ARPF - an Augmented Reality Process Framework for Context-Aware Process Execution in Industry 4.0 Processes

Gregor Grambow, Daniel Hieber, Roy Oberhauser and Camil Pogolski

Dept. of Computer Science
Aalen University
Aalen, Germany
e-mail: {gregor.grambow, daniel.hieber, roy.oberhauser, camil.pogolski}@hs-aalen.de

***Abstract*—Although production processes in Industry 4.0 settings are highly automated, many complicated tasks, such as machine maintenance, continue to be executed by human workers. While smart factories can provide these workers with some digitalization support via Augmented Reality (AR) devices, these AR tasks depend on many contextual factors, such as live data feeds from machines in view, or current work safety conditions. Although currently feasible, these localized contextual factors are mostly not well-integrated into the global production process, which can result in various problems such as suboptimal task assignment, over-exposure of workers to hazards such as noise or heat, or delays in the production process. Current Business Process Management (BPM) Systems (BPMS) were not particularly designed to consider and integrate context-aware factors during planning and execution. This paper describes the AR-Process Framework (ARPF) for extending a BPMS to support context-integrated modeling and execution of processes with AR tasks in industrial use cases. Our realization shows how the ARPF can be easily integrated with prevalent BPMS. Our evaluation findings from a simulation scenario indicate that ARPF can improve Industry 4.0 processes with regard to AR task execution quality and cost savings.**

Keywords*—*Business Process Management Systems; Augmented Reality; Fuzzy Logic; Business Process Modeling Notation; Resource Assignment Automation.

I. Introduction

Today's manufacturing industry heavily relies on smart factories which enable a better customer orientation as well as more efficient and individual production. However, despite the focus on a high automation level and the utilization of autonomous systems, human involvement in complex processes still plays a crucial part. Human workers often have to make important decisions or perform complex tasks, such as machine maintenance.

This paper extends our previous work [1], where we introduced ARPF, that aims to holistically integrate AR worker-based and production processes, utilizing Industry 4.0 smart factories with their Cyber-Physical Systems (CPS) [2], leveraging their wide range of sensor capabilities to provide context-based AR support for human tasks. This not only provides support to the worker itself, during task execution, but also enables comprehensive optimization of production with regard to criteria, such as costs, resource consumption, quality or availability.

While using AR devices to support complex tasks executed by humans is no novelty, the integration of such activities in the global production process remains a challenge. A primary reason for this is that human AR tasks depend on a large number of factors that are not typically represented in the overall higher-level business and production process. This includes the following factors:

- The AR tasks rely on different contextual data sets, e.g., external information sources supporting task execution, such as maintenance manuals, alternative procedures, checklist variability, live data from external systems or sensors of machines, the task executor and their decisions, and context-sensitive AR data such as the relative position of the worker or the machine.
- For maximal effectivity and efficiency, the task must be assigned to the most suitable worker. Simple Staff Assignment Rules (SARs) of contemporary BPMS governing the production processes are only capable of determining if an agent is able to perform a task, but not their level of suitability. For AR tasks in complex Industry 4.0 settings, however, many parameters should be taken into account, such as the position of the worker and the task, the qualification of the worker, or the workload of each worker. Otherwise, task execution might be suboptimal or too expensive, e.g., because of overqualification of the worker or long distances between him and the task. Furthermore, worker safety is usually enforced by government regulations and workers' exposure to hazards such as heat, noise, and danger must be taken into account.
- Usually, workers processing AR tasks are able to communicate via the AR device. However, as the AR tasks are not integrated with the global process, decisions or information provided by the worker cannot be directly utilized in that process, leading to delays or incorrect activity choices.

Contemporary BPMS lack facilities for representing and exploiting such data sets as well as contextual factors. Usually, these systems utilize standard BPM languages such as the Business Process Modeling Notation (BPMN) [3], which were not designed to integrate such information into the process templates. Subsequently, live data and situational knowledge cannot be readily utilized in the process instances based on such process templates.

In prior work we developed an approach for contextual process management [4][5][6] which was tailored towards software engineering processes and did not involve the complex specifics of Industry 4.0 nor AR processes. To overcome the aforementioned limitations, we contribute ARPF, an integrated framework extending current BPMS with the following features:

1) Facilities to model processes that incorporate contextual factors applicable to human AR tasks.
2) Incorporating real-time context data in BPM processes to enable automated context-dependent decision and execution support.
3) An interactive AR activity interface for such processes, enabling bi-directional communication between the process and the AR-supported worker.
4) An intelligent task assignment component capable of utilizing contextual data for fine-grained suitability levels, able to optimally assign workers to tasks.
5) Easy extension of existing BPMS.

This paper extends our previous work [1] with an expanded description providing further AR details and extending the evaluation and related work.

The remainder of this paper is structured as follows: Section II describes the concept and solution approach, while Section III provides realization details. Thereafter, Section IV evaluates the technical capabilities of the implemented system and the empirical results of its usage with AR users. Section V elaborates the background of the research as well as related work. Finally, Section VI provides a conclusion and outlook on future work.

II. Solution Approach

This section describes our concept for a context-aware system with AR support, called AR-Process (ARP) Framework (ARPF). It is conceived as a generic extension that any BPMS can readily integrate, providing facilities for representing contextual and AR information in executable processes in conjunction with an enactment component.

A. Contextual Processes

To enable the application of the ARPF in both new and existing processes and enable easy integration in any BPMS, the contextual information will be integrated into the processes via a generic BPMN 2.0 extension. Extending the BPMN standard not only allows an easy integration (requirement 2), but also allows the reuse of the existing BPMN service and script activities [7], heavily reducing implementation efforts. Such activities provide an intuitive interface between the BPMS and the ARPF. With this approach it is possible to decouple the ARPF from the process itself and provide it as a service to any BPMS supporting BPMN 2.0.

In the following, we will elaborate on the context data and rules or conditions crucial for contextual ARP execution. The context is separated into three major parts: global, process, and activity. A model can be seen in Figure 1.

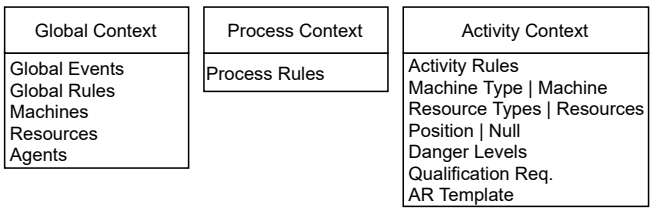

Figure 1. Context data model.

The *global context* represents a cross-process entity containing all required global information. This includes information about different entities external to the BPMS. In particular, all machines, resources, and available agents. Further, the global context should provide facilities for defining conditions regarding the context that must be verified and fulfilled before an activity can be executed (requirement 2), e.g., check if an agent has the required danger clearance for an activity. This is realized by a *global rule set*. Another important factor is external context information that must be provided to the ARPF, e.g., priority changes for customer orders. Such data is incorporated via a global event system. In this manner, real-time context integration into BPM processes on a global level (requirement 2) can be achieved.

