Mobile Cloud Business Process Management System for the Internet of Things: A Survey

CHII CHANG and SATISH NARAYANA SRIRAMA, University of Tartu, Estonia
RAJKUMAR BUYYA, University of Melbourne, Australia

The Internet of Things (IoT) represents a comprehensive environment that consists of a large number of smart devices interconnecting heterogeneous physical objects to the Internet. Many domains such as logistics, manufacturing, agriculture, urban computing, home automation, ambient assisted living, and various ubiquitous computing applications have utilized IoT technologies. Meanwhile, Business Process Management Systems (BPMSs) have become a successful and efficient solution for coordinated management and optimized utilization of resources/entities. However, past BPMSs have not considered many issues they will face in managing large-scale connected heterogeneous IoT entities. Without fully understanding the behavior, capability, and state of the IoT entities, the BPMS can fail to manage the IoT integrated information systems. In this article, we analyze existing BPMSs for IoT and identify the limitations and their drawbacks based on a Mobile Cloud Computing perspective. Later, we discuss a number of open challenges in BPMS for IoT.

CCS Concepts: • **Information systems → Mobile information processing systems**; **Process control systems;**

Additional Key Words and Phrases: Internet of Things, mobile cloud computing, business process management system, service oriented, review, state of the art, challenge

ACM Reference Format:
Chii Chang, Satish Narayana Srirama, and Rajkumar Buyya. 2016. Mobile cloud business process management system for the Internet of Things: A survey. ACM Comput. Surv. 49, 4, Article 70 (December 2016), 42 pages.
DOI: http://dx.doi.org/10.1145/3012000

1. INTRODUCTION

Emerging mature mobile and ubiquitous computing technology is hastening the realization of smart environments, in which the physical objects involved in our everyday life (food, parcels, appliances, vehicles, buildings, etc.) are connected. Many of the electronic devices are now granted with a certain intelligence to work together for us and enhance our lives. The core enabler is the Internet Protocol (IP) that is capable of providing the addressing mechanism for physical objects toward interconnecting everything with the Internet, which is known as the Internet of Things (IoT) [Gubbi et al. 2013]. The goal of IoT is to enhance the broad range of people's lives, including but not limited to agriculture, transportation, logistics, education, and healthcare [Atzori

This research is supported by Estonian Research Council grant PUT360.
Authors' addresses: C. Chang and S. N. Srirama, Mobile and Cloud Computing Laboratory, Institute of Computer Science, University of Tartu, Tartu 50090, Estonia; emails: chii.chang@acm.org, srirama@ut.ee; R. Buyya, Cloud Computing and Distributed Systems (CLOUDS) Laboratory, Department of Computing and Information Systems, University of Melbourne, VIC 3010, Australia; email: rbuyya@unimelb.edu.au.

DOI: http://dx.doi.org/10.1145/3012000

et al. 2010]. The industry predicts that in the year 2020, around 50 billion physical devices will be connected to the Internet [Evans 2011], and the economy revenue value will raise up to $1.9 trillion to $7.1 trillion [Middleton et al. 2013; Manyika et al. 2013; Lund et al. 2014].

Today, IoT has become one of the most popular topics for both industry and academia. Prior to the vision of IoT arising, the Cyber-Physical Systems (CPSs), which interconnected the physical entities with software systems, were usually isolated. Each subsystem has its own topology and communication protocols. In order to integrate the isolated CPS into the IoT vision, one promising approach is to apply a Service-Oriented Architecture (SOA)-based middleware solution [Loke 2003]. Fundamentally, SOA introduces the interoperability of heterogeneous isolated systems. By applying SOA, CPSs can manage individual devices as atomic services or they can configure a group of devices to provide composite services. For example, a mobile Internet-connected device can embed a web service to provide the atomic sensory information service to remote web service clients [Srirama et al. 2006; Chang et al. 2015a]. Further, a composite service in the cloud can exploit the data derived from multiple devices in a specific location to compute and generate the meaningful information to specific users [Conti et al. 2012].

In the last decade, Workflow Management Systems (WfMSs) have become one of the major components of service composition. Among the different WfMSs, Business Process Management Systems (BPMSs) have been broadly accepted as the de facto approaches in WfMS [Dumas et al. 2013], mainly because of the availability of tools and international standards such as Business Process Execution Language (BPEL) [Jordan et al. 2007], Business Process Model and Notation (BPMN) [Object Management Group 2011], and XML Process Definition Language (XPDL) [Workflow Management Coalition 2012]. BPMSs are "generic software system(s) that is driven by explicit process designs to enact and manage operational business processes" [van der Aalst et al. 2003]. BPMSs can provide the highly integrated platforms to manage comprehensive entities and activities involved in IoT systems. BPMSs also provide self-managed behavior in various IoT applications such as smart home systems [Loke 2003; Chang and Ling 2008]; crowd computing [Kucherbaev et al. 2013]; Wireless Sensor Networks [Sungur et al. 2013]; heating, ventilating, and air conditioning (HVAC) systems [Tranquillini et al. 2012]; mobile healthcare [Peng et al. 2014]; and so on. BPMSs let users (e.g., system administrators, domain scientists, regular end-users) easily manage the overall IoT system without getting involved in the low-level complex programming languages. Hence, they can focus on modeling the behavior and business processes of the things. Furthermore, with the optimized process model, the IoT system can provide self-adaptation in which it can autonomously react to or prevent the events based on runtime context information [Chandler 2015]. Next, we summarize a few use cases of IoT-driven BPMS.

1.1. Use Cases of IoT-Driven Business Process Management Systems

Logistics. By integrating the mobile cloud, IoT, and BPMSs, the logistics system can provide real-time tracking and controlling. For example, imagine that a cargo parking area needs to avoid the cargos with dangerous goods from being parked in close proximity. By deploying Radio-Frequency Identification (RFID), wireless sensors, and a mobile ad hoc network at the front-end vehicles, the front-end vehicles can maintain a temporary edge network to identify the goods in their cargos and to inform each other if their driver intends to park the vehicles in close proximity. The distant cloud-side management system can continuously track the environment and provide the instructions to the vehicle drivers regarding where the proper free space for parking the vehicles and cargos is [Glombitza et al. 2011].

Enterprise Resource Planning (ERP). The supply chain process in ERP systems is another example that can be improved by BPMSs for IoT (BPMS4IoT). As described in Schulte et al. [2014], by implementing wireless sensor networks, RFID, or other IoT technologies, the supply chain can be monitored in real time. Suppose Factory A has a production lane that requires the supply from Factory B urgently. Since IoT has been deployed in all of Factory A's partner businesses, Factory A is capable of identifying whether Factory B can produce and ship the supply to them in time or not based on analyzing Factory B's in-stock resources and shipping conditions. In the case that Factory A found that Factory B was unable to fulfill the task in time due to the shipping issue, Factory A can try to find a substitution by either distributing the order to multiple suppliers or finding an alternative supplier that can handle the entire order. If Factory A is unable to find an alternative solution and it has to postpone the production lane, the originally assigned workers for the production lane and the vehicles for shipment can be reallocated to the other tasks in order to reduce the waste of human resources.

Smart Building. BPMS4IoT can improve the efficiency of people's everyday living areas. BPMS enables autonomous actions to be triggered based on certain events that are measured based on the deployed front-end IoT devices. For example, the HVAC systems used in the modern buildings can be monitored and controlled from the remote BPM-based information system precisely via Internet-connected wireless sensor and actuator network (WSAN) devices. A simple smart home system that integrates with the mobile service can identify house residents' movements (via their mobile/wearable devices) in order to maintain the oxygen and temperature level when the residents are coming back home from work. The indoor monitoring devices also can track the home environment and remotely inform the residents of any event that occurred when they are away from home. For large residential buildings, such as the hotel HVAC control system, the BPMS4IoT enhances the efficiency of the management because the system can automatically measure the usage of electricity and the heating system and bill the customer based on the usage instead of charging the fixed rate for each room [Tranquillini et al. 2012].

Healthcare and Ambient Assisted Living (AAL). The BPMS4IoT-based healthcare system integrates cloud services with Wireless Body Area Network (WBAN) [Quwaider and Jararweh 2016], which is formed by numerous wearable sensor devices that can measure blood sugar, body temperature, heartbeat, and so forth. We take the use case described in Dar et al. [2011] as an example.

Imagine a 70-year-old woman, Emily, who is using the remote Eldercare system with WBAN attached to her body to measure her blood sugar level. One day, she feels a small headache and dizziness. The sensor has detected that Emily's blood sugar has violated the predefined threshold. Hence, the report is immediately sent to the hospital's system to inform the physician. The remote physician then performs the remote health monitoring process via the Internet-connected body sensor network attached to Emily. Meanwhile, the system also informs Emily's son via SMS about Emily's health condition. Afterward, Emily receives a prescription and dietary recommendations from the physician. A while later, Emily's son visits her and assists Emily to recover from her health condition back to normal [Dar et al. 2011].

1.2. Problem Statement

Although many IoT application domains have applied BPMSs and they have shown the promising solutions, many of them have not considered the challenges that the BPMS will face in the near-future IoT environments. We summarize the challenges here:

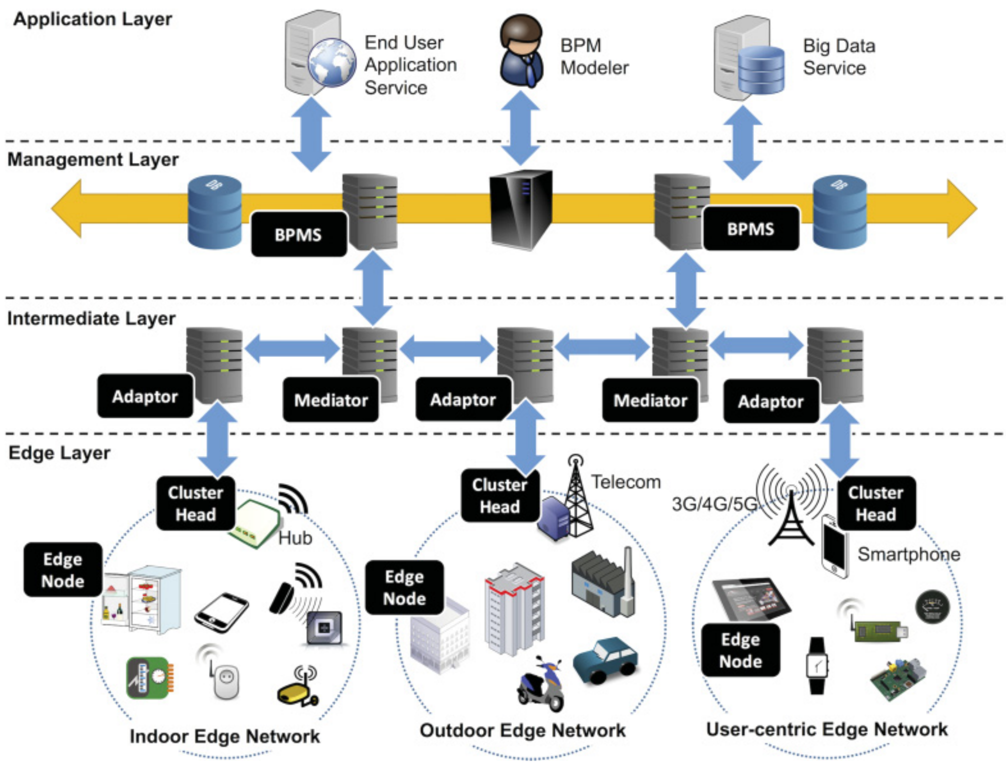


Fig. 1. Simplified common SOA-based BPMS4IoT.

Figure 1 illustrates a common SOA-based BPMS architecture that involves IoT entities and the middleware-based information system. Most BPMS4IoT frameworks are following the design in the figure. Such a design faces the following limitations.

—*Transparency*. As the figure shows, the system has no direct interaction with the edge network (i.e., front-end wireless Internet-connected nodes such as mobile phones, wireless sensors, data collectors, etc.). The design that hides the detail topology and communication at the edge network may cause the process designer to be unable to properly define the behaviors and processes of the devices in terms of adding participants, modifying or customizing the processes, reacting to or preventing failures, and so on. Further, recent IoT management systems often involve edge nodes as a part of the BPMS in the scenarios such as logistics [Glombitza et al. 2011], crowdsourcing processes [Tranquillini et al. 2015], HVAC systems [Tranquillini et al. 2012], healthcare services [Peng et al. 2014], Ambient Assisted Living [Dar et al. 2015], and real-time sensing [Chang et al. 2015b] where mobile and wireless network objects are involved.

