AN INTEGRATIVE APPROACH TO SUPPORT MULTI-PERSPECTIVE BUSINESS PROCESS MODELING

Keletso J. Letsholo, Erol-Valeriu Chioasca and Liping Zhao
School of Computer Science
The University of Manchester
Manchester, U.K.
{letsholk, chioasca, lzhao}@cs.man.ac.uk

Abstract

Business process modeling (BPM) has become a dominating approach for organizations to understand, plan and optimize their business processes. BPM has also become an important part in service systems development because it helps to identify service activities embedded in business processes. Several modeling languages have been developed to support BPM, including IDEF, BPMN and EPC. Yet, our research shows that these languages still lack comprehensive constructs for representing some core business concepts, such as business goals, non-functional requirements and resources. This paper has two related purposes: First, it uses the Zachman Framework to assess the modeling capabilities of some of the popular BPM languages to identify their modeling gaps. The Zachman Framework is known to be comprehensive for representing different business perspectives. Second, the paper proposes an integrative approach to support multi-perspective BPM, so as to bridge the identified modeling gaps. This approach is illustrated through an example. The proposed approach aims at supporting the creation of comprehensive models and facilitating a common understanding of business perspectives regardless of the languages that represent them. The main benefit of the proposed approach is its integration of existing modeling languages that are familiar to business analysts, rather than introducing new languages. Further it provides guidance for business analysts and helps them to select appropriate modeling languages.

Keywords: Business Process Modeling, Zachman Framework, Modeling Perspectives, Modeling Languages.

1. INTRODUCTION

Process modeling has been embraced as an appropriate approach for visually describing how businesses conduct their operations and also for increased awareness of business process knowledge (Curtis, Kellner, & Over, 1992). Moreover, process modeling focuses on understanding the underlying business process elements that are fundamental to the design and implementation of information systems that satisfy the business strategy.

Early efforts on representing the dynamic behavior of organizations, dating back to the 1960s and 1970s, have resulted in several business analysis and modeling languages, including Flowcharting (Schriber, 1980), Business Information Analysis and Integration Technique (BIAIT) (Carlson, 1979), Business Information Characterization Study (BICS) (Kerner, 1979), and Frame-work for Information System Architecture (Zachman, 2003). Business process modeling (BPM), seems to have emerged in the 1980s during the movement of business process design and re-engineering, to represent processes through which work is accomplished within an organization (Curtis et al., 1992; Davenport & Short, 1990). BPM is similar to information systems modeling (ISM) (Curtis et al., 1992; Davenport & Short, 1990), they both share a common set of notations, such as Data flow, Control flow, State Transition, Entity Relationship diagram, Petri Nets (Murata, 1989), and UML (Rumbaugh, Jacobson, & Booch, 1999). A review of BPM and ISM languages is given in (Aguilar-Saven, 2004; Giaglis, 2001).

A business process is an ordered sequence of activities and events, designed to achieve a defined business objective. A comprehensive business process description therefore, needs to incorporate answers to the following questions: what activities are being performed and how, what objects are manipulated; when activities are performed; where and by whom they performed; finally, why are they performed. These interrogatives are referred to as perspectives in conceptual modeling, elaborated in (Curtis et al., 1992; Frankel et al., 2003) as follows:
- Functional perspective: represents what process elements (activities) are being performed.
- Behavioral perspective: represents when and how activities are performed.
- Operational perspective: represents where and by whom in the organization are activities performed.
- Informational perspective: represents what informational entities are being manipulated by activities.
- Goal perspective: represents why activities are performed.

In spite of these perspectives, a prevailing concern in the body of information systems literature is that no single process modeling language is fully capable of describing or representing all the perspectives (Brooks, 1987; Hickey & Davis, 2004; Maiden & Rugg, 1996). Attempts towards addressing this concern have resulted in two possible research paths. One path is towards extending existing methods' schema to incorporate the representation of missing concepts. For example, BPMN (OMG, 2011) is inadequate in representing information and goal perspectives (Recker, 2010; Zhao, Letsholo, Chioasca, & Sampaio, 2012). Efforts have been made towards extending BPMN to incorporate these perspectives into its modeling portfolio (Gorton & Reiff-Marganiec, 2006; Recker, Indulska, & Green, 2007; Saeedi, Zhao, & Sampaio, 2010). However, such efforts are fragmented, and are not aligned with the overall BPMN schema. In the second path, BPM could adopt multi-perspective modeling frameworks similar to the ones proposed in (Frank, 2002; Hickey & Davis, 2004; Kruchten, 1995; Rumbaugh et al., 1999). This paper follows the second path. We posit that a unified modeling approach should entail a coordinated effort on the development of a common framework for model creation, extension, composition, and management.

