# FloBP: An Interdisciplinary Model-Driven Approach for Developing and Executing IoT-Enhanced Business Processes

| | |
|---|---|
| Journal: | *Software and Systems Modeling* |
| Manuscript ID | SOSYM-23-00004472 |
| Manuscript Type: | Regular Paper |
| Keyword: | Internet of Things, Model-Driven Engineering, Business Process, Feature Model, IoT-Enhanced Business Process, Microservices |
| | |

# FloBP: An Interdisciplinary Model-Driven Approach for Developing and Executing IoT-Enhanced Business Processes

Arianna Fedeli[1†], Fabrizio Fornari[1†], Andrea Polini[1†], Barbara Re[1†], Victoria Torres[2†] and Pedro Valderas[2†]

[1] PROS PROcesses & Services Lab, University of Camerino, Via Madonna delle Carceri, 7, Camerino, 62032, Italy.
[2] PROS Research Center-VRAIN Institute, Universitat Politècnica de València, Camino de Vera, s/n, Valencia, 46022, Spain.

Contributing authors: arianna.fedeli@unicam.it; fabrizio.fornari@unicam.it; andrea.polini@unicam.it; barbara.re@unicam.it; vtorres@pros.upv.es; pvalderas@pros.upv.es;
†These authors contributed equally to this work.

## Abstract

The capability to integrate Internet of Things (IoT) technologies into Business Processes (BPs) has emerged as a transformative paradigm, offering unprecedented opportunities for organisations to enhance their operational efficiency and productivity. Interacting with the physical world and leveraging real-world data to make more informed business decisions is of greatest interest, and the idea of IoT-enhanced BPs promises to automate and improve business activities and permit them to adapt to the physical environment of execution. Nonetheless, combining these two domains is challenging, and it requires new modelling methods that do not increase notation complexity and provide independent execution between the process and the underlying device technology. In this work, we propose *FloBP*, a model-driven engineering (MDE) approach separating concerns between the IoT and BPs, providing a structured and systematic approach to modelling and executing IoT-enhanced BPs.

Applying the separation of concerns (SoC) through an interdisciplinary team is needed to ensure that the approach covers all necessary process aspects, including technological and modelling ones. The *FloBP* approach is based on modelling tools and a microservices architecture to deploy BPMN models, and it facilitates integration with the physical world, providing flexibility to support multiple IoT device technologies and their evolution. A smart canteen scenario describes and evaluates the feasibility of the approach and its possible adoption by various stakeholders. The performed evaluation concludes that the application of *FloBP* facilitates the modelling and development of IoT- enhanced BPs by sharing and reusing knowledge among IoT and BP experts.

**Keywords:** Internet of Things, Model-Driven Engineering, Business Process, Feature Model, IoT-Enhanced Business Process, Microservices

# 1 Introduction

An Internet of Things (IoT)-enhanced Business Process (BP) refers to integrating IoT technologies and devices into various aspects of a business's operations and workflows to optimise efficiency, enhance productivity, and enable new capabilities of the organization [1]. These devices can either be sensors (e.g., temperature sensor, camera, heart rate sensor) that provide BPs with real-time data to take more informed decisions [2], or actuators (e.g., air conditioner, heating, watering systems, security systems), that are used as digitised physical resources that join processes as artificial actors, to automate and improve the execution of some of the tasks included in the process [3, 4]. Indeed, with IoT-enhanced BPs, organisations can gain access to invaluable insights into their operations, enabling them to identify crucial patterns and trends. Making data-driven decisions based on these insights can enhance overall efficiency, reduce costs, elevate the customer experience, and foster innovation. This convergence of IoT technologies and BPs can empower organisations to remain agile, optimise their operations, and gain a competitive advantage in the ever-evolving digital landscape [5].

Integrating the domains of IoT and Business Process Management (BPM) poses inherent challenges due to their different abstraction levels and characteristics that need to be handled by *different experts' competencies*, resulting in the high complexity and costs [2]. IoT devices, generally handled by IoT application developers, exhibit high heterogeneity in terms of communication protocols, interaction paradigms, and computing and storage capabilities. This heterogeneity complicates the integration process, requiring solutions to address interoperability and standardization issues. On the other hand, business modellers asked to design processes seek to abstract from the intricate technical aspects of IoT. They aim at focusing on higher-level functionalities and outcomes, avoiding the burden of managing the different dimensions of

IoT devices' heterogeneity [6–8]. Overcoming these challenges requires innovative approaches to enable a coherent convergence of holistic disciplines such as the IoT and BPM, empowering organisations to leverage the benefits of IoT technologies without compromising the agility and efficiency of their BPs [5, 9].

**Contribution.** To reduce the overall complexity of modelling and development of IoT-enhanced BPs, we propose *FloBP*, a model-driven engineering (MDE) approach [10] that aims to handle the interdisciplinary nature of IoT and the BPM domains. Tackling the problem from an interdisciplinary perspective requires handling the intrinsic difficulty in developing these solutions [11, 12]. By fostering collaboration and knowledge sharing between experts, it is possible to streamline the overall solution development process to develop scalable, reusable IoT solutions that meet customer demands while avoiding duplication of effort and ensuring efficient utilisation of resources.

To achieve this, *FloBP* builds over two different approaches, presented in [13] and [14]. The first one, named FloWare [13], proposes a model-driven strategy to explicitly model the IoT domain knowledge through a predefined Feature Models structure. The second one, [14], poses guidelines to develop IoT-enhanced BPs and provides a microservices infrastructure to support the deployment of the underlying processes. As a result, *FloBP* intends to provide support from the design to developing and deploying customised IoT-enhanced BPs.

The major contributions of this paper are the following:

- A modelling approach that redefines the structuring of Feature Model diagrams proposed in [13] so as to permit a more effective integration and usage of this knowledge in order to model IoT-enhanced BPs. This allows the modeller to represent the sensing of different aspects of the IoT as BP elements, relieving him/her from acquiring deep technical expertise in IoT devices and systems. To not increase the complexity of the BP modelling task, we analyse the constructs provided by the BPMN metamodel and define a proposal to specify IoT devices and pull interactions without modifying its metamodel. In addition, we apply the Separation of Concerns (SoC) design principle to permit different experts to contribute to the different development phases and steps of the solution. The SoC ensures that each expert can focus on their respective areas, fostering collaboration within an interdisciplinary team.
- A microservices architecture is designed to streamline the integration of business processes with the physical world. It accomplishes this by establishing a robust framework that fosters a seamless connection between the created models and the underlying IoT technologies. By emphasising a high degree of decoupling, this architecture enables efficient integration of IoT devices, even when diverse technologies support them. This flexibility ensures that various IoT devices can seamlessly interact with the overall system, promoting interoperability and scalability. It leverages the power of modelling tools and platforms to automate and optimise the development process, improving productivity and reducing time-to-market.

A methodology in line with the Design Science Research (DSR) guideline is used for this approach. DSR aims to develop practical solutions that professionals in their field can use. More concretely, solutions - or design artefacts - can be constructs, models, methods, or instantiations [15, 16]. In this paper, our solution is a tool-supported MDE approach for modelling and developing IoT-enhanced BPs. According to the DSR [15], this solution can be categorised as an approach since it provides actionable instructions of a conceptual nature.

**Outline.** The DSR methodology, as described in [16], involves six activities. The first activity is *problem identification and motivation*, which is presented in Section 2. The second activity is the *design and development* of the artefact to support the defined objectives. Sections 3 *defines objectives of the contribution*, while Section 4, and 5 explain the modelling approach for interdisciplinary teams in developing IoT-enhanced BPs. We followed an action-research development approach [17], iteratively studying the problem, applying actions, and analysing the results to meet our goals. The fourth activity is the *demonstration*, where the developed artefact solves instances of the problem. We utilise the tools supporting the MDE approach to develop examples and demonstrate its feasibility, such as the motivation example shown in Section 6. In the *evaluation* activity, we observe and measure how well the artefact solves the problem. A controlled subject-based experiment [15] was conducted to evaluate the effectiveness of our solution, following the guidelines proposed in [18]. The experiment is presented in Section 7. We also analyse the state of the art concerning the integration of the IoT domain inside BPs in Section 8, where we compare our solution with existing ones. Lastly, this paper satisfies the sixth step of DSR in Section 9, as it *communicates* results, limitations, and conclusions to the research community.

## 2 Motivation and Challenges

The *FloBP* approach aims to enhance the utilisation of IoT devices in various private and public spaces within business processes, with the ultimate goal of improving efficiency (in terms of time, cost, sustainability, etc.). This approach facilitates the integration of IoT devices to optimise their usage and maximise benefits. To this end, in this work, we illustrate our approach proposal through a running scenario from the IoT domain, precisely the necessity to transform a typical canteen into a *smart canteen.*

Such a transformation can enhance the overall sustainability of the involved processes, i.e., by managing and controlling environmental parameters inside the canteen, as well as permitting to automatically manage functionalities such as reservation and access control for users, the management of dishes and foods, the management and reduction of waste, and many others [19]. Introducing smart devices to enhance the already provided functionalities and to develop new ones also improves the quality and performance of the services, reduces costs and resource consumption, and engages more effectively and actively with its members by automating various aspects of the smart canteen [20].

To better explain the problem, we will consider the part of a smart canteen scenario in which different IoT devices can be integrated into the processes supporting food distribution and those managing the canteen hall.

**Smart Canteen Food Distribution.** Different factors raise the need to transform the traditional food-providing functionality into a smart one. For example, the food supply in the canteen is performed by various operators who manage the customer's order, prepare the correct dish, and allow payment. This leads to long queues and waiting times that could be too long to be able to pick up your meal, which translates into less time available to consume it. In addition, the general monitoring and cleaning of the canteen area must also be carried out. Managers are in charge of monitoring the level of waste bins and the level of environmental cleanliness, and they have to intervene promptly when necessary. However, these activities can become difficult to achieve when the canteen reaches high volumes of concurrent customers, and integrating IoT devices into the canteen processes to make them more effective, seems to be a profitable direction.

Upon arrival at the canteen, users need to be authenticated. If the software system identifies them as subscribers who have made food reservations, a barrier allows them to access the canteen. Subsequently, the food is served on the tray. Users must then proceed with the payment, which is securely stored in the system once completed. Finally, users can take their food and sit in the canteen hall. In the event that a user takes the food before making the payment, s/he is stopped and asked to proceed with the payment before leaving. The whole process can be supported and use different informative systems used by the operators. Anyway, to make it more effective different IoT devices could be included in supporting BP activities. Clearly, these devices will not operate in an isolated and detached way; they must be integrated within the whole canteen process and coordinated to obtain the required functionalities. The integration of IoT devices inside the overall process is strictly dependent on the customer's requirements. In fact, various types of devices can be utilised to fulfil different functionalities. For instance, when it comes to accessing the canteen, authentication can be accomplished through a range of options, such as employing a smart card placed in an RFID scanner, a fingerprint scanner, a camera-based solution, or even a retina scanner. This preference is reflected in the overall BP, which could need to be adapted or enhanced according to the IoT devices chosen.

Developing these solutions requires engaging different stakeholders, each with their own unique areas of expertise, throughout the entirety of the development process [21–23]. Typically, to model these solutions *business engineers* who specialised in articulating the precise requirements that BPs must fulfil are involved. To do so, they use high-level modelling languages such as the Business Process Model and Notation (BPMN) [24]. If we want these processes to be executable, the process must be deployed in a process engine, and the IoT devices must also be set up and configured. However, business engineers may not have the knowledge and skills to manage the IoT technology required to interact with the IoT devices that participate in the process (e.g., temperature sensors, luminosity controllers, proximity sensors, and alarms). Note that each IoT device may require managing different technologies (e.g., MQTT, Zigbee, Bluetooth or CoAP) to interact with the process and produces a different type of data that needs to be correctly elaborated. This could produce errors in modelling and developing BPs, which can be reflected in a not optimised solution. On the other hand, *IoT experts* could fulfil this task as they possess expertise in supporting the introduction of IoT devices inside an application. However, they may not be aware of the underlying BPs nor even understand the notations used to define them. Finally, for the IoT solution development, *IoT Application Developers* are responsible for the development, deployment, and configuration of IoT devices [23]. Such aspects could have a relevant impact on the supported BPs [25].

Handling this problem with an interdisciplinary approach can yield more effective and efficient IoT-enhanced BPs, resulting in various benefits for organisations. These benefits include improved operational efficiency, enhanced customer experience, and the discovery of new business opportunities. Additionally, an interdisciplinary approach can help reduce the complexity of building customised solutions and increase the reusability of developed artefacts. By fostering collaboration between different experts, organisations can identify and address potential technical, operational, and business challenges, ensuring that the IoT system is aligned with the organisation's strategic goals. Overcoming these challenges is essential for organisations and IoT application developers [26, 27].

Considering the aforementioned challenges, we intended to derive and validate a model-driven development process that can support the interdisciplinary nature of IoT-enhanced BPs, fostering the separation of concerns and the smooth cooperation of different stakeholders, bringing different expertise.

## 3 The FloBP Approach

*FloBP* has been conceived in accordance with the principles of the MDE approach that aims to mitigate development challenges by elevating multiple models to primary development artefacts to derive IoT-enhanced BPs [10]. By adopting a MDE approach, enterprises can use the acquired knowledge and experience to optimise their software development processes. This, in turn, improves the stability and maintainability of the delivered solutions and provides customers with a faster development process [28].

The *FloBP* approach, in line with MDE practices, consists of three main phases aiming at producing different artefacts: the Platform Independent Model (PIM), the Platform Specific Model (PSM), and the Code. Each phase encompasses multiple steps to be carried out by different actors, as depicted in Figure 1 within the Modelling Layer. Moreover, particular attention is given to the Physical and Infrastructure Layers, which provide detail about the

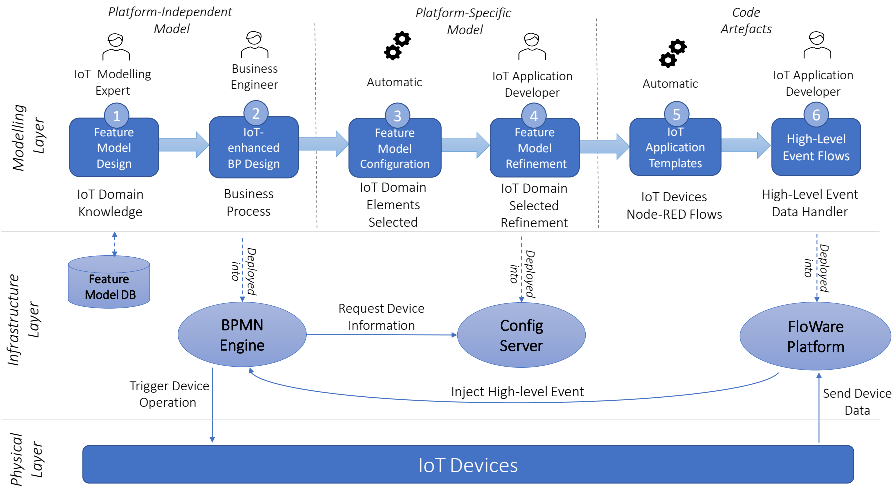


**Fig. 1** Overview of the proposed MDE approach.

microservices structure built for the deployment and runtime execution of the generated software artefacts.

