

Modelling Flexible Collaborative Process: The VCP2M Approach

Fatma Ellouze¹, Mohamed Amine Chaabane¹, Eric Andonoff² and Rafik Bouaziz¹

¹MIRACL, University of Sfax, Route de l'aéroport, BP 1088, 3018 Sfax, Tunisia

²IRIT, University of Toulouse 1, 2 rue du Doyen Gabriel Marty, 31042 Toulouse Cedex, France

Keywords: Collaborative Process, Flexibility, Version, VCP2M.

Abstract: This paper addresses the collaborative processes flexibility issue, which is an important issue in Business Process (BP) Management. Indeed, the strong competition in which organizations are involved lead them to frequently change and adapt their collaborative processes to face new client requirements or to benefit from new collaboration opportunities. More precisely, this paper proposes to adopt a version-based approach to support the modelling of flexible collaborative processes. First it introduces the VBP2M meta-model (Version of BP meta-model) supporting the modelling of flexible internal (*i.e.*, intra-organizational) processes, and then explains how to extend it to define the VCP2M meta-model (Version of Collaborative Processes meta-model) to design flexible collaborative processes, which correspond to processes crossing the boundaries of companies. A specific case study illustrates the modelling of collaborative process versions as instances of VCP2M.

1 INTRODUCTION

Process flexibility is a major challenge that process-aware information systems have to address before their definitive acceptance and use in companies (Reichert and Weber, 2012). This is mainly due to the more and more dynamic, open and competitive context in which companies operate, which lead them to frequently change both their centralized and collaborative processes. Indeed change support is important for processes running within a single company, but also for Collaborative Processes (CPs) crossing the boundaries of companies. More precisely, a CP is *a set of independent processes, where several partners are involved in one global process and each partner has its own process* (Aalst, 2000).

(Reichert and Weber, 2012) have proposed a taxonomy for process flexibility. This taxonomy serves as a basis for evaluating the ability of systems and models to support both centralized and collaborative process flexibility. More particularly, this taxonomy identifies four types of flexibility: (i) *flexibility by variability*, for representing a process differently, depending on the context of its execution, (ii) *flexibility by adaptation*, for handling occasional situations or exceptions which have not

been necessarily foreseen in process schemas, (iii) *flexibility by evolution*, for handling changes in processes, which require occasional or permanent modifications in their schemas, and finally (iv) *flexibility by looseness*, for handling knowledge intensive processes whose schemas are not known a priori and which correspond to non-repeatable, unpredictable, and emergent processes. Such processes require loose specifications.

Flexibility of processes has been investigated in the context of centralized processes. We distinguish between two different approaches: the variant-based approach and the version-based approach. In the variant-based approach, the main notion is the notion of variant, which is an adjustment at run-time of a process schema (Rosemann and Aalst, 2007), (Hallerbach *et al.*, 2010). This approach mainly deals with flexibility by variability, which is one of the four types of flexibility introduced in (Reichert and Weber, 2012). In the version-based approach, the notion of version has been introduced for capturing process changes over time (Zhao and Liu, 2007), (Dadam and Reichert, 2009), (Chaâbane, 2012). This approach is interesting since it addresses flexibility by evolution as the different significant changes on processes are modelled within process versions, flexibility by variability since it is possible

to model alternative versions, depending on the context, and flexibility by adaptation if adaptation can be defined at design-time.

Process flexibility has been less investigated in the context of CPs. In such a context, flexibility may be related to the availability of involved processes or to the update of schema collaboration. Research efforts about CP flexibility mainly address process availability in the context of dynamic inter-organizational processes. Dynamic inter-organizational processes refer to processes where the different partners involved are not necessarily known at design-time, or can evolve at run-time (Chebbi *et al.*, 2006), (Andonoff *et al.*, 2005). The provided solutions support finding new partners offering requested services, along with negotiation, contracting and service execution in separate or comprehensive frameworks.

The main contributions addressing the update of CP schemas have been done in the SOA context (Boukhedouma *et al.*, 2012a), (Boukhedouma *et al.*, 2012b). These contributions mainly consider chained execution and subcontracting CPs, and they provide some patterns for service adaptation. They deal with CPs flexibility by evolution only keeping the last CP schema, but they do not address flexibility by variability and flexibility by adaptation.

