AN ARTIFACT-BASED ARCHITECTURE FOR A BETTER FLEXIBILITY OF BUSINESS PROCESSES

Mounira Zerari and Mahmoud Boufaida
Mentouri University, LIRE Laboratory, Constantine, Algeria

Keywords: Process flexibility, Underspecification flexibility, Process mining, Artifacts, Context execution.

Abstract: Workflow management Technology has been applied in many enterprise information systems. Business processes provide a means of coordinating interaction between workers and organization in a structured way. However, traditional information systems struggle with requirement to provide flexibility due to the dynamic nature of the modern business environment. Accordingly, Adaptive Process Management Systems (PMSs) have emerged that provide some flexibility by enabling dynamic process change during run time. There are various ways in which flexibility can be achieved. One of these kinds of flexibility is flexibility by underspecification. This kind of flexibility is not supported (except YAWL) by current products. In addition, all approaches that currently exist not consider the context of execution of business process management. In this paper we propose an approach that supports flexibility by underspecification and consider context of the business process execution in runtime environment. The main idea is to consider activities as independent part of process. Each activity is encapsulated in an entity (artifact). The decision of which activity (module, component) will be executed depends on context environment and conditions execution. We will reason about the decision taken. We are motivated by make business processes easy to put together from reusable components and to reason on context execution.

1 INTRODUCTION

Today, most enterprise applications include a Workflow management technology. It is clear that the economic success of an organisation is highly dependent on its ability to react to changes in its operating environment. So, it is increasingly necessary for enterprises to streamline their process to improve performance.

Current systems are based on models which they tend to be rigid in format and are not able to easily include either foreseen or unforeseen changes in the context of environment in which they operate.

To this end, the notion of flexibility has emerged as a pivotal research topic in Business Process Management (BPM). In this context, process flexibility can be seen as the ability to deal with both foreseen and unforeseen changes, by varying or adapting those parts of the business process that are affected by them (Schonenberg, H et. al., 2008). Or, in other words, flexibility denotes the capability to reflect externally triggered change by modifying only those aspects of process that need to be changed, while keeping the other part stable (Mulyar N.A et .al.,2006).

Different kinds of flexibility are needed during the BPM life cycle of a process. A range of approaches to achieve process flexibility have been identified. These approaches have been described in the form of taxonomy (Schonenberg, H et. al., 2008).In this paper we propose an approach that focuses on one kind of flexibility. It is flexibility by under specification. This category of process flexibility is the ability to execute an incomplete process model by completing it at runtime, via selection from a pre-defined set of process fragments. The idea is to encapsulate these fragments into entities or components. These components possess interface for communication and a manual or operating instructions. This notion is inspired from artefacts in coordination systems in System multi-agent SMA. The different components of this artefact will be detailed in the following of paper. Based on information collected from the runtime helped by techniques of process mining a decision will be taken to execute one of process fragment. A knowledge base is updated when a

Zerari M. and Boufaida M. (2010).
AN ARTIFACT-BASED ARCHITECTURE FOR A BETTER FLEXIBILITY OF BUSINESS PROCESSES.
In *Proceedings of the 12th International Conference on Enterprise Information Systems - Databases and Information Systems Integration*, pages 359-365
DOI: 10.5220/0002870503590365

Table 1: Evaluation product.

	ADEPT1	YAWL	FLOwer	Declare
Flexibility by design	+	+	+	+
Flexibility by Deviation	-	-	+	+
Flexibility by underspecification				
Late biding	-	+	-	-
Late modelling	-	+	-	-
Static, before placeholder	-	-	-	-
Dynamic before placeholder	-	-	-	-
Static, at placeholder	-	-	-	-
Dynamic, at before placeholder	-	+	-	-
Flexibility by change	+	-	-	+

decision is taken. This approach permits the consideration of information collected on trace (log) execution. Also, it makes business processes easy to put together from reusable components.

The remainder of this paper is organised as follows. Section 2 provides background information on taxonomy of process flexibility, process mining and problem statement. In section 3 we present some definitions of preliminaries concerned several notions used in the proposed approach. Section 4 describes the overall of the proposed approach. Section 5 presents in more details the components of architecture. Finally, we conclude the paper and identify opportunities for future work in section 6.