### *Platform Independent Model*

*IoT Modelling Experts* are in charge of reaching extensive knowledge regarding the possible solution in relation to the IoT domain under consideration. This knowledge is modelled and sent to *Business Engineers*, which can exploit it to develop the IoT-enhanced BP.

To achieve the mentioned objectives, two model kinds are used to separate the concerns related to representing several aspects of the final solution.

**Step 1 - Feature Model Design.** The Feature Model serves as the first model and represents a broad range of IoT elements, including their families, domain features, and dependencies. It employs a cross-tree structure to capture the relationships between features [29]. In this approach, an extended version of the feature model structure proposed by the FloWare approach [13] is used to accommodate the heterogeneity and variability aspects of the IoT devices involved. In detail, a focus is posed on the operations that IoT devices can perform. An IoT Modelling Expert, well-versed in designing and representing specific IoT domains using modelling languages such as Feature Models, is engaged in this phase. By leveraging the feature model, a comprehensive *IoT Domain Knowledge* is derived, focusing on the available IoT systems and devices relevant to the specific IoT solution.

**Step 2 - IoT-enhanced BP Design.** The IoT domain knowledge obtained from the feature model is successively elaborated and utilised in the modelling editor to guide the Business Engineer in making informed decisions during the modelling process. In detail, in developing the BP, particular emphasis is put on the IoT devices involved and their corresponding operations within the model. Indeed, the BP can trigger *on-demand* IoT device operations inserted on it, explicitly demanding an IoT device to act. For example, the BP may require the IoT device to open a window, defined as an on-demand interaction between the BP and the physical world, i.e., the IoT device involved in that operation. In such a way, *FloBP* permits the derivation of a truly operational IoT-enhanced BP. Additionally, *autonomous* interactions between the BP and the physical world can be modelled as high-level event-driven communication. High-level events related to IoT devices can trigger the BP when activated. An example of such an event could be the "room too cold" event detected by a temperature sensor, which can trigger the BP when it comes true. This interaction is considered an autonomous event-driven communication, as the IoT devices autonomously inject data without an explicit request by a process. Once terminated, the resulting IoT-enhanced BP is deployed inside the *BPMN Engine*, a microservice able to execute at runtime the developed model.

### *Platform Specific Model*

The decisions made during the BP modelling, specifically regarding the selected IoT systems, devices and their operations, are then automatically reflected inside the feature model as chosen features in the tree. This provides a comprehensive structure of the expert's decisions based on the business requirements, referred to as the **Step 3 - Feature Model Configuration**. However, technology-dependent information about IoT devices, i.e., how to communicate with them and others, is still missing. In this sense, further refinement is necessary to allow communication between the BP and the IoT devices.

**Step 4 - Feature Model Refinement.** To enrich the selected features with technology-dependent details, the involvement of an IoT Application Developer is crucial. The *IoT Application Developer* is responsible for providing specific information about each IoT device involved in the BP, such as the technical aspects required for deployment in the physical world. This includes details about device connectivity, data types produced by the devices, and other relevant information. Once the IoT Application Developer refines the necessary information, a refined Feature Model at the PSM phase is obtained. This model represents the technological configuration of the solution in charge and is saved in the *Config Server*, a microservice dedicated to this purpose.

### *Code Artefacts*

The refinement of the feature model enables **Step 5 - IoT Application Templates**, where the automatic generation of code artefacts serves as IoT application templates to support the autonomous injection of high-level events inside the BP. These code artefacts are developed based on the information provided in the Platform Specific Model, including device types, communication methods, and data types associated with the IoT devices. These templates are represented as flows within the FloWare Platform, which incorporates the Node-RED tool, an extensively used low-code development engine for facilitating the handling of IoT device data and the development of IoT applications.

**Step 6 - High-Level Event Flows.** The IoT devices have no capability of elaborating their data to produce high-level information, i.e., the room is too hot. At the same time, business engineers should be leveraged by handling this type of information, as they just need to use this to develop the process. To achieve this, a software platform is required to convert the raw data collected from devices into actionable high-level information. This platform would bridge the gap between low-level data and valuable insights, empowering businesses to develop and optimise their processes effectively.

In our approach, this is demanded by automatic templates generated inside the FloWare Platform[1], an IoT platform that incorporated the Node-RED tool[2], which uses a low-code programming language to easily develop event-driven IoT applications. These templates serve as a base for processing data generated by IoT devices and assist the *IoT Application Developer* in modelling and managing those high-level events previously defined in the IoT-enhanced BP. Whenever an event occurs, the high-level event result is sent to the BPMN engine, which triggers the starting or continuation of the IoT-enhanced BP execution.

In the following, we discuss in detail the *FloBP* approach using the smart canteen scenario from Section 2, emphasising the involvement of various actors in this interdisciplinary activity. Section 4 describes the platform-independent IoT-enhanced BPs Modelling activity and actors involved, including Steps 1 and 2 of Figure 1 and the actors involved in modelling BPs with IoT capabilities while remaining platform-agnostic. In Section 5, we discuss the transformative Steps 3 and 4 required for platform-specific IoT-enhanced BPs Modelling and Code Artefacts Development. It also provides an overview of developing code artefacts that seamlessly integrate IoT capabilities into BPs, covering Steps 5 and 6.

## 4 Platform-Independent IoT-enhanced BPs Modelling

In this section, we will discuss the modelling of IoT-enhanced BPs, represented by the *Platform Independent Model* related steps, in particular, covering in detail Steps 1 and 2 of the *FloBP* approach, as illustrated in Figure 1. Section 4.1 focuses on knowledge modelling with respect to a specific IoT scenario, while Section 4.2 illustrates the modelling of IoT-enhanced BPs. The approach is described using the smart canteen scenario introduced in Section 2.

### 4.1 Feature Model Design

The digitisation of processes, such as those implemented in a smart canteen, involves the deployment of sensors and actuators, followed by the inclusion of

---

[1] FloWare Platform: https://github.com/PROSLab/FloWare-Core
[2] Node-RED: https://nodered.org/

the corresponding data into the BP flow. For effective implementation, digitisation must consider the unique characteristics of each deployment context [6]. This means that a particular application scenario comprises various solutions that share many similarities but have to be customised to meet specific needs. Organisations that provide Internet of Things (IoT) based solutions are well aware of this challenge and constantly seek strategies to leverage their knowledge and experience from previous deployment scenarios, enabling them to reuse valuable insights [26, 30].

We have described this necessity in detail in the FloWare approach [13], discussing how an organisation specialising in a specific IoT domain needs to categorise the overall IoT software and hardware solutions offered. We synthesised this concept with the term *crystallised IoT knowledge* to indicate the possibility of representing the entire experience and awareness that a given enterprise acquired in a specific IoT application context. This knowledge can then be used to satisfy each customer's necessities specifically. In FloWare [13], feature models have been selected in order to represent such knowledge.

***Feature Model Structure.*** The feature model structure, leverages feature models to manage the inherent heterogeneity arising from the target IoT domain and the utilised devices, thus crystallising the knowledge pertaining to the variability of functionalities made available in specific solutions and the devices required to provide these functionalities. This structure is built upon the well-known IoT-Lite[3] ontology [31], which describes an IoT Domain as a collection of Systems which can be decomposed into Sub-Systems. Each Sub-System is supported by one or more IoT Devices, which needs to be described through several pieces of information.

The proposed structure, reported in Figure 2, proposes as the root of the model the considered *IoT Domain*, that is then directly linked with the *IoT Systems* possibly relevant for the same domain. At its turn, an IoT system can be decomposed and linked to one or more *IoT Subsystems*, each representing a specific functionality provided by the system. The IoT Modelling Experts are responsible for including, defining, and characterising these systems and subsystems, as well as the needed *IoT Devices* that will constitute the last layer of the tree structure. All the features are linked through unique relationships that establish their mandatory or optional selection.

In the *FloBP* approach, we extended this structure, as reported in Figure 2, providing detailed information regarding IoT devices and their operations. This is needed to permit its effective usage in the modelling of the BP, as detailed in Section 4.2. The IoT device data structure requires the mandatory specification of the device *Type*, i.e., sensor, actuator, or tag and the *Operations* the device can perform. Due to the highly heterogeneous nature of the IoT domain, IoT devices can employ one or more communication methods to transmit their data. To effectively represent this variability, each operation

---

[3] IoT-Lite ontology: www.w3.org/Submission/iot-lite

[4] The IoT Device structure (bottom part of the Figure) is applied for each IoT device inserted inside the model.

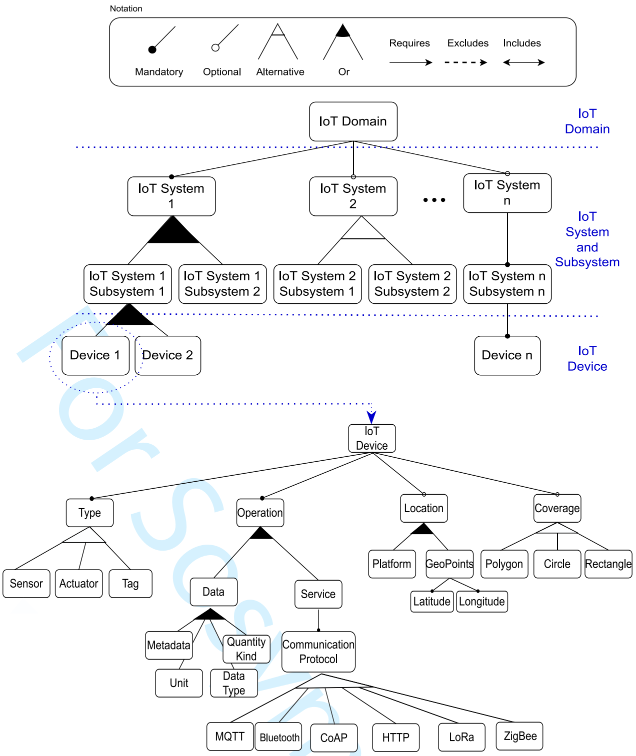


**Fig. 2** IoT Device feature model structure.[4]

necessitates the specification of the *Service*, which denotes the Communication Protocol utilised by the operation. Examples of such protocols include MQTT, Bluetooth, CoAP, HTTP, LoRa, or ZigBee. Furthermore, the schema accommodates optional information related to the device's *Location* and *Coverage*. The *Location* indicates the physical placement of the device, while the *Coverage* defines the operational range of the device.

To fully support this modelling step, we provided a customised version of the *FloWare Tool* as a library deployed inside the ADOxx metamodelling platform[5] that can be used to design the presented feature models. The equipped library can cover all the steps, from the feature model design to its configuration, permitting the generation of artefacts describing different aspects of the IoT application. Thanks to a graphical interface, the resulting platform makes it easy to design and configure all the needed details to derive complete knowledge regarding all the IoT elements involved in a given scenario.

**Feature Model Design (Step 1)**. The first step of the approach asks the IoT Modelling Experts to represent the considered IoT domain through a feature model using the previously described structure. The result of this modelling activity applied inside the FloWare Tool is reported in Figure 3. The

---

[5] FloWare Library: https://www.omilab.org/activities/projects/details/?id=243

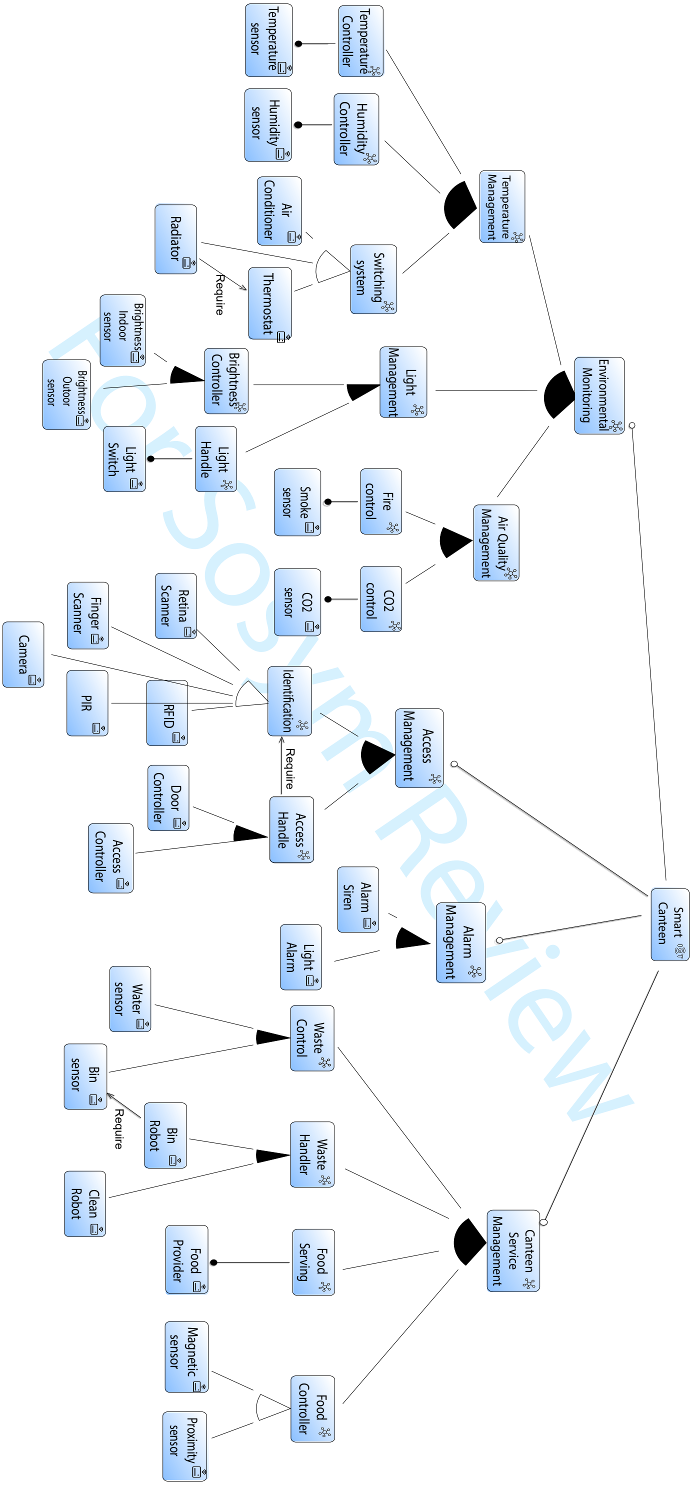
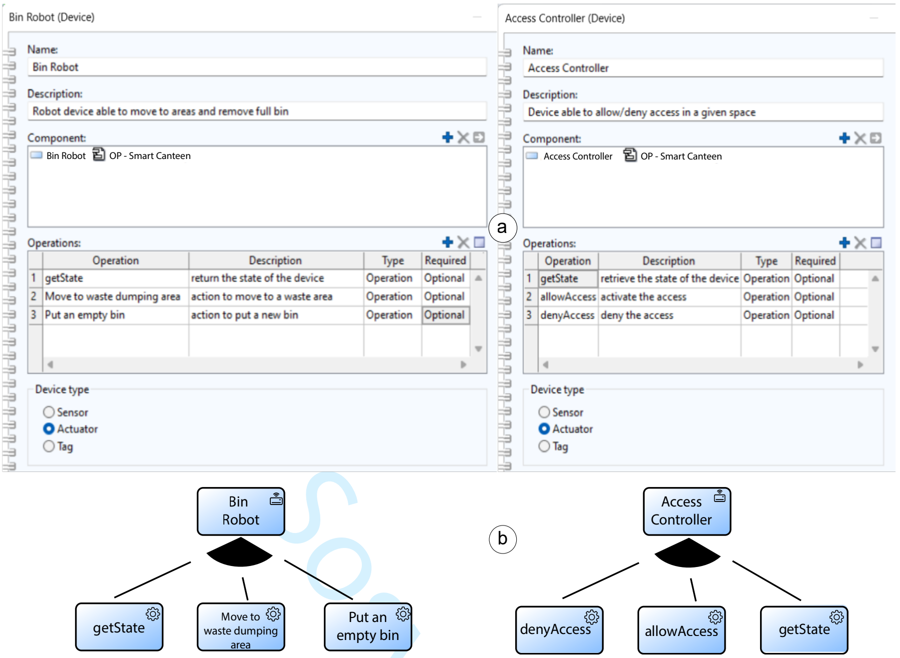


