A Framework for the Collaborative Specification of Semantically Annotated Business Processes

Chiara Di Francescomarino†, Chiara Ghidini, Marco Rospocher, Luciano Serafini, Paolo Tonella†

FBK–irst, Via Sommarive 18 Povo, I-38123, Trento, Italy

Abstract
Semantic annotations are a way to provide a precise meaning to business process elements, which supports reasoning on properties and constraints. Among the obstacles preventing widespread adoption of semantic annotations are the technical skills required to manage the formalization of the semantics and the difficulty of reconciling the different view points of different analysts working on the same business process.

In this paper, we support business analysts in the collaborative annotation of business processes by means of a tool inspired to the Wiki pages model. Using this tool, analysts can concurrently work on process elements, ontology concepts, process annotation or constraint specification. The underlying formalism is not exposed in the Wiki pages, where natural language templates are used.

1. Introduction
Semantic Business Process Management [1, 2] aims at improving the level of automation in the specification, implementation, execution, and monitoring of business processes by enriching process artifacts with explicit semantic annotations, and by extending business process management tools with the most significant results from the area of Semantic Web. As described in [3], semantic annotation of business processes allows analysts to give a precise meaning to the process elements they are modelling, thus improving, among others: the reuse of parts of process models when creating new models; the detection of cross-process relations; the management of change; and providing a structured basis for knowledge transfer and for enabling automated reasoning on the process and its properties. However, semantic annotation involves skills and competences that go beyond the typical background of a business analyst, such as ontology construction and extension, formulation of queries and constraints in descriptive logics. Moreover, the semantics of a business process is almost never unique. Different view points on the process elements and properties bring in different concepts and

†Correspondence to: C. Di Francescomarino, dfmchiara@fbk.eu, FBK–irst, Via Sommarive 18 Povo, I-38123, Trento, Italy
†E-mail: dfmchiara,ghidini,rospocher,serafini,tonella@fbk.eu
constraints. For example, for a seller company relevant concepts are sale quantity or placing stocks, while for a delivery company important notions are delivery or loading and unloading.

Integrating and reconciling different views of the same process is not an easy task. Available tools for process construction (e.g., Hyperwave, InterPROM) provide functionalities for collaborative process definition. However, the problem becomes harder when process elements are given a precise semantics by means of an ontology. In fact, incremental ontology creation and extension is expected to be carried out in parallel with the incremental definition of the process. Available tools do not provide any explicit support to the complex activity of collaborative ontology creation/extension, neither they support the related activities of collaborative semantic annotation of process elements and constraint specification.

In this paper, we present a framework for the collaborative specification of semantically annotated business processes. The framework takes advantage of a shared workspace to store the main artefacts that are manipulated collaboratively, i.e., (1) process; (2) ontology; and, (3) constraints. Analysts work on these artefacts concurrently, without any notion of ownership (so they can modify artefacts initially created by others). Conflicts are managed through mutually exclusive lock, and disputes causing instabilities are resolved through discussion forums. The complexity of the underlying formal ontology and descriptive logics formulas is completely hidden to the user.

We have implemented the proposed framework in a tool, called BP-MoKi, which realizes the principles summarized above using a set of Wiki pages (based on Semantic MediaWiki). We have decided to develop our tool on top of a (semantic) wiki due to the established popularity of wikis as state of the art web-based collaborative tools, and the possibility to use the semantic annotations enriching the informal descriptions provided by the users, a key feature of semantic wikis, to build the structured artefacts underlying our framework. BP-MoKi offers graphical facilities for process definition and for ontology visualization and manipulation. Graphical elements have a textual counterpart, where additional properties (description, annotations, etc.) can be specified. The tool ensures automatic alignment of graphical and textual views. It also performs constraint checking upon any operation terminating with a commit of the workspace artefacts.

We have conducted a case study in which three analysts, each a domain expert from a different company, have integrated the processes of their companies into a unique process, in accordance with the company partnership agreement. The case study is described as a sequence of snapshots, which highlight the interactions among the work performed by different analysts on different parts of the process. Such interactions were successfully and easily managed, thanks to the collaborative facilities offered by BP-MoKi.

The paper is organized as follows: Section 2 presents a motivating example, Section 3 describes the proposed framework in general, while Section 4 provides details about the constraint definition and formalization. Section 5 presents one possible realization of the proposed framework, our tool BP-MoKi. Section 6 describes the application of the framework to the motivating example. Sections 7, 8 present related works and conclusions respectively.

2. A motivating example

In this section we present a scenario motivating our approach and its usefulness in supporting the collaboration activity. The main actors of the scenario are three private companies:
**RS**, a retail seller company, selling products to people; 
**WS**, a wholesale seller company, providing products to retail sellers and 
**D**, a delivery company, carrying products from the provider to buyers.

Though occasionally exploiting services provided by the other two actors, the three companies are independent from each other; each of them has its own business and its own business processes.

The process describing the inventory management of the RS company is depicted in Figure 14*. After the inventory has been checked and a computation of the items sold in the last month is performed with the aim of evaluating how many new items to buy, the order is prepared and the supplier selected. Since the supplier (and therefore his/her internal process) is not fixed, the purchase sub-process cannot be explicitly modelled. In order to be adaptable to the current supplier’s requirements, the RS control flow is not completely specified. For example, no order is specified between the money transfer and the seller’s order confirmation, as well as between the supplier’s order acceptance and the money transfer. Once the stock has been received, it can be tracked and the products can be placed.

Figure 15 depicts the selling process of the WS company*. Whenever a new purchase order is received, product availability is checked. Once the resources have been assessed and an estimated delivery date is received from a delivery service, the company can send an estimated date to the customer and, if necessary, start a new stock production. However, it is necessary to wait for the customer confirmation and for the production completion before initiating product delivery. To this purpose a delivery company is selected and assigned the transportation. Also in this case, due to the variability of the transportation company and, therefore, of its process, the WS delivery sub-process cannot be completely specified: the only known activities in the sub-process are the delivery order and the payment. If no payment is received from the buyer, a notification is sent every 5 days and an extra payment is charged to the customer. After the payment has been received, the purchase order is closed.

Finally, the delivery process of the D company is described in Figure 16*. Whenever a delivery request is received, the D company realizes an assessment of its resources, both in terms of staff and transportation means. The result of this evaluation can be either declining the provisioning of service (if no means or men are available) or the notification of a delivery estimated date to the customer. In the latter case, if the date is accepted, the payment is requested. If the payment is not received within 30 days, the service is cancelled; otherwise, after the payment receipt, a transporter is assigned the delivery of the products. He has therefore to move to the provider place, load the products, move to the destination place and unload the products. He has also to track his job and report it to the company.

In order to improve their businesses, the three companies decide to join. This choice impacts the way of working of each of the three companies. It implies not only the “merging” of the three processes, but also a re-arrangement in the company organization, management, and processes. The main changes will affect the parts of the processes involving the communication among actors. However, ripple effects and policy changes will also involve other parts of the processes. Hence, due to the heterogeneity of viewpoints and partners’ requirements, to the different domain features and to the variety of factors to take into account (e.g. control flow constraints, actor communication mechanisms), the integration results to be a very difficult task. The purpose of our work is to support designers in overcoming

---

*Note that the annotations reported in the diagram are not included in the original process. They are added later, when enriching the process with semantic information. The original models, together with the other artefacts described later in the case study, are available at http://selab.fbk.eu/difrancescomarino/SemanticBPM/*
difficulties by providing a framework and a tool for collaboratively integrating all these different and heterogeneous viewpoints.

3. Framework

We propose a framework for the collaborative specification of semantically annotated business processes based on the notion of shared workspace illustrated in Figure 1. This workspace makes the artefacts necessary for this activity visible to all actors who are contributing to the definition of the annotated process. These artefacts are then collaboratively developed by the actors, according to their role. Typical actors working concurrently on the business process specification are the analysts who are expert of different aspects of the business. For instance, during the integration of the three processes described in Section 2 the organization may ask the delivery expert, the whole-sale expert and the retail expert to collaboratively define the integrated process starting from the pre-existing single ones. These different analysts may specify different parts of the process. They can also modify the parts defined by others, so as to make them consistent with their own modifications. The usage of a collaborative workspace aims at supporting the integration of different perspectives.

The artefacts manipulated in the collaborative workspace are the process itself, a domain ontology used to annotate the process elements, and a set of constraints which makes use of both the business process and the domain ontology. The collaborative workspace includes also a “read only” part composed of a BPMN (Business Process Modelling Notation) ontology and some BPMN axioms, which are used by the collaborative framework to give a precise semantics to the process elements.