—*Agility*. SOA-based IT systems often use interface and middleware technologies to leverage different entities. However, due to the lack of standardization in process modeling and execution levels, systems usually resulted in a complex and inflexible design. For example, based on the architecture described in Figure 1, if two devices that are connected to different back-end servers but located in physical proximity intend to interact with each other, the communication needs to pass through multiple layers, in which the direct interaction does not exist and it also affects the overall performance. As the vision [Conti et al. 2012] indicates, in the future, IoT environments and devices will cooperate automatically for certain tasks in order to improve the efficiency and agility.

1.3. Aim of the Survey

In this article, we review the existing BPMSs for IoT (BPMS4IoT) frameworks to identify whether or not they have addressed the challenges in BPMS4IoT and how they have overcome the challenges from the perspective of Mobile Cloud Computing (MCC).

MCC represents the integration of mobile computing and cloud computing in which it addresses the elastic cloud resource provisioning and the optimized interaction between cloud services and mobile network nodes. Commonly, disciplines in MCC addressed many studies in mobile connectivity, mobility, discovery, resource awareness, decentralized service interaction, and how the system efficiently integrates the mobile entities as process participants with cloud services.

In the past several years, researchers have proposed numerous MCC-related WfMSs, including the mobile device-hosted WfMS engines [Hackmann et al. 2006; Pryss et al. 2011; Sen et al. 2008; Chou et al. 2009], WfMSs for enabling mobile ad hoc cloud computing [Chang et al. 2014a], and WfMSs for IoT that are exploiting both mobile and utility cloud resources [Chang et al. 2012a, 2014b, 2015b]. Therefore, MCC has shown the promising solution for the efficient integration of information systems with various wireless Internet-connected entities for distributed process management. Furthermore, the deployments of existing IoT management systems are widely utilizing wireless sensor networks and mobile and cloud services, which indicates that MCC is the key enabler of BPMS4IoT. Therefore, in this article, we intend to address the issues of BPMS4IoT by applying MCC concepts.

The rest of the article is organized as follows: Section 2 provides a brief review of related literature surveys. Section 3 provides the state-of-the-art literature review on BPMS4IoT frameworks based on the life cycle phases of BPMS4IoT. Section 4 compares the existing frameworks. In Section 5, we identify the open challenges and issues that have not yet been fully addressed in existing works. The article concludes in Section 6.

2. RELATED WORK

There exist a large number of related literature surveys in IoT and BPMS. In this section, we summarize the featured works with categorization next.

—*Comprehensive*. Numerous surveys [Gubbi et al. 2013; Atzori et al. 2010; Vermesan et al. 2011; Miorandi et al. 2012; Borgia 2014; Guo et al. 2013; Da Xu et al. 2014] provide the comprehensive review of existing IoT and related works. They are focused on discussing the background of IoT, emerging technologies, promising applications in various domains, and open challenges.

—*Service discovery*. The term *service* in the IoT domain represents the function provided by the CPS that is interconnected with the physical entities. Considering that a large number of heterogeneous physical entities will be connected to the Internet in the near future, service discovery becomes extremely challenging, especially in the big data service environment in which the data is retrieved from various spatiotemporal service providers. Therefore, a number of literature survey papers [Evdokimov et al. 2010; Villaverde et al. 2014] are focused on the service discovery in IoT.

—*Network technology*. Since the vision of IoT been introduced, researchers have acknowledged the importance of feasible network communication protocols for resource-constrained devices, which are the core front-end elements of IoT. Numerous literature surveys [Mainetti et al. 2011; Palattella et al. 2013; Sheng et al. 2013;

Tonneau et al. 2015] are focused on reviewing the feasibility of the existing and emerging network protocols for IoT.

—*Middleware*. Middleware technology is the enabler to integrate the front-end physical entities with the back-end software systems. Numerous literature surveys are focused on reviewing and comparing existing middleware technologies for IoT. These include an overview of the existing CPS middleware framework [Bandyopadhyay et al. 2011a, 2011b; Chaqfeh and Mohamed 2012], a review on web-service-oriented IoT middleware [Zeng et al. 2011; Issarny et al. 2011], cloud computing and IoT integration [Botta et al. 2016], and a study on how to select proper protocols for the connected devices [Mashal et al. 2015].

—*Domain specific*. There also exist a number of IoT literature surveys that focus on specific research areas such as healthcare and disabilities [Domingo 2012; Islam et al. 2015], urban computing [Zanella et al. 2014; Conti et al. 2012; Salim and Haque 2015], multimedia [Alvi et al. 2015], data mining and big data [Aggarwal et al. 2013; Tsai et al. 2014], social network aspects [Atzori et al. 2012], energy efficiency [Villaverde et al. 2012; Aziz et al. 2013], mobility [Zorzi et al. 2010; Silva et al. 2014; Bouaziz and Rachedi 2014], trust management [Yan et al. 2014], and security [Roman et al. 2013; Sicari et al. 2015; Nguyen et al. 2015; Granjal et al. 2015]. All of them have provided a detailed study on the specific domain when IoT is applied.

—*Business Process Management*. Numerous of BPM-based survey papers have been published in recent years. These papers include but are not limited to the comprehensive survey [van der Aalst 2013] that provides an overview of the process modeling methods, techniques, and tools; business process modeling standards [Ko et al. 2009]; elastic cloud-system-driven BPM [Schulte et al. 2015]; and business process model frameworks [Yan et al. 2012].

Although there exist a large number of literature survey papers in the domain of IoT and BPMS today, to the best of our knowledge, this is the first literature survey that focuses on BPMS4IoT in the MCC perspective. BPMS4IoT faces its specific challenges that have not yet been addressed in the past IoT survey papers.

3. INTEGRATING BPMS WITH IOT

In general, we can consider three phases in the life cycle of BPMS: *(re)design phase*, *implement/configure phase,* and *run and adjust phase* [van der Aalst 2013]. Depending on the focus, some disciplines can further classify each phase into more detailed phases. For example, the authors in their earlier works [van der Aalst et al. 2003; Weske et al. 2004] have separated the *(re)design phase* into the *diagnosis phase* and *process design phase*. The *(re)design phase* can also involve three different phases, including *process discovery*, *process analysis,* and *process redesign* [Dumas et al. 2013]. Although there can be other notions of defining the life cycle, in this article, we apply the recent definition of the BPM life cycle from van der Aalst et al. [2013] to BPMS4IoT.

Figure 2 illustrates the life cycle of BPMS4IoT. The *(re)design phase* involves how to model the connected IoT elements and their related elements, and how their behavior is in the business process. The *implement/configure phase* involves how to practically implement the process model as executable methods. Further, it involves how to deploy the executable methods to the corresponding workflow engines for execution. The *run and adjust phase* corresponds to how the BPMS autonomously monitors and manages the system at runtime, and how it continuously improves and optimizes the process. If the process designer recognizes the need to improve the processes, they will redesign the system and continue with the cycle.

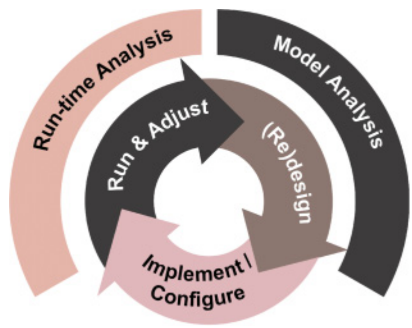
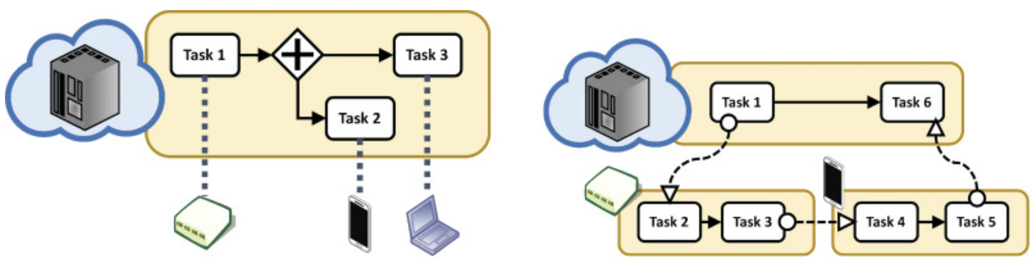


Fig. 2. BPMS4IoT life cycle.



(a) Centralised orchestration (b) Decentralised choreography

Fig. 3. Business process execution: orchestration versus choreography.

Aside from the three main life cycle phases, during the *(re)design phase*, the system developers will perform technologies to analyze the model design (e.g., using simulation). The management team also collects the data (e.g., event logs) in the *run and adjust phase* for process diagnosis.

The following subsections provide a state-of-the-art review and analysis based on each phase of the life cycle.

3.1. (Re)design Phase

The *(re)design* phase of BPMS4IoT involves the following:

(1) *Architecture*—represents the fundamental design of the system, which can be based on the centralized orchestration model, the decentralized choreography model, or the hybrid model that inherits the features from both.
(2) *Modeling*—involves how to model the business processes. Will the process model use only existing methods (e.g., notations of standard BPMN) to design the IoT entities and their activities or will it introduce new elements for best describing the model? Further, what entities need to be considered?
(3) *Transparency*—indicates that the process modeling should provide a comprehensive view of the overall execution environment.

To follow, we discuss the subjects that have been addressed in the existing BPMS4IoT frameworks for the *(re)design phase*.

3.1.1. Orchestration Versus Choreography. BPMSs can be broadly classified into two types: orchestration and choreography (see Figure 3). The orchestration is mainly based on a centralized architecture in which a single management system is managing the entire process execution. On the other hand, choreography represents a system in which in certain stages, a number of external information systems are handling the processes (or the portions of the process). While BPMS4IoT is commonly designed based on the orchestration model, recent works have emphasized the importance of choreography in IoT. For example, Dar et al. [2011, 2015] considered that the centralized orchestration is insufficient to agilely react to events occurring on the edge network, especially in an interorganizational network or in the mobile IoT scenarios, which involve numerous mobile Internet-connected participants.

In such scenarios, there is a need to apply choreography-based architecture to enable a certain degree of business process distribution, in which the edge nodes will need the Business Process (BP) model execution mechanisms and self-management abilities. Moreover, distributing process execution at edge nodes can further enhance the flexibility, agility, and adaptability of the BPMS4IoT [Dar et al. 2015; Peng et al. 2014; Tranquillini et al. 2012].

3.1.2. Existing Versus Extension. Introducing the IoT elements in BPMS is not a straightforward task because IoT devices can be heterogeneous in terms of communication protocols, network topologies (e.g., connectivity), and hardware specifications (e.g., computing power and battery life). In general, there are two approaches to model the entities of IoT:

(1) *Expressing IoT devices as services*. Modeling the IoT devices as URI-based services can simplify the management systems and also be fully compatible with existing tools such as BPMN or BPEL. In this approach, the system expresses IoT devices as regular network services such as XML-based W3C/SOAP web services [Peng et al. 2014], OASIS.Devices Profile for Web Services (DPWS), or OGC Web Service Common, which is communicable via the regular request-response methods. Generally, in this approach, BP model designers assumed that the system can connect to IoT devices directly based on the web service communication. However, in real-world systems, many IoT devices do not work as regular web service entities. Therefore, numerous researchers are focusing on defining new elements for BPMS4IoT.

(2) *Defining new IoT elements*. In general, BPMSs such as ERP systems often assumed that the system will provide automation for all the involved devices, and the system will have the capability to directly invoke all the devices. However, in IoT systems, such an assumption, in many cases, is not applicable [Meyer et al. 2015]. IoT entities that have different capabilities may connect with the management system differently.

    Besides the regular direct IP network connected devices, in many scenarios, the IoT devices connect to the management system via multiple network layers or routings. For example, an actuator device may connect with an intermediary service provided by different devices in order to let the management system access it.

    Another example is when the BPMS involves continuous tasks such as sensor data streaming and Eventing, in which case existing standard-based BPM modeling tools cannot explicitly define the process [Appel et al. 2014]. In such cases, the system needs a more proper way to let designers model the process accordingly in order to fully manage and optimize the systems. Thus, a common approach is to introduce specific IoT elements in BPM to differentiate them from the traditional BPM elements such as service tasks in BPMN.

