Guidelines for Process Model Annotation in Text

Patrizio Bellan[1,2], Mauro Dragoni[1], Chiara Ghidini[1],
Han van der Aa[3], Simone Ponzetto[3]
[1] Fondazione Bruno Kessler, Povo (Tn), Italy
[2] Free University of Bozen-Bolzano, Bolzano, Italy
[3] University of Mannheim, Germany

October 23, 2023

Contents

1 Introduction

This document describes the guidelines to annotate documents describing a *business process* (e.g., documents describing standard operative procedures). Several definitions of business process exist in literature. The most modern and popular definition of business process is likely the one provided by Weske [3]:

> "a set of activities that are performed in coordination in an organizational and technical environment. These activities jointly realize a business goal. Each business process is enacted by a single organization, but it may interact with business processes performed by other organizations."

As we can see business processes are composed of sequences of activities and their interacting participants (organizational roles, and goals just to mention a few) which are typically captured by business process models specified in a business process modeling language such as Petri Nets, BPMN, or DECLARE, just to mention a few. While a unique commonly agreed description of the elements of a business process, along with their definitions and interactions (that is, a meta-model of a business process) does not exist, references and common sense definitions can be found in the literature that provide a starting point for describing the different elements belonging to business processes and their modeling notations [1, 2].

1.1 Aim

The goal of these annotation guidelines is to annotate process elements and their relations in process description documents. The primary aim of these guidelines is to obtain valuable annotations that can be used to train and test machine learning algorithms on. In these guidelines, we concentrate on annotating typical

process elements. We do not perform a finer annotation of process elements (such as the type of activity) since this type of finer classification of process elements can be easily obtained through the syntactic and semantic analyses of the textual fragments annotated.

2 A short introduction on the elements of a Business Process

The annotation of a document describing a process is a difficult task. Being able to identify process elements requires having at least a rough understanding of the typical elements contained in process modeling languages. In terms of languages, we target a procedural language such as BPMN, although the guidelines may also apply to other standard procedural modeling languages such as the UML Activity Diagram.

Figure 1: A Business Process Diagram in the BPMN language.

Figure 2: A Business Process Diagram in the UML AD language.

To provide an overview of these graphical languages, and the type of elements they typically contain, please refer to the diagrams in Figures 1–2, taken from [2], which provides a model - in the two procedural languages of the same scenario of a customer buying a flight ticket from a travel agency. Besides illustrating the scenario, the diagrams are annotated with speech balloons indicating the type of entity denoted by the graphical constructs. These examples are not meant to train the annotators to be good modelers in these modeling languages but

just to illustrate the type of elements that typically compose a business process diagram.

Following the classification made in [2], we can group these constructs in three macro categories:

1. **Behavioural**. These elements are the ones that refer to the so-called *control flow* of the process, that is to the flow determined by the set of activities that are performed in coordination. This category is the most articulated in a business process and contains at least 3 types of objects:
   - *activites* and *events*, that is the things that happen in time[1]. In our example the activity `check the flight offer` or the event `payment received`.
   - *flow objects*, that is constructs that enable the routing of the flow between the activities such as the sequence relation between activities, or the gateways that enable the routing of the flow. In our example the (precedence) relation between `make flight offer` and `check flight offer` or the (mutually) exclusive gateway between `reject offer` and `book and pay flight`, and finally
   - *states*, that is conditions of the world that affect the flow in the process such as the pre-post conditions for the occurrence of an activity or a guard on a gateway. In our example the (un)satisfied status of the customer w.r.t. a flight offer.
2. **Data object**. These elements usually describe, at a high level of abstraction, the objects upon which an activity acts. Examples in the scenario above are the `flight request` and the `flight ticket`. Note that sometimes these data objects complement the activity itself (as in the case of the data object `flight request`, which is produced by the activity `check travel agency website` while in other cases they are implicitly described in the activity itself, as in the case of `flight ticket` with the activity `prepare ticket`. In this latter case, the data object is often left implicit (see guidelines for the identification of data objects in Sec.5).
3. **Organizational**. These elements are usually related to the *who* question, and often describe, at a high level of abstraction, the roles / organizational structures involved in the activities of the process.

Note: Data and Organizational objects do not exist per-se in a business process diagram but they usually refer to the activities. More formally, they are participants of the activity as they participate in the activity itself.

3 Process Elements Annotation

We aim to propose a general annotation schema able to deal with unknown scenarios. The conceptual layers (described in Sec. 2) slightly differ from the Annotation schema proposed in this document. Since a process model element can be described in different ways not always easily predictable. In particular,

[1]While these elements often have a different meaning in some modeling languages we do not distinguish between them here

we decided to break down activity to differentiate among the activity elements it is composed of. For instance, we capture the activity "action" expression and the object the activity acts on in two different annotation layers. This choice allows to easier the annotation workload and it also reduces the possibility of making errors (for example, connecting with a *Sequence Flow relation* the activity data to the actor responsible for the execution of the activity). For instance, we differentiate the expression describing an Activity from the object the activity uses. The overall goal is to annotate process model elements and their relations in documents.

3.1 Implementation

We implemented the annotation schema described in this document in Inception Annotation tool [2]. The schema can be downloaded from w3id.org/pet.

3.2 Layers Overview

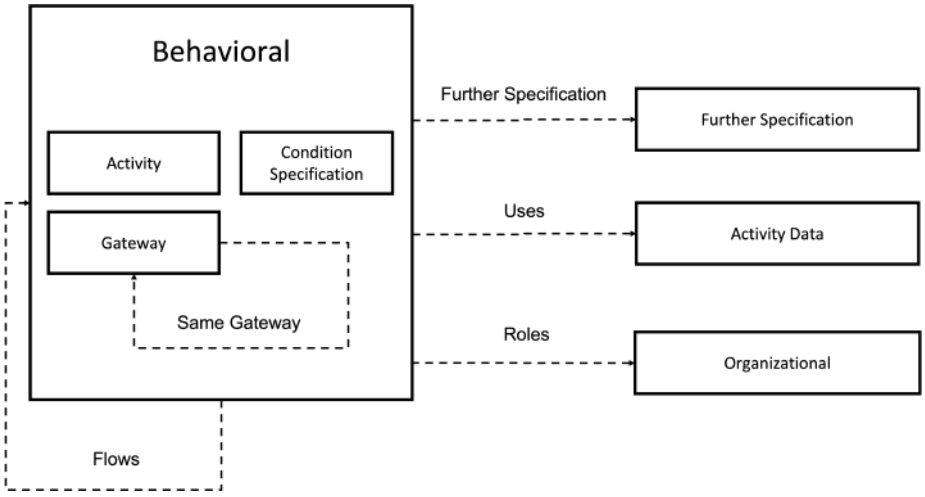

Figure 3: Layers Relations.

The Fig.3 shows the layers' relations. We now explore the process elements considered in this document and their relations, before describing the rules to follow to annotate process elements in text.

3.2.1 Behavioral Layer

The Behavioral layer capture the information about the behavioral elements described and their relations. The Figure 3 shows the relations the Behavioral layer has with the other annotational layers. This is the core layer since it captures activities, gateways, and flow relations. An *activity* element represents a single task performed within a process model. A *gateway* element represents a decision point, the condition specification represents the condition that a process execution instance must satisfy be be allowed to enter a specific branch

[2] https://inception-project.github.io/

of a gateway. A *Flow* is a relation that defines the process logic by connecting all the elements that belong to this layer together.

The Behavioral layer is composed of six features: Element Type, Uses, Flow, Roles, Further Specification, and Same Gateway. Since this layer captures both Activity and Gateways, not all the features are required, but they depend on the Element Type and the situation described in the text. For example, if a text does not describe any Actor Performer the feature Roles is left empty. Rules and guidelines on which features you need to fill are described in the process element's section.

