Journal of Theoretical and Applied Information Technology
10th February 2015. Vol.72 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

GENERATING SERVICES SUPPORTING VARIABILITY FROM CONFIGURABLE PROCESS MODELS

Hanae Sbai, Mounia Fredj, Boutaina Chakir

AlQualsadi research team on Enterprise Architecture
High National School of Computer Science and Systems Analysis, ENSIAS
Mohammed V University of Rabat
Rabat (10000), Morocco
E-mail: Hanae.sbai@um5s.net.ma, Mounia.fredj@gmail.com, boutaina.chakir@um5s.net.ma

ABSTRACT

Today, a PAIS (Process-Aware Information System) is the broadly adopted information system by enterprises, which aims to define how to use services and business processes to achieve the business goals of enterprises. For this purpose, research work in PAIS was interested in aligning business processes and services in order to achieve a good connection between business process models as represented in the process layer, and service models as represented in the application layer In this context, to improve the reuse of business processes and services, the concepts of configurable process models and configurable services (which support variability) have emerged as new leading edge concepts for improving reuse. Indeed, variability refers to the characteristic of a system to adapt, specialize and configure itself with the context of use. More recently, configurable process models become the most popular solution to deal with adaptability in a PAIS. This has increased the research focus in how to extract services from configurable process models. Several approaches have developed solutions for proposing services generation from business process. However, generating configurable services from configurable process model has been neglected in the context of PAIS. In this paper, we propose an MDA approach for an automatic generation of configurable service models from configurable process models.
Keywords: *PAIS; configurable service; configurable process model; variability; MDA*

1. INTRODUCTION

Due to the lack of process control and automation in a data centered information systems, the business process orientation has emerged, by introducing workflow systems and the business process management (BPM) technologies [1]. In this context, the Process-Aware Information System (PAIS) has been introduced as a specific type of information systems which allows the separation of process logic and application code [2]. Moreover, at the design time, process logic is designed using Workflow Nets, UML, EPC, or BPMN; and at runtime, with the evolution of Service Oriented Architecture (SOA), process logic is implemented by developing services. In the literature, the PAIS is defined as a 4-tiers system [2] which is composed of the presentation layer (user interfaces), the process layer (the design of processes), the application layer (execution of the business process as a set of services), and the persistency layer (data storage). So, we can say that a PAIS provides a collaborative solution respecting BPM and SOA concepts [3], as services expose business functions of a given business process. This can help in reducing the gap between these two areas, easing the communication between business and Information Systems (IS) actors and the understanding of business needs [4].

In a competitive environment, the key challenge of enterprises is to reduce the cost and time of application development. **At the process layer**, the research community has been motivated to define common practices by developing the "Configurable Process Model (CPM)", and the "Configurable Service (CS)" solution **at the application layer**. These solutions have been proposed to enhance reuse in PAIS by supporting variability concepts. A CPM aims at describing common practices of enterprises, and at capturing variability of a business process, by defining a set of process variants in a single model [10] [11] [12] [13]. More recently, the emerging issue of the ***service variability***

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

management has been supported by several works [14], [15], [16], [17], [18], and [19].

Furthermore, many research works [4] [5] [6] have proved that the majority of organizations should align existing systems (which are represented by services), with business processes, as a prerequisite task to preserve the consistency of applications after a modification [7]. The most popular approaches [3] [4] interested in the mapping of the business processes to services, have focused on linking BPMN [8] models to SOAml models [9]. While these works are suited for creating service based business process models, there are not sufficient to cope with the creation of configurable services based on the configurable processes, because there do not support the variability concepts. So, it would be necessary to dispose of an alignment approach which ensures the connection between configurable process models and configurable services.

Today, many enterprises adopt the configurable process to deal with the issue of the variability in business processes and services. In this context, CPMs can support the execution of process variant through existing services, and this, by linking each variant to one or more services [11]. However, the number of relations between services and process variants is growing even faster. In this case, generating for each variant the corresponding services becomes a cost and time consuming task. For this purpose, and by analogy to the existing linking between business processes and services, we propose an approach for generating configurable services from configurable process models. We have examined the generation of VARSOAML [18] service models from the Variant Rich BPMN [21], which is the most used variability modeling approach for BPMN models.

Our approach is based on MDA (Model Driven Architecture) which allows developing application decoupled from technology in order to enhance business models reuse. It will enable the automatic generation of services in order to facilitate the development process. The generation is achieved with two transformations levels: CIM2PIM and PIM2PSM. In this paper, we focus on CIM2PIM, the first transformation level. The second one has been introduced in [22], and it represents a model to text transformation providing rules for mapping VarSOAML models to VarWEB services which is a meta-model representing variability in the Webservices plateforms (WSDL Web Service Description Language), BPEL - Business Process Execution Language).

The remainder of this paper is structured as follows: Section 2 introduces briefly variability (for business process and service) and the MDA approach, which are the prerequisite concepts related to our work. In Section 3, we provide an overview of the related work on service generation from business processes. Section 4 presents our proposal for generating configurable services from CPMs. Finally, Section 5 concludes the paper and provides some future directions.

2. BACKGROUND AND FOUNDATIONS

Variability concepts were first introduced in the field of software product line engineering [23]. Variability is generally defined as the ability to change or customize a software system [24]. It refers to the diversity of variations of the manufacturing processes for producing product variants in a product family [24]. The variability is concerned by defining *i)* which parts called «variation points» may vary over time or within a domain space, and *ii)* the different realizations of each variation point which are called the «variants».

