A Data-Centric Approach to Design Resilient-Aware Process Models in BPMN

Simone Agostinelli, Francesca De Luzi, Umberto di Canito,
Jacopo Ferraro, Andrea Marrella, and Massimo Mecella

Sapienza Università di Roma, Rome, Italy
`name.surname`@uniroma1.it

Abstract. The widespread diffusion of Internet-of-Things (IoT) technologies is prompting organizations to rethink their business processes (BPs) towards incorporating the data collected from IoT devices directly into BP models for improved effectiveness and timely decision making. Nonetheless, IoT devices are prone to failure due to their limitations in terms of computational power and energy autonomy, leading to compromise the availability and quality of the collected data, with the risk to prevent the correct execution of the entire BP. To mitigate this issue, resilience is a feature that any data-aware BP should support at design-time, by focusing on the role of available – as an alternative to unreliable – data as a resource for increasing BP robustness to failures. In this paper, we formalize an approach for designing and evaluating resilient-aware BP models in BPMN (Business Process Modeling and Notation) through a maturity model that takes into account their degree of awareness through levels of resilience, which can be computed using the provided formalization. In addition, we show how to extend the metamodel of BPMN 2.0 to address the proposed resiliency levels, and we investigate the feasibility of the approach through a user evaluation.

1 Introduction

With the widespread diffusion of Internet-of-Things (IoT) technologies and the exponential growth of generated data, it is becoming crucial for organizations to rethink their business processes (BPs) towards incorporating the data collected from IoT devices directly into BP models for improved effectiveness and data-driven decision-making [8]. For instance, in the logistics domain, IoT devices provide real-time monitoring of goods transportation in terms of their position or state (e.g., temperature, humidity, etc.), enabling the underlying BPs to optimize their operational efficiency. Nonetheless, when a BP becomes *data-aware*, there are also some side effects in terms of BP reliability. Since IoT devices are prone to failure due to their limitations in terms of computational power and energy autonomy, the risk exists that they might deliver data of low quality or stop working without any previous notice [10], preventing the correct BP execution.

  In this context, a proper design of *resilient* BPs becomes fundamental. Resilience concerns the "*ability of a system to cope with unplanned situations in*

order to keep carrying out its mission" [3]. Satisfying resilience requirements has been often considered as a run-time issue. According to [14,9], many approaches have been proposed to keep BPs running even when some unplanned exceptions occur at run-time, by implementing ad-hoc countermeasures during the execution stage of the BP life-cycle. However, this requires to know precisely where potential mistakes can manifest in the BP. This information, if not explicitly documented in the BP model, may lead to a defective implementation of compensatory strategies for such mistakes. As BP models can explicitly mark and indicate data elements involved in the BP, we can pinpoint the resiliency issues that BP might suffer directly at design-time. This means a shift of focus from *what to do* in case of failures to *what may be affected* when a failure occurs.

The goal of this paper is to provide an approach for designing and evaluating resilient-aware BP models where data are considered as "first class citizens", by driving the improvement of resilience to reduce the possible impact of failures caused by missing/unreliable data due to improper human behavior and/or IoT device errors. Specifically, we introduce a rigorous formalization of the approach that is based on assessing at design-time how available data re-definitions can possibly be exploited to design viable alternatives in the BP model to make it more resilient at run-time. In this direction, a maturity model for resilience awareness is proposed, based on a modeling notation extending BPMN (ISO/IEC 19510:2013 - Business Process Modeling and Notation). The maturity model is organized in five resiliency levels, which can be computed using the provided formalization and allow BP designers to model at an increasing degree of detail how data should be defined to have resilient by-design BP models. In addition, to capture the novel resiliency constructs introduced by our approach, we propose an extension to the BPMN 2.0 metamodel [12] that was exploited to develop a tool, called RES-BPMN, implementing our approach. Finally, we present the results of a user evaluation performed to study the feasibility of the approach.

The rest of the paper is organized as follows. After a discussion of the related work in Section 2, in Section 3 we introduce the main concepts of the BPMN notation and we present a motivating running example. Section 4 specifies the proposed maturity model and the resiliency levels. In Section 5, we show how to extend the metamodel of BPMN to address the resiliency levels. Finally, in Section 6, we investigate the feasibility of the approach and provide a critical discussion about its general applicability, by tracing future work.

2   Related Work

Resilience engineering has its roots in the study of safety-critical systems [6], which aim at ensuring that organizations operating in turbulent settings attain high levels of safety despite a multitude of emerging risks and complex tasks. In the BPM (Business Process Management) field, the concept of resilience has been mainly tackled through the notions of BP *flexibility* [14] and *risk-aware BPM* [20]. Research on BP flexibility has focused on four major needs to make BPs robust to business changes, namely *(i) variability* [15], *(ii) looseness* [1], *(iii)*

adaptation [9], and *(iv) evolution* [4]. However, the ability to deal with changes makes BP flexibility a required, but not sufficient, means for building resilient BPs. While BP flexibility produces "reactive" approaches that deal with exceptions at design-time by incorporating remedial strategies into the BP model, or at run-time if any "known" disturbance arises, BP resilience requires "proactive" techniques accepting and managing change "on-the-fly" rather than anticipating it, to enable a BP to address new emerging and unforeseeable changes with the potential to cascade [11]. On the other hand, while relatively close to the concept of risk-aware BPM, which evaluates operational risks on the basis of historical threat probabilities, resilient BPM shifts attention to the "realized risks" and their consequences, to improve risk prevention and mitigation.

The amount of research works directly addressing BP resilience is quite limited. Among the most relevant, the work of Antunes [2] focuses on developing a set of services integrating resilience support in BPM systems, including detection, diagnosis, recovery and escalation. The approach of Zahoransky [23] investigates the use of process mining to create probability distributions on the time behavior of BPs, which are used as indicators to monitor the resiliency level at run-time and indicate countermeasures if the level drops. The work [22] provides a framework and a set of measures based on the analysis of previous BP executions to evaluate BP resilience. Finally, in our previous work [13], we developed a conceptual approach coupled with a maturity model to build multi-party declarative BPs using OMG CMMN (Case Management Model and Notation).