**Fig. 3** Feature model representing the Smart Canteen IoT Domain



Bin Robot (Device)

Name: Bin Robot

Description: Robot device able to move to areas and remove full bin

Component: Bin Robot OP - Smart Canteen

Operations:

| | Operation | Description | Type | Required |
|---|---|---|---|---|
| 1 | getState | return the state of the device | Operation | Optional |
| 2 | Move to waste dumping area | action to move to a waste area | Operation | Optional |
| 3 | Put an empty bin | action to put a new bin | Operation | Optional |

Device type
○ Sensor
● Actuator
○ Tag

Access Controller (Device)

Name: Access Controller

Description: Device able to allow/deny access in a given space

Component: Access Controller OP - Smart Canteen

Operations:

| | Operation | Description | Type | Required |
|---|---|---|---|---|
| 1 | getState | retrieve the state of the device | Operation | Optional |
| 2 | allowAccess | activate the access | Operation | Optional |
| 3 | denyAccess | deny the access | Operation | Optional |

Device type
○ Sensor
● Actuator
○ Tag

**Fig. 4** Detail of information for the Bin Robot and the Access Controller devices (a) and the generated operations in a feature model form (b).

model's root represents the IoT domain, in our case, the Smart Canteen, and it is linked with the systems that can be included in this domain. For example, the Environmental Monitoring system represents the range of possibilities for monitoring the entire Smart Canteen, including the kitchen, eating hall, and others. Each system can then be decomposed into several IoT subsystems, representing, more specifically, the functionalities that can be developed. For example, in the case of the Temperature Management subsystem, it is possible to monitor environmental values related to the temperature and humidity and to activate systems that can impact those environmental values (e.g. radiators). This is done by modelling the IoT devices in the last layer of the feature model and linking them to the related subsystem. Then, for each device, it is possible to include different information; the name, a brief description, the Device Type (i.e., sensor, actuator or tag), and the device's operations, as reported in Figure 4a. The specification of such information will result in the generation of a feature model fragment that will be successively configured by the BP modeller as specified in Section 4.2. Figure 4b reports the generated feature models for the two devices considered.

All the information inserted inside the feature model is then saved inside a *Feature Model DB* and sent to the BPMN editor to allow BP modelling activities with IoT-related modelling elements. In such a way, the knowledge of IoT experts is transferred to be used by BP experts.

## 4.2 IoT-enhanced BP Design

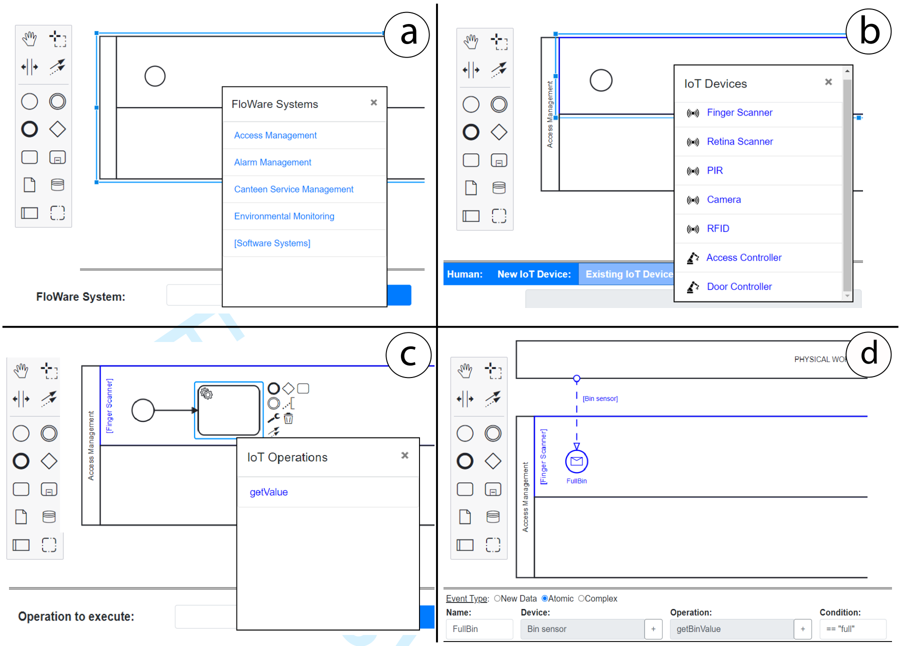
Once IoT devices and their functionalities have been modelled, we need to describe how they participate in the processes of an organisation. To do so, we propose using BPMN since it is a well-known and accepted standard by business process experts both in academia and industry. The notation provides an intuitive and easy way to represent the semantics of complex processes. BPMN is not only used by process designers, who are experts in the usage of the notation to define processes, but also by other process stakeholders such as customers, marketing professionals, or finance employees that just need to analyse them [22, 32, 33]. In addition, BPMN is the most used and preferred modelling language to face the integration of BPs, and IoT [1, 5].

***IoT-enhanced BP Characteristics.*** In order to effectively model IoT-enhanced BPs using the BPMN-based approach proposed in this paper, it is important to understand the key characteristics that need to be considered, as reported in [14].

These characteristics are as follows:

- ***Flow of Coordinated Tasks.*** As traditional BPs, IoT-enhanced BPs require a well-defined flow of coordinated tasks to achieve their goals. These tasks, as illustrated in the motivating scenario in Section 2, can include activities such as user identification, dish dispensing, or payment registration, which need to be executed in a specific logical sequence.
- ***Participation of IoT Devices.*** The IoT devices that participate in the BP can automatically execute some of the tasks included in a process. For instance, the dispensing of dishes could be automatically done by a robot. Thus, we need to consider IoT devices as actors in the process.
- ***Interaction between the BP and IoT Devices.*** The interaction between the BP and the IoT devices can follow two modalities:
    - *On-demand Interaction:* In this type of interaction, the BP explicitly decides when and how to interact with IoT devices based on its business logic. For example, the BP may request an access control device to grant access to a university member, or can ask a robot to dispense dishes, or can activate an alarm.
    - *Autonomous Interaction:* IoT devices can autonomously inject data into the BP without an explicit request. For instance, a finger scanner can automatically inform the BP when a user touches it, or a proximity sensor can inform the BP about the detection of an object. This type of interaction is characterised by the IoT devices independently providing data to the BP, and in their activation.

To effectively model IoT-enhanced Business Processes using BPMN, it is essential to follow specific modelling guidelines as proposed in [14]. In the *FloBP* approach, these guidelines have been expanded to include the predefined feature model structure introduced earlier. It is important to note that this approach does not modify the BPMN metamodel itself but rather adheres



FloWare System:

Operation to execute:

Event Type: ○New Data ●Atomic ○Complex

Name: FullBin

Device: Bin sensor

Operation: getBinValue

Condition: == "full"

**Fig. 5** Some snapshots of the supporting web BPMN modeller. a) allows the selection of an IoT system in a pool *(guideline 1)*, b) allows choosing the IoT device required in a lane *(guideline 2)*, c) allows selecting the operation for that device as an on-demand interaction (*guideline 3*), and d) defining a high-level event as an autonomous interaction *(guideline 4)*.

to it. This ensures that any commercial BPMN engine can successfully execute models created using these guidelines, providing compatibility and interoperability. Furthermore, to support the modelling of IoT-enhanced BPs, we developed and made publicly available the *IoT-enhanced BP web tool*[6]. This tool extends the general BPMN tool editor, including the IoT domain knowledge derived from the feature model to guide the Business Engineers with a knowledge of all the IoT systems, IoT Devices and their operations that could be involved in the process.

In the following, we will discuss the proposed modelling guideline together with a representative example in Figure 5, which shows some snapshots illustrating how the tool supports the inclusion of IoT domain knowledge elements into a BPMN model.

1. ***Pools to represent IoT systems.*** BPMN propose the use of Pools to represent organisational entities that participate in a process. Thus, we propose using Pools to represent IoT systems, reported in the considered feature model, which allows us to organise the participation of IoT devices in the process according to the IoT system they belong to. Other pools that gather other actors unrelated to the IoT domain or that represent external entities can also be included, as it is done in standard BPMN modelling.

---

[6] This tool is available at http://pedvalar.webs.upv.es/iot-enhanced-bp-modeller/

Figure 5a shows the tool providing the user with the list of IoT systems when a pool is selected (note these systems are the ones defined in the feature model in Figure 3).

2. ***Lanes to represent IoT Devices.*** According to good practices in BPMN, lanes should be used to represent the actors that participate in a process. Thus, each IoT device that participates in the process is specified by a lane within the pool representing the IoT system to which the device belongs. Figure 5b shows the list of IoT devices the tool provides to the user when an IoT system is selected. This list is filtered to show only the IoT devices that belong to the IoT system associated with the pool. Note that the pool of this figure is associated with the *Access Management* IoT system, and the list of IoT devices that are provided corresponds with the ones reported in the feature diagram for this specific functionality.
3. ***Service Tasks to represent IoT operations.*** In BPMN, the tasks that are contained within a lane define the actions of the actor represented by the lane. According to the standard, service tasks are those carried out by software. Therefore, in the case of IoT devices, we think that such tasks are the best option to represent their actions. Thus, each IoT device's action required by the BP is defined as a Service Task in the corresponding lane. This supports *on-demand interactions* between the BP and the IoT devices (i.e., when the BP executes one of these tasks, it explicitly demands an IoT device perform an action). Figure 5c shows the list of the operations of an IoT device when a Service Task included in the corresponding lane is selected. In this case, the operations of the *Finger Scanner*, which is the IoT device associated with the lane, are shown.
4. ***Message Flows to represent IoT devices-Physical World interactions.*** An *autonomous interaction* occurs when an IoT device automatically injects some data into the BP without an explicit request by the BP. This type of interaction can be considered an event-driven communication between the physical world in which IoT devices operate and the BP. The BP is interested in the events that occur in the physical world, and it is waiting for the occurrence of these events. BPMN provides the message start event and the message intermediate catch event to define that a process must wait for the reception of an event to either be started or to continue its execution after pausing it, respectively. Thus, autonomous interactions between IoT devices and the BP are represented through message flows drawn between a collapsed pool representing the Physical World in which IoT devices operate, and the rest of the pools that comprise the IoT-enhanced BP. Each message flow is labelled with the name of the IoT device that injects the data into the process and is connected to a message event that receives this data. Figure 5d shows how business engineers can define the condition that must be satisfied to trigger an event. In this snapshot, the condition indicates that the "*FullBin*" event should be injected into the BP when the IoT device's operation *getBinValue* returns a value equal to full.

**IoT-enhanced BP Design (Step 2)**. The possibility to choose between various IoT devices and systems to be incorporated within the required solution, as outlined in the Smart Canteen feature model, is closely aligned with the specific needs and preferences of customers in that particular scenario. Figure 6 describes an exemplary IoT-enhanced BP that effectively defines and optimises food distribution within the smart canteen as presented in Section 2.

As we can see, there are three pools that represent the IoT systems involved in this process: the *Access Management*, the *Canteen Service Management* and the *Alarm Management*. Note that these pools comply with the IoT systems defined inside the feature model in Figure 3. In addition, there is a pool that contains some Software Systems that participate in the process (*Information System* and *Email System*), a collapsed pool that represents the external entity that processes the *Payment*, and a collapsed pool that represents the *Physical World*. Note how several message flows arise from this last pool to represent the autonomous interaction of the IoT devices with the BP.

The food delivery process starts when a ***Finger Scanner*** injects a *Finger Reading* event into the BP. Then, the *Information System* retrieve the subscriber information corresponding to the received reading. If the user is not identified as a service subscriber or a reservation has not been made, the process terminates. Otherwise, the ***Access Controller*** device is requested to allow the user to access the canteen, and the *Information System* loads the dishes of the reservation. Once the full reservation is loaded, the ***Food Provider*** device is requested to dispense the corresponding dishes. Afterwards, the BP waits for one of the following two circumstances, either an external Payment Entity confirms the payment, then the *Information System* registers it, and the process finishes, or the user takes the food before paying. If the second case is observed, the "Tray is taken from the desk" event is automatically injected by a ***Proximity Sensor***. The event is activated when it detects that the tray with the food is taken from the dispensing desk without paying, and a ***Light Alarm*** is activated. At this point, the BP waits for one of these two other events. Either the user puts the tray again in the dispensing desk (this is processed through the *Tray is back on the desk* event, and then the BP stops the ***Light Alarm*** and waits again for the payment), or in case no other event is observed for 5 minutes the BP stops the ***Light Alarm***. As a consequence, the *Information System* stores that missing payment and a reminder payment email is sent to the user by the *Email System*. Afterwards, the process finishes.

We also included the management of a full bin in the BPMN model. A Bin Sensor detects that a bin is full, and through the *BinFull* event, it injects an event into the BP to inform about this issue. Then, the BP requests a ***Bin Robot*** to move the full bin to a dumping waste area and to bring an empty bin.