A ***configurable process model*** deals with how to model business processes that are similar to one another in many ways, yet different in some other ways from one organization, project or industry [13]. A configurable process model is a combination of variability and business process concepts. This combination allows describing a set
of process variants by extending existing business process modeling languages to support variability.

A ***configurable service*** is a service including fixed and variable capabilities. Its modeling artifacts include variability representation. The variability for services can be modeled at multiple levels, namely architecture level, service interface, and implementation levels [14].

The Model Driven Architecture **MDA** [34] is a paradigm that allows separating concerns by splitting implementation choices from specifications of business needs. The MDA proposes three kinds of models:

- The Computation Independent Model (CIM) which describes the system's requirements.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

- The Platform Independent Model (PIM) which specifies the system independently of any platform
- The Platform Specific Model in which systems are implemented using a specific technology.

After introducing these concepts, we present in the next section a state-of-the art of the existing approaches on services generation from business processes.

3. RELATED WORK

In the context of PAIS, and in particular in the BPM and SOA interoperability domain, there is an industrial interest in ensuring connection and mapping between business process models and service models in order to improve the development of service oriented solutions from business process [4].

In this section, we study how the existing approaches in PAIS deal with linking process models to service models. The most relevant works in the service generation from business processes research area focus mainly on linking and generating services from business processes. To analyze dependencies between business processes and web services, the author of [25] suggest a methodology for evaluating the dependencies of a web service. In the SHAPE project (Semantically-enabled Heterogeneous Service Architecture and Platform Engineering), some rules were introduced for mapping BPMN process models to SOAml models [4]. Moreover, the PIMFORSOA project [26] proposed a model driven approach for integrating BPMN and SOAml models. This work developed rules for transforming BPMN models into SOAml models at the structural level. For the same purpose, the work of [3] puts forward the MINERVA's tool support for generating SOAml services from BPMN business process models. Recently, [5] conducts a model driven approach for mapping business processes to services, using BPEL and web service technologies. In [7], a transformational method for modeling BPMN business processes in the context of SOA has also been developed.

Concerning the work on configurable process models, we find that authors have been especially interested in three main issues:

- (i) Business process variability modeling: extensions of business process modeling languages have been proposed to capture the variability [10] [11] [12] [21][27]
- (ii) Configuring process variants: process configuration and mechanisms for individualizing configurable process models have been developed in [12] [13] [28].
- (iii) Managing evolution of variability: some mechanisms have been suggested to adapt CPM [29] [30] and to handle the evolution process for activities [31] and resources [32].

For generating services from configurable process models, few approaches propose technical steps to generate for each variant, one of more services [11] [20]. In [22], a method on generating configurable web services from configurable service models is discussed in order to link configurable service models to technical solutions in SOA.

From this review, we note that the majority of existing approaches in PAIS have taken up the issue of service generation from business process models at two levels:

- Linking business process models to service models
- Linking configurable /classical services models to technical solution in SOA (web services).

We can conclude that the existing approaches lack an effective method that takes into account the concept of variability, *i.e.* generating configurable services from CPM. In addition, the alignment between business and technical requirements for designing executable process is still one of the central problems, because the process specifications obtained from the pure business perspective cannot be executed "as-is", due to the constraints of information systems which are represented by services [33]. So, one can ask the following questions:

- How to develop services (which support the variability concept) that can realize and implement CPMs with enhancing reuse in PAIS?

For this purpose, we propose an MDA based approach for generating configurable services from configurable process models, by analogy with the mapping of business process to classical services approaches. Our approach is a complementary approach of a proposed approach for configurable web service generation from VarSOAML configurable service model [22].

Journal of Theoretical and Applied Information Technology
10th February 2015. Vol.72 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4. OUR PROPOSED APPROACH

A. Overview of the approach

In the context of a PAIS, our proposition is a collaborative solution because it ensures connection of the process and application layers, and aiming to support variability management and to reduce time and cost of the development of CPM based applications. Unlike other approaches that extract classical service models from CPMs, we propose to generate configurable service models from configurable process models. As we mentioned above, our proposition is supported by the MDA approach which is composed of three abstraction levels:

- CIM: it specifies the configurable process model using Variant Rich BPMN modeling language [21].
- PIM: it specifies configurable service models using VarSOAML modeling language [18], independently to platforms.
- PSM: it specifies the implementation of configurable service models using configurable web services [22].

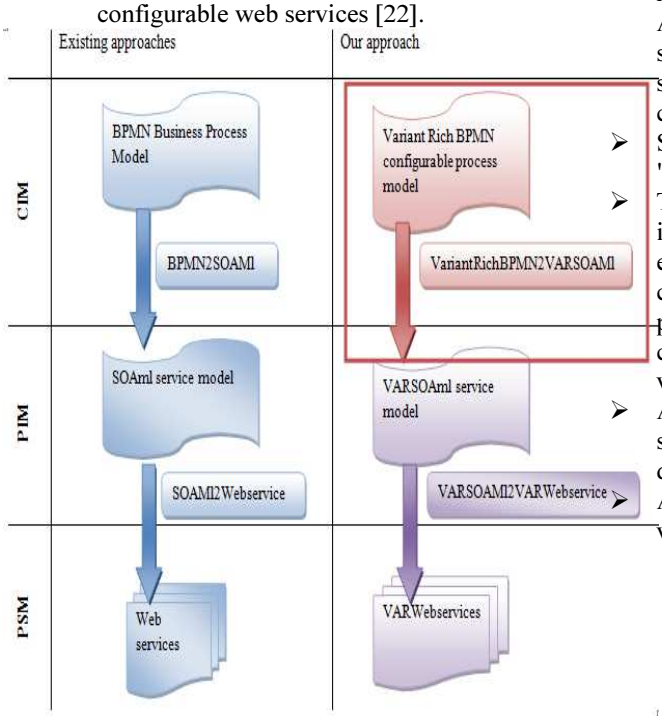

Figure 1: Our Approach For Generating Configurable Services From Configurable Process Models

We can see in Figure 1 the description of our approach compared to the existing approaches. In this paper, we only focus on the CIM to PIM transformation which is a model to model transformation, providing a set of transformation rules for mapping Variant Rich BPMN models to VarSOAML models.