If compared with the aforementioned papers, in this paper we rigorously formalize a maturity model through BPMN to build resilient-aware BP models at design-time by focusing on the reliability of data exchanged within the BP, which is an aspect neglected in the literature. This makes our approach specifically targeted to those BPs that require data awareness for their execution. While data-aware BPM is a highly debated topic in the BPM literature (see [17] for a summary), and it is considered as a major requirement to integrate BPM with IoT technologies [8], here we do not develop a new approach to integrate data into BP models. Conversely, we exploit (and slightly extend) the data features available in BPMN to handle generic BP descriptions that could be immediately implemented via customary BPMN technologies. In a nutshell, our target is to provide a means for evaluating in advance the impact of data-driven disturbances on the BP and improving BP resilience to failures.

3   Business Process Modeling Notation

BPMN provides a standard graphical notation for BP modelling, with an emphasis on control flow. It essentially defines a flowchart incorporating a range of diverse components, including *activity nodes*, denoting business events or items of work performed by humans or software applications, and *control nodes* capturing the flow of control between activities. Activity nodes and control nodes can be connected by means of a flow relation in almost arbitrary ways. BPMN also enables to represent the information flowing through the BP, such as documents,

Event	○ Start Event	✉ Start Message Event	✉ Catch Int. Message Event	✉ Throw Int. Message Event	○ End Event
Activities and Data Elements	Activity	Sub-Process	Data Object	Data Store	
Gateways	XOR Merge Gateway	XOR Split Gateway	Parallel Join Gateway	Parallel Fork Gateway	
Flow Relations and Activity indicators	Sequence Flow	Association Relation	User Task	Manual Task	Service Task

Fig. 1. A core subset of BPMN modeling elements

e-mails and other objects that are read or updated by means of dedicated *data elements*. As shown in Fig. 1, we take into account a (large) subset of BPMN elements including the data and control flow components considered in this paper. Hereafter, we describe the syntax of a BP model defined with such components.

Definition 1 (BP model). *A BP model specified in BPMN is a tuple $\mathcal{N} = \langle \mathcal{O}, \mathcal{A}, \mathcal{G}, \mathcal{E}, \mathcal{F}, \mathcal{C}, Cond, \mathcal{D}, \mathcal{T}_{IN}, \mathcal{T}_{OUT} \rangle$, where:*

- *$\mathcal{O}$ is a set of flow objects, which can be partitioned into disjoint sets of activities $\mathcal{A}$, gateways $\mathcal{G}$ and events $\mathcal{E}$;*
- *$\mathcal{A}$ is a set of activities, which can be atomic (i.e., tasks) or sub-processes;*
- *$\mathcal{G}$ is a set of gateways, which can be partitioned into disjoint sets of parallel gateways $\mathcal{G}_P$ for creating/synchronizing concurrent sequence flows, and XOR decision gateways $\mathcal{G}_R$ for selecting/joining a set of mutually exclusive alternative sequence flows based on data-driven conditions;*
- *$\mathcal{E}$ is a set of events, which can be partitioned into disjoint sets of start events $\mathcal{E}_s$, throw intermediate events $\mathcal{E}_i^t$ (e.g., a message that is sent) or catch intermediate events $\mathcal{E}_i^c$ (e.g., a message that arrives), and end events $\mathcal{E}_e$;*
- *$\mathcal{F} \subseteq (\mathcal{O}) \times (\mathcal{O})$ is the sequence flow relation for connecting flow objects;*
- *$\mathcal{C}$ is a set of possible conditions that evaluate to true or false.*
- *$Cond : \mathcal{F} \cap (\mathcal{G}_R \times \mathcal{O}) \to \mathcal{C}$ is a function that maps sequence flows emanating from XOR decision gateways to conditions in $\mathcal{C}$;*
- *$\mathcal{D}$ is a set of data elements, which can be partitioned into disjoint sets of data objects $\mathcal{D}_{ob}$ (i.e., local data flowing through the BP) and data stores $\mathcal{D}_{st}$ (i.e., persistent databases that can be queried/updated by BP activities/events);*
- *$\mathcal{T}_{IN} \subseteq (\mathcal{D}_{ob} \cup \mathcal{D}_{st}) \times (\mathcal{A} \cup \mathcal{E}_e \cup \mathcal{E}_i^t)$ is the input association relation used to link data elements to activities, end events or throw intermediate events.*
- *$\mathcal{T}_{OUT} \subseteq (\mathcal{A} \cup \mathcal{E}_s \cup \mathcal{E}_i^c) \times (\mathcal{D}_{ob} \cup \mathcal{D}_{st})$ is the output association relation used to link activities, start events or catch intermediate events to data elements;*

Without losing generality, we assume the behavior of BP models specified in BPMN to be ruled by the semantics described in [5].

3.1   Running Example

An example of a BP model is shown in Fig 2. It represents a BP of a smart distribution centre that exploits the data collected by smart devices to perform quality control over perishable food products before distributing them in grocery shops. This BP is part of a real-world case study presented in [21], which we have extended adding the information about the data exchanged during the BP. The anatomy of the BP, which starts when a new pallet of products is delivered to the distribution center with a truck's container, is as follows:

– First, a quick check of the products' quality parameters (level of firmness, color and possible damages) is performed employing an automated optical sorter and by human operators through a visual analysis.
– Secondly, a sensor installed in the truck's container scans the pallet labels to obtain the products' information (e.g., product name, variety, collection date, etc.). Then, a second sensor captures the air temperature and humidity values related to the transport conditions. This information is recorded in a database and then used to evaluate the quality of the products.
– If the products' quality is considered as not adequate, the pallet is discarded. Conversely, if the quality of the products is good, the pallet is moved in the distribution centre and its storage is registered. The pallet is also temporally placed in a refrigerator room to prevent products' deterioration.
– At this point, a randomly selected sample of products is chosen from the pallet and analyzed in a laboratory to detect the presence of bacteria. If bacteria are detected, an alarm is triggered to indicate that the pallet must be discarded. Otherwise, the shipment procedure of the pallet starts.
– Finally, a last analysis is performed on the quality levels of the products in the pallet (e.g., to check if the firmness is optimal). If the quality is evaluated as not excellent, then the price of the products is dropped and the pallet is moved to a priority area to speed up its shipment and avoid further deterioration. When a truck is ready to start the distribution procedure, the pallet is loaded in a container for its shipment, and the BP completes.

