An Operational Model of Variable Business Process

Raoul Taffo Tiam[1,2], Abdelhak-Djamel Seriai[1] and Raphael Michel[2]
[1]*LIRMM, University of Montpellier / CNRS, 161 rue Ada, 34090 Montpellier, France*
[2]*ACELYS, Business Plaza bât. 3 - 159 rue de Thor, 34000 Montpellier, France*

Keywords: Business Process, Variability, Reuse, Software Product Line, Operationalization, Standardization, Completeness, Expressiveness, Separation of Concerns, Feasibility, Industrialization.

Abstract: Software editors concerned to produce faster, better and cheaper, are irreversibly affected by the development of product lines (software factory). The software product line approach offers techniques to increase reuse by explicitly modelling the common and variable characteristics. Considering this approach, the variability is modelled and managed throughout all stages of development. Thus, models of variable business processes are part of the design artefacts in analysis stage. Several models have been proposed to represent variable business processes. However, these models are far from being directly usable in real industrialization of production in software factory. Indeed, deficiencies such as non-representation of variability on all entities of business processes, not taking into account all the possible types of variability, or use proprietary languages, prevent those models to be operational. In this paper, we present these barriers to the operationalization and propose solutions to overcome each of them. The result is an operational model of variable business process, actually used and integrated in a software factory approach.

1 INTRODUCTION

A Software Product Line (SPL) is a way to effectively design a software portfolio, taking full advantage of the similarities between products, while respecting and managing their specificities (van Gurp et al., 2001). The originality of this approach is that it systematizes reuse, proactive reuse, through two complementary development processes: domain and application engineering. Firstly, domain engineering models reusable platform from which various products will be built by explicitly specifying both the common characteristics (*features*) and those variable (specific). It can then produce artefacts (*assets*) for all of these characteristics. Secondly, application engineering allows to derive (configure) a desired product by selecting the appropriate characteristics for the proposed platform. Both processes have the same development activities than conventional processes, such as specification, analysis, design, implementation or testing. Thus, artefacts may be, analysis or design models, design patterns, software architecture, source code, test plans, testing units, documentation, etc. (Krueger and Clements, 2013).

Under this approach, models of variable business process are among the artefacts produced during the analysis phase. A variable business process (or reference business process) is the representation of common and specific elements of several variants of the same business process. Those different variants, or this variability, appear because of fluctuations around the business environment such as legislation, globalization or rationalization. It is therefore important for enterprises to be able to withstand these changes. However, to remain competitive enterprises need to adapt their business processes without changing them completely, it is flexibility (Schmidt et al., 2008). This is made possible through variable business process. So, it is necessary to express and manage variability to ensure better coverage of the target market. A business process linked to an enterprise information system (IS) or to a specific application, is a set of more or less related activities that collectively realize a business goal while defining the roles and relationships of each resource. The business process models capture that coordination of activities. They thus serve as relay between the requirements specification and software used to meet these requirements.

«…*Significant research efforts have been made throughout the last decade, leading to an abundant*

Taffo Tiam R., Seriai A. and Michel R..
An Operational Model of Variable Business Process.
DOI: 10.5220/0005372301620172
In *Proceedings of the 17th International Conference on Enterprise Information Systems* (ICEIS-2015), pages 162-172
ISBN: 978-989-758-098-7

production of variable business process models.» (La Rosa et al., 2013). Despite this fertility, modelling variability in business processes remains a major challenge. Indeed, for such models to be really used in an industrial context, some properties are essential. For example, these models should not be owners, but rather standardized to enable integration with other artefacts produced at other development stages. In addition, they must deal with the variability in its entirety, taking into account all types of variability, or propose solutions readily available and maintainable.

Unfortunately, many of the expected properties, see section 2, are not considered or completed in variable business process models proposed in the literature to make them operational (Ayora Esteras, 2011; La Rosa et al., 2013). Our work aims to bear most of these shortcomings by subscribing to the following objective: making models of variable business process operational, i.e. truly usable in a software factory.

This paper is organized as follows. Section 2 outlines obstacles to operationalization of variable business process models, Section 3 presents our proposal, Section 4 discusses related work, and Section 5 concludes the paper.

2 REQUIRED PROPERTIES FOR OPERATIONALIZATION OF VARIABLE BUSINESS PROCESS MODEL

The goal here is to characterize a solution that takes into account essential and unavoidable properties to the operationalization of variable business process models. In this paper, we mainly consider the following properties: standardization, completeness, expressiveness, separation of concerns (requirements vs business processes), and feasibility (validation and tools).

2.1 Standardization

To standardize a product is to comply with a given standard or reference. Such an effort provides many benefits which rank the reliability, maintainability, uniformity, interoperability, and the use of good practices.

In a software factory, this criterion is essential as it brings together several activities and very often separate processes or existing tools. Therefore, the flow of information between the individual tools and its integrity from one tool to another usually follows the evolution of the tool, it is then performed improperly or not at all. Due to the lack of adequate equipped supports and integrated processes (lack of standardization), the same work is done repeatedly, with a lack of coherence and synchronization, as well as duplication.

