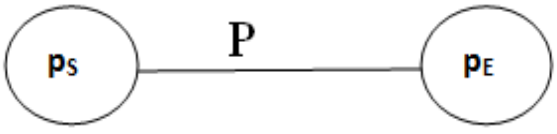

Fig. 1. An abstract process

Keeping in mind the temporal nature of the *process*, we could formalize it by using a predicate **Occurring,** for *process P* i.e. non empty set of processes, occurring over a time element *T* and using **Dur** to express the duration by the following axiom:

Occurring $(P, T) \Rightarrow \exists\, p_S, p_E \wedge \text{Dur}(p_S) = 0 \wedge \text{Dur}(p_E)=0 \wedge \text{Meets}(p_S, T) \wedge \text{Meets}(T, p_E) \wedge \text{Occurring}(p_S, P) \wedge \text{Occurring}(P, p_E)$……..(Axiom 1)

p_s and p_E are the *special processes* i.e. events, that starts and ends a process P respectively. The above axiom refers to a general process which occurs over a time element that may be divisible.

To formalize a singleton set of process *p*, using **Occurring** predicate, **Dur** for duration and occurring over a time element *t* is given below in axiom 2 using temporal relation 'In' given by Allen in [8]:

Occurring $(p, t) \Rightarrow \exists\, p_S, p_E \wedge \text{Dur}(p_S) = 0 \wedge \text{Dur}(p_E)=0 \wedge \text{Meets}(p_S, t) \wedge \text{Meets}(t, p_E) \wedge \neg \exists\, [t_1 \wedge \text{Dur}(t_1) > 0 \wedge \text{In}(t_1, t) \wedge \text{Occurring}(p, t_1)]$……(Axiom 2)

Axiom 2 has shown a singleton i.e. nondivisible, process. *Special process* i.e. event, may also occur when finishing a process and starting a new process i.e. start, end and intermediate event of BPMN. We have shown above that our conceptual category *Process* is general enough to map BPMN's modeling terms/constructs *task, process/sub-process, start event, intermediate event* and *end event*, UML AD's *activity*, *action*, *initial node* and *final activity node*, and EPC's *functions* and *events*.

Definition (Connector): We describe Connector as a set which contains logical operators AND and OR, and is given below:

Connector 'C' = {∧, ∨}

Our proposed conceptual category c*onnector* can map BPMN's modeling term/construct *gateway*, UML AD's *connecting nodes*, and EPC's *logical connectors.*

Definition (Constraint): We describe Constraint as a set 30 derived relation given in [13] gathered in four groups and is given below:

Constraint 'C'= {point to point, point to interval, interval to interval, interval to point}

The conceptual category c*onstraint* can map BPMN's modeling term/construct *sequence flow*, UML AD's *control flow* and EPC's *control flow*.

The formalization of three conceptual categories has paved the way to group main modeling terms of commercial modeling techniques. This effort provides a generalized view of all modeling terms of the modeling techniques discussed in this paper. Now, we would show the aforementioned **PCC** categorization and its subsumption of modeling terms in the table I.

IV. Temporal Model for Business Processes

Ma and Knight [13] have proposed more general time theory that considers point and interval both on equal footing i.e. primitives. In addition, abstract modeling terms have been proposed [18] and we will use them for ontology mapping purposes. BPM terms such as *action, event, business process, sub-process* and *temporal relations* are defined by providing formalisms and discussed in the next sub-section.

TABLE I. Conceptual Categorization of Business Process Modeling Terms

Modeling Notation	Modeling Category	Modeling Terms
BPMN	Process	task, process/sub-process, start event, intermediate event, end event
	Connector	gateways: AND, OR & XOR
	Constraint	sequence flow
UML AD	Process	activity, action, initial node, final activity node
	Connector	connecting nodes: Merge Node, Fork Node, Join Node
	Constraint	control flow
EPC	Process	function, event
	Connector	logical connectors: AND, OR, XOR
	Constraint	control flow