3.1.3. Modelling IoT Elements. As mentioned previously, modeling IoT elements in BPM is not a straightforward task. Although designers can extend the business process modelling languages such as standard BPMN 2.0 and the corresponding tools to introduce the IoT elements by either expressing them as components or simply describing them as Uniform Resource Identifier (URI)-based services, overall, existing BPM solutions

Table I. BPMN Extension for Introducing IoT

Project	IoT System Processes	IoT Device Activity	Physical Entity	Sensor	Actuator	Intermediary Operation	Event Stream	Special Element
IoT-A	PO(↓)	LN(↓)	PA(↑)	TA(∼)	TA(∼)			
Sungur et al.	PO(∼)			ST(∼)	ST(∼)	ST(↑)		
uBPMN				TA/EV(↑)	TA/EV(↑)			DO(↑)
SPUs							DO/ ST(↑)	
makeSense	PO(↑)			TA(∼)				

Subject addressed level: (↑) = high; (∼) = medium; (↓) = low;

PO	BPMN Pool	PA	Participant	TA	BPMN Task	DO	Data Object
LN	BPMN Lane	EV	BPMN Event	ST	BPMN Service Task		

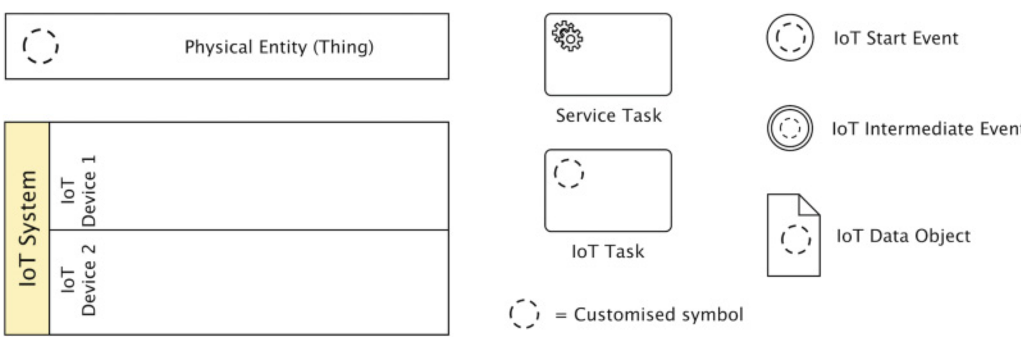


Fig. 4. Common notations used in IoT-driven BPMN.

have not included IoT specifically [Meyer et al. 2015]. Consequently, numerous related works tended to introduce new notations that represent IoT elements in the business process models.

Table I summarizes the IoT elements modeled in the existing BPMS4IoT frameworks. As the table shows, almost all of the modeling frameworks have introduced a sensor element, followed by the actuator and the IoT system processes. We summarize each element next.

IoT System Process and IoT Device Activity. IoT System Process, or *IoT Process*, represents the workflow that involves IoT-related events and activities. In BPMN-based approaches such as IoT-A [Meyer et al. 2013; Sungar et al. 2013] and makeSense [Tranquillini et al. 2012], they notate IoT Process as the Pool (see IoT System in Figure 4). Specifically, the IoT-A project [Meyer et al. 2013] has further modeled IoT device activities in Lane (e.g., IoT Device 1 and 2 in Figure 4) to separate them from the general processes that exclude the detailed IoT tasks. In contrast, makeSense proposed a different design, which specifically separated the IoT-related processes entirely to different Pools. For example, Figure 5 shows an example in which the model separates the IoT-related processes from the Central System.

Physical Entity. Meyer et al. [2015] specifically present a thorough analysis about how BPMN should signify the real-world physical entities such as a chocolate, a bottle of milk, and an animal. Consider that the system may interact with physical entities

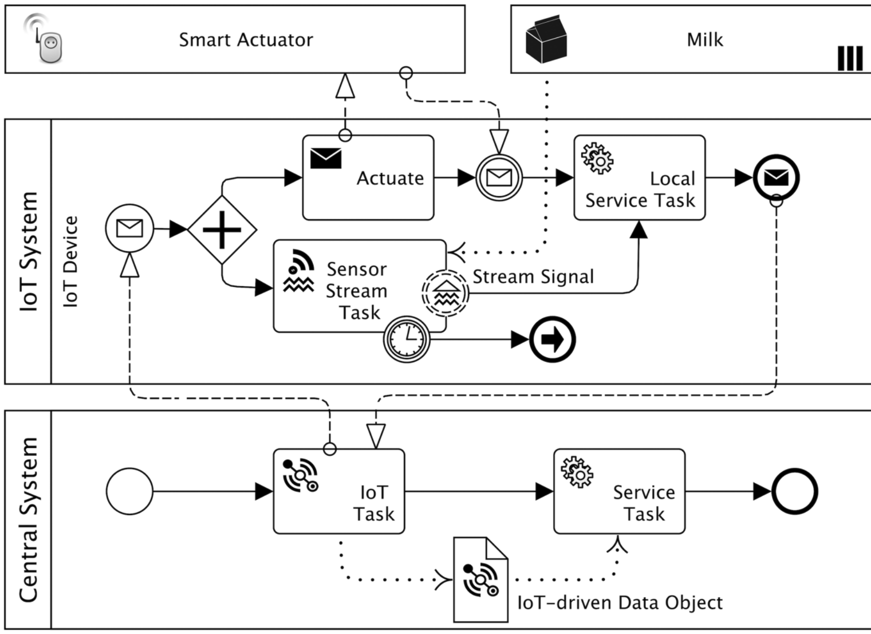


Fig. 5. IoT-driven business process workflow.

with heterogeneous protocols; Text Annotation and Data Object are both less feasible to represent the physical entities. Therefore, their discussion result shows that utilizing Participant notation in BPMN is prevailing for notating physical entities (e.g., Physical Entity (Thing) in Figure 4).

Actuator. An actuator is a controller-type device that can perform a certain command to physical objects. For example, an IoT system can remotely switch on or off a light via an interconnected light switch actuator. Commonly, existing frameworks utilize Task or Service Task to notate the actuator-involved activities in BPMN. However, in the work of Yousfi et al. [2015, 2016], they further introduced the specific *start event* and the *intermediate event* BPMN notations for clarifying what kind of actuator devices are used in the events.

In general, the approach to introduce IoT-related tasks and events in BPMN is to replace the symbols of notation (e.g., the dashed-line circle in Figure 4). Note that in this article, we classify the camera, microphone, and tag readers specified in the work of Yousfi et al. [2015, 2016] as actuator based on the description by Guinard et al. [2010].

Sensor. IoT systems utilize sensors to acquire specific data such as brightness, temperature, an entity's movement, moving direction, and so on. Overall, existing frameworks tend to model sensors differently. The activities of sensors are usually designed as the extension of Task [Meyer et al. 2013; Yousfi et al. 2015] or the extension of Service Task [Sungur et al. 2013] with a specific symbol. Additionally, Yousfi et al. [2015] further introduced Sensor Events for the same purpose as they applied to the actuators described in the previous paragraph.

Event Streaming is a less studied but important mechanism in IoT scenarios. Although existing BPMSs support single events, they lack feasible integration across the process modeling, process execution, and IT infrastructure layer. Hence, Appel et al.

[2014] introduced a specific element—Stream Process Unit (SPU)—to integrate the continuous event streaming processes.

The primary differences between the SPU task and the existing BPMN elements such as Service Task, Loop, or parallel operation are (1) SPU is based on advertisement/subscription of the event streaming process that operates in isolation from the main process workflow, and (2) SPU performs the process continuously to fulfill a single process, which is different from the Loop that is repeating certain processes (e.g., sensor stream task and stream signal in Figure 5).

Intermediary Operation. It represents a process that is responsible for performing advanced activities based on the sensory information. Most of the existing works [Dar et al. 2015; Yousfi et al. 2015; Meyer et al. 2015] consider the sensory activities as the processes to collect the raw context data. However, Sungur [2013] considers that the system should process the raw context data (e.g., interpreted to meaningful information for the user) before sending it to the requester. Such a requirement is less considered in the other works because it is commonly expected that the application will process the raw context data based on the need of the requester. Besides, the requesters may have their own algorithm to process the data. However, in the future interorganizational IoT environment, the data collector may also be interested in providing an additional context interpreting service to their data consumers.

Specific Data Object. Data Object in classic BPMN represents data or a file that is transmitted from one activity to another. Commonly, it is a one-time transmission.

In uBPMN [Yousfi et al. 2015], the authors introduced the *Smart Object* element, which is the subclass of *data object* in BPMN. The *Smart Object* element consists of the type attribute to describe the source of the data (e.g., the data was collected by Sensor Task or Reader Task).

In SPUs [Appel et al. 2013, 2014], the authors introduced the *Event Stream* data object, which is different from the classic data object in BPMNs that represents one single flow of data transmission. The *Event Stream* data object represents the independent streaming data that is being continuously inputted or outputted via the workflow system. Such an approach can specify the sensory information streaming scenario in IoT, which is not considered in the classic BP model design.

Discussion

Most frameworks introduced IoT elements to adapt to specific scenarios. The most conflicted element defined among the literature is the sensor element. Some works prefer to model sensory devices as individual information systems, but some works prefer to hide the details and only consider the corresponding sensor tasks. From the perspective of the real-world IoT implementations (e.g., Chang et al. [2015]), sensor devices are connected to an IP network (either via mediator, gateway, IPv6, or 6LoWPAN [Shelby and Bormann 2011]) and ideally can be accessed as a URI-based RESTful service. Since the standard such as BPMN 2.0 supports URI-based Service Task interaction, it is not clear that differentiating the Sensor Task from the Service Task will bring much effort to the process modeling. On the other hand, a process such as event streaming [Appel et al. 2014], which has not yet been considered in classic BP modeling, is necessary to be addressed as a new element.

Commonly, existing modeling approaches have not considered mobile IoT devices in their model specification. Mobile IoT devices (e.g., wearable devices, flying devices, handheld devices, etc.) have dynamic states, including their location, moving direction, connectivity, and so on. These states highly influenced the process operation. For transparency and agility purposes, this context needs to be considered in the model in order to quickly react to the events. If the assumption is to rely on the cloud middleware to handle the events as separated from the main management system, it loses the transparency purpose.

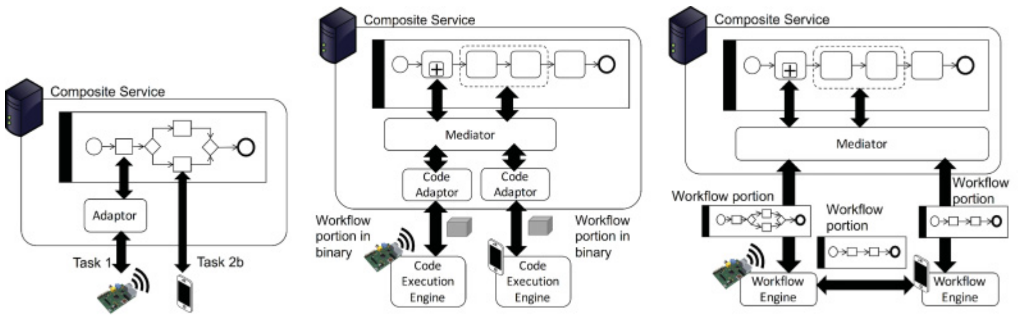
3.2. Implement/Configure Phase

The *implement/configure phase* represents how the system transforms the abstract BP model to the machine-readable and machine-executable software program. The challenges in this phase mainly involve the lack of corresponding tools. Commonly, BP modelers design the BP models in graphical tools, and the tools may generate the machine-readable metadata in order to let the workflow engine execute the processes. However, common tools such as Activiti (http://activiti.org), Camunda (https://camunda.com), BonitaBPM (http://www.bonitasoft.com), and Apache ODE (http://ode.apache.org) do not support many of the protocols used in IoT devices (e.g., CoAP and MQTT). Currently, there is no corresponding tool that can address all the protocols used by the IoT devices. A common approach is to introduce a middleware layer to leverage the embedded service from IoT devices with the common SOAP or REST web services. However, without the proper solution, the system may sacrifice transparency (in BPM perspective), performance (extra overhead), and agility (reaction of runtime events).

3.2.1. Machine-Executable BP Model Approaches. Currently, from the literature studied in this article, BPEL, BPMN, and XPDL are three common standard technologies used in implementation.

BPEL for IoT. Web Services Business Process Execution Language (WS-BPEL, or BPEL in short) is an executable and interoperable service composition language in which BPEL is fully WS* compliant. Ideally, the BPEL metadata is executable in any standard compliant engines as long as the engines have fetched the required corresponding descriptions (WSDL, XML schema, etc.). Notably, both academic research projects [Loke 2003; Glombitza et al. 2011; Yang et al. 2012; Wu et al. 2012; Chang et al. 2015b] and industrial solutions such as the WSO2 IoT Server [WSO2 2016], Site-Where (http://www.sitewhere.org/) and Oracle SOA Suite [Oracle 2016] have introduced BPEL-based BPMS4IoT. However, the standard itself does not natively support RESTful service invocation, which is now the primary approach to providing services from IoT devices. According to the W3C standard, WSDL 2.0 can describe HTTP method-based invocation. Unfortunately, WS-BPEL does not support WSDL 2.0. In other words, the current BPEL standard is mainly for WSDL 1.0 and SOAP-based service composition only. Although numerous commercial BPEL engines have support for RESTful HTTP service, it is not sufficient for the other common protocols used in IoT devices. Moreover, BPEL supports orchestration only. It does not support the choreography natively.