About the variability we are seeing the emergence of a standard, CVL – Common Variability Language (Haugen et al., 2012; Haugen et al., 2013), proposed for its specification, both centred at the user level and realization of product. It is therefore recommended, even required, to refer to this standard too for specification of variability in business process models.

2.2 Completeness

The completeness concerns the possibility to specify the variability in all components of the business process in which it may appear. The standard ISO ENV 40003 (ISO, 2002) defines four views (perspectives) to model enterprise business processes: functional, informational, organizational and resources. Thus, a business process is specified by its activities, events that trigger them, flows that interconnect them, actors involved, business objects that are used or produced and control nodes that coordinate the exchange of information. We must be able to explain the variability that can occur on any one of these components.

Partial specification of variability in business process models is an obstacle to operationalization because unconsidered components are ignored or at best, inappropriately treated.

2.3 Expressiveness

The expressiveness of a solution to specify variability is the ability to express naturally, all types of variability in variable business process models. To offer a wide range of natively concepts not only contributes to the richness of a solution, but also to its simplicity. This criterion is especially important in an industrial context, where models are usually very complex and voluminous.

The general techniques for realization of variability are processed in (Svahnberg et al., 2005), and initially categorized by (Schnieders and Puhlmann, 2006) in the case of business processes. We consider that the variability can minimally be expressed by the following forms:

- *basic*: adding / removing / replacing element, reorganization, typing, setting;

- *composite*: generalization / specialization, expansion / extension point, composition / decomposition, generation, pattern design;
- or *conjunctural*: conditional, temporal.

To make operational models of variable business process, we are therefore concerned by the ability to express all these forms of variability.

2.4 Separation of Concerns: Requirements Vs Business Processes

Separation of concerns addresses the ability to separate components into elementary functions, so they are considered or performed separately. This criterion is even more important in industrial world as it increases the simplicity and reusability of designed artefacts, reduces errors and promotes teamwork. Variability discriminates the *features* of a product in four categories (van Gurp et al., 2001):

- *mandatory*;
- *optional*;
- *variant*;
- and *external*.

Considering this decomposition, we evaluate existing solutions in terms of their compliance with these categorizations. Thus, for a solution to meet this criterion of separation of concerns, it is necessary that the considered elements are not only internal but also external to business processes.

2.5 Feasibility: Validation and Tools

This last criterion is important for the operationalization of variable business process models, and also the applicability and adoption of an approach in industry. This is why we consider two aspects of the Feasibility validation (scaling) and tooling.

Validation concerns the implementation of the proposal on a real industrial case, with a preference when the field is validated by experts or issued from an authoritative repository. We are also interested in availability of tools based on the approach.

The smooth adoption and use in industry of variability specification solution in business process models strongly depends on the scalability of the approach and access to tools for implementation.

3 PROPOSITION

In the following, we explain our approach as well as contributions to the operationalization needs initially expressed.

3.1 Case Study

To facilitate understanding of our proposition, we introduce the case study, an industrial project of software product line in service centre management. In IT systems, a service centre management tool is an interface between an IT service supplier, and the service customer which aims at processing requests for services through a centralized portal. The customer contracts a service catalog, allowing him to choose his service and to know his type parameters upstream for the control (expenses, unit of work, statistics ...), rather than ordering services independently of each other. This tool allows the service provider to standardize framework for better management of service requests, capitalize on the provided services, receive feedback and thus improve both service delivery and profitability.

The ITIL repository defines twenty-two business processes for management and optimization of IT service centres. These business processes involve either services production, governance or management of their life cycles, and they harbour variability. For example, the business process of *Service Request* always starts with a *Request* activity carried out by the customer, follows up with the provider's synchronized activities: *Production* of service and a transversal *Pilotage*. When it is a normal service requested, *Production* consists in a service *Realization*; if it is an incident, it would rather be a *Reparation*; *Resolution* when it is a request to solve a problem; and *Replacement* to a request for change. All those options for *Production* activity are manifestation of variability. Our approach should allow to express and specify this variability in a "variable" business process model. In addition, the model must have required properties for operationalization.

3.2 Standardization

To specify and represent variability in business process models, we implement the CVL standard for specification of variability. CVL is a generic and independent specification to model the variability in models throughout any DSL (Domain Specific Language) defined based on the MOF – Meta Object Facility (ISO/IEC, 2014). It is representative of ongoing efforts involving both the academic community and industry stakeholders to promote the standardization of variability modelling technology.

This specification includes a CVL meta-model, semantics and concrete syntax for expressing variability. CVL consists of three models:

- the *base model* – a model described in UML (ISO/IEC, 2012), BPMN (ISO/IEC, 2013), or any other DSL;
- the *variability model* – a model describing the variability that exists within the *base model*;
- and the *resolution model* – the model that describes how to resolve the variability and create a new model in the base DSL.

The principle of using these models is as follows: a *base model* may have more *variability models* and each *variability model* can have multiple *resolution models*. With *variability model* and *resolution model* properly defined, the user can run a generic CVL model-to-model transformation to generate new resolved models that conform to the *base model*. To represent *features mandatory* or *optional*, with links *requires*, *excludes*, *xor*, *or*… and the cardinality at user-centred level, concepts such as *type*, *composite variability*, *constraint* or *iterator* are offered. While in product realization, concepts such as *placement / replacement fragment*, *fragment substitution*, *value substitution* or *reference substitution* are proposed to make the transformation of a variable model in a resolved one.