In the next section, we present the different meta-models used by our proposal, namely Variant-Rich BPMN, VarSOAML and VARWebservices meta-models.

1) The Variant-Rich BPMN meta-model

We choose the Variant Rich BPMN process model because it is the most popular solution for capturing variability in BPM process models. It is an annotated based approach to represent variability of the BPMN process models using UML2 stereotypes. We define the following meta-model elements (cf. Figure 2):

- The Pool represents the concept of a participant which performs one or more activities. It can be variable, if at least one of the activities contained in the Pool is a variation point.
- A composite activity is a composition of one or more Activities (it is represented by the reflexive composition relation of the Activity element). A composite activity is a source or target of message flows that are sent and / or received by simple activities contained in this composite activity.
- Simple activities can be the "Source" or the "Target" of a given message flow.
- The collaboration can be defined as an interaction between two pools, which exchange messages flows, through two composite activities performed by these two pools. The collaboration can be a variable collaboration if at least one pool is a variation point.
- A message flow is associated with two simple activities that are contained in two different composite activities.
- Activity, resource or data objects can be a variation point or a variant.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

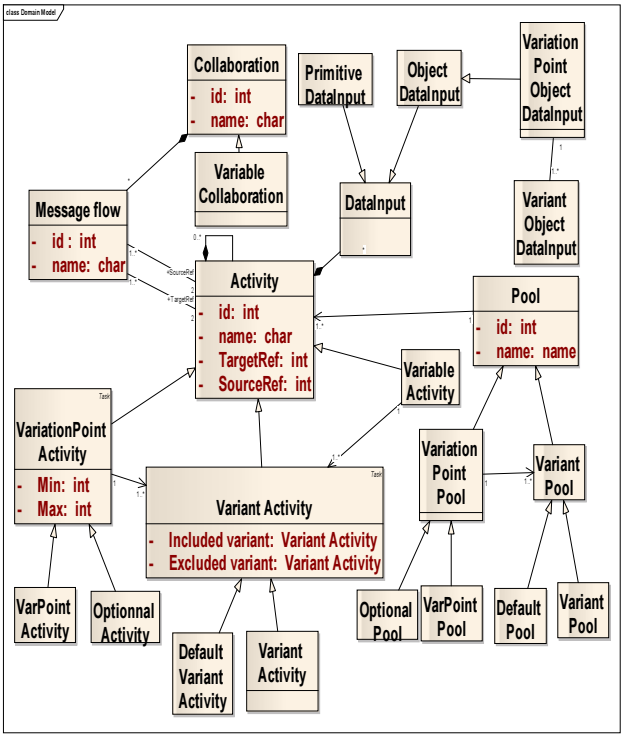

Figure 2: An Extract Of The Variant Rich BPMN Meta-Model

We can summarize the variability concepts and their representation in Table I. These concepts are applied to the process elements. Therefore, activities, resources and data can be variation points or variants.

Table I: Variant-Rich BPMN Concepts

Variability concept		Variability representation
Variation point	Alternative	-« VarPoint » defines an alternative variation point -« Abstract » defines an abstract alternative variation point which can have a set of implementations using dependent variants
	Optional	-« Null » defines an optional variation point which can be extended by a set of optional variants
Variant		-« Default » represents the realization by default of a variation point -« Variant » represents the realization of a variation point
Variable		-« Variable» defines a variable composite element which may contain a set of variation points.
Association {variation point and variants}		There are three kinds: -« implementation » is used with abstract variation point -« inheritance » is used with a simple alternative variation point -« extension » is used with an optional variation point

In the next section, we expose the PIM level meta-model: VarSOAML meta-model [18].

2) The VarSOAML meta-model

VarSOAML is an extension of SOAML supporting service variability modeling [18]. The Figure3 illustrates an overview of VarSOAML meta-model.

This meta-model allows the specification of variability in the following SOAML metaclasses [9]:

- ServiceArchitecture: defines how a set of participants works together.
- ServiceContract: specifies an agreement between providers and consumers of a service.
- ServiceInterface: defines the interface of a participant that provides or consumes a service.
- MessageType: specifies the information exchanged between service consumers and providers.
- Participant: is the type of a provider and/or consumer of services.

To express the variability, VarSOAML is based on three concepts:

- VariableElement: represents a variable area of an artifact that will be a subject of a variation at the resolution time.
- Variation or variation point: specifies a specific place in an artifact which is attached a decision taken at the resolution time.
- Variant: defines a realization of a variation point.

The concepts presented below are used in the elaboration of the variability metaclasses of VarSOAML. We give short descriptions of the main elements in the Table II.

In the next section, we give a short description of the VARWebService [18] used in the PSM layer of our approach, even if we do not expose in this paper the details of the PIM2PSM transformation.

