Business Process Management Journal

Dynamic product line for Business Process Management

Roberto dos Santos Rocha Marcelo Fantinato Lucinéia Heloisa Thom Marcelo Medeiros Eler

Received 27 September 2014
Revised 30 November 2014
14 January 2015
Accepted 26 February 2015

Dynamic product line for Business Process Management

Roberto dos Santos Rocha and Marcelo Fantinato
School of Arts, Sciences and Humanities, University of São Paulo, São Paulo, Brazil
Lucinéia Heloisa Thom
Institute of Informatics, Federal University of Rio Grande do Sul, Porto Alegre, Brazil, and
Marcelo Medeiros Eler
School of Arts, Sciences and Humanities, University of São Paulo, São Paulo, Brazil

Abstract
Purpose – The purpose of this paper is to present the proposal of a Product Line (PL)-based approach for Business Process Management (BPM) projects that cover the entire BPM lifecycle and proposes integrating it with dynamic techniques still not used together.
Design/methodology/approach – The authors carried out this work using the design science research methodology. The authors assessed the proposed approach using a classification procedure created through a series of specific attributes, which enables a comparison of the proposed integrated approach with related works selected from a systematic literature review.
Findings – The comparative assessment has shown that the proposed approach presents the most comprehensive solution than any other similar one suggested for the same purpose, mainly in terms of the coverage of the entire BPM lifecycle and dynamic techniques.
Research limitations/implications – Due to the high-level conceptual nature of the proposed approach, the authors could not evaluate it also in terms of some controlled experiment or a case study.
Originality/value – The proposed approach aims at improving the management of business processes in organizations in a systematic way using concepts and techniques that exist in other areas, but not widely used together yet, such as BPM, service-oriented computing, and Software PL.

Keywords Software product line, Dynamic product line, Service-oriented computing, BPM lifecycle, Dynamic processes
Paper type Conceptual paper

1. Introduction
The current complexity of organizations as well as the competitive market have required dynamism from the Information Technology (IT) in order to provide technical solutions to conduct business (Overby *et al.*, 2006). In this context, Business Process Management (BPM) (van der Aalst *et al.*, 2003; Weske *et al.*, 2004) and Software Product Line (PL for short) (Clements and Northrop, 2001; Pohl *et al.*, 2005) may provide technical and systematic support to improve the competitive edge of organizations (Adam *et al.*, 2012).

BPM can be integrated with Service-oriented Architecture (SOA) aiming at achieving strategic alignment among business-related areas and IT (Kearns and Sabherwal, 2006; Jensen *et al.*, 2008). It is particularly important to synchronize IT resources and efforts with the key strategic business objectives of an organization (Kearns and

This work was supported by Fapesp (São Paulo Research Foundation) and Capes (Coordination for the Improvement of Higher Education Personnel), Brazil. Roberto S. Rocha is grateful for the financial support from UFBA (Federal University of Bahia, Brazil).

Business Process Management Journal
Vol. 21 No. 6, 2015
pp. 1224-1256
© Emerald Group Publishing Limited
1463-7154
DOI 10.1108/BPMJ-09-2014-0091

Sabherwal, 2006). In BPM, the focus is on managing the business processes (BPs) to align the different organization activities in cross-flows (Weske, 2007). On the other hand, SOA provides a modern and flexible platform that supports those processes through a combination of structured IT resources based on the service-orientation paradigm (Papazoglou *et al.*, 2008).

PL explores software reuse for developing a family of products with market-reduced time and with improved quality (Clements and Northrop, 2001; Pohl *et al.*, 2005). PL has been successfully used in various application domains, including BPM (Fantinato *et al.*, 2006, 2010), thus it can provide a systematic approach to increase both the quality and the productivity of organizations that use BPM and SOA to support their BPs (Adam *et al.*, 2012; Jensen *et al.*, 2008). In fact, different approaches and techniques from Software Engineering, or Computer Science in general, have been used to improve the BPM field, in diverse aspects, including efficiency, effectiveness, quality and economy. Some examples of such initiatives involve applying aspect-oriented paradigm (Santos *et al.*, 2012), software metrics (González *et al.*, 2010), component-based development (Fettke and Loos, 2007) and data mining (Jareevongpiboon and Janecek, 2013). In this context, this work advances the BPM field with the same general purpose.

Specifically in terms of PL approaches in Software Engineering, Dynamic PLs are a specific type of PL that produces software to adapt to changes to meet user needs, considering resource constraints (Hallsteinsen *et al.*, 2008). Dynamic PLs are built around the central idea of a typical PL, but with some important differences in terms of dynamic properties (Hallsteinsen *et al.*, 2008), which are: first, dynamic variability – configuration and binding at runtime; second, changes binding several times during its lifetime; third, variation points change at runtime; fourth, deals with unexpected changes; fifth, deals with changes by users, such as functional or quality requirements; sixth, context awareness and situation awareness; seventh, autonomic or self-adaptive properties; eighth, automatic decision making; and finally, individual environment/context situation instead of a market.

Hence, Dynamic PLs are applicable to the BPM domain in order to meet dynamism requirements of IT infrastructure such as stated by Overby *et al.* (2006). However, few works aiming to treat BPM based on Dynamic PL have been proposed so far. According to a Systematic Literature Review (SLR) conducted by Rocha and Fantinato (2013), 15 papers were found which address some dynamic aspect in their proposed approaches, from 63 papers selected as primary studies. Nevertheless, all these 15 papers address only some specific dynamic feature, instead of being a comprehensive dynamic approach of PL for BPM. Moreover, none of them addresses all the phases of the BPM lifecycle.

This paper offers an integrated (symmetrical) conceptual approach essentially to integrate the central, but asymmetrical features of managing BPs. Fundamentally, the proposal is a simulation. Its value derives from mining unfulfilled territories. Yet, assessment and application levels for practice remain uncharted.

The proposed approach is called *Dynamic Product Line for Business Process Management* (DynPL4BPM), which merges concepts of BPM, SOA and Dynamic PLs, in order to improve the management of BPs in organizations in a systematic way. DynPL4BPM was devised defining PL processes for BPM that encompass the entire BPM lifecycle and systematically address dynamic properties at different stages of the BPM lifecycle. These dynamic properties rely primarily on the underlying technological foundation of SOA. We identified these dynamic properties by analyzing existing techniques, which address dynamic properties of BPM. We assessed the DynPL4BPM approach using a classification procedure created through a series of specific attributes, which enables a comparison of the proposed approach with related works selected from a SLR.

The remainder of this paper is structured as follows: Section 2 complements this introductory section with a motivational example which is further used in subsequent sections to illustrate the proposed approach; Section 3 presents basic concepts relating to BPM and PL, and related works; Section 4 outlines DynPL4BPM, proposed in this paper; Section 5 presents the DynPL4BPM stages in more details and includes a discussion regarding the dynamic properties handled in each stage; and, lastly, Section 6 sets out conclusions and suggestions for future works.

2. Motivational example

Commonly, the BPs addressed by organizations today are interorganizational processes, i.e., they extend beyond the boundaries of a single organization (Legner and Wende, 2007). While an organization focusses its activities on its key fields of knowledge, it usually outsources activities over which it does not have full command or knowledge, thereby engaging partner organizations in its BPs – and ultimately establishing interorganizational BPs. In this light, the BPs of a principal organization involve both activities performed within the organizations itself, probably through different departments, and interactions with external partners.

Here we examine an actual case of interorganizational BPs, described by Fuchsloch and Hoyer (2010). It consists of a strategy to internationalize the brand name of a large organization from Hamburg, Germany – called AllSport.

For the purpose of the strategy, the organization's marketing and sales departments have primary responsibility for advertising a new product at the Madrid Marathon in Spain. The Madrid Marathon is a specific instance, and, indeed, there may be other marathons or sporting events at which AllSport could be interested in advertising.

To advertise the product in this specific instance, AllSport should establish a partnership with a local marketing agency with extensive knowledge of local suppliers and service providers, as well as other potential partners. This type of partnership has been common and essential to the success of internationalization strategies, as local marketing agencies are better equipped to respond quickly to the needs of the strategy outlined by the contracting organization. The selected marketing agency was PromoBueno – a mid-sized company located in Madrid, Spain. PromoBueno offers its clients a variety of services, including the organization of promotional activities at fairs and events, brand advertising and marketing campaigns.

The example presented is a typical case in which the application of systematic reuse – through PL principles – is capable of contributing to AllSport's organizational strategy and in which the following ordinary characteristics of dynamic scenarios may occur. Since this case is only a motivation scenario, we do not present details about it and we assume that missing information is inferable or not essential to its understanding.

Below, we present some examples of cases in which reuse can help this organization:

- Service suppliers may not meet the established non-functional requirements such as the contracted Quality of Service (QoS) attributes, which could result in the selection of other service suppliers during runtime.
- The laws of some countries may impose several restrictions on the operations of external companies, requiring flexible organizational processes to change or adapt to different situations and to specific contexts.

- The organization realizes that the pre-determined promotional marketing strategy is not properly adapted and, therefore, requires intervention, such as the dynamic addition of a new activity to the process instance under execution.
- Some organization creates a law – domestic or international, which imposes a series of restrictions on organizations. Consequently, the BP model cannot predict all of the dynamics in the external environment. As such, the business model must comply with the law. In addition, the process instances currently under execution must also comply with these law changes.

This motivational example is important since it represents the type of target scenario for the approach that we present in this paper. This example, like many similar others, must be dynamic and requires flexible business organization processes, in this specific case, for purposes of contracting marketing agencies with primary responsibility for advertising the product in that specific market. In addition, this case offers potential variabilities that we will explore in the description of the approach provided in this paper, among them sporting events, which may vary concerning the specific athletic contest and venue, requiring more flexible BPs to meet the established organizational strategies. Further, this type of organization may use web services to identify sporting events held in a specific geographic region.

3. Theoretical background

This theoretical background section presents the basic concepts related to BPM (Section 2.1) and to PL (Section 2.2), which are used in the rest of the paper. Furthermore, it also presents the related works to the approach prosed in this paper (Section 2.3).

3.1 BPM

BPM has been presented as a key factor to the success of an IT infrastructure prepared for today's organizational demands (Jensen *et al.*, 2008). It oversees how work is performed in an organization to ensure consistent outcomes and to take advantage of improvement (e.g. reduce costs, execution times and error rates) opportunities (Dumas *et al.*, 2013). Moreover, BPM has been seen as a competitive edge as it can determine and exhibit the organizational maturity levels (Fantinato *et al.*, 2010).

According to van der Aalst *et al.* (2003), BPM includes methods, techniques and tools to support the design, enactment, management and analysis of operational BPs. Still according to van der Aalst *et al.* (2003), BPM is therefore an extension of classical workflow management approaches and systems.

A BP consists of a set of tasks performed in a specific sequence to achieve a common business goal (Weske, 2007; Lindsay *et al.*, 2003). BPs are what companies do whenever they deliver a service or a product to costumers (Dumas *et al.*, 2013). The BPM lifecycle includes several phases, such as: BP modeling; BP model instantiation; BP enactment and administration; BP monitoring and auditing; and BP assessment and optimization (van der Aalst *et al.*, 2003; Weske *et al.*, 2004). In the last phase, the execution history can be analyzed, looking for ways to improve the BP, which leads to BP remodeling, restarting the cycle all over again (Weske, 2007). Considering the current market dynamics, each sequence in such lifecycles is usually completed in a very short time, due to the constant need for new versions of the BPs running in organizations (Fantinato *et al.*, 2012).

Several specification and modeling languages and tools have been proposed to be used in BPM, from which the Business Process Model and Notation (BPMN) language

(OMG – Object Management Group, 2011) has become the "de facto" standard language to represent BPs. Nevertheless, other languages such as UML activity diagrams have also been used for modeling BPs (Russell *et al.*, 2006; Zhang and Duan, 2008).