Our standardization effort in the variability specification in variable business process models consists in projection and adaptation of this CVL standard to business processes. Thus, we propose three independent models to specify common, optional and variable features: a *BaseModel* from which it is possible to build one (or more) *VariationModel*, which themselves can entail one (or more) *ResolutionModel*.

Our basic model – *BaseModel* – represents the common characteristics in "families" of business processes.

The variability model – *VariationModel* – describes the variability in the *BaseModel*, in other words the variable characteristics. It indicates that variability using *placement fragments*.

The resolution model – *ResolutionModel* – describes how to resolve the variability and create a new model in the base DSL. It defines with *replacement fragments*, alternatives or options to resolve the variability indicated by *placement fragments*.

Those three models are represented in a synthetic model, a "variable model or reference model", which has both the common *features* of *BaseModel*, optional features of *ResolutionModel* and variable *features* of *VariationModel*. It is described in a DIML© (Domain Independent Modelling Language), which we define based on MOF. This business process modelling language inspired by UML AD and BPMN regardless of their fields, is proposed to bring together the respective communities of IT and business stakeholders. This will be detailed in a forthcoming article.

To specify variability of our previous case study example of business process *Service Request*, we design its reference or variable business process model, including *features* from our three different models, see Figure 1.

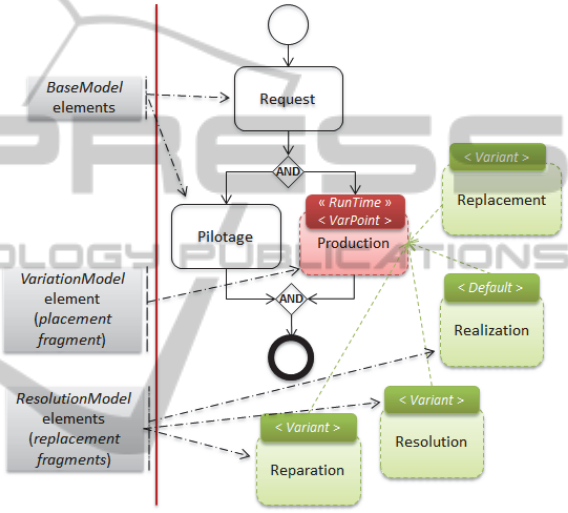

Figure 1: Reference model of business process *Service Request*.

Request and *Pilotage* activities are common *features* as described before, and *Production* activity is variable. Others activities *Replacement*, *Realization*, *Resolution* and *Reparation* are options to replace the variable element.

We use these concepts uniformly for variability specification in all business process aspects.

3.3 Completeness

As mentioned, variability may appear in all components of business process, in activities, events, flow controls or gateways of the same business process as well as in manipulated objects and solicited actors. We propose to use our standardized models consistently across all these components. Examples of Figure 2 and Figure 3derived from our case study illustrate the variability that may appear respectively in the objects (business entities) and actors (roles). This variability must be specified and

represented in the objects and actors, for example, in the same manner as in the variable activity of business process illustrated previously.

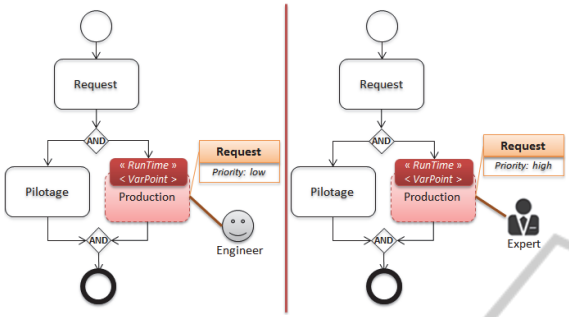

Figure 2: Variability appearance in actors.

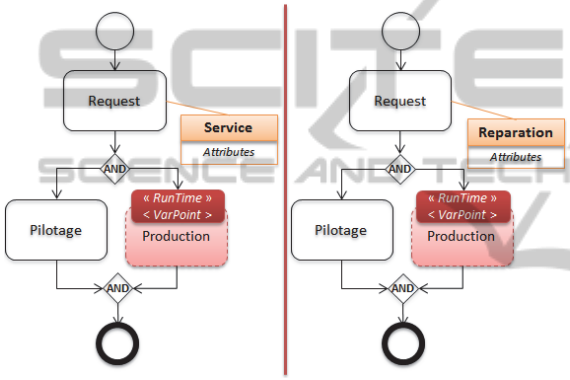

Figure 3: Variability appearance in objects.

As regards the variability in objects, we obtain the following reference model, cf. [Figur]. Variable object *feature* is specified with *placement fragment* of *VariationModel* and optional *features*, *Service* and *Reparation*, are specified with *replacement fragments* of *ResolutionModel*.