The feature **Element Type** defines the type of the process model element marked. The layer behavioral is connected to the layer Activity Data by the *Uses* relation. This feature links an activity to the Activity Data annotated in the layer *Activity Data*. So, this relation allows to connect an activity expression (either verbal or nominal) with the object the activity acts on. Process participants (actors involved in an activity) that are captured in the *Organizational* layer, are bound to activity through the feature **Roles**. The **Further Specification** feature allows to connect an activity to its important details (captured in the *Further Specification* layer). The Further Specification layer captures the important information of an activity that is not captured by the other layers, such as the means of a sending event. The **Same Gateway** feature allows to "connect together" all the parts describing the same gateway, since its description may span over multiple sentences. This means that only gateway elements can be connected by this relation. The Behavioral layer makes a connection to itself through the relation **Flow**. This feature allows us to define the process logic by connecting behavioral elements together in a sequential order.

3.2.2 Activity Data Layer

The Activity Data layer captures the object of an activity expression acts on. This layer has no features.

3.2.3 Further Specification

The Further Specification layer captures important details of an Activity, such as the mean or the manner of its execution. This layer has no features.

3.2.4 Organizational Layer

The Organizational layer is meant to annotate at a high level of abstraction the process participants that are responsible of activities. They typically represent the *Actors* involved in a process. This layer has no features.

3.3 Annotation Details

The annotation strategy used to annotate a process model element in a text is a span of words performed at the word level (also called token level). Also, each annotation mark is contained within the sentence boundaries. Since each layer captures a specific part of a process model, it is not possible neither stack or overlap annotations. Relations between process elements are performed by a linking mechanism captured in predefined features of a layer. For example, a sequence flow relation between two activities is created by filling the feature

Flow of the Behavioral layer (intuitively, by linking the first activity to the second one).

Annotating a document is a subjective task, there is not a right or wrong way to do it. To help you in this process we propose a best practice annotation procedure in Sec.10. To help you better understand the annotations proposed in the examples of the next sections, we color the annotation and its mark as follows: Activity, Activity Data, Further Specification, Actor, Condition Specification, Gateway.

In the next sections, we provide guidelines and rules to follow during the annotation task. We start with Activity (Sec.4) and its elements: Activity Data (Sec.5), Further Specification (Sec.6). Then, we move on to the description of the guidelines to annotate the Actor in the Organizational layer (Sec.7) and how to bind them to activities. Next, we describe how to annotate Gateways (Sec.8) and the Condition Specification associated with gateway's branches (Sec.8.3). Finally, We describe how to define the process logic by means of Flow relations (Sec.9).

4 Activity

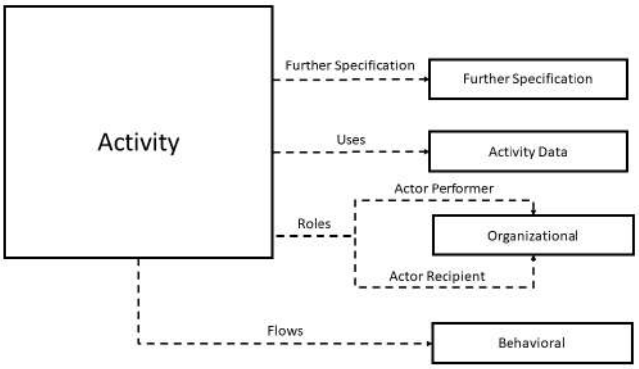

Figure 4: Activity layer and its relations.

We start this Section with the theoretical definition of an Activity, next, we move on to the description of the relations it makes with the other layers, and then, we propose the annotation rules to follow when annotating activities in a text. Then we analyze Activity Data (Sec.5) and Further Specification (Sec.6) since they can be considered components of activity (in BPM terms).

Definition 1 (Activity)**.** *An* Activity *is a single unit of work performed in a process.*

Whereas this theoretical definition well defines the concept of Activity, it is not so trivial to define a precise set of rules to detect activities in text. When annotating activities there are several difficulties to consider. We describe them quickly here and then we refer to the specific list of annotation rules below. In business process model diagrams are usually report two types of activities:

- endogenous activities. These are the ones that are *performed* within the process owner boundaries (e.g., performed by a relevant process actor) and are usually described with the element task in BPMN. As an example think of a process of ordering a delivery pizza: `bake pizza` is an endogenous activity (a BPMN task) likely performed by the pizza baker.

- exogenous activities. These are the ones that *happen*, sometimes within the process owner boundaries (think for instance to an alarm) and often outside the process owner boundaries but affect the process itself. As an example, the latter `receives bake pizza` is an exogenous activity that may trigger the process of ordering a delivery pizza. They are usually described with the element event in BPMN.

At this stage, we do not make any distinction between the two elements. Therefore both endogenous and exogenous activities should be tagged as using the Activity tag.

Concerning granularity, Activity can be of one of two types:

- atomic activity. In this case, the activity is not decomposed into smaller activities. That is, the unit of work is minimal.

- compounded (compound activity). In this case either the activity is further decomposed into two (or more) (sub) activities or it is described at a very high level of detail and one would need further details to understand the set of activities that compose it. In both cases, the activity is not the minimal unit of work. An example of compound activity constituted of two sub-activities is `bakes and sells pizza`. The two sub-activities are `bakes pizza` and `sells pizza`. An example of high-level activity is `ensure adequate management of complaints`, where we know that activities are made, but we do not know exactly which ones.

We decided to propose annotation guidelines to annotate activities at their atomic level. We annotate compound activities (or better high level of granularity) only when their detailed description is not present in the text (see guidelines below).

A further specific challenge when annotating activities is to capture the exact expression used to define it in the text. In a business process diagram, activity is commonly composed of an activity expression (either verbal or nominal) and the object the activity acts on. For instance, in the sentence `the cooker sells pizza` the activity expression `sells pizza`, is formed by the action-verb: `to sell`, and the object used: `the pizza`. The tag Activity should annotate `sells` as Activity, whereas `pizza` is annotated as Activity Data.

As shown in Figure 4, an activity is composed of various parts. An activity annotation has several relations that are captured by means of features in the annotation schema. The feature **Further Specification** allows to capture of important details of an activity, such as the mean or the manner of its execution. The feature **Uses** links the activity expression to the object it acts on (captured in the Activity Data layer). The feature **Roles** allows to link the two possible types of actors (Actor-Performer and Actor Recipient) that are somehow involved or responsible for the activity execution. This feature creates a link between this layer and the Organizational layer. Finally, the feature **Flows** allows us to define the process logic by connecting behavioral elements together.

We try to produce guidelines on how to capture the exact expression used to define it in the text in the rules below.

4.1 Annotation Rules

***Rule* 1.** *An Activity mark should mark only the expression (either verbal or nominal) expressing the activity.*

For instance, you should mark an activity as in the following examples:

Example 1. *Evaluate the seriousness of impact (within 2 calendar days).*

***Rule* 2.** *Do not include in the annotation mark the activity object used by the activity.*

You should mark an activity as in:

Example 2. *ABC Company transmits the data to Company B.*

The Activity data `the data` is left out from the annotation mark of the activity expression `transmits`.

***Rule* 3.** *Exclude from the annotation mark articles, and conjunctions that are at the boundary of the annotation span.*

For instance, you should mark activities as in the following examples:

Example 3. *The managers compiles, saves, and sends the module A to the Company C*

***Rule* 4.** *Exclude from the annotation the modal verbs of the verbal phrase of the activity when it is at the boundary of the annotation.*

You should annotate the activity as in the following example:

Example 4. *ABC Company will send attachment 1 to Company B.*

You should exclude the modal verb `will` from the annotation mark.

***Rule* 5.** *An activity can be expressed with a noun which acts as the verbal expression.*

For instance, consider the following example:

Example 5. *The standard procedure follows two steps: evaluation of complaint and reimbursement.*

This example describes two activities expressed with nouns: evaluation and reimbursement; which intuitively stand for *[evaluate the complaint]* and `reimburse`.