In order to make the management and integration of BPs possible, from a technical point of view, different technologies have been proposed, including, not so recently, the middleware frameworks such as CORBA, DCOM and Java-RMI (Alonso *et al.*, 2004), which were properly used in the intra-organizational context. As the need for interoperability has evolved toward interorganizational cooperation, those existing solutions failed to meet their objectives (Alonso *et al.*, 2004). Such limitations were finally resolved when SOA and their implementations emerged (Leymann *et al.*, 2002), mainly through the web services technology, offering new perspectives to this need and providing, for example, the composition of e-services through Web Services Business Process Execution Language (WS-BPEL) (Jordan *et al.*, 2007) to enable the execution of BPs.

3.2 Software PL

Software PL is a set of software-intensive systems that share a common and managed set of properties that satisfy specific needs of a particular market segment or mission, which are developed from a common set of core assets in a prescribed way (Clements and Northrop, 2001). Different PL approaches share a set of key concepts (Pohl *et al.*, 2005), with slight differences among them. The key concepts usually presented in PLs are: Domain Engineering, Application Engineering, PL Architecture and Variability Management (Clements and Northrop, 2001; Pohl *et al.*, 2005).

The systematic PL Engineering has three processes – Domain Engineering, Application Engineering and PL Management – with all of them using the PL Architecture as a base artifact (Pohl *et al.*, 2005). The PL Architecture is one of the most important PL artifacts since it represents its core infrastructure (Clements and Northrop, 2001) as well as the configurations of modules and components that satisfy a given set of selected features. It also provides ideas for the reuse of opportunities (Kang *et al.*, 1998). Variabilities are tangible differences between products that can be revealed and distributed among PL artifacts (Bosch *et al.*, 2001), including the architecture, components and interfaces between them. Variations are revealable at any step during development, starting with the requirements analysis, and therefore need to be well managed (Pohl *et al.*, 2005). Feature Modeling is one of the most used techniques for Variability Management (Kang *et al.*, 1998).

Dynamic PLs are specific types of PL and produce software that can adapt to changes to meet user needs, considering resource constraints (Hallsteinsen *et al.*, 2008). Dynamic PLs have been identified as a promising strategy to address the design and implementation of changes that need to be carried out at runtime in new areas of application (Burégio *et al.*, 2010). Although Dynamic PLs are built based on the central idea of a typical PL, there are differences between them, since Dynamic PLs have many of the following properties: dynamic variability; configuration and binding at runtime; changes binding several times during its lifetime; variation points change at runtime; deals with unexpected changes; deals with changes by users, such as functional or quality requirements; context awareness and situation awareness; autonomic or self-adaptive properties; automatic decision making; and individual environment/context situation instead of a market (Hallsteinsen *et al.*, 2008).

3.3 Related work

Rocha and Fantinato (2013) conducted a SLR in which 63 works related to PL for BPM were selected as primary studies based on established criteria. Of these works, only 15 address at least one type of dynamic properties. We assessed these 15 approaches using a classification scheme that considers specific attributes regarding BPM, PL and Dynamic PL aspects.

The evaluation criteria established for the BPM aspect were the BPM lifecycle stages, which were: design and analysis; configuration; execution; and evaluation (Weske, 2007). In regard to the PL aspect, we considered the following concepts: Domain Engineering; Application Engineering; PL Architecture; and Variability Management; in addition to the specific PL variability management technique – Feature Modeling. We also adjusted and, subsequently, incorporated the nine properties proposed by Hallsteinsen *et al.* (2008) in the classification scheme.

Table I shows the classification scheme and the evaluation of each one of these 15 related works. Furthermore, Table I also includes, as a pre-reference, what is the purpose of DynPL4BPM using the same classification criteria. We present the assessment and comparison details in the following.

In regard to the BPM Lifecycle criteria, the results laid out in Table I and Figure 1 indicate that the related works consider only the first three phases of the BPM lifecycle – design and analysis (P1), configuration (P2) and execution (P3). Approaches that address the evaluation (P4) phase were not identified, which may have happen because of the inherent complexity of the PL evolution, since changes in a single PL asset could affect other assets and multiple products. Up to that point, the PL approaches for BPM carry out the BPM lifecycle only partially, reaching only the Execution phase. For DynPL4BPM, we propose also the evaluation phase, within the PL management process, enabling conducting of the evaluation tasks, in a continuous way, providing the possibility of completing the BPM lifecycle.

According to the data in Table I and Figure 2, extraction and synthesis of the data identified evidence of the application, in selected related works, of the five PL concepts. PL Architecture (L3), Variability Management (L4) and Feature Modeling (L5) were the most frequent items found, respectively, in 60, 93 and 67 percent of the related works. The second and third most frequent items were Domain Engineering (L1) and Application Engineering (L2), both found in 40 percent of the related works. All related works that presented the Application Engineering also presented the Domain Engineering (see Table I). The coverage, according to the data laid out in Table I, leads to the conclusion that the five PL Concepts were commonly applied, in combination, in the PL approaches for BPM. However, only 20 percent of these works covered all of the five PL concepts, which is the purpose of DynPL4BPM.

In terms of Dynamic Properties of PLs, according to the data in Table I and Figure 3, Dynamic Variability (D1) was the most frequent property, since it appeared in 80 percent of the related works. The second most frequent item was Variation points change at runtime (D3), found in 67 percent of the total. Only one related work, corresponding to 7 percent of the total of 15 assessed works in the comparative analysis covered various dynamic properties, including Deals with unexpected changes (D4), Deals with changes by users (D5), Autonomic or self-adaptive properties (D7), Automatic decision making (D8) and Individual environment/context situation instead of a market (D9). The selected related works did not address the Context awareness and situation awareness (D6) property; and only one related work addressed the Changes binding several times during its lifetime (D2) property.

As affirmed by Hallsteinsen *et al.* (2008), namely, that the PL may only be considered dynamic if it possesses a majority, or all, of these properties, we propose DynPL4BPM as the approach that most approximates an Dynamic PL, in comparison to the other

Table I.
Systematic comparison between DynPL4BPM and similar approaches

Related works	BPM lifecycle				PL concepts					Dynamic PL properties								
	P1	P2	P3	P4	L1	L2	L3	L4	L5	D1	D2	D3	D4	D5	D6	D7	D8	D9
Alférez and Pelechano (2011)	X	X			X	X	X	X	X				X			X	X	
Altintas and Cetin (2008)	X						X						X	X		X	X	X
Barat and Kulkarni (2012a)	X	X					X	X	X	X								
Barat and Kulkarni (2012b)	X	X					X	X	X	X								
Bayer *et al.* (2006)	X	X	X		X	X		X	X	X								
Karam *et al.* (2008)	X	X			X	X	X	X					X				X	
Khoshnevis (2012)	X				X	X		X		X	X	X	X					
Montero *et al.* (2008a)	X	X	X				X	X	X	X		X						
Montero *et al.* (2008b)	X							X	X	X		X						
Nguyen and Colman (2010)	X	X			X	X	X	X	X	X		X						
Nguyen *et al.* (2011)	X	X	X					X	X	X		X						
Rolland and Nurcan (2010)	X							X		X		X						
Schulz-Hofen and Golega (2006)	X	X	X		X	X	X	X	X	X								
Sun *et al.* (2010)	X						X	X		X		X						
Zhang *et al.* (2012)	X	X						X	X	X								
DynPL4BPM	X	X	X	X	X	X	X	X	X	X	X	X	X	X		X	X	

Note: X means in which works each one of the 18 attributes evaluated were found approaches examined here. DyPL4BPM addresses seven dynamic properties, followed by the approaches presented by Altintas and Cetin (2008), addressing five dynamic properties, and Khoshnevis (2012), addressing four dynamic properties. General analysis for all the related works can demonstrate that DynPL4BPM is the most complete one when these comparison criteria are used. For example, although the approach presented by Altintas and Cetin (2008) presents also a good number of dynamic properties, it does that upon a very weak PL for BPM approach, with only one phase of the BPM lifecycle and only one PL concept. Similar general analysis can be done in relation to all the other related works as presented in Table I.

4. Overview of the DynPL4BPM approach

By using concepts of the PL from the Software Engineering field, the DynPL4BPM is an approach proposed to systematize the operational processes for managing the BPs of an organization. Since the BPM context demands flexible and dynamic solutions, more specifically, concepts of Dynamic PLs are used. The DynPL4BPM approach proposed here aims to be comprehensive enough to cover the entire BPM lifecycle, including the treatment of dynamic aspects that the BPM domain requires.

This overview section demonstrates DynPL4BPM. First, it presents the general processes defined for the approach and then it describes an overview of the strategy adopted for the treatment of dynamic properties within such processes.

Figure 4 depicts the DynPL4BPM processes. There are three processes: Domain Engineering, Application Engineering and PL Management, which are equivalent to PL processes but centered on BPM. In such instances, the BP is the "product" or "application" developed by the PL and subsequently executed and evolved.

Three works drove the approach design:

(1) the PL4BPM processes proposed by Fantinato *et al.* (2006) and Fantinato *et al.* (2010), which were used as the original reference;

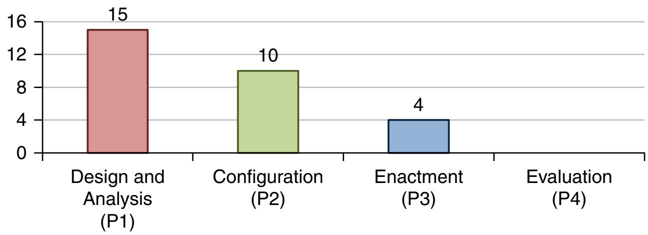

Figure 1. Phases of the BPM lifecycle addressed when applying PL concepts to such a domain