This paper serves two purposes: First, it uses the Zachman Framework to assess modeling capabilities gaps of process modeling languages. Second, motivated by the assessment results and guided by business process schema (Figure 1), it proposes an integrative approach to support multi-perspective modeling of business processes. The approach suggests other requirements modeling languages such as goal-oriented and data-oriented languages, to bridge the identified gaps.

The paper proceeds as follows. Section 12 introduces different perspectives in business modeling, while Section 3 presents an overview of BPM languages and identifies their modeling capability gaps. Section 4 describes our approach and Section 5 illustrates this approach with an example. Section 6 discusses our approach in relation to similar work and Section 7 gives a summary of the paper and identifies future work.

2. PERSPECTIVES IN BUSINESS MODELING

2.1 Business Process Schema

In this Section, we describe a generic schema (Figure 1) that serves as the basis for defining business processes. To ensure comprehensiveness, the schema is developed in line with conceptual frameworks proposed in (Curtis et al., 1992; Frankel et al., 2003). The schema provides a foundation for capturing knowledge, from what is being performed, and how, who performs it, and why is it being performed. The captured knowledge is capable of analyzing and integrating underlying concepts of an enterprise. Goals describe the objectives of a business process and are realized by activities. A business process is a composition of activities structured in some way to achieve the business goal. An Activity may be either atomic or compound and is performed by agents. Activities transform resources into specific products or services. A Resource represents the subject matter in a business process. A Resource may be physical, conceptual or financial, for example book, message or money. A Container denotes the boundary of an operation (physical or conceptual) where actions are performed. Examples of containers include geographical locations, departments, companies, and software systems. An Agent represents responsibility assignments (human or machine). Agents are part of a container, for example employees in a company. Events capture the order in which activities are performed as well as aspects of how they are performed. Examples include decisions, timing and exception handling events.

![Figure 1. Generic Business Process Schema](http://hipore/ijsc)
2.2 Zachman Framework

The Zachman Enterprise Engineering Framework (Zachman, 2003) is a two dimensional, multi-perspective scheme for describing and modeling an enterprise and its software systems. The framework is made of six columns and five rows. The columns classify a business into different facets through What, How, Where, Who, When, and Why interrogatives whereas the rows show these business facets from the perspectives of Planner, Owner, Designer, Builder, and Sub-contractor. Crossing each row by each column, results in a cell, which contains a unique business model.

Although not proved mathematically, Zachman claims that the columns and rows in his framework are both primitive and comprehensive. He states that when used together, these rows and columns provide a complete description of a business. According to our observation, the What, How, Where, Who, When, and Why interrogatives are universal in conceptual modeling (Curtis et al., 1992; Dardenne, Van Lamsweerde, & Fickas, 1993; Lohse, Min, & Olson, 1995; Sowa, 1984). Under the guise of different analogies or synonyms: What (entity, object, subject, material, resource, information, thing); How (operation, process, function, activity, task, action); Where (location, place, container, network, structure); Who (agent, actor, people, machine); When (time, event, exceptions); Why (motivation, purpose, goal, objective, strategy, policy, desired state). This suggests that they constitute a fundamental set of modeling concepts, even if they may not form a complete set.

In this paper, we focus on the Owner perspective due to its core concern in tackling conceptual modeling issues. Figure 2 shows a set of business models from this perspective. A summary of concept mappings between this perspective and business process schema constructs is presented on Table I.

![Figure 2. Zachman Framework: the Owner's Perspective](http://hipore.com/ijsc)

### Table I. Mapping Business Process Schema onto Zachman Models

<table>
<thead>
<tr>
<th>Zachman Model</th>
<th>Business Process Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>What(Data)</td>
<td>Resource</td>
</tr>
<tr>
<td>How(Functional)</td>
<td>Process/ Activity</td>
</tr>
<tr>
<td>Where(Network)</td>
<td>Container</td>
</tr>
<tr>
<td>Who(People)</td>
<td>Agent</td>
</tr>
<tr>
<td>When(Time)</td>
<td>Event</td>
</tr>
<tr>
<td>Why(Motivation)</td>
<td>Goal</td>
</tr>
</tbody>
</table>

3. IDENTIFYING MODELING CAPABILITY GAPS

BPM has received a lot of attention from both researchers and practitioners; consequently, several modeling languages have been developed. This paper reviews BPM languages presented in information systems and requirements engineering literature from the 1980s to present. The following survey and review articles on business process modeling languages (Aguilar-Saven, 2004; Giaglis, 2001; List & Korherr, 2006; Söderström, Andersson, Johannesson, Perjons, & Wangler, 2006),
provided useful information on known application areas, limitations, modeling capabilities, and basic notation constructs of the languages. This paper only describes languages that are well established in research and industry.

