Moving Across Paradigms between the Process Design and Enactment Phase in Enterprise Information Systems

Filip Caron and Jan Vanthienen
Leuven Institute for Research on Information Systems (LIRIS), Faculty of Business and Economics, KU Leuven, Naamsestraat 69, 3000 Leuven, Belgium


Abstract: While the business process management literature often assumes a single approach (e.g. procedural or event-driven) over the process lifecycle, a transition between approaches at different phases in the process lifecycle may significantly reduce the impact of intrinsic trade-offs between process characteristics. This position paper explores several business process strategies by analyzing the approaches at different phases in the process lifecycle as well as the various transitions.

1 INTRODUCTION

Organizations face a continuous pressure to improve process compliance, flexibility, efficiency and effectiveness. While responding to these pressures individually can be demanding, the real challenge is dealing with the intrinsic tradeoffs (e.g. between compliance, efficiency and flexibility). A wide spectrum of business process management paradigms has been presented in the literature, each with its specific trade-offs. Additionally, most of these contributions consider the selection of the optimal approach given the business environment as a one-time choice at the process design phase. However, business processes may also require different tradeoffs at different phases of the process lifecycle.

The contribution of this position paper will be the exploration of various business process strategies that combine the selection of a design-time paradigm and a run-time paradigm (i.e. position selection) with a transition path, resulting in a better fit between the business processes and the business environment. Rather than making a value judgement of the different strategies, we will focus on a discussion of the impact on the process characteristics.

The remainder of this paper is structured as follows: first we introduce and assess the different positions (section 2 & 3), followed by the specification and analysis of the transition strategies (section 4) and section 5 concludes the paper.

2 THE BUSINESS PROCESS PARADIGMS AND LIFECYCLE

Each position in the process lifecycle is defined by the combination of the related lifecycle phase and process paradigm.

Traditionally, the traditional business process lifecycle consists of four phases with distinct roles (Weske, 2007), i.e. process design phase, process implementation phase, process enactment phase and process evaluation phase. The design phase and the enactment phase straightforwardly belong to respectively the design-time and run-time. As the process analysis and evaluation phase often results in specific recommendations/requirements for a process redesign, it can be considered as part of the next cycle’s design-time. The implementation phase can be associated with run-time as it is the implemented process that will be executed.

The business process management literature has proposed a wide spectrum of process paradigms with at the extremes the procedural (Zisman, 1977; OMG, 2006; Ellis and Nutt, 1993) and declarative (Goedertier and Vanthienen, 2009; van der Aalst et al., 2009b; Swenson, 2010) paradigm. In between there exists a wide variety of hybrid paradigms that combine aspects of both extremes (Sadik et al., 2005; van der Aalst et al., 2009a; Schonenberg et al., 2008; Kumar and Yao, 2009; Hallerbach et al., 2010).
Moving Across Paradigms between the Process Design and Enactment Phase in Enterprise Information Systems

3 ASSESSING DESIGN-TIME AND RUN-TIME POSITIONS

Since every position has its own design or execution principles, all positions have different characteristics and consequently appeal to different business requirements. In (Davenport, 1993) Davenport identifies four desirable qualities for business processes: process flexibility, compliance, effectiveness and efficiency. This section analyzes the possible impact of each position’s principles on these characteristics.

- **Process flexibility** is the extent to which an organization can deal with business process change, the ability to accommodate the special needs of particular business process instances (i.e. runtime flexibility) as well as to accommodate process model evolutions (i.e. maintainability).
- **Process compliance** is the extent to which a process is in correspondence with the internally defined business rules and the externally imposed business regulations.
- **Process effectiveness** is the extent to which a business process realizes its business goals.
- **Process efficiency** is the extent to which the organization of the business process is capable of minimizing the amount of utilized resources such as personnel, materials, time, machine capacity.

Additionally, two important characteristics of process modeling languages are taken into account for the evaluation of the design-time positions:

- The **expressibility** of a process modeling language is determined by its ability to express specific process elements, e.g. control-flow, data, execution and temporal information (Lu and Sadiq, 2007; zur Muehlen et al., 2007)
- The level of **comprehensibility** reflects the ability of a process modeling language to define understandable process models that can be easily communicated among various stakeholders (Fahland et al., 2009a).

Tables 1 and 2 represent the assessment results of respectively the design-time and run-time positions.

4 DESIGN-TIME TO RUN-TIME TRANSITIONS: DEFINITION AND ASSESSMENT

Traditional business process management solutions are oriented towards a single process paradigm, e.g. the business processes are modeled using procedural process modeling languages (such as BPMN (OMG, 2006)) and then executed in a procedural enactment environment (such as BPEL (OASIS, 2007)). These design-time to run-time transitions within a single process paradigm are the same paradigm transitions. There also exist business processes that have different requirements at design-time and run-time, e.g. in terms of flexibility. In these cases we analyze design-time to run-time transitions between process paradigms, the cross paradigm transitions.

