



An Integrated Architecture for IoT-Aware Business Process Execution

Stefan Schönig¹(✉), Lars Ackermann¹, Stefan Jablonski¹, and Andreas Ermer²

¹ Institute for Computer Science, University of Bayreuth, Bayreuth, Germany
{stefan.schonig,lars.ackermann,stefan.jablonski}@uni-bayreuth.de

² Maxsymba GmbH and Co. KG, Floß, Germany
aermer@maxsymba.de

Abstract. Business processes are frequently executed within application systems that involve humans, computer systems as well as objects of the Internet of Things (IoT). While several works are emerging on combining BPM and the IoT, the exploitation of IoT technology for system supported process execution is still constrained by the absence of a common system architecture that manages the communication between both worlds. In this paper, we introduce an integrated approach for IoT-aware business process execution that exploits IoT for BPM by providing IoT data in a process-aware way, providing an IoT data provenance framework, considering IoT data for interaction in a pre-defined process model, and providing wearable user interfaces with context specific IoT data provision. The approach has been implemented and evaluated extensively in production industry. The results show that the application of IoT enhanced BPM leads to less machine stops.

Keywords: Process execution · Internet of Things
Wearable interface

1 Introduction

Business process management (BPM) is considered as powerful technology to operate, control, design, document, and improve cooperative processes [1]. Processes are executed within application systems that are part of the real world involving humans, cooperative computer systems as well as physical objects [2]. Internet of Things (IoT), denoting the inter-networking of all kinds of physical devices, has become very popular these days. It enables continuous monitoring of phenomena based on sensing devices, e.g., wearables, machine sensors, etc. Therefore, IoT contributes to the recent trend known as big data. Process execution, monitoring and analytics based on IoT data can enable a more comprehensive view of processes. Embedding intelligence by way of real-time data gathering from devices and sensors and consuming them through BPM technology helps businesses to achieve cost savings and efficiency.

Let us consider a production process where raw material is processed by different machines under the supervision of human operators. In case of product

quality issues, manual human involvements are necessary. Additionally, operators must be aware of current sensor data to decide on tasks to be executed next. Such a scenario might be better manageable when closely linking digital production and machine data with human operators as enabled by the integration of IoT and BPM. Here, the necessity of human activities can be triggered by a BPM engine through the reference of appropriate IoT sensor data in the underlying process model. This way, human operators can be notified seamlessly without loss of time on wearable devices while leveraging current context specific information. As a consequence, the integration of IoT and BPM technology could lead to efficiency gains by reducing reaction time and enhance the quality of task execution.

In literature, several works are emerging on combining IoT and BPM, e.g., monitoring running processes to align them with the state of the things [3, 4]. Still, there are many open challenges to be tackled [5]. In particular, the exploitation of IoT technology for system supported process execution is constrained by the absence of a common architecture that manages and standardizes the communication between both worlds. In this paper, we close this gap by proposing an integrated approach for IoT-aware business process execution that exploits IoT for BPM by *(i)* providing IoT data in a process-aware way, *(ii)* providing an IoT data provenance framework, *(iii)* considering IoT data for interaction in a pre-defined process model, and *(iv)* providing wearable user interfaces with context specific IoT data provision. The approach has been implemented and evaluated extensively in production industry. The results show that the application of IoT enhanced BPM leads to less machine stops because operators need less time to recognize work to be done.

This paper is structured as follows: Sect. 2 describes research background and fundamentals. Section 3 introduces the approach for a IoT-aware process execution. In Sect. 4 we describe the implementation of our approach based on well-known communication protocols. In Sect. 5 we evaluate our approach with an extensive application in production industry. Section 6 gives an overview of related work and Sect. 7 finally concludes the paper.

2 Research Background and Fundamentals

The IoT is the inter-networking of physical objects like *(i)* electronic hardware, e.g., sensors and actuators or *(ii)* humans using wearable digital devices like glasses and smartwatches. Such connected things collect and exchange data between each other. IoT allows things to be controlled remotely across existing network infrastructures, including the Internet [6]. A business process is a collection of related events, activities, and decisions that involve a number of (human) resources. To support processes at an operational level, a BPM system (BPMS) can be used. During process execution, a variety of information is required to make meaningful decisions. With the emergence of IoT, data is generated from physical devices sensing their (business/manufacturing) environment that reflects certain aspects of operative processes. A BPMS deals, a.o., with the enactment of process models that define the interplay between environmental circumstances (depicted as data values), characteristics of participants

(depicted as resource assignments) and corresponding activities to be executed. We consider processes as explicit process representations (pre-defined models), which later are enacted.

Accordingly, sensing and perception the process environment via sensors constitutes the fundamental task of the IoT. Sensor data then must be aggregated, interpreted and made available to the BPMS in order to trigger business process activities or human tasks, respectively. These tasks must then be send in real time to responsible individuals that receive tasks on mobile user interfaces.

2.1 Research Problem

Recently, several researchers raised specific research challenges that need to be tackled in order to align IoT and BPM technology [5]. In this section, we explain a specific subset of those challenges that is tackled by the integration of IoT objects and a BPMS as it is described in the work at hand.