### 3.1 BPM LANGUAGES - AN OVERVIEW

Business Process Modeling Notation (BPMN) (OMG, 2011) is a single diagrammatic notation based on a flowcharting technique. Through this technique a Business Process Diagram (BPD) is produced, which is a network of graphical objects, consisting of activities and flow controls that define activities' performance order. BPMN’s modeling constructs are divided into four basic categories: flow objects, connectors, artifacts and swim lanes.

Integrated DEFinition (IDEF3) (Mayer et al., 1995) is a family of methods developed by US Air Force in the mid-1970s. IDEF3 Process Description Capture method captures the behavioral aspects of a process and represents them in the context of a scenario. IDEF3 supports both process-centric and object-centric descriptions. A process-centric view captures knowledge on “how things work” in an organization, while the object-centric view describes the objects allowable state transitions in a particular process.

UML Activity Diagram (Rumbaugh et al., 1999) represents activities and work-flows within a software system. The basic constructs used includes activity, fork or join control flows, and a swim lane to organize the activities according to their responsibilities.

Petri Nets (Murata, 1989) were designed with an aim of modeling the dynamic behavior of concurrent systems and nondeterministic procedures. A Petri Net is a directed graph with two nodes, places and transitions. Places represent possible states of the system. Transitions represent events or actions that cause the change of state. In process modeling terms, places represent conditions, and transitions represent events. A transition has a number of inputs and output places representing pre-conditions and post-conditions of an event.

Role Activity Diagram (RAD) was developed originally for modeling coordination, but it also used today for modeling business processes. An RAD is a flowcharting notation that describes responsibilities assignments. A role is a set of related responsibilities or activities that can be performed by an agent. In general, RAD comprises a set of activities, decisions and transactions.

Event Driven Process Chains (EPC) (Mendling, 2009) is a flowcharting language used for representing temporal and logical dependencies of activities in a business process. EPC notation offers function type elements to capture activities of a process and event type elements describing pre-conditions and post-conditions of functions.

Additionally, the notation is capable to capturing organizational units and process owners.

Although efforts have been made towards developing a framework for assessing BPM languages (Giaglis, 2001; List & Korherr, 2006; Söderström et al., 2006), these efforts are still not aligned with the overall business process schema. Most assessments have concentrated on a subset of modeling perspectives whilst ignoring others. Several authors (Batini, Ceri, & Navathe, 1992; Moody & Shanks, 1994; Wand & Weber, 1993; Wang & Strong, 1996) have mentioned criteria for judging the quality of an informational perspective. Although their focus is on the informational perspective, we believe the proposed criteria can be used for assessing other perspectives in business process modeling. The most common criteria include:

- **Completeness** – the extent to which the modeling constructs sufficiently express all the perspectives of the modeled domain (Batini et al., 1992; Moody & Shanks, 1994; Wand & Weber, 1993).
- **Consistency** – the extent to which the statements in the model do not contradict each other and minimal overlapping concepts as much as possible (Wand & Weber, 1993; Wang & Strong, 1996).
- **Comprehensibility** – the model should be easy enough to understand so that it could be used as a platform for communication (Batini et al., 1992; Curtis et al., 1992; Moody & Shanks, 1994).

### 3.2 ASSESSMENT RESULTS

The assessment presented in this section uses Zachman Framework as a yardstick to judge the comprehensiveness of modeling languages. We analyzed the modeling notation/schema/meta-model for the languages presented in Table II to check for the availability of constructs or elements to represent Zachman’s perspectives. Table II gives a summary of the evaluation results. The ‘+/−’ symbol denotes that the language has modeling constructs (notation) available to strongly represent that particular perspective. The ‘−/−’ symbol indicates that the perspective is weakly supported by the language’s notation and ‘−/−’ indicates that the perspective is not represented by the language.