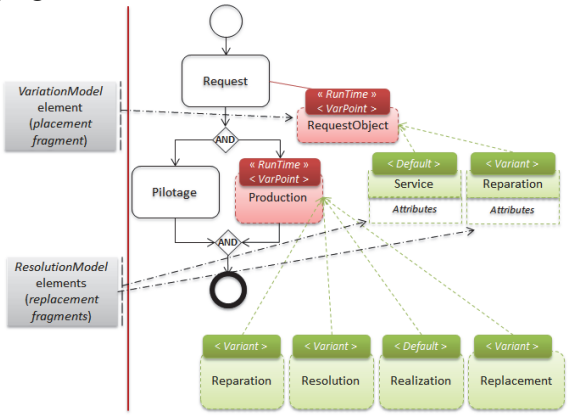

Figure 4: Variability specification in objects.

3.4 Expressiveness

Variability is not expressed in the *BaseModel* which specifies common features, but rather on variable and optional *features*, so respectively through *VariationModel* and *ResolutionModel*.

3.4.1 Expressiveness in *VariationModel*

To express variability in our *VariationModel*, several mechanisms proposed by CVL are implemented, such as *placement fragments* that correspond to the basic category (add / delete / replace *feature*).

However, there are different abstraction levels in business processes, corresponding for example to encapsulation of business sub-processes. We introduce two new concepts that specialize the concept of *placement fragment*, *Variable* and *VarPoint*, to manage variability specification on different abstraction levels. Thus, the *Variable* concept indicates variability at a high level, while *VarPoint* specifies variability at a lowest level. A *Variable feature* denotes variability at a high level and is composed of one or more *placement fragment*(s). These concepts belong to the composite category of variability realization mechanisms.

The example of our case study in figures below, see Figure 5 and Figure 6, reflects the variability that appears on different abstraction level in business process *Service Request*. Variability which appears on business activity (sub-process) *Pilotage* at high abstraction level is specified by *Variable*, and contains a variable task (activity) *Crop* specified by *VarPoint* at low level.

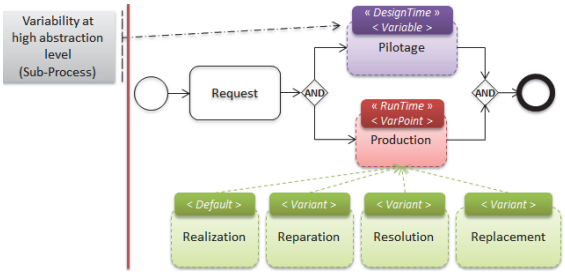

Figure 5: Variable feature – variability specification at high abstraction level (sub-process).

Environmental factors that characterize the variability in specified points are reified in item(s) which belong to *VariationModel* too, representing the context of variation – *VariationContext*. It is formulated by expressions – *ContextExpression*, which are variables of considered domain. This is a conjunctural mechanism of variability specification,

and conditional category more specifically.

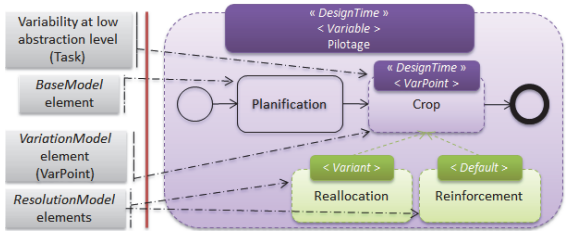

Figure 6: *VarPoint feature* – variability specification at low abstraction level (task).

For example in our case study, the availability of services should be taken into account as shown here Figure 7, specifying that an *Escalation* way in steering is required in case of power failure.

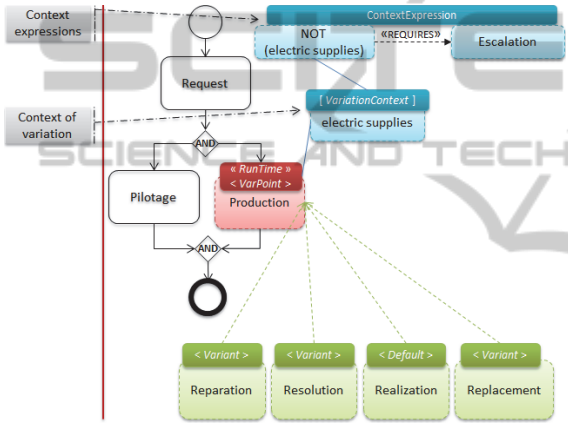

Figure 7: Variation context specification in reference business process model of *Service Request*.

3.4.2 Expressiveness in *ResolutionModel*

Several variability realization mechanisms are also implemented in our *ResolutionModel*, like *replacement fragments* which correspond to the basic mechanisms of adding / removing / replacing artefact. However, these concepts must also be specialized to match the varying business processes.

Thus, a *replacement fragment* is either a normal variant specified by the concept *Variant*. In certain case default variant exists (i.e. the mostly used variant to resolve variability in a business domain) and is specified with the concept *Default*. These *replacement fragments* are associated with the variability points they resolve.

In the example (business process *Service Request*) of our case study, the different variants (options) of the variable activity *Production* are specified on this variability point, see Figure 8.