First of all, IoT sensors need to be placed in a process-aware way in order to be able to collect and record all process relevant IoT data. Therefore, sensors need to be carefully placed at machinery and humans and be digitally accessible. The acquired process relevant IoT information needs to be up-to-date and current. It needs to be clear where the data stems from and where it has been used (cf. *data provenance*), as well as the quality of the data at-hand needs to be ensured. It becomes necessary to find a way to annotate the IoT data's origin and to use this (meta-)information in business process models.

In many processes some activities require the interplay between human operators and software/hardware modules. Furthermore, there is an increasing use of mobile devices fostering the delivery of work items to the right users. Here, an appropriate mapping from activities to user interfaces is needed allowing process participants to perform their work from arbitrary places in the working environment. Participants might suffer from issues which hinder optimal working conditions. Here, the IoT can support the execution of tasks in a process through context-specific knowledge provisioning that is relevant for the users particular context. Sensor data can be leveraged to determine the actual context and to identify information needs.

2.2 Motivating Example

Additionally, we want to motivate the necessity of our approach by example of a real life process from production industry that is visualized in Fig. 1. The example stems from the corrugation area where paper is glued to corrugated paper that depicts the basis for cardboard boxes. The given example is a subprocess that is executed every time paper source rolls run empty, i.e., where new paper rolls need to be spliced with the paper from the low running roll. In order to effectively execute this process, several real time interactions with IoT devices, i.e., sensors and operator equipment, is necessary: the BPMS must be aware of sensor data which indicates that a splice will happen soon, triggering the splice

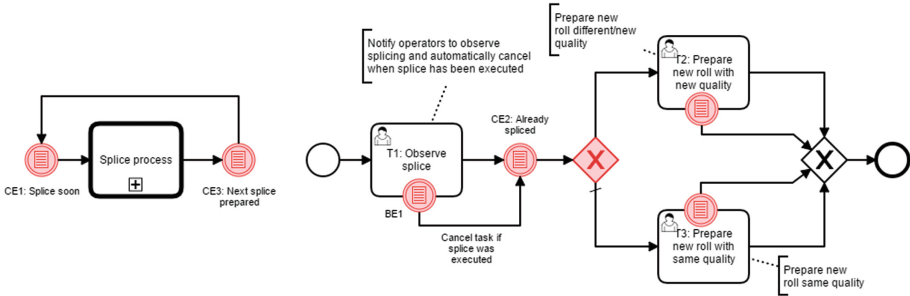


Fig. 1. Exemplary production process: (a) repeating subprocess; (b) splice process

subprocess. Operators located somewhere along the up to 300 m long machinery need to observe the splice process to avoid issues. Therefore, they need to be notified in real time to walk to the splicer. This requires wearable interfaces communicating with the BPMS over the IoT. Depending on a sensor value indicating the next roll quality, the BPMS has to execute different paths. In case the environment changes, operators tasks need to be reordered based on priorities or cancelled by the BPMS. In addition to current tasks to be executed, operators require context specific information at hand, e.g., the location of the splicer and the quality of the next paper roll. Furthermore, operators continuously need to observe viscosity and temperature of the glue to ensure a successful splice process. In the following sections we show that the integration of IoT devices and a BPMS serves as a generalisable solution to the problems above.

3 Integrated Architecture for an IoT-Aware BPMS

We propose a four-steps procedure to provide the necessary information. The first step is connecting IoT objects and their values traceable to a BPMS. We call a single type of value of a certain (sensor) object *variable*. The second step is extending a BPMN process model with data variables participating in the process stemming from physical objects such as machine status or actor positions. The resulting process models must be applicable by default contemporary BPMN execution engines. This way, organizations can reuse existing process models, without having to learn new languages and remodel processes from scratch. The third step is to establish a real time notification interface of triggered activities to process participants by means of mobile devices. In the fourth step, context relevant information stemming from connected objects must be selected and provisioned to users. Our approach poses the following four main requirements.

R1. A BPMS must be aware of current values of IoT objects. Variable attributes, e.g., address and identifiers, must be configurable and traceable, i.e., it must be clear where the data stems from.

R2. *Each defined variable must be referenceable in the executed process model.* Based on current values of certain variable, tasks are triggered or cancelled and decisions are made.

R3. *Responsible users must be notified on mobile devices in real time.* Process participants must be seamlessly notified when human interaction is required, independent of where the user is located.

R4. *Context-specific knowledge must be provisioned to users.* Alongside with activity notification, context-specific and process relevant information must be provisioned to users.