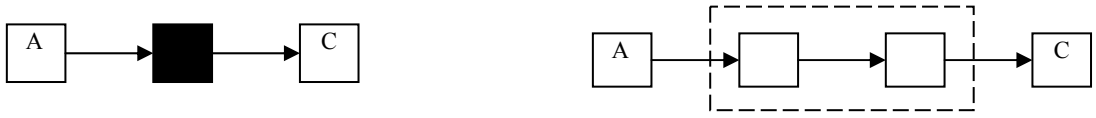
2 PROBLEM STATEMENTS

Several research works have explored the possibility to make BPMs more flexible. Many approaches tend to achieve adaptability, like ADEPT, WASA, or Milano. The basic idea behind these approaches is to deal with dynamic change to fit with changed real world situations. More precisely, a range of approaches to achieve process flexibility have been identified. They can be taken to facilitate flexibility within a process. All of these strategies improve the ability of business processes to respond to changes in their operating environment without necessitating a complete redesign of the underlying process model; however they differ in the timing and manner in which they are applied. Moreover they are intended to operate independently of each other (Schonenberg, H et. al., 2008). Indeed, in one side, each approach of those approaches is interested by one kind of business flexibility (table 1). In another side; they are not to use to actually learn from the change.

In Table 1 evaluation results are depicted, which shows whether a system provides full (+), partial(+/-) or no support (-) for an evaluation criterion. For the full description of evaluation criteria and detailed evaluations for each of the offerings, we refer readers to the associated technical report (Mulyar N.A et .al.,2006).

We can see that all evaluation criterions are supported by several systems. The selected systems cover distinct area of the PAIS technology spectrum, such as adaptive workflow (ADEPT1), case handling (FLOWER) and declarative workflow (Declare). However, we focus on “flexibility by underspecification” criterion evaluation. It is supported by only YAWL which is a more recent initiative based on formal foundations. Flexibility by Underspecification is the ability to execute an incomplete process model at run-time, i.e., one which does not contain sufficient information to allow it to be executed to completion.

Let us notice that this type of flexibility does not require the model to be changed at run time; instead the model needs to be completed by providing a concrete realisation for the undefined parts (Schonenberg, H et. al., 2008). An incomplete process model contains one or more so-called *Place-holders.* Place-holders are nodes which are marked



a) Before realisation b) After realisation

Figure 1: Specification by underspecification.

as underspecified (their content is unknown) and whose content is specified during the execution of these placeholders.

Figure 1 (a) shows an incomplete process model with a placeholder task between A and C. Figure 1 (b) illustrates the realisation of the placeholder b a process fragment from a linked repository of process fragments.

The main question now is *witch process fragment is chosen for the execution of place holders at run time?* I.e. how process fragment are selected from the repository and under which condition? Who take this decision? And how we learn from those decisions?

We consider that the conditions under process models are executed are essential for the flexibility of processes. Indeed, the selection of one process fragment from other one depends on context of business environment execution. In this paper we propose an approach that considers context information in order to execute tasks not defined (Placeholders) in run time execution. The following section will describe the different concepts and notions linked to our approach. We have to use some notions that are presented in the next section.

3 DEFINITIONS OF THE BASIC CONCEPTS OF OUR APPROACH

This paper is based on the integration of two existing technologies: process mining and artifact. This section gives background information needed to understand the implications and leverages of their combination.

3.1 Process Mining

The goal of process mining is to extract information (e.g., process models, or schemas) from these logs. Process mining addresses the problem that most "process owners" have very limited information about what is actually happening in their organization. In practice there is often a significant gap between what is predefined or supposed to happen, and what actually happens. Only a concise assessment of the organizational reality, which process mining strives to deliver, can help in verifying process schemas, and ultimately be used in a process redesign effort (Van Dongen, B.F et. al., 2004).

The idea of process mining is to discover, monitor and improve real processes (i.e., not assumed processes) by extracting knowledge from event logs (e.g., in MXML format).