3) The VARWeb Service meta-model

In order to automatically generate web services from VarSOAML models, we have proposed in [18] the VarWebservices meta-model that allows the specification of web services platform

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

supporting variability. It is the result of the merging of four meta-models:

- WSDL meta-model: inspired of the work of [18]. It contains elements that describe the WSDL specification.
- XSD meta-model: contains elements for the construction of the XML Schema document that describes the types used in the WSDL document.
- BPEL meta-model: allows the generation of BPEL specification.
- ServiceVariability meta-model: contains elements for the generation of the variability specification file that represents the realization of the services variability

In order to proceed to the MDA transformation between the CIM and the PIM layers, we need now to define the mapping between the elements of the source and the target meta-models, which is the subject of the next section.

B. *CIM to PIM transformation: VariantRichBPMN2VarSOAML mapping rules*

In this section, we present some of our mapping rules for the CIM to PIM transformation. The objective is to generate VarSOAML models from Variant Rich BPMN models, *i.e.* generating configurable services from configurable process models In this paper, we only focus on generating the following target configurable service models:

- The service contract model
- The service interface model.

To achieve this transformation, we need to perform two main steps.

- *The first step:* enables the determination of the equivalent elements of the source and target meta-models (respectively Variant Rich BPMN and VarSOAML meta-models).

In the following table (cf. Table III), we expose the mapping rules between Variant Rich BPMN and VarSOAML meta-model elements.

Table Iii: Identification Of Corresponding Elements Of Variant Rich Bpmn And Varsoaml Méta-Models [35]

Variant Rich BPMN meta-model elements	**VarSOAML meta-model elements**	**Transformation Rules name**
Collaboration	Service Contract, Service Interface	Collaboration2ServiceContract& ServiceInterface
Composite Activity	Interface	CompositeActivity2Interface
Simple Activity	Operation and Message Type	SimpleActivity2Operation and Simple Activity2MessageType
Data input	Parameter Operation	Data Input2parameter
Primitive Data Input	Simple Type	PrimitiveDataInput2SimpleType
Object Data Input	Complex Type	ObjectDataInput2ComplexType
Pool	Participant	Pool2Participant
Variable Collaboration	Variable Service Contract, Variable Service Interface	VariableCollaboration2VariableS erviceContract&VariableServiceI nterface
Variable Composite Activity	Interface	VariableCompositeActivity2Inter face
Alternative Variation Point simple Activity	Variation Operation with (Min(1),Max(1)	AlternativeVariationPointactivity 2VariationOperation
Optional Variation Point Activity	Variation Operation with (Min(0), Max(n))	OptionalVariationPoinActivity2V ariationOperation
Variant Activity	Variant Operation	VariantActivity2VariantOperatio n
Variation Point Object Type Data Input	Variable Complex Type	Variation Point Object Type Data Input2Variable Complex Type
Variation Point Activity	Variation Operation and Variable Message Type	VariationPointActivity2Variation Operation VariationPointActivity2Variable MessageType

- *The second step*: consists in defining the mapping rules. The VarSOAML models is mainly consisting of the serviceContract models and services interfaces models. The following rules allow the construction of those models :

- Collaboration2ServiceContract&ServiceInterface
- VariableCollaborationToVariableServiceContract
- CompositeActivity2Interface

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

- SimpleActivity2Operation
- VariationPointActivity2VariationOperation

1) Collaboration2ServiceContract &ServiceInterface

The collaboration represents an interaction between two pools, which exchange messages flows, through two composite activities performed by these two pools. This rule allows the transformation of a collaboration in the VariantRichBPMN to a service contract and also to a serviceInterface with the same description. The description of this rule is given in the following (cf. Figure 3).

Rule name:
Collaboration2ServiceContract&ServiceInterface
Inputs: Collaboration
Outputs: Service Contract, Service Interface
Description:
This rule creates instances of a Service Contract and of a Service Interface of the VARSOAml metamodel from the collaboration of the Variant-Rich BPMN metamodel.

- The name of the Message flow which connects two Composite Activities supported by two different Pools is the name of the Collaboration.
- The name of the Service Contract is the name of the Collaboration. If the type of Collaboration is "variable", then apply *VariableCollaboration2VariableServiceContract&VariableServiceInterface* Rule
- The names of the Roles (which are inherited from the UML Collaboration Meta-class) are the names of the Pools which are involved in the Collaboration.
- The name of the Service Interface is the name of the Collaboration.

Figure 3: Collaboration2ServiceContract&ServiceInterface Rule

2) VariableCollaborationToVariableServiceContract &VariableServiceInterface

The collaboration is variable when at least one of the two pools involved in this collaboration is a variation point. This rule allows the transformation of a Variable collaboration in the Variant-Rich BPMN to a Variable Service Contract and also to a Variable Service Interface with the same description. The description of this rule is given in (cf. Figure 4)

Rule name:
VariableCollaboration2VariableServiceContract& VariableServiceInterface
Inputs: Variable collaboration
Outputs: Variable Service Contract, VariableServiceInterface
Description:
This rule creates instances of the Variable Service Contract of the VARSOAml metamodel from the Variable Collaboration of the Variant Rich BPMN metamodel.

- The name of the Service Contract is the result of the concatenation of the name of the Collaboration and "Variable".
- The name of the Service Interface is the result of the concatenation of the name of the Collaboration and "Variable".

Figure 4: VariableCollaborationToVariableServiceContract &VariableServiceInterface Rule

3) CompositeActivity2Interface

A composite activity is transformed to an interface, and the list of the interface's operations is obtained by applying the rule 'SimpleActivity2Operation' for each simple activity of the composite one (cf. Figure 6).

Rule name:
CompositeActivity2Interface
Inputs: Composite Activity
Outputs: Interface
Description:
This rule creates instances of Interface of the VARSOAml meta-model from an instance of each Composite Activity of the Variant-Rich BPMN meta-model.