***Rule* 6.** *If an activity is mentioned several times in the text, annotate all the mentions of it (in the same way).*

***Rule* 7.** *The feature Same Gateway is always left empty.*

***Rule* 8.** *If the text does not specify how the compound activity is broken down into sub-activities, then we need to annotate it as an activity; otherwise, we do not annotate it. In this latter case, we annotate each specified activity.*

Compound activities can be activities described, with a single verbal or nominal expression at a high level of detail. An example is the activity `ensure adequate management of complaints`. For instance, consider the following example:

Example 6. *Drug Safety ensures an adequate management of complaints via first collecting all the complaints and then evaluating them.*

You should not tag `ensures adequate management of complaints` as an activity since this activity is specified later in the text.Instead, you should tag the two activities `collecting` and `evaluating`.

***Rule* 9.** *A temporal constraint is neither an activity itself nor a part of an activity and it must not be annotated as such.*

For instance, in the following example:

Example 7. *Daily, the chef prepares the pizza dough.*

the expression `daily` is excluded from the activity's annotation mark

***Rule* 10.** *A condition is never an activity, even if describes something that is done or happens in time.*

The rationale for this is that a guard of a control flow point (gateway) typically represents a state condition and a state is not an activity. Consider the following example:

Example 8. *if the buyers spend more than 100 Euro then ...*

`spends` in the expression `spends more that 100 Euro` is not an Activity.

***Rule* 11.** *If a set of activities acts as a guard of a gateway you should mark the set of activities with a single annotation mark.*

To help you better understand this rule, all the process elements described are marked in the following examples. Consider the following excerpt:

Example 9.
(s.0) *The supervisor can approves or reject the reimbursement form.*
(s.1) *If the reimbursement is approved, the secretary makes the bank transaction.*
(s.2) *If the reimbursement is not approved, the secretary sends a letter to the customer.*

In this case, the set of activities approves or rejects acts as a guard of a gateway. Therefore, you are required to annotate the set of activities with a single annotation mark. Figure 5 presents the schematic representation of this excerpt. Now, consider this other excerpt:

Example 10.
(s.0) *The supervisor approves the reimbursement or the supervisor reject it.*
(s.1) *If the supervisor approve the reimbursement, the secretary makes the bank transaction.*
(s.2) *Otherwise, the secretary sends a letter to the customer.*

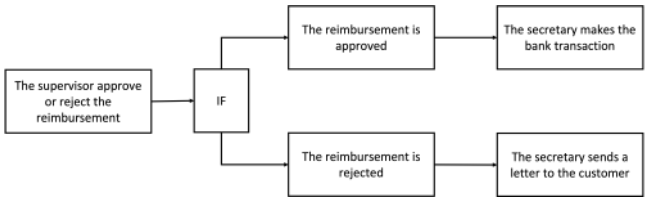

Figure 5: A schematic representation of 9.

This example conveys the same semantics as the previous one. The main difference relies in the fact that the gateway description is repeated in the text. Indeed, the gateway is described in sentence *(s.0)* (without describing any condition specification). The branch conditions are expressed by means of the two activities: approves and reject. Therefore, in cases like the one presented, you do not have to annotate neither the gateway repetition *if...otherwise*, nor the condition specification *the supervisor approves the reimbursement*. Figure 6 presents the schematic representation of this excerpt.

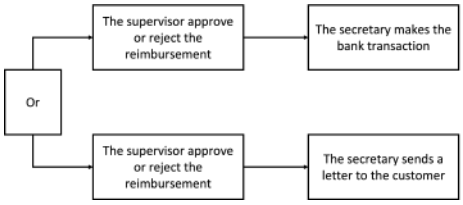

Figure 6: A schematic representation of 10.

***Rule* 12.** *If an activity expression is expressed by means of a phrasal verb in the form verb and one or more following particles, you should annotate the verb and the following particles within a single annotation mark.*

For instance, consider the following example:

Example 11. *The letter is sent back to the customer*

You should annotate the phrasal verb sent back as activity.

***Rule* 13.** *Expression referring to the process execution are not Activity.*

For instance, in the following examples:

Example 12. *The process starts here.*

Example 13. *The process ends.*

In these examples, the expressions: `starts` and `ends` are not annotated as activities since they are not activities. Other possible expressions you may find are: `the After the Process starts`, `the process ends here`, and so on.

***Rule* 14.** *If an activity omits the object the feature Uses is left empty.*

For example, in the following example:

Example 14. *The standard procedure provides the reimbursement.*

the activity `reimbursement` has no object.

***Rule* 15.** *If an activity uses an Activity Data element, link the Activity to the Activity Data adopting the feature Uses. Otherwise, this feature is left empty.*

For instance, consider the following example:

Example 15. *The chef bakes the pizza.*

You should realize the following links:
bakes → Uses → the pizza

***Rule* 16.** *If an Activity has an implicit reference to the Activity Data used, link the explicit mention of the Activity Data to the Activity.*

For instance, consider the following example:

Example 16. *Within 5 working days, the Coordination Unit must conduct the interviews, which are then sent to all Committee Members.*

The Activity `sent` has an implicit reference to the activity data `the interviews`. So, you should link this activity to the explicit mention of its activity data the interviews.

***Rule* 17.** *If an Activity acts on more than one object, link all the Activity Data used (using the feature Uses) to the activity.*

As an example, consider the following example:

Example 17. *The shop sells beautiful bread and warm pizza.*

The activity `sells` uses two objects: `beautiful bread` and `warm pizza`. In this case, you should make two links in the feature *Uses*: one for `beautiful bread` and the other for `warm pizza`. You should realize the following links:
sells → Uses → beautiful bread
sells → Uses → warm pizza

***Rule* 18.** *An Activity can describe the object it uses using a pronoun references. In this case, you should annotate the pronoun as Activity Data and then link the pronoun to the Activity.*

For instance, consider the following example:

Example 18. *The baker bakes beautiful bread and sells it.*

The Activity `sells` acts on the pronoun `it`. In this case, you should link `it` to `sells`.

***Rule* 19.** *If an Activity omits further specification details, the feature Further Specification is left empty. Otherwise, you should link the activity to its Further Specification details (annotated in the Further Specification layer).*

This rule is better explained in Section 6.

***Rule* 20.** *If an activity does not describes the actor/-s involved in its execution the feature Roles is left empty. Otherwise, you should link the Actor Performer and/or the Actor Recipient involved to the Activity.*

This rule is explained in Section 7.

5 Activity Data

Definition 2 (Activity Data)**.** *An* Activity Data *object represents the data or the object directly used by an activity.*

In this Section, we provide the annotation rules to annotate the activity object used by an activity. We defined an annotation layer called *Activity Data* to capture this process element. The Behavioral layer is connected to this layer by the feature *Uses*. This means that the link between an Activity to its Activity Data is annotated in the Behavioral layer. This layer has no features; it is used to capture the span of words describing this information.
An activity data describes the information about *what* of an activity. It describes the object an activity acts on. For example, consider the following example:

Example 19. *The office receives an order from the customer.*

The activity receives acts on the object an order. In general, you should annotate only the object of the activity. To clarify the concept, consider this other example:

Example 20. *The office receives an order sent by email from the customer.*

In this case, you should not include the details about the *how* an activity is carried out in your annotation mark. So, you should annotate as activity data the order, leaving out *sent by email* from your annotation mark. Typically, an Activity Data element is described in noun phrases. However, it is not always the case that the annotation mark marks the entire noun phrase. The choice of selecting the entire noun phrase or its subset depends on the situation described in the text. It may happen that the noun phrase describing an Activity Data is not informative. For example, consider the following sentence:

Example 21. *The Legal department prepares everything for the contract.*

In this case, the activity data *everything* is not informative since it does not convey the proper information about *what* is prepared. So, in cases like this, you should annotate the entire noun phrase including all the relevant details to make the Activity Data informative. Therefore, you should annotate everything for the contract.