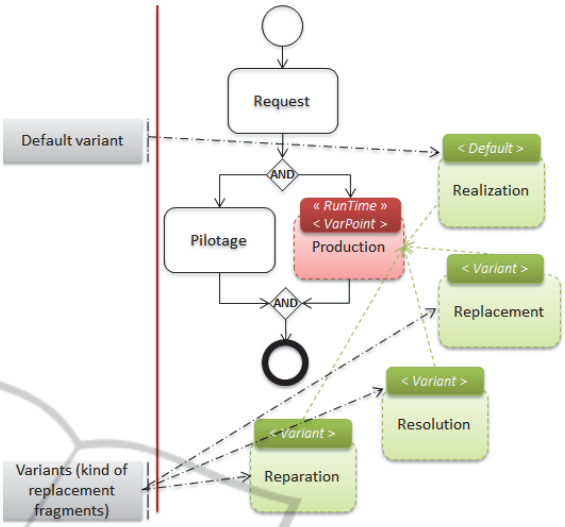

Figure 8: *Variants* and *Default* specification.

Configuration constraints are also specified in our *ResolutionModel*, because they allow the validation of resolved models and help to automate the configuration operations. This variability realization mechanism belongs to the base category because it refers to reorganization. Constraints may be specified indiscriminately on variability points corresponding to variable business process components.

As an example from our case study, Figure 9, the constraints indicate that *Resolution* variants and *Realization* are in mutual exclusion, while the choice of *Resolution* variant requires the one of *Reparation*.

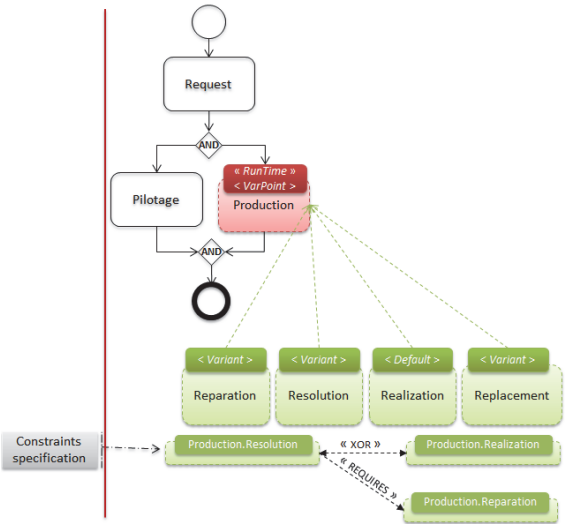

Figure 9: Constraints specification in reference model.

To manage the variability throughout the lifecycle of business process, the variability

resolution time – *ResolutionTime*, must also be specified in *ResolutionModel*. It is a temporal form of variability, so it corresponds to the conjunctural category. That time is at analysis / design – *DesignTime*, or at execution – *RunTime*.

In our case study for example, changes, problems or incidents can be requested as services and thus specificities will only be known at runtime, so the variability resolution time is *RunTime*, see Figure 10.

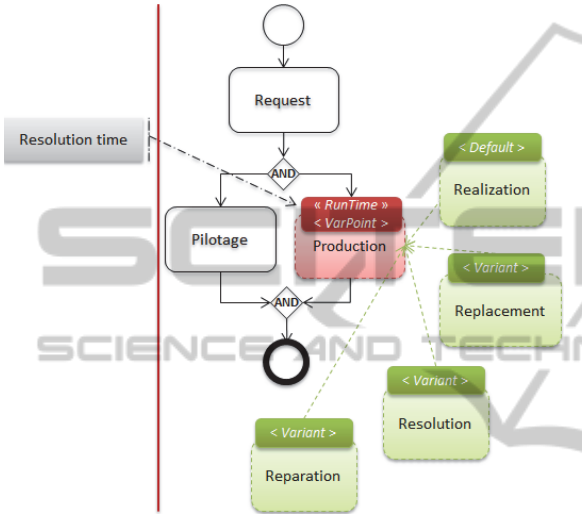

Figure 10: Resolution time specification in reference model of business process *Service Request*.

3.5 Separation of Concerns: Requirements Vs Business Processes

Elements considered so far to express and represent the variability within business process models are internal. We must then identify and treat separately the external factors related to variability specification in business process models. In our concern, these external factors are requirements. Indeed in software factory activities, requirements specification precedes the modelling of variable business processes. Thus, during the product development, requirements specification according to customer particular needs influences the variability resolution in models of variable business processes.

This approach matches external *features* proposed in (van Gurp et al., 2001) article, with the particularity that information discriminated as external *features* in variable business process models is requirements. Therefore, when those *features* (requirements) are taken into account, they should not be buried within the variable business process models but reified in a separate model, see Figure 11. In fact, this solution must be structured through a traceability link proposal, particularly between variable models of requirements and business processes.

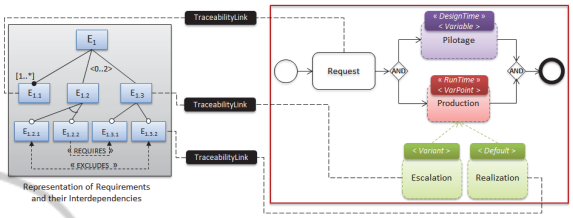

Figure 11: Separation of concerns.

Now, choices of upstream requirements allow obtaining new constraints to consider for variability resolution of downstream variable business processes. This separation effort is essential for the reasons mentioned above in description of operational properties expected in variable business process models. Furthermore, automation of the variability resolution in software factories also requires consideration of such factors separately from variable business process models.