3.2 Process Log

Information systems typically log all kinds of events. Unfortunately, most systems use a specific format. Therefore, an XML format for storing event logs is proposed (Gunther, C.W et. al., 2007).

The basic assumption is that the log contains information about specific tasks executed for specific cases (i.e., process instances).

The XML format is roughly the following:

The root element is the <WorkflowLog> element and it has a number of <Process> sub-elements, each encapsulating execution data of one process, orworkflow. A <Process> element has a number of <ProcessInstance> child elements.

A <ProcessInstance> has numerous <AuditTrailEntry> child elements.

An <AuditTrailEntry> represents one log record and contains an identifier of the activity, the event-type and a timestamp.

4 AN ARCHITECTURE BASED ON ARTIFACT FOR BUSINESS PROCESS FLEXIBILITY

Process flexibility by underspecification raises a number of interesting questions, as indicated in the previous section. In this paper we propose an approach in order to reach process flexibility by underspecification. We focus on the manner that a process fragment will be chosen in order to execute a place holders in run time environment. Our objective is to take into account the information, on the one hand, from context execution. On the other

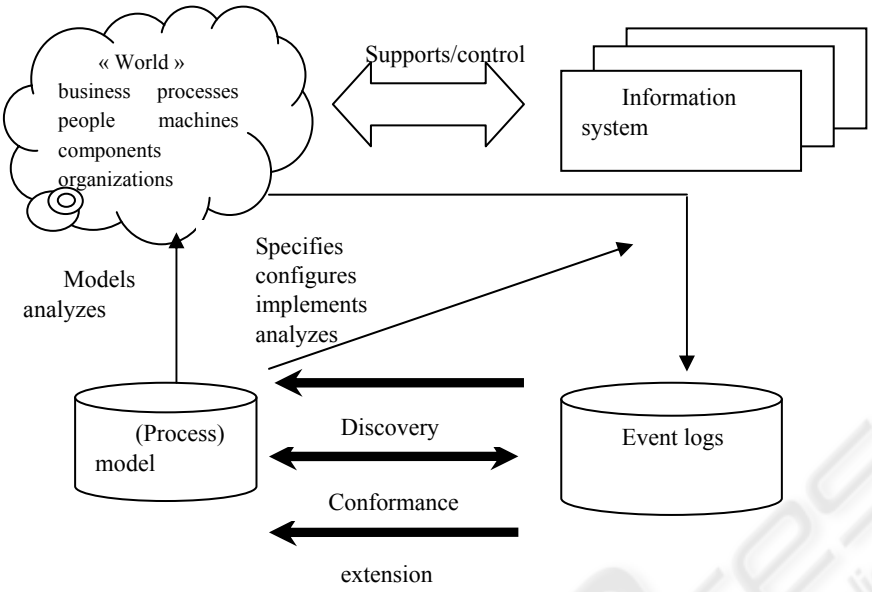


Figure 2: Overview showing three types of process mining: (1) Discovery, (2) Conformance, and (3) Extension (Gunther, C.W et .al., 2007).

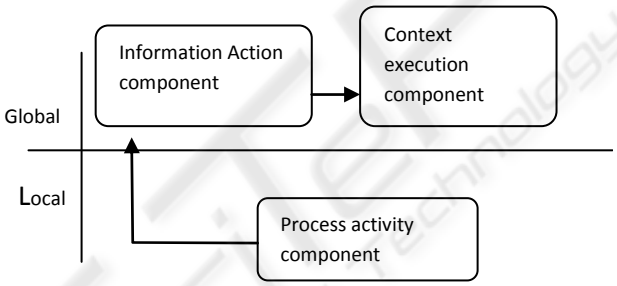


Figure 3: Different views of system.

hand we must learn from this decision in order to apply it to other situations (Static, dynamic before place holders). In a run time environment, several execution contexts exist. Each context execution requires a specific task under specific condition. We started our discussion by this. Indeed, the actions that comprise an activity of a business process are always the same and do not change. However what is changing is the execution context under which the decision to perform a particular activity should be execute.