- Te name of the Interface is the result of the concatenation of the name of the Composite Activity and "Interface".
- The Operations of the Interface are obtained by applying the *SimpleActivity2Operation* for each Simple Activity of the Composite one.

Figure 5: CompositeActivity2Interface Rule

4) SimpleActivity2Operation

The rule 'SimpleActivity2Operation' allows the creation of an operation from an activity as described in the following (cf. Figure 6).

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Rule name: SimpleActivity2Operation
Inputs: Simple Activity
Outputs: Operation
Description:
This rule creates the instances of the Operation of the VARSOAml meta-model from a Simple Activity of the Variant Rich BPMN metamodel.

- The name of the Operation is the name of the Simple Activity.
- If the type of the Simple Activity is a Variation Point then apply *VariationPointSimple2VariationOperation* rule.
- Parameters of an Operation are the DataInput collection of an Activity.
- To define parameters of an Operation, apply DataInput2Parameter rule.

Figure 6: SimpleActivity2Operation

5) VariationPointSimpleActivity2VariationOperation

The rule 'VariationPointSimpleActivity2Variation Operation' allows the creation of a VariationOperation and its associated VariantOperation from a VariationPointSimpleActivity and its variants.

Rule name:
VariationPointSimpleActivity2VariationOperation
Inputs: Variable Composite Activity, Variation Point Simple Activity, Variant Simple Activity
Outputs: Variation Operation, Variant Operation of an Interface
Description:
This rule creates instances of Variation Operation and Variant Operation of the VARSOAml meta-model from the Variation Point Simple Activity and its Variants of the Variant-Rich BPMN meta-model.

- The name of the Variation Operation of an Interface is the name of each Variation Point Simple Activity of the Variable Composite Activity.
- If the Variation Point Activity is alternative then the name of the Variation Operation is the name of the Variation Point Simple Activity and the attributes min is equal to 1and max is equal to 1, else it is optional then the name of Variation Point Simple Activity +Min(0)+Max(n).
- For each Variant of the Variation Point Simple Activity, create Variant Operation of the associated Variation Operation.
- The name of the Variant Operation is the name of the Variant Activity of a Variation Point Simple Activity.

Figure 7:VariationPointSimpleActivity2Variation Operation

In order to validate our approach, we expose in the next section a case study for applying the proposed automatic generation.

5. CASE STUDY : E-HEALTHCARE CONFIGURABLE SERVICES GENERATION

For our case study, we use the configurable process model for E-healthcare systems, proposed in [21]. Our aim is to apply the proposed rules for generating contract service model and service interface model from a configurable process model of an E-healthcare system. The generated models will then be used to generate Variable web services by the application of second transformation of our approach [22].

At the **CIM layer,** we design a configurable process model (cf. Figure 8)which is conformed to the Variant Rich BPMN meta-model (cf. figure 2).

The CPM contains the following elements:

- **Pools**: Patient and Hospital, which are involved in the **collaboration** "Examination". The Pool "Hospital is a Variation Point of type, because it contains variation point activities "Perform examination" and "Treatment processing".

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

- **Collaboration:** "Examination" which is composed of a set of **Message flows** entitled "Examination".
- **Composite activities**: are "Examination" of the Patient Pool and "Examination processing" of the Hospital Pool.
- **Simple activities**: we have three simple activities in this example :
  - "Request for examination" which is contained in Examination **composite activity.**
  - And two others of type **Alternative Variation Points:** "Perform examination" and "Treatment processing", which are contained in the "Examination processing" composite activity.

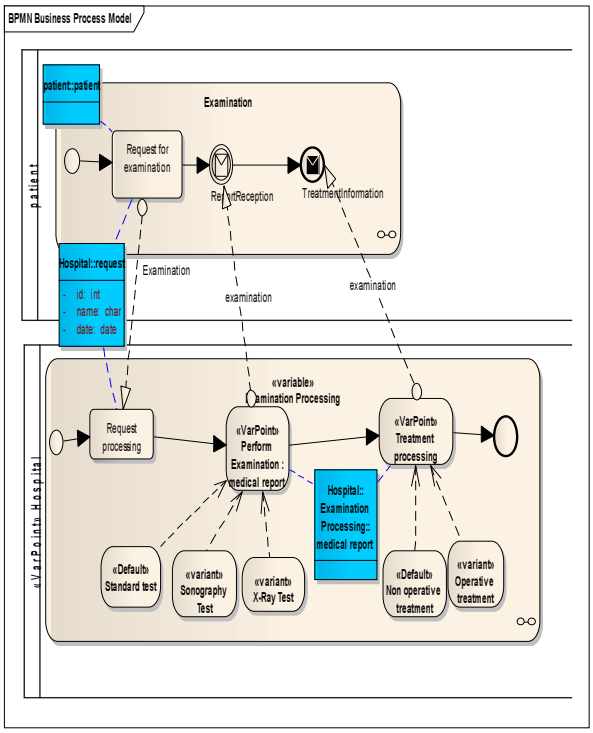

Figure 8: The configurable process model for an E-Healthcare system

To obtain the target configurable service models, we apply our proposed rules. The Table IV exposes the outputs corresponding to each input model of the E-Healthcare system example.

At the **PIM layer,** we obtain a service contract and a service interface models which conform to the VarSOAML meta-model proposed in [18]. The figure 10 describes the VariableContract Examination which interconnects the roles of the Patient and the Hospital. Each role is associated to an interface.