By analyzing the BP behaviour, it is evident that the reliability of the data required to properly run the BP strongly depends on the reliability of the sensors employed for data collection. Any malfunctioning problem in sensors' behavior or connection issue will negatively impact the decision making and, consequently, the execution of the BP. According to [19], seven types of data flow anomalies can be detected in a BP: redundant data, lost data, missing data, mismatched data, inconsistent data, misdirected data, and insufficient data. We notice that all these anomalies can be classified into two main categories of issues related to the *availability* of data and their *quality* degree. In this direction, rather than automatically detecting structural data flow anomalies (e.g., like is investigated in [19]), we propose a maturity model that enables not only to uncover those data whose (un)availability and (low) quality can prevent the BP execution, but also suggests different countermeasures (weighted depending on the nature of the raised issues and the magnitude of their impact) to mitigate these negative effects and improve the BP resilience at design-time.
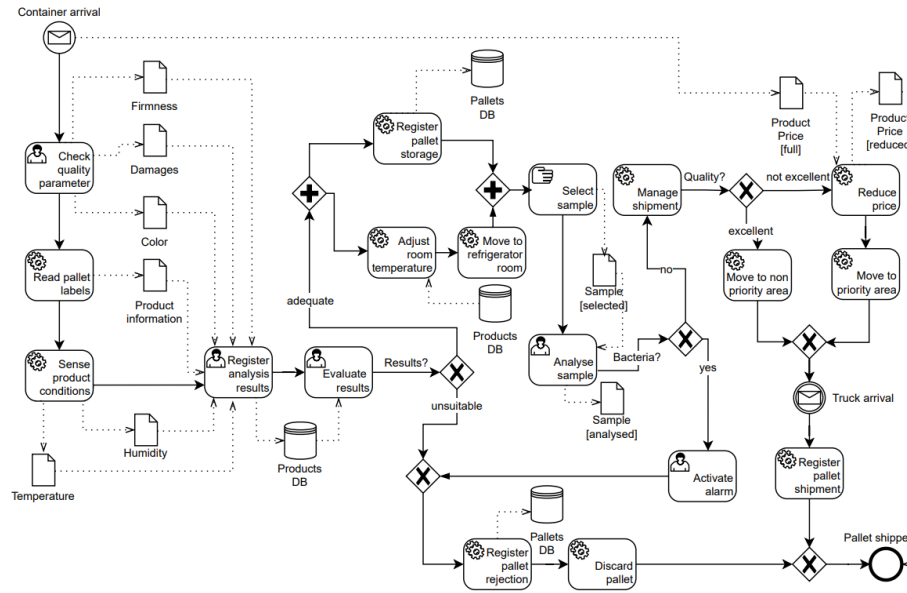
Fig. 2. BP model of the running example

4   Maturity Model

In this section, we present a maturity model with the aim to classify BPs modeled with BPMN in terms of their degree of resilience awareness. As shown in Fig. 3, the maturity model provides 5 levels of resilience awareness, which are defined on the basis of the ability of the BP to adjust itself to the possible unexpected failures with preparedness strategies to increase resilience at design-time. Starting from Level 0 (*No Resilience Awareness*), where resilience is not considered in the BD design, the other levels have been developed based on the three cornerstones of a resilient system as identified by [7]: Early detection (ED), Error tolerant design (ETD) and Recoverability (REC). Specifically, Level 1 (*Failure Awareness*) refers to ED, i.e., the recognition of system's weak signals that could be precursors of abnormal events. Level 2 (*Risk and Quality Awareness*) enforces ED by quantifying the impact of possible failures, and is the precondition for Level 3 (*Alternative Data Awareness*), which implements (ETD) by proposing alternative solutions that enable the system to still function well, but at reduced efficiency and marginally decreased quality. Finally, Level 4 (*Data Recovery Awareness*) refers to REC, which concerns the definition of recovery strategies to recover the system back to a normal state of operations.

4.1   Level 0 - No Resilience Awareness

At this level, a BP is modeled reflecting the desired scenario where it is assumed that all the data elements involved in the BP are available for its correct execution. This is the default situation in BPMN, where the presence of data elements
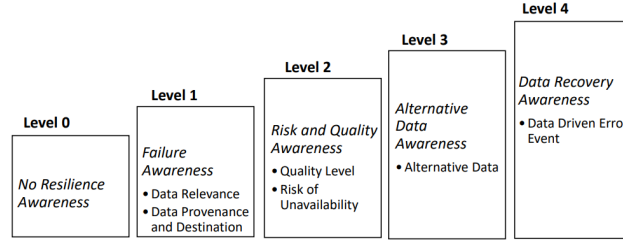
Fig. 3. Maturity Model for designing resilient BPs

is considered as optional in a BP, i.e., a data element is supposed just to provide information details on the BP flow, like happens in the BP of Fig. 2. Thus, at this level, no support is given to resilience and no countermeasure is required.

4.2   Level 1 - Failure Awareness

At this level, the BP is modeled to make it resilient to possible sources of failure due to the *unavailability* of data elements, which might affect one or many activities that are consuming/producing such data. To have a clear map of which relevant data elements may be subject to failures, the BP designer is first required to identify them in the BP model and label them with the tag $\langle true, \mathrm{U}, \mathrm{U} \rangle$. The first tag parameter indicates that the data element will be considered *relevant* for BP execution, i.e., its unavailability may affect the execution of the BP flow objects to which it is connected. In the BPMN metamodel [12], this can be specified by turning the `DataState` parameter to *true* (see Section 5). If a data element becomes relevant, the flow objects that consume that data can not be executed until it becomes available. Similarly, a relevant data element produced by a flow object is checked for availability in output when the execution of the flow object completes. If the data element is not available, an error is thrown. In this paper, we will use the boolean function $State(d)$ that is *true* if a data element $d \in \mathcal{D}$ is relevant. The second and the third tag parameters indicate, respectively, the *quality level* and the *risk of unavailability* of the data element. Both are initially set to U (i.e., UNDEFINED) and have no impact at this level.