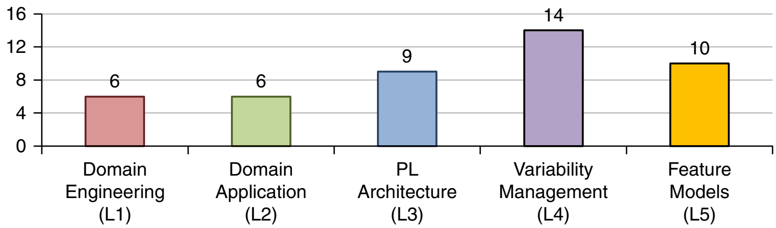

Figure 2. PL concepts applied to the BPM domain

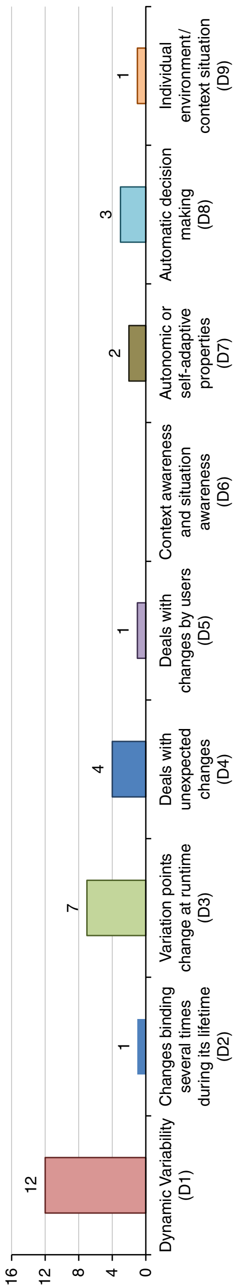

Figure 3.
Properties of Dynamic PLs applied to the BPM domain

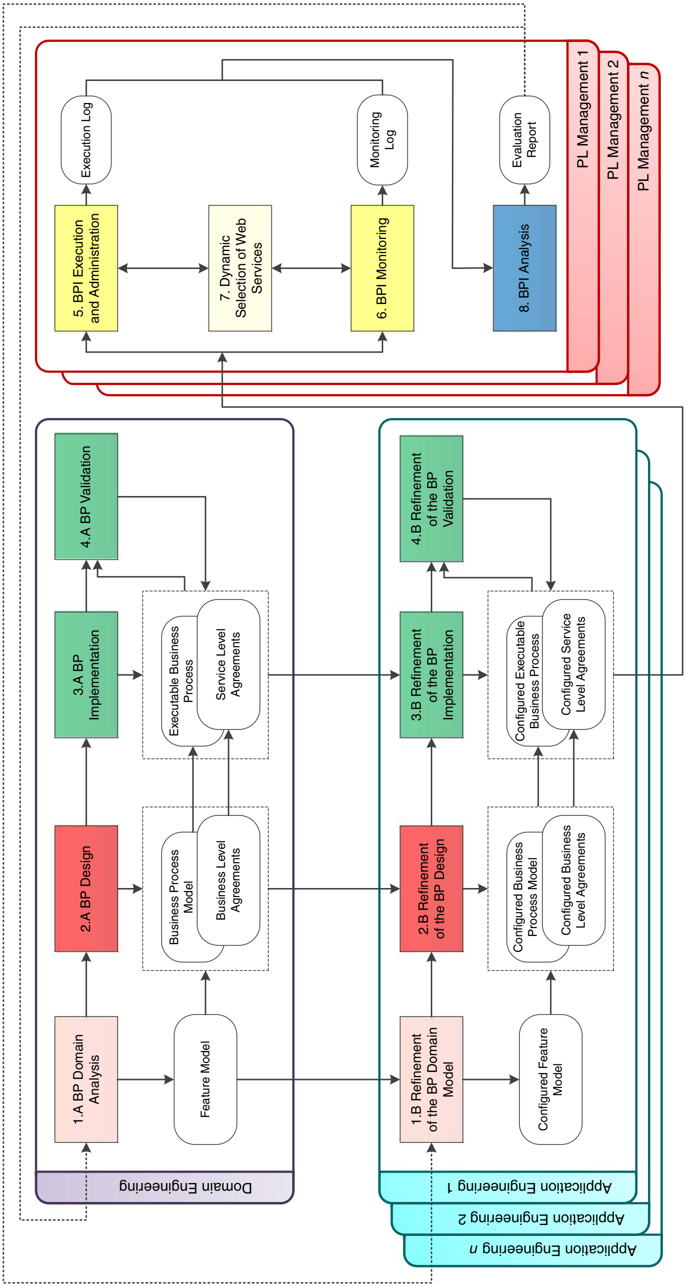

Figure 4.
DynPL4BPM
processes

(2) the three areas of the Framework Software Product Line Practice (PLP) proposed by Clements and Northrop (2001): core asset development, product development and management, which were base for the three processes in Figure 4; and

(3) the four BPM lifecycle activities proposed by Weske (2007): design and analysis, configuration, enactment and evaluation, which were base for the 12 stages in Figure 4.

The Domain Engineering and Application Engineering processes of DynPL4BPM was reused and adapted from the two PL processes proposed for the PL4BPM approach (Fantinato *et al.*, 2006, 2010). The PL4BPM processes comprise five stages, which are focussed only on the first three phases of BPM lifecycle. As a result, DynPL4BPM introduced three new stages in order to encompass the entire BMP lifecycle. To manage and evolve BPs, the PL Management process of DynPL4BPM adapted the concepts of management and evolution of PLs proposed by the PLP approach (Clements and Northrop, 2001).

One of the key objectives of DynPL4BPM is to handle dynamic properties. For example, dynamic selection of web services is already an explicitly property provided to compose the BP under execution at runtime. However, there are also other dynamic properties based on the demands of these approaches and technologies that are addressable within the PL approach for BPM supported by SOA.

Therefore, we investigated which dynamic properties could be incorporated to the DynPL4BPM processes, based on potential benefits and limitations. Approaches of Dynamic PLs are important sources of investigation since they provide strategies for addressing product adaptations at runtime. Then, we identified new types of properties associated with Dynamic PLs that are applicable in the BPM context, having the SOA paradigm as the underlying technological basis.

SOA includes a series of dynamic properties, when evaluated independently, which are extensible to the BPM context using SOA in the BPs enactment. The challenge is to ensure that these dynamic properties are addressed within the PL approach in conjunction with BPM, which had previously displayed benefits even prior to taking into account relevant dynamic properties. Thus, we identified the nine properties of Dynamic PLs cited by Hallsteinsen *et al.* (2008) as a guideline to the proposition of DynPL4BPM because they provide the theoretical bases suited to BPM contexts. From these nine dynamic properties, we selected seven of them as more suitable for DynPL4BPM, which are presented below:

(1) dynamic variability: configuration and binding at runtime;

(2) changes binding several times during its lifetime;

(3) variation points change at runtime: variation point addition;

(4) deals with unexpected changes (in some limited way);

(5) deals with changes by users, such as functional or quality requirements;

(6) autonomic or self-adaptive properties; and

(7) automatic decision making.

As they have not been widely explored yet, even in the general context of PL, the other two dynamic properties, as cited by Hallsteinsen *et al.* (2008), were not considered. These other two dynamic properties are: "Context awareness and situation awareness" and "Individual environment/context situation instead of a market."

In the next section, we present each DynPL4BPM process and detail the stages, including a discussion regarding the dynamic properties handled by each stage.

5. Description of the DynPL4BPM processes and stages

This detailed section sets out the stages for the three processes of DynPL4BPM – Domain Engineering (Section 5.1), Application Engineering (Section 5.2) and PL Management (Section 5.3), including treatment of dynamic properties for BPM through the application of identified techniques. Aiming at a better understanding of the approach being proposed, the motivation scenario presented in Section 2 is used here as an illustrative example. For this example, we present some excerpts of the main artifacts generated by the application of the DynPL4BPM.

5.1 Domain engineering

The Domain Engineering process establishes and maintains the reusable PL platform for BPM and sets out the shared points and variables of the BP. The resulting artifacts should be BP templates for the Application Engineering stages. Figure 5 shows an overview of the Domain Engineering process and the following subsections detail each stage of this process.

1.A BP domain analysis. This BP Domain Analysis stage produces the BP Domain Model, an artifact that defines, to a high degree, the main properties of the BP, including the organizations or departments involved in the BP, the service demands to be offered and used by each party involved, and possible non-functional requirements associated to those demands. The main artifact to be produced is a Feature Model (Kang *et al.*, 1998) that defines the mandatory, optional and alternative properties, to which all of the other artifacts produced in the remaining stages must be linked. The Feature Model should clearly delineate which properties represent services that implements the BPs and which services share the same binding time. Considering the motivation scenario presented in Section 2, the Feature Model presented in Figure 6 was elaborated as part of the illustrative example of the artifacts to be produced using DynPL4BPM.

According to this Feature Model, "Business Opportunity" and "Promotional Marketing" are examples of mandatory features, i.e. features that must be included in the configured feature model, during the process Application Engineering. Moreover, "Media" is an example of optional feature so that it can or cannot be selected to be part of the configure feature model. Moreover, "Promotional Events" and "Offers" are examples of alternative features – XOR and OR – respectively.

We recommend the use of the method proposed by Lee *et al.* (2008) to address these concerns and the dynamic properties of the PL in this BP Domain Analysis stage. This technique restructures the Feature Model by dividing it into two categories: first, orchestrating service layer – representing behavioral features, i.e., those that only define the sequence of the BP and which are not necessarily carried out by web services and second, molecular service layer – representing computer services features, i.e., a predefined task to be conducted by an IT system or a person. Figure 7 depicts the result of the application of this method to the Feature Model shown in Figure 6.

By using the strategies presented above, this BP Domain Analysis handles the dynamic properties "Dynamic variability" and "Variation points change at runtime" that are related to PL variability management.