In addition to such global information, each process type may also have specific contextual conditions, e.g., if a specific process should only access a subset of the available machinery or use only special types of tools. To achieve this, a *process context* is employed that can overlay applicable portions of the global context if required. It further contains an additional *process rule set*. The latter is similar to the global rule set, but is limited to the processes of this type.

The third important entities requiring contextual information are concrete activities. To support these, an *activity context* is defined. It contains specific information for a single activity of a specific activity type and can be further specified during the ARP execution. As for the process and global context, an *activity rule set* is present to enable fine grained conditions on the activity level. On this level, however, a set of additional contextual information is required to enable an efficient assignment of the best suitable worker for each task. This incorporates data such as the danger levels the task may involve, the qualification to successfully complete it (both defined as a dynamic set of key value pairs, containing values between 0 to 1), and the position of the activity represented by a three-dimensional vector X, Y, Z. Machine types can also be defined, as well as additional resources required for the activity, allowing the inclusion of machine context data directly into the process. Finally, the information, which AR-Component should be displayed to the worker while executing the activity (requirement 3) must be present. This is achieved by the AR Template.

B. Data Models

In addition to the contextual information added to the processes that governs how activities should be executed effi-

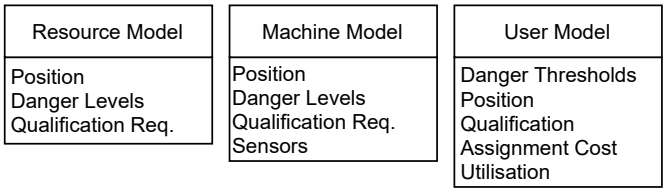

Figure 2. Actor models.

ciently, the ARPF also requires information about the physical entities involved in process execution. In particular, three entities are crucial: the workers, the machines where activities (e.g., a maintenance task) are executed and their position, and resources required for such activities (e.g., materials, tools). To provide such information to the ARPF, three models are created, which can be found in Figure 2.

As simple BPM engines do not provide entries for resources and machines, new models must be created. Both contain a position, connected danger levels (e.g., noise with machines or chemical hazards with resources), and the required qualification to safely and efficiently work with the machine/resource. Machines usually also contain sensors providing real-time information about important production parameters. These are also included in the model.

Finally, the BPMS user/agent concept has to be extended, as current BPMS lack sufficient information to support AR activities intelligently. In order to assign agents to activities, the BPMS must possess compatible models. This can be achieved by extending the agent model of a BPM engine with values for position, qualification, and danger thresholds. Further, in most cases, cost-effective activity execution is also a requirement. Therefore, we incorporate information about the additional cost of an assignment of this agent (e.g., salary of an external worker, or weekend surcharge if it is not part of the contract) and the current utilization of the worker to avoid unbalanced workloads.

C. Process Modeling

In current BPMSs, there are rather limited and generic facilities to add context data to processes. This concerns the process modeling tools as well as the processes themselves. To overcome these limitations, our approach for adding contextual data for ARP execution is to realize such datasets as an extension for the most prevalent process language currently, BPMN 2.0. That way, the integration in a BPMS can be readily achieved, as any BPMN 2.0 compatible modeling tool can be easily extended (cf. requirement 1). To show the feasibility of this approach, we implemented a prototype extension integrated into a prevalent BPM modeling tool. Further details can be found in [8].

With an extended modeling tool at hand, a process engineer can add all contextual information and dependencies crucial for ARP execution support to new as well as existing process models without programming knowledge. Users, machines, and additional resources can be specified, including relevant parameters such as their position. With appropriate data structures in place, relevant live data (e.g., from machine sensors) can be incorporated in the processes as they are executed. This data, in turn, is utilized by the components of the ARPF to provide more efficient task assignments and more effective support of the AR activities in the process.

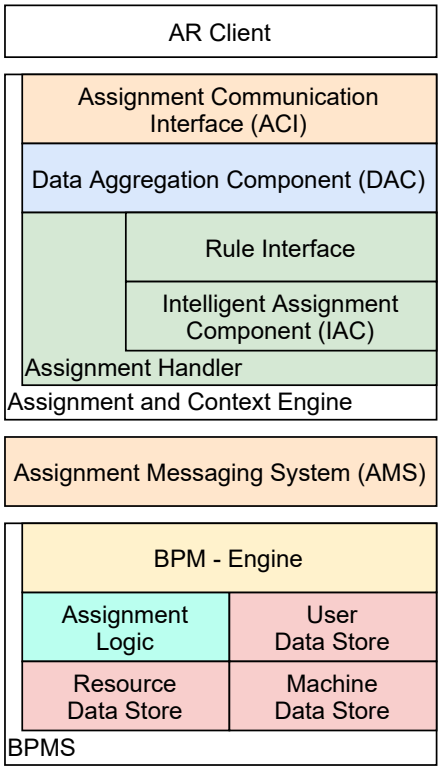

Figure 3. ARPF concept architecture.

D. AR - Process Enactment

The core architecture of the ARPF for contextual ARP enactment is depicted in Figure 3.

The core component of ARP process enactment is the Assignment and Context Engine (ACE). To provide a generalized and independent solution, this component is decoupled from the utilized BPMS. This permits a finer engineering of the ACE independent of the utilized suite. To achieve this, the ARPF incorporates two language- and platform-neutral generic communication components. The Assignment Communication Interface (ACI) enables communication between the ACE and both the BPMS, as well as the client software on the AR devices, while the Assignment Messaging System (AMS) manages live data from the ARPF environment. That way, the ACE can be realized independent of any preexisting programming language or BPMS limitations. This allows the usage of the ARPF with a wide range of existing BPMS (requirement 5).