BPMN for IoT. BPMN has been the most popular approach in the literature studied in this article. These works include IoT-A [Meyer et al. 2013], VITAL [VITAL-IoT 2016; Canracaş et al. 2011], BPMN4WSN [Sungur et al. 2013], MOPAL [Peng et al. 2014], SPUs [Appel et al. 2014], makeSense [Casati et al. 2012; Tranquillini et al. 2012, 2015], and uBPMN [Yousfi et al. 2015; Dar et al. 2015]. Although originally BPMN was only a graphical BP model standard introduced by the Object Management Group (OMG), OMG also introduced the metadata form in XML format. BPMN provides flexibility for designers to introduce their own extension of BPMN elements. Hence, it is possible to introduce new IoT entities and elements semantically using BPMN, XML schema, and semantic description methods. Although the extension grants flexibility, there is no interoperability between different design models because they are tightly bounded onto the corresponding execution engine and hence result



(a) Device Participating in BP (b) Devices execute code-converted BP workflow (c) Devices execute BP workflow

Fig. 6. Different approaches for integrating physical things in BPMS.

in isolated solutions. In other words, the BPMS designed based on makeSense cannot cooperate with uBPMN-based BPMS.

XPDL for IoT. XPDL (http://www.xpdl.org/) is a standard introduced by the Workflow Management Coalition (http://www.wfmc.org/) for interchanging the graphical business process workflow models to XML-based meta-models. Numerous projects have used XPDL to enable workflow execution on mobile devices [Chou et al. 2009; Chen and Shih 2011] or other CPS devices [Kefalakis et al. 2011]. Ideally, XPDL is a standard continuously updated to be compatible with the latest BPMN standard. Although its primary purpose is to serve as an XML interchange of BPMN, developers can also use it as the metamodel of the other BP workflow definitions. Accordingly, the standard's document indicates that XPDL is a standard for interchanging any type of workflow model to machine-readable code. Hence, compared to BPEL, XPDL can be more flexible in terms of implementation and execution.

Discussion

Overall, existing BP model standards natively do not support the need for the distributed process in the near-future IoT systems. Among them all, BPMN and XPDL provide the flexibility of the extension; hence, they are applicable when interoperability is not the concern of the system. Although some other BP modeling approaches exist (e.g., Petri-Net model-based system [Kaneshiro et al. 2014; Thacker et al. 2010]), there is no corresponding linkage between the model and the executable program.

3.2.2. Process Execution Approaches. The *process execution* represents how the workflow engines execute the transformed machine-readable process model. In general, the workflow engines are hosted either on the central management server or on the participating IoT devices in the edge network.

Considering the classical centralized system, where the system does not require performing complex processes on participants, the workflow engines are only hosted in the central system, in which the IoT devices can operate as the regular request/response operation-based services. Conversely, if the system requires process distribution, the IoT devices may need to embed process execution engines. Overall, there are three types of situations in BPMS4IoT. We described them next.

Participating in BP. In this case, the participative node only performs certain tasks based on request/response or publish/subscribe mechanism (see Figure 6(a)). For example, in the nursing home scenario described in Pryss et al. [2015], the hospital's BPMS can remotely assign tasks to the front-end nurses via their mobile devices. Moreover, since the mobile devices act as the mediators, they can either allocate manual tasks to the nurses or retrieve the data from the patients to the remote hospital BPMS.

Practically, the front-end devices used in this approach can simply enable socket channels to maintain the communication between themselves and the central system. Alternatively, the front-end devices can embed web services as the service providers [Srirama et al. 2006; Liyanage et al. 2015] (e.g., Task 2b in Figure 6(a)), which provide services directly without maintaining a long-period communication channel with the distant central system.

Executing BP Model-Compiled Code. In this approach, the system translates the BP model metadata generated from the BP model editor to the executable binary code or to a specific programming language (see Figure 6(b)). In particular, this approach may be more suitable for low-level resource-constrained devices to execute the BP models. Generally, in this approach, the system relies on the middleware technologies to translate the BP model to the executable code, then send the code to the IoT devices for execution. Particularly, numerous BPMS4IoT frameworks [Glombitza et al. 2011; Caracaş and Kramp 2011; Casati et al. 2012; Tranquillini et al. 2012, 2015] have utilized this approach to enable IoT/WSN devices participating in BP execution without the need of embedding complex software middleware components (e.g., workflow engine) on the devices.

Executing Standard BP Model. Directly porting or implementing a workflow engine on the IoT device enables the best flexibility (see Figure 6(c)). Moreover, it also enables a flexible way of performing process choreography between the back-end cloud service and the front-end IoT devices or between numerous front-end IoT devices, in which the front-end IoT devices situated in the edge network are executing the workflow [Pryss et al. 2015]. However, such a mechanism requires higher computational power devices (e.g., high-end smartphones). In the meantime, numerous research projects have introduced the standard-based workflow execution engines for mobile operating systems.

Table II lists a number of featured workflow engines designed for mobile devices. Further, the table also describes certain additional elements supported by the workflow engines.

- *Standard*—describes which workflow modeling or description standard the engine supports. Overall, only Presto [Giner et al. 2010] and Dar et al. [2015] support BPMN. Fundamentally, BPMN is a graphical tool, and it requires an extra mechanism to convert the graphical BP model to machine-readable code. Hence, it is understandable that most engines have chosen to support the XML-based modeling standard (i.e., BPEL and XPDL).
- *Workflow distribution*—illustrates how the engine handles the workflow migration between different entities. Specifically, handling workflow migration between different entities (e.g., between cloud and mobile or between mobile and mobile) is an important mechanism to support the machine-to-machine communication in the near-future IoT applications [Dar et al. 2015]. Hence, it is foreseeable that such a feature may become a requirement in future BPMS4IoT.
- *Protocol*—describes the supported protocols of the engine. Generally, earlier engines [Hackmann et al. 2006; Sen et al. 2008; Pajunen and Chande 2007] aim to be fully compliant with the web service standard. Hence, SOAP was their primary consideration. However, since the wireless IoT devices are usually resource constrained, recent approaches intend to apply lightweight protocols. Although the workflow description standards (e.g., BPEL and BPMN) natively do not include the definition of lightweight protocol-based service invocation such as HTTP-based RESTful web

Table II. Comparison of Workflow Engines for Mobile Devices

	Standard	Workflow Distribution	Protocol	Awareness	Platform
Sliver [Hackmann et al. 2006]	BPEL	—	SOAP	—	Java ME
CiAN [Sen et al. 2008]	BPEL	Device-to-Device	SOAP	—	Java ME
Pajunen et al. [Pajunen and Chande 2007]	BPEL	—	SOAP	—	Java ME
AMSNP [Chang et al. 2012a]	BPEL	Cloud and device	RESTful HTTP	Resource aware	iOS
SPiCa [Chang et al. 2014a]	BPEL	Device-to-Device	RESTful HTTP	Resource aware	iOS
SCORPII [Chang et al. 2015b]	BPEL	Cloud and device	RESTful HTTP	Resource aware	Android
MAPPLE [Pryss et al. 2011]	Cutomised	Cloud and device	Stand-alone	Resource aware	.NET
EMWF [Chou et al. 2009]	XPDL	—	—	—	Windows CE
ERWF [Chen and Shih 2011]	XPDL	—	—	Environment aware	SISARL
Dar et al. [Dar et al. 2015]	BPMN	Cloud and device	CoAP/ MQTT/ RESTful HTTP	—	Android

services, Constrained Application Protocol (CoAP), or even the customized lightweight protocols [Pryss et al. 2011; Giner et al. 2010], many engines have supported the need.

- *Awareness*—denotes whether the engine supports a certain awareness mechanism to handle the runtime events or not. Overall, a number of existing engines have supported *resource awareness*, which represents the strategy to improve the reliability of the software or hardware resource used in executing the workflow tasks. Specifically, ERWF [Chen and Shih 2011] supports *environmental awareness*, in which the workflow engine can react to the environmental changes at runtime to maintain the performance.
- *Platform*—describes which platform or operating system the engine supports. As the table shows, most projects developed their engines based on smartphone OSs, except ERWF [Chen and Shih 2011], which is targeted on the SISARL sensor devices (http://www.sisarl.org/) that have lower power than regular smartphones.

Discussion

The drawback of the *executing BP model-compiled code* approach is that every time the process model changes, the system needs to again convert the model to the executable source code and to deploy the code to the IoT devices. Conversely, the embedded workflow engine approach can perform rapid changes in the model and execution. However, it may also consume more hardware resources of the device.

The engines described in Table II bring the possibility of distributed business process workflow execution between *cloud-to-mobile* and *mobile-to-mobile*, in which the system can support choreography-based composition, toward bringing the highly flexible and scalable mobile cloud-based BPMS, which can be useful for many IoT systems such as the logistics and the AAL use cases described previously in Section 1.1.

3.3. Run and Adjust Phase

The *run-and-adjust phase* represents the system execution runtime after the deployment of the BPMS. In general, this phase does not involve any redesign or new implementation. Specifically, since the run-and-adjust phase only performs the predefined management activities [van der Aalst 2013], the BPMS should provide adaptive mechanisms for the system administration. For example, the system should record the runtime execution history, handle the events, and allow the adjustments.

Following are the major mechanisms the BPMS4IoT should address in the *run-and-adjust phase*.

3.3.1. Monitoring and Control. BPMS4IoT involves various front-end devices forming edge networks in which groups of static or mobile devices are participating (e.g., collaborative sensing). Commonly, in a design such as a cloud-service-based IoT environment [Botta et al. 2016], the system requires multiple layers to enable the activity monitoring of the edge networks. In other words, the IoT devices need to connect to the broker, sink, or super-peer (the head of an edge network) devices in order to enable the communication between themselves and the back-end systems using publish-subscribe protocols (e.g., MQTT).

Monitoring in *run-and-adjust phase* involves two types: *process monitoring* [van der Aalst 2013] and *device status monitoring* [Dar et al. 2015].

—*Process monitoring* is the comprehensive system monitoring, which involves how the system operates the IoT activities and how it handles the runtime events. At this stage, the system collects the *process monitoring* record log files for further analysis in order to improve the system in the *redesign phase*.
—*Device status monitoring*. Since the activities of BPMS4IoT involve a large number of wireless network devices, the runtime device failure can sometimes cause the entire system to fail. Hence, *device status monitoring* also plays an important role in BPMS4IoT.

Monitoring the edge network is a crucial task because it can involve many factors such as the hardware damage occurring in the edge network device, the cluster head of the edge network losing its connectivity with the other edge peers, or the data broker node losing its connectivity to the back-end system due to either hardware or the Internet provider failure. Overall, most existing BPMS4IoT frameworks have not broadly studied runtime device failure issues. Among the existing frameworks, Dar et al. [2015] applied a generic approach to handle the runtime device failure. In their approach, the system assigns a periodical report task to each front-end device. If the central server does not receive the report from a particular device exceeding the timeout threshold, the system will consider that the device is failed. Hence, the system will perform the substitution.

3.3.2. Fault Tolerant. The IoT paradigm involves a large number of wireless-network-connected devices. Generally, these devices have less stability in connection and with the low battery capability. Therefore, a proper system needs to consider the resource-constrained devices and to support the corresponding solutions, such as reactively replacing the failed devices/activities with substitutions immediately and autonomously in order to retain the processes [Dar et al. 2015]. Further, in order to optimize the system to react or even prevent the failure, the system needs to distribute and govern a certain business process to the edge network. For example, considering the unreliable mobile Internet connection, Peng et al. [2014] proposed a framework for distributing and executing tasks at the edge-network mobile nodes in offline mode. Explicitly, with such a mechanism, when an edge network lost its Internet connection with the distant management system, the devices in the edge network can still continue the processes. Furthermore, they can send the process output and the monitored records to the distant management system once they are back in connection.

3.3.3. Context Awareness. *Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves* [Dey 2001]. BPMS4IoT needs to address context awareness in terms of contingencies, personalization, and efficiency [Sheng et al. 2014].

Contingencies involve the unpredictable connectivity and accessibility of pervasive services and wireless network devices. Generally, there are two schemes to address contingencies in BPMS4IoT: *proactive* and *reactive*.