From the assessment results, we note that BPM languages differ in their modeling capabilities. Some modeling languages focus mainly on one business perspective while partially focused on other perspectives. For example, some have focused primarily on functions, others on behavior, and yet others on informational resources. In general, Functional and Time perspectives are well supported or represented by BPM languages. Data perspective is weakly supported while Network and People's perspectives are supported by some languages (BPMN, AD, RAD) and not supported by others (IDEF3, PN). Any BPM languages do not explicitly support the Motivation perspective.
3.3 DATA AND GOAL ORIENTED MODELING LANGUAGES

In Table II, Goal-oriented and Data-oriented requirements modeling languages are included in the assessment. These languages specialize on perspectives weakly supported by BPM languages. Hence, these modeling languages are used to complement BPM languages. Data-oriented modeling languages (e.g., Entity Relationship Model, Data Flow Diagram, and UML Class Diagram) capture and represent informational entities within a software system. These entities can be streamlined into building blocks (objects, classes, methods, messages, inheritance and the like) used in all phases of development, from requirements to implementation.

Goal-oriented modeling languages (e.g., i* Framework (Yu, 1995), KAOS (Dardenne et al., 1993), TROPOS (Castro, Kolp, & Mylopoulos, 2002)) uses the concept of Goal to reason about requirements and to handle non-functional requirements. A Goal specifies what the system must meet and achieve, but not how. Thus, a Goal is realized by several atomic functions causing state changes on informational entities. At this level, the analyst is primarily concerned with exposing “why” particular behaviors, informational and structural aspects were chosen to be included in the system requirement.

Table II. Modeling capability gaps under Zachman framework

<table>
<thead>
<tr>
<th>Zachman Framework</th>
<th>BPMN</th>
<th>IDEF3</th>
<th>AD</th>
<th>PN</th>
<th>RAD</th>
<th>EPC</th>
<th>i*</th>
<th>KAOS</th>
<th>TROPOS</th>
<th>ER</th>
<th>DFD</th>
<th>CD</th>
</tr>
</thead>
<tbody>
<tr>
<td>What (Data)</td>
<td>-/+</td>
<td>+/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
</tr>
<tr>
<td>How (Functional)</td>
<td>+/+</td>
<td>+/+</td>
<td>+/+</td>
<td>-/+</td>
<td>+/+</td>
<td>+/+</td>
<td>+/+</td>
<td>+/+</td>
<td>+/+</td>
<td>+/+</td>
<td>-/+</td>
<td>-/+</td>
</tr>
<tr>
<td>Where (Network)</td>
<td>+/+</td>
<td>+/+</td>
<td>-/+</td>
<td>-/+</td>
<td>+/+</td>
<td>+/+</td>
<td>+/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
</tr>
<tr>
<td>Who (People)</td>
<td>+/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>+/+</td>
<td>-/+</td>
<td>+/+</td>
<td>+/+</td>
<td>+/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
</tr>
<tr>
<td>When (Time)</td>
<td>+/+</td>
<td>+/+</td>
<td>+/+</td>
<td>+/+</td>
<td>+/+</td>
<td>+/+</td>
<td>+/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
</tr>
<tr>
<td>Why (Motivation)</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
<td>+/+</td>
<td>+/+</td>
<td>-/+</td>
<td>-/+</td>
<td>-/+</td>
</tr>
</tbody>
</table>

Legend: +/+ Notation available / possible to represent; -/+ Notation partially available / possible to represent -/ Notation not available / not possible to represent
AD - Activity Diagram; PN - Petri Nets; KAOS - Goal-Driven Requirements Engineering; ER - Entity-Relationship Model; DFD - Data Flow Diagram; CD - UML Class Diagram

4. THE PROPOSED APPROACH

This section presents the proposed integrative BPM approach which combines key strength of several other modeling languages to adequately represent all the aspect of a business process. Two major processes support the proposed approach. First, is to classify business domain knowledge according to the six facets of Zachman Framework (separation of concerns). The various models are not fully independent; concepts from one model are connected to the other, based on the business process schema. Second, selecting an appropriate modeling language to represents each of the perspectives. The merit of using combination of modeling languages lies in the fact that a specialized language can be complemented with other languages, where the former is lacking.

4.1 SEPARATION OF CONCERNS

This component classifies business domain knowledge according to different modeling perspectives. An effective requirements engineering approach must reconcile the need to achieve separation of concerns with the need to satisfy broadly scoped requirements and constraints. By treating all concerns as equal, we can choose any set of concerns as a base for addressing the problem under investigation. This faceted view of business concepts makes it possible to handle crosscutting concepts (concepts shared by multiple perspectives).

An important component when classifying business concepts is the business domain ontology. This is ontology of business concepts and their relationships, classified according to six modeling facets, namely: Goal, Functional, Informational, Behavioral, Agent, and Location. Based on keywords captured during requirements elicitation, software analyst with the help of the business domain ontology can form operational definitions. Operational definitions are formed by mapping or relating keywords to relevant facets in the domain knowledge ontology, hence adding meaning to the keywords.