Therefore, this paper addresses CP flexibility issue focusing on the update of CP schemas, taking into account not only chained execution and subcontracting CPs but also loosely coupled CPs (Aalst, 2000). It advocates a version-based approach as versions are known to be a powerful technique to address process flexibility and more precisely flexibility by evolution, flexibility by variability and flexibility by adaptation. In this paper we extend the VBP2M meta-model (Versioned of Business Process Meta-Model) (Chaâbane, 2012), which is a previous contribution for modelling centralized process flexibility using the versioning technique and considering the main perspectives of processes. Moreover, flexible centralized processes modelled as instances of VBP2M can be graphically visualized, simulated and validated (Ben Said *et al.*, 2010) (Ellouze *et al.*, 2013). Unfortunately VBP2M does not address the flexibility of CPs. Thus we propose VCP2M (Version of Collaborative Process Meta Model), an extension of VBP2M supporting the modelling of versions of CPs.

This paper is organized as follows. Section 2 introduces the background of the paper, *i.e.*, the version concept and the VBP2M meta-model for modelling flexible centralized processes. Section 3

presents the necessary concepts for CPs and illustrates these concepts within the *Subsea Pipeline* CP example that will be used throughout the remainder of the paper. Section 4 is dedicated to the presentation of VCP2M, an extension of VBP2M for flexible CPs modelling using versions. More precisely, this section introduces the VCP2M meta-model and illustrates how to model two versions of the *Subsea Pipeline* example by instantiation of this meta-model. Finally, section 5 recaps our contribution and gives some perspectives for future works.

2 BACKGROUND: MODELLING FLEXIBLE PROCESSES

This section presents VBP2M (Version of Business Process Meta-Model), our previous contribution supporting flexible centralized process modelling (Chaâbane, 2012). However, this section foremost introduces the notion of version as it is defined in database and software engineering fields, before presenting VBP2M.

2.1 The Version Concept

A version corresponds to one of the significant states an entity (in the context of the paper, a process, an activity...) may have during its life cycle.

When created, an entity is described by only one version. The definition of every new entity version is done by derivation from a previous one. Such versions are called derived versions. Derived entity versions are linked by a derivation link: they form a derivation hierarchy. Moreover, several versions may be derived from the same previous one. They are called alternative versions (or variants).

We defend that the notion of version subsumes the notion of variant, which rather corresponds to the different ‘alternative’ states (*i.e.*, states representing choices) an entity may have during its life cycle. When considering versions, we also model ‘evolutionary’ states (*i.e.*, states representing the evolution of an entity, independently from any choice) in addition to the ‘alternative’ ones. Thus, with using versions, it is possible to address flexibility by evolution as the different significant changes on processes are modelled within process versions, flexibility by variability since it is possible to model alternative versions, depending on the context, and even flexibility by adaptation if adaptation can be defined at design-time.

2.2 VBP2M

Figure 1 below gives an UML class diagram of VBP2M.

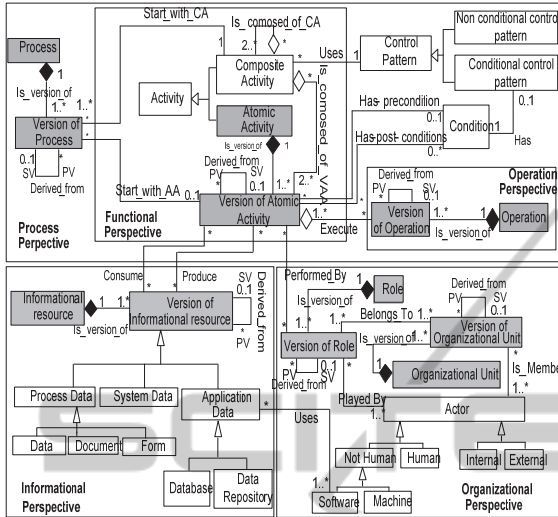


Figure 1: VBP2M for Modelling Flexible Processes using Versions.

This meta-model has the following features: it supports the modelling of the five main perspectives of processes, and, as defended in (Chaâbane *et al.*, 2009), it is simple as it only defines the core (basic) concepts of each of these perspectives. Moreover, it distinguished between versionable classes (*i.e.*, classes for which we handle versions) which are visualized in grey, from normal classes (*i.e.*, classes for which we do not handle versions), which are visualized in white.