We try to produce guidelines on how to capture the exact expression used to define it in the text in the rules below.

DISCLAIMER We adopt the term *Uses* to name the link between an Activity and its Activity Data. In BPMN this type of link is called *Association Flow*.

5.1 Annotation Rules

***Rule* 21.** *Usually an Activity Data is expressed by a nominal expression (a noun phrase). Please annotate the entire noun phrase describing the Activity Data. Otherwise, mark the prepositional phrase describing the activity data*

For instance, you should mark an Activity Data as proposed in the following example:

Example 22. *I edit the module.*

In case the activity is expressed by verbs like *to tell*, *to inform*, etc. you are required to annotate the entire prepositional phrase including the preposition, as in the following example

Example 23. *I inform the X Departmenet about the failure.*

***Rule* 22.** *If an Activity Data is mentioned several times in the text, annotate all the mentions of it (in the same way).*

***Rule* 23.** *An Activity Data can be described at different levels of detail. In the annotation please include all the contiguous important details of the object, including contiguous adjectives only if relevant w.r.t. the process described.*

For instance, you should annotate as proposed in the following example:

Example 24. *I create a job description for the vacant position in the IT department.*

Now, consider the situation described in the following example:

Example 25. *The assembling department orders to buy the stock of nails to assemble the product to the purchasing department.*

In this situation, there are two consecutive verbs (*orders* and *to buy*) that make the choice of deciding which is the activity and which is the activity data tricky. As a general rule, you should annotate the first verb as activity and the dependent verb plus the object used by it as activity data. You should annotate this example as:
The assembling department orders to buy the stock of nails to assemble the product to the purchasing department.
cA different situation you may encounter regards cases where there are two consecutive verbs and the first verb is too generic. This kind of situation requires attention since it is not predictable *a-priori* how to annotate them. Consider the following example:

Example 26. *The secretary begins to write the letter.*

The verb *begin* is very generic. Following the discussion above, you should have annotated begin as activity and to write a letter as activity data. This interpretation implies the existence of two activities, one regarding the beginning of the task of *writing the letter* and the other, probably described later in the text, of *ending of writing the letter*. This emphasizes that it is important to represent these two activities. However, in case this is not the meaning the text conveys, or if there is not the ending task described later in the text, the activity to annotate regards only the verb *to write*. therefore, you should annotate write as activity and the letter as activity data.

***Rule* 24.** *Exclude from the annotation mark unnecessary details of the object if they are not relevant to the process described.*

For instance, you should annotate activity data as proposed in the following examples:

Example 27. *I send a complete description report following the guidelines described in the attachment 1 to request further information about...*

What is important to capture in this example are the details of the report (`a complete description report`), not how it was created (`following the guidelines described in attachment 1 to request further information about...`)

***Rule* 25.** *An Activity can act on more than one Activity Data. In this case, you should mark each Activity Data separately and then link them all to the Activity using the feature Uses in the Behavioral layer.*

You should annotate as in the following example:

Example 28. *I send the report, the reimbursement, and the letter to the Support Office.*

In this example, there are three Activity Data used by the Activity `send`: `the report`, `the reimbursement`, and `the letter`. You should link each Activity Data to the Activity as follows:

send → Uses → the report
send → Uses → the reimbursement
send → Uses → the letter

***Rule* 26.** *Exclude from the annotation mark punctuation symbols and conjunctions that are at the boundary of the annotation span.*

Consider the following example:

Example 29. *I send the report and the dataset.*

In this case, you should mark the two activity data as separate objects, excluding the conjunction `and`.

***Rule* 27.** *At the end of the annotation process, any Activity Data must be connected to one or more Activity objects.*

6 Further Specification

In the previous Sections, we provided rules to annotate an Activity and the Activity Data it uses. However, it may happen that some important details of an Activity are left out from annotations. For example, the manner or the mean specification of an Activity is not captured with the previous annotation layers. For instance, consider the following example:

Example 30. *the office sends the report by email.*

In this example the mean by which the report is sent (`by email`) is not captured neither by the Activity `sends`, nor by the Activity Data `the report`. Since the means of sending the report may be an important aspect of the Activity `sends`, we introduce the layer Further Specification to capture them. These details typically participate in the Activity's label construction. However, this is not a rule, since there could be many situations in which a further specification detail is important but it does not participate in the label construction.

For example the expression `by email` in ex.30 could participate in the label construction of the activity `sends` making the activity label `sends the report by email`, or it can be modeled as a sending event in a BPMN diagram. In general, in the Further Specification layer, you should annotate the information about *How* or/and the *Why* an activity is carried out.
This layer has no feature. It serves to capture the span of words that represent a Further Specification of an Activity. The Behavioral layer is connected to this layer by the feature *Further Specification*. This means that the link between an Activity to its Further Specification element is annotated in the Behavioral layer. These types of elements are typically described using prepositional phrases. However, it is not always the case that a prepositional phrase describes a Further Specification element, or that a Further Specification is described only in prepositional phrases. Also, there are cases in which you should not annotate the entire sub-phrase, but you are required to mark only a portion of it. Besides this simple example, we try to produce guidelines on how to capture the exact expression used to define it in the text in the rules below.

6.1 Annotation Rules

***Rule* 28.** *Please include the opening preposition in the annotation mark, if present.*

You should annotate further specification elements as proposed in the following examples:

Example 31. *The database is checked to see whether the table has new records.*

Example 32. *The customer submits a claim by sending in relevant documentation.*

Example 33. *The office report is searched within the database.*

***Rule* 29.** *Include in the annotation mark all the contiguous details of a Further Specification element that you think it is important to capture.*

For instance, consider the following example:

Example 34. *The manager sends the data by email.*

In this example, the expression by email is an important detail about the activity sends. Now, consider this other example:

Example 35. *The manager sends the data immediately.*

In this case, the expression `immediately` is not a Further Specification detail relevant for the Activity sends, therefore it is not annotated as such.

***Rule* 30.** *Exclude from the annotation mark all unnecessary details. These details are typically described in the sub-phrases (of, for example, a prepositional phrase).*

Consider the following example:

Example 36. *The office sends the report by email through an online email service that will check the presence of the virus.*

you should annotate only the important details (by email) and exclude unnecessary details `through an online email service` and `that will check the presence of virus` since they are not relevant to the process described.

***Rule* 31.** *Link each Further Specification element to its Activity using the feature Further Specification of the Behavioral layer.*

Consider the following example:

Example 37. *The office sends the report by email.*

So, you should realize the following connection
sends → Further Specification → by email

***Rule* 32.** *At the end of the annotation process, any Further Specification element must be connected to its Activity.*

7 Actors

An actor is an *Organizational* element of a process model diagram that represents *who* is involved in an activity. In order to capture this information, we introduce the Organizational layer.

We adopt the following definition of actor:

Definition 3 (Actor)**.** *An* Actor *is a person, a department, an organization, or an artificial agent that is responsible for an activity.*

Considering the Behavioral macro category, it is possible to differentiate between endogenous and exogenous activities. Endogenous activities are *performed* within the process owner boundaries (e.g., performed by a relevant process actor). This type of activity has typically one actor, the *Actor Performer*, that is responsible for its execution. Exogenous activities are the ones that can trigger/or that are triggered by another activity/or by an external event that happens and so triggers the activity. This type of activity typically represents a share of information and data between different participants within a process model. For example, a sending activity involves two actors: who send the data (the Actor-Performer) and who receive the data (the actor-recipient). Typically, this type of activity crosses the process owner boundaries and it involves more than one participant. Therefore, we defined two different types of actors based on the role they play in an Activity: the *Actor Performer* and the *Actor Recipient*. The Actor Performer typically represents the activity owner; who is responsible for the Activity that generates a trigger or the data. The Actor Recipient typically represents who is responsible for the activity that receives the trigger or the data.