***Management of High-Level Events.*** In the *FloBP* approach, the abstract representation of events injected into the BP from the physical world is evident. However, specific details about how these events are generated from

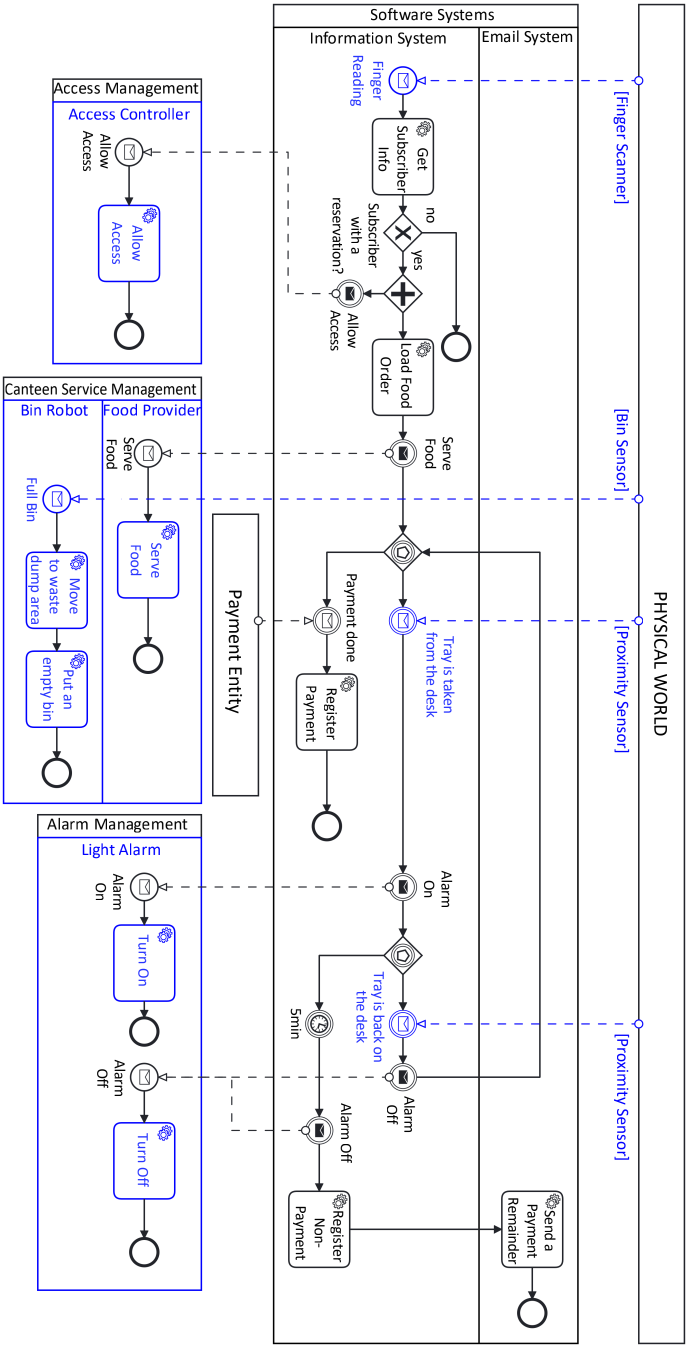


**Fig. 6** BPMN model that describes the dispensing of dishes in the Smart Canteen. The IoT devices involved and their on-demand (service tasks) and autonomous (message flows) interactions are depicted in blue.

the physical world have not been presented. For instance, the ***FullBin*** event is generated when the *Bin Sensor* detects that the bin is full and needs to be changed. Similarly, the ***Tray is back on the desk*** event is injected into the model when the user places the tray back on the desk, but only if the *Alarm* is on. The reason for this is that the event-triggering logic and autonomous interaction with IoT devices are not directly handled by the BP engine that executes the BPMN model. Instead, we have applied the Separation of Concerns (SoC) principle and delegated this responsibility to the IoT devices themselves. It is important to note that IoT devices typically lack the computing capabilities to generate high-level events directly. For example, a Bin Sensor can provide real-time data on the bin's level but may not have the ability to interpret this data at a higher level of abstraction. To address this challenge, we have chosen to delegate the management of the event-triggering logic, including the autonomous interaction of IoT devices, to the FloWare Platform. This platform leverages Node-RED flows, which are responsible for interacting with IoT devices and generating the high-level events required by the BP at runtime. This approach is further explained in detail in Section 5 of the paper.

Note that Business Engineers are those who better know the business requirements that must drive the definition of this event-triggering logic. Thus, the BPMN editor we developed to support the proposed modelling approach (presented below) provides Business Engineers with a user interface to (1) define the conditions that trigger the high-level events included in the BPMN model and (2) send these definitions to the FloWare platform that can support them. In this way, Business Engineers can completely define an IoT-enhanced BP considering the sequence of tasks, the on-demand interaction with IoT devices, and the autonomous interaction defined in terms of conditional events.

The web tool presented in this section supports automatically generating a preliminary feature model PSM from BPMN models. It allows the automatic generation of a preliminary feature model PSM for each scenario. Each of these PSMs includes a selection of features according to the IoT devices required by each scenario. Additionally, it is worth noting that the web tool checks the rules defined in the feature model. We explained above that the feature model could introduce constraints that, for instance, make mandatory some specific IoT devices. Thus, when the web tool generates a preliminary feature model PSM it checks whether these constraints are satisfied or not. If a constraint is not satisfied, the tool provides a visual error to the user requiring them to take action to fix the problem.

## 5 Platform-Specific IoT-enhanced BPs Modelling and Development

Once the whole BP has been modelled, the result is an IoT-enhanced BP in which all the elements that are involved in the scenario are reported, and the model also specifies how the various elements interact with each other. In this

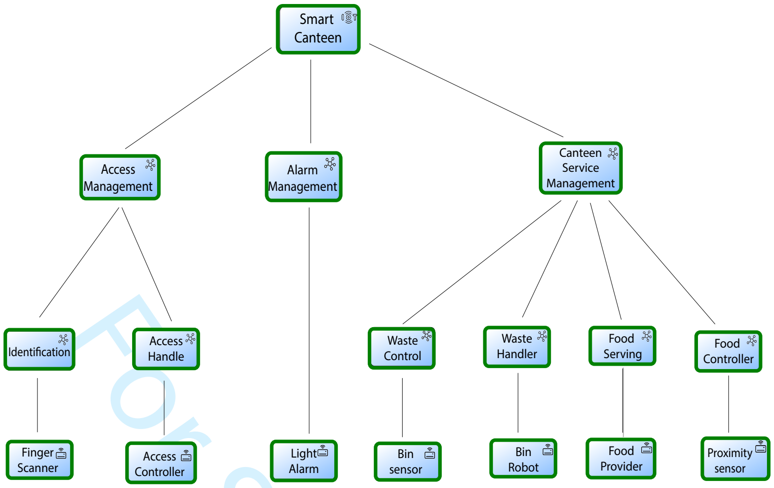
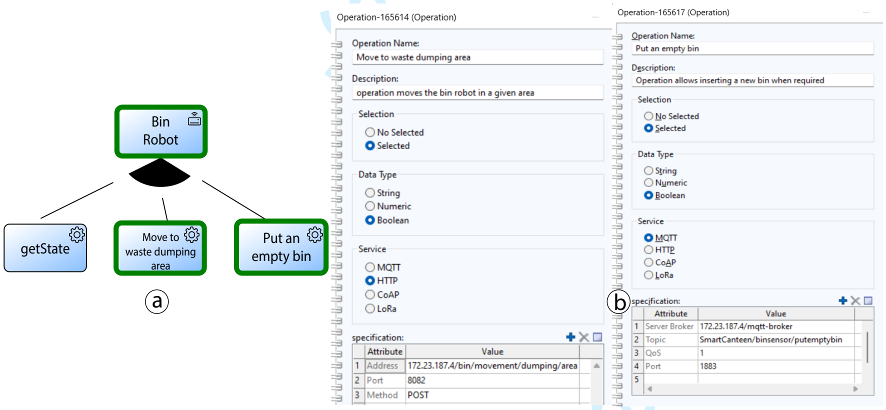


**Fig. 7** Smart Canteen Feature Model Selection. For exemplification purposes, in this figure, only the selected features appear.



Operation-165614 (Operation)

Operation Name: Move to waste dumping area

Description: operation moves the bin robot in a given area

Selection
○ No Selected
● Selected

Data Type
○ String
○ Numeric
● Boolean

Service
○ MQTT
● HTTP
○ CoAP
○ LoRa

specification:

|  | Attribute | Value |
|---|---|---|
| 1 | Address | 172.23.187.4/bin/movement/dumping/area |
| 2 | Port | 8082 |
| 3 | Method | POST |

Operation-165617 (Operation)

Operation Name: Put an empty bin

Description: Operation allows inserting a new bin when required

Selection
○ No Selected
● Selected

Data Type
○ String
○ Numeric
● Boolean

Service
● MQTT
○ HTTP
○ CoAP
○ LoRa

specification:

|  | Attribute | Value |
|---|---|---|
| 1 | Server Broker | 172.23.187.4/mqtt-broker |
| 2 | Topic | SmartCanteen/binsensor/putemptybin |
| 3 | QoS | 1 |
| 4 | Port | 1883 |
| 5 |  |  |

**Fig. 8** Feature Model Refinement step: (a) Detail on the Bin Robot operations selected, and (b) the refinement of its specific information.

section, we will detail Steps 3 and 4 of the approach to derive a Platform-Specific Model artefact able to perform on-demand requests to the Physical World. Then, we will continue through Steps 5 and 6, presenting also the code artefact phase.

**Feature Model Configuration (Step 3).** In this step, the choice of IoT systems, IoT devices and operations to be included in the BP results in the automatic execution of a feature model configuration on the starting feature model. Indeed, this is illustrated in Figure 7, where the BP process decisions are reflected as the feature model configuration, with the corresponding selected features highlighted in green. Also, the operations of each IoT device included in the IoT-enhanced BP are automatically selected. As illustrated in Figure 6, taking as an example the *Bin Robot* device, the IoT-enhanced BP includes the ***Move to waste dumping area*** and the ***Put an empty bin***. This selection is reported inside the feature model, as shown in Figure 8a. Clearly, in case a constraint is violated, the system notifies the user as the configuration is not admissible.

**Feature Model Refinement (Step 4)**. At this point, the refinement of the feature model information is necessary to allow the BP to actually communicate with the IoT devices to request to perform operations. The IoT Application Developer, that is the actor involved in this step, must include technological information for each device operation selected within the model. This Feature Model Refinement step is illustrated in Figure 8b, where the operations' details are included. As shown, the expert must select the type of communication protocol related to that operation. Once selected, it is necessary to add specific information, as in the case of the ***Move to waste dumping area***. In this case, the selection of the HTTP protocol requires specifying the Address, Port, and Method, to reach the device. In contrast, the ***Put an empty bin*** operation can be reached through the MQTT protocol. In this case, different parameters such as the Server Broker, the Topic, the Quality of Service (QoS), and the Port need to be inserted. In addition, for each operation, it is necessary to select the type of data that the operation can retrieve, defined as *Data Type*. This allows the automatic and correct elaboration of the operations data retrieved. In this case, both operations retrieve a Boolean data type, defined as a true/false response to the requested operation.

After refining all the elements inserted inside the IoT-enhanced BP and consequently selected inside the feature model configuration, a Platform-Specific Model is obtained. This model, comprehensive of all the configurations inserted, is sent to the *Config Server* and stored.

**IoT Application Templates (Step 5)**. Once the Feature Model refinement is completed, the FloWare tool elaborates that data and automatically generates IoT application templates to be deployed directly within the FloWare platform, the developed software that is integrated with the Node-RED tool, a widely used software for the development of event-driven IoT solutions. These templates are developed as interconnected visual components to set primitive data processing streams of IoT devices and display that data through a dashboard. A detailed explanation of how these templates were generated is deferred to [13]. In this approach, we extend those ones, allowing the IoT Application Developer to handle the high-level events modelled directly inside the IoT-enhanced BPs.

**High-level Event Flows (Step 6)**. Figure 9 shows the templates related to the events previously described in Section 4.2. Each high-level event to be developed is automatically generated inside a group, that presents a *comment* node (in white in the figure) in which the device involved and both the entire event description and the condition, are reported. This could guide the IoT Application Developer to know which event must be developed. Different nodes are automatically provided for each flow. First, a node to retrieve the device

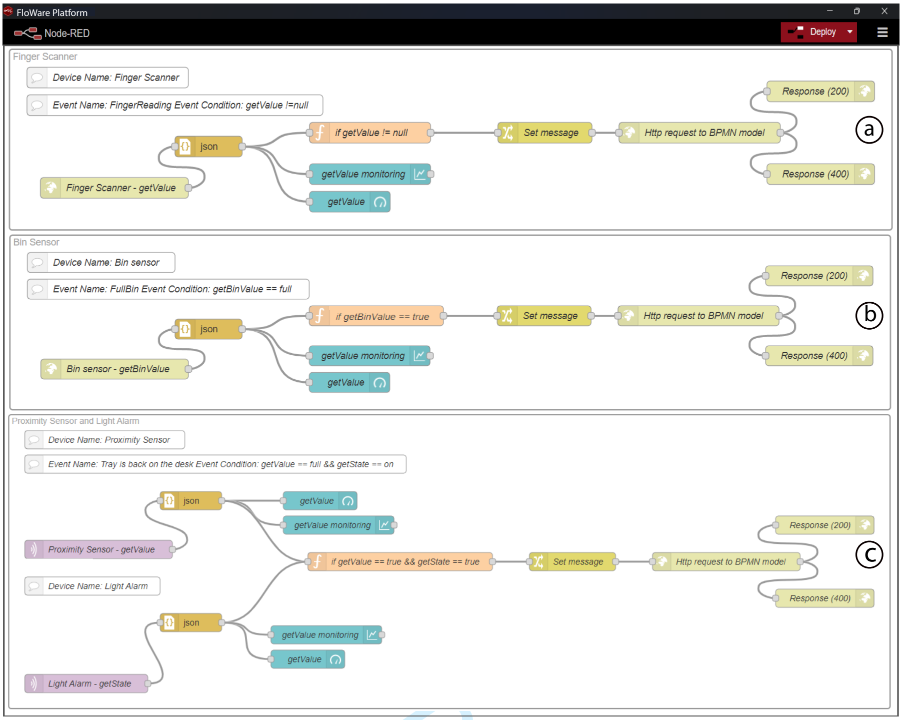


**Fig. 9** High-Level Events automatically generated inside the FloWare Platform.

information is generated. In this case, different nodes can be used, depending on the Communication Protocol used by that device. For example, in Figure 9, both the HTTP protocol (the green node) and the MQTT one (the violet node) are generated following the configuration retrieved. Indeed, each node specialises in performing a precise task and needs specific information to be inserted. The only information necessary to use those nodes (e.g., the port, address, topic and others) is the one already provided inside the Feature Model PSM configuration, which is automatically translated and inserted inside the correspondent one. The IoT device name and the operation name (e.g., getValue, getState, and others) are indicated as a visual label in the component itself. The data received is then read and formatted from its data type into JSON format through a specific node. Then, different nodes allow graphical visualisation of the IoT device data using various widgets on a dashboard. A function node analyses the received value and checks if it meets the event condition. Additional nodes, to set the event message, send the HTTP request to BPMN Model, and wait for the response, are used to set event message parameters for injecting the event into the IoT-enhanced BP.

The result of this activity for the development of high-level events modelled in the BP is reported in Figure 9. Figure 9a presents the ***FingerReading*** event. This event represents the scenario in which the user tries to scan his/her finger to have access to the canteen through the *Finger Scanner* device.

Through the *getValue* operation, the high-level event retrieves the IoT device data and injects the event inside the BP when the retrieved data is different than *null*.

Figure 9b illustrates the ***FullBin*** event. This event is associated with the *Bin Sensor*, an IoT device that aims to control the level of the bin in the canteen. The event is injected into the BP when the condition is equal to full. Since the bin sensor itself lacks the capability to directly understand whether the bin is full or not, it communicates this information in a boolean format, specifically as a true or false value. This data information is taken into account during the feature model configuration stage, as depicted in Figure 8. To enable the BP to respond appropriately when the bin is full, the IoT application developer is responsible for developing the necessary function. This function should evaluate the received boolean value from the *Bin Sensor* and trigger the relevant actions or workflows within the BP when the condition indicates that the bin is indeed full.