Once identified the relevant data elements, to make the BP model compliant with Level 1, the BP designer must first indicate the "provenance" and the "destination" of each relevant data object, i.e., which activity/start event/catch intermediate event produces the data object and which activity/end event/throw intermediate event consumes the data object. This can be done in BPMN exploiting the Association relation. Similarly, for each relevant data store, it must be specified at least a flow object that reads/updates data from/into it. Consequently, a *Level-1 compliant model* can be formally defined as follows:

Definition 2 (Level-1 compliant model). *Let $\mathcal{N} = \langle \mathcal{O}, \mathcal{A}, \mathcal{G}, \mathcal{E}, \mathcal{F}, \mathcal{C}, Cond, \mathcal{D}, \mathcal{T}_{IN}, \mathcal{T}_{OUT} \rangle$ be a BP model. $\mathcal{N}$ is said to be a "Level-1 compliant model" iff, for each $d \in \mathcal{D}$ with $State(d) = true$ one of the following conditions holds:*
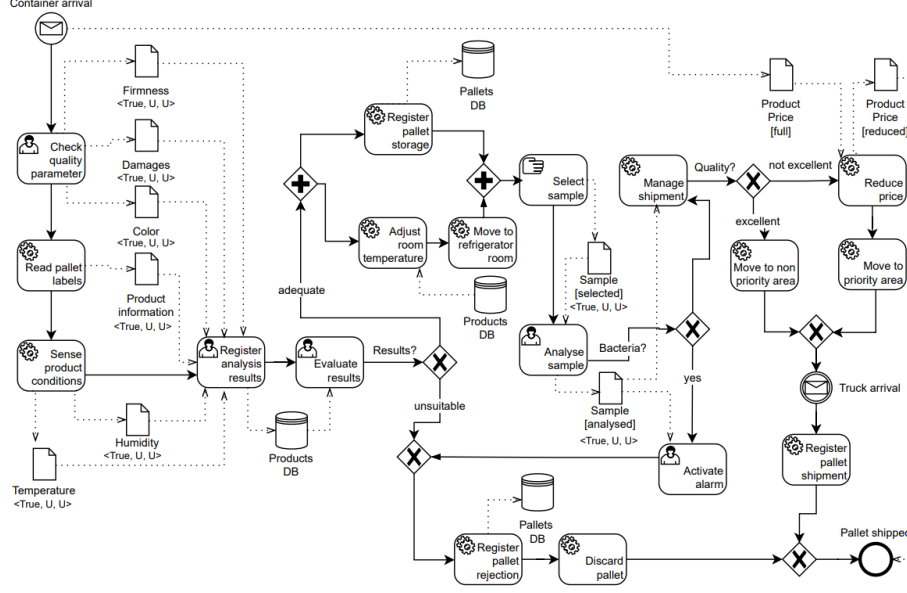
Fig. 4. Level 1 (Failure Awareness) compliant BP model

- $d \in \mathcal{D}_{ob}$, and there exist $x \in (\mathcal{A} \cup \mathcal{E}_s \cup \mathcal{E}_i^c)$, $y \in (\mathcal{A} \cup \mathcal{E}_e \cup \mathcal{E}_i^t)$, $t_i \in \mathcal{T}_{IN}$ and $t_o \in \mathcal{T}_{OUT}$ such that $t_i = \langle d, y \rangle$ and $t_o = \langle x, d \rangle$.
- $d \in \mathcal{D}_{st}$ and there exist $y \in (\mathcal{A} \cup \mathcal{E}_e \cup \mathcal{E}_i^t)$ and $t_i \in \mathcal{T}_{IN}$ such that $t_i = \langle d, y \rangle$, or $x \in (\mathcal{A} \cup \mathcal{E}_s \cup \mathcal{E}_i^c)$ and $t_o \in \mathcal{T}_{OUT}$ such that $t_o = \langle x, d \rangle$.

Let us consider the BP of the running example. To increase the resiliency level of the model we should set as relevant all those data whose unavailability may lead to possible failures, i.e., the data collected by smart devices (e.g., *Firmness*, *Humidity*, *Temperature*, etc.) or obtained after a visual/automated analysis performed by human operators (e.g., *Damages*, *Sample [analyzed]*). Then, to make the BP fully compliant with Level 1, we must check that the relevant data objects are associated to their producer/consumer. Thus, we need to add an output association from the data object *Sample [analyzed]* to the activities *Activate Alarm* and *Manage Shipment*, as shown in Fig. 4. If this data object becomes unavailable or unreliable, the risk exists that the alarm is wrongly triggered or the shipment of products with bacteria is performed with severe effects.

4.3 Level 2 - Risk and Quality Awareness

While at Level 1 the BP designer declares which data elements are likely subject to failures, at Level 2 there is a first attempt to concretely quantify the *quality level* and the *risk of unavailability* associated to such data elements. For the sake of simplicity, in the rest of the paper we assume the quality level/risk of a relevant data element bound to only four discrete values: U - UNDEFINED, L -
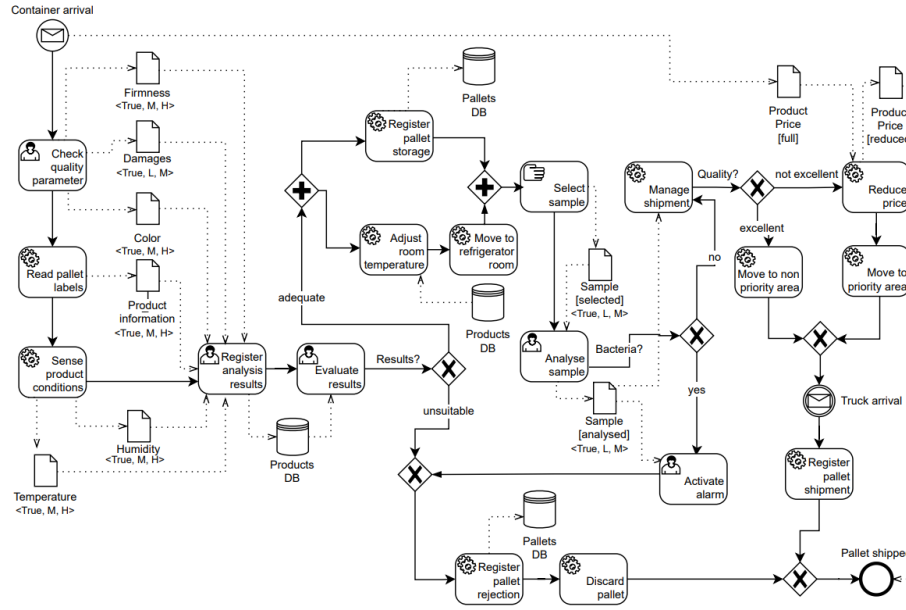
Fig. 5. Level 2 (Risk and Quality Awareness) compliant BP model