The main concepts of VBP2M are *Process*, *Activity*, *Control Pattern*, *Operation*, *Informational Resource* and *Role*. A process performs activities which can be atomic or composite. The first activity of a process is explicitly linked to it via the *start_with_CA* or *start_with_AA* relationships, and the next activities are found via the *is_composed_of_CA* or *is_composed_of_VAA* relationships. A composite activity is composed by other activities, which may be in turn composite or atomic, and which are coordinated by control patterns. Control pattern may be conditional (*e.g.*, if, while, etc.), or not (*e.g.*, sequence, fork, etc.). An atomic activity executes one or several operations (from the operation perspective). It has a start condition (precondition), final conditions (post-conditions) and manipulates (*i.e.*, consumes and/or produces) informational resources (from the informational perspective). An atomic activity is

performed by role, which can be played by actors belonging to organizational unit (from the organizational perspective).

Moreover, a versioning pattern is introduced to make some classes of VBP2M versionable, *i.e.*, able to handle versions. For each of these classes, the versioning pattern permits to model both entity of the versionable concept (*e.g.*, *Process*) and corresponding versions (*e.g.*, *Version of Process*). In addition, two relationships are introduced: (i) the *is_version_of* relationship links a versionable concept with its corresponding versions and, (ii) the *derived_from* relationship describes version derivation hierarchy between versions of a same versionable concept. This later relationship is reflexive and the semantic of the both sides of the relationship are: a version (SV) succeeds another one in the derivation hierarchy and, a version (PV) precedes another one in the derivation hierarchy.

Regarding versionable concepts, VBP2M proposes to model versions for concepts belonging to the five perspectives of processes. The idea is to keep change history for each concept involved in the description of the way business is carried out. Therefore, the versioning pattern is used to model versions of the following concepts: *Process*, *Activity*, *Operation*, *Informational resource*, *Role* and *Organizational Unit*, respectively belonging to the process, functional, operation, informational and organizational perspectives of processes. (Chaâbane, 2012) defends the idea that versioning these concepts is enough to guide companies facing the fast changing environment in which they are involved nowadays.

3 COLLABORATIVE PROCESSES: CONCEPTS AND RUNNING EXAMPLE

First, this section introduces the main concepts of CPs and then illustrates these concepts within the Subsea Pipeline CP, from the TPS Tunisian petroleum company.

3.1 Concepts for CPs

As indicated in the introduction, a CP is a set of independent partner processes interacting together to reach a common goal corresponding to a value-added service. Such a process is not in control of a single partner process, it is enacted by the different partner processes involved in the collaboration.

These partner processes may belong to one or several companies. If they belong to a single company, we are in an intra-organizational context (centralized context), while if they belong to different companies we are in an inter-organizational process context (collaborative process context), crossing the boundaries of each company.

In a CP, the different partner processes playing different roles in the collaboration exchange messages which correspond to synchronisation activities between them (Aalst, 2000). Thus we distinguish the activities supporting message exchange, called public activities, from the ones which do not support message exchange. Public activities define how the different partner processes interact together: some of them correspond to the sending of messages while others correspond to the receiving of messages. Other activities correspond to *private* activities; they are performed to achieve a specific goal within a partner process (they correspond to pieces of work of the partner process).

Consequently, for each partner process, we distinguish between public and private processes (Chebbi et al., 2006): a *public process* gathers the public activities, i.e. the send and receive activities of the considered partner process, while a *private process* gathers the private activities of the considered partner process, which are local to the partner.

In addition, we define, for each partner process, a local view of the CP. This *local view* is composed of the (public and private) processes of the considered partner, and of the public processes of the other partners to which the considered partner directly interacts. Finally, we define the *global view* of the whole CP as the merge of all the public processes of the involved partner processes. The section below will illustrate these notions within the *Subsea Pipeline* example.

3.2 Subsea Pipeline CP Example

The Subsea Pipeline CP from the TPS Tunisian

petroleum company involves two partner processes: TPS, which needs to replace any one of its old damaged subsea pipeline, and SAROST, which will be solicited for this replacement. The CP is the following.