Many BPMS are provided as a standalone BPM engine (e.g., Camunda [9], jBPM) and therefore require external software to build a fully functional BPMS capable of managing all crucial data sets for contextual process enactment. To overcome these limitations and provide an easy way to extend BPM engines,

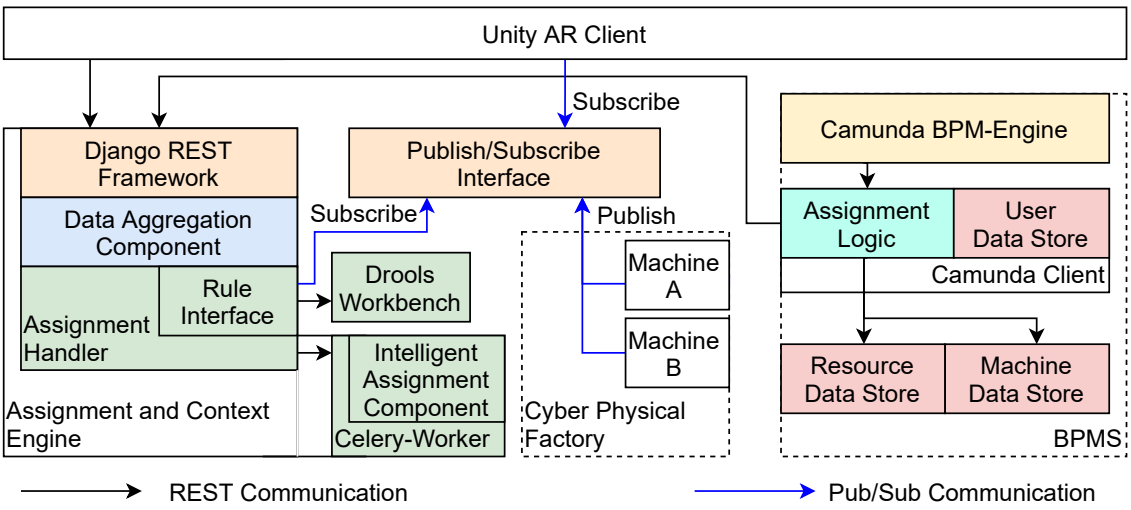

Figure 4. ARPF camunda implementation.

we provide three generic Data Stores (DS): a User-DS, a Resource-DS, and a Machine-DS. These contain additional context information as specified by the aforementioned data models (cf. Figure 2), such as a more refined user model, information about all machines used in the factory, and resources required to complete tasks. This extended context data is required in the assignment process and during the activity execution in the AR-Client.

The Assignment Logic Component (ALC) of the BPM engine is used as a bridge between the engine and ACE. It aggregates all required context data for assignments; however, it is also possible to integrate the assignment request completely in the process itself via service or script tasks defined in the BPMN 2.0 standard [7] (requirement 5).

If an assignment request is sent to the ACE via the communication interface, the request is forwarded to the Data Aggregation Component (DAC) and validated for completeness. If some required context data is missing, the DAC will request it from the corresponding DS. Afterwards, the assignment request is forwarded to the Assignment Handler. The handler can then calculate a specific assignment score for the requested activity and agents in the Intelligent Assignment Component (requirement 4). If required, a presorting can be applied in a rule engine via the Rule Interface. Further preconditions of assignments (e.g., only assign the task if a sensor value is below a certain threshold) can be handled by the rule interface. To guarantee an optimal fine granular assignment score calculation fuzzy sets are utilized [10]. In this case, these are to be preferred to other solutions like Machine Learning (ML) and chaining. In contrast to ML approaches, with fuzzy sets no preexisting data sets are required nor is a training phase required, as weights can be defined directly and transparently according to the user's own knowledge and experience. Further, a fine granular calculated score between 0 and 1 is possible instead of the simple true or false of a chaining approach. Further details on our algorithm and its implementation can be found in [11].

The final component of the ARPF is the AR-Client. The latter should be implemented as generic as possible to be available to a wide range of AR devices, e.g., tablets, goggles and even smartphones. The client is able to request all relevant process data via the ACI, and activities can be started, executed and completed in the AR-Client without the need to change to another software client e.g., a PC-interface, or web-client (requirement 3). Thanks to the provided AMS, it is further possible to consume real-time context changes on multiple levels (e.g., a global change of activity priorities or sensor data send from a machine connected to the activity being executed) (requirement 2).

III. Realization

This section describes the technical realization of the ARPF. It further details the communication between the components. While this section describes its integration with Camunda as a BPM engine and the AristaFlow BPM Suite [12] to demonstrate its capabilities with two mature and prevalent BPMS, the framework can be used with all BPM-Engines supporting REST-calls or external code execution either via extensions or script tasks. The provided AR-Client can further be used with a majority of current AR-devices.

The prototype was implemented using Python due to its rapid prototyping capabilities and large spectrum of available libraries. As a base image for the ACE, a Django server was used which can be readily scaled for production deployment. To implement the ACI, the Django REST framework was integrated, providing a REST interface on top of the Django service. For the AMS, handling the real-time machine sensor communication, the Publish/Subscribe (Pub/Sub) system MQTT [13] was chosen, utilizing the Eclipse Mosquitto broker as the main component. As both technologies use well-defined industrial standards, an easy integration in BPMS is supported.

Figure 4 shows the architecture for the implementation of the ARPF with Camunda. Compared to the concept from Figure 3, some minor changes were made and the communication

specified. The implementation of the AristaFlow BPM-Suite follows the same base architecture; however, the full suite is provided by AristaFlow, removing the need for our own Data-Stores or an ALC. The communication is symbolized by colored arrows in Figure 4.

While the AristaFlow Suite does not require extensions, the Camunda solution requires implementation of a minimal BPM-Suite around the engine itself. This could either be realized as a single Java application relying on the Camunda Java API or using REST. In order to stay consistent with the general architecture, we use REST for our minimal BPM-Suite and split it into three sections. The Camunda BPM engine in its base version, a Camunda Client Django server containing the Assignment Logic, and the User-DS as well as a final Django Server containing the Resource and Machine Data Store. In order to connect the ARPF to a BPMN process template, it is required to create a service or script task sending a REST call to the Camunda Client. This call must contain the process instance id that can be acquired during process runtime in the same activity. During the process execution, the Camunda engine then calls the Assignment Logic via the created activity and triggers the assignment process. The Assignment Logic confirms the request to the engine and then spawns a new process handling the request. It then aggregates all data required for this assignment and sends an assignment request to ACE.

The Django REST Framework based ACI receives the assignment request and then executes the data aggregation component, validating that all required data for an assignment is available. In the Camunda implementation all required data is already present, in the AristaFlow implementation, the required data can be received from predefined endpoints. Afterwards the Assignment Handler is called. If preconditions are implemented (e.g., confirming the temperature of a machine sensor), the Rule Interface takes action. It first subscribes to all required machine sensor data endpoints via the Mosquitto Broker and then calls the connected rule engine via REST. In the implementation Drools [14] is used for the Camunda Implementation while AristaFlow provides its own XPath based solution. The preconditions can either be run in a loop (e.g., waiting for a sensor to cool down) until the condition is fulfilled, or in single-shot mode, aborting the assignment if the check is negative.

If the assignment is aborted, a response is sent to the Camunda Client/AristaFlow suite which are then required to provide a fallback plan, e.g., a retry after some time, a fallback process, or human intervention.