Low, M - Medium, H - High. The pair of parameters quality level/risk enables to build a kind of "criticality degree" that supports the BP designer to identify those data elements that might have more impact in case they are unavailable or have a poor quality. Note that, to switch from Level 1 to Level 2, it is required that, for any relevant data element, the quality level/risk are both assigned to a value different from U, i.e., they become objectively quantifiable. Of course, the choice of the values to associate to both parameters depends on the domain under observation. From a formal perspective, we introduce two functions $Quality(d)$ and $Risk(d)$ returning the quality level and the risk of unavailability of a relevant data element $d \in \mathcal{D}$, and we define a *Level-2 compliant model* as follows.

Definition 3 (Level-2 compliant model). *Let* $\mathcal{N} = \langle \mathcal{O}, \mathcal{A}, \mathcal{G}, \mathcal{E}, \mathcal{F}, \mathcal{C}, Cond, \mathcal{D}, \mathcal{T}_{IN}, \mathcal{T}_{OUT} \rangle$ *be a Level-1 compliant model.* $\mathcal{N}$ *is said to be a "Level-2 compliant model" iff, for each* $d \in \mathcal{D}$ *with* $State(d) = true$, *then* $Quality(d) \neq$ U *and* $Risk(d) \neq$ U.

In the case of our running example, many data objects are the results of activities performed automatically through the support of smart sensors supported by sophisticated software. For example, the first quality check involves the use of an optical sorter to measure the firmness of the products contained in the pallet and detect their color. Similarly, other sensors installed in the pallet or in the truck container allow for a precise detection of products' information, temperature and humidity ((H)igh data quality). However, the electronic components of these devices are subject to deterioration due to their continuous usage, requiring scheduled/ad-hoc maintenance actions in case of malfunctioning

Element	Name	Annotator	Name
	Alternative Data Element	**X**	*No alternative for a data element*
	Data Driven Error Event	**R**	*Recoverable data element*

Fig. 6. Novel modeling elements and annotators

((M)edium risk of data unavailability). This means that data objects *Firmness*, *Color*, *Product Info*, *Temperature* and *Humidity* will be associated with the label $\langle true, M, H \rangle$. Conversely, to identify damaged products, a visual inspection is conducted, meaning a (potential) (M)edium quality level for the data object *Damages*. Similarly, the quality of *Sample [analyzed]* depends by the specific sample chosen, which leads to a (M)edium value for this parameter (cf. Fig. 5).

4.4 Level 3 – Alternative Data Awareness

Based on the information about the sources of failures and their potential impacts, the BP designer can decide to include alternative data in the BP model. Starting from the data elements with a higher risk of unavailability and lower data quality, the BP designer specifies if there are alternative data sources and how to reach them. To this aim, we introduce the function $Alt(d)$, which associates to a relevant data element $d \in \mathcal{D}$ an alternative data element $d_{al} \in \mathcal{D}$, or the special keyword 'X' if no alternative exists for d. This enables us to define data elements that act as *primary* data sources for some activities/events and others that work as their *alternatives*. As shown in Fig. 6 and in Fig. 7, we represent an alternative data element through a new BPMN icon with a shape identical to a "traditional" data element, but with a dashed border attached to the primary data source. If the BP designer is aware that no alternative is possible for a primary data, then the dashed border icon is labeled with 'X'.

Definition 4 (Level-3 compliant model). *Let $\mathcal{N} = \langle \mathcal{O}, \mathcal{A}, \mathcal{G}, \mathcal{E}, \mathcal{F}, \mathcal{C}, Cond, \mathcal{D}, \mathcal{T}_{IN}, \mathcal{T}_{OUT} \rangle$ be a Level-2 compliant model. $\mathcal{N}$ is said to be a "Level-3 compliant model" iff, for each $d \in \mathcal{D}$ with $State(d) = true$, then: (i) there exists $d_{al} \in \mathcal{D}$ such that $d_{al} \neq d$ and $Alt(d) = d_{al}$, or (ii) $Alt(d) = X$.*

In our running example, we can associate the primary data objects having some risk of unavailability with a "backup" alternative version of the data. For example, if the optical sorter stops working, the human operators can employ a portable penetrometer to measure the products' firmness, and a spectrophotometer to perform color measurement based on spectral reflectance. Similarly, temperature and humidity can be obtained through portable temperature and humidity probes. Also the product information can be obtained employing a manual scanner. Of course, using manual devices to perform continuous measurements rather than automatic sensors can decrease the quality of the collected data. This means that the alternative data objects *Penetrometer Firmness*, *Spectral Reflectance*, *Product Info via manual scan*, *Temperature probe measurement* and *Humidity probe measurement* will be associated with the label $\langle true, L, M \rangle$.
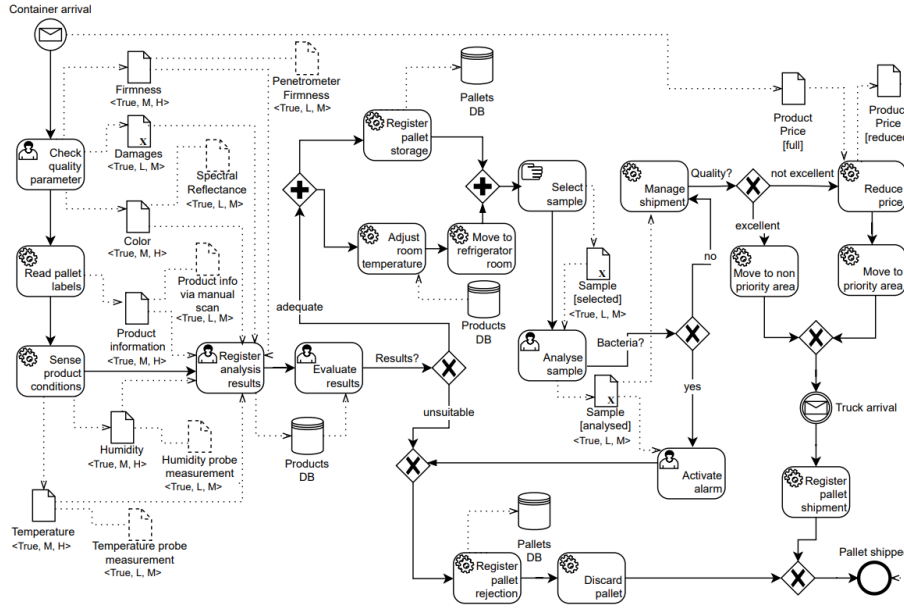
Fig. 7. Level-3 (Alternative Data Awareness) compliant BP model