Conceptual Categorization

A. Abstract Business Modeling Terms

BPM terms *action, process, and sub-processes* [18] are associated with non-instantaneous activity and *event* is associated with instantaneous activity i.e. point. An *action name* is an identifier that describes a certain type of non-instantaneous activity. For instance, "push a cart", "cut wire" and so on. They used a, a_1, a_2, …, etc., to denote action names, and write the set of action names as $\boldsymbol{A}$. Without confusion, they simply call an action name, say *a*, action *a*. It is important to note that a given type of action may perform once, more than once over different time moments, or may not even perform at all. An *event name* is an identifier that describes a certain type of instantaneous activity. For instance, "departure at", "start cut wire", "finish cut wire" and so on. They used e, e_1, e_2, …, etc., to denote event names, and write the set of event names as *E*. Without confusion, we may simply call an event name, say *e*, event *e*. Like action, a given type of event may occur once, more than once at different time points, or may not even occur at all. A *business process name* is a set of action names and set of event names.

They have used logical connectors to establish the relation between action and event i.e. ∧, ∨. Allen and Hayes [9] introduced 13 relations in his famous Interval Algebra; however Ma and Knight [13] from these 13 relations derived 30 temporal relations to constrain the order of the actions and events. In terms of the single primitive relation *Meets*, other binary relations over points/intervals can be classified into 4 groups:

- Point – Point: {Equal, Before, After}
- Point – Interval: {Before, After, Meets, Met_by, Starts, During, Finishes}
- Interval – Point: {Before, After, Meets, Met_by, Started_by, Contains, Finished_by}

- Interval – Interval: {Equal, Before, After, Meets, Met_by, Overlaps, Overlapped_by, Starts, Started_by, During, Contains, Finishes, Finished_by}

The next section will provide ontology mapping based on the categorization provided in section III.

V. Ontology Mapping

To provide formal semantics for commercial modeling techniques and languages mentioned in section III, there are different extensions provided separately in the literature. However, there is no effort has been made to provide an ontology mapping for them.

There are different framework/systems available in the literature which discusses the ontology of modeling tools and techniques i.e. process, event and action. Our effort is to achieve a comprehensive ontology mapping using **PCC** categorization introduced in section III that will map commercial modeling terms to formal modeling terms in [18] and shown in table II.

TABLE II. Ontology Mapping Table for BPM Terms/Constructs

Modeling Notation	Modeling Category	Modeling Terms	Abstract Modeling Terms
BPMN	Process	task, process/sub-process, start event, intermediate event, end event	action, event, process
	Connector	gateways: AND, OR & XOR	logical operators ∧, ∨
	Constraint	sequence flow	temporal relations
UML AD	Process	activity, action, initial node, final activity node	action, event, process
	Connector	connecting nodes: Merge Node, Fork Node, Join Node	logical operators ∧, ∨
	Constraint	control flow	temporal relations
EPC	Process	function, event	action, event, process
	Connector	logical connectors: AND, OR, XOR	logical operators ∧, ∨
	Constraint	control flow	temporal relations

Ontology Mapping

The above table can also be shown in fig 2:

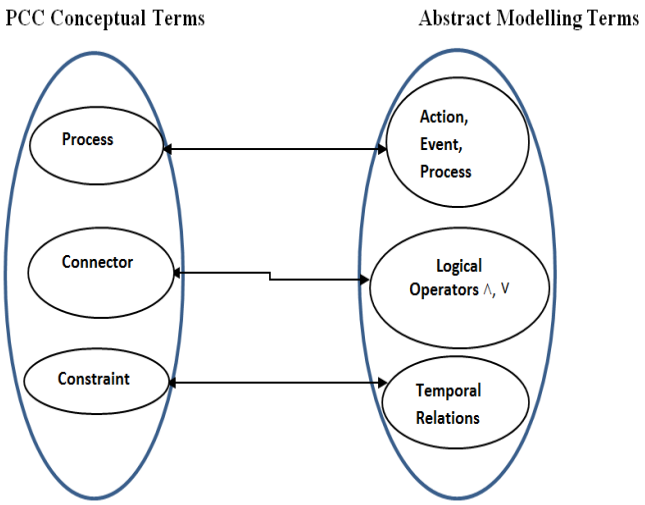

Fig. 2. Ontology Mapping

Ontology mapping using **PCC** categorization provides a platform for business process modeling techniques to have formal semantics which are intuitive but formal. This attempt will lead to provide a step towards foundation for business process modeling based on temporal logic.