In Figure 9c, a more complex high-level event called ***Tray is back on the desk*** is presented. This event is triggered when a user attempts to take food without paying and, upon being alerted by an alarm, decides to put the tray back on the desk. The event is associated with two components: the *Proximity Sensor* and the *Light Alarm*, as previously depicted in the BP diagram shown in Figure 6. In this event, the proximity sensor retrieves data using the *getValue* operation, while the light alarm utilises the *getState* operation. Both of these operations exchange data through the MQTT protocol, represented as the violet components in the diagram. The received values are analysed in real-time. The condition *"getValue==full"* is satisfied when the proximity sensor detects a presence and sends a *true* value. Simultaneously, if the *getState* operation of the light alarm device meets the condition *"getState==on"* and its value is *true*, indicating that the light alarm is already activated, the event can be injected into the BP. Once these conditions are met, all the necessary parameters are set to send a message to the corresponding IoT-enhanced BP. At this point, the BP continues its execution, stopping the *Visual Alarm* by requesting the related operation, and prompting the user again for payment.

Overall, once the developed event occurs, it sends a message to the BP, signalling its occurrence. It is, therefore, the task of the BP to manage the verified event and the whole process in general.

## 6 Supporting Architecture Prototype

Previous sections have introduced an MDE approach to developing IoT-enhanced BP through BPMN models and feature models. By following the different steps proposed by this approach (see Figure 1), different software artefacts are produced. These artefacts are deployed into a microservice architecture proposed to execute the IoT-Enhanced BP they implement. Microservices [34] propose an architectural style where applications are decomposed into small independent building blocks (the microservices), each of them focused on a single business capability. Microservices communicate with each other with lightweight mechanisms, and they can be deployed and evolved independently, which leads to more agile developments and technological independence between them [35].

The generated software artefacts and the microservices in which they are deployed are the following:

- One or more BPMN models which define the business processes to be supported. They are deployed into a BPMN Engine microservice that executes them.
- A Feature Model PSM, which is deployed into a Config Server microservice that is in charge of interpreting it and providing the configuration data that is required to interact with the IoT devices.
- A set of Node-RED flows that manage the autonomous interaction of IoT devices (i.e., event triggering) and which are deployed into the FloWare Platform as a high-level events Manager microservice.

According to [36], a way of preliminary evaluating the proposal of a new architecture is through developing a prototype. Next, we introduce a realisation of the architectural solution presented in this paper as a prototype involving mapping technology choices onto the solution concepts. In addition, we used this implementation to perform a preliminary evaluation in which the hypothesis that we wanted to validate was the following:

H1: *It is feasible to execute IoT-enhanced BPs modelled with BPMN and Feature Models with the proposed architectural solution.*

## 6.1 Proof of Concept Implementation

We performed a proof-of-concept implementation to support the motivating example presented in this paper. Figure 10 graphically illustrates the realisation of the proposed architecture. The technological decisions we took to create this implementation were the following:

- The Config Server was implemented as a Spring Boot[7] application in Java. A MySQL[8] DBMS was used to store the configuration data of each operation. This application was deployed into a Linux system. It published a REST API, which provided endpoints to (1) receive the Feature Model PSM through a POST HTTP connection and store the operation data in the database; and (2) provide this data for each operation through GET HTTP connections.
- The BPMN Engine was implemented with a Camunda[9] engine that was deployed into a Windows system. The Camunda Engine provides a REST API with several endpoints that allow, among other tasks, to deploy a BPMN model into the engine and inject an event into a process. Camunda was endowed with an IoT module implemented in Java that oversees the

---

[7] Spring: https://spring.io/
[8] MySQL: https://www.mysql.com/
[9] Camunda Engine: https://camunda.com/

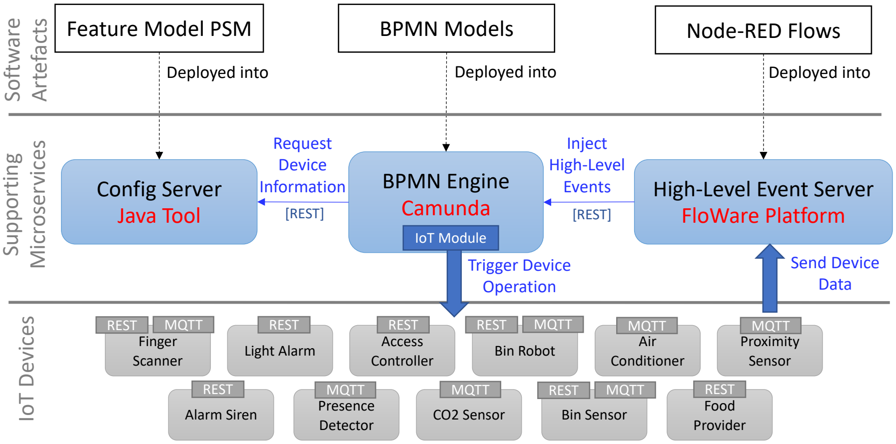


**Fig. 10** Microservice architecture implemented as a proof of concept

execution of IoT device operations. This module was bound to each Service Task included in a BPMN model. Thus, Camunda delegated the execution of each Service Task to this module, which consumed the REST API of the Config Server to know the data of the operations and used this data to request the operation execution to the corresponding IoT Device.

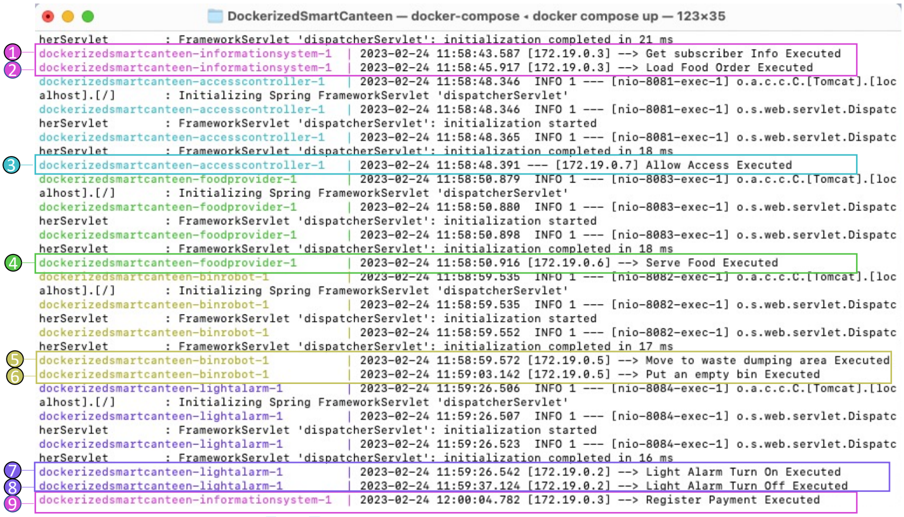
- The high-level event Sender was supported by the FloWare Platform, which was installed in a Windows system. This platform includes the Node-RED tool that was used to execute the flows that were configured to monitor the state of the IoT devices, check the conditions defined by the Business Engineers, and inject the corresponding events into a process by using the Camunda REST API when required.
- IoT devices were emulated through dockerized apps. Each IoT device was developed as a Java application that implemented the operations defined in the feature model. In addition, to allow interaction with them, they provide a REST API implemented with Spring, or the MQTT-based Mosquitto[10] messaging broker, or both. Each application was deployed into a docker[11] package, so they have their own IP address.

## 6.2 Testing the Prototype

Once the prototype was implemented, we evaluated its correct performance through the execution of the motivating example. To do so, we deployed the BPMN models in Figure 6 into the Camunda engine. Also, the Feature Model PSM shown in Figures 7 and 8 was deployed into the Config Server. Finally, every defined Node-RED flow, as the one presented in Figure 9, was deployed into the high-level event Sender microservice.

---

[10] Mosquitto: https://mosquitto.org/
[11] Docker: https://www.docker.com/



```
herServlet          : FrameworkServlet 'dispatcherServlet': initialization completed in 21 ms
dockerizedsmartcanteen-informationsystem-1  | 2023-02-24 11:58:43.587 [172.19.0.3] --> Get subscriber Info Executed
dockerizedsmartcanteen-informationsystem-1  | 2023-02-24 11:58:45.917 [172.19.0.3] --> Load Food Order Executed
dockerizedsmartcanteen-accesscontroller-1   | 2023-02-24 11:58:48.346  INFO 1 --- [nio-8081-exec-1] o.a.c.c.C.[Tomcat].[loc
alhost].[/]         : Initializing Spring FrameworkServlet 'dispatcherServlet'
dockerizedsmartcanteen-accesscontroller-1   | 2023-02-24 11:58:48.346  INFO 1 --- [nio-8081-exec-1] o.s.web.servlet.Dispatc
herServlet          : FrameworkServlet 'dispatcherServlet': initialization started
dockerizedsmartcanteen-accesscontroller-1   | 2023-02-24 11:58:48.365  INFO 1 --- [nio-8081-exec-1] o.s.web.servlet.Dispatc
herServlet          : FrameworkServlet 'dispatcherServlet': initialization completed in 18 ms
dockerizedsmartcanteen-accesscontroller-1   | 2023-02-24 11:58:48.391 --- [172.19.0.7] Allow Access Executed
dockerizedsmartcanteen-foodprovider-1       | 2023-02-24 11:58:50.879  INFO 1 --- [nio-8083-exec-1] o.a.c.c.C.[Tomcat].[loc
alhost].[/]         : Initializing Spring FrameworkServlet 'dispatcherServlet'
dockerizedsmartcanteen-foodprovider-1       | 2023-02-24 11:58:50.880  INFO 1 --- [nio-8083-exec-1] o.s.web.servlet.Dispatc
herServlet          : FrameworkServlet 'dispatcherServlet': initialization started
dockerizedsmartcanteen-foodprovider-1       | 2023-02-24 11:58:50.898  INFO 1 --- [nio-8083-exec-1] o.s.web.servlet.Dispatc
herServlet          : FrameworkServlet 'dispatcherServlet': initialization completed in 18 ms
dockerizedsmartcanteen-foodprovider-1       | 2023-02-24 11:58:50.916 [172.19.0.6] --> Serve Food Executed
dockerizedsmartcanteen-binrobot-1           | 2023-02-24 11:58:59.535  INFO 1 --- [nio-8082-exec-1] o.a.c.c.C.[Tomcat].[loc
alhost].[/]         : Initializing Spring FrameworkServlet 'dispatcherServlet'
dockerizedsmartcanteen-binrobot-1           | 2023-02-24 11:58:59.535  INFO 1 --- [nio-8082-exec-1] o.s.web.servlet.Dispatc
herServlet          : FrameworkServlet 'dispatcherServlet': initialization started
dockerizedsmartcanteen-binrobot-1           | 2023-02-24 11:58:59.552  INFO 1 --- [nio-8082-exec-1] o.s.web.servlet.Dispatc
herServlet          : FrameworkServlet 'dispatcherServlet': initialization completed in 17 ms
dockerizedsmartcanteen-binrobot-1           | 2023-02-24 11:58:59.572 [172.19.0.5] --> Move to waste dumping area Executed
dockerizedsmartcanteen-binrobot-1           | 2023-02-24 11:59:03.142 [172.19.0.5] --> Put an empty bin Executed
dockerizedsmartcanteen-lightalarm-1         | 2023-02-24 11:59:26.506  INFO 1 --- [nio-8084-exec-1] o.a.c.c.C.[Tomcat].[loc
alhost].[/]         : Initializing Spring FrameworkServlet 'dispatcherServlet'
dockerizedsmartcanteen-lightalarm-1         | 2023-02-24 11:59:26.507  INFO 1 --- [nio-8084-exec-1] o.s.web.servlet.Dispatc
herServlet          : FrameworkServlet 'dispatcherServlet': initialization started
dockerizedsmartcanteen-lightalarm-1         | 2023-02-24 11:59:26.523  INFO 1 --- [nio-8084-exec-1] o.s.web.servlet.Dispatc
herServlet          : FrameworkServlet 'dispatcherServlet': initialization completed in 16 ms
dockerizedsmartcanteen-lightalarm-1         | 2023-02-24 11:59:26.542 [172.19.0.2] --> Light Alarm Turn On Executed
dockerizedsmartcanteen-lightalarm-1         | 2023-02-24 11:59:37.124 [172.19.0.2] --> Light Alarm Turn Off Executed
dockerizedsmartcanteen-informationsystem-1  | 2023-02-24 12:00:04.782 [172.19.0.3] --> Register Payment Executed
```

**Fig. 11** Logs obtained in the execution of the dish dispensing process. Format: `Container ID | Date & Time [Container IP] --> Operation Message`

According to the BPMN model presented, the business process must start when a "*Finger Reading*" is obtained (see Figure 6). In addition, other events such as "*Full Bin*", "*Tray is Taken from the Desk*", and "*Tray is Back on the Desk*", must also be injected into the BPMN engine from the physical world to complete the execution of the processes. The high-level event Sender must inject these high-level events through the execution of the Node-RED flows. In a real scenario, these flows would monitor the state of IoT devices to decide whether or not to inject the corresponding events. In this testing experiment, these flows were connected to the emulated IoT devices, which were manually configured to generate the data needed to trigger the events, so we could test the execution processes.

To analyse the correct execution of the processes, we made each emulated IoT device log the execution of each operation. After the execution of each process was completed, we analysed the generated logs to check that operations were executed as it was defined in the BPMN model. As a representative example, Figure 11 shows the logs obtained for the execution of the dish dispensing process.

To start the process of dish dispensing, we made the Finger Scanner publish a reading of a subscriber into a queue defined in its internal Mosquitto broker. Then, a Node-RED flow that was subscribed to this queue got the reading and injected the "Finger Reading" event into Camunda. At this point, Camunda executed all the tasks defined in the model until it must wait for the payment: (1) *Get Subscriber Info*, (2) *Load Food Order*, (3) *Allow Access* and (4) *Serve Food*. Then, we made the Bin Sensor that detected a bin was full in such a way a Node-RED flow could inject the corresponding event into Camunda, and the engine executed the tasks to (5) *Move the bin to a waste dumping* *area* and (6) *Put an empty one*. Afterwards, we made the Proximity Sensor of the dispensing desk to publish the detection of an object moving away. Then, the event *"Tray is Taken from the Desk"* was injected into Camunda by a Node-RED flow, and Camunda (7) *Turned the light alarm on*. Afterwards, we made the Proximity Sensor to publish the detection of an object, a Node-RED flow injects the "*Tray is back on the desk*" event into Camunda, and the BPMN engine (8) *Turned the alarm off*. Finally, we generated the event of *Payment done* that is produced by an external entity, and Camunda asks for (9) *Registering the payment*, which finished the process.

**Conclusions.** According to the generated logs, we could conclude that the realisation of the proposed architecture successfully executed the motivating examples. This means that: (1) the high-level event Sender correctly analysed the data produced by the IoT devices to inject high-level events into the BP Engine; (2) the BP Engine properly interacted with the Config Server to ask for the data required to execute operations; and (3) the BP Engine properly interacted with the IoT devices to execute the operations according to the logic defined in the BPMN models. Thus, we could conclude that the feasibility of the proposed approach (hypothesis H1) is validated.