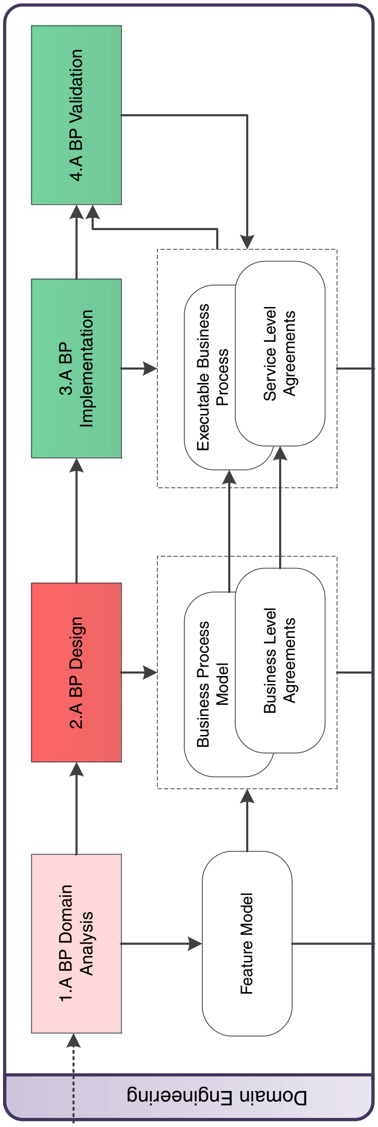

Figure 5.
Detailed view on the domain engineering process of DynPL4BPM

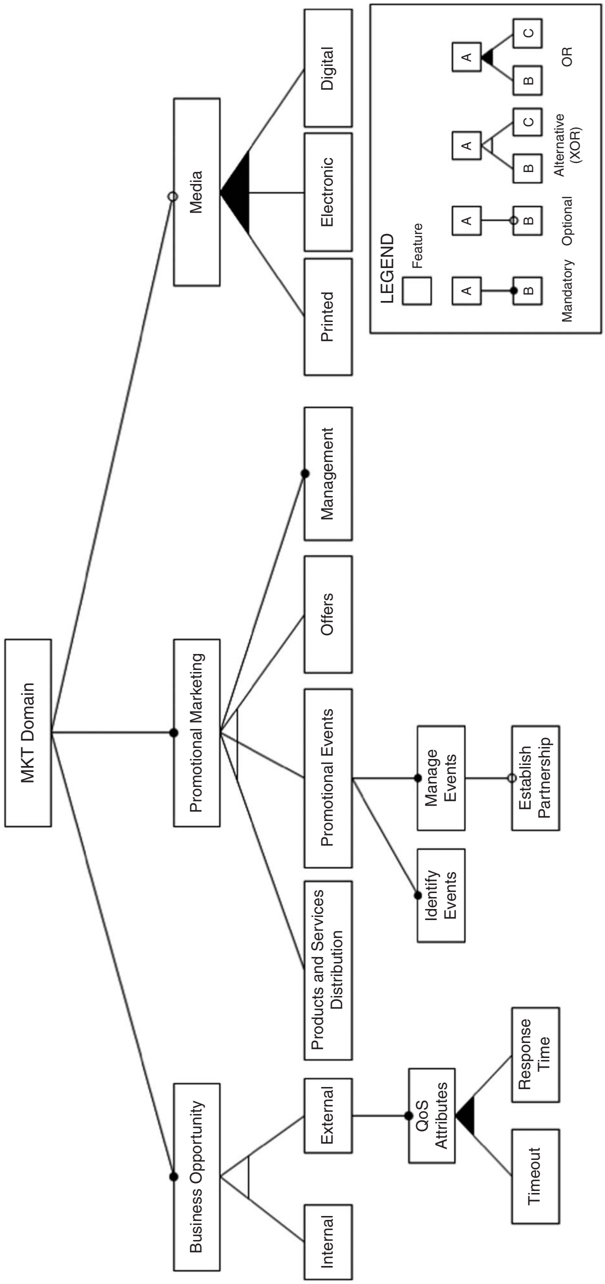

Figure 6.
A feature model representing the marketing domain

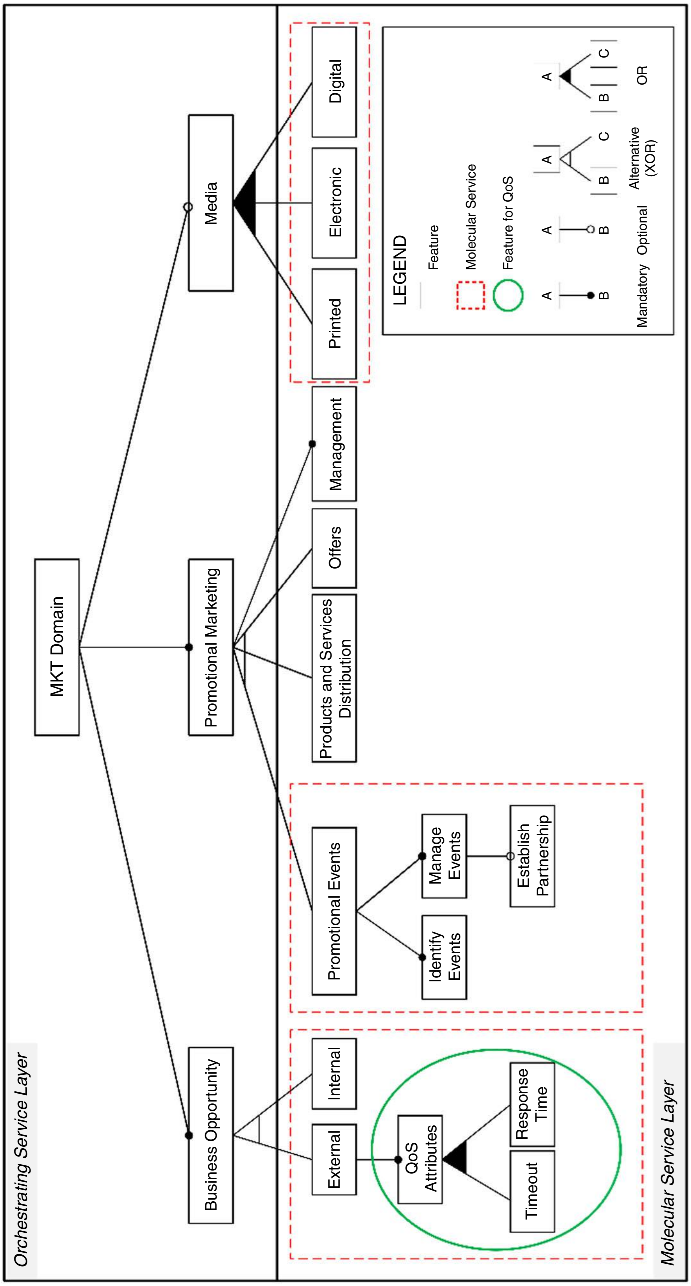

Figure 7.
A refined Feature model based on two service categories after application of the method proposed by Lee *et al.* (2008)

2.A BP design. This BP Design stage produces artifacts related to the BP Design of the PL for BPM, i.e., artifacts that connect the BP Domain with the BP Implementation. DynPL4BPM provides two inter-related artifacts for this BP Design stage: BP Model and Business Level Agreements (BLA). The first, developed in the BPMN, defines the flow of modeled activities to meet the functional properties required in the BP Domain Model and, the second, for which a standard has not yet been established, defines the quality rules associated to the modeled BP for the purpose of meeting the non-functional properties required in the BP Domain Model. Both artifacts should be systematically linked to the Feature Model to enable identification of which elements are, for their part, mandatory, optional and alternatives.

In addition to the basic process modeling notation – in this case, BPMN – supporting the development of the process model in the PL context also requires a technique that captures how to use the information on the Feature Model to obtain the basic structure of the BP Model. For DynPL4BPM, we selected the Business Family Engineering (BFE) (Montero *et al.*, 2008a, b) approach since: first, it provides a rigorous description of the Feature Model and second, it establishes a mapping that defines how to use the information drawn from the Feature Model to obtain the basic structure of the BP. We applied BFE and obtained a BPMN model for the Feature Models presented in Figure 6. Figure 8 shows the high-level view of the resulting BPMN model.

BFE technique always generates a "Non-Overlap Composition," i.e., the activities of the BP model do not consume or provide any artifact and no sub-process is generated in order for all activities to be carried out independently. Thus, as a following step, the original model generated by BPE, laid out in Figure 8, was adapted in order to take into account the logical sequencing of the activities, i.e., the activities that should be carried out concomitantly or other constraints in terms of the execution order of the activities based on the Domain Model elaborated in the previous stage. Figure 9 shows the adapted process. I this procedure, we grouped the features, of the Feature Model presented in Figure 7, in the BPMN model by using the "Group" element with the purpose of facilitating the identification of the mapping conducted between the Feature Model (Figure 7) and the BP Model (Figure 9).

This BP Design stage handles the dynamic property "Deals with changes by users, such as functional or quality requirements" by using late binding and late modeling techniques, as discussed by Weber *et al.* (2009). Through late binding, the activities are modeled simply as reserve space, namely as an abstract BP Model. At runtime, an adequate service is selected and connected to the process activity. DynPL4BPM incorporates aspects of late binding, which possesses an abstract model defined at design time. The services that execute those activities are selected at runtime. Late Modeling consists in defining parts of the process or the entire process at runtime (Weber *et al.*, 2009). The model defined at runtime may or may not be controlled by guidelines or restrictions, both defined at design time.

3.A BP implementation. This BP Implementation stage generates BP artifacts at a more technical level, i.e., at the BP execution level. To this end, DynPL4BPM provides two inter-related artifacts: Executable BP and the associated Service Level Agreements (SLA). The former, developed in WS-BPEL and WSDL – as shown in Figures 10 and 11, respectively – and derived from the BP Model, defines the executable version of activities in order to meet the functional properties required in the BP Domain Model. The latter, written in WS-Policy or WS-Agreement and derived from the BLAs, defines the QoS rules for the associated Executable BP, with a view to meet the required non-functional properties. Figure 12 shows part of a SLA specification.

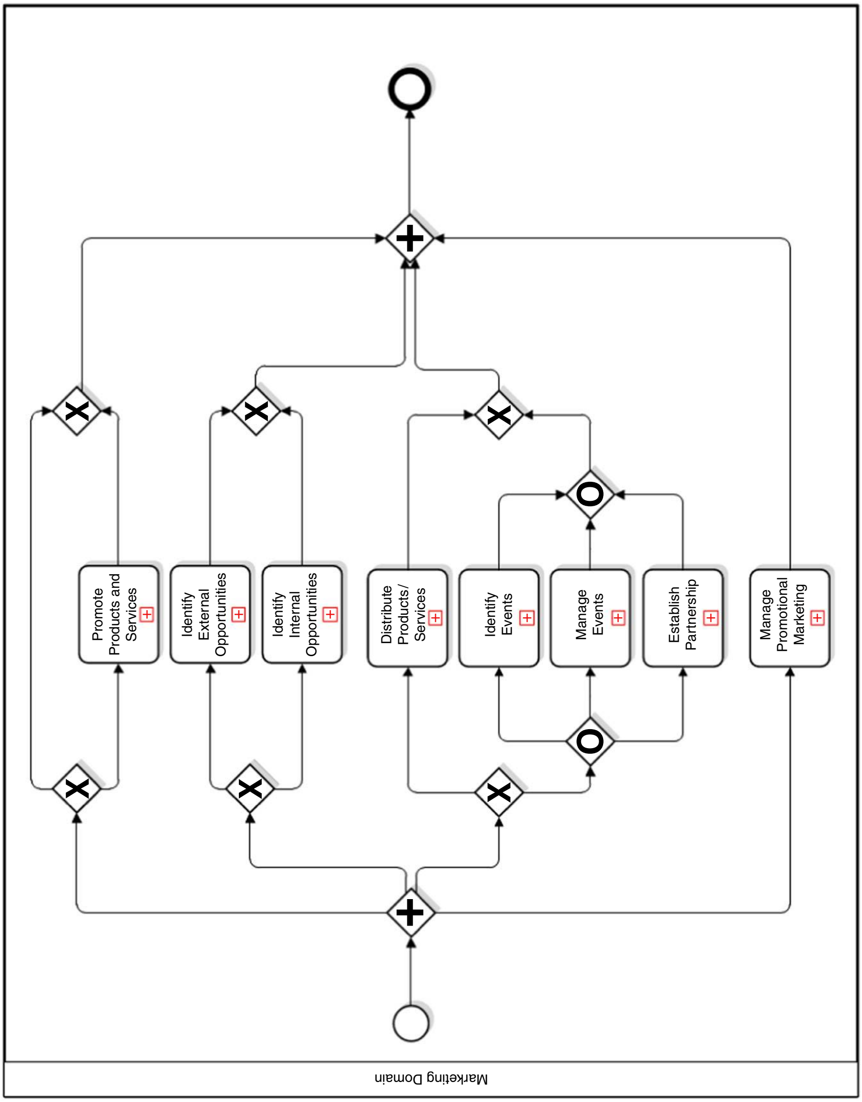

Figure 8.
A BP Model obtained through application of the BFE technique proposed by Montero *et al.* (2008)