To show the mapping is comprehensively achieved the aim of this paper; we will discuss an illustrative example in next section graphically representing it using modeling techniques used in section III and IV.

VI. Graphical Representation

The time theory outlined above in terms of the single *Meets* relation allows a simple graphical representation of any set of temporal knowledge. Each time element *t* is denoted as a directed arc of the graph labeled by *t* (and its duration if it is known), with a pair of nodes which are called the start-vertex, and the end-vertex, of the arc, respectively.

- Each relation *Meets*(t_i, t_j) is represented by means of merging the end-vertex of t_i and the start-vertex of t_j as a common vertex, of which t_i is an in-arc and t_j is an out-arc, respectively. In this case, arc t_i is said to be adjacent to arc t_j.

In general, the temporal order relation between two time elements may be given in any form of those 30 as classified in section IV. However, as defined above in section IV, each of these temporal relations can be derived from the single *Meets* relation. Therefore, all the knowledge about the temporal relations over a given collection of time elements (points and/or intervals) can be transformed and stored as a table of *Meets* relations in the knowledge base. We use the following graphical representation to represent the available knowledge:

Time point, can be represented as ⟹

Time moment can be represented as ⟶

Time interval can be represented as - - - - -▶

start/end vertex: ○

This graphical representation of the underlying logical structure forms the link between the temporal theory, and practical business process diagrams. For instance, will consider a process graphically represented in the aforementioned three commercial BPM languages and afterwards will represent the same process graphically using logically defined terms to show the expressiveness, simplicity and generality of the logical structure.

A. An Illustrative Example:

*BPMN***:** Fig 3 describes an example business process using BPMN. The business process begins with a start event to execute the first task **Record the Claim**. **Calculate the Insurance Sum** and **Record the Claim** are part of the Pool **Financial Claim Specialist**. After the task **Record the Claim** the pool **Claim Administrator** is responsible for the process. The exclusive gateway splits the flow, because the decision has to be made if the insurance sum has a minor amount or major amount. If the insurance sum has a minor amount, then only task **Contacting the Garage** is processed. But if the insurance sum has a major amount then **Contacting the Garage** concurrently starts with the **Checking History of the Customer**. An inclusive gateway combines the different paths. The gateway conforms to the logical operator OR. After the task **Examination of Results** the decision has to be made if the company **Pay for the Damage** or **Does Not Pay for the Damage**. After that decision the case is closed.