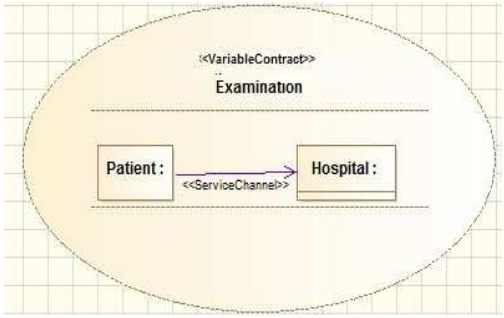

Figure 9: the Target variable service contract model

To illustrate the ServiceInterfaces obtained by the system, the figure 11 describes the service interface "Examination" which provides the interface "ExaminationProcessing" and uses the interface "PatientExamination". Each interface's operation corresponds to an activity in the process model. Hence, the Variation Point Simple Activity "Perform examination" (cf. Figure 8) is transformed to the variation operation of the "examination processing" interface (cf.

Figure 10).

By following all the development approach described in the figure 1, we will obtain for the PSM layer the WSDL and the variability specification corresponding to each serviceInterface identified in the PIM layer. For instance, the figure 12 illustrates the variability specification document associated with the Examination interface. The detail of the generation used in this level was previously explained in [18].

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

```
<variability service ="Examination" name ="
ExaminationVariability">
<operations>
  <variationOperation name = "PerformExamination"
min = "1"   max = "1" portype =
"ExaminationprocessingPortType">
     <variantOperations>
       <variantOperation name = "standardTest"
boundElement ="standardTest" mutexVariants=""
requiredVariants=""/>
       <variantOperation name = "x-RayTest"
boundElement ="x-RayTest" mutexVariants=""
requiredVariants=""/>
          <variantOperation name = "sonographyTest"
boundElement ="sonographyTest" mutexVariants=""
requiredVariants=""/>
     </variantOperations>
  </variationOperation>
 <variationOperation name = "treatementProcessing" min
= "1"  max = "1" >
… </variationOperation>
</operations>
<messages>
…
</messages>
<types>
..</types>
</variability>
```

Figure. 12: Variability Specification Document Associated With The Web Service Examination

6. CONCLUSION

With the evolution of information systems especially towards the PAIS, which is an information system based on services and business processes to achieve the business goals of enterprises, aligning business processes and services become very important. Indeed, establishing systematically this link can ensure responsiveness and flexibility of Information system. However, this kind of link becomes very complex with the apparition of configurable models that deal with variability as a way to increase models reuse among different contexts. Hence, to address this issue, we propose an MDA based approach for the generation of configurable service models from configurable process models. This work is a complementary approach of the work of generating configurable web services from configurable service models [22]. The configurable process models and the configurable service models are respectively specified using the Variant Rich BPMN and VarSOAML languages. We use these languages because they are extensions of BPMN and SOAml standards to support variability. We have proposed a set of mapping rules to transform the CPM to a configurable service model, focusing on the service contract and service interface models.

Our future work will give a prototype which enables the automation of the proposed generation, by using the ATL language (Atlas Transformation Language) for model to model transformation.

REFERENCES:

[1] M. Reichert, S Rinderle, U. Kreher, and P. Dadam, “Adaptive process management with ADEPT2”, *Proceedings ICDE'05*, 2005, pp: 1113-1114.

[2] B. Weber, S. Sadiq, M. Reichert, “Beyond rigidity - dynamic process lifecycle support”, *Computer Science*, 2009, Vol. n° 32, pp: 47–65.

[3] A. Delgado, I. García R. Guzmán, F., Ruiz, and M. Piattini, “Tool support for Service Oriented development from Business Processes”, *2nd International Workshop on Model-Driven Service Engineering, (MOSE’10)*, Málaga, Spain, 28 June-2 July, 2010.

[4] B. Elvesaeter, D. Panfilenko, S. Jacobi, and C. Hahn, “Aligning business and IT models in service-oriented architectures using BPMN and SoaML”, *Proceedings of the First International Workshop on Model-Driven Interoperability (MDI '10), Oslo, Norway. October 5, 2010.*

[5] W. Hummer, P. Gaubatz, M. Strembeck, U. Zdun, and S. Dustdar, “Enforcement of entailment constraints in distributed service-based business processes, *Information and Software Technology*, November 2013, Vol. 55, Issue 11, pp: 1884–1903.

[6] A. Kabzeva and P. M¨uller, “Toward Generic Dependency Management for Evolution Support of Inter-Domain Service-Oriented Applications”, *European Conference on Service-Oriented and Cloud Computing (ESOCC 2012)*, Bertinoro, Italy, 2012, pp: 35-40.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[7] A. Ratkowski, "Transformational Modeling of BPMN Business Process in SOA Context", *Advances in Intelligent Systems and Computing,* 2013, Vol. 224, pp: 335-344.

[8] Business Process Modeling Notation 2.0 Beta2. *Object Management Group*, 2011.http://www.bpmn.org.

[9] Service Oriented Architecture Modeling Language (SOAml), *OMG*, 2012.

http://www.omg.org/spec/SoaML/1.0.1, 12.

[10] M. Rosemann and W.M.P.Van der Aalst, "A Configurable Reference Modeling Language", *Information Systems*, 2008, Vol. n°32, pp: 1–23.

[11] F. Gottschalk, "Configurable Process Models", *PhD thesis, Eindhoven University of Technology*, Netherlands, 2009.

[12] M. La Rosa, "Managing Variability in Process-Aware Information Systems", PhD Thesis, *Queensland University of Technology*, Brisbane, Austria, 2009.

[13] A. Hallerbach, T. Bauer and M. Reichert, M., "Correct Configuration of Process Variants in Provop", *Technical Report, university of Ulm*, Germany, 2009.

