Product-Based Workflow Design for Monitoring of Collaborative Business Processes

Marco Comuzzi and Irene Vanderfeesten

School of Industrial Engineering, Eindhoven University of Technology
P.O. Box 513, 5600MB Eindhoven, The Netherlands
{m.comuzzi,i.t.p.vanderfeesten}@tue.nl

Abstract. Monitoring of cross-organizational processes requires the definition and implementation of monitoring processes that can deliver the right information to the right party in the collaboration. Monitoring processes should account for the temporal and aggregation dependencies among the monitoring information made available by the set of collaborating parties. We solve the problem of designing monitoring processes in collaborative settings using Product-Based Workflow Design (PBWD). We first discuss a methodology to apply PBWD in this context and then propose an architecture to implement the methodology using a service-oriented approach.

Keywords: Monitoring, business process design, cross-organizational processes.

1 Introduction

Continuous monitoring of a business process can be defined as the set of methodology and tools to collect and disseminate relevant information about the process execution to interested stakeholders simultaneously with, or within a reasonably short period after, the occurrence of relevant events in the process [6]. Continuous monitoring has straightforward benefits, such as the opportunity for process providers to detect anomalies in (almost) real time and apply control actions on-the-fly.

Research on (continuous) monitoring in cross-organizational processes has usually taken an *information-centric* perspective, focusing on the definition of monitoring requirements for the collaborating parties and their evolution [6, 11], the design of architectures and tools to capture monitoring information [17], the detection of contract violations, given the available monitoring information [3], or the verification of the compliance of execution logs to a process specification [18].

Although the information-centric view can suffice for intra-organizational process monitoring, where all monitoring information is produced in a given business domain, in cross-organizational settings researchers stress the importance of process- and communication-oriented mechanisms to transmit relevant information to interested parties across the collaborative network [7, 11]. In other

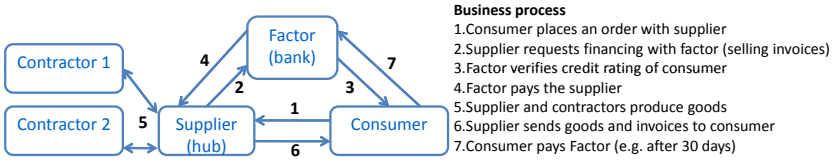

Fig. 1. Factoring example

words, once the monitoring information is captured and made available by the collaborating parties, a process must be built to allow a specific party to retrieve (or be delivered) the monitoring information in the right way. Such monitoring process should account for the temporal and aggregation dependencies among monitoring information.

Let us consider the running example of a business network for factoring in the manufacturing industry (see Fig. 1), constituted by a supplier, a set of contractors (two in our case), a consumer, and a factor. The supplier, in particular, acts as the coordinating hub of the set of contractors, which execute the process required to deliver the product ordered by the consumer. Factoring is a financial transaction whereby a business (supplier) sells its account receivables (invoices) to a third party financial institution (factor) in exchange for immediate payment. Factoring allows consumers to obtain financing at an interest rate, i.e. the one provided by the factor, lower than the one they could obtain directly from the supplier [10]. In this context, continuous monitoring may help reducing the risks associated to the collaboration, e.g. the risk that the supplier will not deliver the goods as promised or the risk that the consumer will not be able to pay. Therefore, continuous monitoring may help to further increase the benefits that the involved parties achieve through the collaboration.

In this scenario, the supplier wants to monitor when the consumer has made the payment to the factor and when the factor has received such payment. If the consumer does not pay, in fact, the supplier should not agree to further transactions in the future involving the same consumer. Note that (i) payment confirmations are not conveyed to the supplier in the process depicted in Fig. 1 and (ii) a temporal dependency exists between monitoring information, i.e. if the supplier checks the factor acknowledgment of the payment without knowing if the consumer has actually sent out the payment, he or she may have an inconsistent view on the process (and may take wrong corrective actions accordingly).

The factor, similarly, wants to monitor the progress and quality of the process on the supplier side to reduce its own risk. The consumer, in fact, may not be satisfied with the goods, e.g. because of late delivery or poor quality, and, as a consequence, may not be willing to pay the factor according to established terms. This information may be delivered either by the supplier, or being reconstructed through more detailed progress information made available by the contractors and aggregated correctly. Again, note that progress information is not conveyed to the factor in the business collaboration depicted in Fig. 1.

The latter example also shows that there could be different alternatives for a party to obtain the required monitoring information and each alternative may be characterized in non-functional terms, e.g. in terms of cost and quality. For instance, progress information made available by contractors may be of higher quality and cost, whereas the supplier may only have limited visibility on the progress of an order once this is outsourced to contractors, and therefore may provide such information at a cheaper price.