3.1 Connecting Current Data of IoT Objects to BPMS

First, we formally define IoT object data that is regularly, i.e., in a certain interval int acquired from digitally accessible devices and stored in a IoT middleware database. Data recorded from IoT objects is an ordered sequence σ of events e_j , i.e., $\sigma = \langle e_1, \dots, e_{|\sigma|} \rangle$. In general, an event e is defined as tuple of attributes-value pairs, i.e., $e = \left((attr_1^e, val_1^e), \dots, (attr_{|e|}^e, val_{|e|}^e) \right)$. The set of all attributes $ATTR^e$ and the corresponding values VAL^e of an event e are therefore defined as $ATTR^e = \left\{ attr_1^e, \dots, attr_{|e|}^e \right\}$, $VAL^e = \left\{ val_1^e, \dots, val_{|e|}^e \right\}$. Each $attr_i^e$ has an unique variable identifier var and a timestamp $time$, i.e., $\forall e \in \sigma (var^e, time^e) \subseteq ATTR^e$. Each attributes-value pair $(attr_i^e, val_i^e)$ is dedicated to exactly one IoT variable. With $l^e(var_i^e) = val_i^e$ we denote the value val_i^e of a certain variable var_i^e in event e . A total order of events is implemented as follows: $\forall 1 \leq j < k \leq |\sigma| (time^{e_j} < time^{e_k})$. Therefore, the current value of a certain variable var_i is given as $l^e(var_i^{e_{|\sigma|}}) = val_i^{e_{|\sigma|}}$. All other attributes are optional. Based on these definitions, we ensure that IoT data variables are configurable and traceable (cf. R1). In the next step, current data of each connected IoT object must be sent to the BPMS (cf. R1). Therefore, we need to map IoT data to enacted process models of the BPMS. Consider a set of process models P where each p contains a set of data variables D_p . Each variable $d \in D_p$ has an unique identifier var_d . The underlying assumption is that each participating IoT variable is referenced by the same unique identifier in the corresponding process model. If we want to establish a connection between the data of an IoT object and a process variable then both identifiers have to be the same, i.e., $var_i = var_d^p$. Having established such a semantic mapping, only current data of each connected object is sent to the BPMS, i.e., a sending procedure sp is initiated for each variable $var_i^{e_{|\sigma|}}$ recorded in $e_{|\sigma|}$ sending the latest acquired values to the BPMS, i.e., $\forall 1 \leq i \leq |e| val_i^{e_{|\sigma|}}$.

Example: Let us assume, we acquire data from a sensor indicating the restmeters of paper on a specific roll RM_1 , the quality of the paper roll to be used next QU and the current glue temperature GT from a temperature sensor in an interval $int = 10s$. The acquired IoT data is then exemplarily given as shown in Table 1. Consider a process model p where two data variables are referenced

Table 1. Example acquired IoT object data

Event	$attr_i^e(var, \dots, time)$	val_i^e	Data type
1	$(RM_1, \dots, 2018-01-09 10:15:32)$	300	Historic
1	$(QU, \dots, 2018-01-09 10:15:32)$	211 C	Historic
1	$(GT, \dots, 2018-01-09 10:15:32)$	120	Historic
2	$(RM_1, \dots, 2018-01-09 10:15:42)$	133	Current
2	$(QU, \dots, 2018-01-09 10:15:42)$	211 C	Current
2	$(GT, \dots, 2018-01-09 10:15:42)$	115	Current

and evaluated. These variables are defined as $var_{d_1}^p = RM_1$ and $var_{d_2}^p = QU$, respectively, in the model. The procedure sp is initiated every 10 s. Therefore, the last execution of sp sends the values 133 and 211C to the BPMS where these values are dedicated to the corresponding process variables $\{d_1, d_2\} \in D_p$. Note, that data referring to GT is not sent to the BPMS.

3.2 Enrichment of Process Models with IoT Variables

As described in Sect. 3.1, we consider a set of process models P where each p contains a set of process variables D_p that semantically correspond to a variable of an IoT object. Based on the established real time connection described before, these IoT aware process variables, identified by $var_{d_i}^p$, can be referenced in enacted process models in various ways. We assume that a given process model p is defined as BPMN conform process diagram, one of the most used formalisms for process modeling, representing the IoT aware process to be executed. A BPMN process diagram specifies which activities are executed in a process as well as their control flow relationships. To be able to infer when activities start or end based on the state of the variables, the diagram must capture this information (cf. R2). This step requires the process designer to enrich the given BPMN diagram by including information on how and where the connected IoT variables influence the process. We consider several possible interaction possibilities:

a. IoT based trigger events. Sometimes we only want a process to start or to continue if a certain condition is true. In BPMN conditional events define an event which is triggered if a given condition is evaluated to true. It can be used as an (i) *Intermediate Conditional Event*; as a (ii) *Boundary Event* or as a (iii) *Conditional Start Event*. In case of (i) current values of used IoT variables $var_{d_i}^p$ trigger an *Intermediate Catch Event* and the execution continues to the next activity. In case of (ii) the BPMS checks if the process environments changed based on the current values of the connected IoT variables. If the given condition is satisfied, then the corresponding activity will be interrupted.

Example: Consider the example process in Fig. 1 where the control flow reaches the conditional event $CE1$ and the condition is, for example, $RM_1 < 200$. Given a current IoT sensor value of $val_{RM_1}^{e|\sigma|} = 133$ the condition will be satisfied and the BPMS triggers the subprocess *Splice process*. Similarly, if the splice has

Input: T : Set of available tasks $T = \{t_1, \dots, t_{|T|}\}$
Input: U : Set of registered users $U = \{u_1, \dots, u_{|U|}\}$

```

foreach  $u \in U$  do
   $deviceId$ : identifier of mobile device dedicated to  $u$ 
   $tasksToSend \leftarrow \emptyset$ 
  foreach  $t \in T$  do
    if  $u$  among candidate users of  $t$  then
       $tasksToSend \leftarrow tasksToSend \cup \{t\}$ 
  if  $tasksToSend > 0$  then
     $sendTasks(deviceId)$ 

```

Algorithm 1. Distributing assigned human tasks to wearable devices

been executed successfully, i.e., $val_{RM_1}^{e|\sigma|} > 200$, then *Observe splice* is aborted by means of the corresponding boundary event *BE1*.

b. IoT based decisions. In BPMN an data-based gateway, is used to model a decision in the process. Similar to conditional events, current IoT variable values can be used to decide which sequence flow is selected for continuing the process.