Different process lifecycle phases may require different tradeoffs between process characteristics. Consequently, cross paradigm transitions may in practice result in a better fit between the business process aware information systems and the business environment.

4.1 Same Paradigm Transitions

**Procedural - Procedural Transition.** Different transformation strategies between procedural process modeling languages and procedural process execution languages have been proposed in the literature (Decker et al., 2008; Ouyang et al., 2009). However, due to a conceptual mismatch between the standard procedural process modeling and execution languages, translation techniques are generally only offered for a core subset of the procedural process modeling constructs (Recker and Mendling, 2006).

**Impact on Process Characteristics.** A set of research contributions on process variant management (based on querying process variants before run-time) falls into the scope of the procedural-procedural transition, e.g. (Lu et al., 2009). While this procedural-procedural transition might have a positive impact on the process flexibility, managing changing compliance requirements can be challenging for repositories with large collections of process variants. The impact on process efficiency and effectiveness depends on the quality of the process variants and the number of different situations captured by the process variants in the repository. In (Reichert et al., 2009) a technique is presented for adding, replacing and moving activities.

**Example Process Case.** The procedural-procedural transition can be recommended for business processes in a stable environment with predictable execution paths, e.g. processing standard and static items (such as online orders).

**Declarative - Declarative Transition.** The common declarative process modeling languages have their roots in formal logic. Due to this formal foundation the translation of a high-level declarative process
Table 1: Design-Time Positions Analysis.

<table>
<thead>
<tr>
<th>Description</th>
<th>Procedural process modeling</th>
<th>Declarative process modeling</th>
<th>Hybrid process modeling</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Impact on process characteristics</strong></td>
<td><strong>Positive</strong>: Comprehensible models, process efficiency</td>
<td><strong>Positive</strong>: Flexibility, compliance by design, maintainability (traceability and absence of duplication), high expressibility (Lu and Sadiq, 2007)</td>
<td>Combination of procedural and declarative constructs resulting in process models with placeholder activities or procedural process models with rule-based adaption</td>
</tr>
<tr>
<td></td>
<td><strong>Negative</strong>: Overspecification (Sadiq et al., 2005), maintainability issues (Fahland et al., 2009b)</td>
<td><strong>Negative</strong>: Limited comprehensibility (large and unstructured sets of formal rules) (Fickas, 1989)</td>
<td><strong>Moderation of impact</strong>: Improved flexibility and reduced maintainability issues (declarative placeholder activities), Improved efficiency and reduced flexibility (procedural placeholder activities), improved flexibility (rule-based adaption)</td>
</tr>
<tr>
<td><strong>Languages examples</strong></td>
<td>Petri Net (based) modeling (Zisman, 1977; Ellis and Nutt, 1993), BPMN (OMG, 2006) and UML Activity Diagram (OMG, 2004)</td>
<td>ConDec (Pesic and van der Aalst, 2006), DeCeSerFlow (van der Aalst and Pesic, 2006) and BPCN (Lu et al., 2009)</td>
<td>Placeholder activities (Sadiq et al., 2005; van der Aalst et al., 2009a; Schoenenberg et al., 2008), rule-based adaption (Kumar and Yao, 2009; Hallerbach et al., 2010)</td>
</tr>
</tbody>
</table>

Table 2: Run-Time Positions Analysis.

<table>
<thead>
<tr>
<th>Description</th>
<th>Procedural process enactment</th>
<th>Declarative process enactment</th>
<th>Hybrid process enactment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Impact on process characteristics</strong></td>
<td><strong>Positive</strong>: high efficiency</td>
<td><strong>Positive</strong>: high flexibility, assured compliance</td>
<td>The base process is executed according to the principles of the paradigm, whenever a placeholder activity is executed a paradigm switch takes place</td>
</tr>
<tr>
<td></td>
<td><strong>Negative</strong>: limited flexibility</td>
<td><strong>Negative</strong>: little support, limited efficiency</td>
<td>Characteristics of process parts are determined by their paradigm. Flexibility as a service improves flexibility</td>
</tr>
<tr>
<td><strong>Language examples &amp; execution environments</strong></td>
<td>BPEL (OASIS, 2007) and YAWL (van der Aalst and Ter Hofstede, 2005)</td>
<td>LTL (Pesic et al., 2008), CTL (Yu et al., 2006), PLMflow (Zeng et al., 2002), ECA rules (Kappel et al., 1998) and event rules (Paschke and Boley, 2009)</td>
<td>Chameleon (Sadiq et al., 2005) and subprocesses encapsulated in a service (van der Aalst et al., 2009a)</td>
</tr>
</tbody>
</table>

Modeling a model into enactable rules is rather straightforward (e.g. translation of a ConDec process model into LTL expressions (Pesic et al., 2008)).

**Impact on Process Characteristics.** During the declarative-declarative transition no (additional) factors that affect the business process’s flexibility, compliance, efficiency or effectiveness will/can be introduced.