In this paper, we propose a methodology to design monitoring processes in collaborative business settings. The methodology considers as input the monitoring information made available by the collaborating parties and builds monitoring processes embedding temporal and aggregation dependencies among monitoring information. Moreover, monitoring information in our methodology can be described also in non-functional terms, e.g. by cost, quality, and availability. Among the set of possible alternatives, the proposed methodology allows the selection of the monitoring process satisfying also the party non-functional requirements, e.g. the minimum cost monitoring process or the highest quality process, given a budget constraint.

The design of the monitoring process for cross-organizational business processes is framed as a PBWD (Product-Based Workflow Design, [16, 20, 22]) problem. PBWD is an analytical method for automatically deriving business process specifications from the set of information products involved in the process and their dependencies. In the monitoring of collaborative processes, information products are represented by the monitoring information made available by the actors involved in the collaboration.

The paper is organised as follows. Related work is discussed in Section 2, while Section 3 introduces the background on PBWD and explains its novel application to the problem of cross-organizational process monitoring. The architecture to integrate the PBWD design of monitoring processes in a service-oriented environment is presented in Section 4, while conclusions are eventually drawn in Section 5.

2 Related Work

Monitoring of cross-organizational business processes has been investigated, from a requirements engineering perspective, in [7] and [11]. In order to achieve a successful collaboration, both papers stress the importance of process- and communication-oriented mechanisms to transmit relevant information to interested parties across the network. Similarly, [5] considers the need to define external information requirements, i.e. information required by a consumer from its providers to correctly monitor and enforce a multiparty contract.

From the design and implementation perspectives, the CrossFlow [8] and CrossWork [9] projects consider architectural support for cross-organizational business processes. Both projects investigate issues such as the design of flexible architecture to support monitoring [9] and the definition of specific monitoring points in electronic contracts [8]. Monitoring, however, is still considered

information-centric, i.e. the dependency among monitoring information products produced by various sources or the aggregation of monitoring information from several parties are not considered.

Research on Web service-based business processes has also extensively investigated the issue of monitoring. Web service-based processes are intrinsically cross-organizational, since each orchestrated Web service can in principle be exposed by a different organization. In this context, we can distinguish between *intrusive* and *non-intrusive* monitoring [2, 13]. The former involves the interleaving of service and monitoring activities at runtime, whereas the latter separates the business from the monitoring logic, since information relevant for monitoring can be captured non-intrusively while a process is executing, e.g. intercepting service operation calls and responses or from the log of the process engine [13]. Cross-organizational settings require non-intrusive monitoring, since it would not be feasible to implement a different instrumentation to satisfy the monitoring requirements of each different party with which an organization has to interact. Furthermore, while the aforementioned approaches only consider the monitoring of performance variables, such as response time and availability, or simple conditions describing the behavior of a service, in this paper we take a business perspective, since PBWD considers business-related information that is deemed relevant by the user of a process.

The innovation of the approach presented in this paper concerns also the aggregation, according to user-specific dependencies, of monitoring information to design a customized monitoring process. Web service-based process monitoring considers also aggregation of monitoring information from multiple sources [2, 13]. However, monitoring information, such as the timestamps of service calls or process variables, are meaningful only at a technical level, and they require further translation before becoming meaningful to and, therefore, relevant for a process user [12]. PBWD considers only informational products meaningful at a business level and, therefore, enables us to design monitoring processes using informational products that do not need translation.

3 Using PBWD for Collaborative Process Monitoring

This section introduces some background on the PBWD approach and shows how PBWD can be used to generate monitoring processes. PBWD is a scientifically grounded method for business process (re)design. The focus of this method is on the design of processes that deliver informational products, the so-called *workflow processes*. The PBWD methodology takes the structure of the *informational product*, which is described in a Product Data Model (PDM), as a starting point to derive a process model. Informational products are, for instance, a decision on an insurance claim, the allocation of a subsidy, or the approval of a loan. Based on the input data provided by the client or retrieved from other systems, the end (informational) product is constructed step-by-step. In each step new information is produced based on the specific data available for the case.

Over recent years, PBWD has shown to be a successful business process (re)design method [15]. For instance, the annual reporting process for mutual funds at a large Dutch bank was successfully redesigned using PBWD. The insights in the informational product, achieved by PBWD, led to a 50% decrease in throughput time [20].

PBWD is particularly well-suited for achieving a process-centric view on monitoring in cross-organizational processes because of two reasons:

- *Clean-sheet approach to process design*: PBWD builds process models directly from the specification of the informational products involved in the process and their dependencies. This approach is a perfect fit for monitoring cross-organizational business processes, since the monitoring process must be built from the monitoring information made available by the collaborating parties. Note that, in highly dynamic collaborations, partner selection is *late-bound*, and collaborating parties are selected dynamically as they become available [9];
- *Cost- and quality-aware process design*: PDMs, i.e. available informational products and their dependencies, can be enriched with information about the cost of producing an information product or its quality for the interested stakeholders. PBWD can then derive various process specifications for the same PDM that differ for their overall costs and quality. The possibility to tune the costs and the quality of the process is an essential feature to derive the most suitable monitoring process for a given stakeholder. When a process is not mission critical, for instance, the monitoring process can be designed by maximizing its quality given a budget constraint, whereas in contexts characterized by severe quality requirements, for instance in highly regulated industries, such as healthcare, monitoring processes can be designed by minimizing the monitoring costs while guaranteeing a given required level of quality.