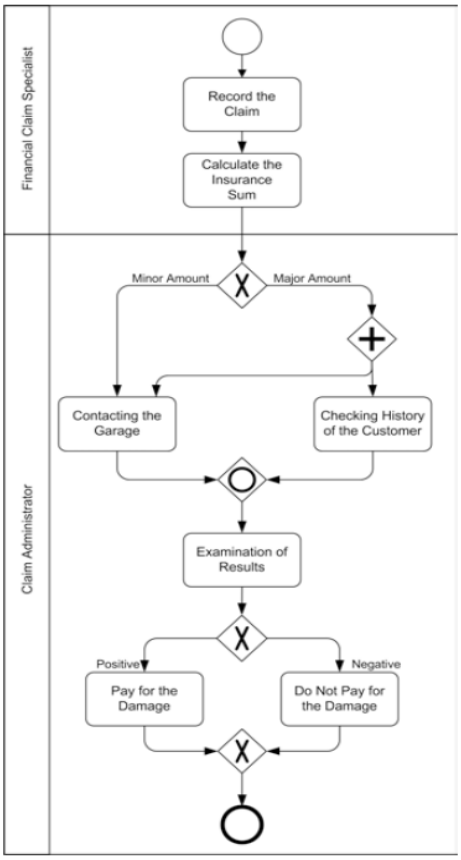

Fig. 3. Business Process using BPMN

UML AD: A business process example in UML AD is shown in fig 4. The business process starts with an *initial node*, to activate the first action; **Record the claim**, Record the claim passes the token to the next action to **Calculate the Insurance Sum**. These two actions part of activity partition **Financial Claim Specialist**. After calculating the insurance sum the path Is split up by a decision node into two alternative flows, depending if the insurance sum has a minor amount then the action **Contacting the Garage** starts. If the insurance sum has a major amount, then the flow is split up into parallel paths by a fork node. That means that the actions **Contacting the Garage** and **Checking History of the Customer** are executed concurrently. A merge node combines the different flows, and accepts the token as well as of one path or both paths. The action **Examination of Results** decides that the claim is handled either positive or negative. Therefore a decision node splits up the path in two alternative flows, with the actions **Pay for Damage** or **Do Not Pay for Damage**. After that decision the business process ends with a flow final activity node.

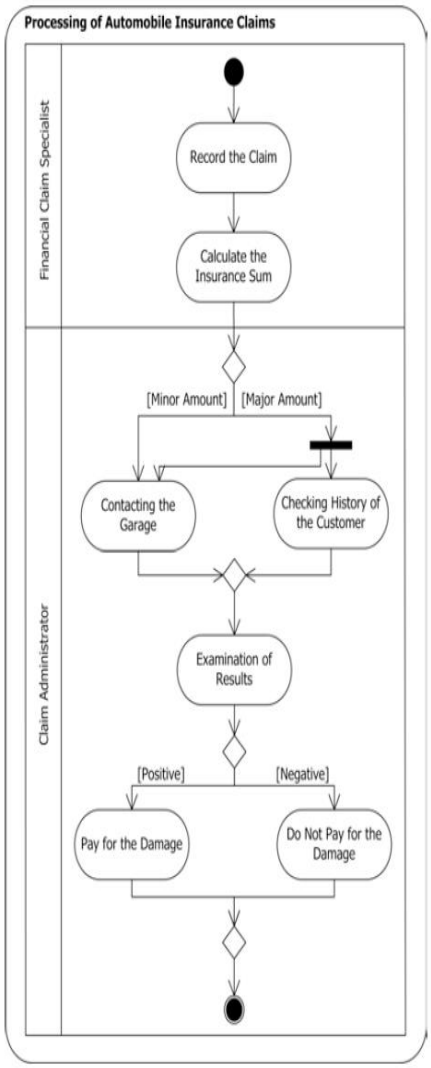

Fig. 4. Business Process example using UML AD

EPC: Fig 5 shows a business process example in EPC. The business process starts with event **New Claim Submitted**. The function **Record the Claim** starts after the first event. After the event **Claim Recorded** the function **Calculate the Insurance Sum** begins. The organizational role **Financial Expert** is responsible for the functions **Record the Claim** and **Calculate the Insurance Sum,** the path of business process splits up into alternative flows, depending of the insurance sum has a **Minor Amount** or **Major Amount.** If the insurance sum has a minor amount, then the function **Contacting the Garage** starts. If the insurance sum has major amount then the functions, **Contacting the Garage** and **Checking History of the Customer** starts concurrently. These two functions are connected with the next event. **Results Collected** by an OR-Join. At that time the organizational role **Claim Administrator** is responsible for the business process. Due to the fact that either **Checking History of the Customer** or **Contacting the Garage** is executed, or both functions are processed, the OR-Join is needed. If the results are collected, then the function **Examination of Results** starts processing, to decide if the claim is handled **Positive** or **Negative**. If the claim is handled positive, then the insurance company **Pays for the Damage**, otherwise not. In both situations **Case is Closed**.