The Behavioral layer is connected to this layer by the feature *Roles*. This feature has two sub-features (or tags) used to specify the type of relation between an Activity and its process participants: *Actor Performer* and *Actor Recipient*. This means that the link between an Activity to its participant/-s is annotated in the Behavioral layer.

In general, an actor is expressed in noun phrases or using pronoun expressions. However, there could be various situations in which it is not always easy to determine who is the actor, or the actor is followed by unnecessary details,

or the actor is not explicitly mentioned. We try to produce guidelines on how to capture the exact expression used to define it in the text, the rules, and how to link it to Activity, below.

7.1 Annotation Rules

***Rule* 33.** *Annotate the entire noun phrase mentioning the Actor.*

Example 38. *The customer submits the claim*

***Rule* 34.** *Exclude from the annotation mark any further specification details of the mention.*

Consider the following example:

Example 39. *the Coordination Unit at the Town Planning Authority sends the report.*

the further specification details of the actor `at the Town Planning Authority` are unnecessary w.r.t. the process model description, and so, they are excluded from the annotation mark. But, there are cases in which further details are important. For example, consider the following example:

Example 40. *The Coordination Unit of the Company A sends the report to the Coordination Unit of the Company B.*

In this case, it is important to keep the details about the Company and Actor belonging to `of Company A` and `of Company B`. Indeed, if we leave out such details, annotating only `the Coordination Unit`, it results to be impossible to distinguish between the two (different) Actors.

***Rule* 35.** *Exclude from the annotation mark prepositions, punctuation symbols, adverbs, and conjunctions that are at the boundary of the annotation span.*

For instance, consider the following example:

Example 41. *The officer, the chief manager, and the secretary receive the report.*

***Rule* 36.** *Do not annotate mention of an Actor if it is not involved in any activity.*

Consider the following example:

Example 42. *The Coordination Unit is part of the Safety Department. When the Safety Department receives an order by email ....*

In this example, the expression `The Coordination Unit` and the (first mention of) `the Safety Department` are not Actors since they do not participate in any Activity. Only the second mention of `the Safety Department` is annotated as Actor since it corresponds to the Actor-Performer of the activity `receives`.

***Rule* 37.** *If an actor is mentioned several times in the text, annotate all the mentions that are compliant with Rule 36.*

***Rule* 38.** *If many Actors are mentioned in a list, annotate each Actor with an annotation mark.*

You should annotate each Actor with a single annotation mark, as proposed in the following example

Example 43. *The Coordination Unit sends the email to the customer, to the Support Office, and to the Safety Department.*

***Rule* 39.** *If two actors stand in exclusive relations, consider them as a single actor. You should mark both actors within a single annotation mark.*

For instance, you mark the following example as shown:

Example 44. *When an employee or the secretary receive an email they must inform the supervisor.*

Since, in cases like this it is not possible to capture the exclusive relation between the two actors an employee and the secretary.

***Rule* 40.** *You should link the closest Actor of an Activity to the Activity.*

***Rule* 41.** *If an Actor is who performs/is responsible for an Activity, it should be linked to the Activity as Actor Performer.*

Consider the following example:

Example 45. *ABC Company ICT Manager compiles the attachment 1 (section 1) and sends it.*

The Actor `ABC Company ICT Manager` is the Actor Performer of the Activity `compiles`. You should realize the following activity-actor relations:

compiles → Roles.Actor Performer → ABC Company ICT Manager
sends → Roles.Actor Performer → ABC Company ICT Manager

***Rule* 42.** *If an Actor is who receives data or a trigger from an Activity, it should be linked to the Activity as Actor Recipient.*

Consider the following example:

Example 46. *ABC Company ICT Manager compiles the attachment 1 (section 1) and sends the compiled form to the Support Unit.*

Regarding the Activity `sends`, we have two actors: `ABC Company ICT Manager` and `the Support Unit`. `ABC Company ICT Manager` is the Actor-Performer of the Activity since he is responsible for sending the data. `the Support Unit` is the Actor Recipient of the Activity since he is who receives the data sent. Therefore, you should create the following activity-actor relations:

compiles → Roles.Actor Performer → ABC Company ICT Manager
sends → Roles.Actor Performer → ABC Company ICT Manager
sends → Roles.Actor Recipient → the Support Unit

***Rule* 43.** *At the end of the annotation process, any Actor must be connected to one or more Activities.*

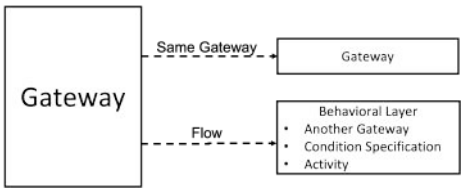

Figure 7: Layers Relations.

8 Gateways

A gateway represents a control flow point in a process model where the process flow is *split* into alternative paths, or whether multiple possible paths are *merged* (aka joined).

In this guideline, we consider two types of gateways: exclusive (XOR) gateway and parallel (AND) gateway. We decided to not cover the *or gateway* in this guidelines since it may be problematic in many ways when modeling and also to annotating it. Moreover, an *or gateway* is less frequent and was not present in the documents we had the opportunity to analyze.

Gateways describe the behavior of a process model and so they are captured in the Behavioral layer of the annotation schema. The relations this process element makes with the other process elements are illustrated in Figure 7. A Gateway element makes a connection to itself by the feature *Same Gateway*. This relation allows us to connect together the various parts of a gateway description. Indeed, a gateway description may be fragmented over multiple sentences. The gateway element is connected to other behavioral elements by the *Flow* relation. Since a gateway is a Behavioral element, it has all the features inherited from the Behavioral layer. As happened for the previous process elements, not all the features are always required. The choice of which feature fill must depend on the type of the gateway and the situation described in the text. We explain which features need to be filled, how to capture the exact expression used to define it in the text, and the rules in the specific section of the AND Gateway (Sec.8.1), of the XOR Gateway (Sec.8.2) and the Condition Specification associated to a specific branch of an XOR Gateway (Sec.8.3).

Moreover, besides the well-defined theoretical definitions of Gateways, the situation one may find in the text is different. Indeed, in process description a split or a merging point can be explicitly described or there could be an implicit reference to them. For example, it is common to find an explicit description of the split point of the process flow and an implicit (omitted) description of its merging point.

8.1 AND Gateway

Definition 4 (AND Split)**.** *An AND split or parallel split is a point in a process model where a single thread of control splits into multiple threads of control which are executed concurrently. [3] (pp. 128-129)*

Definition 5 (AND Join)**.** *An AND join is a point in a process model where multiple concurrent threads converge into one single thread of control. It is an*

assumption of this pattern that each incoming branch is executed exactly once. [3] (pp. 129)

From the definition above, a And Gateway describe a point in the process model in which several activities are executed in parallel and then the paths are (usually) merged. Whereas a split point is always explicitly described (by means of some expressions) in the text, this could be not so for the merging point. Indeed, in many cases, the merging point is inferred from the textual semantics, but no explicit expressions are present in the text. Due to this variability, we decided to not annotate merging points, even when they are explicitly described in the text. We capture the annotation of a merging point by means of *Flows* relations among behavioral elements in the Behavioral layer (described in Sec. 9). Indeed, an Activity with multiple incoming flows acts as a merge node. Annotating an And Gateway is not an easy task since it is impossible to define rigorous rules on the words you should mark as Gateway. We try to provide guidelines and rules below.

Particular attention should be paid to the word *and* since this word is commonly used in text to convey either the meaning of simultaneity, or a chronological order. for instance, consider the following examples:

Example 47. *The baker kneads the dough and he checks the temperature of the oven.*

Example 48. *The baker kneads the bread and he sells it in his shop.*

In ex.47, the *and* provide the meaning of simultaneity. The two activities are executed in parallel. To say in other words, the *The baker kneads the dough meanwhile (and) he checks the temperature of the oven.* The two parallel activities belong to two branches of an AND gateway; they are two independent tasks executed concurrently. In most cases, there are often other semantic markers in the sentence (such as *simultaneously*, *at the same time*, etc..) that help the reader to understand this particular meaning for the word *and*.