## 7 Case Study Evaluation

In this section, we evaluate the proposed *FloBP* approach from the perspective of the three different roles considered in it, i.e., the IoT Modelling Expert, the Business engineer, and the IoT application developer. To this end, we propose to validate the following hypothesis:

"*The proposed FloBP approach and the supporting infrastructure allows an effective collaboration to construct and deploy an IoT-enhanced BP, the flow of coordinated tasks, the IoT devices participating in the BP, and the interactions that the process must have with IoT devices.*"

To do so, we arranged a usability experiment where participants were asked to define the corresponding model according to the role played. To this end, we provided them with the infrastructure implementing the proposed architecture and asked them to work on the Smart Canteen Food Distribution scenario, presented in Section 2. We applied a case study-based evaluation by following the research methodology practices provided by [18]. These practices describe how to conduct and report case studies and recommend how to design and plan the case studies before performing them. Next, we introduce the experiment by describing its participants, design, execution, analysis of the results, discussions of the results, and validity threats.

### 7.1 Participants

A total of 20 subjects participated in the experiment, plus the authors that played the role of IoT Modelling Experts. The number of participants recruited was designed to facilitate their distribution into two balanced groups of ten participants each. Each group played the role of a specific developer (i.e., business engineers and IoT application developers). It is good to note that the authors played the role of IoT Modelling Experts and created the initial feature model that was required to develop the Smart Canteen scenario, which is reported in Figure 3.

In particular, we formed the following two groups:

1. 10 participants between 28 and 54 years old (4 female and 6 male) were members of the VRAIN Institute at the Universitat Politècnica de València, Spain. These participants had experience in BPMN modelling, so they played the role of business engineers.
2. 10 participants between 25 and 36 years old (2 female and 8 male) from the PROS Laboratory at the University of Camerino, Italy. They had to play the role of IoT Application Developers, so they were experienced in the IoT domain.

The participants of each group we recruited in such a way we can guarantee that members of the same group had all a similar profile. However, in order to be sure and detect possible shortcomings, we propose they fill in a questionnaire with some questions related to their experience and background. In addition, as we explain further, the training sessions done during the experiment were used to teach participants the technology required to apply our approach, which they had not experienced. These sessions were also used to reinforce some basic notions we consider opportune from the analysis of the questionnaire results.

## 7.2 Design

To perform usability experiments, it is necessary to clarify how usability can be measured (affected variables). According to the standard ISO 9241-11 (1999)[12], the main affected variables concerning usability requirements are (1) effectiveness, (2) efficiency, and (3) user acceptance. To measure effectiveness and efficiency, we based on [37]. The effectiveness was measured as the grade of task completion obtained when comparing a task's result with a predefined master result. The efficiency was measured as the time needed to complete a task. Inspired by [38], this time was compared with the time obtained by an expert on the modelling approach when performing the same task. Regarding user acceptance, it was measured through a NASA-TLX questionnaire [39]. Thus, the instruments that were used to carry out the experiment are as follows:

- *A demographic questionnaire*: a set of questions to know the level of the users' experience in Business Process modelling, BPMN, feature modelling, IoT, and microservices.
- *Work description*: the description of the activities that the subjects should carry out, i.e., using our MDE approach to define, deploy and execute the corresponding part of the IoT-enhanced process that supports the running scenario, i.e., the smart canteen.

---

[12] ISO Standard: https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-2:v1:en

- *A NASA-TLX questionnaire*: it was used to evaluate the perceived mental/physical/temporal demand, performance, effort and frustration on a 100-point scale with 5-point steps. This questionnaire was extended with an additional open question.
- *A time form*: it was defined to capture the start and completion times of the proposed activities.

### 7.3 Execution

We organised two one-day workshops (one in Spain and another in Italy) with two sessions of two hours each. In both workshops, during the first session, participants were asked to complete a demographic questionnaire to capture their backgrounds. Then, they were trained in our MDE approach, introducing the modelling languages used at each stage of the MDE process and the tools provided to create and configure the different artefacts required during the process. The objective of this training part was to provide participants with an overview of the whole process, i.e., which artefacts are produced and consumed along the process and who is responsible for their creation, edition/configuration, and deployment.

In the second session, participants were asked, according to the role played by each one, to create the model that supports the scenario of the smart canteen, writing down the starting and ending times of the task.

To perform the experiment, the feature model representing the Smart Canteen scenario reported in Figure 3 was used. It is important to note that the definition of this model does not contain low-level details, such as communication protocol, etc., about the real devices being used in the different scenarios. This means this model exists as a single instance that can be successively updated and is used to build one or more business process models. Business modellers (i.e. participants of Group 1) were provided with this feature model and were asked to perform their corresponding development tasks. Then, the artefacts developed by Group 1 were provided to participants of Group 2 in order to complete the development process. In particular, the tasks and the roles assigned in this session were the following:

1. **Business Process model (BPMN model).** This task was asked to be performed by *Business Engineers* (Group 1). In this case, this model was created by taking as input the feature model provided. Different business process tasks and events were linked to different IoT devices described in the provided feature model. This model is finally deployed into the BPMN engine to be ready for its execution. Once this is completed, a Preliminary feature model low-level definition is built automatically. This model contains just the IoT devices required for the execution of the BPMN model defined previously by the BP modeller.
2. **Feature Model Refinement and High-level Events definition.** This task was asked to be performed by *IoT Application Developers* (Group 2) based on the BPMN model created by business engineers. From this model,

a preliminary feature model selection was automatically generated. The refinement of this model included the configuration of all the IoT-specific device data that was required actually to work with the selected devices. It was deployed into the Config Server so the running instances of the BPMN model could invoke it. Then, Node-RED Flow Templates were built automatically and included the flows required between IoT devices to run the scenario modelled in the BPMN model properly. These flow templates needed to be completed to support all the high-level events required by the BPMN process to start or complete its tasks. Once flows were defined, they were deployed inside the FloWare Platform.

Once each development task was completed, each participant was asked to fill in the NASA-TLX questionnaire to obtain their feedback, which complemented the notes we took on the observation of their behaviour throughout the entire session.

## 7.4 Analysis of the Results

**Effectiveness.** We measured the effectiveness as the grade of task completion in such a way the different asked models and artefacts were completed if it was logically and syntactically correct. To facilitate this evaluation, a master model was used as a reference point. The models created for each participant were independently evaluated by two of us to reduce subjectivity. Next, both corrections were analysed together, and an agreed mark was decided for each model by the two evaluators. For task 1, we obtained grades between 60% and 100%, obtaining an average mark of 81%. For task 2, we obtained grades between 70% and 100%, obtaining an average mark of 86%. Thus, we can consider that our MDE approach is effective enough to support the design and execution of IoT-enhanced BPs.

Regarding Task 1, all the participants of Group 1 were able to perform the proposed modelling task. They properly used the developed web tool in order to create the BPMN model that was required by the motivating example. The most significant problems that we detected were related to the use of pools or lanes since some participants initially defined IoT devices as independent pools. They forgot that IoT devices must be created as lanes of a previously created pool that represent the FloWare system in which the IoT device was classified. Other participants created a duplicated pool that represents the same IoT system, including only one lane associated with a different IoT device in each pool. The correct solution was creating a unique pool with tool lanes associated with the two IoT devices. Once we reminded participants of the correct use of pools and lanes in our modelling approach, they found the organisation of IoT devices in IoT systems a very useful characteristic in order to create IoT-enhanced BPs when a great number of devices are available. Another modelling problem we detected was that some participants created specific lanes or pools to model the IoT Devices that trigger the high-level events, instead of defining them in a message flow connected to the Physical World pool. We use this modelling solution since it facilitates including not only events triggered by a unique IoT device but also complex events in which several IoT devices or other context conditions must be considered, as we demonstrated in our previous work [14],[25]. However, due to the suggestions of some participants, we plan to investigate whether our modelling solution can be adapted to use separate pools to represent high-level events instead of using only one Physical World pool. As far as the deployment of the BPMN model into the BPMN engine of the microservice architecture (see Figure 10), no significant issues were detected since this task was automatically done by the web tool we provided participants with.

Regarding Task 2, all participants in Group 2 successfully completed the required task. They were able to successfully configure the IoT devices required for the dishware dispensing process using the capability model and create the necessary Node-RED flows to support the high-level events defined in the BPMN model. Participants generally found the support provided by the FloWare platform to be useful. However, some participants suggested the possibility of integrating tools for working with Node-RED streams and feature models into a single tool. Currently, using separate tools allows us to translate the configuration of the IoT scenario through feature models to any IoT platform, with all the advantages of maintenance and evolution that this approach brings. We believe that creating a new tool to integrate with the feature model editor would be a complex task and may not provide the aforementioned benefits. Nonetheless, we plan to thoroughly study this option as part of our future work.

**Efficiency.** It was measured by comparing the times obtained by participants in the performance of the proposed task with the times taken by expert users such as us. Table 1 gathers these times and shows that the efficiency obtained was 0.75 for Task 1 and 0.73 for Task 2, which are quite acceptable values considering that the better efficiency is 1.

**Table 1** Results of the efficiency. Times in minutes

| **Subjects** | **Task 1** | **Task 2** |
|---|---|---|
| *Experts* | 22,18 | 12,10 |
| *Average(Experts)* | 20 | 11 |
| *Participants* | 23,25,20,20,27,25,35,25,30,28 | 10,15,15,20,15,20,15,15,10,15 |
| *Average(Participants)* | 26.50 | 15 |
| **Efficiency** | **0.75** | **0.73** |

**User Acceptance.** The results of the NASA-TLX questionnaire are shown in Table 2. In this questionnaire, the highest scores represent the worst results. Thus, mental load (*Men.L*), physical/temporal demand (*Phy. D*, *Temp. D*), effort (*Effort*), and frustration (*Frust*) are rated between very low (value 0) and very high (value 100); and the performance (*Perf*) is rated between very good (value 0) and very bad (value 100). Table 2 shows the average (*Avg*), the median (*Med*), the standard deviation (*SD*), the best result (*Best*), and the worst result (*Worst*).

In order to compare tasks, the NASA-TLX proposes to calculate a pondered global workload for each task [39]. To facilitate the interpretation of this global score, [40] presents a descriptive analysis of over 1000 global NASA-TLX scores from over 200 publications. This analysis obtained an average global score of 48.74. The minimum and maximum scores were 8 and 80, respectively. As we can see in Table 2, the global workload obtained for each task is lower than the average obtained in this analysis, which let us consider the obtained results as good.

From a general point of view, although Task 2 obtained slightly better results than Task 1, both tasks were ranked with acceptable values in the analysed factors. The obtained values lead us to consider that participants felt comfortable enough when creating an IoT-enhanced BP by means of BPMN, feature models, and Node-RED flows. The performance was the best-valued factor in both tasks which indicates that participants found the proposed development environment useful and efficient. The worst valued aspect was also the same in both tasks: the mental load. It is an expected result if we consider the mental effort that participants had to make during the experiment in order to properly understand both the new MDE approach we presented to them and the scenario we asked them to develop.

**Table 2** NASA-TLX results

| **Param.** | **Task 1** | | | | | **Task 2** | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | *Avg* | *Med* | *SD* | *Best* | *Worst* | *Avg* | *Med* | *SD* | *Best* | *Worst* |
| *Men.L* | 26.50 | 22.50 | 19.30 | 10 | 75 | 22.50 | 17.50 | 18.89 | 5 | 65 |
| *Phy.D* | 8.50 | 5.00 | 11.07 | 5 | 40 | 20.50 | 5.00 | 27.83 | 5 | 75 |
| *Temp.D* | 23.50 | 17.50 | 21.09 | 5 | 75 | 19.50 | 10.00 | 17.23 | 5 | 50 |
| *Perf.* | 17.50 | 12.50 | 18.45 | 5 | 65 | 12.50 | 7.50 | 10.07 | 5 | 35 |
| *Effort* | 23.50 | 17.50 | 19.73 | 10 | 75 | 18.00 | 17.50 | 14.94 | 5 | 50 |
| *Frust.* | 21.50 | 15.00 | 20.55 | 5 | 75 | 18.50 | 10.00 | 19.44 | 5 | 55 |
| **Global** | **22.24** | | | | | **28.22** | | | | |

## 7.5 Discussion of the Results

The results obtained in this experiment allow us to accept the proposed hypothesis and conclude that the presented model-driven approach allows an effective collaborative development to create IoT-enhanced Business Processes in an interdisciplinary way.

As we have explained in the previous sections, the participants of each group, which plays a different development role, could perform their development activities without the need to participate in the development of the other software artefacts. For instance, Business Engineers were able to create the BPMN model required by the Smart Canteen scenario without participating in the creation of the initial feature model that includes the abstract representation of IoT devices. In the same way, IoT App Developers were able to configure the IoT devices required by this model as well as create the Node-RED flows that support high-level events without participating in the creation of the BPMN model.

Also, note that the analysis of the previous subsection has introduced the main results obtained for the proposed development activities in an individual and independent way. However, during the experiment, we were also able to evaluate that the proposed development environment worked fine during the collaborative development.

The web tool that supports the creation of BPMN models and the customised FloWare platform were integrated in order to interchange the software artefacts that were required to create an IoT-enhanced BP according to the proposed model-drive approach (See Figure 1). The web tool was able to import the feature model with the abstract descriptions of IoT devices that were created with the customised FloWare platform and generate a preliminary feature selection based on the modelled process. This feature selection was loaded by the customised FloWare platform in order to be completed with the corresponding IoT device configuration. In the same way, the web tool was able to send the high-level events defined in the process to the customised FloWare platform in order to be supported by Node-RED flows. In general, the environment worked successfully, and only some minor bugs affecting communication among the tools were detected and fixed accordingly.

Finally, we think the proposed solution can facilitate further maintenance and evolution of an IoT-enhanced BP. The decoupling of the supporting software artefacts, as well as the independence provided to developers, can contribute to this issue. However, these challenges require a more precise evaluation.

## 7.6 Threats to Validity

The various threats that could affect the results of this experiment and the measures that we took were the following:

***Validity of conclusions.*** This validity concerns the relationship between the treatment and the outcome. The random heterogeneity of subjects threatened our experiment. This threat appears when some users within a user group have more experience than others. As commented above, we recruit participants in such a way that members of the same group had all similar profiles. This helps to minimise the heterogeneity of subjects. In addition, this threat was also minimised with (1) the demographic questionnaire that allowed us to evaluate the knowledge and experience of each participant beforehand and detect possible shortcomings; and (2) the training sessions in which all subjects participated to have a similar background in the technologies required to perform the proposed tasks. In these training sessions, we taught participants the technology required to apply our approach, and we also reinforced some basic notions we consider opportune from the analysis of the questionnaire results.

***Internal validity.*** Our experiment was threatened by the hypothesis guessing threat: when people might try to figure out the purpose and intended result of the experiment and are likely to base their behaviour on their guesses. We minimised this threat by hiding the goal of the experiment (i.e., the hypotheses to be validated were not shared with the participants). Note also that we introduced some subjectivity when grading the created solutions by comparing them with a master one. To reduce this problem, each solution was evaluated twice. In addition, some participants asked for clarifications during the experiment regarding using our model-driven approach. We answered all these questions by clarifying notions already introduced either in the case study presentation or in the training sessions. We were very careful not to introduce any help about the solution they needed to develop. Despite this, this may be considered a threat to this experiment.

***External validity.*** This type of validity concern is related to conditions that may limit our ability to generalise the experiment's results to industrial practice. This treatment is reduced by making the experimental environment more realistic. Thus, we provided participants with an experimental setting representative of industrial practice. Note that participants of Group 1, to create a BPMN model, could use the *IoT-enhanced BP web tool* that is an extended version of BPMN.io[13], one of the most used open-source BPMN modellers. Instead, participants of Group 2 used the customised *FloWare Modelling Tool* to derive the feature model configuration and apply technological information inside that, and the *FloWare platform*, integrated with the Node-RED tool, for developing event-driven IoT solutions. In addition, participants did not face the development of a toy example, but they were proposed to support an example based on a real scenario [19].