3.6 Feasibility: Validation and Tools

To rate the soundness of our approach, we implement it through a prototype and also validate it on an industrial case.

3.6.1 Tools

We develop a prototype to implement our approach using Eclipse PDE (Plugin Development Environment) with an Eclipse RCP (Rich Client Platform) as target environment. Developments are based on EMF (Eclipse Modelling Framework), GMF (Graphical Modelling Framework) and OCL (Object Constraint Language) (ISO/IEC, 2012). EMF is used to define the abstract syntax, GMF the concrete syntax and OCL specifies static semantics, see Figure 12.

Component 1 defines vocabulary (concepts) and semantics of our modelling language (DIML) of variable business processes. Its development is based on the EMF framework. The sub-component 1 translates in OCL the rules and constraints to resolve variability. Component 2 defines the notation corresponding to previously proposed vocabulary. It is developed using the GMF framework. Component 3 designs the target platform for using previous

components and model variable business processes. It creates four different views (perspectives) and toolboxes for designing these models. Its development is based on Eclipse RCP technology. Designed variable business process models are persisted in files conforming to standard XMI (XML Metadata Interchange).

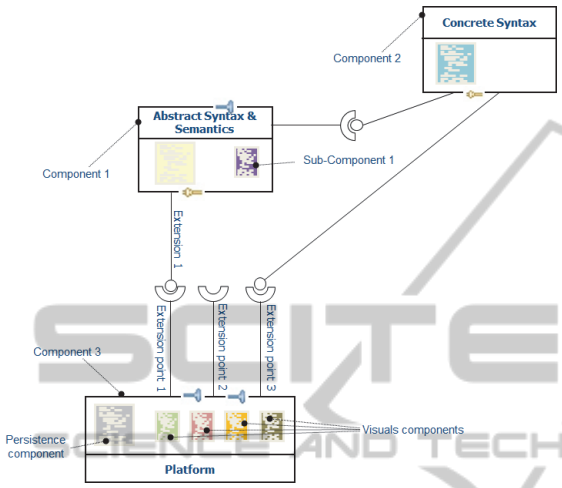

Figure 12: Prototype architecture.

3.6.2 Validation

The complete approach we propose has been implemented to specify the variability of business processes in an industrial case study described above. We illustrate some of the most representative results obtained with the prototype.

The reference model of transverse business sub-process *Pilotage* is set in Figure 13 with some configuration constraints. These constraints are translated into OCL to automate the validation of configurations. The commitment deadlines relating to service level agreement are *TTO* (Time To Own) and *TTR* (Time To Realize / Repair / Resolve / Replace), where *TTR* is variable.

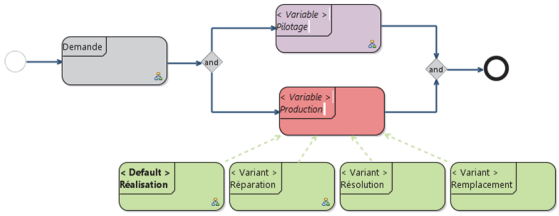

Figure 14: Extract of reference model of variable business process named *Service Request*.

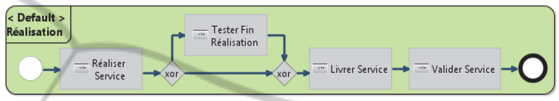

Figure 15: *Realization* – default variant of *Production*.

We are able to specify and represent all the variability that appears in the service centre business processes. The resulting models were taken over and integrated by business referents and the other stakeholders of project owners (client). Following different learning curves, these models were used by the engineers of the company (service provider) to support analysis before developing the final product.

4 RELATED WORK

Several research studies have focused on the variability specification within business processes. We selected five of the most significant and representative approaches. C-EPCs (Configurable Event-driven Process Chains) proposed by Rosemann and van der Aalst (2003), then enriched by La Rosa et al., (2008; 2011) with the language

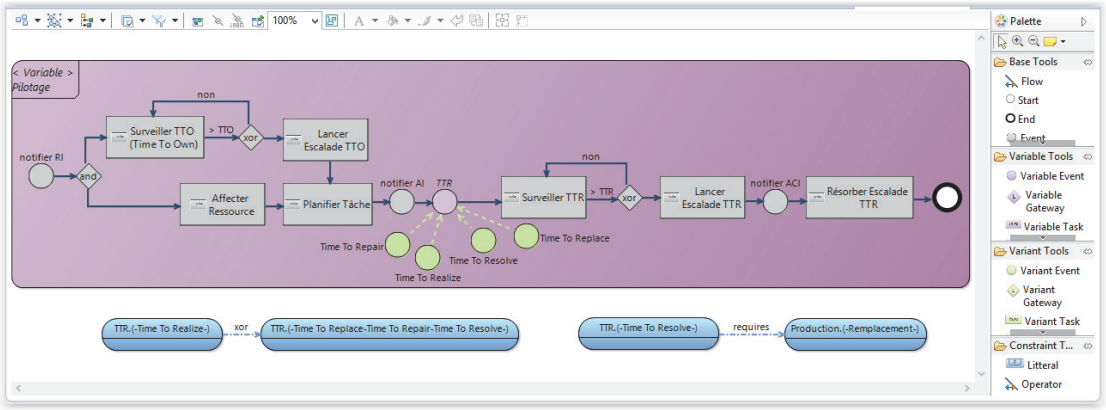

Figure 13: Details of reference model of variable business sub-process *Pilotage* with some constraints.