The use of *and* in ex.48 is different since it provides the meaning of chronological order. To say in other words, *The baker bakes the bread and he (then)sells it in his shop.* These two activities are executed one after the other; these two activities are linked in a sequential relation.

8.1.1 Annotation Rules

***Rule* 44.** *Annotate all the contiguous words that provide the semantics of concurrency between two or more different paths.*

Consider the following example:

Example 49. *The doctor checks the database, and in the meantime the patient compile the form.*

In the example above, the span of words that conveys the meaning of concurrency is `in the meantime`, not solely `meantime`. The same rules apply in these examples:

Example 50. *The officer sends the report. Meanwhile the Chief checks the database.*

Example 51. *The doctor checks the database while the patient compiles the form.*

Example 52. *The office employee informs the supervisor. In the same time, the customer completes the complaint form.*

Example 53. *The office employee informs the supervisor. In the meantime, the Cashier prints the file.*

***Rule* 45.** *The features: Uses, Roles, Further Specification are always left empty.*

***Rule* 46.** *If the description of a Gateway spans over multiple textual fragments describing the same gateway, you should connect all the gateway fragments using the feature Same Gateway. You should link the first fragment to the second one, the second fragment to the third one, and so on.*

8.2 XOR Gateway

Definition 6 (XOR Split)**.** *An XOR split or exclusive or split is a point in a process model where one of several branches is chosen.[3] (pp. 130)*

Definition 7 (XOR Join)**.** *An XOR join or exclusive or join is a point in a process model where two or more alternative threads come together without synchronization. It is an assumption of this pattern that exactly one of the alternative branches is executed. [3] (pp. 130)*

The same considerations made for the And Gateway are still valid for the Xor Gateway. The major difference relies on the fact that in this case, a process instance must satisfy a specific condition before being allowed to take a particular path (branch) in the process model. The condition that must be satisfied is captured by the *Condition Specification* element of the Behavioral layer. For example, in:

Example 54. *If the customer calls or writes back declining the mortgage, the case is sent to the Office.*

In this example, we marked both the Gateway expressed by the expression `If` and its Condition Specification expressed in `the customer calls or writes back declining the mortgage`. Intuitively, only in those cases in which the customer makes a call and expresses the intention of declining the mortgage or writes a document in which expresses the intention of declining the mortgage, the process flow will go through this branch and the activity `the case is sent to the Office` takes place. In cases in which the branch condition is not satisfied, the activity will not take place, and the process instance will pass through the other branch. Consider the next example:

Example 55. *the customer can either withdraw the contract or confirm it*

In this example, the expression conveying the meaning of a gateway is represented by the two words *either* and *or*. Each of the two gateway expressions describes a branch. In this case, you should link the two gateway expressions by means of the *Same Gateway* feature since both provide a description of the same gateway. Notably, in this example, there are no Condition Specifications. Instead, the example describes the activity of each branch. Therefore, you should pay particular attention when annotating the gateway and its branch's elements. From the example above is clear that a gateway and its branches can be expressed in many unpredictable ways, such as using the construct *either...or*. We

postpone the discussion, annotation guidelines, and rules on how to annotate Condition Specification to the next Section. Here, we try to provide guidelines and rules below only for the Xor Gateway.

8.2.1 Annotation Rules

***Rule* 47.** *Annotate all the contiguous words that provide the semantics of a control point in which the process flow is split in two or more different paths.*

You should annotate Xor Gateway as in the following examples:

Example 56. *if the buyers spends more than 100 Euro then..*

Example 57. *Another reminder is sent and so on until the completed questionnaire is received.*

***Rule* 48.** *Exclude from the annotation mark any condition specification.*

You should perform the annotation of a gateway expression as in the following example:

Example 58. *In case of the customer calls to decline the mortgage, the case is archived.*

You should annotate the gateway as shown, excluding from the annotation mark the Condition Specification `the customer calls to decline the mortgage`.

***Rule* 49.** *The features: Uses, Roles, and Further Specification are always left empty.*

***Rule* 50.** *If the description of a Gateway spans over multiple textual fragments, you should connect all the gateway fragments together using the feature Same Gateway. You should link the first fragment to the second one, the second fragment to the third one, and so on.*

Consider the following examples:

Example 59. *In case of rejection, the employee receive an email; otherwise the employee does something.*

Example 60. *If no new records are found, then the chief officer should check the database. If new returns exist, then register them into the database.*

In both examples, the two gateway fragments annotated describe two branches of the same gateway. The description of the gateway spans over multiple sentences. In cases like the one presented, you should connect all the fragments referring to the same gateway together as follows:

(ex.59) `In case of` → Same Gateway → `otherwise`

(ex.60) `If` → Same Gateway → `If`

***Rule* 51.** *If the merging point of a gateway is explicitly described in the text, DO NOT annotate it as a gateway.*

Consider the following example:

Example 61. *If new returns have been filed, reconcile them with the existing account defaulters table.*
At this point the employee sends the data to the supervisor.

In this example, the merging point is expressed by `At this point`. However, you should not annotate it since it is captured using flow relations.

8.3 Condition Specification

A Condition Specification element specifies the condition that must be satisfied by a process instance in order to be allowed to take a particular branch of a control point. Therefore, this process element is always bound with an Xor Gateway. Annotating Condition Specifications is not an easy task since it is impossible to define rigorous rules on the words you should mark. Different from other process elements in which a subphase within a sentence describe a process model element, a Condition Specification is not expressed only by mean of noun phrases, prepositional phrase, or subordinate sentence. There are many unpredictable ways to describe a branch condition. In general, a Condition Specification could be described by the mean of a noun or adjective that expresses the condition, as in the following examples.

Example 62. *If rejected then the case is sent to the supervisor.*

Example 63. *If the customer spends more than 100 Euro then..*

Example 64. *A reminder is sent until the questionnaire is not fulfilled.*

We try to provide basic guidelines and rules below to annotate this element below.

8.3.1 Annotation Rules

***Rule* 52.** *Annotate with a single annotation mark all the contiguous words that provide the semantics of the condition that a process instance must satisfy to be allowed to take a particular branch.*

You should annotate a Condition Specification as presented in the following example:

Example 65. *In case of the amount is greater than 500 Euro, then the report is sent to the supervisor for approval.*

***Rule* 53.** *If the text describes more than one Condition Specification, do not break the annotation. Annotate all the contiguous conditions with a single annotation mark.*

You should annotate a Condition Specification as in

Example 66. *In case that the customer calls or writes back declining the mortgage, the case is sent to the Office.*

***Rule* 54.** *At the end of the annotation process, any condition Specification element must be connected to one Gateway only.*

9 Flows

A *Flow* object captures the execution flow among Behavioral elements. In particular, we consider a single type of flow, the **sequential flow** relation.

Definition 8 (Flow)**.** *A* Flow *object is a directional connector between activities in a Process. It defines the activities' execution order.*

Definition 9 (Sequence Flow)**.** *A* Sequence Flow *object defines a fixed sequential relation between two activities. Each Flow has only one source and only one target. The direction of the flow (from source to target) determines the execution order between two Activities.*

For example, if activity A precedes activity B, then there is a sequential relation (so, a sequential flow object) from A to B. The goal is to reconstruct the process execution order of the behavioral elements. For instance, pretend there are three activities: A, B, and C. B is executed after A; C is executed after B. You should connect A to B and B to C.

A flow relation is a link between two or more Behavioral elements. This relation is captured in the feature *Flows* of the Behavioral layer.

DISCLAIMER In the extent to reduce the annotation workload and in the meantime reduce the possibility of making errors, we decided to keep the tag *Sequence Flow* to connect a gateway expression to the branch's condition (called in this guidelines *Condition Specification*), even if a more appropriate name for this type of flow should be *conditional flow*.