It is worth to notice that no alternatives exist for the data objects *Damages*, *Sample [selected]* and *Sample [Analyzed]*, i.e., the BP designer is declaring her awareness that these data represent single point of failures (cf. Fig. 7).

4.5   Level 4 − Data Recovery Awareness

In the previous level, we have discussed how the presence of alternative data allows us to substitute primary data sources if they are missing or unreliable. However, the quality of an alternative data is usually lower than its original counterpart, and sometimes this can be not adequate to progress with BP execution. To mitigate this issue, the final level of our maturity model pushes a BP designer to specify remedial actions to improve the quality of a data to a degree that is comparable to its original counterpart. These actions are triggered employing a new modeling element, named *data-driven error event*, which can be embedded in a event sub-process. In BPMN, event sub-processes are used to capture global BP exceptions and define recovery procedures. We represent a data-driven error event with a document marker within the event shape (see Fig. 6). In our maturity model, we let the BP designer deciding if a data element requires to be restored trough a recovery procedure; if this is the case, then the icon of the data element to be recovered must be labeled with 'R'. At this point, to switch from Level 3 to Level 4, for any "recoverable" data element $d \in \mathcal{D}$, a data-driven error event $e_v \in \mathcal{E}_s$ is coupled with d and followed by a sub-process including the remedial actions to adjust its quality. From a formal perspective, we introduce the function $Rev(d)$ that associates to d a data-driven error event e_v, or the special keyword "NR" if d is considered as not recoverable.
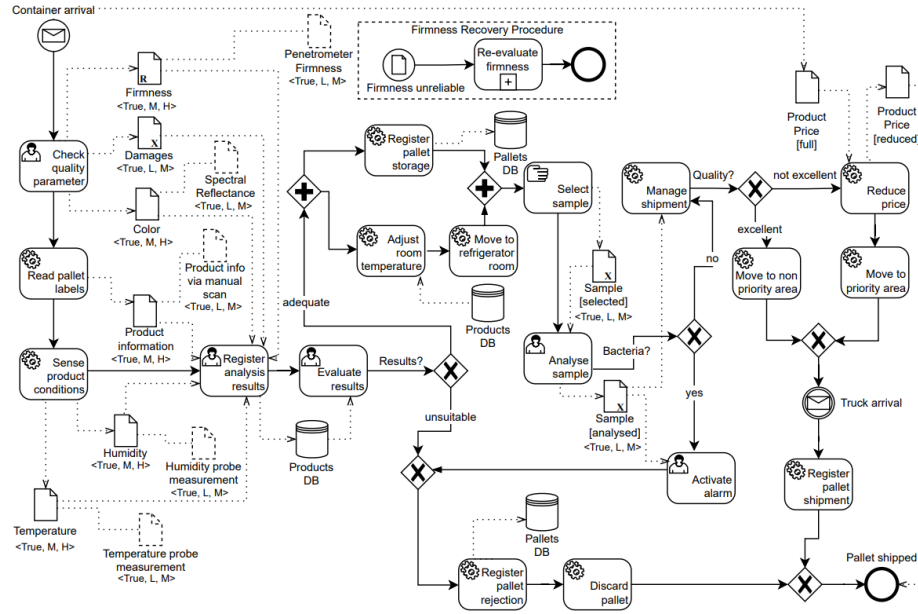
Fig. 8. Level-4 (Data Recovery Awareness) compliant BP model

Definition 5 (Level-4 compliant model). *Let* $\mathcal{N} = \langle \mathcal{O}, \mathcal{A}, \mathcal{G}, \mathcal{E}, \mathcal{F}, \mathcal{C}, Cond,$ $\mathcal{D}, \mathcal{T}_{IN}, \mathcal{T}_{OUT} \rangle$ *be a Level-3 compliant model.* $\mathcal{N}$ *is said to be a "Level-4 compliant model" iff, for each* $d \in \mathcal{D}$ *with* $State(d) = true$, *then: (i) there exist a data-driven error event* $e_v \in \mathcal{E}_s$, *an end event* $e_n \in \mathcal{E}_e$, *a sub-process* $a \in \mathcal{A}$, *an event sub-process* $a_{es} \in \mathcal{A}$, *and two sequence flows* f_1 *and* f_2 *such that* $Rev(d) = e_v$, $f_1 = (e_v, a)$, $f_2 = (a, e_n)$, *and* $\{e_v, f_1, a, f_2, e_n\} \in a_{es}$, *or (ii)* $Rev(d) = NR$.

Concerning our running example, we can assume that if the optical sorter stops working and the amount of pallets to be checked is too high, then employing the portable penetrometer to measure the products' firmness becomes too time consuming for the human operators. Therefore, the BP designer can mark the data object "Firmness" with a 'R' and associate it to the data-driven error event called "Firmness unreliable". As shown in Fig. 8, this will trigger the starting of a recovery procedure that, for example, instructs to move the pallet in another area of the distribution center where an auxiliary optical sorter is located by restoring the availability and quality of the original data object "Firmness". However, the enactment of the recovery procedure requires additional time and effort to be enacted, making it feasible only in exceptional cases. Of course, similar considerations can be made for the other relevant data objects in the BP.