C-iEPCs (C *integrated* EPCs); PESOA approach or VRPM (Variant-Rich Process Models) (Puhlmann et al., 2005; Schnieders and Puhlmann, 2006); BPFM (Business-Process Family Model) (Park et al., 2009); PROVOP (PROcess Variants by OPtions) approach (Hallerbach et al., 2010); and Ayora et al., works (Ayora Esteras, 2011; Ayora et al., 2012). We analyse the shortcomings of these approaches to the operationalization properties of the models they offer.

4.1 Standardization

The C-EPCs approach uses its own C-iEPCs language for modelling variable business processes, making it an *ad-hoc* and proprietary solution which entails standardization.

This criterion is considered in VRPM and BPFM approaches which offer possibilities to use standards like BPMN or UML AD by proposing independent solutions of the business process modelling language. Therefore, they do not use the standard for variability specification, but languages inspired by FODA – Feature-Oriented Domain Analysis (Kang, and al., 1990) called «*FODA-like*» languages.

The PROVOP approach defines a set of structured blocks among others to represent the base model, its options and configuration context. It is an *ad-hoc* and owner language, thereby giving unstandardized properties.

Ayora et al., propose a solution independent of the business process modelling language and push this effort to standardize the variability specification language. However to specify variability, they use two formalisms: CVL standard and a *FODA-like* language. This clearly indicates a lack of uniformity in their variable business processes models, and a lack of standardization as well.

4.2 Completeness

With contributions of La Rosa and al., the C-EPCs approach allows to specify variability in roles (organizational resources) and manipulated objects. So, this approach is able to specify variability in every business process component, that is completeness.

VRPM approach does not allow to specify variability in control nodes (gateways), nor in the organizational resources (actors and roles).

BPFM approach can only specify variability in variable flow and activities of the business process, so it does not consider all business process components where variability can appear.

In PROVOP it is only possible to specify the variability points at input or output of control nodes. Therefore, variability specification in organizational resources and objects is only partially treated as they are modelled as activities attributes.

Ayora et al., take this criterion into account from the beginning and manage all component parts of business processes, allowing them to validate completeness in variability specification.

4.3 Expressiveness

In C-EPCs it is possible to specify a variable model composed by union of all variants, which ignores for example information on reference model or default variants. This approach uses few basic mechanisms of realization of variability but not composite nor conjunctural technics.

VRPM approach models a reference business process in which it is possible to specify common, optional, variable, and default elements, as well as variability points. They use basic and composite mechanisms for realization of variability but not conjunctural ones.

BPFM is an approach that perceives the basic variability realization technics, but we deplore absence of several composite and conjunctural mechanisms.

The PROVOP approach offers basic mechanisms of variability realization but suffers from shortcomings regarding composite and conjunctural ones.

Ayora et al., works are expressive enough but that expressiveness failed for example to indicate variability on different abstraction levels such as with encapsulation (not achieving all composite variability mechanisms).

4.4 Separation of Concerns: Requirements vs Business Processes

On this property we note that apart from the C-EPCs approach, there is no solution in related works that satisfies this separation of concerns between *features* of requirements and variable business processes. Indeed, in existing models of variable business processes the *features* considered are mainly internal to business process like uncorrelated to other artefacts produced by related activities of software factory as requirements specification or design. In the best case of C-EPCs approach those features are considered but buried within variable business process models.

4.5 Feasibility: Validation and Tools

C-EPCs approach satisfies both aspects of feasibility since it has been validated on a real case from film industry and is implemented by *Synergia* tool.

The VRPM approach for its part has been applied to an industrial hotel reservation case but is only partially implemented by an Eclipse plugin.

Meanwhile BPFM approach is not validated on an industrial case yet and its implementation via an Eclipse plugin is partial.

The PROVOP approach is partially implemented by *ARIS* tool and has two applications cases in automotive and health domains. However, the areas of these industrial cases were not validated by domain experts or issued from an authoritative repository.

Finally Ayora et al., works are applied to a business process of admission in university, so it is not an industrial case, and its implementation via the *MOSKitt* tool is not available yet.

4.6 Synthesis

We analyse state of the art solutions in terms of key properties for industrialization of variable business process models. This assessment is summarized in the table below where scores are assigned based on the level of satisfaction of the above properties.

Abbreviations: Standardization (Sta), Completeness (Com), Expressiveness (Exp), Separation of Concerns (Sep), Feasibility (Fea).

Table 1: Evaluation of existing approaches.

	Sta	Com	Exp	Sep	Fea
C-EPCs	0	1	½	½	1
VRPM	0	0	½	0	½
BPFM	0	0	½	0	0
PROVOP	0	½	½	0	½
Ayora and al.	½	1	½	0	0
Our Approach	1	1	1	1	1

Legend
0 ≡ unsatisfied property;
½ ≡ property partially satisfied;
1 ≡ satisfied property.

We note that no other approach than ours satisfies all the required properties for an operational model of variable business process, in fact they all have at least one unsatisfied property. Thus, our proposal for variability specification in business processes addresses an issue where the challenge remains topical.