Analysts working collaboratively on a given business process carry out four main activities: (1) incremental process construction; (2) ontology definition or extension; (3) constraints specification; and (4) addition of semantic annotations. These four activities are illustrated in detail in the final part of this section. What is important to note here is that there is no precedence relationship or prescribed workflow in the execution of these four activities. They can be executed concurrently, in any order, and the collaborative workspace supports multiple analysts working on different artefacts and carrying out multiple activities at the same time. To realize such a concurrent working environment we need to address two main problems: (i) concurrent modification of the same artefact; and (ii) instabilities deriving from incompatible modifications that are repeatedly done and undone. For the first problem, we adopt a solution, widely used to address concurrent database accesses, which is based on the acquisition of a lock (i.e., a mutually exclusive access). When an analyst starts working on an artefact, she acquires the lock on it and the workspace provides such an artefact to the other analysts in “read only” mode. When the editing is finished, the analyst commits the changes. This produces an update of the workspace, which triggers an automated verification of the constraints on the new version of the workspace. It also results in the release of the lock on the changed artefact, which becomes available to the other analysts. Incompatible changes are instead managed by resorting to solutions widely used in collaborative content management systems (e.g., Wikipedia). Once a problem on some artefact modification is detected (with the help of automated change monitoring and analysis tools), the first attempt to solve the conflicts consists of initiating a discussion forum, which involves the contributors who made the conflicting changes, as well as experts about the object of the dispute. The project or team leader is in charge of starting such a forum. If no consensus is achieved by the discussants participating
in the forum, the solutions used in collaborative content editing involve voting (with different voters having different weights) and/or authoritative decisions by an expert or by the project leader.

Another key characteristic of the collaborative workspace is that the four components illustrated in Figure 1 together with their inter-connections are theoretically grounded in a formal representation of semantically annotated business processes illustrated in [4]. In that work, we have defined and implemented these components as parts of a modular Business Processes Knowledge Base (BPKB), expressed using the semantic web language OWL, based on Description Logics [5]. The choice of abstracting away the formal representation of the artefacts in the collaborative workspace was made to provide the actors with friendly ways to access and modify them. This is a fundamental requirement especially for artefacts as the domain ontology and the constraints, which require specialized competences and skills, if manipulated in their formal representation: those are not typical skills of business process analysts. Nevertheless, grounding the informal representation in a formal representation is an important asset to give a precise semantics to the annotated business process diagrams and to make use of formal representation and reasoning techniques to support the business analysts in their design and management tasks. As an example, these techniques can be used to support the creation of valid diagrams by formally specifying and automatically verifying sets of constraints on Business Process Diagrams that involve both knowledge about the domain and the process structure, as we will illustrate in Section 4. An in depth presentation of the formal representation is out of the scope of this paper. Even so, it is important to note here that an alignment between the informal representation provided in the workspace and the underlying formal representation is maintained by the tool implementing the workspace, as described in Section 5.

3.1. Process Construction

The main purpose of the collaborative workspace is to obtain annotated Business Process Diagrams (BPDs) specified using BPMN†. The collaborative framework provides instruments for the graphical specification of BPDs. In addition to the graphical representation, each element of the process is also represented by means of a textual template. This is used to record additional properties of the element, such as its description, its annotations, or the logs of a discussion carried out to resolve a conflict on the element itself. The graphical and textual instruments or the specification of BPDs can be used collaboratively, in a wiki style interaction, by the different analysts to get an overview of the produced BPDs, to insert new process knowledge or to revise / refine already present process knowledge as described in Section 5.

At the underlying formal level, each BPD element is considered as an instantiation of an element specified in the “read only” BPMN ontology‡.

Consider for instance the small portion of a BPD depicted in Figure 2. The data-based XOR gateway in this process is considered by the system as an instantiation (say, g) of the class data_based_exclusive_gateway in the BPMN ontology. Similarly, the sequence flow object and the task in the diagram are considered instantiations (say, s and t) of the classes sequence_flow and task in the

---

†OMG - BPMN v1.1 - http://www.omg.org/spec/BPMN/1.1/PDF
‡We assume availability of a BPMN ontology such as the one described in our previous work [4] and available online at: http://dkm.fbk.eu/index.php/BPMN_Ontology
BPMN ontology. Thus, at the formal level the three elements in Figure 2 are represented by the three assertions:

\[
\begin{align*}
\text{data-based-exclusive-gateway}(g) \\
\text{sequence-flow}(s) \\
\text{task}(t)
\end{align*}
\]

Similarly, the structure of the process (i.e. the connection between different elements) is represented by means of relations between instantiations of BPD elements. For instance the fact that the sequence flow \(s\) originates from the gateway \(g\) and terminates in task \(t\) is represented by the statements

\[
\begin{align*}
\text{has-sequence-flow-source-ref}(s, g) \\
\text{has-sequence-flow-target-ref}(s, t)
\end{align*}
\]

As we stated above, this instantiation is automatic and transparent to the analysts’ activity, but it is necessary to give a precise semantics to the process elements. Thus whenever the business process design tool is used to draw a process element, an instantiation relationship, and all the corresponding formal assertions are automatically generated by the collaborative workspace, depending on the type of process element being created.
3.2. Ontology Construction

The domain ontology is necessary to give a precise semantics to the terms used to annotate business processes. This ontology is typically constructed together with the process and the collaborative workspace supports incremental process and ontology construction. In fact, even if existing domain ontologies can be used to annotate processes, they often need to be adapted to the specific needs of the process being designed. Ontology construction is also a collaborative activity. For instance the retail expert may specify the portion of the process concerning her expertise and at the same time she may introduce the concepts needed for the retail aspects of the business process among which a concept to store stock information. Other analysts may refine or extend the ontology later, with more concepts, as soon as the process grows.

The domain ontology is formally represented as a (set of) OWL ontology(es) that describes a specific business domain. The domain ontology can be an already existing business domain ontology (e.g. RosettaNet or similar standard business ontologies), a customization of an existing ontology, or an artefact developed on purpose. Top level ontologies such as DOLCE [6] can be included as standard components of the domain ontology and used either as the starting point for the domain ontology construction or to provide typical annotation patterns to the BPD objects, as we describe in Section 4. Again, analysts interact with this artefact by using graphical and natural language templates which do not expose the formal ontology structure explicitly.

3.3. Constraints definition

The ability to define constraints is a crucial step in supporting the creation of valid diagrams, which not only comply with the basic requirements of the process semantics, but also satisfy properties that take into account the domain specific semantics of the labels of the different process elements. For instance, an important requirement for a valid retail seller process should be the fact that the activity of storing the stock information has to occur after the stock receipt. The collaborative workspace makes use of constraints as the one just described to ensure that important semantic properties of process elements are satisfied.

We distinguish between two different kinds of constraints: merging axioms and structural constraints (see [7]). Merging axioms are specified to constrain the use of annotations. They indicate which process elements can (or cannot) be annotated by which domain concepts. Structural constraints encode properties that apply to the particular business process under construction. Such constraints
include precedence relationships (e.g., a given type of activity must precede/follow some other activity). Other examples are cases where the semantics of BPMN is further constrained (e.g., inclusive gateways are not allowed).

To support the analysts in this activity, it is possible to define a set of predefined templates in which (constrained) natural language is used to express the constraints. These templates are then automatically translated into their formal description, having the form of DL axioms. Given the importance of constraints to specify properties over annotated business processes we describe them further by means of examples in Section 4.

3.4. Process Semantic Annotation

Analysts are required to annotate process elements with concepts taken from a domain ontology. The collaborative framework allows the analysts to associate elements of the domain ontology with the BPMN elements that are supposed to refer to that particular concept. This is done in the natural language based template which describes the element to be annotated. To simplify the task, the collaborative framework provides also suggestions, and presents the analysts with a list of admissible annotations, based on the constraints specified upon it.

At the underlying formal level, an annotated process element becomes also an instance of the domain ontology concept it is annotated with. Thus if task “Task 1” in Figure 2 is annotated with the concept to\_store\_stock\_information contained in the domain ontology, then task $t$ is not only an instantiation of the class task as described in Section 3.1 but also an instantiation of the class to\_store\_stock\_information contained in the domain ontology, represented by the assertion to\_store\_stock\_information($t$).

4. Expressing Requirement Constraints over Annotated Business Processes

To ensure that important requirements are satisfied, we make use of constraints. We distinguish between merging axioms and structural constraints. Notationally, in this section we use the prefix $BDO$ (Business Domain Ontology) to indicate terms and expressions contained in the domain ontology, while we use $BPMNO$ (BPMN Ontology) to indicate terms and expressions contained in the read only BPMN ontology of Figure 1.