If the preconditions have been fulfilled, the assignment request is forwarded to the Intelligent Assignment Component (IAC), which itself is detached from the ACE to a celery worker. Utilizing the Celery Python framework, all assignment calculations are outsourced from the ACE and do not bind resources, therefore the I/O operations of the backend are not affected, even if many concurrent assignments are calculated. Each fuzzy assignment calculation is assigned its own processor for optimal execution speed. After the

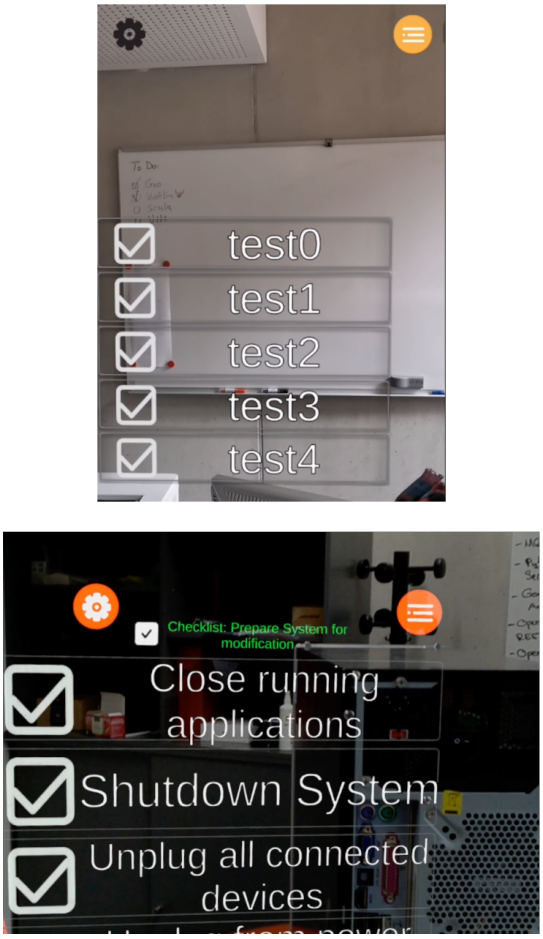

Figure 5. AR-Client, top table/mobile phone version, bottom Magic Leap 1 version.

calculation is finished the IAC sends the assignment to the Celery Client/AristaFlow Suite handling the assignment update in the BPM engine.

The AR-Client is implemented using the Unity AR Foundation framework; this allows the creation of a generic AR frontend usable with a majority of present AR devices such as AR goggles, tables, or phones. Figure 5 shows an early version for the tablet/mobile phone user interface, as well a later version for the Magic Leap 1. Instead of communication with the BPM-Suite itself, the AR-Client communicates via REST with the ACE and all requests to other sources are handled by the ACE. This enables the creation of a truly generic frontend independent of the BPMS, as all requests are parsed to the required model in the ACE. With this approach combined with a powerful AR interface, the user is able to complete and perform all activities in the AR-Client without the need to utilize another software solution or device. As the BPM workflow is still handled solely by the BPMS, it is however possible to switch at any moment to another solution (e.g., the Camunda Tasklist or the AristaFlow Client) if the worker deems it more beneficial, e.g., filling a long form or accessing specific resources.

While all process management communication is handled

via REST between the AR-Client and ACE, the client can also access the Pub/Sub data via the Pub/Sub interface. It is therefore possible to see all relevant sensor values of a machine while working on it, or receive global updates (e.g., a change of priority or information a new assignment).

The final component of the ARPF is the Pub/Sub Interface, handling all MQTT messages. This contains all machines sensor data for the Rule Interface or the AR-Client, as well as global worker specific updates such as a new assignment or priority updates. While the Camunda Client makes no use of global events via MQTT, the event feature is implemented in the AristaFlow suite.

In our prototype a Cyber Physical Factory is simulated using the OPC-UA protocol to connect machines' sensor data to the ARPF. As OPC-UA supports MQTT, this can be achieved in an easy and generic way, further easing the implementation into existing production environments.

To enable the creation of context-aware processes, a new BPMN modeler is created as an extension of the open source Camunda Modeler. While it is possible to create processes using the ARPF with any BPMN 2.0 modeler, a specific implementation comes with certain advantages. The modeler is linked to the different data stores and can therefore display all available machines, resources, and workers as specific entities or groups (e.g., CNC mill, maintenance workers, etc.) during the modeling of processes. This allows the process engineer to easily include the context during process creation. Further, it is possible to see available rules of the connected rule engine, enabling their integration as preconditions to activities. Moreover, the ARPF specific assignment request is moved to the background and no longer visible in the BPM template as a separate task. Removing them from the user visibility greatly reduces the potential for an overloaded user interface and directs the focus on the more relevant elements.

IV. Evaluation

ARPF was evaluated following a bipartite approach. In the first evaluation, ARPF technical capabilities were evaluated in a simulation environment. This approach was chosen over a real factory test environment for the benefit of a safe and more controlled environment, easy reproducibility, and providing a large set of test runs. The second part of the evaluation consisted of an empirical evaluation focusing on the AR interface.

Both evaluations used a test setup integrating Camunda as the BPM engine and Drools as the rule engine.

A. Simulation of Worker Activities in an Industry 4.0 Setting

The complete framework was deployed on a virtual Linux server with 90GB main memory. However, the memory consumption never exceeded 24GB during our evaluation and can easily be halved by removing the Drools rule engine. The AnyLogic simulation was run on a Lenovo T495 with 14GB main memory utilizing Arch Linux as an operating system. To simulate values for the machine sensor, an OPC-UA server was hosted, utilizing a common industrial standard for this use case.

The evaluation was used to compare a BPMS using the ARPF against a plain BPM engine. To simulate workers and a realistic workflow, an AnyLogic simulation model was created and two simulation setups were configured.

As an environment, a factory with $21504m^2$ and a total of 29 machines requiring maintenance every 16 hours were created. The first maintenance was scheduled between 0 to 16 hours after start of the simulation. Further, the machines had an average breakdown interval of 36 hours. If a machine required maintenance or repair, a new Camunda process instance with the required worker qualification and the machine's position was started. The activity takes between 1 to 3 hours and requires an engineering qualification of 4 for maintenance and 6 for repairs. Other qualifications (electric, computer, bio_chemical) were not required and set to 0. As most modern manufacturing environments contain hazards requiring special training and regulations dangers were implemented in the simulation represented by values for noise: 0.01, heat: 0.03, electricity: 0.05, and chemicals: 0.02. While these values are quite abstract, they can easily be further refined and specified. A total of 5 workers (the agents in this use case) were available to complete these activities. Four internal workers, waiting in a maintenance building in the factory hall and one external worker, waiting 165 meters away. The external worker is used to display the need for highly trained personal which often has to be contracted by external service providers. The internal workers had engineering qualifications of 4, 5, 6 and 7 while the external worker had an engineering qualification of 8. The other qualification values were set to 0 to avoid bias. Their danger thresholds were set to 0.7 for all values. The usage of the external worker further was connected to an additional cost of 25000 (250€/activity), while the usage of internal workers incurred no additional costs. In their idle state, a worker checked every 5 minutes if a new activity was available. If they were working, they immediately checked after completion of their current activity for another enqueued activity. If no activity was enqueued, they switched back to the idle state and moved to their starting position. The simulation was separated into 5 work-shifts (each 8 hours long) with a break of 4 hours between shifts. During this break, workers were allowed to complete their current activity, but could not start new ones nor was it possible for machines to create a new task during the break. At the beginning of each work-shift, all tasks are reassigned and the danger thresholds of workers are reset to their default.