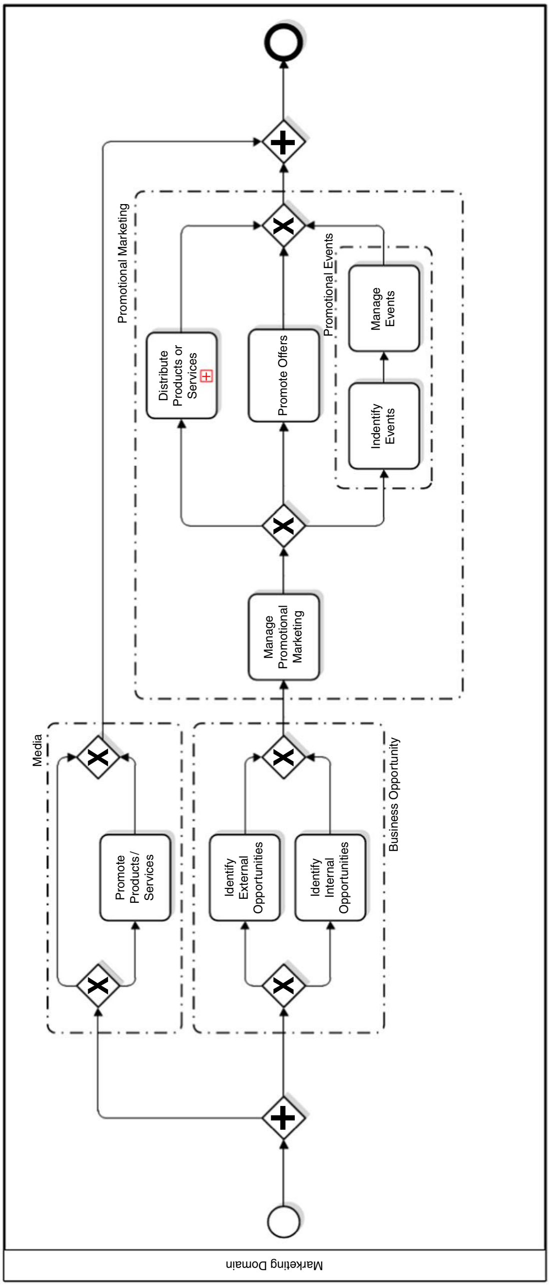

Figure 9.
A refined BP Model

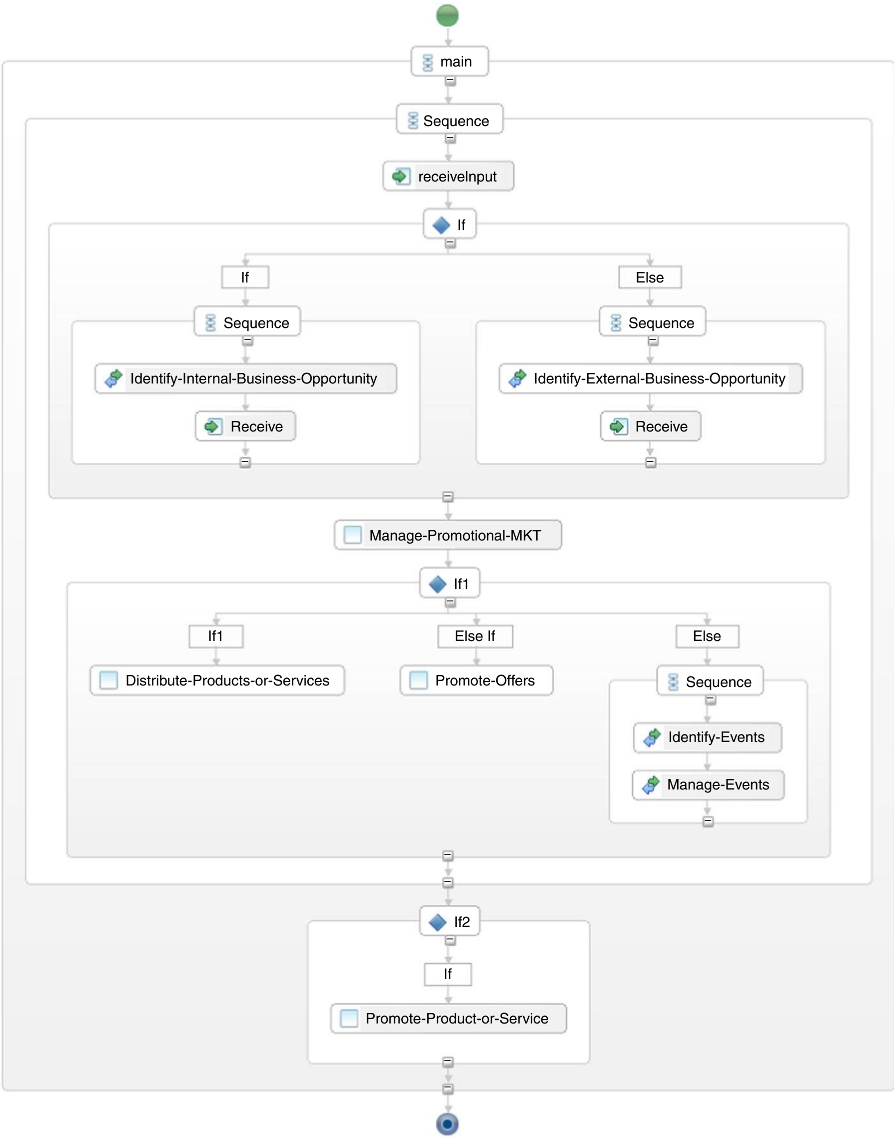

Figure 10. Executable BP developed in WS-BPEL

Nevertheless, both artifacts must be systematically linked to the respective parts of the artifacts used for their derivations, which, for their part, are connected to the features of the Feature Model, in order to enable identification of which elements are either mandatory, optional and alternative. In this way, both artifacts are usable as templates during the Application Engineering process. To ensure traceability between artifacts, the two templates generated in this BP Implementation stage – Executable BP and SLAs – should be annotated. The annotations correspond to comments in XML, connecting the elements generated through an identifier.

```
(…)
<bpel:sequence name="main">
<bpel:flow name="Flow"><bpel:sequence name="Sequence">
        <bpel:receive name="Receive"></bpel:receive>
        (…)
        <!-- Feature_ID = Business Opportunity <<core>> -->
        <bpel:if name="If">
        <bpel:sequence>
        <bpel:invoke name="Identify-Internal-Business-Opportunity"></bpel:invoke>
        <bpel:receive name="Acknowledge"></bpel:receive>
        </bpel:sequence>
        <bpel:else>
        <bpel:sequence>
        <bpel:invoke name="Identify-External-Business-Opportunity"></bpel:invoke>
        <bpel:receive name=" Acknowledge"></bpel:receive>
        </bpel:sequence>
        </bpel:else>
        </bpel:if>

        <!-- Feature_ID = Promotional Marketing <<core>> -->
        <bpel:empty name="Manage-Promotional-MKT"/></bpel:empty>
        <bpel:if name="If">
        <bpel:empty name="Distribute-Products-or-Services"></bpel:empty>
        <bpel:elseif>
        <bpel:empty name="Promote-Offers"></bpel:empty>
        </bpel:elseif>
        <bpel:else>
        <bpel:sequence name="Sequence">
        <bpel:invoke name="Identify-Events"></bpel:invoke>
        <bpel:invoke name="Manage-Events"></bpel:invoke>
        </bpel:sequence>
        </bpel:else>
        </bpel:if>
        <bpel:sources>
               <bpel:source linkName="link1"></bpel:source>
        </bpel:sources>
</bpel:sequence>
        <!-- Feature_ID = Promote Product or Service -->
        <bpel:if name="If1">
        <bpel:invoke name="Promote-Product-or-Service"></bpel:invoke>
        <bpel:targets>
        <bpel:target linkName="link1"></bpel:target>
        </bpel:targets>
        </bpel:if>
        <bpel:links>
        <bpel:link name="link1"></bpel:link>
        </bpel:links>
</bpel:flow></bpel:sequence>
```

Figure 11. Executable BP developed in WSDL

```
<wsdl:definitions targetNamespace="http://tempuri.org/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    …
    xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy" ... >
(…)

<!-- Feature_ID = Business Opportunity <<core>> -->
<wsdl:service name="BusinessOpportunity">
<wsdl:port name=" " binding=" ">
<wsp:Policy>
<wsp:ExactlyOne>
        <wsp:All>
                       <wsp:Assertion …></Assertion>
        </wsp:All>
        </wsp:ExactlyOne>
</wsp:Policy>
<soap:address location="SOAPAddress"/>
</wsdl:port>
</wsdl:service></wsdl:definitions>
```

Figure 12. A SLA specification in WS-Policy

Non-functional properties may be related to BLA and SLA alike, where BLA is aimed at guaranteeing fulfillment of these requirements from a BP Model perspective, while SLA is designed to guarantee quality at the service level. The ideal outcome is that the business objectives, specifically in terms of quality, be first represented through BLAs during modeling of BPs in BPMN, so that these BLAs could serve as income for developing SLAs during the establishment of Executable BP in WS-BPEL and WSDL, in a top-down fashion. However, there is still no adequate solution for the representation of BLAs and the subsequent derivation to SLAs. Although this translation mechanism does not exist yet, the non-functional requirements identified in the "2.A BP Design" stage needs to be transformed into an SLA. To this end, W3C recommends the usage of WS-Agreement and WS-Policy to formalize the non-functional needs of web services within an Executable BP.

Lastly, this BP Implementation stage handles the dynamic property "Changes binding several times during its lifetime," which is related to the dynamic composition of BPs. In sum, two classes classify the composition strategies: creating of the abstract service and binding moment (Fluegge *et al.*, 2006). Creating of the abstract service refers to the introduction of an abstract activities workflow, i.e., activities that do not own implementation mechanisms and that, as such, must be connected to concrete services for execution purposes.

4.A BP validation. This BP Validation stage consists in validating the artifacts generated during the "3.A BP Implementation" stage, namely, Executable BP and SLA artifacts, with a view to guarantee that these artifacts are in conformity with the defined requirements of the PL for BPM, as modeled in the "1.A BP Domain Analysis" stage and designed in the "2.A BP Design" stage. The validations carried out in this BP Validation stage should consider the generic context of the BP developed in the PL. Specifically, the various possibilities of application of such BP, which may occur in different contexts during the Application Engineering process, should be taken into account for the BP Validation.

In traditional Software Engineering, Software Testing activities are responsible for the validation of the software implementation. At this point, there are various test levels that address different default types, specifically: unit test, integration test and system test (Lübke, 2007). A correlation between traditional software tests and service composition tests is also possible. In this light, these test levels are applicable to web service compositions.

In the specific case of DynPL4BPM, testing may consider both the BP functional properties, represented by the WS-BPEL specification, and the non-functional properties, represented by the WS-Policy or WS-Agreement specification. One test challenge in the PL context, when applied to the Domain Engineering, involves testing a generic artifact, which is useful in different configurations during the Application Engineering and, therefore, should be tested through a variety of combinations. However, integration and system tests are not feasible in this context, to the extent that testing different combinations of the domain components leads to an exponential increase in the test configurations (Tevanlinna *et al.*, 2004).

5.2 Application engineering

This second process uses the reusable PL platform for BPM in a specific organization or inter-organization context. The Application Engineering stages accomplish this action on the basis of artifacts defined in the Domain Engineering stages, through keeping the common points and resolving the variabilities of the BP. For each specific desired application context sought in the same domain, this process should be executed once. Figure 13 shows an overview of the process and the following subsections detail each stage of this process.

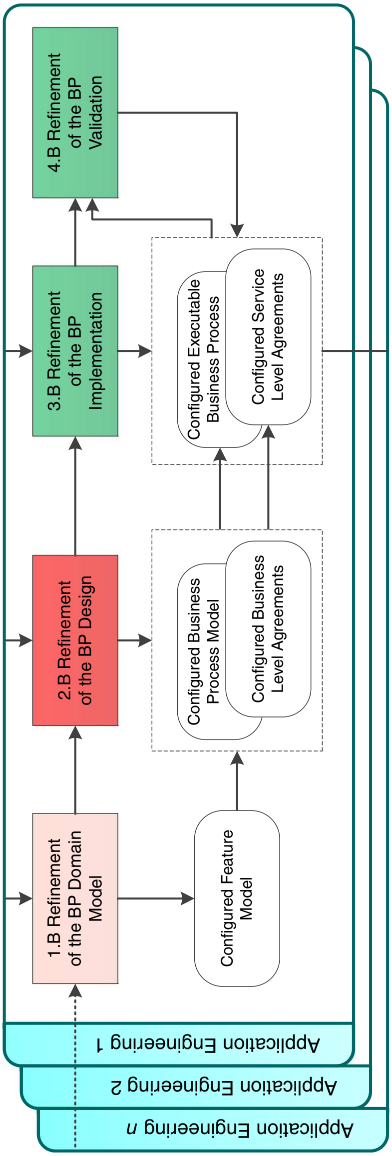

Figure 13.
Detailed view on the application engineering process of DynPL4BPM

1.B Refinement of the BP domain model. This refinement stage refines the BP Domain Model, which was created in the Domain Engineering process, to ensure that it represents, in a high level of abstraction, the specific properties needed so that a BP can be created for a specific need. These properties include, for example: service subsets offered to and needed by each party involved, and specific non-functional requirements related to these demands. The Feature Model created in the Domain Engineering process should be configured for purposes of selecting optional and alternative properties, which will define precisely how the remaining artifacts of the PL for BPM should be derived for creation of this specific BP, as all of them are systematically linked to the features of the Configured Feature Model. Figure 14 shows an example of a Configured Feature Model.

This refinement stage handles the dynamic properties "Dynamic variability" and "Variation points change at runtime" that are related to PL variability management.