**Example Process Case.** These transitions are suitable for business processes in a highly evolving environment and/or business processes with non-predictable execution paths. As declarative process management systems might provide limited support at run-time (Weber et al., 2009), this transition type will be most suited for experts dealing with unique cases (Schmidt, 2006) (e.g. non-standardized health care processes).

**Hybrid - Hybrid Transition.** The hybrid-hybrid transition can only be applied on hybrid process models that contain placeholder activities. Within the context of this type of hybrid paradigm, the business process paradigms of each process parts determine which same paradigm transition will be used for that
process part.

Impact on Process Characteristics. As these transitions are intrinsically similar to the previously described same strategy transitions, we argue that the impact on the process characteristics of this transition is determined by those same strategy transitions. It should be noted that process variant management approaches in this context will be easier to maintain than those presented in the context of procedural-procedural transitions, since the base structure is not duplicated.

Example Process Case. Hybrid-hybrid transitions will be used for business processes that contain both process parts with stable and highly evolving environments and/or that consist of both process parts with predictable and non-predictable execution paths (e.g. an advisory project with structured administrative activities and unstructured problem solving parts).

4.2 Cross-Paradigm Transitions

While the impact of the same paradigm transition on the desirable characteristics of a business process is rather limited, the impact of the cross-paradigm transitions can be rather extensive.

Procedural - Declarative Transition. The procedural process model is translated into a set of event-based business rules (e.g. preconditions), which can be used for a declarative process enactment (Casati et al., 1998).

Impact on Process Characteristics. Since the implicit constraints governing the procedural process model are exactly mapped on event-based business rules, the issue of overspecification is not dealt with. However, process flexibility slightly increases compared to the procedural-procedural transition due to the possibility of a run-time replacement or addition of a service task and the ability to define extra event-based business rules to deal with temporary circumstances (Dumas et al., 2005). The process effectiveness and efficiency are determined by the quality of the process model and the declarative execution environments.

Example Process Case. The procedural-declarative transition is suitable for use within the context of distributed processes, for which the process environment remains relatively stable and the ability to dynamically deal with temporary circumstances is valued (e.g. order-to-cash process in virtual organizations).

Declarative - Procedural Transition. Before runtime a systematic procedure is used for the construction of an optimal control-flow with reference to a particular characteristic. This procedure is closely related to artificial intelligence planning techniques (Hendler et al., 1990; Ferreira and Ferreira, 2005).

Impact on Process Characteristics. While the declarative process specifications provide extensive design-time flexibility, run-time flexibility remains limited to the flexibility offered by procedural enactment. However, the declarative process model in combination with a time-efficient planning algorithm, allows for a rapid adoption of new compliance requirements. Moreover, when the procedural workflow engine does not support any of the run-time flexibility enhancing techniques, compliance can be easily checked against the declarative process model.

The use of an artificial planning algorithm might positively affect both the process efficiency and effectiveness, since an optimization criterion needs to be specified. In addition, compared to declarative process enactment the end-user will be sufficiently guided and supported.

Example Process Case. The declarative-procedural transition is useful for processes that require far-reaching redesigns at regular intervals and/or for processes that benefit from an optimization with reference to a certain criterion. These processes, however, are at the same time relatively stable in the periods between those redesign phases.

Hybrid - Procedural Transition. Within the context of this transition the focus is primarily placed on hybrid models of the second type, the process models that combine a full procedural specification with a set of business rules. Before run-time the procedural reference model is customized to the specific needs of a particular case by applying the set of customizing business rules (Kumar and Yao, 2009; Hallerbach et al., 2010).

Impact on Process Characteristics. Due to the hybrid process model as a starting position, a neat approach to process variant management is provided. Compared to the process variant management approach introduced in the procedural-procedural transition, maintenance of requirements is not needlessly complicated since there is no duplication of the base process. However, the customization must be performed correctly and completely in order not to affect the process effectiveness and compliance.

Example Process Case. This type of hybrid-procedural transition will most likely be used for supporting a set of business processes that all only slightly differ from a specific reference process. We also expect the business processes to have predictable execution paths (e.g. claim handling for different insurance
products).

**Translating Declarative Placeholders.** Naturally, hybrid process models with placeholder activities can be transformed into procedural execution models as described in the declarative-procedural transition, which also results in a hybrid-procedural transition. Characteristics of the transition in this context are comparable with those of the declarative-procedural transition.

5 **CONCLUSIONS**

Designing information systems that provide support for operational business processes with the right level of process flexibility, compliance, efficiency and effectiveness can be a challenging task. This position paper promotes a clear distinction between the different points in the process life cycle. Furthermore, the paper elaborated on the transitions between design-time and run-time; in addition to the same paradigm transitions three interesting cross-paradigm transitions were presented.

**REFERENCES**


Dumas, M., Fjellheim, T., Milliner, S., and Pernici, B. (1998). Paradigm transitions three interesting cross-paradigm design-time and run-time; in addition to the same paradigm transitions three interesting cross-paradigm transitions were presented.


Swenson, K. (2010). Mastering the unpredictable: How adaptive case management will revolutionize the way that knowledge workers get things do.