[14] S.H. Chang and S.D. Kim, "A variability modeling method for adaptable services in service-oriented computing," *11th International Software Product Line Conference*, 2007. SPLC, 2007, pp. 261-268.

[15] M. Koning,, C.Sun, M. Sinnema, and P. Avgeriou" VxBPEL: supporting variability for Web services in BPEL", *Information and Software Technology*, 2009, Vol n°51, pp:258–269.

[16] C. Sun, R. Rossing, M. Sinnema, P. Bulanov, and M. Aiello, "Modelling and managing the variability of web service-based systems," *Journal of Systems and Software, Elsevier, 2010,* Vol n°83, pp. 502–516.

[17] N.C. Narendra, K. Ponnalagu, B. Srivastava, and G.S. Banavar, "Variation-Oriented Engineering (VOE): Enhancing Reusability of SOA-based Solutions," *IEEE International Conference on Services Computing, SCC'08, July 7-11, 2008.*

[18] B. Chakir, M. Fredj and M. Nassar , "Promoting reuse in web services by managing variability", *The 3rd IEEE International Conference on Multimedia Computing and Systems (ICMCS'12)*, Tangier, Morocco, May 10-12, 2012.

[19] Abu-Matar, and M., Gomaa, H., "An Automated Framework for VariabilityManagement of Service-Oriented Software Product Lines", *IEEE on Service Oriented System Engineering, (SOSE'13)*, San Francisco Bay, USA, March 25-28, 2013.

[20] W.M.P.Vander Aalst, "Configurable Services in the Cloud: Supporting Variability While Enabling Cross-Organizational Process Mining", *Coopis'2010*, Crete, Greece, October 24-26, 2010.

[21] F. Puhlmann, A. Schnieders, J. Weiland, and M. Weske, "Variability Mechanisms for Process Models", *PESOA Report*, 2005.

[22] B. Chakir, M. Fredj, and M. Nassar, "A model driven method for promoting reuse in SOA-solutions by managing variability", *IJCSI International Journal of Computer Science Issues*, Vol. 9,-Issue 3, No 3, 2012.

[23] C.W. Krueger "Variation Management for Software Production Lines", *SPLC, Springer*, 2002.

[24] VanderMaßen, T., Lichter, H., "Modeling variability by UML use case diagram", *REPL, AVAYA*, 2002.

[25] L. Gönczy, "Dependability Analysis of Web Service-based Business Processes by Model Transformations", *In Proc. of the First YR-SOC Workshop*, Leicester, United Kingdom, 2005.

[26] G. Benguria,, X. Larrucea, B. Elveseater, , T. Neple,, Beardsmore, A., Friess, M., "Platform Independent Model for Service Oriented Architectures. Enterprise Interoperability: New Challenges and Approaches", *Springer Verlag*, ISBN-10: 1846287138, 2007, pp: 23-32.

[27] C. Ayora, "Modelling and Managing Variability in Business Process Models", *Polytechnic university*, Valancia, Spain, 2011.

[28] C. Rolland, S. Nurcan, "Business Process Lines to deal with the Variability", *International Conference on System Sciences (HICSS)*, Hawaii, USA, January 5-8, 2010

[29] I. Berger, P. Soffer and A. Sturm, "Extending the Adaptability of Reference Models", *IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans*, 40(5), 1045–1056, 2011.

[30] C. Ayora, *V. Torres,* B. Weber, *M. Reichert and* V. Pelechano, " Enhancing Modeling and Change Support for Process Families through Change Patterns". *BMMDS/EMMSAD*, pp: 246-260.

[31] H. Sbai, M. Fredj, and L. Kjiri, "A pattern based methodology *for evolution management in business process reuse", IJCSI International Journal of Computer Science Issues, V*ol. 11, Issue 1, N°1, January 2014.

[32] H. Sbai, M. Fredj, and L. Kjiri, "Towards a process patterns based approach for promoting adaptability in configurable process models", *the 15th International Conference on Enterprise Information Systems (ICEIS 2013)*, 4-7 July, ESOE, Angers Loire Valley, France, 2013.

[33] M. Henkel and J. Zdravkovi, "Supporting development and evolution of service-based processes". *In: ICEBE*, IEEE CS, 2005, pp : 647-656.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[34] OMG. Model driven architecture guide v1.0.1. http://www.omg.org/mda, 2003.

[35] H. Sbai, M. Fredj, B. Chakir and L. Kjiri "On managing evolution of configurable services based on configurable processes", *the 2nd IEEE World conference on Complex Systems (IEEE WCCS'14),* November 10-11, Agadir-Morocco, 2014.