—*Proactive* scheme aims to prevent the occurrence of problems from IoT devices at runtime. For example, in the MOPAL project [Peng et al. 2014], the authors have defined certain context-aware rules (e.g., current CPU capability, battery level, geographical location, etc.) that constrain the workflow task execution on the IoT devices.
—*Reactive* scheme commonly seeks for substitution for the workflow task execution. In the SOA-based BPMS for home automation system proposed by Chang and Ling [2008], the connected devices belong to specific categories depending on their associated context. For example, both workflow tasks "sound alarm" and "switch on TV→raise TV volume" can generate the same type context—"loud noise" to wake up the user. Hence, when the default setting "sound alarm" is failing for any reason, the system can trigger the substitution, which is "switch on TV→raise TV volume" to achieve the same purpose.

Personalization involves service provisioning based on the requesters' preferences. For example, in the smart ubiquitous computing domain [Peng et al. 2014; Dar et al. 2015], context awareness has usually considered the entity's context. Ordinarily, the entity in most cases is the human user him- or herself, and the context can be the person's heartbeat rate, blood, breath, physical movement, and so forth. Comparatively, works in the AAL domain [Loke 2003; Yousfi et al. 2015; Chang et al. 2015b] may consider environmental context, such as the temperature, noise level, and density of the crowd, in which case the central entity (the user) does not have direct control of it.

Efficiency involves energy efficiency, the cost of deployment, and the cost of communication.

—*Energy efficiency* is one of the major concerns in BPMS4IoT. In order to conserve the energy of IoT devices, Caracaş [2011, 2012] specified the *time* context factor in the BP model. Basically, in their BP model, each IoT task should associate with Timers for controlling the sleep/wakeup time of the device. Alternatively, an interorganizational collaborative data brokering strategy is also a promising approach. For example, in the work of Chang et al. [2015], the IoT devices, which belong to different organizations, are collaboratively brokering the data to their distant back-end server. In general, the data collector device will automatically seek for collaboration when its battery is getting low. Ideally, such an approach can reduce the unnecessary energy consumption compared to sending data individually.
—*Deployment*. Deploying IoT devices individually can be costly. One possible strategy is to reuse the already-deployed system and provide an integration platform such as SOA-based cloud service that can enable the need and also provide the interoperability between the BPMS of different organizations. A BPMS4IoT can compare the cost of deploying its own devices and the cost of utilizing the third party's resources. In

contrast, the second option can result in better efficiency in deployment. For example, in the GaaS project [Wu et al. 2012], the cloud-based gateway platform enables IoT systems from different parties to work together. Similarly, in the Adventure project [Schulte et al. 2014], the system utilizes the virtual-machine-based cloud to provide an IoT-based manufacturing ERP instance, which facilitates the ERP procedure and also helps manufacturers find partners for the production.

—*Communication*. The cost of communication at runtime can be very dynamic, not only due to the traffic changes of the Internet Service Provider (ISP) side but also influenced by the number of IoT devices involved and where the requester is located. For example, when a ubiquitous IoT application requires using WSN in its surroundings, the process is mainly relying on the distant data center. Explicitly, such an approach can cause high latency when the Internet connection is not in good condition. Therefore, many approaches utilize proximity-based resources for data acquisition and processing-intensive applications. For example, the concept of mobile ad hoc cloud computing can cater to such need [Loke et al. 2015]. Alternatively, cloudlet-based MCC [Gao et al. 2012] is also an efficient option. Further, the extension of cloudlet known as Fog computing [Bonomi et al. 2012] has been getting industry attention recently, although existing BPMS4IoT frameworks have not yet explicitly addressed the Fog computing-enabled system.

3.3.4. Scalability. Scalability is one of the major requirements for management of large-scale connected devices [Conti et al. 2012; Teixeira et al. 2011; Issarny et al. 2011; Borgia 2014]. In existing BPMS4IoT frameworks, scalability involves two specific topics: the growing volume of stream data and the growing number of BP participative devices.

- The growing volume of stream data is derived from various data sources cross different parties. The large volume of data is utilized to provide the need for domain-specific applications such as disaster recovery, urban computing [Salim and Haque 2015], and social computing [Wang et al. 2007; Chang et al. 2012b]. In order to handle the large volume of stream data from different sources, a common approach is to utilize service-oriented Enterprise Service Bus (ESB) architecture together with elastic cloud computing resources [Appel et al. 2014].
- The growing number of BP participative devices represents the execution of BP and will involve numerous heterogeneous IoT devices. Existing works that have addressed this topic can be classified into two types: centralized and decentralized.
  —In the centralized solution, Wu et al. [2012] introduced the Gateway as a Service (GaaS)-based architecture that enables interorganizational collaboration among the IoT devices. The GaaS architecture is based on the foundation of cloud services (Infrastructure as a Service, Platform as a Service, Software as a Service). By integrating the cloud services and the mediating technology, enterprises can connect their IoT entities as Web of Things (WoT) collaboratively.
  —In the decentralized solution, Dar et al. [2015] proposed a framework that enables self-management in edge networks. It is realized by hosting embedded workflow execution engines on the participative IoT devices. Since the workflow execution engines can execute the standard BPMN workflow model, the system can dynamically distribute the processes among the IoT devices in the edge network without pre-establishing the topology. Hence, it achieves the choreography-based BP scalability.

Discussion

The major purpose of BPM is to optimize the processes of organizations. The BPMS designed for IoT needs to clearly address the challenges in the field. Currently, existing works in BPMS4IoT are still in early stages. Most works are focusing on proposing the

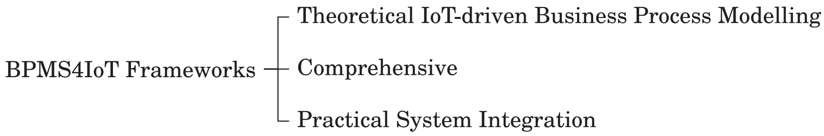


Fig. 7. Taxonomy of BPMS4IoT frameworks.

solutions for the previous two phases—*(re)design* and *implement/configure*. Although some works discussed in this section have addressed a few issues in the *run-and-adjust phase*, they have not proposed concrete, generic solutions for the specific challenges involved in the IoT environment.

The subjects described in this section highly influence the success of the IoT system when mobile is involved (e.g., in AAL, logistics use cases). Specifically, each of the subjects faces the same challenges as in many MCC solutions. For example, in the literature study proposed by Fernando et al. [2013], the authors have summarized numerous projects that have addressed fault tolerance, context awareness, and scalability in mobile and wireless networks. For this reason, it indicates that many runtime management solutions proposed for MCC can be applied in BPMS4IoT.

4. COMPARISON OF BPMS4IOT FRAMEWORKS

Research projects in BPMS4IoT propose their frameworks for different objectives. Overall, we can classify these frameworks into three types (see Figure 7): (1) frameworks for theoretical IoT-driven business process modeling, (2) frameworks for the comprehensive solution that covers both modeling and practical system integration, and (3) frameworks for practical system integration only.

In the following discussion of BPMS4IoT frameworks, we divide them into two framework comparison sections: theoretical modeling and practical system integration. Afterwards, we discuss the feasibility of the frameworks in the different IoT deployment models.

4.1. Modeling and Architecture Design

In this section, we discuss the modeling approaches of these frameworks, including the frameworks that focused only on modeling and frameworks that provided comprehensive solutions. First, we summarize each involved framework.

IoT-Driven Business Process Modeling Frameworks

—**IoT-A** [Castellani et al. 2012; Meyer et al. 2011, 2013; Sperner et al. 2011] is one of the Seventh Framework Programme for Research and Technological Development (FP7) projects that focuses on designing IoT-driven BP models and system architecture for future ERP systems. In the system architecture of IoT-A, the *things* of IoT are specifically representing nonelectronic physical entities such as chocolate, bottles of milk, animals, and so on. Further, *things* are connected via IoT devices (e.g., RFID reader, wireless sensors, etc.) to information systems as resources. The system can further adapt the resources to atomic or composite IoT services for external applications to interact. Previously, Section 3.1 discussed the BP model design of IoT-A. Accordingly, the analysis and the proposed metamodel of the BPMN extension in the IoT-A project provide guidance for the (re)design phase of BPMS, especially in clarifying the differences between the real-world physical entities, IoT devices, and IoT services, which can be a useful foundation for developing BPMS4IoT.

—**Sungur et al.** [2013] proposed a framework for extending BPMN with WSN elements. They identified the model requirements of WSN based on the characteristics of WSN devices and services. Further, they proposed a design for the WSN-driven BPMN extension, which includes a number of WSN-specific notations such as Sense Task, Actuate Task, and Intermediary Operation Tasks. Moreover, the framework provides the corresponding XML Schema for each proposed element and a modeling tool based on the extension of the web-based Oryx BP editor (http://bpt.hpi.uni-potsdam.de/Oryx).

—**uBPMN** [Yousfi et al. 2015, 2016] is a project aimed at introducing ubiquitous elements in BPMN. The authors defined the BPMN Task extension for Sensor, Reader, Collector, Camera, and Microphone. Each element also has a corresponding BPMN-Event symbol for Start Event and Intermediate Event. Additionally, uBPMN also introduced an IoT-driven Data Object called Smart Object to represent the data transmitted from the IoT devices. In general, the framework aims to serve as guidance similar to the IoT-A project.

Comprehensive Frameworks

—**makeSense** [Casati et al. 2012; Tranquillini et al. 2012, 2015]. The FP7 project makeSense aims to provide a comprehensive solution that facilitates the programming and system integration with WSN. As a comprehensive solution, makeSense involves both theoretical IoT/WSN-driven BP model design and practical system integration software.

In the model design, makeSense introduced two separated BPMN concepts: Intra-WSN Pool and WSN-aware Pool.

- *WSN-aware Pool*—represents regular BPMN Pool, which describes the main workflow of the operation. The WSN-aware Pool associates with Intra-WSN by denoting the IoT/WSN activities in a simplified WSN activity notation.
- *Intra-WSN Pool*—describes the detailed workflow description of the IoT/WSN tasks. Previously, Figure 5 showed a similar design example.

The decomposed design introduced in the makeSense project helps modelers with a clear perspective of the integration system. Further, the project has implemented the proposed modeling approach as an extension of Signavio Core Components called the BPMN4WSN editor.

—**ASPIRE** [Kefalakis et al. 2011]. The modern smart manufactory ERP system utilizes RFID technologies to facilitate the supply chain processes. However, such systems commonly require the engagement of low-level RFID-driven programming tasks, which is time-consuming, mainly because of the lack of the standard integration model. In order to overcome the problem, the FP7 project ASPIRE aims to tackle such issue by introducing a new specification and its practical platform.

The ASPIRE project introduces the AspireRFID Process Description Language (APDL) specification, which is the extension of XPDL, to leverage the EPCGlobal specification with BPMS. Fundamentally, APDL extends the feature of the EPC-Global architecture in generating automatic business process events that can assist the filtering of RFID stream data.

Accordingly, APDL contains two main concepts for describing the business processes: (1) Open Loop Composite business process (OLCBProc), which describes the BP execution among different individual systems (i.e., the interorganizational BPMS), and (2) Close Loop Composite Business Process (CLCBProc), which describes the execution within one individual system (i.e., the intraorganizational BPMS).

The project has developed a modeling tool called Business Process Workflow Management Editor (BPWME), which is an Eclipse IDE plugin that allows the modeler to configure the RFID-driven BP for both the OLCBProc and CLCBProc levels.

—**SPUs** [Appel et al. 2014]. Event-driven Process Chains (EPCs), the flowchart-based approach for BP modeling, are commonly used in ERP systems. In order to integrate an IoT-driven ERP system in which the system needs to handle the IoT device generated event streams, the Event Stream Processing Units (SPUs) project proposes a middleware framework to translate the EPCs to the service-oriented BPMN-based system.

The SPUs project provides a comprehensive solution in both model design and execution platform. In order to model the IoT-driven event streams, the SPUs project introduces a number of new EPCs elements, together with the mapping approach between the EPC elements and BPMN. Specifically, the new BPMN element *event stream task* can represent the continuous event stream data processing task. The flow of such task proceeds by the boundary noninterrupt signals attached to the task. Ideally, such a design provides flexibility for the system to handle the event stream independently. Further, for the model design, the project has also developed the extension of Software AG's ARIS Process Performance Manager platform for including the proposed BP model approach.

—**Caracaş et al.** Caracaş and Kramp [2011] and Caracaş [2012] of IBM Zurich Research have proposed a highly integrated WSN-driven BPMS framework for HVAC in which the IoT/WSN devices are participating in BP based on executing the dynamically deployed tasks. In general, the project's theoretical BP modeling focuses on a WSN-driven workflow pattern design. The proposed patterns include:

(1) completion of asynchronous operation in WSN;
(2) parallel starting asynchronous operation in WSN;
(3) WSN task exception handling;
(4) significant states of asynchronous interface;
(5) addressing IoT/WSN devices in BPMN;
(6) single and multiple message receiving in WSN; and
(7) processes involving Time Division Multiple Access (TDMA) protocols.