The basic idea is to encapsulate the activity actions executed in reusable translucent boxes with a "manual". Those boxes are considered as entities used by manager service to achieve the goal of an activity (place holder). Each entity is characterized by: interface to use (actions and perceptions), function (description of services rendered), its attributes (parameters and internal variables exhibited) and Operating instructions (instructions).

Let us consider the system with two viewpoints (figure 3):

- *The local: the view of the action*

It is the set of instructions that compose an activity. It acts directly and at exact moment without any vision of the future.

- *The global: the view of the context activity*

It is the position of the activity in its context.

Communication local-global: local sent to global information perceived by the action.

The system architecture is constituted of several components:

- Process activity component: instruction of a process activity is encapsulated in the artifact as mentioned above.
- Information Action component: we can consider it as a management service. The decisions of what activity will be executed are taken by this component. A knowledge base is update in order to learn from the situation.

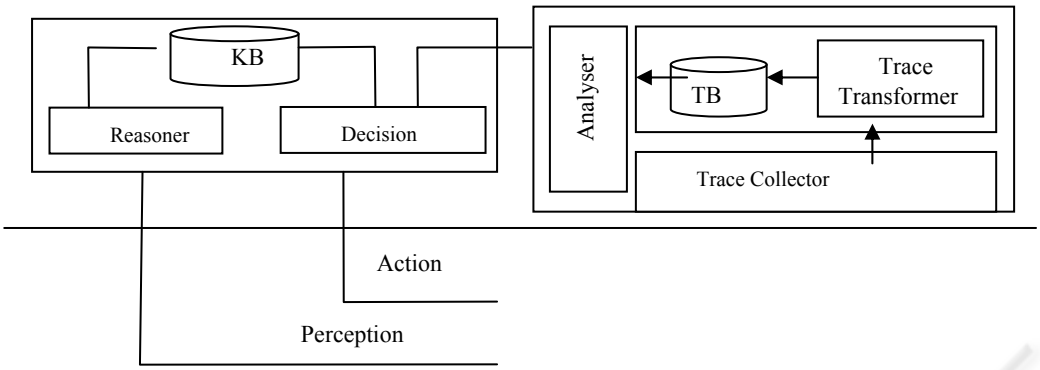


Figure 4: Architecture of the proposed system.

- Context execution component: the decision taken by precedent module is taken according to information context execution. The objective of this entity is the perception of condition context. That information is communicated to action information in order to consider theme when the process activity module is chosen (place holder).

Each component is itself composed of several modules and each has its own task. Those modules are more described in the next section.

5 DESCRIPTION OF DIFFERENT COMPONENTS

Figure 4 illustrates the position (layout) of different modules of architecture. The functional description of each one will be presented in this section.

- Trace Collector: it is a monitoring module. It permits us to capture different situation on execution of processes. Here our support is event log of the execution of process activity. Event log are created, which record the sequence of activities executed for each case.
- Trace Transformer: in order to use this event logs XML format, called the Mining XML (MXML) format that could be used as input to the different tools could be exploited. By converting simulated or real-life logs to the MXML format, one could use the mining techniques in multiple contexts. The result will be saved in Trace Base (TB).
- Analyzer: Process mining targets the automatic discovery of information from an event log.

This discovered information can be used to deploy new systems that support the execution of business processes or as a feedback tool.

- Process activity component: we mentioned above that the activity of a process is encapsulated in an entity considered as a component. This component is largely inspired by artifact in Coordination in MAS (Dinont C. et. Al., 2006)

Thus, this entity has:

- Use interface: defined as a set of operations. Two kinds of operations: execution of an action and the perception of the end of an action (figure 4).
- Function: it describes the service proposed by the entity. Aim of activity. Specification: formal description of the behaviour of the activity.
- Manual: it's a set of formal instructions witch describe the manner that other component (decision maker and reasoner) use this entity. Those instructions are described by a formalism based on process algebra. For more details you can refer to (Viroli, M., Omicini, A, 2007). Nevertheless a short description of principles can be presented. An Instruction is:

$$I ::= 0 \mid !\alpha \mid ?\pi \mid I; I \mid I + I \mid (I||I) \mid D(t1, \ldots, tn)$$