Fig. 2 shows the steps of the methodology for creating monitoring processes using PBWD for a specific stakeholder in the collaboration. After having analyzed the monitoring requirements of the stakeholder, the application of PBWD to monitoring processes design involves three steps.

First, the Product Data Model (PDM) of the stakeholder is designed. The PDM, which is the starting point for the PBWD method, is similar to the concept of a Bill-of-Materials (BoM) [14] used in manufacturing environments to manage and control production processes. Since (digital) information is more flexible than physical products, however, the PDM contains more complex structures than a BoM, such as re-use of information or alternative paths to produce an information element.

Second, among the set of all paths in the PDM that can lead to the correct production of monitoring information, the path which satisfies the non-functional requirements of the stakeholder is selected. As discussed later, we consider the cost, (data) quality, and availability dimensions to specify non-functional requirements.

In the third step, a process model is generated for the chosen path.

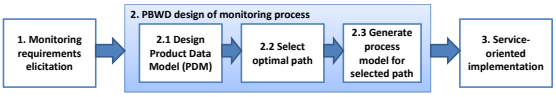

Fig. 2. Methodology for PBWD of monitoring processes

Eventually, the last step in the methodology concerns the implementation of the monitoring process. In this regard, we discuss an architecture for the implementation of the methodology in a service-oriented environment. This discussion is made in Section 4.

From now on, the paper will focus on the application of PBWD to design monitoring processes to satisfy the monitoring requirements of the consumer (stakeholder) in the running example depicted in Fig. 1. In order to decrease its own risk, the consumer is interested in monitoring (i) the status of the payment and (ii) the progress of requests on the supplier side. Monitoring information on the payment can be reconstructed as a combination of information on when the factor sent out the payment and information on when the supplier received the payment. If for instance, the factor has sent out the payment, but this has not been acknowledged by the supplier in a reasonably short period, then the payment may not have been successful.

Monitoring information on the progress of an order can be provided either by the supplier or directly by the contractors. The supplier has only a low-quality view on the order progress, e.g. the supplier may only report that the order has been sent to contractor 1, but he cannot access the details of the internal enactment of contractor 1's process. More detailed monitoring information can be provided directly by contractors.

3.1 The Product Data Model

Fig. 3 summarizes in a PDM the information products available to satisfy the monitoring information requirements of the consumer. A PDM is constituted by information products and operations. Information products in the PDM are depicted by circles, while the operations performed on the input elements to produce the output are represented by hyperarcs. Each operation has zero or more input elements and has exactly one output element, i.e. the information product obtained through the combination of its input elements. A PDM may contain alternative paths to produce a certain information element. Hence, we define a *path* as any sub-graph of the PDM. A *complete* path is a sub-graph of the PDM containing the root element.

In our example, the correct monitoring information for the consumer *(MON)* is obtained combining information on the delivery of the payment *(PAY) and* information on the progress of the request *(PRO)*. The monitoring information *PAY* can be obtained either as information on when the factor has sent out the payment (PF), information on when the supplier has received the payment (PS), *or* a combination thereof. The progress report (PRO) can be obtained

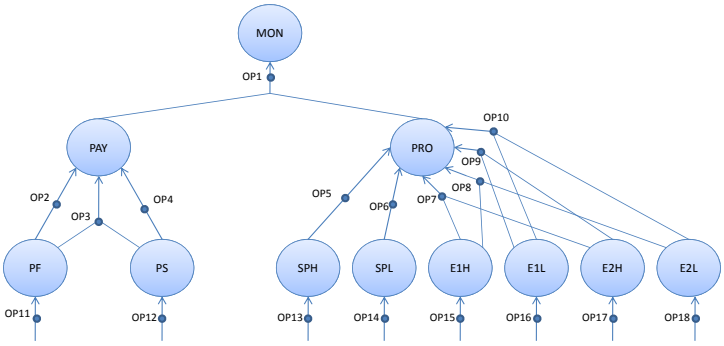

Fig. 3. Factoring monitoring product data model

Table 1. Constraints on example monitoring product data model.