These patterns can serve as guidance for model designers who are involved in WSN-driven workflow designs. Additionally, the authors have implemented the modeling design as the extension of the web-based Oryx BPMN2.0 editor.

Discussion

Table III provides a comparison of the modeling frameworks. Specifically, modeling-based frameworks aim to introduce elements in existing modeling standards such as BPMN. In general, a complete solution should include the graphical model approach, its associated metamodel, and the model schema. Finally, the modeling approach needs to also provide a practical tool. For example, the approach can provide an extension of an existing BP model editor, which also can generate a machine-readable metamodel for further use in the *implement/configure phase*. As the table shows, most frameworks have provided the complete need of modeling frameworks. Moreover, the comparison also shows that most modeling frameworks only focus on the model for centralized BP execution and intraorganizational BPMS. Although two frameworks have addressed distributed BP execution, they have not fully addressed how to model the IoT-driven BP model in terms of representing the IoT tasks, data objects, and events. Similarly, the only framework that involves interorganizational BPMS also has not provided a solution for modeling the IoT elements in the interorganizational level. These indicate the research gap of BPMS4IoT in the modeling domain.

4.2. System Integration

In general, we can classify the system integration frameworks into two taxonomies (see Figure 8): the *atomic BP participation* denotes a model that allows processes involving IoT devices as static operations such as HTTP/CoAP service invocations

Table III. Comparison of IoT-Driven Business Process Modeling Frameworks

	Scope		BP Operation		Modeling Methods					Modeling Tools				
	IA	IR	CN	DI	EL	MM	PP	SM	SP	SC	OY	BW	AR	PT
IoT-A					✓	✓				✓				
makeSense	✓		✓	✓	✓				✓	✓				
ASPIRE	✓	✓	✓					✓	✓			✓		
Sungur et al.					✓	✓		✓			✓			
uBPMN	✓		✓		✓	✓		✓						
SPUs	✓		✓		✓								✓	
Caracaş et al.	✓		✓	✓			✓				✓			

✓ = subject addressed/supported; (blank) = not addressed/supported.

Scope		BP Operation		BP Modeling Approach			
IA	Intraorganizational	CN	Centralized	EL	IoT element	PP	Process pattern
IR	Interorganizational	DI	Distributed	MM	Metamodel	SP	Specification
				SM	Schema		

Modeling Tool			
SC	Signavio GmbH Signavio Core Components	OY	Oryx BPMN editor
BW	Business Process Workflow Management Editor	PT	Proposing modeling editor tool
AR	Software AG - ARIS		

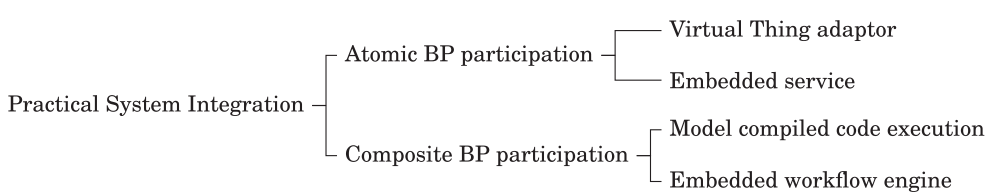


Fig. 8. Taxonomy of integration frameworks.

where the behaviors of the IoT devices cannot be reprogrammed at runtime. On the other hand, in the *composite BP participation* model, the behavior of IoT devices can change dynamically at runtime based on the workflow model assigned to them. Furthermore, based on the discussion in Section 3.2.2, each of the taxonomies can further be split into two subtaxonomies. We summarize the system integration frameworks in each subtaxonomy as follows.

Virtual Thing Adaptor-Based Integration

—**Adventure** [Schulte et al. 2014]. The goal of FP7 project Adventure (ADaptive Virtual ENTerprise manufacturing Environment) is to develop a platform that can help the manufacturing process as a virtual factory in the cloud. Considering that the source of production will become highly distributed in the future, the platform utilizes IoT technologies to assist in the governance of the processes.

With Adventure, the system creates each production plan as an instance managed in the cloud, in which the cloud instance of production proceeds with the BPMS that can discover and integrate the production system from client manufacturers.

In order to deploy the BP model, the proposed Cloud Process Execution Engine (CPEE) will translate the BPMN to the proposed Domain-Specific Language (DSL), which specifies the code to express the workflow and is directly executable as Ruby language.

Basically, the architecture of Adventure follows a general SOA model, in which the Cloud BPMS communicates with the IoT/WSN devices of different manufacturers via the flow of *Gateway-to-Adaptor→Adaptor to IoT / WSN devices*, which indicates that the expectation of IoT/WSN devices is to provide or to install the adaptor software that can enable the communication. Particularly, an expectation of Adaptor can be the classic HTTP mobile web servers.

—**ASPIRE** [Kefalakis et al. 2011]. The system integration of ASPIRE provides a run-time middleware—AspireRFID Programmable Engine (PE). Generally, the PE handles the low-level configuration between the central system and RFID applications. Further, it deploys the configuration generated from BPWME (BP editor) to the executable workflow for the system.

—**SPUs** [Appel et al. 2014]. As mentioned previously, SPUs propose the modeling approach as the extension in the ARIS Process Performance Manager platform. Accordingly, the project has developed an ESB-based middleware, Eventlet, which can execute the output BPMN metadata from ARIS.

Eventlet, which mainly communicates by WSDL/SOAP, provides the event stream filtering mechanism based on Java Message Service (JMS). Further, since the framework utilizes ARIS, which is one of the well-known process mining tools [van der Aalst 2015], it indicates the potential extension for introducing system optimization mechanisms in the future.

—**GaaS** [Wu et al. 2012]. The Gateway as a Service (GaaS) project aims to propose a platform that can easily integrate heterogeneous IoT devices with web-based BPMS. The architectural design of GaaS is based on cloud-centric SOA, which consists of three main layers:

  ○ *WoT Infrastructure layer* corresponds to IaaS, which shares resources with third-party systems. In this layer, the front-end IoT devices connect with their own back-end servers and the back-end servers can join the WoT Infrastructure layer as Gateway services.
  ○ *Service and Business Operation layer* corresponds to PaaS, which handles the service composition and business process management.
  ○ *Intelligent Service layer* corresponds to SaaS, which provides application and UI to end-users.

The IoT device integration in GaaS derives from the Valpas Gateway developed by Aalto University based on ThereGate. Additionally, ThereGate is a practical development of the Nokia Home Control Center (HCC). It is a Linux-based platform with an open interface and a software engine called ThereCore, which integrates IoT devices operated on ZigBee/Z-Wave/Bluetooth protocols.

Embedded Service-Based Integration

—**RWIS** [Yang et al. 2012]. The RESTFul geospatial Workflow Interoperation System (RWIS) project proposes a new platform that utilizes Open Geospatial Consortium (OGC) Web Processing Services (WPSs) for sensor devices to provide geographical composite services. RWIS utilizes XPDL to describe BP models and the system will translate the XPDL to BPEL for execution.

RWIS organizes Sensor Planning Service (SPS) and Sensor Observation Service (SOS) as Sensor Information Accessing (SIA) workflow. Further, it deploys OGC's Web Processing Service (WPS) and OGC's Web Coverage Service (WCS) Sensor Information Processing (SIP) workflow. Generally, the system achieves the interoperation in OpenWFE-based SIA workflow and BPEL-based SIP workflow. From the practical perspective, RWIS is focusing on integrating IoT with BPMS fully based on the compliance of OGC's web service standards.

—**Decoflow** [Loke 2003] is a service-oriented WfMS proposed for home automation. The Decoflow project introduces a modeling language called DySCo for modeling the abstract BP workflow. Fundamentally, the Decoflow runtime system is based on service-oriented BPEL4WS, in which the communication between the management system and devices relies on the embedded SOAP web services hosted on the devices. The project also proposes a graphical design tool, which can generate and validate the BPEL metamodel for execution. Further, the extension of the project introduced a *fault-tolerant scheme* in which the system solves the runtime device failure by using context-aware failure substitution. The faulty device will be replaced by substitution [Chang and Ling 2008] based on the same class of context they can provide.

Model Compiled Code Execution-Based Integration

—**makeSense** [Casati et al. 2012; Tranquillini et al. 2012, 2015] provides *model-to-execution* mechanisms for the system integration. It mainly supports three approaches:

(1) The proposed BPMN4WSN compiler will generate the IoT/WSN device executable binary code from the metamodel generated from the proposed BPMS4WSN editor.
(2) If the IoT/WSN device is powerful enough to embed the Process Engine (i.e., for executing the workflow model), the BPMN4WSN compiler will generate an executable workflow for the device.
(3) If the IoT/WSN device is communicated via the proxy device, the BPMN4WSN compiler will generate proxy configuration and send the configuration metadata to the proxy device.

The makeSense project has tested the prototype on Contiki devices that can execute the output from the proposed BPMN4WSN compiler. Overall, makeSense is suitable for the systems that require distributed and dynamic deployed BP workflow execution at the edge network of IoT systems.

—**Caracaş** [2011, 2012] propose a model compiler middleware that can translate the editor-generated BP model to executable program source code in C# or Java language, which is executable by IBM Mote Runner OS-based devices.

Further, considering that the runtime deployment-based BP participation can influence the power consumption of IoT/WSN devices, the project also proposes a strategy to reduce the power consumption based on the optimized sleeping scheme. Generally, the scheme utilizes the machine-learning algorithm that learns from the power consumption of the wakeup on IoT/WSN devices and autonomously reconfigures the process.

—**Presto** [Giner et al. 2010] is a pluggable software architecture for leveraging Tag-based IoT technologies with BPMS. It focuses on supporting human workers' participation in the BP in which the mobile devices serve as the medium between human works and the system. Fundamentally, Presto's primary role is to assist the human-worker-involved workflow task allocation. It can automatically allocate tasks to workers depending on certain contextual factors such as how many pending tasks exist on the worker's to-do list.

Presto also proposed a middleware platform, Parkour, which is capable of translating the Parkour model (a customized modeling specification) to the Ecore metamodel (part of the Eclipse Modeling Framework). Afterwards, the Ecore metamodel can be translated to the execution languages of a specific platform such as Java, Android, or iOS.

—**LTP** [Glombitza et al. 2011]. The Lean Transport Protocol (LTP) project introduces a middleware framework to integrate IoT devices with BPMS with the enhancement of the proposed new protocol stack. In this project, IoT devices communicate with the BPEL system using LTP-enhanced messages. Additionally, LTP provides SOAP message compression (SMC) to improve the communication performance in the edge network of the IoT system.

Overall, the project facilitates the BP participation from IoT devices by utilizing the dynamic model-compiled code deployment approach in which the BPEL workflow model is compiled into C++ code. Afterwards, the IoT devices that have embedded GNU Compiler Collection (GCC) C++ compilers can execute the compiled code.

Embedded Workflow Engine-Based Integration

—**MOPAL** [Peng et al. 2014]. The MCC system is a common approach used in healthcare applications where physicians can remotely allocate a sequential task to the healthcare nurses via their mobile devices, which act as mediums. Commonly, such application is mainly relying on the cloud-side server for deploying, executing, and managing the entire BP workflow. However, it also faces the problem derived from the unstable mobile Internet connection.

The MOPAL project aims to solve the problem by introducing the disconnected workflow execution approach in which the physician-defined workflow can be dynamically executed on the nurses' mobile device without a need for the Internet. In general, the architectural design of MOPAL follows the common SOA that utilizes the native components of mobile devices (e.g., camera, microphone) as services. Hence, the embedded workflow engine on the mobile device can access the native components seamlessly.

The framework also contains a customized BPMN workflow developed based on XPath and SQLite. The project has implemented the MOPAL prototype for Android OS. In general, MOPAL decouples the BP execution between the distant cloud servers and the front-end mobile mediums.

Further, for runtime management, MOPAL defines two context constraints to reduce the runtime failure caused by the context influences. The first class of the constraints is the assignment constraints, which are based on the matchmaking between the workflow task assignment and the profile and context information of the candidate participant. The second class of the constraints is the execution constraints, which are related to the context of the physical world such as current geographic location and time.

—**Dar et al.** [2015]. AAL applications often utilize smartphones as mediums to interact with the heterogeneous front-end RESTful service-based IoT environments. In order to fully support the need for highly dynamic workflow deployment, the system requires the mechanism for dynamically deploying and executing the workflows on smartphones.