Example: Depending on the next paper quality to be used in the production order the BPMS has to decide how to continue the process, i.e., based on the current value of $val_{QU}^{e|\sigma|}$ either task *T2* or task *T3* will be triggered.

c. IoT based loops. In order to model repeated behavioural patterns, IoT variables can be used to define event-driven loops. This way, end-to-end processes can be broken up into comprehensible *micro-processes*.

Example: In order to support the recurring splice process in Fig. 1(a), the subprocess is surrounded by the conditional events *CE1* and *CE3*. While the positive evaluation of the given conditions in *CE1*, i.e., splice happening soon, triggers the subprocess to be executed, *CE3* checks if all preparations for the next splice have been executed. In case the corresponding IoT stemming data values fulfil the given event condition, the process continues *CE1*.

3.3 Establishing the Real Time Mobile User Interface

It is important that process participants are seamlessly notified when human interaction is required, independent of where the user is located. This requires a real time notification on mobile devices of responsible users (cf. R3). Therefore, it is necessary to define a mapping of users to corresponding mobile devices that serve as wearable, user-specific process cockpits and task lists, respectively. This is achieved by specifying a dedicated mobile device identifier for each defined user in the BPMS. During process execution, the currently available tasks for a specific process participant are then directly sent to the specified mobile device. The algorithm to distribute the currently available human tasks to defined mobile devices is given in Algorithm 1. The actual operator to device mapping and the task distribution is described in detail in Sect. 4.

Example: The splice process in Fig. 1(b) requires operators to manually observe the splice at the specific machinery (*T1*). Therefore, in case the condition in

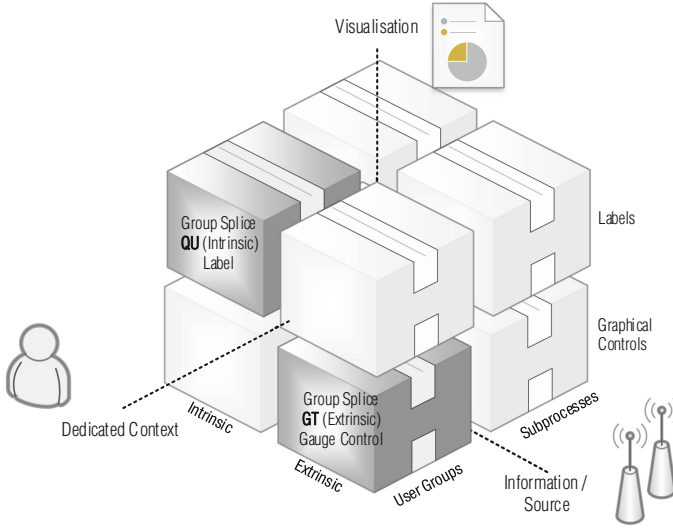


Fig. 2. Three dimensions of context-specific IoT information provision

CE1 evaluates true, the responsive operator needs to be notified in real time to be able to walk to the splice site in time. Hence, as soon as task *T1* becomes available, the list of currently available tasks of assigned operators, implemented as a smartwatch application is updated.

3.4 Context-Specific Information Provision

Alongside with activities, context-specific and process information must be provisioned to operators to improve the quality of task execution (cf. R4). In order to ensure that the provisioned information is of value for operators, the following three dimensions need to be considered when defining data that should be delivered to certain process participants (cf. Fig. 2). Based on these dimensions, Algorithm 2 distributes IoT information in a context aware way to corresponding users.

a. Dedicated Context - Which entities allow for a separate context definition? There are different entities where different contexts can be defined for. IoT variable information can be dedicated to subprocesses and delivered alongside with tasks of this subprocess to respective users. Furthermore, IoT variable data can be dedicated to specific user groups or roles. Within a location-aware BPM scenario, the information context can also be defined based on locations.

b. Information/Source - Which IoT information should be provisioned? IoT data can be classified according to the factors of the context framework defined in [7], i.e., *intrinsic* or *extrinsic* context information. Intrinsic IoT variable information reflects data used in the process model and is therefore directly related to the