The petroleum company TPS initiates the process and prepares a Tender Specifications (TS) describing the requested pipeline replacement, and submits it to SAROST, a company specialized in subsea pipeline installing and maintenance. Then, SAROST carries out a feasibility study and answers either in a positive way sending back to TPS a quote for the pipeline replacement, or in a negative way explaining why it refuses to do the requested job. When the quote is received and accepted by TPS, then it prepares an order for replacement and sends it back to SAROST which proceeds to the subsea pipeline replacement. To do so, SAROST first specifies the necessary team and equipment and then proceeds to the assembling and the welding of pipes on shore by welders and controller’s inspectors. The next activity is the laying of pipes offshore by the divers. Finally, when the installation is over, tests have to be performed and then an acceptance certificate is prepared and sent to TPS which signs it in turn. Note that the assembling and welding, laying and subsea control have to be repeated until reaching the pipeline length. Figures 2 and 3 below illustrate the different private views of partner processes.

To illustrate the concepts introduced in section 3.1 within the Subsea Pipeline example, we give below the process of TPS in Figure 2, the local view of the collaboration for TPS in Figure 3 and the global view of the collaboration for both TPS and SAROST in Figure 4. The BPMN notation (OMG, 2011) is used to illustrate these concepts.

Section 4 below deals with modelling of versions of collaborative processes using VCP2M, an extension of VBP2M taking into account CP concepts. To illustrate CP version modelling, we extend the previous example considering a second version of this CP.

In this new version, TPS subcontracts the activity

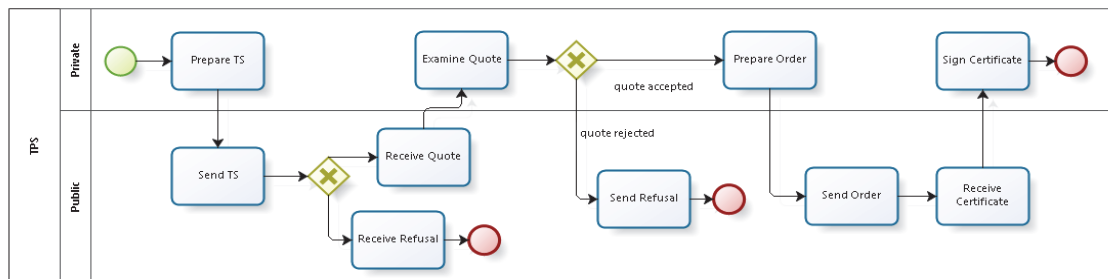


Figure 2: TPS’s Process.

refers to the first version of the SAROST process). VIP1 starts with a composite activity, CA1, which is a sequence involving the first version of TPS' start event, the activities to be performed, and the first version of TPS' end event. Due to lack of space and for clarity reasons, we only show the start event of TPS, V1E1, along with the two first activities of TPS: *Prepare TS* represented as the version V1AA1, and *Send TS*, represented as the version V1AA2. We do not detail the others activities but just indicate that another composite activity is used to indicate that, after Send TS, there is a choice to perform. Regarding VIP2, its starts with a composite activity, CA5, which is a sequence involving the first version of SAROST' start event, the activities to be performed, and the first version of SAROST' end event. For the same reasons, we only focus on SAROST' start event and the *Receive TS* and *Feasibility Study* activities, represented as versions respectively denoted as V1AA4 and V1AA5. The first version of the collaboration, V1C1, involves the two versions of TPS's and SAROST's processes, VIP1 and VIP2 (green relationship), and each of the process plays a specific role in the collaboration (respectively customer and supplier). In addition, the message M1 is exchanged in this collaboration (green relationship); it refers to the V1IR1 informational resource and it is involved in the receive/send relationship (between V1AA2 and V1AA4), represented in blue.

The second version of the collaboration involves (red relationship) a second version of TPS's process (V2P1), the first version of the SAROST's process (VIP2, it means it is the same version of this process involved in the two versions of the collaboration), and the first version of the Coff's process (VIP3). This version is a sequence of P3' start event, three versions of the *Receive Request for TS*, *Prepare TS*, and *Send TS* activities, and P3' end event. Regarding VIP2, it is a sequence of P1' start event, V1AA3 (first version of *Send Request for TS* activity), and V2AA2 (second version of *Send TS* activity, derived from V1AA2). Because of space limitation, we do not detail anymore this process version. In this collaboration, two messages are exchanged (red relationship): V1M1 exchanged between TPS and SAROST as in the first collaboration version, and V1M2, correspond to a message exchanged between the two public activities *Send Request for TS* (V1AA3) from TPS and *Receive Request for TS* (V1AA6) from Coff (represented in blue).