I can be an atomic Instruction: Behaviour ZERO (0), execution of the action !α , and the perception of the end of the action ?π. I may also be structured using different operators: ";" for the sequential composition, "+" for the choice and "| |" for parallel composition. The concept of recursive can be assured by the invocation of D (*t*1, . . . ,*tn*) of another basic instruction. We can consider the following manual as:

((!a ; ?end_ a) + (!b ; ?end_ b)) || (!c ; ?end_c) ....(i)

We have to define the use interface of our entity. It's a basic interface. We have two kinds of operations: Action and perception. Those operations are derived from the precedent component (i.e. analyzer).

Those actions are extracted and inspired from events of a process activity logged whenever an activity is executed. For example this sequence is logged whenever an activity is executed without any exceptions or complication:

```
Shedule—Start—Complete.
```

Furthermore, there are a lot of different special cases. For example, an activity may be cancelled while being in state "Scheduled". The order of events is:

```
Schedule--Withdraw.
```

An activity may be cancelled while running, i.e., it is in state Active. Such a sequence is mapped into

```
Schedule--Start--Abort.
```

We can resume all this in a FSM (Fig. 5 shows this FSM Dinont, C and Mathieu, P, 2006)).

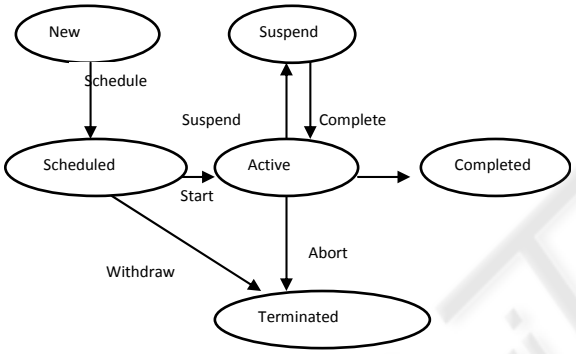


Figure 5: A FSM describing the event type.

So, we can deduce action operations and perception ones. Possible action operations are: start(input data), suspend, resume, complete, and possible perception operations are: `completed(result), Terminated, suspended.`

The next step now is the description of the manner that this entity will be used by the reasoner. Hence, we describe the manual that will be executed in order to use this entity. (i) is an example of a such manual.

The different activities of a business process are expressed by this manual. For example:

```
   Simple_staffware:=
((!registration ; ?end_registration) ;
   (!send_questionnaire;
?end_send_questionnaire);!receive_quest
ionnaire;
?end_receive_questionnaire);(!evaluate
;?end_evaluate) ;simple_staffware..(ii)
```

This manual or operating instruction is used in order to be exploited by reasoner and decision maker for reasoning. It's more detailed in a next point.

- Reasoner and decision maker:

The essential element necessary for programming the using of the entity (artifact) is how to use the artifact (the operating instructions) as in the previous example. In order to assure the semantic link with knowledge base (KB) of reasoner the operating instructions given alone are of no utility. We need additional information to enable it to decide what actions they will launch on artifacts and understand the perceptions that it will receive in return. This is done using semantics associated with mental actions and perceptions (Dinont, C and Mathieu, P, 2006). It is defined in a table which indicates for each action the mental state in which the component must decide to initiate this action and for each charging the changes in mental status. For illustration we take the manual (ii) mentioned above. We have to indicate to decision maker when he may decide to perform the action:

Action	Precondition	Perception	Effect
Registration	¬ registration	End_registration	registration

The effects permit to update the knowledge base. This principle will be applied to all the operating execution. The module decision maker will be referred to analyser to know what the operating instruction for this artifact is.