Figure 3 shows the architecture for classifying business domain knowledge. The Goal facet includes concepts that describe purpose, objective, intention, qualities and desired states of a business process. A Goal is decomposed into several tasks, which are instances of the Functional facet. The concepts for the Functional facet are identified as verbs, denoting actions or activities that manipulate informational entities. Activities may take some input and transform it into a particular output. The Informational facet captures resources (physical or

http://hipore.com/ijsc
conceptual), which are often identified as nouns or noun phrases that may or may not be coupled with a state descriptor (e.g., available or unavailable, etc.). Behavioral facet is made up of concepts that describe the order in which activities are performed. These include sequences, timing and exception handling events. Location facet groups concepts that denote geographical locations where actions are performed. Additionally they may denote containers where informational entities are stored. Finally the Agent facet classifies concepts that denote responsibility assignments (i.e., human or machine).

4.2 Selecting Modeling Languages

According to Hickey & Davis (2004), software analysts use a particular modeling language for any combination of the following reasons. (1) It is the only language they know. (2) It is their favorite language for all situations. (3) They are following a certain methodology that prescribes that particular language. (4) They understand intuitively that the language is effective in that particular situation. The fourth reason is an ideal level of maturity expected from analysts, but it is not always the case. Analysts do not always understand intuitively available languages and their context of application.

We propose the use of a modeling language ontology that contains knowledge on the key properties of modeling languages. This knowledge will support analysts when selecting an appropriate language for the problem at hand. Figure 4 shows a partial listing for the BPMN descriptor; note that only four facets are shown (Functional, Behavioral, Location, and Agent). This is so because BPMN is lacking in representing Informational and Goal perspectives. The selection is based on how close the language’s properties are to the business concepts identified from a requirements specification. The selection criteria do not fix or restrict any aspect of the process model to any modeling language. The choice is guided by the characteristics of the problem and the solutions domains.

![Figure 3. Architecture for classifying business concepts](image)

![Figure 4. Partial listing of a BPMN descriptor](image)

![Figure 5. Matrix for selecting complementary modeling languages](image)
of modeling perspectives. A separate diagram for a subset of business perspective(s) represented is created per modeling language. The output is a set of models, making up a business process model, incorporated into a requirements specification document to be used in the next phases of software development.

4.3 Managing Inconsistency among Perspectives

We regard an inconsistency as any situation in which two models representing different perspectives do not obey some relationship that should hold between them. This definition covers inconsistencies between models represented in different notations. This definition subsumes logical contradiction, but does not require us to translate specifications into a formal notation to detect inconsistencies. In order to detect inconsistencies, relationships that should hold between different perspectives are to be explicitly stated. The relationships may refer to both syntactic and semantic aspects of the models. They may also be process relationships, in the sense that two models may be inconsistent because they are at different stages of development.

The Zachman framework provides no mechanism to ensure the consistency or interoperability of the models created using the different languages, unless this task is accomplished by choosing a truly integrated family of languages (based on common metamodels). However, Nuseibeh, Kramer, & Finkelstein (1993) and (1994) proposed a flexible framework for expressing the relationships between multiple viewpoints in requirements specifications. This framework encourages multiple representations and is a deliberate move away from attempts to develop monolithic specification languages. It is independent of any particular software development method. Viewpoints are defined as loosely coupled, locally managed, and as distributable objects, which may overlap with each other. These overlaps are the focal point for consistency checking between viewpoints (Nuseibeh, Kramer, & Finkelstein, 2003). A viewpoint template, consisting of five slots, represents viewpoints: a style, a work plan, a domain, a specification, and a work record. The complex relationships between viewpoints should conform to inter-viewpoint rules defined in the work plan slot.

From our observation, the perspectives proposed in this paper can be incorporated into the style slot and work plan slot of the viewpoint template. In particular, the model representation can be defined in the style slot and the separation of concerns and selection of modeling languages can be defined in the work plan. The consistency between models representing different perspectives can be checked through defining the inter-viewpoint check actions in the work plan slot and invoking them when required. Consistency checking is performed by applying a set of rules, defined by the method, which express the relationships that should hold between particular Viewpoints (Nuseibeh et al., 1994). These rules define partial consistency relationships between the different representation schemes. This allows consistency to be checked incrementally between viewpoints at particular stages, rather than being enforced as a matter of course. Furthermore, inconsistencies can be managed within individual perspectives. The modeling languages used in our framework (see Figure 5) are based on reliable ontologies (metamodels with semantic rules), to ensure the required consistency and determine the expressive power of modeling as required by the respective modeling perspective.