4.1. Merging axioms

These constraints are expressions used to state the correspondence between BPMN elements and elements of the domain ontology. Intuitively, they define criteria for correct / incorrect semantic annotations, or in other terms, from a modelling perspective, for stating which concept of the domain can/cannot be represented as a given BPMN process element (and vice-versa). Examples of these criteria are:

\[
\text{A BPMN activity can be annotated only with actions of the domain ontology (and not, e.g., with objects);} \tag{1}
\]

\[
\text{A BPMN data-object cannot be annotated with actions or events of the domain ontology (but, e.g., with objects);} \tag{2}
\]
A BPMN event can represent only events of the domain ontology (and not, e.g., objects); (3)
The purchasing activity is a complex activity that can be represented only as a sub-process; (4)
The delivering activity is a complex activity that can be used to annotate only a sub-process; (5)
The stock delivering activity is a complex activity that can be represented only as a sub-process. (6)

Expressions (1)–(3) describe “domain independent” criteria as they relate elements of BPMN, such as data-objects, activities or sub-processes to elements of a general ontology, such as the top-level DOLCE ontology [6]. These kinds of constraints can be thought of as “default” criteria for correct / incorrect semantic annotations which hold in a variety of different domains. They can be provided as a “default” component of the domain ontology in the workspace. The advantage of having these criteria already included in the BPKB is that in many situations it might be the case that the analysts, which are usually very focused on their application domain, forget to add them explicitly while they may tend to add more domain-specific constraints; note however that these “default” criteria could still be modified by the analysts to reflect the actual annotation criteria for the specific domain at hand.

The remaining expressions (4)–(6) are instead “domain specific” criteria as they constrain the usage of domain specific terms, such as the complex actions to_purchase, to_deliver and to_deliver_stock to annotate only certain BPMN elements, that is, sub-processes, and not others, e.g., tasks, which are instead suitable to be annotated with simple (atomic) activities. Constraint (4) refers to the process of the RS company, constraint (5) refers to the process of the WS company, while constraint (6) refers to the process of the D company.

To allow the business analysts to specify these kinds of positive and negative annotation criteria, in [4] we have introduced four different constructs, equivalent to the corresponding forms in the modelling perspective:

- **annotatable only by** (i.e., can represent only). The merging axiom \( x \xrightarrow{AB} y \) states that a BPMN element of type \( x \) can be annotated only with (can represent only) a domain specific concept equivalent to or more specific than \( y \);
- **not annotatable by** (i.e., cannot represent). The merging axiom \( x \xrightarrow{nAB} y \) states that a BPMN element of type \( x \) cannot be annotated with (cannot represent) a domain specific concept equivalent to or more specific than \( y \);
- **annotates only** (i.e., can be represented as). The merging axiom \( y \xrightarrow{A} x \) states that any domain specific concept equivalent to or more specific than \( y \) can be used to annotate only (can be represented as) BPMN elements of type \( x \);
- **cannot annotate** (i.e., cannot be represented as). The merging axiom \( y \xrightarrow{nA} x \) states that any domain specific concept equivalent to or more specific than \( y \) cannot be used to annotate (cannot be represented as) BPMN elements of type \( x \).

Thus, expression (4) can be formalized as the merging axiom: to_purchase \( \xrightarrow{A} \) sub-process, which in turn is represented as the DL statement \( \text{BDO:to\_purchase} \sqsubseteq \text{BPMNO:sub\_process} \), where BDO and BPMNO are labels used to indicate the BPMN ontology and the domain ontology respectively. The automatic translation of the four annotation constructs into DL axioms is the basis for the
translation of the informal expressions such as (1)–(6) into a formal set of expressions. Note that although the meaning of \( x \xrightarrow{nAB} y \) and \( y \xrightarrow{nA} x \) coincide, we provide both primitives as, depending on the case to be modeled, one may be more intuitive than the other. The formal representation of the merging axioms allows reasoning to: (i) check whether the annotations satisfy or violate the annotation criteria expressed by the merging axioms, and (ii) identify the list of admissible annotations that can be suggested to the analysts in the collaborative framework. If a violation of a constraint occurs, explanation techniques similar to the ones described in [8] can be used to provide an indication of what went wrong and how the analysts can repair the annotation or revise the merging axiom(s).

In spite of the advantages provided by merging axioms, their definition has a cost as they need to be specified by the analysts in order to be used. To support the creation of merging axioms, we have implemented a first library of domain independent merging axioms between BPMN and DOLCE (see [9] for a detailed description). Based on this work, expression (1) can be represented with the merging axiom activity \( AB \) process (identifying action with class process in DOLCE), expression (2) can be represented with the merging axiom data_object \( nAB \) perdurant (where DOLCE class perdurant is a general class covering both processes and events) which in turn is represented with BPMNO: data_object \( \sqsubseteq \) BDO: perdurant; and similarly with the other expressions.

4.2. Structural Constraints

These constraints are expressions used to state specific properties that apply to the process under construction. Differently from merging axioms, these expressions can have many different forms to match a variety of different properties of the process\(^6\). Here we focus on the two main forms: (i) containment constraints (including existence constraints), and (ii) precedence constraints, with an emphasis on how typical requirement patterns can be formally represented using these constraints.

4.2.1. Containment Constraints

Containment constraints are expressions of the form \( X \) contains \( Y \) or \( X \) is contained in \( Y \) and are used to represent the fact that the BPD or certain graphical elements contain other graphical elements. As such they can be used to express also informal statements of the form \( \exists Y \) and \( \exists \neg Y \), which are rephrased in the containment constraint diagram \( X \) contains \( Y \) and diagram \( X \) does not contain \( Y \). An example of a simple containment constraint of the form \( X \) contains \( Y \) is provided by requirement:

\[ \text{The purchase activity contains an activity that sends the payment} \]  

expressed by the retail seller expert, while an example of non-existence constraint is provided by the following global requirement which all the experts have decided to impose on the integrated process:

\[ \text{Inclusive gateways cannot be used in the process diagram} \]

Containment constraints can be encoded in DL using specific roles (i.e., binary relations) of the BPMN ontology which formalise the containment relations existing between different BPD objects.

\(^6\)The interested reader can refer to [7] for a formal description of how to encode different types of constraints over annotated BPDs in DL.
as described by specific attributes in [10]. Examples of these roles, used in DL to represent object properties and data properties, are:

- **has\_embedded\_sub\_process\_sub\_graphical\_elements**. This role corresponds to the GraphicalElement attribute of an Embedded Sub-Process, as described in [10], and represents all of the objects (e.g., Events, Activities, Gateways, and Artifacts) that are contained within the Embedded Sub-Process;
- **has\_pool\_process\_ref** which corresponds to the ProcessRef attribute and is used to represent the process that is contained within a pool;
- **has\_process\_graphical\_element** which corresponds to the GraphicalElements attribute of BPMN and identifies all of the objects that are contained within a process;
- **has\_business\_process\_diagram\_pools** which allows to relate a BPD with the pools it contains.

Requirement (1) refers to a containment between two activities, and therefore imposes a constraint between a (sub-)process and the graphical elements it must contain. Thus it can be formalized by means of the role **has\_embedded\_sub\_process\_sub\_graphical\_elements** as follows:

\[
\begin{align*}
\text{BDO:to purchase} & \sqsubseteq \text{BPMN:embedded\_sub\_process} \\
\text{BDO:to purchase} & \sqsubseteq \Box \text{BPMN:has\_embedded\_sub\_process\_sub\_graphical\_elements. (BPMN:activity} \\
& \quad \cap \text{BDO:to send payment})
\end{align*}
\]

Here, axiom (3) makes the purchase activity necessarily a sub-process (as it contains further activities), while axiom (4) states that purchase contains an activity that sends the payment. What is important to note here is that using the roles listed above we can provide an immediate encoding for a number of requirement patterns that can be automatically translated into DL axioms without any intervention of the business analysts. In fact, if we abstract away from the specific activities in requirement (1), we can obtain a requirement pattern of the form

\[
\text{The activity X contains the activity Y}
\]

which can be immediately encoded in the axiom skeleton

\[
\begin{align*}
\text{BDO:X} & \sqsubseteq \Box \text{BPMN:embedded\_sub\_process} \\
\text{BDO:X} & \sqsubseteq \Box \text{BPMN:has\_embedded\_sub\_process\_sub\_graphical\_elements. (BPMN:activity} \\
& \quad \cap \text{BDO:Y})
\end{align*}
\]

Now, by replacing X and Y in the expression pattern (5) the business analysts can specify their domain dependent requirements, which can be automatically encoded in DL axioms by replacing X and Y in (6) and (7) with the appropriate terms of the domain ontology.