5 CONCLUSIONS

We performed an independent solution of business process modelling compatible with standards such as BPMN and UML. Our approach for variability specification in business process models satisfies the key properties required for operationalization (effective use) of designed models in software factories. Indeed, we project the CVL standard for variability specification and adapt it to business processes. The standard is used uniformly on all variable business process component. This is the standardization property. We take into account all business process components in which variability may appear: completeness. Our proposition offers a native and natural way to specify all forms of variability throughout the lifecycle of business processes: expressiveness. Our approach provides a beneficial separation of concerns, separating features from requirements and variable business processes in separate models which contribute to modularity. Finally, we take advantage of the enterprise context of this research to apply our proposals to a case study corresponding to a real industrial project. To further validate the feasibility of our approach, we prototype a tool that supports our proposition.

In our further work we will address the problem of traceability between requirements and development models, particularly variable business process models. As requirements specification remains a separate activity from the design and implementation of software in the development process, this separation negatively impacts product quality and customer satisfaction. It is then required to operationalize the requirements specification and identify at the same time traceability links between requirements and variable business process, allowing to take into account these external *features* in variability resolution. In fact, the explicit management of such information is a decision support for the selection of appropriate variants. It can help in monitoring the fulfilment of requirements of product or customer as well as the computation of impact in cost and delays for example.

ACKNOWLEDGEMENTS

We would like to thank the National Association of Research and Technology (ANRT in French) for its contribution to this research.

REFERENCES

Ayora Esteras, C., 2011. *Modeling and Managing Variability in Business Process Models,* Valencia: Polytechnic University of Valencia.

Ayora, C., Torres, V., Pelechano, V. & Alférez, G. H., 2012. *Applying CVL to Business Process Variability Management.* New York, NY, ACM, pp. 26-31.

Hallerbach, A., Bauer, T. & Reichert, M., 2010. Capturing Variability in Business Process Models: The Provop Approach. *Journal of Software Maintenance and Evolution,* XXII(6-7), pp. 519-546.

Haugen, O., Wasowsk, A. & Czarnecki, K., 2012. *CVL - Common Variability Language.* Proceedings of the 16th International Software Product Line Conference (SPLC '12).

Haugen, O., Wasowski, A. & Czarnecki, K., 2013. *CVL: Common Variability Language.* New York, NY, ACM, pp. 277-277.

ISO/IEC, 2012. *Object Constraint Language (OCL) 2.3.1.* Information Technology - Object Management Group (OMG).

ISO/IEC, 2012. *Unified Modeling Language (UML) 2.4.1.* Information Technology - Object Management Group (OMG).

ISO/IEC, 2013. *Business Process Model and Notation (BPMN) 2.0.1.* Information Technology - Object Management Group (OMG).

ISO/IEC, 2014. *Meta Object Facility (MOF) Core 2.4.2.* Information Technology - Object Management Group (OMG).

ISO, 2002. *Enterprise Integration - Framework for Enterprise Modeling.* International Standardization Organization.

Kang, K. C. and al., 1990. *Feature-Oriented Domain Analysis (FODA): Feaseability Study.* Technical Report CMU/SEI-90-TR-21 ESD-90-TR-222.

Krueger, C. & Clements, P., 2013. Systems and Software Product Line Engineering with BigLever Software Gears. New York, NY, *ACM*, pp. 136-140.

La Rosa, M. and al., 2008. Beyond Control-Flow: Extending Business Process Configuration to Roles and Objects. *Lecture Notes in Computer Science,* Volume 5231, pp. 199-215.

La Rosa, M., Dumas, M., ter Hofstede, A. H. & Mendling, J., 2011. Configurable multi-perspective business process models. *Information Systems,* XXXVI(2), pp. 313-340.

La Rosa, M., van der Aalst, W. M., Dumas, M. & Milani, F. P., 2013. *Business Process Variability Modeling: A Survey,* Brisbane: QUT ePrints.

Park, J., Moon, M., Yun, S. & Yeom, K., 2009. An Approach to Enhancing Reusabilities in Service Development. New York, NY, *ACM*, pp. 143-150.

Puhlmann, F., Schnieders, A., Weiland, J. & Weske, M., 2005. *Variability Mechanisms for Process Models,* s.l.: PESOA (Process family Engineering in Service-Oriented Applications).

Rosemann, M. & van der Aalst, W. M., 2003. A Configurable Reference Modeling Language. *Information Systems,* XXXII(1), pp. 1-23.

Schmidt, R., Regev, G. & Soffer, P., 2008. Requirements for Flexibility and the Ways to Achieve It. *International Journal on Business Process Integration and Management, III*(1), pp. 1-4.

Schnieders, A. & Puhlmann, F., 2006. *Variability Mechanisms in E-Business Process Families.* pp. 583-601.

Svahnberg, M., van Gurp, J. & Bosch, J., 2005. A Taxonomy of Variability Realization Techniques: Research Articles. *Software - Practice & Experience,* 35(8), pp. 705-754.

van Gurp, J., Bosch, J. & Svahnberg, M., 2001. On the Notion of Variability in Software Product Lines. Washington, DC, *IEEE Computer Society*, pp. 45.