Operation	C	Q	A	Operation	C	Q	A
$Op01$	0	1	1.0	$Op10$	0.5	0.5	1.0
$Op02$	0.6	0.3	1.0	$Op11$	0.2	1	0.9
$Op03$	0.8	0.8	1.0	$Op12$	0.7	1	0.7
$Op04$	0.1	0.7	1.0	$Op13$	1	1	0.7
$Op05$	0.1	0.4	1.0	$Op14$	0.8	1	0.9
$Op06$	0.1	0.2	1.0	$Op15$	0.5	1	0.63
$Op07$	0.8	1	1.0	$Op16$	0.3	1	0.7
$Op08$	0.6	0.7	1.0	$Op17$	0.3	1	0.7
$Op09$	0.6	0.7	1.0	$Op18$	0.1	1	0.8

either from the supplier (SPx) *or* from the contractors (Eyx). Note that x represents the quality of the provided monitoring information, i.e. High or Low ($x \in \{H, L\}$), whereas y identifies the contractor ($y \in \{1, 2\}$). The supplier and contractors 1 and 2 can provide two different types of monitoring information, that is, more or less accurate (see *SPH* and *SPL* or *E1H* and *E1L*).

Operations in the PDM signify the production of an information product. In particular, *leaf* operations (*Op11* to *Op18*) signify the production of monitoring information made by the supplier, contractors, and the factor, while the other operations signify the combination of data elements to produce the parent. In this respect, for instance, *Op07* signifies the combination, made by the consumer, of high quality monitoring information from contractors (*E1H* and *E2H*) to produce the *PRO* data element, whereas *Op03* signifies the combination of payment information from the factor (*PF*) and the supplier (*PS*) to produce the payment monitoring information *PAY*.

We express a path by listing its set of operations. For instance, the path $\{Op11, Op02, Op13, Op05, Op01\}$ is complete, since it leads to the production of the root element *MON*.

Apart from the functional structure, a PDM also contains additional information concerning the non-functional aspects of operations. Three dimensions characterize an operation from the non-functional point of view:

- **Cost (C).** Represents the cost of executing the operation. In other words, for leaf operations, it is the cost sustained by an actor to provide the correspondent monitoring information product, whereas for other operations it is the cost for the monitoring stakeholder (consumer in our case) to combine the leaf information products to obtain the parent product. Costs are summative, i.e. the cost of an information product is the sum of the costs of operations executed to obtain such an information product [1];
- **Quality (Q).** Represents the quality of the information product perceived by the *stakeholder* of the monitoring product model. Quality of monitoring information should be intended as *fit for use*, i.e. the ability of a piece information to satisfy the monitoring information requirements of its user [23]. Fit for use quality of an information product is aggregated using the minimum value of data quality over all operations required to create the information product [1];
- **Availability (A).** Represents the probability that an operation produces its output element(i.e., the output element becomes available after the execution of the operation). Availability is aggregated multiplying the availabilities of all considered operations [1].

Table 1 shows the values assigned to the non-functional dimensions for the PDM of Fig. 3. Note, for instance, that the high quality progress status information, i.e. obtained through operation *Op13* or *Op15*, is more costly than the corresponding low quality information, i.e. the one obtained through *Op14* and *Op16*, respectively. This because a supplier needs to capture more information from the infrastructure executing the process to provide high quality monitoring information. Note also that the cost for the consumer of combining monitoring information produced by other parties is very low. We assume, in fact, that monitoring information is made available digitally and, therefore, its aggregation is almost costless. Concerning availability, note that aggregation operations executed by the consumer, e.g. *Op07* or *Op03*, have always highest availability, whereas the availability of the leaf operations may not be optimal, since it depends on the availability of the infrastructure in which monitoring information is captured by the supplier(s) or the factor.

Apart from the cost, quality, and availability, other properties can be also considered. Ardagna and Pernici [1], for instance, consider execution time and reputation in addition, while Vanderfeesten et al. [21] also consider duration of execution.

3.2 Optimal Path in the PDM

After the design of the PDM, the next step in our methodology (see Fig. 2) is the selection of the optimal path. As explained in the previous section, the PDM may accommodate several alternative paths to produce the end product. The objective of this step is therefore to select a *complete* path that satisfies the requirements of the considered stakeholder, in terms of cost, quality, or availability of the monitoring information, or a combination thereof.

Table 2. Values for the different dimensions (Cost, data quality, availability) for each path in the example product data model.

Path	Operations	Total Cost	Total Quality	Total Availability
1	Op01 Op02 Op11 Op06 Op14	1,7	0,2	0,81
2	Op01 Op02 Op11 Op05 Op13	1,9	0,3	0,72
3	Op01 Op02 Op11 Op08 Op15 Op18	2,0	0,3	0,43
4	Op01 Op02 Op11 Op10 Op16 Op18	1,7	0,3	0,50
5	Op01 Op02 Op11 Op07 Op15 Op17	2,4	0,3	0,38
6	Op01 Op02 Op11 Op09 Op16 Op17	2,0	0,3	0,44
7	Op01 Op03 Op11 Op12 Op06 Op14	2,6	0,2	0,57
8	Op01 Op03 Op11 Op12 Op05 Op13	2,8	0,4	0,50
9	Op01 Op03 Op11 Op12 Op08 Op15 Op18	2,9	0,7	0,30
10	Op01 Op03 Op11 Op12 Op10 Op16 Op18	2,6	0,5	0,35
11	Op01 Op03 Op11 Op12 Op07 Op15 Op17	3,3	0,8	0,26
12	Op01 Op03 Op11 Op12 Op09 Op16 Op17	2,9	0,6	0,31
13	Op01 Op04 Op12 Op06 Op14	1,7	0,2	0,63
14	Op01 Op04 Op12 Op05 Op13	1,9	0,4	0,56
15	Op01 Op04 Op12 Op08 Op15 Op18	2,0	0,7	0,17
16	Op01 Op04 Op12 Op10 Op16 Op18	1,7	0,5	0,39
17	Op01 Op04 Op12 Op07 Op15 Op17	2,4	0,7	0,15
18	Op01 Op04 Op12 Op09 Op16 Op17	2,0	0,6	0,34