Annotating Flows is a difficult task. We provide guidelines and rules that you should follow below; then we provide hands-on examples that will help you understand how to annotate flows in different situations in the next section. Before presenting the annotation guidelines about flows, we briefly discuss why we do not annotate Start and End events.

Definition 10 (Start Event)**.** *A Start Event indicates where a particular process will start*[3]*.*

Definition 11 (End Event)**.** *It indicates the temporary end of a process*[3]*.*

As we said in the previous Sections, we do not annotate any *start event* or *end event*, even when they are explicitly described in the text. We consider any activity or any gateway that has no incoming flow as it is preceded by an implicit start event. In the same way, we consider any activity, gateway, or condition specification without any outgoing flow as followed by an implicit end event. Regarding the end event and the gateway element, the situation is slightly different. We consider an exclusive or split gateway that has only one outgoing flow as having another not-described branch that leads to an end event if the condition of the described branch is not met. Intuitively, a concurrent split gateway has always at least two outgoing flows described in the text.

[3]definition taken from www.businessprocessglossary.com

9.1 Annotation Rules

***Rule* 55.** *Connect all the Behavioral elements following the sequential order described in the text.*

***Rule* 56.** *The relation arrow matter. Therefore, a sequential relation from A to B is different from a sequential relation from B to A.*

***Rule* 57.** *At the end of the annotation process, all the behavioral elements must be connected together. If a behavioral element has no connection, it is not an element of the process described. Thus, the annotation mark should be deleted.*

9.2 Hands-on examples

9.2.1 Annotate Sequence Flows between Activities

In this section, we illustrate how to annotate a Sequential Flow between two Activities. Consider the following example:

Example 67. *The baker bakes the bread and sells it.*

The example describes two Activities that stand in a sequential order: `bakes` and `sells`. Being in a sequential relation means that the first Activity proceeds with the second Activity. In order to annotate the sequential flow, you must connect `bakes` to `sells` using the *Sequence Flow* relation (by filling in the feature *Flow*). A representation of the example, in BPMN, is shown in Figure 8

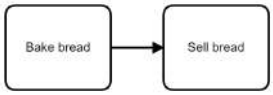

Figure 8: Example of Sequential Flow connection between the two activities described in ex.67.

9.2.2 Annotate Sequence Flows with Gateways

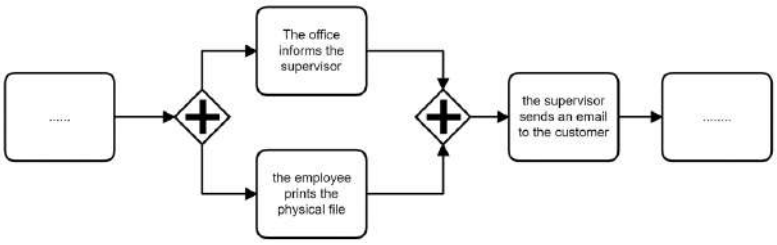

Figure 9: BPMN representation of ex.68.

Annotating Flows with a gateway is a bit more complicated than annotating flows among activities.

And Gateway.
Consider the following example:

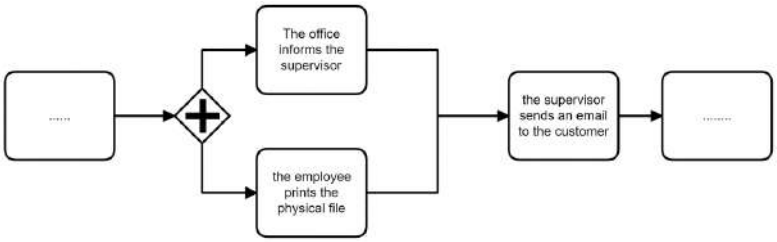

Figure 10: Representation of the connections that should be realized when annotating the flows of ex.68.

Example 68. *... The office informs the supervisor. In the meantime, the employee prints the physical file. Then, the supervisor sends an email to the customer. ....*

The two activities `informs` and `prints` are executed in parallel. Adopting a BPMN terminology, each of the two parallel activities belongs to a branch of an And Gateway. Then, at the end of these two activities, the activity `sends` is executed. This means that there is an implicit merging point before the third activity. Figure 9 shows a BPMN representation of this textual fragment. Again, as we stated above, we do not annotate merging points even when they are explicitly described in the text. The strategy we adopt to connect the elements belonging to an And Gateway together is shown in Figure 10. We start by connecting the gateway expression (In the meantime) to each activity (informs and prints). These two connections define the two branches of the control flow point. Then, we connect each of the two activities to the third activity (sends). Adopting this mechanism, we capture the presence of a merging point. Therefore, you should realize the following connections:

In the meantime	→	Flows.Sequence Flow	→	informs
In the meantime	→	Flows.Sequence Flow	→	prints
informs	→	Flows.Sequence Flow	→	sends
prints	→	Flows.Sequence Flow	→	sends

A representation of the example, in BPMN, is shown in Figure 9 and the annotation representation is proposed in Figure 10

Xor Gateway.

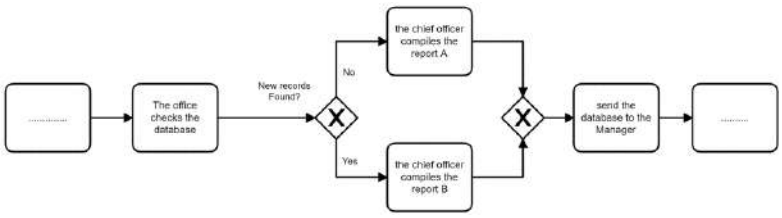

Figure 11: BPMN representation of ex.69.

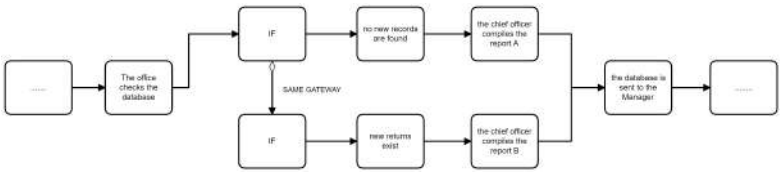

Figure 12: Representation of the connections that should be realized when annotating the flows in ex.69.

Example 69. .... *The office checks the database. If no new records are found, then the chief officer compiles the report A. If new returns exist, then the chief officer compiles the report B. At this point, the database is sent to the Manager.* ....

The example describes a typical case of an exclusive gateway with two conditional statements: `no new records are found` and `new returns exist`. Therefore, only one activity is executed by a process instance: `compiles the report A` or `compiles the report B`. Then, at the end of these two branches, the two alternative paths are merged together and then connected with the activity `sends`. Figure 11 shows a BPMN representation of this textual fragment. A schematic representation of the strategy we adopt to connect elements of Xor Gateway together is shown in Figure 12. We start by connecting each gateway expression (If) to its Condition Specification. We connect each Condition Specification to its activity, and then we connect the two mentions of the same gateway together adopting the Same Gateway relation. Finally, we connect each activity of each branch to the third activity (sends). Adopting this mechanism, we capture the implicit merging point correctly. Therefore, you should realize the following connections:

checks	→	Flows.Sequence Flow	→	(*i*) If
(*i*) If	→	Flows.Sequence Flow	→	no new records are found
(*ii*) If	→	Flows.Sequence Flow	→	new returns exist
(*i*) If	→	Flows.Same Gateway	→	(*ii*) If
no new records are found	→	Flows.Sequence Flow	→	compiles (the report A)
new returns exist	→	Flows.Sequence Flow	→	compiles (the report B)
compiles (the report A)	→	Flows.Sequence Flow	→	is sent
compiles (the report B)	→	Flows.Sequence Flow	→	is sent

A representation of the example, in BPMN, is shown in Figure 11 and a schematic representation of the annotation you should realize is presented in Figure 12

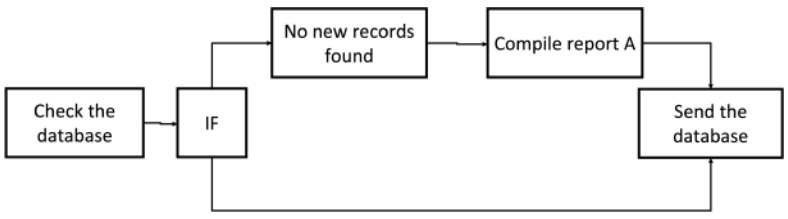

Figure 13: A representation of the sequence flow relations among the behavioral elements described in ex.70.