Input: C : Set of defined contexts $C = \{c_1, \dots, c_{|C|}\}$
Input: U : Set of registered users $U = \{u_1, \dots, u_{|U|}\}$
Input: V_I : Set of intrinsic variables $V_I = \{v_{i1}, \dots, v_{|V_I|}\}$
Input: V_E : Set of extrinsic variables $V_E = \{v_{e1}, \dots, v_{|V_E|}\}$

```

foreach  $c \in C$  do
   $relVars_c$ : set of relevant attributes-value pairs for context  $c$ 
   $relVars_c \leftarrow \emptyset$ 
  foreach  $v_i \in V_I$  do
    if  $v_i$  among related variables of  $c$  then
       $val_{v_i} \leftarrow$  receive current value from workflow engine
       $relVars_c \leftarrow relVars_c \cup \{(v_i, val_{v_i})\}$ 
  foreach  $v_e \in V_E$  do
    if  $v_e$  among related variables of  $c$  then
       $val_{v_e} \leftarrow$  receive current value from IoT middleware
       $relVars_c \leftarrow relVars_c \cup \{(v_e, val_{v_e})\}$ 
  foreach  $u \in U$  do
     $deviceId$ : identifier of mobile device dedicated to  $u$ 
    if  $u$  among related users of  $c$  then
       $sendInformation(relVars_c, deviceId)$ 

```

Algorithm 2. Provisioning context-specific IoT information

currently executed process instance. Extrinsic IoT data is information that is not necessarily related to a running processes but might influence process execution as well, e.g., production hall temperature. Furthermore, the granularity of provisioned information must be adjustable w.r.t. different processes. In some cases it might be helpful to see more information, in some cases it might be better to see less information to prevent users from information overload.

c. Visualisation - How is a certain context-specific IoT variable presented or visualised? Context-specific IoT information must be visualised in a proper way. Depending on the class of information (intrinsic, extrinsic) or the data type (string, number) different positions on the interface and representation styles are appropriate. Intrinsic, instance specific information that is relevant for the execution of a certain activity can be represented as information below the activity name. Extrinsic, environmental data that might be of importance to process execution, can be represented as separate controls on additional tabs.

Example: In the splice process in Fig. 1(b), a context is defined for specific groups of operators. Only the users assigned to group *Wet-End* will receive information w.r.t. the splice process. Members of this group receive intrinsic, instance specific information like the quality of the paper roll to be used next. Since this is highly relevant data for executing task T_2 , this information is presented directly below the activity name. Furthermore, users receive the restmeters of paper on a specific roll (intrinsic) and the current glue temperature (extrinsic). Both values are numbers that are important information for operators but not directly related to activities. Thus, these values are visualized on additional tabs on the operators smartwatch. Figure 2 shows the characteristics in each dimension of the exemplarily described IoT variables.

Table 2. MQTT communication between IoT objects and BPMS

Topic	Description	Direction
$/\{variable_id\}/data$	IoT sensor data	IoT to BPMS
$/\{actor_id\}$	Device configuration	BPMS to IoT
$/\{actor_id\}/tasks$	Tasks of specific actor	BPMS to IoT
$/\{actor_id\}/\{variable_id\}$	Context data for specific user	BPMS to IoT
$/\{actor_id\}/command$	Actors actions (claim, complete, cancel)	IoT to BPMS
$/keepalive$		IoT to BPMS

4 Architecture and Implementation

The described approach has been implemented based on a three layer architecture that is visualised in Fig. 3. It consists of the following layers: (i) IoT objects, i.e., sensors as well as wearable devices of human participants; (ii) IoT infrastructure and communication middleware; and (iii) the BPMS. The layers are connected based on standard communication protocols. To allow the IoT objects at layer (i) to communicate with the IoT middleware at layer (ii) and the BPMS, respectively, a Message Queue Telemetry Transport (MQTT)¹ Broker is used. MQTT is a queue-based publish/subscribe protocol, which is especially suited for applications where computing power and bandwidth are constrained. The used MQTT topics are listed in Table 2. IoT objects, i.e., sensors or actuators, represent publishers. They are connected to an IoT gateway using specific architectures such as Profibus, LAN, WLAN or Bluetooth. A specific IoT variable v_x is acquired and published on a MQTT topic $/v_x/data$. Through a MQTT Broker the acquired data is sent to an acquisition application at layer (ii) that stores IoT data into a high performant NoSQL database that follows the database scheme described in Sect. 3.1. In our implementation we used the latest version of the Apache Cassandra database. A distribution application at layer (ii) keeps the BPMS updated with the latest IoT values. All running instances of a particular process receive the corresponding data value. The application cyclically acquires the values from the database in a Key-value structure and sends them to the BPMS. In our architecture we used the latest version of the Camunda BPMS and therefore communicated with the workflow engine by means of the Camunda Rest API², i.e., *PUT*, *POST* and *GET* HTTP requests as described in Fig. 3. The tools at layer (ii) ensure that process relevant information stemming from the IoT is up-to-date. Through the acquisition tool, IoT data meta information is provided that makes clear where the data stems from. Given the current IoT data values, the engine calculates available activities. As a mobile user interface we implemented an *Android* based smartwatch application that subscribes to specific MQTT topics. The distribution application cyclically requests the

¹ <http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html>.

² <https://docs.camunda.org/manual/7.8/reference/rest/>.