In order to select the optimal path, different constraints may be considered. In this paper we use the following scenarios to illustrate our approach: (i) optimal path considering the availability dimension only (A-path), (ii) optimal path minimizing costs given a minimum quality level (C_q-path), and (iii) optimal path maximizing quality given maximum costs and minimum availability level ($Q_{c,a}$-path). We chose these type of constraints to exemplify our methodology. In the general case, however, the selection of the optimal path should be seen as the optimization of a utility function. Stakeholders can define the utility function, for instance, as the weighted sum of partial utility functions on individual dimensions [1].

From the PDM of Fig. 3 and the operation properties of Table 1, we derive 18 *complete* paths for the monitoring process that may all serve the need for monitoring the consumer order fulfillment process. These paths are reported in Table 2. Among the available complete paths, we now discuss the above mentioned optimal ones:

***A*-path.** The path with the highest availability, i.e. the highest probability of delivering the required end product in the form of monitoring information (*MON*), is path 1. It produces the end product *(*MON) by executing operations *Op01, Op02, Op11, Op06, Op14*. The total availability of this plan is 0.81.

C_q-path. A cost optimal path given a minimum quality level is the monitoring path with the lowest cost that satisfies a quality requirement set by the consumer. Suppose the consumer sets the threshold for the quality level to 0.5. Then, paths 9, 10, 11, 12, 15, 16, 17, and 18 are to be considered. The path with the lowest cost is selected from this subset. The cost optimal path given a minimum data quality level therefore is path 16, with quality 0.5 and cost 1.7.

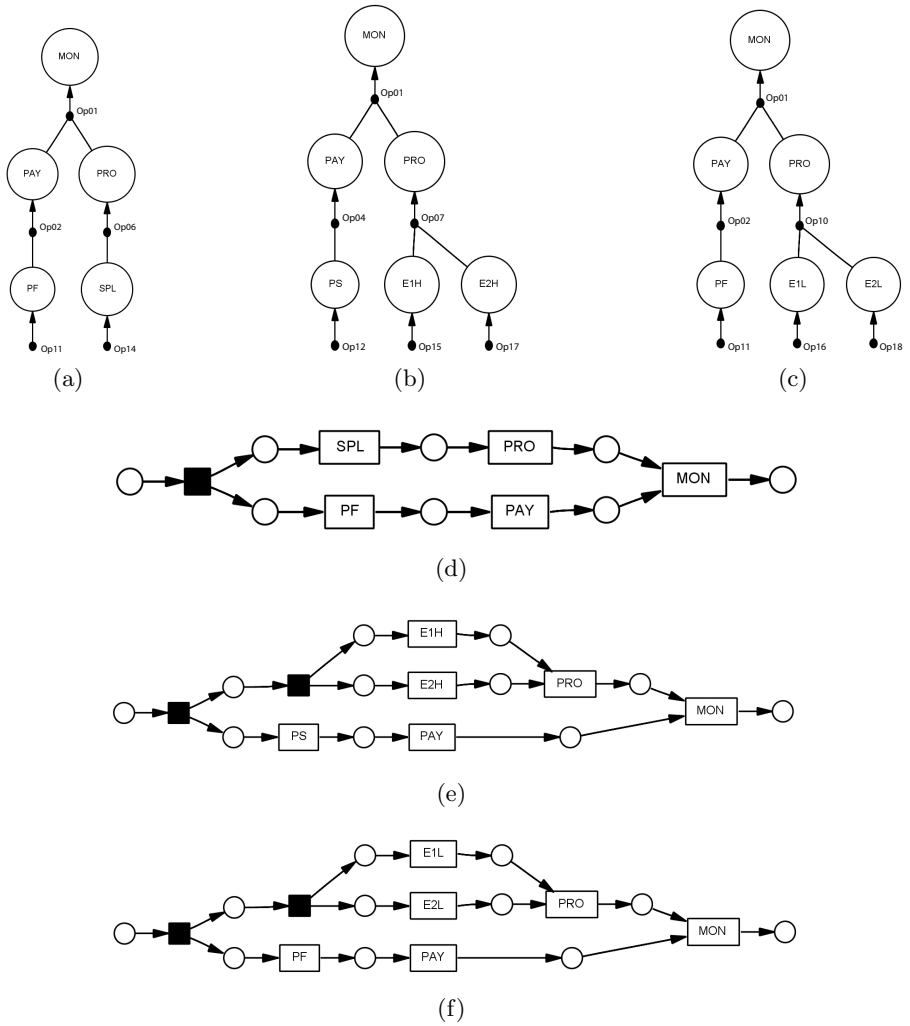

Fig. 4. (a) PDM of the A-path (path 1); (b) PDM of the C_q-path (path 16); (c) PDM of the selected $Q_{c,a}$-path (path 4); (d) Process model for path 1; (e) Process model for path 16; (f) Process model for path 4.