In the Camunda Setup (called CMD-Setup in the following), the workers fetched their activities directly from Camunda. All activities of the simulation were available to all of the workers and no further verification performed. If an activity is available to the group, the workers try to claim it and, if successful, work on it. In the ARPF Setup (further called ARP-Setup), the workers checked their personal worklist at the Assignment Engine REST API. If their personal worklist contains an activity, they start to work on it, otherwise they

TABLE I. ANYLOGIC ARP EVALUATION.

	ARP	Camunda
work_time	2103.31	2310.60
idle_time	524.49	396.38
avg_overqual	0.12	0.08
avg_tasks_day	3.52	3.62
violations	0.00	5.12
traveled_distance	9304.40	9502.27
cost	2000.00€	4600.00€
max_avg_underqual	0.00	-0.02
downtime_maintain	439.83	293.14
downtime_repair	218.90	249.32

TABLE II. ANYLOGIC WORKER EVALUATION.

	ARP-int	ARP-ext	CMD-int	CMD-ext
work_time	2358.93	1080.83	2336.76	2205.95
idle_time	324.34	1325.07	381.82	454.65
avg_overqual	0.04	0.42	0.05	0.19
cost	0.00	200.00	0.00	460.00
avg_tasks_day	4.00	1.60	3.60	3.68
violations	0.00	0.00	4.88	6.10
traveled_distance	9952.32	6712.70	9386.80	9964.14
max_avg_underqual	0.00	0.00	-0.02	0.00

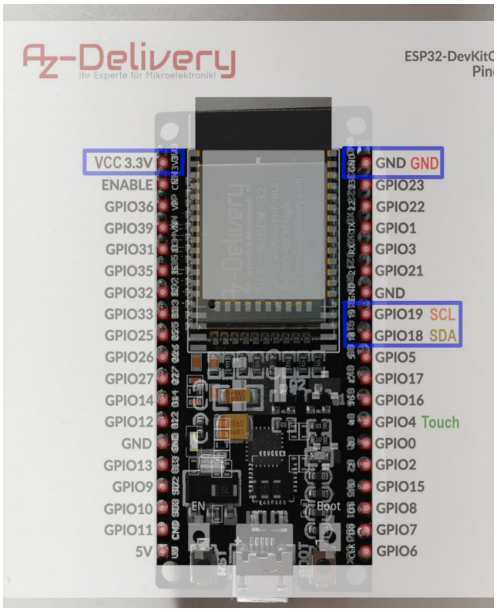

Figure 6. ESP32 with AR overlay.

remained idle.

The five workdays were simulated for both configurations, using the same seed for the simulations random number generator. This process was repeated 10 times with different seeds to get statistically relevant test data. For the ARPF the model introduced in Section III was used. The qualification value was weighted half, to increase utilization of the more qualified workers and reduce the downtime of the machines. Further adjustment of the weighting could lead to heavily deviating results. An optimal weighting has to be configured according to the needs of the activities.

Table I shows a general comparison between the CMD and ARP simulation, while Table II shows a detailed comparison of internal and external worker stats in both simulations. In the following, values from Table I will be discussed and argued with the values from Table II.

The average work time and total activities per worker are lower in the ARP run, while the utilization of the internal workers (ARP-int) is slightly increased and the external utilization (ARP-ext) is heavily reduced. The average idle time is increased, which results from the low external utilization. The heavily reduced average cost of a simulation run, if using the ARPF instead of a plain BPM engine is due to the preferred use of internal workers. The increase in overqualification while using ARP instead of plain Camunda can be explained with the low weighting of qualification in the algorithms and no under-qualification, in opposition to the CMD-Setup, where under-qualification was generally present (to make it more realistic, under-qualified workers required 60 minutes longer than qualified workers). Taking a look at Table II, the main source of overqualification in the ARP simulation comes from the usage of the external worker, who was mainly used for activities below their qualification. This happened because the workload was too high and could be resolved by employing another internal worker with lower qualification to help out with these activities. This would lead to reduced costs and downtime. Optimization in the simulation or company values is needed rather than an adaptation of the algorithm.

The traveled distance for the internal workers is slightly increased in the ARP simulation compared to the CMD run. This correlates with the increased workload, and a stronger weight regarding the distance could reduce this effect. While the time for maintenance in the ARP run is around 40% higher than in the CMD-Setup, the actual down time for repairs could be reduced. This would increase the overall efficiency, as machines scheduled for maintenance still function properly while fast intervention is required on broken down machines. Further, the cost could be reduced to 43% of the CMD-Setup.

While the ARPF also utilized rules via Drools to validate if the work on the machine was safe by checking the values of the machine's temperature sensor against a max threshold, the base BPM engine did not provide such features. Violations against this precondition can be found under violations in Table II. In a real environment this would either lead to a safety regulation violation or would require a change of tasks for the worker, leading to even lower performance.

Finally, the ARPF could support workers more efficiently with their tasks, as it displays AR-instructions according to the qualification of the user. This could lead to a further speedup which has to be evaluated in a real-world setup.

Concluding, the ARPF worked as expected and the IAC produced comprehensible results. The utilization of ARPF in the simulations reduced the downtime of machines through failure and prevented any safety regulation violations.

B. Empirical Evaluation with AR Users

In order to gather empirical insights about the AR interface while guaranteeing compliant execution regarding the covid restriction present at the time, we conducted the AR evaluation