# 8 Related Work

Several works face the necessity of modelling BPs by including IoT elements, the so-defined *IoT-enhanced Business Processes* [1]. In the following, we discuss existing works that seek to include IoT concepts within BPs by extending the existing BPMN metamodel or using the original BPMN notation. Finally, we discuss the above-analysed works and compare them with this work.

## *BPMN Metamodel Extensions*

Different works aim to extend the original BPMN notation with new concepts to model requirements imposed by IoT systems and devices. Some of these works focus on extending the event element of the BP. In detail, in [41], authors introduce an extended BPMN metamodel to model and execute IoT event stream processing units within BPs. They add tasks for *event stream specification* and *event stream processing*, allowing the management of IoT stream events and parameters. Similarly, in [42], the authors extend the event elements of BPMN to support IoT devices introducing new attributes, such

---

[13] BPMN.io: https://bpmn.io/

as *device ID* and *sensor data*, to model events involving IoT devices. Another approach [43] proposes a multi-level framework for real-time monitoring and response to real-world events using IoT technology. They introduce a new *event annotation* element to specify the binding points between external events and the BP model. In [44], the authors extend *event* and *task* BPMN elements to represent IoT input technologies and physical objects, combining the business model with a decision model to analyse changing environments and make decisions accordingly. The work in [45] introduces a three-layer architecture for IoT-aware BP execution. They extend BPMN with data variables to enrich the models with data obtained from physical objects and specify the influence of IoT devices on the process.

To explicitly represent IoT-specific elements, [46] extends BPMN with new classes: *Sensor Device*, *Sensor Service*, and *Handler*. These extensions allow for modelling and designing IoT-enhanced BPs. In [47], the authors extend event BPMN with elements for representing context data of devices and introduce the SenSoMod ontology to separate concerns about IoT sensors and their relationships. For modelling complex cyber-physical systems, [48] introduces an extension to BPMN, specialising service tasks as *physical tasks* and *cyber tasks* for activities like monitoring and control. The approach in [49] introduces new elements, such as *PhysicalEntity*, *SensingTask*, and *ActuatingTask*, to model interactions between IoT devices and BPs. It relies on the jBPM toolkit for deployment.

In the context of Industry 4.0, [50] proposes a process modelling language and method tailored to this domain. New BPMN elements such as *IoT Device*, *IoT Data*, and *Private/Public/Hybrid clouds* are presented in a conveyor belt industry case study. In [51], the WSN task concept is introduced to represent the generic meta-abstraction actions of a Wireless Sensor Network. They extend the BPMN metamodel with a meta-abstraction perspective to represent sensors, actuators, and control systems. In detail, BPMN pools are used to represent WSN elements, while Service Tasks were enriched with additional elements to represent the meta-abstraction *action* and *tag* elements proposed. Code generation is used to execute the parts of the BP model that cannot be executed in a process engine. The authors of [52] extend the BPMN metamodel to represent physical entities and their interaction with devices. They introduce concepts such as *PhysicalObject*, *SensingTask*, *ActuatingTask*, *SensingAssociation*, and *ActuatingAssociation*.

In [53], authors propose adding IoT-related concepts and elements to BPMN, including *IoT events*, *devices*, *data*, *event gateways*, and *event subprocesses*. Different ontologies are used to model these concepts. The work in [54] extends BPMN by providing additional attributes to *tasks*, *task groups*, and *sub-processes*. These attributes are used to reference external models and model the interactions between the BP and the sensor network. In [55], an extension of the BP representation is provided through annotation elements. These elements support the specification of service tasks related to IoT devices and retrieve information using ontologies. The focus of this work is representing IoT-enhanced BPs for simulation-based analysis of complex systems for developing digital twins. Finally, [56] discusses extending the BPMN metamodel with IoT elements for handling *sensors*, *actuators*, and *sensor groups*. The approach focuses on modelling rather than execution or technology details.

### *Approaches Using the Standard BPMN metamodel*

In contrast, other studies propose to use the original BPMN constructs to model IoT-enhanced BPs. In this case, the BPMN notation is often used to construct a non-executable modelling artefact that needs to be transformed into another language or technology to be executed.

Specifically targeting Guard-Stage-Milestone (GSM) technology, in [57], the authors propose a method for monitoring cross-organisational BPs using smart objects. The proposed approach involves creating BP models following the BPMN standard and generating declarative extended GSM specifications from them semi-automated. These GSM specifications are then implemented and executed on smart objects, with a dedicated infrastructure required for each object. In [58], the authors investigate the suitability of BPMN for modelling wireless sensor network (WSN) applications. The authors examine the capabilities of BPMN in representing the unique characteristics of WSNs, such as data aggregation, energy consumption, and routing. They compare BPMN with other modelling techniques commonly used in WSNs, such as Petri nets and state charts, and evaluate its ability to capture the WSN requirements, concluding that BPMN is a suitable technique for modelling WSN applications. Then, Java and C# codes are generated to be deployed on the Mote Runner WSN platform, the run-time environment for mote-class wireless sensor networks. In [59], the author's suggestion is to incorporate smart objects into the BP through the use of jBPM, which is a software suite that complies with BP Model and Notation (BPMN) standards. This suite allows for creating application logic by combining local and remote service tasks using the BPMN workflow model. To achieve this integration, a Java-based programming framework is an intermediary between the smart objects and the BP definition, generating all the necessary components. This implies that the technical support is only limited to the jBPM technology to interact with the Java framework. The concept and software prototype for integrating smart devices, such as smartphones and smartwatches, as resources in BPs are presented in [60]. The authors argue that using smart devices can enhance BP performance and increase employee satisfaction. In this case, devices are represented as resources (pool) in a BPMN model and use service tasks, defined in the paper as smart device tasks, to manage them. The integration with any IoT-related technology is allowed thanks to the External Service Task component, which interact with any other functionalities implemented outside of the model. In [61] propose to define BPs at the process layer using standard BPMN and achieve the integration between IoT devices and BPs at the technical level through the Bosch IoT Things service. They suggest using BPMN

| ***Paper*** | ***Modelling*** | | | | | | ***Development*** | ***Deployment*** |
|---|---|---|---|---|---|---|---|---|
| **N.** | **BPMN As-Is** | **IoT-Modelling Level** | **Context Data** | **SoC** | **Interdisciplinary Team** | **MDE Based** | **Technology Independent** | **Execution Support** |
| [41] | - | - | ✓ | - | - | - | Extended process engine | ✓ |
| [42] | - | - | ✓ | - | - | - | - | - |
| [43] | - | | ✓ | ✓ | - | - | - | - |
| [44] | - | ✓ | ✓ | ✓ | - | - | - | - |
| [45] | - | - | ✓ | - | - | - | - | - |
| [46] | - | ✓ | - | - | - | - | Extended process engine | ✓ |
| [47] | - | ✓ | ✓ | ✓ | - | - | - | - |
| [48] | - | ✓ | - | - | - | - | - | - |
| [49] | - | ✓ | - | - | - | - | - | - |
| [50] | - | ✓ | - | - | - | - | - | - |
| [51] | - | ✓ | - | - | - | - | Specific code generated | ✓ |
| [52] | - | ✓ | - | - | - | - | - | - |
| [53] | - | ✓ | ✓ | ✓ | - | - | - | - |
| [54] | - | ✓ | ✓ | ✓ | - | - | - | - |
| [56] | - | ✓ | - | - | - | - | - | - |
| [55] | - | ✓ | - | - | - | ✓ | - | - |
| [57] | ✓ | ✓ | - | - | - | - | Prop. Infrastructure | ✓ |
| [58] | ✓ | ✓ | - | - | - | - | Mote Runner Prop. | ✓ |
| [59] | ✓ | ✓ | - | - | - | - | jBPM toolkit | ✓ |
| [60] | ✓ | ✓ | - | - | - | - | ✓ | ✓ |
| [61] | ✓ | - | ✓ | - | - | - | - | - |
| [62] | ✓ | ✓ | - | - | - | - | Callas | ✓ |
| ***FloBP*** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

**Table 3** Comparison of the analysed works. - : not supported; ✓: supported

tasks to represent the activities performed by IoT devices and BPMN events to capture the triggering of events by these devices. The existing BPMN gateways can be employed to model decision points or alternative paths based on IoT device inputs. Despite the authors do not explicitly mention the deployment process of the BPMN models that integrate IoT devices, we can infer that the deployment of the BPMN models, once created and defined, is strictly related to the Bosch IoT Things service as a platform to execute the BPs involving IoT devices. Finally, in [62], BPMN is used to model the behaviour of IoT devices within a BP by defining the activities and events that take place, as well as the order in which they occur. IoT devices are represented as tasks within the BPMN model, and their interactions with other tasks and events can be defined using various BPMN elements. However, the IoT devices integration is limited only to devices for which the Callas programming language and its virtual machine are available. This does not allow a technology-independent solution.

### *Comparison and Discussion*

As a summary, Table 3 presents the most important characteristics of the above-introduced approaches regarding the development of IoT-enhanced BPs compared with the one proposed in this work.

We compared the analysed approaches in three macro-areas: Modelling, Development, and Deployment. In terms of Modelling, we considered the following characteristics: *BPMN As-Is*, as the possibility to use the BPMN metamodel and its elements without additional changes; *IoT-Modelling Level*, whether the modelling approach incorporates IoT devices and/or their interactions; *Context Data*, as the support for managing context data at the modelling level; *Separation of Concerns*, as the ability to separate different concerns within the overall solution; *Interdisciplinary Team*, as the involvement of multiple professional actors and their collaboration within the approach. *MDE Based*, whether the approach is based on MDE.

Regarding the Development aspect, we examined the characteristic of *Technology Independent*, as The ability to execute IoT-enhanced BP models without being dependent on a specific engine or proprietary solution. Lastly, for the Deployment aspect, we analysed the characteristic of *Execution Support*, ensuring that the execution of IoT-enhanced BPs is supported.

The analysis reveals that some approaches suggest *Extend the BPMN metamodel* to include new IoT elements, preventing the use of numerous BPMN engines available for executing BPs. This implies that most BPMN-based approaches that extend the metamodel cannot guarantee *Technology Independence*, as they become incompatible with existing engines. To execute processes, providing specific *Execution Support* is necessary, as in the case of [41, 46] where an existing process engine is extended to support their new constructs, and [51], which generates code to execute the part of the BP model that cannot be executed in a process engine. However, these solutions provide execution platforms that are extremely coupled with a specific technology, which results in difficulty to be maintained and evolving with the changing technology requirements. Integrating IoT concepts into BP modelling can increase cognitive complexity and hinder effective stakeholder communication. In addition, studies [63, 64] have identified challenges in the excessive extension or enrichment of BP models with IoT information that can undermine the effectiveness of communication mechanisms among stakeholders. On the other hand, approaches using *BPMN metamodel As-Is* offer benefits such as maintaining the simplicity of the notation and compatibility with existing BPMN engines for execution. All the approaches that do not extend the metamodel provide execution support for the deployment of IoT-enhanced BPs through common engines. Indeed, many of these approaches transform BPMN models into other languages or technologies for execution and integration with IoT devices, resulting in strong dependence on specific technologies.

Most of the presented approaches integrate *IoT* devices or their interaction at the *Modelling Level.* This allows an accurate representation of the entire process as they consider the behaviour of the IoT device's data to perform operations. Approaches that do not explicitly model IoT devices rely on external components for establishing connections. Some use external event annotators [41–43], while others extend the BPMN model with data variables retrieved from IoT devices [45]. Furthermore, few of these approaches consider *Context Data* description at the modelling level, especially the processing of low-level sensor data to obtain high-level information suitable for BPs. The design and comprehension of processes involving a large number of IoT devices and low-level data manipulation can lead to high complexity.

Several approaches employ the *Separation of Concerns* design principle to address the integration of IoT in modelling proposals. These approaches propose frameworks that separate the BPMN modelling concern from IoT integration. For instance, [44] combines an extended BPMN model with a Decision Model to define decisions based on IoT data. In [47], a sensor model is combined with a BPMN extension to represent sensors, context, and their relationships. [53] proposes semantic description using ontologies to integrate BPs and IoT elements. Finally, [54] links a functional model based on a sensor ontology to import device data with BPMN models, ensuring the separation of concerns between these domains.

None of the above-described approaches considers the *Interdisciplinary Team* necessary to develop IoT-enhanced BPs. In managing different contexts, such as the IoT and the BP ones, we believe it is necessary to explicitly clarify actors and the various tasks they should perform. This way, different experts with unique backgrounds can manage multiple aspects of IoT-enhanced BPs.

Finally, only in [55] the development of IoT-enhanced BPs is supported by a MDE methodology. The authors propose an MDE approach that extends the BPMN metamodel for the simulation of digital twins through IoT-enhanced BPs. Different modelling phases are highlighted to reach the simulation aspect, retrieving device data to automatically reconfigure the simulation BP models ad then to make the digital twin continuously coherent and compliant with its physical counterpart.

***FloBP.*** Our approach uses the original primitives of BPMN to specify IoT-enhanced BPs. Still, unlike other approaches, our models can be deployed and executed in any BPMN-compliant engine, regardless of the technology used by IoT devices. This technological flexibility is possible because our proposal relies on a microservice architecture, where microservices are intermediaries between the BP and IoT devices. Microservices provide a standard way to interact with IoT devices through an API, which can be implemented in different languages and frameworks, depending on the device being used. Unlike other approaches, we pay attention to how low-level sensor data can be processed to obtain high-level information data that is more appropriate for the BP. Indeed, we apply the Separation of Concerns principle by combining BPMN and feature models. The IoT devices that participate in the BP and the high-level events that must be managed within the process are represented in the BPMN model by using the standard notation. Thus, the high-level requirements of an IoT-enhanced BP are all defined in one model, which provides a more intuitive and cohesive view to facilitate their analysis. We propose a microservices architecture to support the execution of IoT-enhanced BP models. Overall, the entire approach is based on the MDE methodology. In this sense, we believe it is necessary to provide a level of abstraction in modelling IoT-enhanced BPs and simplify the entire development process. In addition, by applying MDE, we maximise reusability through standardised models (both feature models and BPMN ones), simplifying design processes by incorporating recurring design patterns, i.e., the feature model structure, and promoting better communication among individuals and teams working on the IoT solution by standardising the best practices to use in this application domain.

## 9 Conclusions and Future Work

In this work, we have presented *FloBP*, an interdisciplinary MDE approach that supports the development of IoT-enhanced business processes. The approach follows the Separation of Concerns principle, where expert actors from different domains are involved in each step. We address heterogeneity in both the IoT and BP domains. Initially, we focus on providing a modelling structure that represents the comprehensive knowledge of a specific IoT domain. This knowledge is acquired and modelled using feature models, serving as a foundation for BP modelling. Through BP modelling, we create an IoT-enhanced BP that can interact with IoT devices to perform operations and retrieve results. Additionally, the IoT-enhanced BP can be designed to trigger high-level events based on IoT device data. The decisions within the BP are reflected in the feature model as selected features. Experts then provide the technical information required for these selected features. Once completed, high-level events are automatically generated based on the inserted model information. These high-level events serve as template flows to handle device data and trigger the IoT-enhanced BP when specific events occur.