5   Extending BPMN

One key feature of BPMN relies on its well-defined metamodel that facilitates BP model exchangeability and tool integration. In the BPMN 2.0 specification

document [12], the metamodel is represented by UML class diagrams, including object classes with required and optional attributes. Since all valid BPMN models must conform to the specifications of the metamodel, we need to extend the BPMN metamodel inserting the novel elements to design resilient models. In this direction, BPMN provides an "extension by addition" mechanism that enables the definition and integration of domain-specific concepts and ensures the validity of the BPMN core elements [18]. The following elements are needed to specify valid BPMN extensions. An *Extension Definition* is a named group of new attributes that can be used by BPMN elements, and consists of many *Extension Attribute Definitions* that define the particular attributes, whose values can be defined by the *Extension Attribute Value* class. To exploit the extension capabilities of BPMN, we have customized the well-known procedure for the methodical development of valid BPMN extensions provided by Stroppi in [18], which consists of the following steps (RES-BPMN is the name of our extension):

1. define a CDME (Conceptual Domain Model of the Extension) as UML class diagram that is able to capture the novel resiliency aspects;
2. define the RES-BPMN model based on the previous CDME model;
3. transform RES-BPMN into an XML Extension Definition Schema (EDS);
4. transform the XML EDS into an XML Schema Document.

Since our work mainly focuses on conceptual aspects and aims to create a maturity model, only the first two steps of the procedure are shown here. First, we identified a set of UML Class diagrams to be modified for capturing the novel BPMN elements (cf. Fig. 6): `Data Object`, `Data Store`, `Data Association` and `Event`. Then, for each of them, we created the CDME model, whose classes are typed as standard BPMN Concepts. Finally, the RES-BPMN model was derived by the application of the model transformation rules covering all possible CDME configurations to extend the existing Class Diagrams. For the sake of space, we focus here just on the extension of the `Data Object` Class Diagram (cf. Fig. 9). The complete list of CDME models and UML Class diagrams is available in an online appendix at: `https://tinyurl.com/bpm2022`.

As shown in Fig. 9, we introduced new attributes to the BPMN standard, which are highlighted in bold. For failure awareness (Level 1), we exploit the existing optional `DataState` attribute, which indicates that the unavailability of a data object may affect the execution of the BP flow objects to which it is connected. By default, its value is set to *false*. For risk and quality awareness (Level 2), we defined two attributes: `risk_lev` and `quality_lev`, which allow respectively to capture the unavailability risk and quality level of a data object through four values: U, L, M and H. By default, at Level 1, these attributes are set to U, i.e., their values are unknown a-priori. Alternative data awareness (Level 3) is captured with a boolean attribute `isAlternative`. In particular, for a given data object, `isAlternative` can be set either to *true* if an alternative version of the data exists (the association between a data and its alternative is made explicit throw a new class `DataAlternativeAssociation` created within the `DataAssociation` class) or *false*, i.e., there is no alternative for the data object. Finally, data recovery awareness (Level 4) is addressed by setting the
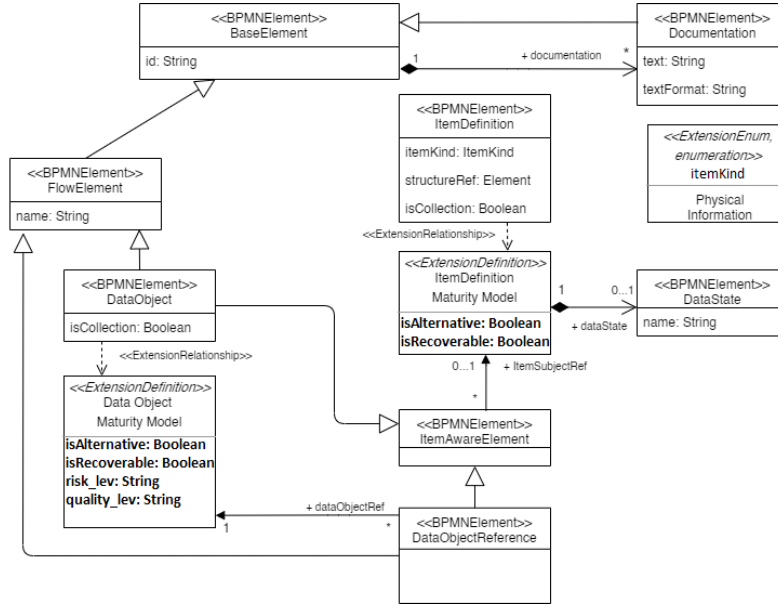
Fig. 9. RES-BPMN UML Class Diagram of Data Object Class

attribute `isRecoverable` to *true*, which indicates that the BP designer can provide a recovery strategy for the data object through a data-driven error event. This is captured within the `Event` class.

6   User Evaluation and Concluding Remarks

Extending the metamodel of BPMN has allowed us to develop a software tool, called RES-BPMN, which implements our approach to systematically design resilient BP models in BPMN and check their compliance with the different levels of the maturity model. In the case of non compliance with a certain level, the tool suggests the steps to refine the BP model to achieve the desired level of resilience. RES-BPMN has been developed as an extension of bpmn.io, an open source BPMN web modeler provided by Camunda, and it is written in Javascript using NodeJS framework on top of two libraries: *diagram-js* and *bpmn-moddle*. Thus it can run into modern browsers requiring no server back-end. RES-BPMN can be downloaded at: `https://github.com/bpm-diag/RES-BPMN`.

   Being RES-BPMN the only tool available in the literature for the specification of resilient-aware BP models in BPMN, no direct comparison was possible against other BP modeling tools. For this reason, we opted to investigate the feasibility of our approach through a usability evaluation of the user interface (UI) of the tool coupled with a thinking-aloud session, where the users were asked to explicitly execute a modeling task with an external evaluator observing them, indicating the methodological issues found while interacting with the UI. The users were selected from universities (2 professors and 4 PhD students), business

(2 managers) and manufacturing companies (2 managers), and declared to be knowledgeable (60 %), skilled (20 %) or experts (20 %) in BP modeling.