5 CONCLUSIONS

This paper has presented VCP2M, a meta-model addressing CP flexibility issue. VCP2M extends VBP2M, a previous contribution for modelling flexible business processes using versions. More precisely, VCP2M extends VBP2M introducing

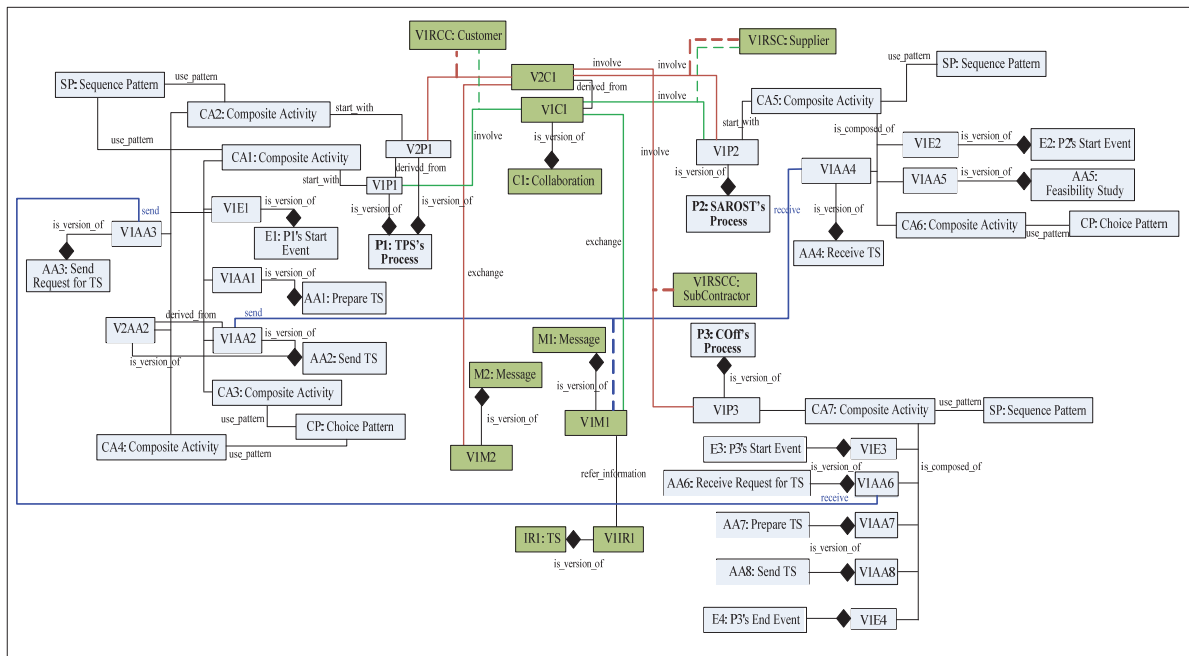


Figure 7: Partial Instantiation of VCP2M.

necessary concepts for modelling versions of collaborations such as participant, message, private and public processes for collaborations. The paper also highlights how VCP2M can be instantiated to model versions of collaboration for the Subsea Pipeline example.

The advantages of VCP2M are the following. First, the notion of version is well-suited to address process flexibility in a collaborative context since it supports the modelling of process evolution, variability and adaptation. Second, VCP2M considers chained, subcontracting and loosely coupled CPs whose control may be centralised or distributed (Breu *et al.*, 2013). Third, VCP2M defines the core concepts of collaborative process modelling (Malekan and Afsarmanesh, 2013).

As future works, we have planned to address dynamic aspects of VCP2M defining operations for both managing collaboration versions, and automatically deducing from public and private activities of processes, the corresponding private and public processes along with their local and global views. In addition we will also introduce the notion of context for collaborations in order to feature them and ease their reuse.