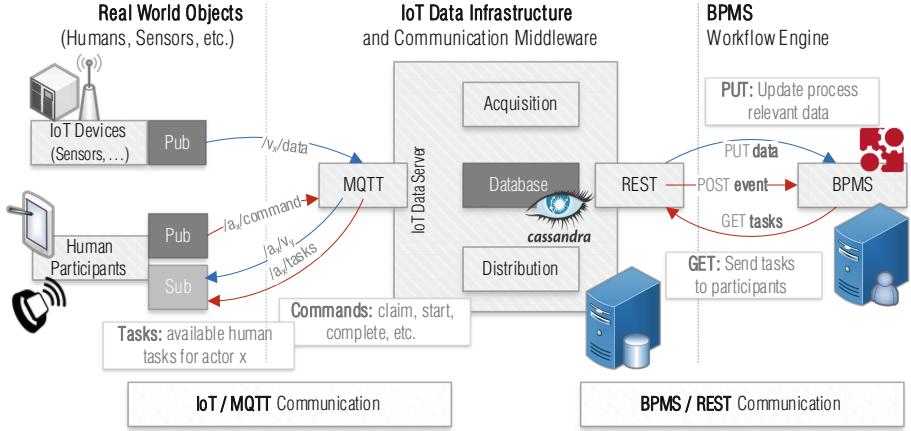


Fig. 3. Integrated communication architecture for IoT and BPMS

current user tasks from the Camunda API for each defined user and publishes to the correct MQTT topic, given the mobile device identifier, i.e., smartwatch device, configured on the BPMS (cf. Algorithm 1). The process of the device recognizing its configuration is implemented as follows: the distribution application cyclically checks the user configuration in the BPMS. When a change is detected, it publishes the new configuration to the topic $/\{actor_id\}$. The smartwatch of a certain actor subscribes to the topic of its specific device identifier. Having established such connections, the smartwatch communicates with the MQTT broker by subscribing to the following topics: the current tasks for a specific operator are published on the topic $/\{actor_id\}/tasks$. The device sends operators commands, such as *complete task* to the topic $/\{actor_id\}/command$. The content of the message is forwarded straight to the BPMS using a *POST* request. Context-specific IoT data is sent to actors on topic $/\{actor_id\}/\{variable_id\}$ based on Algorithm 2. To prevent the MQTT service at the watch to be killed, we implemented a keep alive communication (topic = */keepalive*).

5 Evaluation and Industrial Application

In this section, we describe the evaluation of the proposed approach by means of an extensive application in corrugation industry. Due to increasing automation and staff reduction, less operators are available to control such a production line. Hence, interactions between users and machinery requires several location changes of users between control panels that result in delayed information flows. These delayed reaction times are frequently the reason for increased deficient products. The overall corrugation BPMN process model that is executed by a Camunda BPMS as described in Sect. 4 is visualised in Fig. 4. After initialising internal helper variables the control flow splits into different BPMS parallel paths

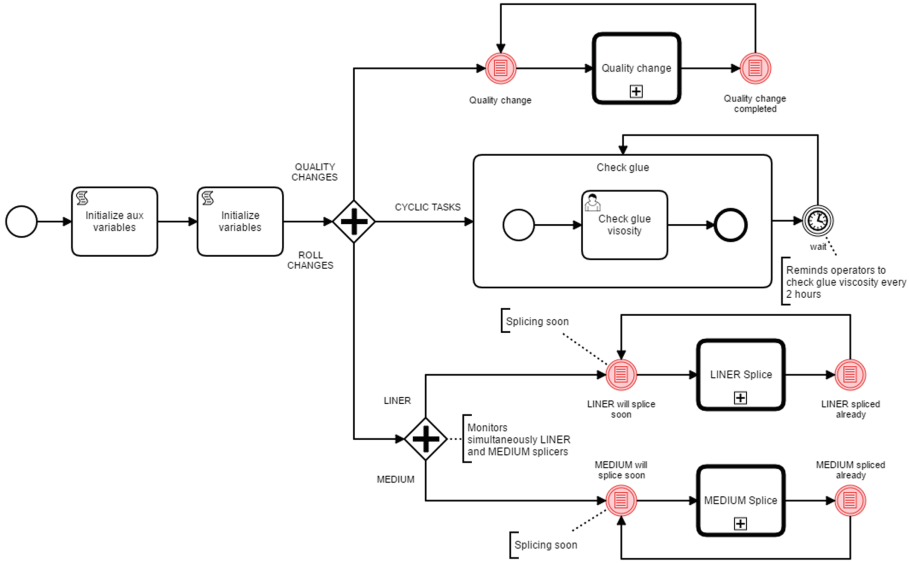


Fig. 4. Complete IoT enhanced process in corrugation industry (Color figure online)

where each path calls a specific subprocess depending on certain IoT based sensors conditions to be fulfilled (cf. red coloured Conditional Events). Each of these subprocesses reflects necessary operations that need to be performed to control production, e.g., the splice subprocess in Fig. 1. The implemented process contains all IoT enhanced modelling concepts that have been introduced in Sect. 3.2. To directly notify operators when human actions are needed, plant personal has been equipped with smartwatches, i.e., mobile process cockpits as described before and as shown in Fig. 5. Therefore, a user-group model has been defined in the BPMS. Here, available operators were assigned to a specific area of production that depicts their area of responsibility, e.g. machine *MF1*. Thus, depending on the area operators are working, the BPMS assigns a different set of tasks. Operators are then pointed to new human tasks through visual, acoustic and, in case of noisy environments through haptic signals like vibration alarms. Furthermore, operators are used more effective because low priority work is aborted in order to perform high priority work that could lead to machine stops.