5. Case Study: Ordering Material Process

In this Section, we illustrate the applicability of our approach through a process for requesting and ordering material. This scenario is used for training new employees and enforcing company-purchasing standards. The following is a narrative description of the process:

“The first thing we do is request material using a Purchase Request form. Then the Purchasing department either identifies our current supplier for the kind of material requested or sets out to identify potential suppliers. If we have no current supplier for the needed item, Purchasing requests bids from potential suppliers and evaluates their bids to determine the best value. Once a supplier is chosen, Purchasing orders the requested material. Those requesting material must first prepare a Purchase Request. The requester must then obtain the Account Managers approval or that of the designated backup, for the purchase. Purchase Requests submitted for Account Manager approval must include the Account Number for the Project that will fund the purchase. Account Managers or their designated backup, are responsible for, and must approve, all purchases made against their project accounts. After the Account Manager approves the purchase, an authorization signature may be required. To avoid a potential conflict of interest, the requester cannot be the same individual who approves or authorizes the request. Purchase Requests involving Direct projects require an authorization signature whereas Indirect projects do not. Once all the appropriate signatures are in place, the requester submits the signed Purchase Request to Purchasing. Purchasing then orders the requested material and tracks it as a Purchase Order (Mayer et al., 1995, p. 12)”.

http://hipore.com/ijsc
5.1 SEPARATION OF CONCERNS

Figure 6 shows the identified business concepts from the scenario classified according to the different perspectives. Goals identified include: Request material, Order requested material, Authorize purchase request, Select supplier, etc. The “Order requested material” goal is decomposed into three functions (Figure 6) which manipulates the purchase request and the purchase order form. The purchasing department performs these functions. The account manager is responsible for approving purchase requests, and in doing so, they perform two functions, first they check the account number; second they sign the purchase request form if a signature is required.

Figure 6. Partial Classification of business concepts from the Order Material Process

5.2 SELECTING MODELING LANGUAGES

For this process, BPMN is selected as the base language. As already pointed out BPMN is inadequate in representing Informational and Goal perspectives, thus we select IDEF3 and i* Framework to complement BPMN. In the following sections, we illustrate how the selected languages are used to represent the different perspective.

Functional, Behavioral, Location, Agent. We represent the mentioned models using BPMN, mainly for the following reasons. Firstly, BPMN is a process-centric notation, capable of representing different types of processes; Secondly, BPMN supports the representation of multiple cross-cutting processes between different actors, departments and enterprises with the use of lanes and pools. Thirdly, BPMN schema fully supports the representation of events and control flows.

The business process diagram in Figure 7 consists of a network of activities (Functional), order of their performance (Behavioral), responsibility assignments (Agent) and responsibility boundaries (Location). Activities are denoted by rounded-corner rectangles and are connected by unidirectional lines indicating the order of execution. Activities are contained within lanes and pools. Decision points are denoted by gateways, for example, making a decision on whether to request for an authorization signature or not, is depicted by an XOR gateway. An intermediate timing event is also included to signify that purchasing department waits for two days before tracking their purchase order.

Materials Company (container) is represented with pool, and it is divided into four lanes (Requester, Account manager, Authorizer, Purchasing department), representing various agents and their roles. Supplier is external to the Materials Company therefore represented with a different pool. Activities for the Supplier are not shown on the diagram; however, interactions between the two enterprises are denoted by message flows.

Informational. For Informational modeling, we use IDEF3 Object Schematics, which is capable of capturing, managing, and displaying object-centered descriptions of a process. For instance, object transformations and state transitions during process execution. The following objects are identified: purchase request, purchase order, material, and account number. A purchase request form is a key document in this process, and is ultimately transformed into a purchase order via a sequence of activities. Figure 8 shows state transitions traversed by the purchase request form in an occurrence of an activity. From unprepared to prepared following the “prepare purchase request” activity, then from prepared to approved after the account manager authorizes the purchase request. We have changed the Unit of Behavior (UOB) symbol in IDEF3 to match the activity symbol supported by BPMN. Furthermore, we changed the round circle object symbol, to avoid a conflict with an actor symbol in i* modeling framework discussed next.