REFERENCES

- Aalst, W. M., 2000. *Loosely coupled inter-organizational workflows: modeling and analyzing workflows crossing organizational boundaries*, *Information & Management*, vol. 37, n°2, pp. 67–75.
- Aalst, W. M., 2004. Business Process Management Demystified: a tutorial on Models, Systems and standard for workflow Management, *Lectures on concurrency and Petri Nets, Advances in Petri-Nets, LNCS 3098, 2004*, pp 1-65.
- Hallerbach, A., Bauer, T., Reichert, M., 2010. *Capturing Variability in Business Process Models: the Provop Approach*. *Software Maintenance*, vol. 22, n°6-7, pp. 519–546.
- Andonoff, E., Bouzguenda, L., Hanachi, C., 2005. Specifying Web Workflow Services for Finding Partners in the Context of Loose Inter-organizational Workflow, *Int. Conference on Business Process Management, Nancy, France*, pp. 120–136.
- Ben Said, I., Chaabane, MA., Andonoff, E., 2010. A Model Driven Engineering Approach for Modelling Versions of Business Processes using BPMN. *Int. Conference on Business Information Systems, Berlin, Germany*, pp. 254–267.
- Boukhedouma, S., Oussalah, M., Alimazighi, Z., Tamzalit, D., 2012. Service based Approach for Adaptability of Workflow Models - the Subcontracting Architecture. *Int. Conference on Enterprise Information Systems, Wroclaw, Poland*, pp. 224–231.
- Boukhedouma, S., Oussalah, M., Alimazighi, Z., Tamzalit, D., 2012. Adaptability of Service Based Workflow Models – the Chained Execution Architecture. *Int. Conference on Business Information Systems, Vilnius, Lithuania*, pp. 96–107.
- Breu, R., Dustdar, S., Eder, J., Huemer, C., Kappel, G., Kopke, J., Weber, B., 2013. Towards Living Inter-Organizational Processes. *Int. Conference on Business Informatics, Vienna, Austria*, pp. 363–366.
- Chebbi, I., Dustdar, S., Tata, S., 2006. *The view-based approach to dynamic inter-organizational workflow cooperation*, *Data & Knowledge Engineering*, vol. 56, n°2, pp. 139–173.
- Chaâbane, MA., Andonoff, E., Bouzguenda, L., Bouaziz, R., 2009. Versions to Address Business Process Flexibility Issue, *Int. Conf. on Advances in Databases and Information Systems, Riga, Latvia*, pp. 2–14.
- Chaâbane, MA., 2012. De la modélisation à la spécification des processus flexibles: une approche basée sur les versions. *PhD thesis dissertation*. University of Toulouse, September 2012.
- Dadam, P., Reichert, M., 2009. The ADEPT Project: a decade of research and development for Robust and Flexible Process Support. *Computer Science – R&D*, vol. 23, pp. 81–97.
- Fdhila, W., Rinderle-Ma, S., Reichert, M., 2012. Change propagation in collaborative processes scenarios. *Int. Conference on Collaborative Computing*, Pittsburgh, Pennsylvania, USA, pp. 452–461.
- Ellouze, F., Chaâbane, M.A., Bouaziz, R., Andonoff, E., 2013. Modelling and Simulation Versions of Business Process using Petri Nets. *Int. Conference on Advances in Databases, Knowledge, and Data Applications, Sevilla, Spain*, pp. 150-158.
- OMG 2011. Business Process Model and Notation (BPMN), Version 2.0. Available at: <http://www.omg.org/spec/BPMN/2.0/>.
- Reichert, M., Weber, B., 2012. *Enabling Flexibility in Process-Aware Information Systems: Challenges, Methods, and Technologies*. Springer (eds).
- Rosemann, M., van der Aalst, W., 2007. A Configurable Reference Modeling Language. *Information Systems*, vol. 32, n°1, pp. 1–23.
- Malekan, S., Afsarmanesh, H., 2013. Overview of Business Process Modeling Languages Supporting Enterprise Collaboration. *Int. Conference on Business Modeling and Software Design, Noordwijkerhout, The Netherlands*, pp. 24-45.
- Zhao X., Liu, C., 2007. Version Management in the Business Change Context, *Int. Conference Business Process Management, Brisbane, Australia*, pp 198–213.