Dar et al. propose a framework for enabling the RESTful service integration-based workflow system with the feature of the choreography support. In this case, the system can distribute the workflow to different front-end IoT devices for execution.

The project has implemented a prototype on an Android OS device based on the ported version of the Activiti BPM engine. Additionally, the framework includes common protocols used in IoT systems such as CoAP for constrained service invocation and MQTT for event stream subscription.

Furthermore, the event stream mechanism enhances the runtime device failure detection. The system monitors the health of the BP execution by utilizing the MQTT-based periodical reporting method. If the system detects the failure, it is capable of performing substitution by reassigning the workflow to a different device for execution.

Overall, the choreography feature provided by this project reduces the need for the frequent transmission between front-end BP activities and the distant cloud, which is also a well-studied approach in the MCC field [Chang et al. 2015b].

—**SCORPII** [Chang et al. 2015b]. In the IoT technology-assisted smart urban area, heterogeneous devices are discoverable and they can provide various information. When an AAL application relies on the information provided by the surrounding IoT devices, it faces the challenge of how to rapidly discover the devices that can provide relevant information in a timely manner.

In order to resolve the question, the SCORPII project provides a middleware framework based on dual BPEL workflow execution engines hosted on the dynamically launched utility cloud and the user's mobile device. The workflow system is capable of dynamically assigning the service discovery tasks between cloud and mobile in order to achieve the best cost-performance efficiency.

In detail, SCORPII's mobile-embedded BPEL workflow execution engine is a customized engine developed for iOS devices. The engine utilizes GDataXML (XPath) and CocoaHTTPServer for the practical implementation. The prototype supports the main operations of BPEL such as sequential and parallel task executions.

Further, for the runtime management, SCORPII supports the resource-aware cost-performance index scheme to identify the most efficient way of executing the workflow tasks between the cloud and the mobile device. Generally, the decision making is based on the environmental factors such as how large the service description data from the distant discovery servers.

Other Related Frameworks

Beside the frameworks described earlier, there are other relevant frameworks with little involvement of BPMS. Following is the summary of these frameworks. Note that the comparison table does not include these frameworks.

—**EBBITS** [Furdik et al. 2013] is an FP7 project that mainly focuses on CPS integration using LinkSmart SDK (https://linksmart.eu/). Accordingly, the project plans to utilize BPMS for orchestration with the RESTful service-embedded IoT devices. EBBITS refers the work of IoT-A for their BPMS-related components.

—**VITAL** [VITAL-IoT 2016] is an FP7 smart city project that proposes a new domain-specific language-based BP modeling and configuration approach. The approach aims to introduce the modeling language as a new specification for the citywide system integration that can support the composition among interorganizational IoT management systems.

—**edUFlow** [Jung et al. 2012] is a practical integration framework that composes the open-source Esper event correlation engine and GlassFish Message Queue to enable an IoT-driven event stream processing platform. The project also provides a GUI-based editor for process configuration and runtime monitoring.

Table IV illustrates an overview of the existing BPMS4IoT integration frameworks. Based on the characteristics from the Mobile Cloud Computing perspective, these frameworks may fulfill different needs in integrating mobile/wireless IoT devices into BPMS.

Discussion

The comparison indicates that the Virtual Thing Adaptor-based integration model is commonly applied in the large-scope enterprise systems such as manufacturing and logistics where the primary purpose of the integration is to trace the items. In general, the IoT-driven manufacturing and logistics systems only utilize the fairly simple IoT device such as RFID/RFID readers or simple function sensors.

Table IV. Comparison of System Integration Frameworks

	Domain Use Case	Integration Model	IoT Device Integration Platform / Protocol	BP Model Standard	BP Execution Engine	Prototype IoT Device Platform	Runtime Management
Adventure	Manufacturing	Virtual Thing adaptor	Cloud-centric WS	BPMN	Customised BPMN engine	—	—
ASPIRE	Logistics	Virtual Thing adaptor	ASPIRE PE	XPDL	BPWME & Aspire-RFID middleware	RFID	—
SPUs	Logistics	Virtual Thing adaptor	SOAP	BPMN	Software AG ARIS platform	EPC/RFID	Event stream filtering
GaaS	AAL/ Healthcare	Virtual Thing adaptor	ThereGate	BPEL	Apache ODE	ZigBee/Z-wave sensors	—
RWIS	Location-based service	Embedded service	OGC WS	XPDL & BPEL	RWIS platform	OGC sensors	—
Decoflow	Smart home	Embedded service	SOAP WS	BPEL	GlassFish	—	—
makeSense	HVAC	Compiled code†	Contiki C	BPMN	Model compiled C code execution	Contiki OS devices	—
Caracas et al.	HVAC	Compiled code†	IBM Mote Runner	BPMN	Model compiled C#/ Java code execution	Mote Runner VM/OS devices	Energy efficiency
Presto	Smart business	Compiled code†	OSGi	—	Parkour / Java OSGi	OSGi on Android	—
LTP	Logistics	Compiled code†	LTP & SOAP	BPEL	Model compiled C++ code execution	GCC on Android	—
MOPAL	AAL/ Healthcare	Embedded WF engine‡	SOAP	BPMN	Customised BPMN engine	Android	Constraint execution
SCORPII	AAL/ Healthcare	Embedded WF engine‡	RESTful HTTP	BPEL	Customised BPEL engine	Android	Cost-performance balancing
Dar et al.	AAL/ Healthcare	Embedded WF engine‡	RESTful HTTP/ CoAP/MQTT	BPMN	Ported Activiti BPM engine	Android	Reactive failure substitution

†Model compiled code execution.
‡Embedded workflow engine.

Embedded service-based approaches deploy the IP-based web services on devices to enable the common service invocation between the *smart devices* and the management system. Generally, this model is fully compliant to BPEL4WS-based BPMS and it usually applies to the location-based systems or the smart home systems.

In the case of a large-scale IoT-driven system such as HVAC or a smart building system where the system requires a certain level of self-management and self-configuration, there is a need to provide dynamic process configuration on the resource-constrained IoT devices. However, IoT-devices used in HVAC or smart building systems usually have constrained resources, in which implementing the standard-compliant workflow execution engines on them is not performance efficient. Therefore, the model compiled code execution-based approach becomes the promising solution. Accordingly, the model compiled code execution-based frameworks usually provide a complete solution that covers everything from BP modeling tools to the deployment of the runtime system.

As Table IV shows, all the projects that utilized the embedded workflow engine-based approaches were originally developed for AAL scenarios. Explicitly, such scenarios utilize smartphones as mediums for interacting with the IoT-driven ubiquitous environments and they require an on-demand timely response and also choreography-based service composition. Hence, the embedded workflow engines become a feasible option.

According to the study of this section, current frameworks are in the early stage since they have not explicitly addressed optimization in a BPMS. Optimization involves scalability and continuously improving the BP model using techniques like process mining or process discovery.

Further, only a few frameworks have considered fault tolerance and context awareness in the runtime management of BPMS4IoT. At this stage, it is not clear whether the existing process models can already address the fault tolerance and context awareness or not, especially when the IoT system involves mobile objects. Since the BPMS4IoT projects commonly design their BPMSs for the static participant and controlled environment, integrating BPMS with IoT can raise new issues. For instance, addressing fault tolerance requires the process model to describe the detail process of the involved entities. The BP model that does not address the activities of the IoT entities (e.g., sensor, actuator, reader, etc.) may not be able to identify the cause of the failure, and hence, the BP modeler cannot design the corresponding recovery processes for the potential failures.

4.3. Application Comparability Comparison

In this section, we try to identify the feasibility of the existing BPMS4IoT framework when we apply it to recent IoT systems. Generally, we can classify the recent IoT systems' architecture into three main deployment models, which are fundamentally similar to the three MCC model types that have been discussed in Fernando et al. [2013]. The three models are (1) Distant Data Center (Figure 9(a)), which refers to the Distant Mobile Cloud model (Figure 9(d)); (2) Fog computing [Bonomi et al. 2012] (Figure 9(b)), which refers to the Mobile Edge Cloud model (Figure 9(e)); and (3) Ad hoc Computing [Kortuem et al. 2010] (Figure 9(c)), which refers to Mobile Crowd Computing [Loke et al. 2015] (Figure 9(f)).

We summarize the three IoT models as follows:

—**Distant data center** is a classic model in which the cloud-side representations (e.g., virtual thing adaptor) of the IoT/mobile devices are delegating the BP for the devices. Most BPMS4IoT frameworks apply this model.
—**Fog computing** refers to mobile edge computing [Patel et al. 2014] where the system utilizes the MCC-driven *cloudlet* [Satyanarayanan et al. 2009] concept. Generally, a

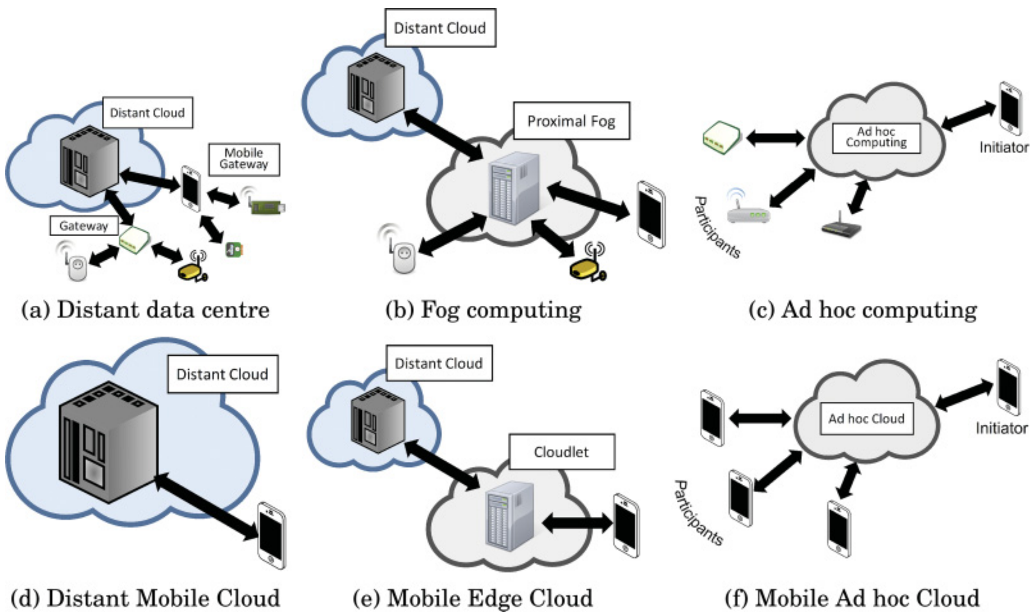


Fig. 9. Correlative MCC and IoT deployment models. Note: the mobile devices can be replaced by any type of IoT devices.

cloudlet is a Virtual Machine (VM)-enabled server machine colocated in a cellular base station or a WLAN access point (e.g., CISCO Grid router). Notably, it is one of the major models of the IoT system, which is to fulfill the need for the rapid response from location-based ubiquitous sensing and context recognition processes. It is foreseeable that the future BPMS4IoT will highly involve in this model. For instance, the *cloudlet* can compile the BP model, encode/decode the message, handle the runtime monitoring, and process the event stream. Further, it can work as the middleware or adaptor between the distant cloud and the front-end mobile nodes.

—**Ad hoc computing** can refer to Mobile Crowd Computing [Loke et al. 2015] and Mist computing [Pulli et al. 2011; Preden et al. 2015; Martin 2015], which focuses on utilizing the edge network devices as the process participants. Generally, this model enables the devices to communicate with each other and collaboratively perform the tasks together. Such cooperative business process distribution requires a highly flexible execution mechanism. Hence, an embedded workflow engine is the basic requirement to meet the high automation need.

Table V illustrates the compatibility between the existing BPMS4IoT frameworks and IoT system models. This classification is based on the design of their system architectures and deployment approaches. Since the *ad hoc computing* model requires flexible and timely process execution, the frameworks that provide embedded workflow engines are more suitable than the others. The *Fog computing* model requires distributed process execution to Fog nodes situated in close proximity to the requesters. Therefore, the frameworks that provide the middleware for deploying model-compiled code to IoT devices or the frameworks that utilize embedded workflow engines are more feasible. As the table shows, most frameworks are compatible with the classic *distant data center* model because they either lack the capability for distributing processes or their architectural designs and BP model designs have not considered the applications based on Fog or mobile ad hoc environments.