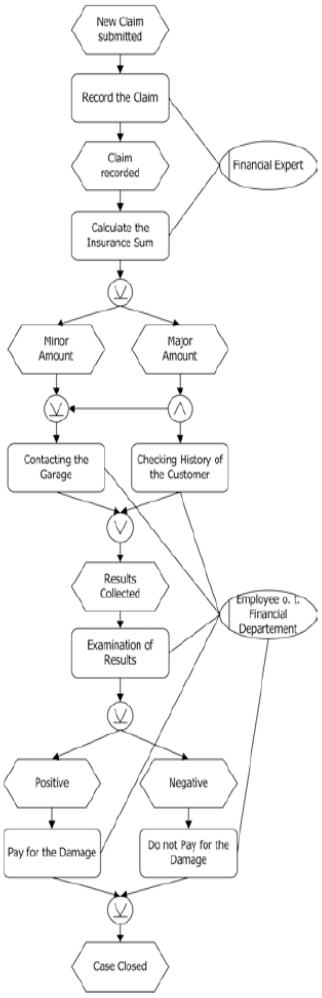

Fig. 5. Example business process of an EPC

After using **PCC** categorization and graphical representation described above, the aforementioned example can be shown in fig 6, using abstract modeling terms/constructs. Notice that there are both absolute and relative times in this model showing temporal ordering of processes.

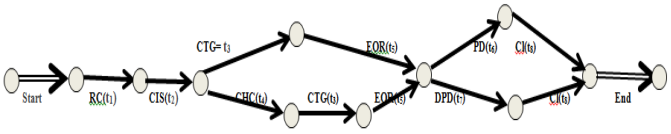

Fig. 6. Business Process using Abstract Modeling constructs

Note: RC: Record the Claim: t_1; **CIS**: Calculate the Insurance Sum: t_2; **CTG**: Contacting the Garage: t_3; **CHC**: Checking History of the Customer: t_4; **EOR**: Examination of Results: t_5; **PD:** Pays for the Damage: t_6**; DPD**: Does Not Pay for the Damage: t_7; Cl: Closed the Case: t_8.

Using *Meets* relation, the above can be represented as

$$\text{Meets}(t_1, t_2) \wedge (\text{Meets}(t_2, t_3) \wedge \text{Meets}(t_3, t_5) \wedge \text{Meets}(t_5, t_6) \wedge \text{Meets}(t_6, t_8)) \vee (\text{Meets}(t_2, t_4) \wedge \text{Meets}(t_4, t_3) \wedge \text{Meets}(t_3, t_5) \wedge \text{Meets}(t_5, t_7) \wedge \text{Meets}(t_7, t_8))$$

We have seen above that abstract modeling terms have subsumed all modeling elements of aforementioned commercial modeling techniques shown in fig 3, 4 and 5. We also have used a simpler graphical representation to verify the logical structure that is intuitive and expressive than others to represent concepts and knowledge in a simpler but composed way.

VII. Conclusion & Future Work

In this paper we presented a framework for BPM using ontological mapping. A versatile conceptual categorization introduced and subsequently formalized the categories that can group and map modeling terms used by different tools and techniques to formal notion. There are a very large number of BPM languages so our effort is to work towards bridging the gap for business process modeling to be grounded on formal logic. Some efforts carried out in providing formal semantics and have given partial results so far. We have consolidated it by proceeding along two lines. One is methodological, necessary to provide its ontological mapping, and another is empirical, necessary to check its value in real business settings as given below:

- the fact that BPMN, UML AD and EPC does not provide formal grounding to represent their constructs, being it specified in an informal way;
- Ontology mapping for business process modeling based on logic provides a step towards in providing a formal foundation.

The absence of formal grounding in modeling techniques will always result in some loss of data or semantics of the control flow. After a careful analysis of the literature we identified temporal logic as a possible candidate to match the above requirements. To fill this gap, it is envisaged by the author to provide an axiomatic system based on temporal logic

for the business process modeling that will formally ground the modeling notion and provide a unified graphical representation for process which would be simpler and easy to design and more suited to majority of the user's needs.

ACKNOWLEDGMENT

We are thankful to Vice Chancellor of University of Greenwich who has sponsored this project.

REFERENCES

[1] A.E. Bell. Death by UML fever: self-diagnosis and early treatment are crucial in the fight against UML fever", ACM Queue, Vol. 2(1), 2004, pp. 72-80.

[2] A.-W. Scheer, O. Thomas and O. Adam. *Process Modeling Using Event-Driven Process Chains.* In "Process-Aware Information Systems", edited by M. Dumas, W. hal-00656686 van der Aalst, A.H.M ter Hofstede. Wiley-InterScience, 2005, pp 119-145.