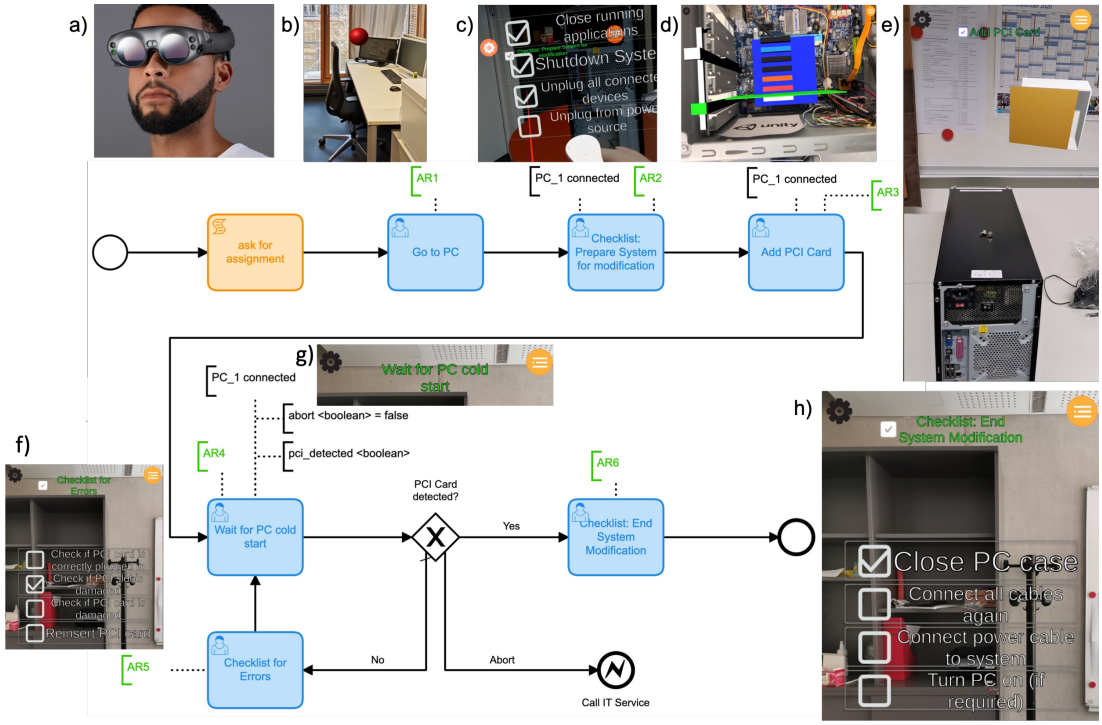

Figure 7. AR evaluation workflow display.

with 22 computer science and electrical engineering bachelor students in the semesters 2-8. Of these subjects, only 3 had advanced experience with AR devices, seven had no prior AR experience at all, ten had used an AR device before (two of which had used AR devices multiple times but possessed no advanced AR usage competency). The evaluation consisted of two use cases which had to be completed with AR support and with instructions on paper, one half starting with paper, the other half with AR. The two tasks were:

1) Connecting an ESP32 controller to a display
2) Installing a PCI card in a tower PC

In our pre-evaluation, the usage of a handheld AR device (phone or tablet) proved to be a hinderance in this use cases, as the AR device's camera had to be focused on the object to provide guidance, while, at the same time, both hands were required to perform the tasks efficiently. To overcome this hindrance, we selected a smart goggle (Magic Leap 1) for the final evaluation. With this approach the subjects could move the target-object into focus by looking at it, enabling the AR overlay, and had both hands at their disposal at the same time.

In use case 1 AR was primarily used to support the subjects by providing an overlay on top of the ESP32 controller, highlighting the required pins (cf. Figure 6). Further, the subjects were able to work through the BPM process only using the AR device, completing tasks and inputting data directly into the BPM engine. This was meant to acclimate the subjects with the technology.

Figure 7 depicts the process template for use case 2. In this scenario, an additional part (PCI card) has to be installed into a PC, analogous to adding or replacing a machine part in an industrial scenario. As with use case 1, the subjects were able to directly interact with the process through their Magic Leap 1 (Figure 7a) and its pointing device, which could be hung on a belt if not required.

The following explains the separate workflow steps of use case 2 in detail:

1) "ask for assignment" script task: the BPM-Engine automatically triggers the IAC via the ARPFs ACI, at instance start, to determine the optimal worker for the process instance and assigns the selected worker in the BPMS (in this case the test subject)
2) The subject is notified of the new assignment in the AR App, via a red dot at the menu in the top right corner
3) "Go to PC" [AR1 task]: red spheres (anchors) are used to guide the subject to the destination (Figure 7b)
4) "Checklist" [AR2 task]: displays a pre modification checklist (Figure 7c), disconnecting power and other safety measures.
5) "Add PCI Card" [AR3 task]: AR-Video-Overlay is shown on how to open the PC (Figure 7e) and how to install the PCI card (Figure 7d). Afterwards the PC gets reconnected to power and restarts.
6) "Wait" [AR4 task]: subject awaits the startup and automatic system check (Figure 7g). This was mocked for the evaluation.
7) "Checklist" [AR5 task]: if an error was detected the subject was provided with a trouble shooting checklist (Figure 7f)
8) "Checklist" [AR6 task]: if no errors occurred the completion checklist (Figure 7h) is shown.

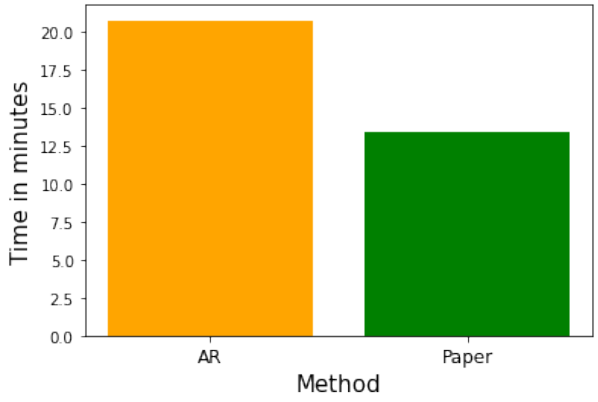

Figure 8. Task processing speed both evaluations combined.

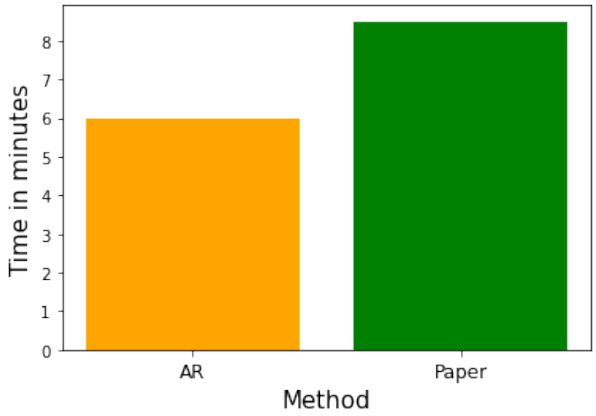

Figure 9. Task processing speed AR5 task in evaluation 2.

All subjects were able to complete the use cases in AR and with paper instructions, the starting order (AR first vs paper first) made no significant difference in the working speed. Overall, the conduction of all tasks (both use cases combined) with AR took around 50% longer than the completion of all tasks using the paper method as depicted in Figure 8. However, the handling of the second checklist ([AR 5]), was executed faster in AR than the paper version, cf. Figure 9. Further the subjects tended to forget check off points on paper more often than in the AR app, heavily reducing the error rate from 31% on paper to 10% using AR.

Most of the problems with the AR app further originated from problems with the precise anchoring of the overlays and the usage of the pointing device (which had to be put down and picked up again) rather than gesture control.