### 4.2.2. Precedence Constraints

Precedence constraints are used to represent the fact that certain graphical objects appear before others in the BPD. In their general form they are expressions of the form *X precedes Y* and *X is preceded by Y*. From this very general form several variations can be obtained. Significant examples are: *X is always preceded by Y* in all possible paths made of sequence flows, and *X is once preceded by Y* in...
at least a path composition of sequence flows. Further refinements of these constraints are \( \text{X is always immediately preceded by Y} \) and \( \text{X is once immediately preceded by Y} \). These latter constraints require that X is a graphical object immediately preceded by Y, which is connected to it by means of a sequence flow. Finally the precedence constraints \( \text{X generates Y} \) and \( \text{X is generated by Y} \) require that X activates an event Y by means of a message flow or vice-versa. Two simple examples of precedence constraint are provided by the following requirements expressed by the retail seller expert and by the wholesale seller expert respectively:

\[
\text{The activity of storing the stock information is always preceded by an activity of receiving the stock} \quad (8)
\]

\[
\text{The activity of reminding the payment generates a reminder event (in the customer pool)} \quad (9)
\]

Precedence constraints can be encoded in DL using specific roles of the BPMN ontology which formalize the connection between graphical objects. In particular the key roles we can use are:

- has\_sequence\_flow\_source\_ref and has\_sequence\_flow\_target\_ref.
- has\_message\_flow\_source\_ref and has\_message\_flow\_target\_ref.

These roles represent the SourceRef and TargetRef attributes of BPMN and identify which graphical elements the connecting object is connected from and to respectively. The first two roles refer to connecting object which are sequence flow, while the other two roles refer to message flow.

Using the has\_sequence\_flow\_source\_ref and has\_sequence\_flow\_target\_ref roles, constraint (8) can be formalized in DL by means of two quite complex statements below:

\[
\begin{align*}
\text{BDO:to store stock info} & \sqsubseteq \forall \text{BPMN:has sequence flow target ref. BDO:to receive stock}. \\
\forall \text{BPMN:has sequence flow source ref. BDO:to receive stock}. & \quad \text{BDO:to receive stock} \equiv \neg \text{BPMN:start event} \sqcap ((\text{BDO:to receive stock} \sqcap \\
\text{BPMN:activity}) \sqcup \forall \text{BPMN:has sequence flow target ref.}.
\end{align*}
\]

(10) (11)

The statements above use has\_sequence\_flow\_source\_ref and has\_sequence\_flow\_target\_ref, together with an auxiliary concept BDO:to receive stock\^\*. In a nutshell the idea is that the concept BDO:to store stock info is immediately preceded, in all paths defined by a sequence flow, by a graphical object of type BDO:to receive stock\^\* as stated by axiom (10). This new concept is, in turn, defined as a graphical object which is not the start event and either it is an activity of type BDO:to receive stock or it is preceded in all paths by BDO:to receive stock\^\* as stated by axiom (11). Again, the important point here is not the specific formalisation provided by axioms (10) and (11) but the fact that by replacing the specific activities with X, Y and Y\^\* we can obtain a general encoding for the requirement pattern of the form

\[
\text{The activity X is always preceded by an activity Y} \quad (12)
\]
as follows:

\[
\begin{align*}
\text{BDO:}X & \sqsubseteq \forall \text{BPMN:has_sequence_flow_target_ref}. \\
\forall \text{BPMN:has_sequence_flow_source_ref.BDO:}Y & \\
\text{BDO:}Y & \equiv \neg \text{BPMN:start_event} \sqcap \forall \text{BPMN:activity} \\
\forall \text{BPMN:has_sequence_flow_target_ref}. \\
\forall \text{BPMN:has_sequence_flow_source_ref.BDO:}Y &
\end{align*}
\]  

(13)  

(14)

Similar axiom schemata can be provided for the other precedence patterns illustrated above. Again, the business analysts can use and instantiate patterns such as (12) to obtain specific requirements such as (8), which are in turn automatically encoded in formal expressions by using the axiom schemas above.

5. Tool

To support the collaborative specification of semantically annotated business processes we have developed a prototypical version of BP-MoKi, a collaborative tool based on Semantic MediaWiki (SMW)\(^\circ\) [11]. Inspired by our previous work presented in [12], BP-MoKi offers specific support to edit and semantically annotate business processes, to edit ontologies, and to build an OWL implementation of the BPKB. In order to cover such a variety of functionalities, rather than building BP-MoKi completely from scratch, we opted to start from well-established and robust tools available, the main ones being MediaWiki, Semantic MediaWiki, Semantic Forms, and the Oryx Editor, adapting and extending them for the specific features of our framework.

The choice of developing BP-MoKi on top of a semantic wiki was actually made for several reasons. First, wikis are web-based systems, hence they are accessible virtually from every place in the world: this feature is particularly suitable since members of the modelling team may not be able to physically participate in meetings (e.g. different towns, different buildings). Second, SMW is a state of the art collaborative editing tool, and users are quite familiar with wikis and editing of wiki pages, due to the growing popularity of wiki-based web sites. Third, wikis provide a uniform interface for accessing the different components required to semantically annotate business processes (i.e. ontologies and process). Fourth, the natural language descriptions inserted into a semantic wiki can be structured according to predefined templates, with the help of semantic annotations (e.g. properties). As a consequence, the informal descriptions in natural language contain enough structure to be automatically translated into a formal representation, thus allowing the reuse of informal descriptions for the automatic creation of the BPKB. Fifth, many important functionalities (access control\(^\parallel\) and permissions, tracing of the activity, semantic search, etc.) are already provided by the SMW framework, without needing to install specific client applications. Furthermore, due to the modular way in which the SMW is built, missing functionalities can be added by integrating additional extensions implementing...

\(^{\circ}\) See http://semantic-mediawiki.org - We currently use MediaWiki v1.14 and SMW v1.4.2.  
\(^{\parallel}\) Mediawiki does not natively support any locking mechanism on pages, and was not written to provide per-page access restrictions. However, we are currently implementing these functionalities in order to prevent concurrent editing of the same page, as described in Section 3.

13
them. Sixth, all the actors involved in the modelling activities can also interact with each others and exchange further ideas and comments using the discussion SMW’s built-in functionality. Finally, only a web-browser is required on the end user side to use the system.

As reported in the literature, it has to be noted that general wikis usually present also some disadvantages, especially for what concern knowledge management (see e.g. [13], [14], [15]). Among them,

- usually wikis are open, i.e., anyone can edit / view the content, and this is not desirable when dealing with e.g., confidential information;
- if not managed properly, wikis may be prone to SPAM and vandalism;
- the retrieval of knowledge is usually not as easy as creating the content;
- usually users need to learn the wiki syntax in order to appropriately use the system.

However, in the context of our tool, the danger of these disadvantages is limited as access restriction mechanisms, form-based editing and SMW semantic annotations are employed.

The main idea behind BP-MoKi is that the ontologies and the process are represented as a collection of interrelated wiki pages connected by typed links. To each concept of the BPMN ontology, each concept of the Business Domain ontology, and each process element of the Business Process, a distinct and unique wiki page in BP-MoKi is associated. Each page contains an informal but structured description of the element it represents. This description is actually used to automatically create the formal representation in the BPKB. A typical page contains:

- an informal description of the element in natural language (images or drawings can be attached as well), whose purpose is to document the model and clarify it to users not familiar with the formal representation of processes or ontologies;
- a structured part, where the element is described by means of triplets of the form (subject, relation, object), with the element itself playing the role of the subject. The purpose of this latter part is to represent the intra/inter-connection between the elements of the models (e.g., the subclass relation between elements of ontologies, or the annotation of an element of the business process with a concept from the business domain ontology).

The user fills a page via forms, so he/she does not need to know any particular syntax or language to edit the content of a BP-MoKi page. In what follows we present the specific support that the current version of the tool provides for each of the four activities envisaged in our collaborative modelling approach.