Goal. To represent underlying motivations and intentions behind activities and flows in other process models, we use Strategic Dependency model presented in i* modeling framework. Figure 9 shows Strategic Dependency model of the Materials Company. Nodes represent actors or process participants; these are denoted as lanes in the business process diagram in Figure 7. Links between two actors indicates that one actor depends on the other in order for the former to achieve a particular goal. For instance, Requester’s intention is to request material. In the process of requesting material, they depend on the Account Manager and Authorizer to approve and authorize the purchase request respectively. The process description did not explicitly state goals, objectives and or quality measures; hence, we make some assumptions during modeling (see Figure 9).
6. RELATED WORK

6.1 EVALUATING DIFFERENT BPM LANGUAGES

Although some efforts have been made towards developing a framework for assessing BPM languages, they are still fragmented and have limited coordination with the overall business process meta-model. Furthermore, a number of studies have been published on the evaluation of BPM languages. List & Korherr (2006) proposed a similar evaluation framework to the one proposed in this study. Their process meta-model is categorized according to the perspectives proposed by Curtis et al. (1992), which are of Organizational, Functional, Behavioral, and Informational perspectives. These perspectives can be mapped to four models in Zachman framework. However, the Goal and Agent perspectives are not explicitly evaluated.

Giaglis (2001) presented a taxonomy of business process modeling and information systems modeling languages. This classification is to assist decision makers in comparatively evaluating and selecting suitable modeling languages. The taxonomy classifies modeling languages into four facets: informational; organizational (where, who); behavioral (when, how) and functional.

http://hipore.com/ijsc
6.2 Existing Multi-Perspective Frameworks

The concept of multi-perspective models has long been recognized in the information systems discipline. General purpose modeling languages such as UML (Rumbaugh et al., 1999) dates back to the mid-1990s. UML in principle supports modeling of a wide range of application domains in object-oriented software systems. However, it does not provide sufficient concepts and graphical representations that are appropriate for all business perspectives (Frank, 2002). UML suffers from several shortcomings (see Mendling (2009) for an evaluation).

A theory of maximum ontological completeness (MOC) and minimum ontological overlap (MOO) based on observations of how users combine modeling languages is developed in Green, Rosemann, & Indulska (2005). This theory is useful in understanding how users utilize modeling languages to obtain maximum expressiveness from specific languages. The evaluation carried out for this theory imply that in practice analysts combine different modeling languages, hence, an integrative approach to guide this practice is needed.

Frameworks such as CIMOSA – Computer Integrated Manufacturing Open System Architecture (Kosanke, 1995), and ARIS – Architecture of Integrated Information Systems (Scheer, 2000) offer multi-perspective frameworks for information system architectures. ARIS specifies four views for business process modeling (Function, Organization, Data and Output), which are similar to the ones proposed in CIMOSA (Function, Information, Resource and Organization). However, these frameworks lack or did not specify modeling languages to represent their particular views. Although EPC is suggested in ARIS, our evaluation shows that EPC is inadequate in representing What, Who and Why models.

A similar approach to the one presented in this paper is MEMO – Multi Perspective Enterprise Modeling framework (Frank, 2002). This framework offers a set of visual modeling languages for describing corporate strategies, business processes, resources or information models. MEMO offers three specialized languages that support the construction of these models. The languages further support the representation of various interrelated concepts within the models. A downside to this initiative is that, it introduces new modeling languages. Given the abundance of process modeling languages, software engineers must do more than just create new languages. According to Scheer (2000), only a few of process modeling languages created are widely accepted in the BPM community. Therefore, an understanding of modeling capabilities and gaps of existing languages is required before inventing new languages. Furthermore, ways on how these gaps can be bridged need to be devised.
The approach presented in this paper goes further than just identifying modeling perspectives. It promotes the leveraging of existing modeling languages through suggesting them for modeling the identified perspectives. Furthermore, the approach acts as a guide for selecting appropriate modeling language(s) for representing different perspectives of a business process. The selected modeling language is then complemented with other languages where the former is lacking. We do not advocate for a single modeling language that represents all modeling perspectives and still be applicable in all modeling situations. Such a language, according to Curtis et al. (1992) and Giaglis (2001), would generate complex models resulting on difficulties in communication. Rather, we argue for separation of concerns and the alignment of concerns with the business process schema. Thus, a specialized language can effectively represent each concern. For this purpose, a comprehensive business process schema is required. The emphasis for our approach is on creating comprehensive models, which entail a common understanding of concepts regardless of how they are represented.

6.3 Tools to Support Multi-Perspective Frameworks

A typical software development task typically requires a combination of modeling languages, which may not require the use of several modeling tools, which in turn calls for model interchange and cross-consistency checking capabilities. However, some developers have built tools that support multi-perspective frameworks in modeling. Thus, such tools can support a framework such as proposed in this paper.