$Q_{c,a}$-path. The third scenario concerns the determination of the quality optimal path given maximum costs and a minimum level of availability. If the consumer has a budget constraint of at most 2.0 and wants to be for at least 50% sure that the monitoring information is delivered, then the highest quality possible is achieved by paths 2 and 4. Since there are two optimal paths and we want to have just one prescriptive process model, it has to be decided which one of the paths is best. In general, if more than one path satisfy the given

constraints, then several approaches can be chosen to select an optimal path. These include: (i) random selection of one path among the identified optimal ones, (ii) selection (by the monitoring stakeholder) of one of the optimal paths by ranking the criteria, or (iii) selection of the path involving the lowest number of operations. As an illustration, we choose the second case and we define the cost as being the second most important criterion after the quality. From paths 2 and 4 we now select the one with the lowest cost (path 4).

3.3 Derivation of Process Models

After the determination of the optimal path in the PDM with respect to a certain criterion, a process model can be generated for the monitoring process (see Fig. 4). Several algorithms have been proposed to transform the PDM into a process model executable, for instance, by a workflow management system [22]. We use the algorithm described in [19] to automatically generate a process model for the optimal path of the PDM. This algorithm is implemented in the ProM framework for process analysis [20]. The resulting process models of our three optimal paths are discussed below. These process models are represented in the Petri Net language. The transitions (squares) represent the operations in the PDM and are named after the output element of the operation, e.g. transition *MON* indicates the operation that produces data element *MON* based on the input elements *PAY* and *PRO*.

***A*-path.** Fig. 4(d) shows the process model for the optimal path with respect to the availability. There are two parallel branches in the process model that can be executed concurrently: (i) a branch in which first the element *PF* is determined followed by the element *PAY*, and (ii) a branch in which *SPH* is determined followed by *PRO*. Once both elements *PAY* and *PRO* are determined, the final element *MON* can be produced. Note that the black activity at the left hand side of the process model is a 'silent' activity, i.e. it is added only for routing purposes but does not process information. Using this process, the consumer retrieves monitoring information on the progress of its request only from the supplier and information on the payment only from the factor.

C_q-path. Fig. 4(e) depicts the process model for the optimal path C_q. Again there are two parallel branches in the process model, both providing input to produce *MON*. On the one hand, *PAY* is obtained using *PS* first. On the other hand, *PRO* is determined by *E1H* and *E2H*, which can be determined in parallel as well.

$Q_{c,a}$-path. Fig. 4(f) shows the process model for the optimal path $Q_{c,a}$. It is similar in structure to the process model of Fig. 4(e), but it uses elements *PF*, *E1L*, and *E2L* in place of *PS*, *E1H*, and *E2H*, respectively.

4 Implementing the Methodology

In this section we discuss the last step of the methodology proposed in this paper (see Fig. 2), i.e. its implementation in a service-oriented environment.