Overall, the AR approach proved slower for the people with no/little AR experience and requires further work, but people recognize its potential usefulness. This checks up with other AR projects, e.g., WART [15].

C. Findings and Discussion

While both evaluations resulted in positive findings, ARPF clearly isn't suitable for every use case nor user group. Although the framework itself provides powerful and efficient assignment algorithms and can be tweaked to the specific need of the BPM setup, some setup is still required. This includes specifying the requirements and dangers for each task, adding the additionally required agent data (e.g., their qualification), etc. The more complex the processes are and the more agents exist in the system the more setup time is involved. However, the value of ARPF increases as more agents are available for tasks, as it can provide error-less exact assignments for a large set of agents. The initial setup process can further be accelerated by using templates for tasks, with default requirements already specified, and only manually tweaking special tasks with hard constraints, such as hazards or hard requirements regarding agent qualification.

Furthermore, the evaluation has shown that the AR support can greatly reduce the error rate and accelerate the execution of suitable tasks. However, the AR support can also lead to slowdowns of tasks if human agents are not familiar with AR devices or the tasks are not suited for AR. This makes ARPF vulnerable for the golden hammer anti-pattern [16]. Therefore, a thoughtful selection has to take place where to use AR support and where to rely on the traditional methods.

Overall the evaluation shows that ARPF can be an powerful tool and extension to classical BPMSs when applied correctly. Small companies most likely would not profit from the assignment component, as the initial setup would be cost prohibitive. Medium/large companies, however, could greatly benefit from the more sophisticated assignment process. The AR support has to be applied with care to suitable tasks, but used correctly could be beneficial for companies no matter their size.

V. Related Work

Generalized context models are difficult to achieve and are not prevalent, as a survey on context models conclude [17]. An example is presented in [18]. The model is heavily tailored towards general pervasive computing scenarios and lacks several components crucial for Industry 4.0 AR processes. In contrast, the ARPF context model presented is rather specific and yet readily extensible, due to its three-layer context based on global, process, and activity context. Furthermore, the integration of context into process languages is challenging because they are not flexible enough, as stated in [19]. The contribution also proposes a BPMN extension for context integration, which is, in turn, tailored heavily towards mobile processes and not suitable for Industry 4.0 production.

Focused on context processing the Java Context Aware Framework [20] is a technical object- and service-oriented framework targeting modeling context changes via rules. However, the processing of such rules is forwarded to the application layer. In JCOOLS [21], this limitation is overcome by integrating JCAF with the Drools rule engine. The approach taken is rather complicated and generic, lacking support for both programmers and end users.

Examples of context modeling approaches include Coutaz and Crowley [22] and Ghiani, Manca, and Paternò [23]. However, these approaches primarily target the creation of context

rules by the application developer that can later be completed with concrete values by end users, without providing the execution infrastructure.

There are also contextual approaches for Industry 4.0 production. Giustozzi et al. [24] provide a context model for industry 4.0 processes. Some of the mentioned entities are similar to the ones in the ARPF. However, the model is ontology-based and the paper primarily deals with logical relations of the concepts, which makes concrete implementation in an industry-ready system problematic. Furthermore, only a model is presented, lacking other components for integration process enactment. BPMN4CPS [25] combines BPMN with CPS to add resources and context data to a business process for increased automation, but it does not integrate AR directly. Another model for Industry 4.0 production based on ML is presented in [26]. This model, however, is also not applicable for enactment of AR processes, as it primarily deals with predicting the degradation of the state of machines.

Another approach is taken by Tasdemir and Toklu [27]: it focusses on fuzzy task assignment and integrates BPM concepts. The described system is not suitable for the Industry 4.0 scenario, as it focuses on teams and the social relationships of the worker in the team. In addition, it lacks other components such as a real-time data context model.

Work related to the combination of context with AR tasks includes Blattgerste et al. [28], where AR glasses provide mobile assistive instructions. However, it was largely restricted to one concrete scenario rather than a generic business process. In BPMN4SGA [29] BPMN is extended for Smart Glasses, but primarily for documentation purposes rather than actionable AR content. In contrast, in our approach AR Actions are modeled and implemented via predefined AR templates containing attributes covering nearly all BPMN elements. Our AR application interprets the templates and sends feedback to the BPMN modeling application, avoiding the necessity of implementing or syncing steps with the BPM engine. SenSoMod [30] adds context-awareness to conventional non-production applications such as email, calendar, etc. Gronau & Grum [31] combined the Knowledge Modeling and Description Language (KMDL) with AR, projecting sensor data and process step association onto the visible machines, yet it lacks concrete tailored task guidance. HoloFlows [32] is an AR process modeling approach for the Internet of Things (IoT), utilizing a simple state-machine and custom notation that lacks BPMN support and integration with mature BPMS - vital for production settings.

In summary, ARPF provides a unique approach for contextual processing for Industry 4.0 processes with human AR tasks, supporting integration with existing BPMS and utilizing a BPMN extension to include AR and context in new and existing process models. Other approaches lack the inclusion of information needed for representing processes and their connection to AR devices and workers, machines, and resources with their specific contextual properties and rules. In addition, most of these approaches do not present an integrated framework for comprehensively supporting process enactment in such complicated domains utilizing real-time data.

VI. Conclusion

This contribution described our ARPF approach for incorporating contextual factors crucial for AR tasks into Industry 4.0 production processes. The presented framework incorporates components to simplify the integration of such factors when modeling the processes and utilizes live data from different sources while executing them. This enables context-aware process enactment, which can improve process quality capabilities such as optimal task resource assignments, improved cost efficiency, and better support for AR user activities. Furthermore, by providing bi-directional communication interfaces between the process and the AR task, the latter can be seamlessly integrated into the process.

We further implemented a prototype integrating our approach with two prevalent BPMS. The prototype shows that the integration with real BPMS is feasible and achievable with little effort. Further, we conducted an evaluation executing a comprehensive simulation scenario with our prototype. Our findings suggest that our approach can lead to various improvements for Industry 4.0 processes with AR tasks. Task assignments can be improved by incorporating contextual factors. Further, AR task execution can be better supported with matching contextual information. The empirical part of our evaluation focused on AR tasks and showed slower execution times but better accuracy and lower error rates with AR support. However, this evaluation also suggests potential because the users were not used to AR devices. We thus expect faster execution times to be observed in daily usage. Thus, the overall process execution can be improved, resulting in better resource usage and cost savings. Moreover, other factors, such as worker safety can also be taken into account and be seamlessly integrated into the processes.

Future work includes: the optimization of our context-integrated process editor to improve its appearance and usability; integration of ARPF with further BPMS; application of ARPF to other domains; further improvements to the BPMN 2.0 extension; and a comprehensive empirical evaluation in a real production environment.