5.1. Process Construction

To support creation and editing of a business process, BP-MoKi integrates Oryx**, a state of the art collaborative tool for the graphical modelling of business processes. Oryx is web-based, so it allows to model the process directly in the browser and to share it with partners by using common repositories. Its editor supports different process modelling languages, including BPMN, and the possibility to adapt

**See http://bpt.hpi.uni-potsdam.de/Oryx
it to personal needs by means of extensibility mechanisms. Figure 3 displays a screenshot of BP-MoKi during some editing activities of a process\textsuperscript{††}. Each element of the process described in Oryx is associated to a page in BP-MoKi. This page allows adding a semantic annotation to the element of the process and inserting additional information/documentation about that element of the process. In general, this is not possible with a generic, off-the-shelf BPMN editor (e.g., a natural language description of an activity, references to external source documents, etc). Each time the process edited through Oryx is saved, BP-MoKi automatically aligns the pages describing the process.

5.2. Ontology Construction

Each concept of the Business Domain ontology is accessible in BP-MoKi via a page similar to the one in Figure 4. To support the development of the ontology, BP-MoKi provides the following functionalities, accessible via a Wiki-style menu:

\textbf{Import Functionalities.} Using the \textit{import of available business domain ontology} functionality, the user can set up BP-MoKi with an available business domain ontology. Using the \textit{input of structured lists of domain ontology concepts} functionality, the user can create new concepts in the ontology by inserting a

\begin{footnotesize}
\begin{enumerate}
\item BP-MoKi currently integrates Oryx Editor v1.12. A newer version (v1.15) of Oryx is available, and we plan to update BP-MoKi with it.
\end{enumerate}
\end{footnotesize}
Modify business domain ontology element: To estimate date

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description:</strong> The company makes an estimation of the date of completion of a product</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hierarchical Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Is a:</strong> To estimate</td>
</tr>
<tr>
<td><strong>Is part of:</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Additional Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add another</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Merging Axioms</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Annotates only:</strong> Cannot annotate:</td>
</tr>
</tbody>
</table>

Figure 4. An example of a BP-MoKi page representing a concept of the domain ontology.

list of terms, where indentation is used to indicate a hierarchy, either taxonomy or partonomy, according to user’s choice.

**Ontology Management Functionalities.** This set of functionalities provides basic support for creating, editing and deleting ontology elements. Pre-defined templates, containing for instance properties for specifying a taxonomy or partonomy, are loaded when the page describing the element is created or edited.

**Visualization Functionalities.** These functionalities allow to produce different types of graphical overviews of the ontology: they help the modellers to deal with the global picture of the ontology and not only with single ontology elements.

In the **tabular view**, the user sees a table listing all the concepts of the domain ontology, where for each concept some relevant information is shown, e.g., its description, the concepts of which it is a specialisation, and so on.

In the **tree view**, a tree shows the hierarchy of the domain concepts according to either the subclass or part of relation. This tree-like view, an example of which can be seen in Figure 5, is dynamically created from the content of the BP-MoKi pages. The user has the possibility to expand/collapse parts of the tree, thus efficiently browsing even large and complex ontologies. The user can also rearrange the taxonomy and partonomy of concepts in the domain ontology via drag ‘n’ drop. Furthermore, the
user can add/delete/rename concepts directly within the browser. The changes performed within the browser are propagated to the pages describing the elements involved.

The BPMN ontology described in [16] is already pre-installed in BP-MoKi (even though the tool allows to replace it with another one). No user-editing of the BPMN ontology is allowed within the tool, but the user can easily browse the ontology with the same visualisation functionalities described for the Business Domain ontology.

5.3. Constraints definition

In the current version of the tool, *annotatable only by* and *not annotatable by* constraints can be added via a form included in the pages describing the concepts of the BPMN ontology. With the hints provided by autocompletion, the user can easily insert the desired element from the BDO. Similarly, *annotates only* and *cannot annotate* constraints can be inserted via a form included in the pages describing the concepts of the BDO, with autocompletion support with respect to the BPMN concepts (see for example the *annotates only* and *cannot annotate* forms in the lower part of Figure 4).

Structural constraints are defined in separate pages, where the user can add new constraints guided by simple forms encoding the requirement patterns described in Section 4.2. Figure 6 shows how the user can add / edit activities containment constraints according to the expression pattern (5): autocompletion support is provided to support the insertion of the activities considered, while a drop-down box allows the user to choose between the *contains* and *does not contain* variants of the containment constraint.
Finally, thanks to the constraints listing functionalities, the user can easily browse in a tabular view all the constraints created, grouped by type (e.g., all *annotates only* constraints).

5.4. Process Semantic Annotation

To support the semantic annotation of the business process, the page corresponding to each of the elements of the process includes a form where the user can insert, with the support of autocompletion, an element from the BDO. An example of annotation of a business process element is shown in Figure 7. Thanks to the business process element listing functionality, the user can easily browse - in a tabular view - all the annotations inserted so far.

5.5. Models export and annotation correctness check.

The ontology and the process developed with BP-MoKi can be independently exported using the corresponding functionalities. The BDO, formally represented in the OWL language, is exported as a file in OWL/RDF format, while the business process is exported as an XML file that contains the structural description of the process, as well as the details to graphically render the process workflow in Oryx.

BP-MoKi also provides the functionality to save the whole Business Process Knowledge Base into a single OWL file, which formalizes in OWL all the components of the knowledge base, according
Modify business process element: Estimate a date for the production completion

<table>
<thead>
<tr>
<th>Element Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description:</td>
</tr>
<tr>
<td>ID:</td>
</tr>
<tr>
<td>Name:</td>
</tr>
<tr>
<td>Activity Type:</td>
</tr>
<tr>
<td>Task Type:</td>
</tr>
<tr>
<td>Subprocess Type:</td>
</tr>
<tr>
<td>Status:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Annotations</th>
</tr>
</thead>
<tbody>
<tr>
<td>BPMN Element:</td>
</tr>
<tr>
<td>BD Ontology Concept:</td>
</tr>
</tbody>
</table>

Figure 7. An example of a BP-MoKi page representing an element of the business process.

to [4]. Once the OWL representation of the knowledge base is created, a java tool‡‡ integrated into BP-MoKi can be invoked in order

1. to check the correctness of the semantic annotation of the business process,
2. and to verify the satisfiability of the constraints imposed on the business process.

Both these tasks are accomplished by the tool by executing some reasoning tasks over the BPKB created.

We performed some experiments in order to evaluate in particular the performance of DL reasoners for checking the satisfiability of structural constraints. We considered four processes of increasing size and complexity. The number of BPMN graphical elements and of semantic annotations contained in the four BPDs is provided in Table I. The machine used for the experiments is a 1.86GHz Intel Core 2, with 2 Gb of RAM and running Windows XP.

‡‡The java tool is based on Jena libraries and Pellet reasoner

“This check requires some carefulness, as OWL semantics is based on the Open World Assumption (i.e., a failure in proving a statement does not imply that the statement is false) and does not satisfy the Unique Names Assumption (i.e., two entities with different identifiers are distinct objects), which make it difficult to use OWL for data validation in a closed world. To overcome this issue, we rely on the techniques and tools discussed in [17], where an Integrity Constraint semantics for OWL axioms is provided in order to enable closed world constraints validation.
Table I reports the time spent by the Pellet reasoner to classify the BPKBs which encode the annotated processes without constraints, and the BPKBs enriched with a set of constraints similar to the ones described in the paper. Note that, in the case of unsatisfiable classes, which originate from conflicting requirements, an off-line usage of DL reasoners and explanation techniques similar to the ones described in [8] can be also useful to provide justifications to the business experts. Figure 8 shows an explanation obtained by using the Explanation Workbench plugin for Protege-4 of the unsatisfiable concept to_estimate_date in the case the assertion BDO:to_estimate_date ⊑ BPMN:embedded_subprocess is added to the knowledge base, together with the constraint BDO:to_estimate_date ⊑ BPMN:task. In our future works, we plan to integrate a similar justification functionality directly into BP-MoKi.

Table I. Evaluation results.

<table>
<thead>
<tr>
<th>Process</th>
<th>Process Elements</th>
<th>Semantic Annotations</th>
<th>Classification of BPKB</th>
<th>Classification of BPKB with Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROCESS_1</td>
<td>79</td>
<td>13</td>
<td>5s</td>
<td>205s</td>
</tr>
<tr>
<td>PROCESS_2</td>
<td>196</td>
<td>30</td>
<td>20s</td>
<td>143s</td>
</tr>
<tr>
<td>PROCESS_3</td>
<td>324</td>
<td>56</td>
<td>28s</td>
<td>338s</td>
</tr>
<tr>
<td>PROCESS_4</td>
<td>629</td>
<td>112</td>
<td>750s</td>
<td>1141s</td>
</tr>
</tbody>
</table>

Figure 8. Explanation generation.
6. Case Study

In this Section, we describe the application of the proposed semantically-based collaborative approach for the integration of three independent processes and we investigate the benefits of the approach and the tool we used for the integration by analyzing different factors, such as the use of semantic annotations, of a shared ontology and local and global constraints. In the following we will first describe the case study, by detailing a possible application scenario for our framework (Subsection 6.1), and then we will define and answer the research questions we investigate with the current case study (Subsection 6.2).