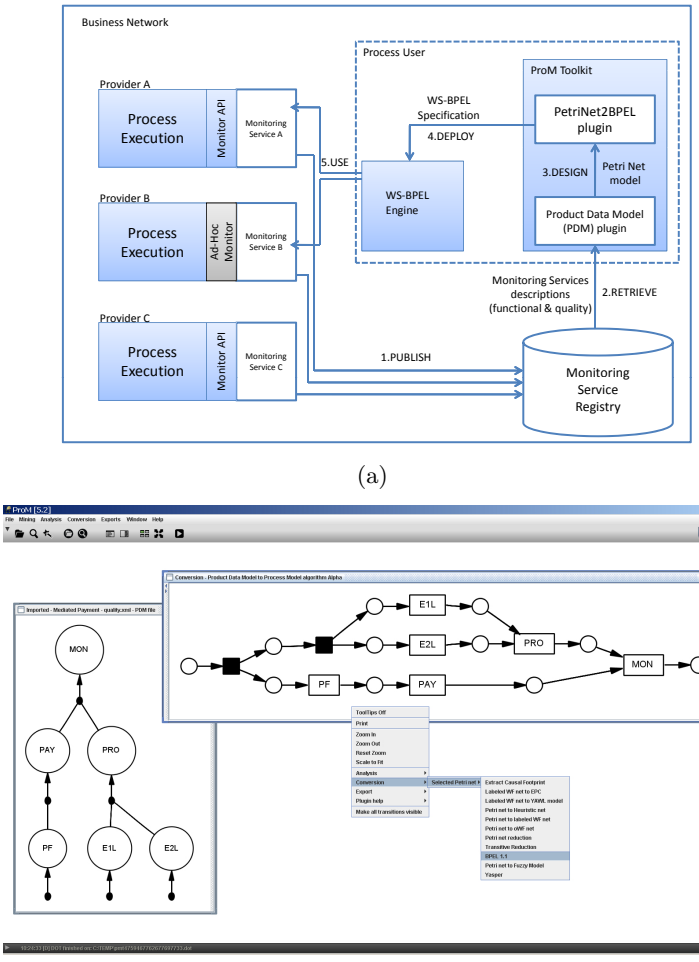

Fig. 5. (a) PBWD-based monitoring process creation architecture; (b) The ProM toolkit, showing a PDM, the relative process model, and the WS-BPEL export functionality

We take a service-oriented approach to the definition of the product data and the implementation of monitoring processes (see the architecture in Fig. 5(a)). The combination of PBWD and service orientation for design and execution of monitoring processes, respectively, can address the need for structural and operational dynamicity in business networks [9]. Concerning network structure, the actors in the network can be dynamically replaced. Suppliers of spare parts in an automotive industrial district, for instance, can be dynamically substituted by the car manufacturer as new suppliers providing more convenient or higher quality options become available. Concerning network operations, the network

business processes can be reconfigured on-the-fly as new business opportunities arise. Insurance companies, for instance, may decide to outsource part of their claim management process only on a temporary basis, e.g. to manage an abnormal amount of claims resulting from a possible fraud. Both scenarios require the dynamic set-up or update of monitoring processes as the network dynamically evolves. This is addressed by our methodology at design time, by adopting PBWD, and at runtime, by using a service oriented approach, where services providing monitoring information can be dynamically orchestrated in the monitoring processes obtained through PBWD.

The elementary monitoring information products are the monitoring information that can be made available by actors in the business network to other actors for monitoring purposes. e.g. the progress information made available by the supplier or the factor in our running example. In the information system or process engine executing a process to be monitored, monitoring information can be captured from various sources and through different mechanisms, such as (i) native APIs of the actor's ERP system, e.g. SAP monitoring architecture, workflow or BPEL engine and (ii) ad-hoc instrumentation, e.g. through the development of event captors or other online process inspection techniques [2].

Irrespectively of how monitoring information is captured, the process provider can make such information available to users by exposing a Web service [6] implementing, for instance, a different operation for each leaf element in the monitoring product data model. The cost, quality, and availability of the monitoring information (see Table 1) can be then specified in a policy document [4], that can be attached to such service before its publication.

In a service-oriented architecture, process providers publish their monitoring services in a service registry (STEP 1 in Fig. 5(a)) and process users browse the registry to get service descriptions and build their monitoring PDM (STEP 2). Note that, in Fig. 5(a), process *provider* and *user* should be intended as roles, since an actor in the business network can act at the same time as a provider of processes and a user of processes contributed by other actors.

The architecture depicted in Fig. 5(a) supports also the creation of an executable monitoring process. Specifically, a process user in the business network interested in building a monitoring process retrieves the required monitoring service description from the registry. Service descriptions are then used to build a monitoring product data model using the ProM toolkit (STEP 3). Currently, as discussed in the previous sections, the algorithms for obtaining monitoring process models (STEP 3), expressed as Petri nets and satisfying given quality constraints, have been implemented as plugins of the ProM toolkit. ProM also provides a plugin for the translation of models from Petri nets to (abstract) WS-BPEL specifications (see Fig. 5(b)). The abstract WS-BPEL specification is bound by the monitoring stakeholder to the required monitoring services in the registry, in order to make it executable, and deployed in a process engine (STEP 4). When in execution, a monitoring process will use the monitoring services originally published by actors in the business network, according to the

aggregation and dependency constraints specified in the monitoring PDM and the derived monitoring process model (STEP 5).

In respect of the scenario depicted in Fig. 5(a), most of the steps of our methodology, such as the definition and retrieval of monitoring services from the registry, the binding of the WS-BPEL specification obtained through ProM to actual services in the registry, and the deployment of the WS-BPEL specification in the process engine, are still executed manually. Future work will concern the implementation of an integrated approach in which the aforementioned activities could be fully automated.

5 Conclusions

This paper presents an innovative application of PBWD, that is, the product-based design of monitoring processes in collaborative business networks. The innovation brought about by this paper is twofold. On the one hand, given a collaborative scenario and available monitoring information, we propose a methodology to design from scratch a monitoring process that matches the process user's requirements, in terms of cost, (data) quality, and availability of monitoring information. On the other hand, we discussed an architecture for implementing the proposed methodology.