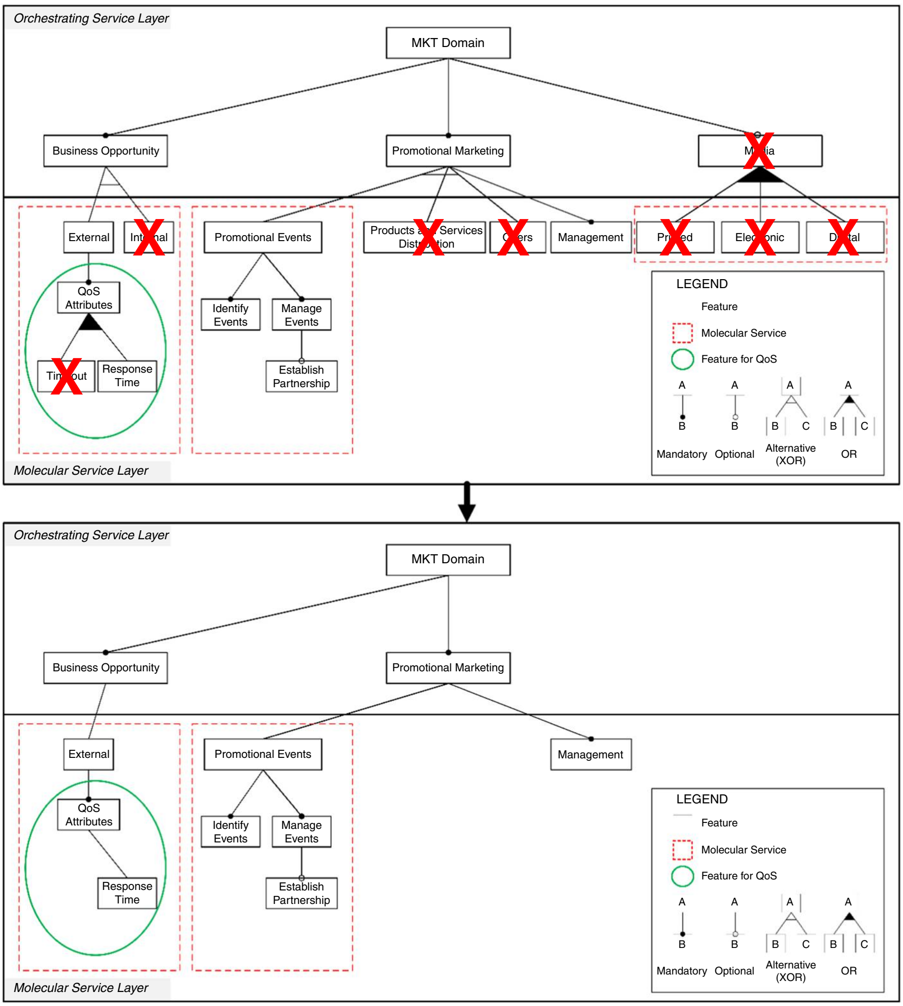

Figure 14.
A configured feature model

2.B Refinement of the BP design. This refinement stage refines the BP Design artifacts, which were created in the Domain Engineering process intended to be used here as templates. To this purpose, DynPL4BPM derives the two inter-related artifacts – BP Model and BLAs – based on the specific context desired for the new BP as represented in the Configured Feature Model. For each specific BP derived from a single domain, different refinements of the BP Design must be done, depending on which optional and alternative features were selected during the "1.B Refinement of the BP Domain Model" stage. During the refinement of the BP Design, a Configured BP Model and Configured BLAs are generated. According to the illustrative example presented in Figure 15, all of the elements associated to mandatory features or to optional and/or alternative features selected during configuration of the Feature Model are maintained. On the other hand, the optional and/or alternative features not selected shall cause the removal of the respective parts of the BP Model and BLAs templates.

This refinement stage handles the dynamic property "Deals with changes by users" that is related to Late Binding and Late Modeling techniques as a complement to the respective stage in Domain Engineering (Weber *et al.*, 2009). DynPL4BPM incorporates Late Binding aspects by including an abstract model defined at design time, while the services to carry out the process' activities can be defined at runtime. Only adjustments

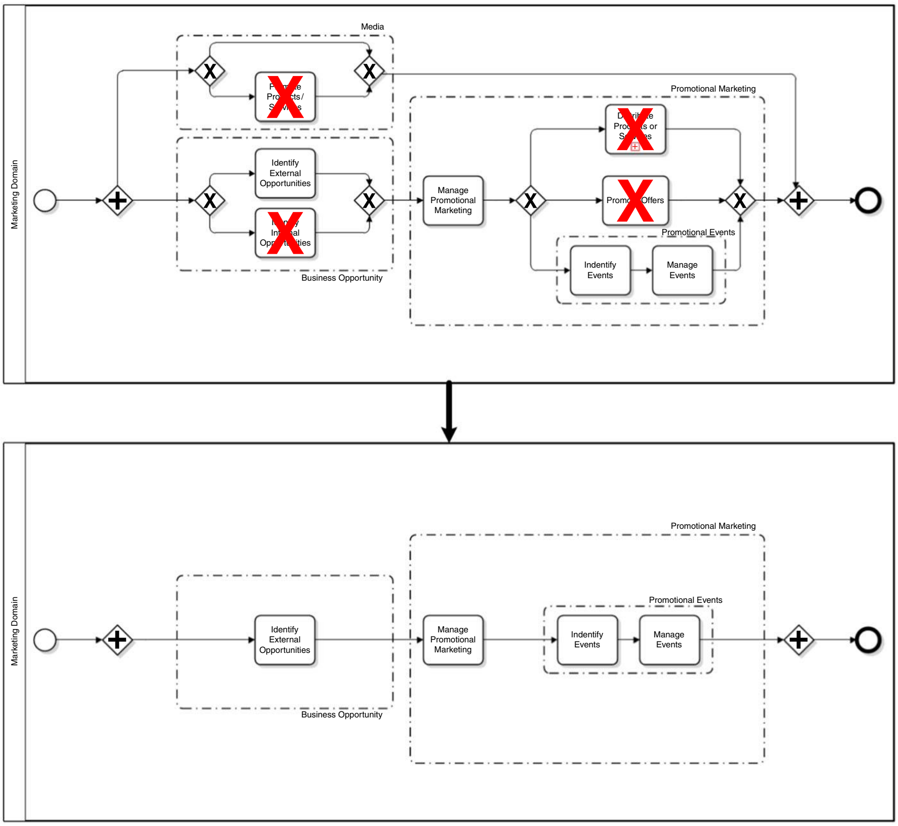

Figure 15.
A configured BP Model

necessary for each organization's context should be executed. For BP Design defined during Domain Engineering, general guidelines or constraints are defined. On the other hand, during this refinement stage, the specific needs of the involved organizations shall be considered. In regard to Late Modeling, it could also be a relevant contribution to the "Deals with changes by users" property related to Dynamic PLs.

3.B Refinement of the BP implementation. This refinement stage transforms the artifacts generated in the previous stage into artifacts to enable BP execution. It employs as templates the inter-related artifacts of Domain Engineering: Executable BP and SLAs. The artifacts generated in the "3.A BP Implementation" stage are transformed, respectively, into a Configured Executable BP and Configured SLAs. According to the illustrative example presented in Figure 16, similarly to the previous stage, all of the parts of the templates associated to mandatory features or to optional and/or alternative features selected during configuration of the Feature Modeling are maintained in the Configured Executable BP and Configured SLAs. On the other hand, the parts associated to optional and/or alternative features not selected in the Configured Feature Modeling are removed from the template.

This refinement stage handles the dynamic property "Changes binding several times during its life-time" that is related to the dynamic composition of the BPs. We based the composition strategy used in DynPL4BPM on the manual workflow building with biding at runtime (Fluegge *et al.*, 2006). On the basis on the BP Implementation artifact templates, which are ready to connect the services at runtime, only the necessary adjustments should be carried out here, by adapting the requirements – functional and non-functional – to the specific needs of each organization.

4.B Refinement of the BP validation. This refinement stage validates the artifacts generated from the previous stage, i.e. Configured Executable BP and Configured SLAs, to ensure they conform to the requirements of the PL for BPM. The specific desired application for this BP shall be taken into account. In contrast to what should occur during the "4.A BP Validation" stage, the validations carried out in this refinement stage must consider the specific context of the refined BP being developed, i.e., the specific application created during the Application Engineering process. Integration and system tests were not feasible in the Domain Engineering cycle, as the effort to test different combinations of the domain components result in an exponential increase in test configurations (Tevanlinna *et al.*, 2004). As such, an appropriate strategy for this refinement stage is the "Distribution of Responsibilities" (Neto *et al.*, 2011), which consists in specifying test levels for each BP lifecycle. In other words, the service compositions may be tested individually in the Domain Engineering; whereas the pertinent integration, system and acceptance tests may occur during the Application Engineering. This refinement stage handles the dynamic properties "Deals with unexpected changes," "Autonomic or self-adaptive properties" and "Automatic decision making" that are related to runtime monitoring.

5.3 PL management

This third process manages the set of instances of a same BP that is to be executed, monitored and, possibly, evaluated for purposes of optimization, based on the BP derived from the Application Engineering process. The results of the evaluation can lead to the evolution of the PL for BPM via two possible tracks: first, in the Domain Engineering process – adjustments to the reusable platform artifacts in general or second, in the Application Engineering process – a new application of the reusable PL

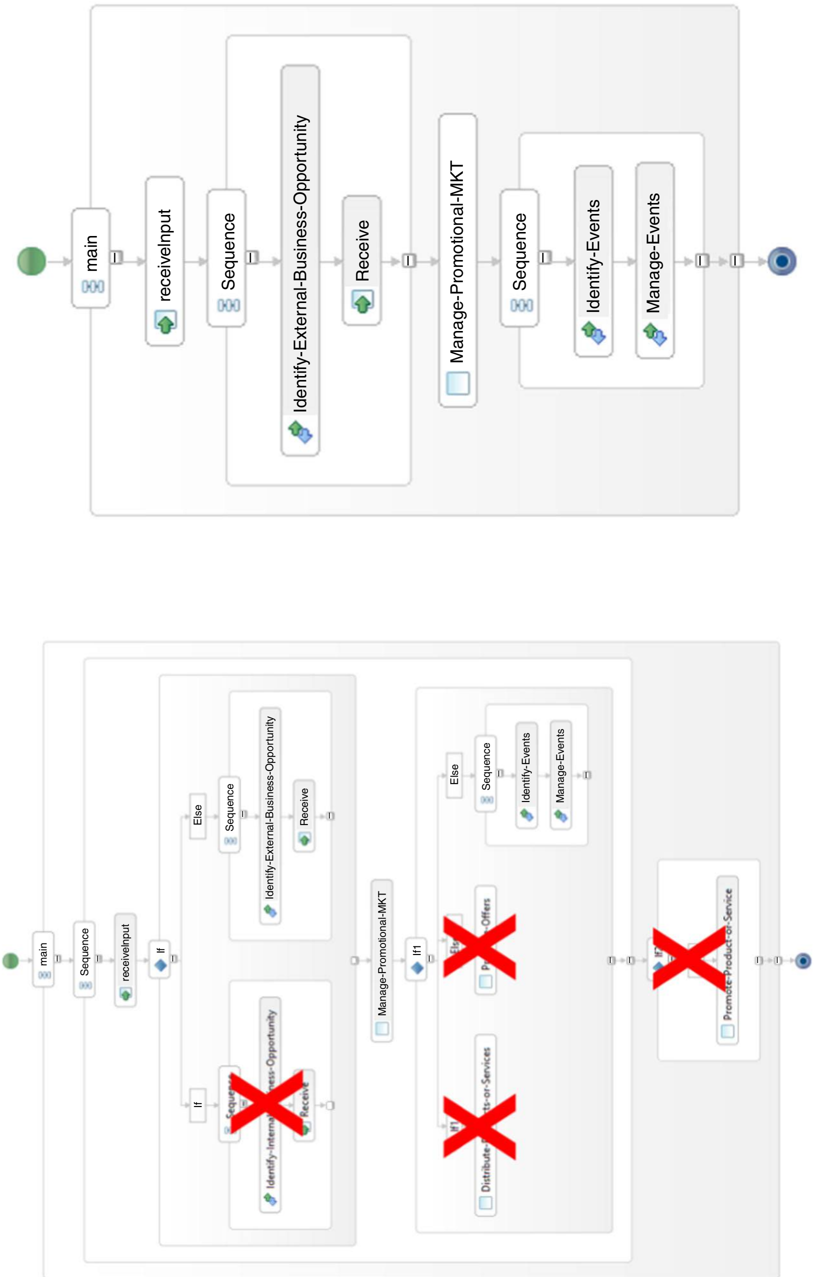

Figure 16.
Configured
executable BP

platform for BPM, which simply initializes as a new set of configurations. Figure 17 shows an overview of the process and the following subsections detail each stage of this process.