In addition to currently available human tasks the IoT infrastructure provides diverse context-specific information on the smartwatch interface of operators. Depending on the specific group a user is assigned to, the wearable device offers diverse context information to operators: at the *Dry-End* (the area where produced corrugated paper leaves the plant), e.g., the remaining time of current production job, the remaining time to next stack transport, or the current production speed. Users at the *Wet-End* (the area where original paper is inducted to the plant) receive continuously information w.r.t. the next necessary roll change or occurring error and defects of machinery modules. In addition,



Fig. 5. Wearables: (a) unclaim/complete task; (b) tasks; (c) and (d) context info

operators receive error messages and environmental information from the different plant modules. This way, concrete and goal-oriented information in error cases or warning messages for supply shortfalls can be transmitted to operators and enhance the overall process transparency and thus the quality of task execution (cf. Challenges in Sect. 2.1). Through the described implementation it was possible to significantly reduce reaction time intervals. The amount of deficient products was decreased and the overall quality of the produced corrugated paper has been improved. The overall equipment downtime was significantly decreased, since problems have been prohibited or recognized in advance and were solved proactively. Hence, the overall equipment efficiency could be increased effectively. To quantify these findings, we analysed process execution with the Camunda Statistics Plugin³. We tracked the corrugation process (*i*) for five days without operators using wearable devices and (*ii*) other five days with operators being notified using smartwatches. In particular, we measured the average instance throughput time for a splice processes. The effectiveness of the approach has been measured based on machine stop times and waste reduction. On average, 100 splices are executed per shift, i.e., 8 h of production. In case (*i*) we recorded a total stop time of 180 min, i.e., on average 12 min per shift. In case (*ii*) the stop time has been decreased to 60 min in total, i.e., 4 min per shift on average. The results show that the application of the IoT enhanced BPMS leads to less machine stops because users need less time to recognize work to be done.

6 Related Work

Several approaches have been proposed to relate IoT objects with business processes. An overview of related approaches is given in Table 3. The table summarizes the support of each approach for IoT-aware process modelling, execution,

³ <https://github.com/camunda/camunda-cockpit-plugin-statistics>.

Table 3. Overview on related approaches of the integration of IoT and BPM

Approach	Modelling	Execution	Monitoring	Mobile UI	Context
<i>IAPMM</i> [8,9]	✓	✗	✗	✗	✗
<i>BPMN₄CPS</i> [10]	✓	✗	✗	✗	✗
<i>BPMN for IoT</i> [11–13]	✓	✗	✗	✗	✗
<i>IoT/WS-BPEL</i> [14]	~ (<i>BPEL</i>)	✓	~ (<i>BPEL</i>)	✗	~ (<i>BPEL</i>)
<i>IoT/WS-BPEL</i> [15,17,18]	~ (<i>BPEL</i>)	✓	~ (<i>BPEL</i>)	✗	~ (<i>BPEL</i>)
<i>ADiWa</i> [19]	~ (<i>concept</i>)	~ (<i>concept</i>)	✗	✗	✗
<i>Extended-GSM</i> [2–4]	✓	✗	✓ (<i>via GSM</i>)	✗	✓
<i>This work</i>	✓	✓	~ (<i>BPMS</i>)	✓	✓

and monitoring as well as the the availability of a mobile user interface and the possibility to provide (IoT) context information. In [8,9] the authors present the Internet-of-Things-Aware Process Modeling Method (*IAPMM*) that covers requirements analysis. It extends the BPMN meta-model to model IoT-aware processes. The approach in [10] (*BPMN₄CPS*) also describes an extension of BPMN in which the process logic is split into the cyber part, the controller and the physical part. Furthermore the authors extended BPMN by new task types. Some more notation concepts in BPMN for IoT are described in [11–13]. The main focus is on the modeling of real world properties. Also in [11,12] the authors present an extension of BPMN with new modeling concepts. None of the described approaches provides details on how to execute these models. In [14] an approach for implementing an IoT-aware execution system given in WS-Business Process Execution Language (WS-BPEL) is introduced. It extends BPEL by *context variables* which are automatically updated. The authors implemented a prototype which is compliant with every WS-BPEL engine. Other approaches implementing BPEL extensions are presented in [15,16]. The variables are updated using the publish/subscribe paradigm. Another extension for WS-BPEL (*Context₄BPEL*) with features to manage context events to allow the asynchronous reception of events, query context data and evaluate transition conditions based on context data is described in [17]. In [18] the authors integrate distributed resources into WS-BPEL by formalizing a fragment of WS-BPEL together with the WSRF (Web Services Resource Framework). In [19] the authors propose an approach for enabling IoT-based agile business processes. They provided concepts for extending models by triggers for variance. The approaches in [2–4] rely on the information coming from artifacts involved in the process to understand how a process evolves. By adopting an extension of Guard-Stage-Milestone (GSM), it is possible to monitor the process even when the control flow is not respected. The work presented in [20] introduces a lightweight process engine for enabling mobile applications. The authors describe requirements and concepts for mobile process applications and a prototypical mobile user interface. However, this work does not comprise IoT related aspects.