A different situation one may encounter is when the text specifies only a branch of the gateway. Consider the following example:

Example 70. *.... The office checks the database. If no new records are found, then the chief officer compiles the report A. At this point, the database is sent to the Manager. ....*

In this situation, only one branch of the gateway is described in the text. Its alternative branch presents no activity. Therefore, correctly capturing the meaning described, along the connection from the gateway through the branch described, you should also realize a direct connection from the gateway to the next common activity. The connections you should realize are the following:

checks	→	Flows.Sequence Flow	→	If
If	→	Flows.Sequence Flow	→	no new records are found
no new records are found	→	Flows.Sequence Flow	→	compiles
compiles	→	Flows.Sequence Flow	→	is sent
If	→	Flows.Sequence Flow	→	is sent

A BPMN representation of this semantics is presented in Figure 13

10 The complete annotation procedure - Best Practices

As we said in the introduction, annotating a document is a subjective task. Here, we present our best practices to help you start the annotation process.

1. Start the annotation procedure identifying all the activities described in the text (Behavioural layer).
2. For each Activity, identify and connect the Activity Data, and the Actors (performer and recipient), if present.
3. For each Activity identify and connect its Further Specification, if present.
4. Identify the Gateways in the text, if present.
5. For each XOR Gateway, mark the corresponding Condition Specification, if present, and then link it to the gateway.
6. Link all the behavioral process elements together (using Sequence Flow relation) and provide the overall process logic described.
7. At the end of the annotation process, all the process elements marked in the text are connected together.

11 How to Annotate Process Description using Inception - A quick user manual

This is a quick user manual not meant to be completed[4]

[4]The complete manual can be found here: https://inception-project.github.io/releases/20.2/docs/user-guide.html

Figure 14: Inception Interface.

Figure 15: Inception Interface.

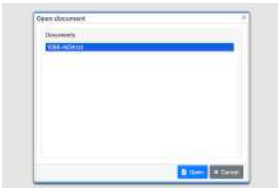

Figure 16: Open a Document to Annotate.

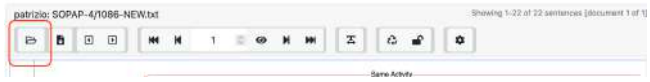

Figure 17: Open another Document to Annotate.

Open a document. After the login, Inception shows you an interface like the one presented in Fig.14. On the left end side, you can select the annotation project you participate in. Now you can access the project by selecting *Annotation* on the left end side menu of the screen, as shown in Fig.15.

At this point, Inception shows you a menu, like the one presented in Fig.16 where you can choose the document you want to annotate. As you can see, opening a document to annotate in Inception is very simple. If you finished annotating a document, or you just want to annotate another document, you need to press the *open document button* (the button highlighted in Fig.17) and select the file you want to annotate.

Annotation Interface. The annotation interface is very intuitive. In the upper toolbar, you can navigate the document (highlighted in blue in Fig.18) In the upper right corner (highlighted in red in Fig.18) you can select the layer you want to annotate. If you make an error, don't worry. You can always delete the erroneous annotation mark, by clicking the delete button on the right of the screen. It may happen that Inception does not show you all the lines of the text because the text is too long. You can know which set of lines is shown by looking at the toggle bar in the upper center side of the interface (blue box of fig.18). If you want Inception to show more lines, you can change the line settings. To do so, click on the gear button (highlighted with a purple box in

the figure).

To export the annotation you made in a file, just click on the second button (highlighted with a green box in the figure) and select the format you prefer.

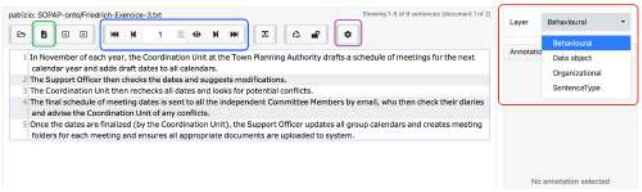

Figure 18: Open a Document to Annotate Button.

Performing a Token Level Annotation. In order to perform a Token-Level annotation, select the entire word/-s you want to annotate. The span of words you selected is now annotated with the layer you selected before. To perform an annotation that spans over multiple words, select all the words with a single selection operation.

Performing a Link. Creating a link between two elements is very easy. Pretend you want to link an Actor-Performer to an Activity. First, you should select the Activity you want to link by a single click on the annotation mark. Navigate through the features of the activity and click on Actor Performer. At this point the box changes color (from white to orange) as shown in Fig. 19. Now perform a single click on the annotation mark of the Actor you want to connect. Et voila', the link is created!

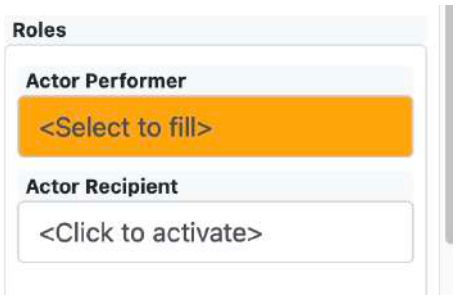

Figure 19: Performing a link between Actor and Activity.

In the case where you need to link more than one Actor-Performer to an Activity, you can create as many links as you need. To do so, instead of clicking on the box, click on the combo box (as shown in fig20), select the type of link you want to make (Actor-Performer in this case), and then select the actor you want to link. The same procedure applies for every link feature (Coref. Mention, Same Gateway, Activity Data, and so on...)

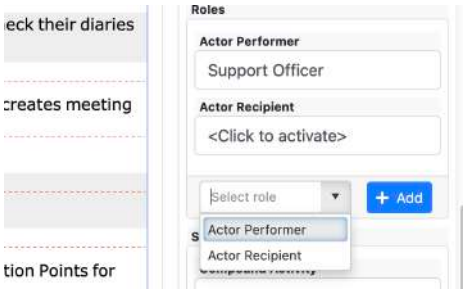

Figure 20: Add an extra actor link.

12 Layers and Tagsets

12.1 Layers

- Behavioral
- Activity Data
- Further Specification
- Organizational

12.2 Behavioral - Process Element Type Tagset

- Activity
- AND Gateway
- XOR Gateway
- Condition Specification

12.3 Behavioral - Activity.Uses Activity Data Tagset

- Uses

12.4 Behavioral - Activity.Roles Tagset

- Actor Performer
- Actor Recipient

12.5 Behavioral - Activity.Flows Tagset

- Sequence Flow

12.6 Behavioral - Activity.Further Specification Tagset

- Further Specification

12.7 Behavioral - Gateway.Same Gateway Tagset

- Same Gateway

References

[1] G. Adamo, Chiara Di Francescomarino, and Chiara Ghidini. Digging into business process meta-models: A first ontological analysis. *Advanced Information Systems Engineering*, 12127:384 – 400, 2020.

[2] Greta Adamo, Stefano Borgo, Chiara Di Francescomarino, Chiara Ghidini, Nicola Guarino, and Emilio M. Sanfilippo. Business processes and their participants: An ontological perspective. In *Proceedings of the 16th International Conference of the Italian Association for Artificial Intelligence (AI*IA 2017)*, volume 10640 of *Lecture Notes in Computer Science*, pages 215–228. Springer International Publishing, 2017. ISBN 978-3-319-70169-1.

[3] Mathias Weske. *Business Process Management - Concepts, Languages, Architectures, 2nd Edition*. Springer, 2012. ISBN 978-3-642-28615-5.