6 CONCLUSIONS AND ENGOING WORK

In this paper, we presented architecture of a system for increasing flexibility of business process management. We focused on flexibility by underspecification. This kind of flexibility is not supported by the entire product except YAWL. YAWL use worklet to achieve process flexibility. They also not consider execution environment of processes. Our approach uses the notion of artifact to encapsulate the functionalities of activities of business process. It permits us to consider activities first-class programming abstractions. They can also be instantiated, modified, and disposed dynamically. Our approach integrates the notion of semantic due to process algebra. We also reason on this artifact and on execution environment. The informations are extracted from the context of execution using

process mining techniques. Process mining permit to transform execution log to Mining XML format. This format will be exploited to take the decision witch artifact will be executed. Our approach is also a new way to look to BPMs specially environment of execution of them. In this paper we have outlined a system that would need prototyped his different components. Our ongoing works will be: first, implementing different components applied to a case study. We have also to complete and extend the basic model of artifacts. An extension of reasoning mechanism can be developed. A mapping between a Petri net (MXML by process mining tool) and process algebra will be interesting.

Finally, we mention that the main advantages of this approach are the exploitation of the formal side of the process algebra and Petri nets for verification and validation purpose.

REFERENCES

Alves de Medeiros, A., Guzzo, A., G, G. Greco, Wil M. P. van der Aalst, A. J. M. M. Weijters, Boudewijn F. van Dongen and Saccà, D. ,2007. Process Mining Based on Clustering: A Quest for Precision. *Business Process Management Workshops 2007: 17-29*

Dinont, C., Mathieu,P., Druon, E., and Taillibert, P., 2006. Artifacts for time-aware agents. *AAMAS 2006: 593-600*.

Dinont, C., Mathieu, P., Druon, E., 2006. Les artifacts de calcul Une solution aux délibérations longues. *JFSMA 2006.*

Gunther, C. W., Rinderle-Ma, S., Reichert, M., van der Aalst, W. M. P. and Recker, J. (2007) 'Using Process Mining to Learn from Process Changes in Evolutionary Systems', *Int. J. Business Process Integration and Management, Vol. 1, Nos. 1/2/3, pp.111–111*.

Mulyar N. A., M. H. Schonenberg, R. S. Mans, N. C. Russell and W. M. P. van der Aalst. Towards a Taxonomy of Process Flexibility (Extended Version). *BPM Center Report BPM-07-11, BPMcenter.org, 2007.*

Rozinat, A., van der Aalst, W. M. P.2006, Decision Mining in ProM. *Business Process Management 2006: 420-425.*

Rubin,V., Günther, C. W., Wil M. P. van der Aalst, Ekkart Kindler, Boudewijn F. van Dongen, Schäfer,W.,2007. Process Mining Framework for Software Processes. *ICSP 2007: 169-181.*

Schonenberg, H., Mans, R., Russell, N., Mulyar, N., and Wil M. P. van der Aalst, 2008. Process Flexibility: A Survey of Contemporary Approaches. *CIAO! / EOMAS 2008: 16-30.*

Viroli, M., Omicini, A., and Ricci, A., 2007. Agent & Artifact (A&A) ARTIFACT-BASED ENVIRONMENT FOR MAS. *Seminar at LIP6 Paris, March 2007*.

Viroli, M., Omicini, A. 2007. ReSpecT Nets: Towards an Analysis Methodology for ReSpecT Specifications. *Electr. Notes Theor. Comput. Sci. 180(2): 123-144 (2007).*

Van Dongen, B. F., Wil M. P. van der Aast, Multi phase Process mining : building instance graph. *ER 2004, LNCS 3288, pp. 362–376, 2004.*

Wil M. P. van der Aalst, Boudewijn F. van Dongen, Joachim Herbst, Laura Maruster, Guido Schimm, A. J. M. M. Weijters 2003. Workflow mining: A survey of issues and approaches. *Data Knowl. Eng. 47(2): 237-267 (2003).*

Wil M. P. van der Aalst, Adams, M., Arthur H. M. ter Hofstede, Pesic,.M., Helen Schonenberg, H., 2009. Flexibility as a Service. *DASFAA Workshops 2009: 319-333.*

Weber, B , Reichert, M., Rinderle-Ma, S., Wild, W., 2009. Providing Integrated Life Cycle Support in Process-Aware Information Systems. *Int. J. Cooperative Inf. Syst. 18(1): 115-165 (2009).*