After a preliminary training session on introducing RES-BPMN, starting from the (not-resilient) BP shown in Fig. 2 and its description, the users were requested to systematically increase its resiliency level using the features and feedback provided by the tool. All the users were able to complete their task (providing different valid solutions) within the maximum available time (15 minutes). As soon as a user completed the task, we administered a SUS questionnaire [16]. SUS consists of 10 statements evaluated with a 5-point numerical scale that ranges from 1 ("strongly disagree") to 5 ("strongly agree"). At the end of the questionnaire, an overall score is assigned to it. We compared the score against the benchmark presented in [16], which associates to each range of the SUS score a percentile ranking varying from 0 to 100, indicating how well it compares to other 5,000 SUS observations performed in the literature. Since the obtained average SUS score was 80.8, according to the benchmark, the tool's usability corresponds to a rank of A, which indicates a degree of usability almost excellent.

We also collected valuable insights about the practical applicability of the approach during the thinking-aloud sessions. In particular, the users criticized the absence of an indicator to quantify the distance between a BP model and the complete achievement of a resiliency level. In this direction, as a future work, we plan to develop such an indicator exploiting our formalization of resiliency levels and measuring the number of modeling elements that are not compliant with the definitions in Section 4. In addition, by associating the quality level and the risk of unavailability of data elements with numeric weights, we can use them to build a quantifiable "criticality value" that identifies the data that might have more severe negative effects in case of their unavailability of low quality. This value could enrich the above indicator to provide a better understanding of the impact and the risks of a non-compliance with a resiliency level.

A second threat to the feasibility of the approach is about the practical conditions and assumptions under which it can be considered as effective. In particular, the users pointed out that the existence of alternatives might not be always guaranteed; analogously, resilience might also be affected by other factors different from data, like resource unavailability, temporal constraint violations, etc. In this paper, we focused on the data as main source of failures affecting BP resilience, and covering other potential factors is out of the scope of this work. However, the investigation of such factors is in the list of future works.

To sum up, we believe that measuring the usability of the UI of RES-BPMN is as a good preliminary indicator to validate the feasibility of our approach. The resiliency levels introduced in this paper, being based on a well-known standard such as BPMN, go in the direction of providing a reference framework for developing novel techniques and metrics to address BP resilience towards more accurate quantitative analysis. Of course, a general acceptance of the maturity model needs an extensive empirical evaluation of the approach.

References

1. van der Aalst, W.M.P., Pesic, M., Schonenberg, H.: Declarative workflows: Balancing between flexibility and support. Computer Science - R&D **23**(2) (2009)
2. Antunes, P., Mourão, H.: Resilient Business Process Management: Framework and services. Expert Systems with Applications **38**(2), 1241 – 1254 (2011)
3. Caralli, R.A., Allen, J.H., White, D.W.: CERT resilience management model: A maturity model for managing operational resilience. Addison-Wesley (2010)
4. Dadam, P., Rinderle, S.: Workflow evolution. In: Encyclopedia of Database Systems, pp. 3540–3544. Springer (2009)
5. Dijkman, R.M., Dumas, M., Ouyang, C.: Semantics and analysis of business process models in BPMN. Information and Software Technology **50**(12) (2008)
6. Hollnagel, E., Woods, D.D., Leveson, N.: Resilience Engineering: Concepts and Precepts. Ashgate Publishing, Ltd. (2007)
7. Jain, P., Pasman, H.J., Waldram, S., Pistikopoulos, E., Mannan, M.S.: Process Resilience Analysis Framework (PRAF): A systems approach for improved risk and safety management. Journal of Loss Prev. in the Proc. Ind. **53** (2018)
8. Janiesch, C., et al.: The Internet of Things Meets Business Process Management: A Manifesto. IEEE Systems, Man, and Cybernetics Magazine **6**(4), 34–44 (2020)
9. Marrella, A., Mecella, M., Sardina, S.: Supporting adaptiveness of cyber-physical processes through action-based formalisms. AI Communications **31**(1) (2018)
10. Moore, S.J., Nugent, C.D., Zhang, S., Cleland, I.: Iot reliability: a review leading to 5 key research directions. CCF Transactions on Pervasive Computing and Interaction **2**(3), 147–163 (2020)
11. Müller, G., Koslowski, T.G., Accorsi, R.: Resilience - A new research field in business information systems? In: BIS. Springer (2013)
12. OMG: Business Process Modeling and Notation, Version 2.0.2 (Jan 2014), `http://www.omg.org/spec/BPMN/2.0.2/`
13. Plebani, P., Marrella, A., Mecella, M., Mizmizi, M., Pernici, B.: Multi-party business process resilience by-design: A data-centric perspective. In: CAiSE (2017)
14. Reichert, M., Weber, B.: Enabling Flexibility in Process-Aware Information Systems - Challenges, Methods, Technologies. Springer (2012)
15. Rosa, M.L., Aalst, W.M.V.D., Dumas, M., Milani, F.P.: Business process variability modeling: A survey. ACM Computing Surveys (CSUR) **50**(1), 1–45 (2017)
16. Sauro, J., Lewis, J.R.: Quantifying the User Experience: Practical Statistics for User Research. Morgan Kaufmann (2016)
17. Steinau, S., et al.: DALEC: A Framework for the Systematic Evaluation of Data-Centric Approaches to Process Management Software. SOSYM **18**(4) (2019)
18. Stroppi, L.J.R., Chiotti, O., Villarreal, P.D.: Extending bpmn 2.0: method and tool support. In: BPMN Workshop. Springer (2011)
19. Sun, S.X., Zhao, J.L., Jr., J.F.N., Sheng, O.R.L.: Formulating the Data-Flow Perspective for Business Process Management. Inf. Syst. Res. **17**(4), 374–391 (2006)
20. Suriadi, S., Weiß, B., et al.: Current research in risk-aware business process management: Overview, comparison, and gap analysis. CAIS **34**(1) (2014)
21. Valderas, P., Torres, V., Serral, E.: Modelling and executing iot-enhanced business processes through BPMN and microservices. J. Syst. Softw. **184**, 111139 (2022)
22. Zahoransky, R.M., Brenig, C., Koslowski, T.: Towards a process-centered resilience framework. In: ARES (2015)
23. Zahoransky, R.M., Koslowski, T., Accorsi, R.: Toward Resilience Assessment in Business Process Architectures. In: SAFECOMP (2014)