To support the interdisciplinary development process, we employ a collaborative development environment with various tools. These tools enable professionals to independently fulfil their development responsibilities without direct involvement in other artefact development. However, it is crucial to maintain integrated tools for data interchange necessary during the entire modelling process. A microservices infrastructure provides technological support for the overall approach. The microservice architecture allows each artefact to be developed with a high degree of independence and facilitates the decoupled execution of software components. This architectural approach aligns with the interdisciplinary nature of IoT-enhanced BPs and effectively addresses their concerns. The *FloBP* approach was extensively discussed within the context of the Smart Canteen scenario, resulting in a successful modelling and development solution. Furthermore, a case-study validation was conducted with users, yielding valuable insights and confirming the effectiveness of the *FloBP* approach in facilitating the development of IoT-enhanced business processes.

Looking ahead, we have a clear roadmap for further enhancements and advancements. One crucial aspect is simulating the complete IoT-enhanced BP before deploying the IoT solution. This simulation could provide valuable insights into performance, identify potential bottlenecks, and enables optimisation of efficiency. Additionally, we aim to extend the validation of the *FloBP* approach to involve further experts from both academia and industry, broadening its applicability and ensuring its effectiveness in diverse contexts.

**Acknowledgments.** This work has been partially supported by Marche Region in implementation of the financial programme POR MARCHE FESR 2014-2020, project "Miracle" (Marche Innovation and Research fAcilities for Connected and sustainable Living Environments), CUP B28I19000330007, and by the MIUR project PRIN "Fluidware" (A Novel Approach for Large-Scale IoT Systems, n. 2017KRC7KT). Also, this work is part of the RDI project PID2020-114480RB-I00 funded by MCIN/AEI/1013039/501100011033.

## References

[1] Torres, V., Serral, E., Valderas, P., Pelechano, V., Grefen, P.: Modeling of IoT devices in business processes: A systematic mapping study. 22nd IEEE Conference on Business Informatics, CBI 2020 **1**, 221–230 (2020)

[2] Janiesch, C., Koschmider, A., Mecella, M., Weber, B., Burattin, A., Di Ciccio, C., Fortino, G., Gal, A., Kannengiesser, U., Leotta, F., *et al.*: The internet of things meets business process management: a manifesto. IEEE Systems, Man, and Cybernetics Magazine **6**(4), 34–44 (2020)

[3] Beverungen, D., Buijs, J.C.A.M., Becker, J., Ciccio, C.D., van der Aalst, W.M.P., Bartelheimer, C., vom Brocke, J., Comuzzi, M., Kraume, K., Leopold, H., Matzner, M., Mendling, J., Ogonek, N., Post, T., Resinas, M., Revoredo, K., del-Río-Ortega, A., Rosa, M.L., Santoro, F.M., Solti, A., Song, M., Stein, A., Stierle, M., Wolf, V.: Seven paradoxes of business process management in a hyper-connected world. Bus. Inf. Syst. Eng. **63**(2), 145–156 (2021)

[4] Zhang, H., Babar, M.A., Tell, P.: Identifying relevant studies in software engineering. Inf. Softw. Technol. **53**(6), 625–637 (2011)

[5] Compagnucci, I., Corradini, F., Fornari, F., Polini, A., Re, B., Tiezzi, F.: A systematic literature review on iot-aware business process modeling views, requirements and notations. Software and Systems Modeling, 1–36 (2022)

[6] De Cremer, D., Nguyen, B., Simkin, L.: The integrity challenge of the internet-of-things (iot): on understanding its dark side. Journal of Marketing Management **33**(1-2), 145–158 (2017)

[7] Gupta, B.B., Quamara, M.: An overview of internet of things (iot): Architectural aspects, challenges, and protocols. Concurrency and Computation: Practice and Experience **32**(21), 4946 (2020)

[8] Feljan, J., Karapantelakis, A., Mokrushin, L., Inam, R., Fersman, E., Azevedo, C., Raizer, K., Souza, R.: Tackling iot complexity. Ericsson review (English ed.) **95**(2), 60–69 (2017)

[9] Chapline, G., Sullivan, S.: Systems engineering for lifecycle of complex systems. Engineering Innovations (NASA) (2010)

[10] Schmidt, D.C.: Model-driven engineering. Computer-IEEE Computer Society- **39**(2), 25–31 (2006)

[11] Chen, S., Xu, H., Liu, D., Hu, B., Wang, H.: A vision of iot: Applications, challenges, and opportunities with china perspective. IEEE Internet of Things journal **1**(4), 349–359 (2014)

[12] Lee, I.: The Internet of Things for enterprises: An ecosystem, architecture, and IoT service business model. Internet of Things **7**, 100078 (2019)

[13] Corradini, F., Fedeli, A., Fornari, F., Polini, A., Re, B.: Floware: a model-driven approach fostering reuse and customisation in iot applications modelling and development. Software and Systems Modeling, 1–28 (2022)

[14] Valderas, P., Torres, V., Serral, E.: Modelling and executing iot-enhanced business processes through BPMN and microservices. J. Syst. Softw. **184**, 111139 (2022)

[15] Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Q. **28**(1), 75–105 (2004)

[16] Peffers, K., Tuunanen, T., Rothenberger, M.A., Chatterjee, S.: A design science research methodology for information systems research. J. Manag. Inf. Syst. **24**(3), 45–77 (2008)

[17] Avison, D.E., Lau, F., Myers, M.D., Nielsen, P.A.: Action research. Communications of the ACM **42**(1), 94–97 (1999)

[18] Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empir. Softw. Eng. **14**(2), 131–164 (2009)

[19] Bagayatkar, T., Pawar, S., Salvi, V., Gawande, K.: Virtual smart canteen system. In: 2022 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES), vol. 1, pp. 503–508 (2022). IEEE

[20] Anirudh, A., Pandey, V.K., Sodhi, J., Bagga, T.: Next generation indian campuses going smart. International Journal of Applied Business and Economic Research **15**(21), 385–398 (2017)

[21] Motta, R.C., de Oliveira, K.M., Travassos, G.H.: On challenges in engineering iot software systems. In: Proceedings of the XXXII Brazilian Symposium on Software Engineering, pp. 42–51 (2018)

[22] Nysetvold, A.G., Krogstie, J.: Assessing business process modeling languages using a generic quality framework. Advanced Topics in Database Research **5**, 79–93 (2006)

[23] Hadzovic, S., Mrdovic, S., Radonjic, M.: Identification of iot actors. Sensors **21**(6), 2093 (2021)

[24] BPMN: Business Process Model and Notation Concepts. http://www.omg.org/spec/BPMN/20100501 Accessed 2023-25-01

[25] Valderas, P., Torres, V., Serral, E.: Towards an interdisciplinary development of iot-enhanced business processes. Business & Information Systems Engineering, 1–24 (2022)

[26] Vuppalapati, C.: Building Enterprise IoT Applications, 1st edn. CRC Press, Boca Raton (2019)

[27] Kazmi, A., Jan, Z., Zappa, A., Serrano, M.: Overcoming the heterogeneity in the internet of things for smart cities. In: Podnar Žarko, I., Broering, A., Soursos, S., Serrano, M. (eds.) Interoperability and Open-Source Solutions for the Internet of Things, pp. 20–35. Springer, Cham (2017)

[28] Gascueña, J.M., Navarro, E., Fernández-Caballero, A.: Model-driven engineering techniques for the development of multi-agent systems. Eng. Appl. Artif. Intell. **25**(1), 159–173 (2012)

[29] Benavides, D., Trinidad, P., Ruiz-Cortés, A.: Automated reasoning on feature models. Seminal Contributions to Information Systems Engineering: 25 Years of CAiSE, 361–373 (2013)

[30] da Cunha, C., Agard, B., Kusiak, A.: Design for cost: Module-based mass customization. IEEE Transactions on Automation Science and Engineering **4**(3), 350–359 (2007)

[31] Bermudez-Edo, M., Elsaleh, T., Barnaghi, P., Taylor, K.: Iot-lite: a lightweight semantic model for the internet of things. In: 2016 INTL IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (uic/atc/scalcom/cbdcom/iop/smartworld), pp. 90–97 (2016). IEEE

[32] Harmon, P., Wolf, C.: Business Process Modeling Survey. Business Process Trends. http://www.bptrends.com/surveys/Process_Modeling_Survey-Dec_11_FINAL.pdf Accessed 2023-02-15

[33] Leopold, H., Mendling, J., Günther, O.: Learning from quality issues of bpmn models from industry. IEEE software **33**(4), 26–33 (2015)

[34] Lewis, J., Fowler, M.: Microservices: a Definition of this New Architectural Term. https://martinfowler.com/articles/microservices.html Accessed 2023-02-21

[35] Fowler, M.: Microservice Trade-Offs. https://martinfowler.com/articles/microservice-trade-offs.html Accessed 2023-02-21

[36] Völter, M.: Software architecture-a pattern language for building sustainable software architectures. In: EuroPLoP, pp. 31–66 (2006)

[37] Vogel-Heuser, B., *et al.*: Usability experiments to evaluate uml/sysml-based model driven software engineering notations for logic control in manufacturing automation. Journal of Software Engineering and Applications **7**(11), 943 (2014)

[38] Zou, Y., Zhang, Q., Zhao, X.: Improving the usability of e-commerce applications using business processes. IEEE Transactions on Software Engineering **33**(12), 837–855 (2007)

[39] Hart, S.G., Staveland, L.E.: Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research **52**, 139–183 (1988)

[40] Grier, R.A.: How high is high? a meta-analysis of nasa-tlx global workload scores. In: Proceedings of the Human Factors and Ergonomics Society Annual Meeting, vol. 59, pp. 1727–1731 (2015). SAGE Publications Sage CA: Los Angeles, CA

[41] Appel, S., Kleber, P., Frischbier, S., Freudenreich, T., Buchmann, A.: Modeling and execution of event stream processing in business processes. Information Systems **46**, 140–156 (2014)

[42] Chiu, H.-H., Wang, M.-S.: Extending event elements of business process model for internet of things. In: 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing, pp. 783–788 (2015). IEEE

[43] Mandal, S., Hewelt, M., Weske, M.: A framework for integrating real-world events and business processes in an iot environment. In: Panetto, H., Debruyne, C., Gaaloul, W., Papazoglou, M., Paschke, A., Ardagna, C.A., Meersman, R. (eds.) On the Move to Meaningful Internet Systems. OTM 2017 Conferences, pp. 194–212. Springer, Cham (2017)

[44] Yousfi, A., Batoulis, K., Weske, M.: Achieving business process improvement via ubiquitous decision-aware business processes. ACM Transactions on Internet Technology (TOIT) **19**(1), 1–19 (2019)

[45] Schönig, S., Ackermann, L., Jablonski, S., Ermer, A.: An integrated architecture for iot-aware business process execution. In: Gulden, J., Reinhartz-Berger, I., Schmidt, R., Guerreiro, S., Guédria, W., Bera, P. (eds.) Enterprise, Business-Process and Information Systems Modeling, pp. 19–34. Springer, Cham (2018)

[46] Cheng, Y., Zhao, S., Cheng, B., Chen, X., Chen, J.: Modeling and deploying iot-aware business process applications in sensor networks. Sensors **19**(1), 111 (2018)

[47] Dörndorfer, J., Seel, C.: A framework to model and implement mobile context-aware business applications. Modellierung 2018 (2018)

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

[49] Meyer, S., Ruppen, A., Magerkurth, C.: Internet of things-aware process modeling: integrating iot devices as business process resources. In: Advanced Information Systems Engineering: 25th International Conference, CAiSE 2013, pp. 84–98 (2013). Springer

[50] Petrasch, R., Hentschke, R.: Process modeling for industry 4.0 applications: Towards an industry 4.0 process modeling language and method. In: 2016 13th International Joint Conference on Computer Science and Software Engineering (JCSSE), pp. 1–5 (2016). IEEE

[51] Mottola, L., Picco, G.P., Oppermann, F.J., Eriksson, J., Finne, N., Fuchs, H., Gaglione, A., Karnouskos, S., Montero, P.M., Oertel, N., *et al.*: make sense: Simplifying the integration of wireless sensor networks into business processes. IEEE Transactions on Software Engineering **45**(6), 576–596 (2017)

[52] Sperner, K., Meyer, S., Magerkurth, C.: Introducing entity-based concepts to business process modeling. In: Business Process Model and Notation: Third International Workshop, BPMN 2011, Lucerne, Switzerland, November 21-22, 2011. Proceedings 3, pp. 166–171 (2011)

[53] Suri, K., Gaaloul, W., Cuccuru, A., Gerard, S.: Semantic framework for internet of things-aware business process development. In: 2017 IEEE 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), pp. 214–219 (2017). IEEE

[54] Gao, F., Zaremba, M., Bhiri, S., Derguerch, W.: Extending bpmn 2.0 with sensor and smart device business functions. In: 2011 IEEE 20th International Workshops on Enabling Technologies: Infrastructure for

Collaborative Enterprises, pp. 297–302 (2011)

[55] Bocciarelli, P., D'Ambrogio, A., Panetti, T.: A model-based framework for iot-aware business process management, vol. 15 (2023)

[56] Kirikkayis, Y., Gallik, F., Winter, M., Reichert, M.: Bpmne4iot: A framework for modeling, executing and monitoring iot-driven processes. Future Internet **15**(3) (2023)

[57] Baresi, L., Meroni, G., Plebani, P.: A gsm-based approach for monitoring cross-organization business processes using smart objects. In: Reichert, M., Reijers, H.A. (eds.) Business Process Management Workshops, pp. 389–400. Springer, Cham (2016)

[58] Caracaş, A., Kramp, T.: On the expressiveness of bpmn for modeling wireless sensor networks applications. In: Business Process Model and Notation: Third International Workshop, BPMN 2011, Lucerne, Switzerland, November 21-22, 2011. Proceedings 3, pp. 16–30 (2011)

[59] Dar, K., Taherkordi, A., Baraki, H., Eliassen, F., Geihs, K.: A resource oriented integration architecture for the internet of things: A business process perspective. Pervasive and Mobile Computing **20**, 145–159 (2015)

[60] Wehlitz, R., Rößner, I., Franczyk, B.: Integrating smart devices as business process resources – concept and software prototype. In: Service-Oriented Computing – ICSOC 2017 Workshops, pp. 252–257 (2018)

[61] Friedow, C., Völker, M., Hewelt, M.: Integrating iot devices into business processes. In: Advanced Information Systems Engineering Workshops: CAiSE 2018 International Workshops, pp. 265–277 (2018)

[62] Domingos, D., Martins, F.: Using bpmn to model internet of things behavior within business process. International Journal of Information Systems and Project Management **5**(4), 39–51 (2017)

[63] Melcher, J., Mendling, J., Reijers, H.A., Seese, D.: On measuring the understandability of process models. In: Business Process Management Workshops: BPM 2009 International Workshops, Ulm, Germany, September 7, 2009. Revised Papers 7, pp. 465–476 (2010). Springer

[64] Zugal, S., Pinggera, J., Weber, B.: Assessing process models with cognitive psychology. Enterprise modelling and information systems architectures (EMISA 2011), 177–182 (2011)