[3] B. C. Bruce. A Model for Temporal References and Application in a Question Answering Program. *Artificial Intelligence*, An International Journal Vol 3, 1972, pp.1-25.

[4] E. Börger. Approaches to modeling business processes: a critical analysis of BPMN, workflow patterns and YAWL." *Software & Systems Modeling* Vol 11(3), 2012, pp 305-318.

[5] E. Börger and B. Thalheim. A method for verifiable and validatable business process modeling. In Advances in Software Engineering, Vol 5316 of LNCS, 2008, pp 59-115. Springer-Verlag.

[6] E. Kindler. On the semantics of EPCs: a framework for resolving the vicious circle, paper presented at Business Process Management: 2nd International Conference, 2004, BPM 2004, Potsdam.

[7] J. Allen. Maintaining Knowledge about Temporal Intervals. *Communication of ACM*, Vol 26, 1983, pp.123-154.

[8] J. Allen. Towards a General Theory of Action and Time, *Artificial Intelligence*,Vol 23, 1984, pp 123-154.

[9] J. Allen, and J. Hayes. Moments and Points in an Interval-based Temporal-based Logic, *Computational Intelligence*, Vol 5(4), 1989, pp 225-238.

[10] J. Harrington. Business Process Improvement: The Breakthrough Strategy for Total Quality, Productivity and Competitiveness. McGrawHill, New York, USA, 1991.

[11] J. L. Peterson. Petri Nets. ACM Computing Surveys Vol 9(3):1977, pp 223–252.

[12] J. L. Peterson. Petri Net Theory and the Modeling of Systems. Prentice Hall, 1981.

[13] J. Ma and B. Knight. A General Temporal Theory, *The Computer Journal*, Vol 37(2), 1994, pp 114-123.

[14] M. B. Vilain. A System for Reasoning about Time. *Proceedings of AAAI*, Vol 1, 1982, pp.197-201.

[15] M. Hammer. Reengineering work: Don't automate. Obliterate. Harvard Business Review Vol 68 (4), 1990, pp 104–112.

[16] M. Hammer and J. Champy. Reengineering the Corporation: A Manifesto for Business Revolution. NewYork, USA, 1993

[17] M. Koskela,and J. Haajanen. Business process modeling and execution: tools and technologies report for the SOAMeS project", VTT Research Notes No. 2407, VTT Technical Research Centre of Finland, Espoo, 2007

[18] .M. Petridis, J. Ma, and B. Knight. Temporal model for business Process" Intelligent Decision Technologies, Vol 5(4), 2011, pp 321-331.

[19] N. Russell, W.M.P. Van Der Aalst, A.H.M. Ter Hofstede, and P. Wohed. On the suitability of UML 2.0 activity diagrams for business process modeling, Proceedings of the 3rd Asia-Pacific Conference on Conceptual Modeling, Australian Computer Society Vol. 53, 2006, pp. 95-104.

[20] P. Wohed. Pattern-based Analysis of UML Activity Diagrams, Beta, Research School for Operations Management and Logistics, Eindhoven, 2004.

[21] P. Wohed, W. M. P. van der Aalst, M. Dumas, A.H.M. ter Hofstede, and N. Russell. On the Suitability of BPMN for Business Process Modeling, Business Process Management, Vienna, 2006, pp. 161-76.

[22] R. Dechter, I. Meiri and J. Pearl. Temporal Constraint Networks. *Artificial Intelligence*, Vol 49, 1991, pp.61-95.

[23] R. Maiocchi. Automatic Deduction of Temporal Information. ACM Transactions on Database Systems, Vol 4, 1992, pp.647-688.

[24] T.H. Davenport. Process Innovation: Reengineering Work through Information Technology. Harvard Business School Press, Boston, MA, USA, 1993.

[25] M. Weske, Concepts, Languages, Architectures. Vol. 14. 2007, Berlin: Springer-Verlag.

[26] W.M.P. Van der Aalst. Formalization and verification of event-driven process chains", Information and Software Technology, Vol 41(10), 1999, pp. 639-50.