A first area of improvement for this work concerns the implementation of the architecture reported in Fig. 5(a) and, specifically, the connections between the implemented ProM's plugins and the service-oriented process execution environment. Future work will also concern the extension and refinement of the product-based generation of monitoring processes. In particular, we plan to consider additional non-functional dimensions, such as reputation, and more complex utility functions for capturing the monitoring stakeholder requirements. Constraints describing monitoring products can also become dependent on the type of monitoring information and the type of stakeholder requiring it. Hence, we want to investigate the issue of provider and user profiling for automatically designing more customized monitoring processes. Also, while this paper considers the monitoring requirements for a stakeholder at the process level, we are planning to consider also instance-level monitoring requirements, i.e., monitoring requirements that can change with every different instance involving a given stakeholder. Finally, from the modeling perspective, we want to investigate the opportunity of specifying monitoring processes as choreographies, e.g. in BPMN 2.0, for capturing more complex dependencies among monitoring information.

References

1. Ardagna, D. and Pernici, B. Adaptive Service Compostion in Flexible Processes. *IEEE Transactions on Software Engineering*, 33(6):369–384, 2007.
2. Baresi, L., Guinea, S., Nano, O., and Spanoudakis, G. Comprehensive monitoring of BPEL processes. *IEEE Internet Computing*, 14(3):50–57, 2010.

3. Baresi, L., Guinea, S., Pistore, M., and Trainotti, M. Dynamo + Astro: An integrated approach for BPEL monitoring. In *Proc. ICWS*, 2009.
4. Cappiello, C., Comuzzi, M. and Plebani, P. On automated generation of web service level agreements. In *CAiSE*, pages 264–278, 2007.
5. Chiu, D. K. W., Karlapalem, K., Li, Q., and Kafeza, E. Workflow view based E-contracts in a cross-organizational E-services environment. *Distrib. Parallel. Dat.*, 12:193–216, 2002.
6. Comuzzi, M., Vonk, J., and Grefen, P. Continuous monitoring in evolving business networks. In *Proc. CoopIS*, pages 168–185, 2010.
7. Daneva, M. and Wieringa, R. A requirements engineering framework for cross-organizational erp systems. *Requirements Engineering*, 11:194–204, 2006.
8. Grefen, P., Aberer, K., Hoffner, Y. and Ludwig, H. CrossFlow: cross-organizational workflow management in dynamic virtual enterprises. *Comput. Syst. Sci. & Eng.*, 5:277–290, 2000.
9. Grefen, P., Eshuis, R., Mehandijev, N., Kouvas, G., and Weichart, G. Internet-based support for process-oriented instant virtual enterpises. *IEEE Internet Comput.*, pages 30–38, 2009.
10. Kappler, L. The role of factoring for financing small and medium enterprises. *Journal of Banking and Finance*, 30:3111–3130, 2006.
11. Kartseva, V., Hulstijn, J., Gordijn, J. and Tan, Y.-H. Control patterns in a healthcare network. *European Journal of Information Systems*, 19:320–343, 2010.
12. Kotsokalis, C. and Winkler, U. Translation of service level agreements: A generic problem definition. In *ICSOC/ServiceWave Workshops*, pages 248–257, 2009.
13. Moser, O., Rosenberg, F., and Dustdar, S. Non-intrusive monitoring and service adaptation for ws-bpel. In *Proc. WWW*, 2008.
14. Orlicky, J. Structuring the Bill of Materials for MRP. *Production and Inventory Management*, pages 19–42, Dec 1972.
15. Reijers, H. *Design and Control of Workflow Processes: Business Process Management for the Service Industry*, volume LNCS 2617. Springer, 2003.
16. Reijers, H., Limam, S. and van der Aalst, W. Product-Based Workflow Design. *Jorunal of Management Information Systems*, 20(1):229–262, 2003.
17. Robinson, W. A requirements monitoring framework for enterprise systems. *Requirements Engineering*, 11:17–41, 2006.
18. Rozinat, A. and van der Aalst, W. Conformance checking of processes based on monitoring real behavior. *Information Systems*, 33(1):64–95, 2008.
19. van der Aalst, W. On the Automatic Generation of Workflow Processes based on Product Structures. *Computers in Industry*, 39:97–111, 1999.
20. Vanderfeesten, I. *Product-Based Design and Support of Workflow Processes*. PhD thesis, Eindhoven University of Technology, Eindhoven, the Netherlands, 2009.
21. Vanderfeesten, I., Reijers, H., and van der Aalst, W. Product-Based Workflow Support. *Information Systems*, 2011. (to appear).
22. Vanderfeesten, I., Reijers, H., van der Aalst, W., and Vogelaar, J. Automatic Support for Product Based Workflow Design: Generation of Process Models from a Product Data Model. In *OTM 2010 Workshops*, pages 665–674, 2010.
23. Wang, R. A product perspective on total data quality management. *Communications of the ACM*, 41(2):58–65, 1998.