Grundy, Hosking, & Mugridge (1998) proposed a software architecture, Change Propagation and Response Graphs (CPRGs), for managing inconsistencies in multiple view software development tools. This environment provides mechanisms for developers to configure inconsistency detection, allowing developers to choose the most suitable approach to inconsistency management for different artifacts, tools and process stages. Ptech, Inc. (2001) developed a methodology-independent integrated modeling environment called FrameWork. It is based on an object-oriented data structure whose semantics is fully defined via metamodels. Being a methodology neutral, FrameWork is claimed to have wide applicability (such as to be able to capture, analyze and design data and link it to organizations, activities, locations, etc.). FrameWork offers support for the Zachman framework via the Zachman framework Control Panel, containing a template and most of the model types required.

6.4 Limitations of the Proposed Approach

A prevailing concern for the proposed approach is the interoperability of data between multiple perspectives, which can be represented by different notations on different tools or even on different machines. This concern is augmented by the syntactic and semantic differences of information to be exchanged between perspectives. Integrating languages requires common concepts. The higher the level of semantics that is provided by those common concepts the tighter the integration (Frank, 2002). For instance: If two languages share a common notion of an integer, they are less integrated than two languages that share a common notion of an application level concept - because in the second case the chances for deviating interpretations are clearly lower. Our objective is to use modeling languages that conform to the Meta-Object Facility (MOF) standard developed by the Object Management Group (Kleppe, Warmer, & Bast, 2003). Hence, from a formal point of view, it will be easy to define common concepts for a set of modeling languages. However, metamodels for modeling languages such as KAOS, i*, IDEF, etc, are not yet defined in MOF standard.

Requirement modeling research is done in different communities and there is no common platform for agreeing on the definitions for notations. As a result, different models are created and managed autonomously by heterogeneous tools. There is a need for a holistic approach towards modeling that would allow different modeling perspectives to share relevant information. However, with the advent of frameworks such as Model-Driven Architecture (MDA) (Kleppe et al., 2003) a significant progress has been made towards developing common modeling standards, of which our integrative approach can benefit from.

7. Conclusions

This paper presents an integrative approach to support multi-perspective BPM, which is guided by the six models of Zachman Framework (What, How, Where, Who, When, Why). The Zachman Framework is known to be comprehensive for representing different business perspectives and its models are universal in conceptual modeling. A comprehensive business process model should incorporate information such as what activities are being performed; what informational entities are being manipulated; when and how are they performed; where and by who are they performed; finally, why are they performed. The comprehensiveness of a BPM language is thus judged on how well it represents the above-mentioned perspectives.
The contribution made by this paper is twofold. First it provides an evaluation of commonly used BPM languages against Zachman Framework. The evaluation results show that BPM languages in their current scope are inadequate to represent the Informational and Goal perspectives. Second, rather than extending or inventing new modeling languages, the paper presents an integrative approach that uses existing informational and goal-oriented languages to bridge the identified modeling gaps. The main benefit of the proposed approach is its integration of existing modeling languages that are familiar to business analysts, rather than introducing new languages. The proposed approach provides guidance for business analysts and helps them to select appropriate modeling languages.

In acknowledging that not all software analysts understand intuitively what modeling language(s) to use for problems at hand, we intend to further our work by developing a language selection tool. The tool will support the matching of business problems to appropriate modeling language(s), hence, assisting analysts in making informed decisions. The intention is not to restrict any perspective to a particular language, but rather, select appropriate languages for each modeling perspective based on the nature of the problem at hand.

8. REFERENCES


Authors

Keletso J. Letsholo is a doctoral student in the Software Systems Research Group at the University of Manchester, UK. His current research investigates ways of automating the process of building traceable analysis models from a textual requirements specification written in English. His grand vision is to contribute to the continuing quest for better ways to improve our ability to develop and maintain analysis models.

Keletso received a BSc in Computer Science from University of Botswana, and an MSc in Computer Science (Software Engineering) from the University of Wollongong, Australia.

http://hipore.com/ijsc
Erol-Valeriu Chioasca is a PhD candidate in the School of Computer Science at the University of Manchester. His main research area is in Software Engineering, specifically in Requirements Engineering. His current research focuses on using natural language techniques to support automated requirements modeling and analysis.

Dr. Liping Zhao is an academic member of School of Computer Science at the University of Manchester. Her research focuses on the discovery and development of reusable software patterns, symmetries and models, and the application of these fundamentals to model-driven development and service-oriented computing. A pioneer in service sciences, she co-funded and led the first academic network on service sciences in the UK. Her contribution to software patterns and the service sciences agenda has earned her three IBM Faculty Awards.