6.1. Scenario.

The processes to be integrated in the proposed case study are the three processes of the motivating example described in Section 2: one of them represents the management of the inventory realized by a retail seller (RS) company; the second one the order management performed by a wholesale seller (WS) company and the third one the dispatch process executed by a delivery (D) company. Three domain experts, one for each company (and corresponding domain), are involved in the process integration: (1) the RS expert, responsible for the part of the process that will be realized by the RS partner; (2) the WS expert, involved in the design of the part of the process related to the WS partner; and (3) the D expert, in charge of the D company’s part of the process.

6.1.1. Scenario Description.

In the following we provide and comment some snapshots of the artefacts in the workspace, captured at different times of a specific execution of the process integration, with the aim of showing how our approach can be applied for supporting collaborative process integration.

Snapshot 0: initial domain ontology agreement. Initially, the three experts agree on a generic domain ontology, transversal to the three domains, that will be used as starting point for the incremental construction of the final domain ontology artefact reported in Figure 9. The ontology, modelled by exploiting the tool functionalities for the ontology construction, contains both the main concepts characterizing each specific domain (e.g., to select supplier for the RS company and to produce stock for the WS company) and shared concepts (e.g., to estimate and to estimate date).

Snapshot 1: process semantic annotation and ontology enrichment. Each of the experts is asked to semantically annotate his/her business process activities and events with concepts contained in the ontology by using the natural language based template provided by BP-MoKi and exploiting the suggestions provided by the tool for avoiding the violation of the “default” merging axioms. Some of the concepts required for the semantic annotation are already in the ontology, while others need to be added, thus enriching the ontology. However the experts have to be careful in the ontology enrichment, in order to avoid synonyms and redundancies.

For instance, Figure 14 depicts the semantically annotated process of the RS company. When annotating his process, the WS expert realizes that the to check inventory concept added to the ontology by the RS expert and used for annotating his process (Figure 10(a)), could be replaced by the more generic concept to check availability, that could be used also for the annotation of one of the activities of his company’s process (Figure 10(b)). Thus, the WS expert starts a discussion forum for renaming
the concept and, finally, the three experts agree on the concept renaming. The tool replaces the process annotations that use the old concepts with the new ones (Figure 10(c)). The final semantically annotated process of the WS company is depicted in Figure 15.

Similarly, while annotating her “Ask for payment” activity, the D representative notices that both the concept payment and money (and the derived concepts) have been used, thus making the ontology redundant. She therefore suggests to remove money (and the derived concepts) and to replace all its occurrences with payment; all the experts agree on the proposal. The semantically annotated process of the D company is depicted in Figure 16.

**Snapshot 2: constraint definition.** Before proceeding with process merging, the experts identify the constraints local to their processes that should not be violated in the integration phase. Some of them explicitly define the communication between pairs of partners, thus guiding the integration, others
describe the control flow of a single pool and a third kind contains global constraints stated together by the three experts. The identified constraints, expressed by the experts using the natural language constructs described in Section 4, are reported in Table II and classified according to the constraint type and expert definer.

For example the RS company identifies a set of constraints describing the communication with the supplier partner (from cc1 to cc6). The RS expert, however, also identifies some constraints only related to the control flow of the pool of his process (pc1, pc4, pc5).

The WS expert, on his turn, constrains the communication with both the customer (constraints from cc8 to cc12) and the deliverer partner (constraints from cc15 to cc18). Some of these constraints are symmetrical to those defined by the RS expert (cc8 is the symmetric of cc1, cc10 of cc3, cc11 of cc4 and cc12 of cc2). Others, instead, do not have any correspondence in the RS process (cc9, cc13 and cc14). Finally, a last set of constraints rules the WS pool control flow (pc2, pc6, pc7 and pc8).

The D expert defines the constraints related to the communication of the main pools of her process with the partners. Also in this case, some of them are symmetric to those defined by the WS expert (cc25 is the symmetric of cc16 and cc19 of cc15). Others, instead, involve activities and events that do not yet have any correspondent in the partner pool (cc17, cc22, cc23 and cc24).

In this initial phase, the experts also agree on two global constraints. One of them is a very general constraint related to the use of inclusive gateways (gc1), while the other is inspired by the policy changes related to the partner selection after the agreement among the three partner companies (gc2).

**Snapshot 3: RS and WS process merging.** After completing the local constraint definition step, the RS and WS experts start merging their processes. The merging activity is non-trivial: in fact, it has to take into account the constraints defined for each pool, the choices performed by the partners and the changes in policies that the new synergy creates. For example, the internal flow in the “Purchase” subprocess of the RS process is not completely defined, while the symmetrical part is very well described in the WS process. The constraints defined for the communication between pools do not only guide
Pool constraints.

pc1 RS The activity of purchasing is a complex activity and can be represented only as a sub-process.
pc2 WS The activity of delivering is a complex activity and can be represented only as a sub-process.
pc3 D The activity of stock delivering is a complex activity and can be represented only as a sub-process.
pc4 RS The activity of purchasing contains an activity that sends the payment.
pc5 RS The activity of storing the stock information is always preceded by a stock receipt event.
pc6 WS The activity of estimating a date is always preceded by the activity of checking the availability.
pc7 WS The activity of sending an estimate is always preceded by the activity of estimating a date.
pc8 WS The activity of delivering is always preceded by the order confirmation event.

Communication constraints.

cc1 RS The activity of sending an order generates an order receipt event (in the supplier pool).
cc2 RS The activity of sending a payment generates a payment receipt event (in the supplier pool).
cc3 RS The activity of confirming an order generates an order confirmation event (in the supplier pool).
cc4 RS The activity of cancelling an order generates an order cancellation event (in the supplier pool).
c c5 RS The order acceptance event is generated by an activity that accepts orders (in the supplier pool).
cc6 RS The order refusal event is generated by an activity that rejects orders (in the supplier pool).
cc7 RS The stock receipt event is generated by an activity that delivers orders (in the supplier pool).
cc8 WS The order receipt event is generated by an activity that sends orders (in the customer pool).
cc9 WS The activity of sending an estimate generates an estimate receipt event (in the customer pool).
cc10 WS The order confirmation event is generated by an activity that confirms orders (in the customer pool).
cc11 WS The order cancellation event is generated by an activity that cancels orders (in the customer pool).
cc12 WS The payment receipt event is generated by an activity that sends payments (in the customer pool).
cc13 WS The activity of reminding the payment generates a reminder event (in the customer pool).
cc14 WS The activity of notifying a charge generates an charge notification event (in the customer pool).
cc15 WS The activity of requesting the delivery generates a delivery request event (in the pool).
cc16 WS The activity of sending the payment generates a payment receipt event (in the deliverer pool).
cc17 WS The activity of requesting an estimate generates an estimate request event (in the deliverer pool).
cc18 WS The estimate receipt event is generated by an activity that sends estimations (in the deliverer pool).
cc19 D The delivery request event is generated by an activity that requests deliveries.
cc20 D The activity of sending an estimate generates an estimate receipt event.
cc21 D The activity of refusing a request generates a request refusal event.
cc22 D The request confirmation event is generated by an activity that confirm requests.
cc23 D The request cancellation event is generated by an activity that cancels requests.
cc24 D The activity of requesting payment generates a payment request event.
cc25 D The payment receipt event is generated by an activity that sends payments.
cc26 D The activity of delegating the delivery generates a delivery request event (in the transporter pool).
cc27 D The activity of unloading stocks generates a stock receipt event (in the customer pool).

gc1 D Inclusive gateways cannot be used in the process diagram.
gc2 D The activity of selecting a partner is always preceded by a cancellation activity or by a refusal event.
gc3 D The activity of cancelling order always precedes an activity that sends information.
gc4 D The activity of sending information generates an information receipt event.
gc5 D The information receipt event always precedes an activity that stores the information.