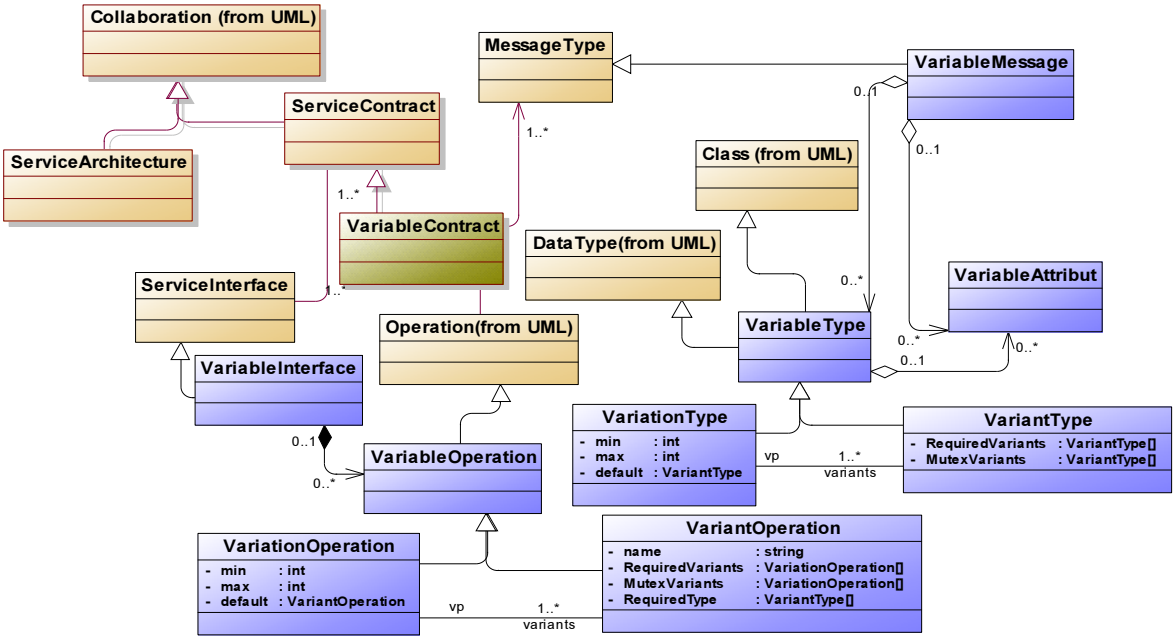

Figure 3: Overview of VarSOAML meta-model [18]

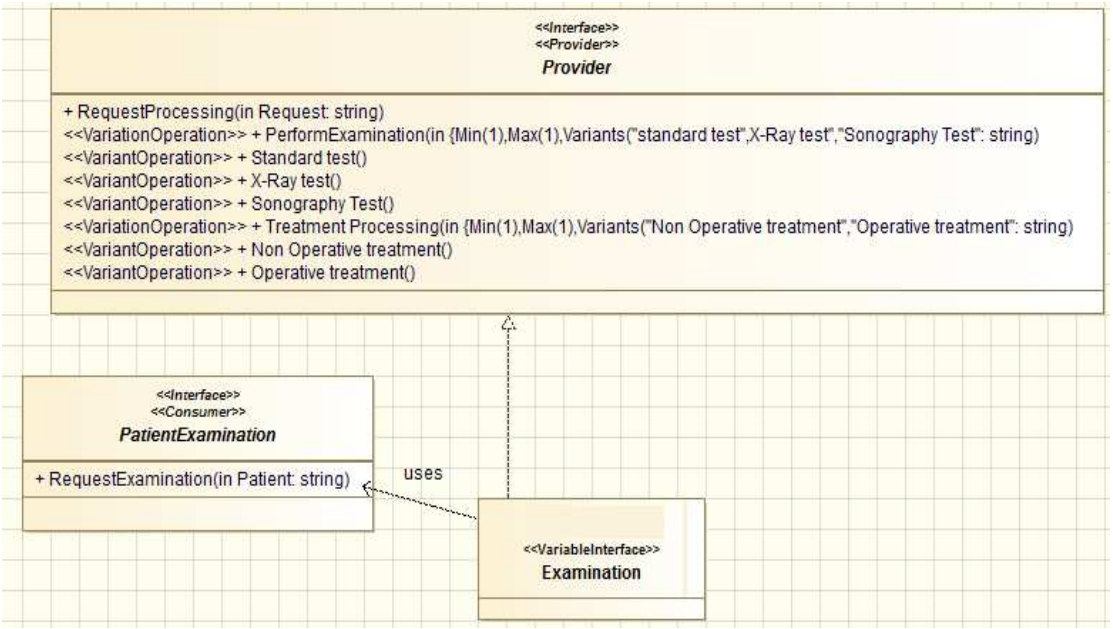

Figure 10: The target variable service interface model

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table II: VarSOAML concepts

VarSOAML concepts	Definitions
« VariableInterface»	Defines the service interface as a variable element which contains variation operations
«VariationOperation»	Specifies a variation point operation of a required or provided interface
«VariantOperation»	Specifies a variant operation which is the realization of a variation point operation
«VariableParticipant»	Represents a variable component which requires or provides service interfaces
«VariableContract»	Represents the variability of a collaboration
« VariableServiceArchitecture»	Represents the variability of the service architecture which contains variable elements (variable contracts or variable participants)
« VariationType»	Represents a variation point for a DataType or Class element
« VariantType»	Specifies a variant at the level of a DataType or Class
« VariableAttribute »	Specifies a variable element for a « Property »
« VariableMessageType »	Expresses a variable element for a «MessageType»
{min(1), max(1)}	Defines an alternative variable element
{min (0), max(n)}	Defines an optional variable element

Table IV: VariantRichBPMN2VarSOAML transformation

Input	Output	Corresponding transformation rules
Variable collaboration: Examination	Variable service contract and Variable examination Service interface	VariableCollaboration2 VariableServiceContract&VariableServiceInterface
Composite activity: Examination	Interface: Examination	Compositeactivity2Interface
Variable composite activity: Examination processing	Interface: Examination processing	
Variation point activity: Perform examination	Variation operation: Perform examination	VariationpointSimpleActivity2Operation
Variation point activity: Treatment processing	Variation operation: Treatment processing	
Variant Activities : Standard Test, X-Ray Test and Sonography Test	Variant operations: Standard Test, X-Ray Test and Sonography Test	VariantActivity2VariantOperation
Variant Activities: Operative Treatment, Non operative Treatment	Variant Operations: Operative Treatment, Non operative Treatment	

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Activity: Request for examination	Operation: Request for examination	Simpleactivity2operation
Activity: Request processing	Operation: Request processing	