5 BPI execution and administration. This management stage provides administrative support execution of BP Instances (BPI). Each BP derived from the Application Engineering process is executed to meet a specific organizational demand. Given the conceptual nature of BPs, the same process may be executed several times for a set of different input data, referred here as BPI. BPI execution can be translated as invocations of the web service which compose the Executable BP in WS-BPEL, considering the QoS levels established in the SLAs. There are two possibilities for

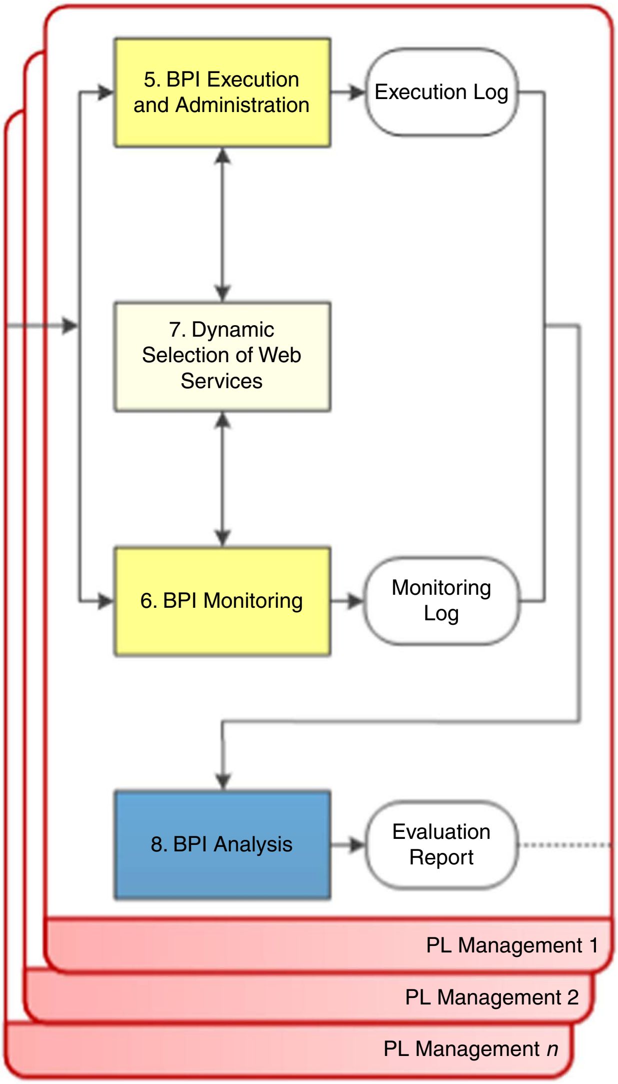

Figure 17.
Detailed view on the PL management process of DynPL4BPM

service invocation: static, when the web services are defined at design time, i.e., during the "3.A BP Implementation" or "3.B Refinement of the BP Implementation" stages; or dynamic, when the web services are defined at runtime – in this case, during the "7 Dynamic Selection of Web Services" stage. BPM Systems (BPMS) may be used to manage BPI executions and to ensure that they are in conformity with the execution restrictions specified in the Executable BP Model. BPMS also makes it possible to maintain a log of BPI executions, an essential task for controlling PL evolution from BP Evaluation.

6 BPI monitoring. This management stage verifies compliance with the SLAs associated to the respective BP. If QoS violations are detected, a recovery strategy should be applied. A possible reaction should be to carry out the "7 Dynamic Selection of Web Services" stage to choose a new web service more satisfactory. This BPI Monitoring stage also produces an important log – the Monitoring Log – for each BPI under execution. This artifact is also used as an entry point to the "8 BPI Analysis" stage. This management stage handles the dynamic properties "Deals with unexpected changes," "Autonomic or self-adaptive properties" and "Automatic decision making" that are related to runtime monitoring. Effective monitoring and recovery strategies for WS-BPEL service compositions should include a model of common data, monitoring management infrastructure and recovery framework. The Unified Framework proposed by Baresi *et al.* (2008) is appropriate for BPI Monitoring, as it enables the accommodations of various monitoring approaches which can be added as plug-ins.

7 Dynamic selection of web services. This management stage searches and selects web services at runtime, in order to: first, implement BP activities which were deliberately not linked to a web service at design time so that organizations could have flexibility to make this selection dynamically at runtime or second, substitute other web services which have failed to meet the SLAs contracted during the BPI execution. This management stage handles the dynamic property "Changes binding several times during its lifetime" that is directly related to the automatic composition of workflow-based services. Workflow-based techniques are used mainly when the consumer has a defined process model. Although Tizzo *et al.* (2011) proposed their technique based on workflows, the selection of services through this approach is not appropriate for the DynPL4BPM processes, as it only considers functional requirements. An adequate technique must take into account non-functional requirements as well.

8 BPI analysis. This management stage performs an analysis of a specific BPI or of a set of BPIs to potentially optimize the BP under execution. The goal is to provide an opportunity to optimize the reusable platform at the domain level or, at least, the BP used as the basis for generating instances at the application level. This activity results in an Evaluation Report useful to indicate new versions of the whole PL for BP or merely a specific BP in which the BPI is based. To evaluate the BPIs, process mining techniques are applicable in an effort to detect bottlenecks or other identifiable trends in the BPIs, based on the Execution Log and Monitoring Log. Process mining is a research discipline that encompasses aspects relating to the Artificial Intelligence field – such as machine learning and mining data. Process mining aims at discovering, monitoring and improving real knowledge extraction processes based on event logs generated (van der Aalst, 2011). Based on the items obtained from the application of process mining, the result of the BPI Analysis can lead to evolution of the PL for BPM in two ways: first, maintenance and consequent evolution of the reusable platform in general in the Domain Engineering process or second, a new application for a reusable PL platform for BPM in the Application Engineering process through a new set of configurations in the Refinement of the BP Domain Model process.

6. Conclusion and discussion

Flexibility in conducting organizational BPs is the main challenge the DynPL4BPM approach seeks to address. In this light, the stages of DynPL4BPM incorporated dynamic properties, at least partially, with a view to aligning the BPM and SOA paradigms.

The main contributions of this work are: first, definition of complete PL processes for BPM that covers all the BPM lifecycle, including the Evaluation phase, which is not covered by the similar approaches and second, treatment of a majority of the dynamic properties cited by Hallsteinsen *et al.* (2008), at different stages of the BPM lifecycle.

In regard to definition of complete PL processes for BPM, the similar approaches related to DynPL4BPM that were found can be considered incomplete, given that such approaches do not address the Evaluation phase. Considering that BPM extends to classic Workflow approaches through adding the Evaluation phase (van der Aalst *et al.*, 2003), consequently the approaches prior to DynPL4BPM can be considered only PLs for Workflow, but not PLs for BPM. As such, it is possible to conclude that DynPL4BPM is a step toward a comprehensive PL approach for BPM.

In terms of dynamic properties, DynPL4BPM proposes the integrated use of a series of existing techniques which together covers seven of nine of the properties considered (Hallsteinsen *et al.*, 2008). From 15 related works, containing PL approaches for BPM, which present at least one of these properties, in average they present only two of them. Two of them represent five and four properties of Dynamic PLs (Altintas and Cetin, 2008; Khoshnevis, 2012) but, on the other hand, they could not actually be considered complete PL approaches for BPM, since they encompass, for example, only the first phase of the BPM lifecycle.

So far, we based the assessment conducted on the proposed approach on comparing it to similar approaches considering a set of expected features. Since no case study or experiment has been conducted to assess the application of DynPL4BPM in solving a problem, as metrics were not collected in order to draw more robust conclusions, we intend as next steps to move forward in the assessment of the proposed approach. Moreover, we intend, together with new assessments, to conduct actions that enable organizations interested in the approach trying to put into practice the processes proposed so that they could provide us feedback and we could carry out improvements accordingly.

References

Adam, S., Riegel, N., Doerr, J., Uenalan, O. and Kerkow, D. (2012), "From business processes to software services and vice versa – an improved transition through service-oriented requirements engineering", *Journal of Software: Evolution and Process (JSEP)*, Vol. 24 No. 3, pp. 237-258.

Alférez, G. and Pelechano, V. (2011), "Context-aware autonomous web services in software product lines", *Proceedings of the 15th International Software Product Line Conference, SPLC 2011, IEEE Computer Society, Munich*, pp. 100-109.

Alonso, G., Casati, F., Kuno, H. and Machiraju, V. (2004), *Web Services: Concepts, Architectures and Applications*, 1st ed., Springer-Verlag, Berlin, Heidelberg.

Altintas, N.I. and Cetin, S. (2008), "Integrating a software product line with rule-based business process modeling", *Proceedings of the VLDB Workshop Trends in Enterprise Application Architecture, TEAA 2005, Springer-Verlag, Trondheim*, pp. 15-28.

Barat, S. and Kulkarni, V. (2012a), "A component abstraction for business processes", *Proceeding of the 2nd International Workshop on Reuse in Business Process Management, rBPM 2012, Springer-Verlag*, *Clermont-Ferrand*, pp. 301-313.

Barat, S. and Kulkarni, V. (2012b), "A component abstraction for business processes", *International Journal of Business Process Integration and Management (IJBPIM)*, Vol. 6 No. 1, pp. 29-40.

Baresi, L., Guinea, S. and Pasquale, L. (2008), "Towards a unified framework for the monitoring and recovery of BPEL processes", *Proceedings of the Workshop on Testing, Analysis, and Verification of Web Services and Applications, TAV-WEB 2008, ACM Press*, *Seattle, WA*, pp. 15-19.

Bayer, J., Kose, M. and Ocampo, A. (2006), "Improving the development of e-business systems by introducing process-based software product lines", *Proceedings of the 7th International Conference on Product-Focused Software Process Improvement, PROFES 2006, Springer-Verlag*, *Amsterdam*, pp. 348-361.

Bosch, J., Florijn, G., Greefhorst, D., Kuusela, J., Obbink, J.H. and Pohl, K. (2001), "Variability issues in software product lines", *Proceedings of the 4th International Workshop on Software Product-Family Engineering, PFE 2001, Springer-Verlag*, *London*, pp. 13-21.

Burégio, V.A., Almeida, E. and Meira, S.R.d. L. (2010), "Characterizing dynamic software product lines – a preliminary mapping study", *Proceedings of the 4th International Workshop on Dynamic Software Product Lines, DSPL 2010, Lancaster University*, *Jeju Island*, pp. 53-60.

Clements, P. and Northrop, L. (2001), *Software Product Lines: Practices and Patterns, SEI Series in Software Engineering*, 3rd ed., Addison-Wesley, Boston, MA.

Dumas, M., la Rosa, M., Mendling, J. and Reijers, H.A. (2013), *Fundamentals of Business Process Management*, 1st ed., Springer-Verlag, Berlin, Heidelberg.

Fantinato, M., Gimenes, I.M.S. and Toledo, M.B.F. (2006), "Web service e-contract establishment using features", *Proceedings of the 4th International Conference on Business Process Management, Lecture Notes in Computer Science*, *Vienna*, Vol. 4102, pp. 290-305.

Fantinato, M., Gimenes, I.M.S. and Toledo, M.B.F. (2010), "Product line in the business process management domain", in Kyo, C.K., Sugumaran, V. and Park, S. (Eds), (Org.), *Applied Software Product Line Engineering*, 1st ed., Auerbach Publications, Boca Raton, FL, pp. 497-530.

Fantinato, M., Toledo, M.B.F., Thom, L.H., Gimenes, I.M.S., Rocha, R.S. and Garcia, D.Z.G.A. (2012), "Survey on reuse in the business process management domain", *International Journal of Business Process Integration and Management*, Vol. 6 No. 1, pp. 52-76.