Table II. Local and global constraints defined by the three experts. For each of them an identifier and the expert who specified it are reported.
Figure 11. A detail of the modifications applied to the RS and WS processes due to the constraint violation in the integration phase: the process elements added to the original processes are depicted with a dark background, while the circle around a process element denotes the element removal.

the two experts in defining the messaging flow between the two pools, but also in determining the new flow of the merged process. In fact, once events and activities already in the pool are connected with message flows (constraints cc1 and cc8, cc2 and cc12, cc3 and cc10, cc4 and cc11), the violation of other constraints raised by the tool induces the experts to introduce an estimate receipt event (cc9), a reminder event (cc13), a charge notification event (cc14), to cancel the service confirmation and the service cancellation events and to re-arrange the control flow in the sub-process, by adding and removing some gateways and introducing a new to send payment activity. The framework also identifies and locates the violation of the global constraint (gc2) that implies the cancellation of the to select supplier activity preceding the “Purchase” sub-process. The modifications imposed by the constraint violations and the corresponding violated constraint are shown in Figure 11. In detail, the new process elements are depicted with a dark background and a circle is reported around the removed elements. Finally, some refinements like the introduction of the unspecified “Purchase from a new supplier” sub-process are suggested by the policy changes. The final process obtained after the merging of the RS and the WS processes is shown in Figure 17. In this picture the dark background indicates all the process elements added to or changed in the original processes because of the violation of some constraint or due to process integration and policy changes.

**Snapshot 4: RS-WS and D process merging.** Once the RS and WS processes have been merged and the constraints for the D process defined, the third pool can also be integrated into the merged process. Also in this case, the symmetric constraints related to the communication between the WS Company and the D Company pools and between the D Company and the RS Company pools suggest the messaging flow between the two pairs of pools (cc9 and cc19, cc16 and cc25, cc18 and cc20). The non-symmetric constraints guide the experts in the introduction of a new estimate request start event (cc17), of a request refusal and a payment request event in the WS pool (cc21 and cc24, respectively) and of a
Figure 12. A detail of the modifications applied to the WS and D processes due to the constraint violation in the integration phase: the process elements added to the original processes are depicted with a dark background.

to confirm request and to cancel request activities in the “Delivery” sub-process of the WS company pool (cc22 and cc23). Moreover, the violation of two of the specific pool constraints defined by the WS expert (pc6 and pc7), suggests the D expert the introduction of two new process elements (i.e., to estimate date and the to send estimate activities). All the changes imposed by constraint violations are shown in Figure 12.

Finally, new activities (e.g., “Select a new delivery company”), gateways and sub-processes (“Delivery with generic delivery company”) are added to the integrated process, due to refinements and policy changes. The integrated process is represented in Figure 18.

Snapshot 5: new global constraints definition. Later the experts agree on the need to keep trace of possible reciprocal cancellations and refusals in order to be able to avoid similar situations in the future. They therefore define a new set of global constraints (gc3, gc4, gc5). The tool identifies the violations raised by these new constraints in the integrated process, so that the experts can easily add the missing activities and events (as shown in Figure 13). The final complete integrated process is shown in Figure 19.

6.1.2. Scenario Analysis.

Although the presented scenario is relatively small, it has features that can be found in real scenarios and its analysis reveals some interesting data related to the complexity of the integration of different views and to the usefulness of adopting a collaborative approach. Actually, each artefact in the shared workspace was managed and evolved collaboratively.

Process construction. The analyzed case study requires the integration of four pools (the main pools in the initial processes of the three partners), containing 87 process elements (i.e., activities, gateways and events) in total. In detail, 24 process elements (13 activities and 5 events) are contained in the
RS pool, 36 elements (14 activities and 7 events) in the WS pool and 27 elements (11 activities and 11 events) in the D pools. In the merging phase new process elements were required: 11 in the RS pool, 23 in the WS pool and 8 in the D pools. Some of them have been added as a consequence of the constraints defined by the experts of the partner companies (7 in the RS pool, 9 in the WS pool and 4 in the D pools), while others are due to global constraints commonly agreed by the three experts (4, 14, and 4, respectively). The final integrated process contains 124 process elements: 53 are activities and 34 events. Among them, 25 activities and 6 events are contained in the RS pool, 21 and 15 in the WS pool and 17 and 13 in the D pools.

**Ontology construction.** In the use case, the 60 concepts of the initial ontology have been incremented up to 105 concepts: 18 concepts have been added by the RS expert, 17 by the WS expert and 10 by the D expert. Moreover, because of redundancies, 1 of them has been renamed and 3 concepts, introduced by the WS expert, have been removed.
**Constraint definition.** Beyond the “default” merging axioms, 3 process specific axioms have been defined, one by each expert. Moreover, 2 containment constraints and 35 precedence constraints have been defined by the experts: 8 by the RS expert, 13 by the WS expert, 9 by the D expert and 5 by the three experts together as a consequence of the policy changes induced by their partnership. The compliance to the specification of most of these constraints (in particular those related to the communication among pools and the global constraints) required the introduction of new activities.

**Semantic annotation definition.** 50 out of the 105 concepts in the ontology have been used by the experts for the semantic annotation of the modelled activities (14 by the RS expert only, 14 by the WS expert only, 11 by the D expert only and 11 by more than one expert) and for the constraint definition (47 concepts, 3 of which have not been used for the annotation of any process element, have been exploited for expressing constraints). In two cases, the concept renaming made by one expert impacted semantic annotations of other experts.

Even in this small example, in fact, managing and coordinating the distributed work carried out by three experts would have involved substantially more effort, for example for fixing errors caused by violated constraints, if not directly supported by BP-MoKi.

### 6.2. Research Questions.

We analyze the presented scenario in order to understand the advantages provided by the use of the framework in activities involving multiple perspectives such as the integration of different processes. In detail, we will answer the following questions:

- **(RQ1)** Does the use of semantic concepts for annotating the process and defining constraints simplify the integration of different perspectives on the same process?
- **(RQ2)** Does the sharing of the ontology help resolving redundancy (and ambiguity) in semantic annotations introduced by different experts?
- **(RQ3)** Does automatic detection of constraint violation help different experts converging on an integrated process which complies with all experts’ requirements?
- **(RQ4)** Does the collaborative definition of global constraints help detecting and resolving integration problems that arise only when different perspectives are merged?

The focus of **RQ1** is the role of semantic concepts in the integration of different perspectives. It investigates whether shared concepts, used both for clarifying the semantics of process elements and defining constraints, help in detecting where and how to integrate different perspectives of the same process. **RQ2**, instead, deals with the use of a shared ontology for the definition of the semantic concepts used for the process annotation and constraint definition. In particular, it analyzes the usefulness of the shared ontology in supporting the disambiguation of concepts (and hence annotations), thus avoiding redundancies and ambiguities in annotations. In **RQ3**, the advantages of the automated detection of constraint violations are analyzed. In detail, the research question investigates the support (in terms of effort required and benefits) provided by the tool to the experts for the generation of a final integrated process still compliant with all the requirements they have expressed. Finally, **RQ4** moves the focus from the constraints defined by considering a single perspective (that have to be preserved by the integrated process) to global constraints, which are not relevant before the integration, but are fundamental for the integrated process. In particular the research question
investigates whether the collaborative definition of global constraints during the integration helps in
detecting and resolving problems.

In the following, by analyzing the data collected during the execution of the integration scenario
described in Subsection 6.1, we will answer the research questions just defined.

**RQ1.** The integration scenario suggests that the use of semantic concepts for annotating processes
and defining constraints (in particular those concerning the communication between pools) guide the
experts in the realization of the integrated process by binding process elements of one process to those
of another. In fact, for example, the presence in the “partner pool” of a process element annotated with
the same concept used for characterizing the “partner element” involved in a communication constraint
(i.e., the generator activity or the generated event) allowed analysts to detect the source or the target of
the message flow to be inserted in the integrated process. Such a mechanism was further strengthened
in the described scenario by the presence of symmetric communication constraints, i.e., constraints
defined for each of the two processes involved in the communication in a symmetric way. For example,
in the integration between the RS Company and the WS Company processes, the pairs of symmetric
constraints cc1 and cc8, cc2 and cc12, cc3 and cc10, cc4 and cc11 guided the experts in the introduction
of four new message flows. However, the advantages provided by the use of semantic concepts for the
process annotation and the constraint definition could be invalidated by the presence of redundancies
or ambiguities in their use. For example, if the activity that sends the order in the RS Company process
would have been annotated with a concept other than `send order` (e.g., `request service`), it would
have been difficult to identify it when looking for “the activity generating the `order receipt` event” of
the WS Company pool (cc8). Hence, we can affirmatively answer **RQ1**, provided that the semantic
concepts are used in a correct way, avoiding redundancies and misuses.