7 Conclusion and Future Work

In this paper, we introduced an integrated approach for IoT-aware business process execution that tackled several open challenges in this area of applied research. As a fundamental basis for the integration, we introduced an IoT data provenance framework. This technique allows us to consider current IoT data for interaction in arbitrary pre-defined process models that can be enacted by contemporary BPM execution systems. As demanded in many business cases, users need to be notified in real time when new tasks occur. This has been implemented by means of wearable user interfaces with configurable context specific IoT data provision. The approach has been implemented and evaluated extensively in production industry. For future research, we will focus on several IoT related issues in the area of BPM, e.g., the analysis of process execution event logs under consideration of IoT data.

References

1. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management, vol. 1. Springer, Heidelberg (2013). <https://doi.org/10.1007/978-3-642-33143-5>
2. Meroni, G., et al.: Artifact-driven process monitoring: dynamically binding real-world objects to running processes. In: CAiSE Forum (2017)
3. Meroni, G., Di Ciccio, C., Mendling, J.: An artifact-driven approach to monitor business processes through real-world objects. In: Maximilien, M., Vallecillo, A., Wang, J., Oriol, M. (eds.) ICSSOC 2017. LNCS, vol. 10601, pp. 297–313. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69035-3_21
4. Meroni, G., et al.: Multi-party business process compliance monitoring through IoT-enabled artifacts. *Inf. Syst.* **73**, 61–78 (2018)
5. Janiesch, C., Koschmider, A., et al.: The Internet-of-Things Meets Business Process Management: Mutual Benefits and Challenges. *CoRR*, vol. abs/1709.03628 (2017)
6. Schönig, S., Jablonski, S., Ermer, A., Aires, A.P.: Digital connected production: wearable manufacturing information systems. In: Debruyne, C., Panetto, H., Weichhart, G., Bollen, P., Ciuciu, I., Vidal, M.-E., Meersman, R. (eds.) OTM 2017. LNCS, vol. 10697, pp. 56–65. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-73805-5_6
7. Mahdi, R., Jablonski, S., Schönig, S.: Extrinsic dependencies in business process management systems. In: ICEIS (2018)
8. Petrasch, R., Hentschke, R.: Process modeling for industry 4.0 applications - towards an industry 4.0 process modeling language and method. In: International Joint Conference on Computer Science and Software Engineering (2016)
9. Petrasch, R., Hentschke, R.: Towards an internet-of-things-aware process modeling method - an example for a house surveillance system. In: Management and Innovation Technology International Conference (2015)
10. Graja, I., et al.: BPMN4CPS: a BPMN extension for modeling cyber-physical systems. In: WETICE, pp. 152–157. IEEE (2016)
11. Meyer, S., Ruppen, A., Magerkurth, C.: Internet of things-aware process modeling: integrating IoT devices as business process resources. In: Salinesi, C., Norrie, M.C., Pastor, Ó. (eds.) CAiSE 2013. LNCS, vol. 7908, pp. 84–98. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38709-8_6

12. Meyer, S., Ruppen, A., Hilty, L.: The things of the internet of things in BPMN. In: Persson, A., Stirna, J. (eds.) CAiSE 2015. LNBIP, vol. 215, pp. 285–297. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19243-7_27
13. Sperner, K., Meyer, S., Magerkurth, C.: Introducing entity-based concepts to business process modeling. In: Dijkman, R., Hofstetter, J., Koehler, J. (eds.) BPMN 2011. LNBIP, vol. 95, pp. 166–171. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25160-3_17
14. Domingos, D., Martins, F., Cândido, C., Martinho, R.: Internet of things aware WS-BPEL business processes context variables and expected exceptions. *J. Univ. Comput. Sci.* **20**(8), 1109–1129 (2014)
15. George, A.A.: Providing context in WS-BPEL processes. Master’s thesis, University of Waterloo (2008)
16. George, A.A., Ward, P.A.: An architecture for providing context in WS-BPEL processes. In: Conference of Advanced Studies on Collaborative Research (2008)
17. Wieland, M., Kopp, O., Nicklas, D., Leymann, F.: Towards context-aware workflows. In: CAiSE Workshops and Doctoral Consortium, vol. 2, p. 25 (2007)
18. Mateo, J.A., Valero, V., Diaz, G.: BPEL-RF: a formal framework for BPEL orchestrations integrating distributed resources, preprint [arXiv:1203.1760](https://arxiv.org/abs/1203.1760) (2012)
19. Schmidt, B., Schief, M.: Towards agile business processes based on the Internet of Things. In: Advanced Manufacturing and Sustainable Logistics, pp. 257–262 (2010)
20. Schobel, J., Pryss, R., Schickler, M., Reichert, M.: A lightweight process engine for enabling advanced mobile applications. In: Debruyne, C., Panetto, H., Meersman, R., Dillon, T., Kühn, E., O’Sullivan, D., Ardagna, C.A. (eds.) OTM 2016. LNCS, vol. 10033, pp. 552–569. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48472-3_33