Acknowledgments

The authors wish to acknowledge Felix Gräber's contribution regarding the technical foundation of the IAC. This work was partially funded via the PARADIGMA project by "Zentrales Innovationsprogramm Mittelstand" (i.e., the "Central Innovation Programme for small and Medium-Sized Enterprises (SMEs)"), of the Federal Ministry for Economic Affairs and Energy of the Federal Republic of Germany.

References

[1] G. Grambow, D. Hieber, R. Oberhauser, and C. Pogolski, "Supporting augmented reality industry 4.0 processes with context-aware processing and situational knowledge," in *eKNOW 2021*, 07 2021, pp. 29–36.

[2] R. Baheti and H. Gill, "Cyber-physical systems," *The impact of control technology*, vol. 12, no. 1, pp. 161–166, 2011.

[3] T. Allweyer, *BPMN 2.0: introduction to the standard for business process modeling*. BoD–Books on Demand, 2016.

[4] G. Grambow, R. Oberhauser, and M. Reichert, "Contextual injection of quality measures into software engineering processes," *Int'l Journal on Advances in Software*, vol. 4, no. 1&2, pp. 76–99, 2011.

[5] G. Grambow and R. Oberhauser, "Towards automated context-aware software quality management," in *2010 Fifth International Conference on Software Engineering Advances*. IEEE, 2010, pp. 347–352.

[6] G. Grambow, R. Oberhauser, and M. Reichert, "Towards automatic process-aware coordination in collaborative software engineering," in *6th Int'l Conference on Software and Data Technologies*. ScitePress, 2011, pp. 5–14.

[7] *Business Process Model and Notation(BPMN) - Version 2.0.2*. Object Management Group, 12 2013, retrieved: 2021.06.10. [Online]. Available: https://www.omg.org/spec/BPMN/2.0.2/PDF

[8] G. Grambow, D. Hieber, R. Oberhauser, and C. Pogolski, "A context and augmented reality bpmn and bpms extension for industrial internet of things processes," in *9th International Conference on Business Process Management (BPM 2021) Workshop Proceedings*. Springer, 2021.

[9] Camunda. Last Visited: 2021.06.10. [Online]. Available: https://camunda.com

[10] L. A. Zadeh, "Fuzzy logic," *Computer*, vol. 21, no. 4, pp. 83–93, 1988.

[11] G. Grambow, D. Hieber, and R. Oberhauser, "Intelligent task assignment in industry 4.0 production processes utilizing fuzzy sets," in *INTELLI 2021, The Tenth International Conference on Intelligent Systems and Applications*, 07 2021, pp. 1–9.

[12] M. Reichert, "Enabling flexible and robust business process automation for the agile enterprise," in *The Essence of Software Engineering*. Springer, Cham, 2018, pp. 203–220.

[13] G. C. Hillar, *MQTT Essentials-A lightweight IoT protocol*. Packt Publishing Ltd, 2017.

[14] M. Proctor, "Drools: A rule engine for complex event processing," in *Applications of Graph Transformations with Industrial Relevance*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 2–2.

[15] D. Haniff and C. Baber, "User evaluation of augmented reality systems," in *Proceedings on Seventh International Conference on Information Visualization, 2003. IV 2003.*, 2003, pp. 505–511.

[16] A. H. Maslow, *The Psychology of Science - A Reconnaissance*. Harper & Row, 1966.

[17] C. Bolchini, C. A. Curino, E. Quintarelli, F. A. Schreiber, and L. Tanca, "A data-oriented survey of context models," *SIGMOD Rec.*, vol. 36, no. 4, p. 19–26, Dec. 2007.

[18] K. Henricksen, J. Indulska, and A. Rakotonirainy, "Modeling context information in pervasive computing systems," in *Pervasive Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 167–180.

[19] J. Dörndorfer and C. Seel, "A meta model based extension of bpmn 2.0 for mobile context sensitive business processes and applications," *Proc WI 2017*, pp. 301–315, 2017.

[20] J. Bardram, "The java context awareness framework (jcaf) – a service infrastructure and programming framework for context-aware applications," vol. 3468, 05 2005, pp. 98–115.

[21] J. Park, H.-C. Lee, and M.-J. Lee, "Jcools: A toolkit for generating context-aware applications with jcaf and drools," *Journal of Systems Architecture*, vol. 59, no. 9, pp. 759–766, 2013.

[22] J. Coutaz and J. L. Crowley, "A first-person experience with end-user development for smart homes," *IEEE Pervasive Computing*, vol. 15, no. 2, pp. 26–39, 2016.

[23] G. Ghiani, M. Manca, and F. Paternò, "Authoring context-dependent cross-device user interfaces based on trigger/action rules," *Proceedings of the 14th International Conference on Mobile and Ubiquitous Multimedia*, pp. 313–322, 2015.

[24] F. Giustozzi, J. Saunier, and C. Zanni-Merk, "Context modeling for industry 4.0: an ontology-based proposal," *Procedia Computer Science*, vol. 126, pp. 675–684, 2018.

[25] I. Graja, S. Kallel, N. Guermouche, and A. H. Kacem, "Bpmn4cps: A bpmn extension for modeling cyber-physical systems," in *2016 IEEE 25th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, 2016, pp. 152–157.

[26] J.-R. Ruiz-Sarmiento *et al.*, "A predictive model for the maintenance of industrial machinery in the context of industry 4.0," *Engineering Applications of Artificial Intelligence*, vol. 87, p. 103289, 2020.

[27] C. Tasdemir and C. Toklu, "Qos driven dynamic task assignment for bpm systems using fuzzy logic," in *Emerging trends and challenges in information technology management*. Hershey, Pennsylvania (701 E. Chocolate Avenue, Hershey, Pa., 17033, USA: IGI Global, 2006.

[28] J. Blattgerste, B. Strenge, P. Renner, T. Pfeiffer, and K. Essig, "Comparing conventional and augmented reality instructions for manual assembly tasks," in *Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments*, ser. PETRA '17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 75–82.

[29] J. Vogel and O. Thomas, "Generating smart glasses-based information systems with bpmn4sga: A bpmn extension for smart glasses applications," in *Wirtschaftsinformatik*, 02 2019.

[30] J. Dörndorfer, C. Seel, and D. Hilpoltsteiner, "Sensomod-a modeling language for context-aware mobile applications," in *Multikonferenz Wirtschaftsinformatik (MKWI), 1435-1446*, 03 2018.

[31] M. Grum and N. Gronau, "Integration of augmented reality technologies in process modeling - the augmentation of real world scenarios with the kmdl," 7 2017, pp. 206–215.

[32] R. Seiger, M. Gohlke, and U. Aßmann, "Augmented reality-based process modelling for the internet of things with HoloFlows," in *Enterprise, Business-Process and Information Systems Modeling*. Springer International Publishing, 2019, pp. 115–129.