**RQ2.** An attempt towards the resolution of the problem of redundancies and ambiguities relies on
a shared ontology that is initially collaboratively created, and then referred to as well as extended and
updated according to the single expert’s needs, upon other experts’ approval. In the analyzed scenario,
the presence of the ontology allowed the identification of 2 concept redundancies. One of them was
only a potential redundancy, since it had been detected by the expert when adding a new concept to the
ontology: the WS expert wanted to add the `check availability` concept when the `check inventory`
was already in the ontology. In the other case, the expert who added the redundant concept did not
notice the existence of a similar concept already in the ontology. The redundant concept was discovered
only later by another expert, thus imposing the removal of the redundant concept and the renaming of
the removed concept occurrences. Hence, with respect to **RQ2**, only a partially affirmative answer can
be given: sharing the ontology supports experts in the identification of redundancies, though the use of
the ontology requires experts’ effort and is however subject to human errors. Providing or improving
automatic mechanisms for suggesting annotations (e.g., [18]) or for checking ontology redundancy
could support experts in this difficult and time-consuming task.

**RQ3.** The three pictures depicting the modifications applied to the integrated process under
construction because of constraint violations (Figure 11, Figure 12 and Figure 13), provide a hint
on the impact of automated constraint violation detection. In detail, the violation of 13 out of the 40
constraints brought the experts to the introduction of (at least) one new process element (not contained
in the original process) in the integrated process. The only removal of an activity is instead due to the
violation of the constraint gc2. Hence, we can conclude that **RQ3** has an affirmative answer although
a mechanism for the automatic interpretation, location and resolution of constraint violation (in our
future works) would further facilitate the convergence to an integrated process compliant with the different experts’ needs (i.e., satisfying all the constraints defined by the experts).

RQ4. Table II shows the 5 global constraints collaboratively defined by the experts at different times of process integration: the first (gc1) can be attributed to a shared structural choice; the remaining 4 describe policy changes due to the new synergy among the three companies (hence none of them is relevant if applied to a single process). While gc1 did not impact the integration (i.e., it has never been violated during the integration process), the others did, and induced the experts to introduce new process elements accordingly. However, not all the integration problems can be identified and collaboratively expressed as global constraints. Some of them, in fact, are too specific (e.g., the creation of new policies for unforeseen situations and refinements or adjustments due to global changes) to be detected even when all the experts collaborate together. For example the need to add a new sub-process “Purchase from a new supplier” in the RS Company pool or a new sub-process ‘Delivery from a new generic delivery company” in the WS Company pool belong to this set of undetected and unexpressed integration problems, that are hence left to the experts’ care. We can hence conclude that the collaborative definition of global constraints facilitates the process integration, though some of the integration problems cannot be detected and expressed as constraints.

The answers given to the considered research questions strengthen our positive expectation with respect to the framework and BP-MoKi’s ability to support designers (with different expertise and with different views) in the collaborative definition of business processes. The use of unique semantic concepts for process annotation and (global and/or local) constraint definition, by providing a shared knowledge for humans, encourages the collaboration among experts involved in the process definition. Such a collaboration is further strengthened by automatic mechanisms for the detection of constraint violations, that help experts in detecting integration problems, thus both reducing effort and increasing the process compliance to the experts’ requirements. Nevertheless, we are aware that the framework (and the tool) can be further improved by providing the experts with more automated mechanisms, as, for example, tools for the automated suggestion of process annotations, for the automated checking of redundancies in the shared ontology, as well as for the automated interpretation, location and possibly resolution of constraint violations. We plan to investigate these enhancements in our future work.

7. Related Works

The problem of adding formal semantics to business processes has been extensively investigated in the literature [19, 20, 21, 22, 23, 24, 25, 26, 27]. We can roughly divide the existing proposals into two groups: (1) those adding semantics to specify the dynamic behavior exhibited by a business process [19, 21, 22], and (2) those adding semantics to specify the meaning of the entities of a BPD in order to improve the automation of business process management [23, 24, 25, 26, 27]. We clearly belong to the second group.

Thomas and Fellmann [25] consider the problem of augmenting EPC process models with semantic annotations. They propose a framework which joins process model and ontology by means of properties (such as the “semantic type” of a process element). Markovic [28] considers the problem of querying and reasoning on business process models. He presents a framework for describing business processes which integrates functional, behavioural, organizational and informational perspectives: the elements of the process are represented as instances of an ontology describing the process behaviour (based
on $\pi$-calculus), and the annotations of these elements with respect to the ontologies formalizing the aforementioned perspectives are described as relation instances. Born et al. [3] propose to link the elements of a business process to the elements of an ontology describing objects, states, transitions, and actions. These proposals differ substantially from ours, which establishes a set of subsumption (aka subclass or is-a) relations between the classes of the two ontologies being integrated (BPMN meta-model and domain ontology), instead of associating annotation properties to the process instances. This difference has a direct impact on the kind of constraints that can be automatically enforced (e.g., BPMN elements annotatable by domain concepts). De Nicola et al. [26] propose an abstract language (BPAL) that bridges the gap between high-level process description (e.g., in BPMN) and executable specification (e.g., in BPEL). The formal semantics offered by BPAL refers to notions such as activity, decision, etc., while the problem of integrating process model and domain ontology is not their focus. In Weber et al. [19], semantic annotations are introduced for validation purposes, i.e., to verify constraints about the process execution semantics. In their work, semantic annotations with respect to a background ontology are used to ensure that an executable process model behaves as expected in terms of preconditions to be fulfilled for execution and its effects. In Ly [29], the domain knowledge, expressed by means of metamodels, is integrated in adaptive Process Management Systems and used for detecting semantic conflicts. The process semantic correctness can be checked both when modeling and changing templates as well as when applying ad-hoc changes at process instance level.

In the SUPER project [24], the SUPER ontology is used for the creation of semantic annotations of both BPMN and EPC process models in order to support automated composition, mediation and execution. Recently, Groener and Staab [27] presented a pattern-oriented approach in which OWL representation and reasoning capabilities enables expressive process modelling and retrieval. Their process formalisation considers the language primitives of the UML-Activity Diagram, a language which allows to express simpler processes than BPMN, and the connection with the domain knowledge involves the representation of terminological information about activities and subactivities only. Our work represents an extension of the existing literature in that we provide an ontology based approach that supports automated verification of semantic constraints about BPMN process entities and structure as well as their relations with domain specific concepts.

Collaborative business process definition and process integration is supported by commercial tools (e.g., Hyperwave) and by research prototypes (e.g., InterPROM). It has also been the subject of some research works (e.g., [30, 31]). However, no work attempted so far to define a framework for the collaborative definition of semantic business processes based on Wiki-like tools, even though the need for creating a symbiosis between project management and collaborative tools has been recently recognized [32].

Our work builds upon the existing literature on the specification of the semantics of process entities, trying to merge these techniques with the mechanisms supported by collaborative tools such as Wikis. Our approach is quite unique in that it takes into account the collaborative nature of the work carried out by different experts and business analysts, their different competences and skills, and it supports them in the incremental process definition and annotation, ontology creation, constraint specification and verification.
8. Conclusions and Future Work

We have presented a framework and a tool for the collaborative specification of semantically annotated business processes. In our case study, three analysts, with experience and competences in different domains, have worked concurrently for the specification of an integrated process, starting from three single processes. Their activity was sequenced into 5 consecutive snapshots, each strongly dependent on the output of the others. We have shown how the functionalities provided by our tool support the business analysts in: (1) incrementally defining the process and the ontology; (2) making each part of the process compliant with the enforced constraints; (3) evolving the concepts in the ontology, so as to reach a common understanding of their meaning; (4) reusing ontology concepts and process elements whenever analysts can take advantage of the work carried out by others.

In our future work, we will increase the functionalities available in BP-MoKi, supporting additional templates for common patterns of constraints. We will investigate domain independent top level ontologies that may be used as a starting point for ontology construction. We will also improve the usability of the tool, thanks to the execution of further case studies and experiments.

REFERENCES


Figure 14. The RS company process semantically annotated with concepts contained in the shared ontology.
Figure 15. The WS company process semantically annotated with concepts contained in the shared ontology.
Figure 16. The D company process semantically annotated with concepts contained in the shared ontology.
Figure 17. The partial integrated process obtained by merging the RS and the WS processes. The added or changed process elements are represented with a dark background.
Figure 18. The integrated process obtained by merging the RS-WS integrated and the D processes. The added or changed process elements are represented with a dark background.
Figure 19. The final integrated process obtained by adding process elements in order to solve the violations raised by the definition of new global constraints. The added process elements are represented with a dark background.

39