Fettke, P. and Loos, P. (2007), "Framework and meta-model for specifying business components", *Business Process Management Journal (BPMJ)*, Vol. 13 No. 5, pp. 628-643.

Fluegge, M., Santos, I., Tizzo, N. and Madeira, E. (2006), "Challenges and techniques on the road to dynamically compose web services", *Proceedings of the 6th International Conference on Web Engineering, ICWE 2006, Springer-Verlag*, *Palo Alto, CA*, pp. 40-47.

Fuchsloch, A. and Hoyer, V. (2010), "Implementing cross-organizational business processes with enterprise mashups: Hype or reality?", *Proceedings of the 16th Americas Conference on Information Systems', AMCIS 2010, AIS, Lima*, pp. 165.

González, L.S., Rubio, F.G., González, F.R. and Velthuis, M.P. (2010), "Measurement in business processes: a systematic review", *Business Process Management Journal (BPMJ)*, Vol. 16 No. 1, pp. 114-134.

Hallsteinsen, S., Hinchey, M., Park, S. and Schmid, K. (2008), "Dynamic software product lines", *Computer*, Vol. 41 No. 4, pp. 93-95.

Jareevongpiboon, W. and Janecek, P. (2013), "Ontological approach to enhance results of business process mining and analysis", *Business Process Management Journal (BPMJ)*, Vol. 19 No. 3, pp. 459-476.

Jensen, C.T., Charters, I., Amsden, J., Darlington, S., Owen, M., Herness, E. and Irassar, P. (2008), "Leveraging SOA, BPM and EA for strategic business and IT alignment", IBM White Paper, December, pp. 10.1-10.28.

Jordan, D., Evdemon, J., Alves, A., Arkin, A., Askary, S., Barreto, C., Bloch, B., Curbera, F., Ford, M., Goland, Y., Guizar, A., Kartha, N., Liu, C.K., Khalaf, R., Konig, D., Marin, M., Mehta, V., Thatte, S., van der Rijn, D., Yendluri, P. and Yiu, A. (2007), 'Web services business process execution language version 2.0'.

Kang, K.C., Kim, S., Lee, J., Kim, K., Shin, E. and Huh, M. (1998), "FORM: a feature-oriented reuse method with domain-specific reference architectures", *Annals of Software Engineering (ASE)*, Vol. 5 No. 1, pp. 143-168.

Karam, M., Dascalu, S., Safa, H., Santina, R. and Koteich, Z. (2008), "A product line architecture for web service-based visual composition of web applications", *Journal of Systems and Software (JSS)*, Vol. 81 No. 6, pp. 855-867.

Kearns, G.S. and Sabherwal, R. (2006), "Strategic alignment between business and information technology: a knowledge-based view of behaviors, outcome, and consequences", *Journal of Management Information Systems (JMIS)*, Vol. 23 No. 3, pp. 129-162.

Khoshnevis, S. (2012), "An approach to variability management in service-oriented product lines", *Proceedings of the 34th International Conference on Software Engineering, ICSE 2012, IEEE Computer Society*, *Zurich*, pp. 1483-1486.

Lee, J., Muthig, D. and Naab, M. (2008), "An approach for developing service oriented product lines", *Proceedings of the 12th International Software Product Line Conference, SPLC 2008, ACM Press*, *Limerick*, pp. 275-284.

Legner, C. and Wende, K. (2007), "The challenges of inter-organizational business process design – a research agenda", *Proceedings of the 15th European Conference on Information Systems, ECIS 2007, AIS*, *St. Gallen*, pp. 106-118.

Leymann, F., Roller, D. and Schmidt, M.-T. (2002), "Web services and business process management", *IBM Systems Journal (IBMSJ)*, Vol. 41 No. 2, pp. 198-211.

Lindsay, A., Downs, D. and Lunn, K. (2003), "Business processes-attempts to find a definition", *Information and Software Technology (INFSOF)*, Vol. 45 No. 15, pp. 1015-1019.

Lübke, D. (2007), "Unit testing BPEL compositions", in Baresi, L. and Nitto, E.D. (Eds), *Test and Analysis of Web Services*, Springer-Verlag, Berlin, pp. 149-171.

Montero, I., Peña, J. and Ruiz-Cortés, A. (2008a), "From feature models to business processes", *in Proceedings of the 2008 IEEE International Conference on Services Computing, Vol. 2 of SCC 2008, IEEE Computer Society*, *Honolulu, HI*, pp. 605-608.

Montero, I., Peña, J. and Ruiz-Cortés, A. (2008b), "Representing runtime variability in business-driven development systems", *Proceedings of the 7th International Conference on Composition-Based Software Systems, ICCBSS 2008, IEEE Computer Society*, *Madrid*, pp. 228-231.

Neto, P., Runeson, P., Machado, I., Almeida, E., Meira, S. and Engstrom, E. (2011), "Testing software product lines software", *IEEE Software*, Vol. 28 No. 5, pp. 16-20.

Nguyen, T. and Colman, A. (2010), "A feature-oriented approach for web service customization", *Proceedings of the 2010 IEEE International Conference on Web Services, ICWS 2010, IEEE Computer Society*, *Miami, FL*, pp. 393-400.

Nguyen, T., Colman, A. and Han, J. (2011), "Modeling and managing variability in process-based service compositions", *Proceeding of the 9th International Conference on Service-Oriented Computing, ICSOC, Springer-Verlag, Paphos*, pp. 404-420.

OMG – Object Management Group (2011), "Business Process Model and Notation (BPMN) Version 2.0", available at: www.omg.org/spec/BPMN/2.0/PDF/

Overby, E., Bharadwaj, A. and Sambamurthy, V. (2006), "Enterprise agility and the enabling role of information technology", *European Journal of Information Systems (EJIS)*, Vol. 15 No. 2, pp. 120-131.

Papazoglou, M.P., Traverso, P., Dustdar, S. and Leymann, F. (2008), "Service-oriented computing: a research roadmap", *International Journal of Cooperative Information Systems (IJCIS)*, Vol. 17 No. 2, pp. 223-255.

Pohl, K., Böckle, G. and van der Linden, F.J. (2005), *Software Product Line Engineering: Foundations, Principles and Techniques*, Springer-Verlag, Berlin, Heidelberg.

Rocha, R.S. and Fantinato, M. (2013), "The use of software product lines for business process management: a systematic literature review", *Information and Software Technology*, Vol. 55 No. 8, pp. 1355-1373.

Rolland, C. and Nurcan, S. (2010), "Business process lines to deal with the variability", *Proceedings of the 43rd Hawaii International Conference on System Sciences, HICSS 2010, IEEE Computer Society*, *Koloa, HI*, pp. 1-10.

Russell, N., van der Aalst, W.M.P., ter Hofstede, A.H.M. and Wohed, P. (2006), "On the suitability of UML 2.0 activity diagrams for business process modelling", *Proceedings of the 3rd Asia-Pacific Conference on Conceptual Modelling, APCCM 2005, Australian Computer Society*, *Hobart*, pp. 95-104.

Santos, F.J.N., Cappelli, C., Santoro, F.M., Leite, J.C.S.P. and Batista, T.V. (2012), "Aspect-oriented business process modeling: analyzing open issues", *Business Process Management Journal (BPMJ)*, Vol. 18 No. 6, pp. 964-991.

Schulz-Hofen, J. and Golega, S. (2006), "Generating web applications from process models", *Proceedings of the 2nd International Workshop on Model Driven Web Engineering, MDWE 2006, ACM Press*, *Palo Alto, CA*, pp. 6.

Sun, C.-a., Rossing, R., Sinnema, M., Bulanov, P. and Aiello, M. (2010), "Modeling and managing the variability of web service-based systems", *Journal of Systems and Software (JSS)*, Vol. 83 No. 3, pp. 502-516.

Tevanlinna, A., Taina, J. and Kauppinen, R. (2004), "Product family testing: a survey", *ACM SIGSOFT Software Engineering Notes (SEN)*, Vol. 29 No. 2, pp. 1-6.

Tizzo, N., Coello, J. and Cardozo, E. (2011), "Improving dynamic web service selection in WS-BPEL using SPARQL", *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, SMC 2011, IEEE Computer Society*, *Anchorage, AK*, pp. 864-871.

van der Aalst, W. (2011), *Process Mining: Discovery, Conformance, and Enhancement of Business Processes*, 1st ed., Springer-Verlag, Berlin, Heidelberg.

van der Aalst, W.M.P., Hofstede, A.H.M.t. and Weske, M. (2003), "Business process management: a survey", *Proceedings of the International Conference on Business Process Management, BPM 2003, Springer-Verlag*, *Eindhoven*, pp. 1-12.

Weber, B., Sadiq, S. and Reichert, M. (2009), "Beyond rigidity – dynamic process lifecycle support", *Computer Science – Research and Development (CSRD)*, Vol. 23 No. 2, pp. 47-65.

Weske, M. (2007), *Business Process Management: Concepts, Languages, Architectures*, 1st ed., Springer-Verlag, Berlin.

Weske, M., van der Aalst, W. and Verbeek, H. (2004), "Advances in business process management", *Data & Knowledge Engineering (DKE)*, Vol. 50 No. 1, pp. 1-8.

Zhang, H., Zou, H., Yang, F. and Lin, R. (2012), "From requirements to business processes development", *Proceedings of the IEEE 9th International Conference on Services Computing, SCC 2012, IEEE Computer Society*, *Honolulu, HI*, pp. 672-673.

Zhang, M. and Duan, Z. (2008), "From business process models to web services orchestration: the case of UML 2.0 activity diagram to BPEL", *Proceedings of the 6th International Conference on Service-Oriented Computing, ICSOC 2008, Springer-Verlag*, *Sydney*, pp. 505-510.

About the authors

Roberto dos Santos Rocha is a PhD Student at the University of São Paulo, Brazil. Master in Sciences (2012) at the University of São Paulo; Specialist in Management of Information Systems (2009) at the Federal University of Lavras; Bachelor in Business Administration (2006) at the Faculty of Technology and Sciences, Brazil. His main research interests are: business process management, service-oriented computing, software product line and geographic information systems for disaster management.

Dr Marcelo Fantinato is an Associate Professor at the University of São Paulo, Brazil. PhD in Computer Science (2007) and Master of Engineering (2002) at the University of Campinas; Bachelor in Computer Science (1999) at the State University of Maringa, Brazil. He worked as a software testing specialist at the CPqD Foundation (2001-2006) and as a Specialist in research and development at the Motorola (2006-2008), Brazil. He has a Green Belt Certification in the Motorola Six Sigma program for process improvement. His main research interests are business process management, service-oriented computing, software product line and software testing. Dr Marcelo Fantinato is the corresponding author and can be contacted at: m.fantinato@usp.br

Dr Lucinéia Heloisa Thom is an Assistant Professor at the Federal University of Rio Grande do Sul (UFRGS), Brazil. She was a Visiting Scientist at the University of Grenoble (2010-2011), France, and at the University of Ulm (2007-2009), Germany. Bachelor in Computer Science at the University of Santa Cruz do Sul (1999); Master (2002) and PhD (2006) in computer science at the UFRGS, Brazil. Part of her PhD was developed at the University of Stuttgart (2004-2005), Germany. Her research interests are in the area of workflow patterns, process design, IT support for healthcare processes and ontology.

Dr Marcelo Medeiros Eler is an Assistant Professor at the University of São Paulo, Brazil. PhD (2012) and Master (2006) in computer science and computational mathematics (2007) at the University of São Paulo, Brazil. Part of his PhD was developed at the National Research Council in Pisa (2010-2011), Italy. Bachelor in Computer Science (2003) at the University of Western São Paulo, Brazil. His main research area is software engineering and includes the following lines: aspect-oriented programming, component-based development, software testing and service-oriented computing.