Table V. Compatibility in MC-BPMS4IoT Models

	Distant Data Center	Fog Computing	Ad Hoc Computing
IoT-A	✓		
Sungur et al.	✓		
uBPMN	✓		
Adventure	✓		
ASPIRE	✓		
SPUs	✓		
GaaS	✓		
RWIS	✓		
Decoflow	✓		
makeSense	✓	✓	
Caracas et al.	✓	✓	
Presto	✓	✓	
LTP	✓	✓	
MOPAL	✓	✓	✓
SCORPII	✓	✓	✓
Dar et al.	✓	✓	✓

5. POTENTIAL ISSUES AND OPEN CHALLENGES

Based on our study, in this section, we identify a number of challenges that have not yet been fully addressed in existing BPMS4IoT frameworks.

5.1. Challenges in (Re)design

5.1.1. IoT-Driven Business Process Model Standardization. Currently, there is no common agreed-upon way to model the IoT entities and their activities in business process model tools. Besides the different devices that require different ways of modeling (e.g., modeling actuator is different from modeling data collector), different projects have their own perspective in modeling the same types of entities. For example, based on the review of this article, there are at least four different approaches to model the sensor devices. Further, there is a need to model the IoT devices in detail in order to ease the procedure of transforming the process model into the machine-readable metadata form for the execution. Further, the need for the standard model in BPMS4IoT is not only about the notations; it also involves the metamodel level. Ideally, it is better to have the capabilities for the BP modeler to define what kind of protocol and operation the IoT devices use and how the processes perform the message flow. Such a need is required for BPMSs that need to compose information from different sources dynamically.

The potential approach in this domain may compose the modeling standards [Ko et al. 2009] with the recent design described in Section 3.1.3 to propose a generic specification that is applicable in different IoT use cases.

5.1.2. Hybrid Computational Process Architecture. The performance-related challenge in IoT motivated various edge network computing models, specifically, the Fog Computing model [Bonomi et al. 2012], which utilizes the VM-enabled grid router machine to replace the partial mechanism of the distant cloud services, and Mobile Edge Computing model [Patel et al. 2014], which utilizes the server machine colocated with the cellular network base stations as the computational resources. Furthermore, the Mist Computing model [Pulli et al. 2011; Preden et al. 2015; Martin 2015] or the Mobile Crowd Computing model [Loke et al. 2015], which utilizes proximal IoT/Mobile devices as the resources for computational offloading, is also showing as a promising approach in improving the performance of IoT systems. Since all these models involve vast devices that operate with their own OS independently, composing these models in BPMS4IoT requires a more technical design to support the reliability of the system because of the dynamic nature of the heterogeneous mobile network environment.

5.1.3. BPM in the Large. BPMS4IoT shares similar *design phase* challenges with *BPM in the Large* [Houy et al. 2010]. For instance, the complexity derived from the large scope of interorganizational BPMS4IoT requires the extensive process models that involve situation awareness. In other words, the BP models need to be self-adaptive in different situations based on the stakeholders. Specifically, the challenge in the large-scope system involves an adaptive BP model design and runtime management solution for the interorganizational BPMS4IoT. Accordingly, existing BP model frameworks have not addressed this domain.

5.2. Challenges in Implement/Configure

5.2.1. Heterogeneity. The heterogeneity of the participative IoT devices in BPM involves the challenges in connectivity, discoverability, mobility/accessibility, self-management, and self-configuring. Here, we discuss the heterogeneity of participants in different layers.

—*Devices*. Although different IoT devices have different capabilities, they may achieve the same purposes. Hence, BPMS4IoT requires an efficient ontology model to classify the IoT devices. For example, both the modern high-end smartphone and the low-power Raspberry Pi can perform the sensory data collection tasks. On the other hand, since their computational power is quite different, it can affect the overall performance of the process. Therefore, the BPMSs need to clearly define the IoT entities.

—*Embedded Service*. Services provided by the IoT devices depend on the communication protocol such as classic Bluetooth, Bluetooth Smart LE, CoAP, Alljoyn (https://allseenalliance.org), MQTT, and AMQP (https://www.amqp.org). These common IoT protocols work in different ways, and they influence the performance of the management system. Future BPMS for IoT requires an adaptive solution to provide self-configuration among the different connectivities.

—*Service Description*. Existing frameworks have introduced various approaches to interacting with the IoT entities. For example, web service-oriented frameworks have applied WSDL, WADL [Hadley 2006], and DPWS [Driscoll et al. 2009]; frameworks that focus on WSN have applied SensorML (http://www.ogcnetwork.net/SensorML) and SenML [Jennings et al. 2013]; frameworks that are concerned about energy efficiency have utilized IETF CoAP (RFC7252) and CoRE standards. Further, W3C has recommended JavaScript Object Notation for Linked Data standard for describing the IoT entities. Consequently, it is now becoming impossible to rely on one single standard to enable autonomous Machine-to-Machine (M2M) communication in IoT due to the lack of a global common standard for machine-readable metadata.

—*Service Discovery*. In IoT, relying on a global central repository is less possible. It is foreseeable that in the future, IoT systems will rely on a large-scale federated service discovery network established on a mesh network topology. Moreover, the proximity-based and opportunistic discovery also stands for an important role to map the physical visibility with the digital visibility. The use cases of the interorganizational BPM, crowdsourcing, crowdsensing, social IoT, real-time augmented reality, and ambient assistance for mobile healthcare will all need such a feature to support the rapid establishment of M2M connection and collaboration in physical proximity, which indicates that the BP model design needs to consider the adaptive service discovery.

5.2.2. Urban Computing. The future IoT in a public area will also involve a collaborative sensing network [Salim and Haque 2015; Loke et al. 2015; Loke 2015] in which different organizations or individuals may share the already deployed resources such as sensors and sensory data collector devices [Wu et al. 2012]. Therefore, the process models for such an environment will become very complex and will face many challenges such as privacy and trust issues. The privacy issue involves what resources can be shared and how they will be shared. The trust issue involves how an organization provides the trustworthy resource or sensory data sharing. These issues will further require adaptive Quality-of-Service (QoS) models and the scalable Service-Level Agreement (SLA) schemes to adapt to different situations and preferences. In summary, the collaborative IoT environments need to address the following challenges.

—*Intraorganizational Distribution*. Distributed process in the same organization face basic challenges in mobility and integrating the resource-constrained participants. Although the model compiled code execution approaches [Glombitza et al. 2011; Caracaş and Kramp 2011; Casati et al. 2012; Tranquillini et al. 2012, 2015] are promising and efficient, they lack standard approaches to convert the designed process models to machine-readable, executable program. Further, deploying the process model (either the raw model or the transformed version) to the edge nodes also involves challenges in performance, especially in a large-scale IoT environment where the events can occur quite often and the frequency of changing processes is high.

—*Interorganizational Cooperated Devices*. Collaborative IoT devices bring many new possibilities in IoT. Organizations can share their deployed IoT devices with one another in order to reduce the deployment cost. However, it will raise new challenges in privacy, trust, quality of service, service-level agreement, and negotiation. Currently, existing frameworks have not fully addressed these issues. Additionally, the interorganizational BP model in IoT requires further study on the adaptive extension of the Public-to-Private workflow abstraction [van der Aalst and Weske 2001] in which the organizations need to design the common agreed-upon high-level BP model standards.

5.3. Challenges in Run and Adjust

5.3.1. Runtime Monitoring and Event Streaming. As mentioned in the previous section, runtime monitoring in BPMS4IoT is a crucial task because it involves many factors that relate to mobile computing and the wireless sensor network domain. Generally, existing BPMS4IoT frameworks have not broadly studied this topic. Although the time-out solution proposed by Dar et al. [2015] is capable of identifying the runtime failure of the edge network device, considering if the failed node is the cluster head of the edge network and there is no alternative node that can replace the cluster head, or if the failure is the current Internet provider of the edge network, the monitoring procedure of the edge network will fail entirely. Researchers in BPMS4IoT need to further investigate this issue.

Existing event streaming schemes [Jung et al. 2012; Appel et al. 2014; Tranquillini et al. 2015] were designed for specific scenarios. There is a lack of generic BP model for defining the streaming-type activities for intraorganizational, interorganizational, and edge networks. In order to develop the generic streaming BP model, developers need to understand how the front-end devices integrate with the back-end cloud services in the reliable channel and protocols. Further, they need to consider the performance and cost efficiency.

5.3.2. Energy Efficiency. The large-scale connected things provide various possibilities and also raise numerous challenges. One research interest in both academia and industry is energy conservation. In general, the IoT environment involves a large number of devices deployed in high density. These devices include battery-powered and AC-powered devices.

Besides the industrial standards such as CoAP and MQTT, which can reduce the energy consumption from data transmission, the application layer can also apply strategies such as utilizing cloudlet-based architecture [Gao et al. 2012] or utilizing a collaborative interorganizational data brokering scheme [Chang et al. 2015]. In the past, many works discussed such a collaborative network environment [Feamster et al. 2007; Sofia and Mendes 2008; Middleton and Bryne 2011; Frangoudis et al. 2011; Garikipati and Shin 2013; Cao et al. 2015], and the related commercial services have already existed for many years (e.g., http://www.fon.com).

As the recent perspective in applying *software-defined networking* (SDN) to the sensing cloud [Distefano et al. 2015], the interorganizational collaboration can be realized in a higher level controlled by a software system instead of depending on infrastructure hardware compliance. Hence, it leads to a research direction in developing the new BP model design that composes SDN, WSN, and edge computing.

5.3.3. Context Awareness. There exists a number of literature surveys in context-aware workflow management [Ardissono et al. 2007; Smanchat et al. 2008; Tang et al. 2008], which can serve as the starting point for addressing context awareness in BPMS4IoT. However, they focus on the classic systems in which there are no mobile IoT devices or on the common wireless IoT devices involved in the workflow processes.

Commonly, a system can utilize the rule-based schemes mentioned previously in Section 3.3.3 or utilize the machine-learning schemes based on the Bayesian network, Markov chain to achieve context awareness. In fact, recent research trends in context awareness have started applying process mining techniques [Jaroucheh et al. 2011; Pileggi et al. 2015a, 2015b]. Since process mining is a popular technique in BPM [Dumas et al. 2013], those process-mining-based context recognition schemes can be quite promising in supporting context-aware BPMS4IoT.

5.3.4. Scalability. Besides the two topics mentioned previously in Section 3.3.4 (i.e., the growing volume of stream data and the growing number of BP participative devices), scalability in BPMS4IoT also involves a growing number of location-based real-time applications, which is not a widely studied topic in BPMS, mainly because it emerged when the recent information systems were highly relying on the distant data centers for all the processes. The location-based real-time applications such as urban AAL [Chang et al. 2015b] requires low-latency response. However, the growing number of mobile users in the urban area increase the network traffic, in which the transmission speed of BP tasks (i.e., between the end-user application and the distant server) makes it hard to satisfy users. In order to resolve the issue, researchers in this domain can consider utilizing a hybrid infrastructure that composes the distant mobile cloud-based BPMS4IoT with mobile edge cloud and mobile ad hoc cloud. Such a design further involves the BP model design and the challenge in runtime monitoring and on-demand reconfiguration.

5.3.5. Intelligence. Intelligence in BPMS4IoT involves challenges in efficient collecting and preprocessing data. These topics involve wise decision-making strategies supported either by the tools such as the Complex Event Processing tool of the open-source Esper engine used in edUFlow [Jung et al. 2012] and as a part of the integration framework LinkSmart used in EBBITS [Furdik et al. 2013], Tibco (http://www.tibco.com), Drools (http://www.drools.org), or other machine-learning, data mining, artificial intelligence techniques. Since Cabanillas et al. [2013] have discussed a list of major challenges in the event stream monitoring of BPMS for logistics that involves both a cloud-side ERP system and the front-end vehicle-based mobile nodes, their work can serve as a guidance for the research in this direction.

Another major challenge of intelligence is the continuous optimization in BPMS4IoT. On this topic, Process Mining and Process Discovery techniques show promising solutions. For instance, Gerke et al. [2009] have proposed a process mining algorithm to improve an RFID/EPCGlobal-based logistic system. Zhang et al. [2012] and Carolis et al. [2015] have proposed the process mining schemes to improve AAL applications. All these schemes are directly related to BPMS4IoT. However, they were designed for specific application domains. Developers in different domains (e.g., mobile IoT, which involves mobility and unreliable connection issues) need further research to develop the most feasible solution for their systems.

5.3.6. Big Data Service. The deployment of BPMS4IoT will forward the enterprise systems to the Big Data era. In general, Big Data in IoT represents a vast volume of data generated from the IoT networks, and organizations can make use of them, which was not possible before. Ideally, organizations will gain benefit from the Big Data to improve and enhance their business processes more efficiently and more intelligently [Chandler 2015]. In order to realize the vision, BPMS4IoT needs to address the challenge of *varying forms of data*. The data of IoT comes in various formats from different coexisting objects in IoT networks. The information system can utilize machine-learning mechanisms to identify the correlation between the data from different objects and generate meaningful information. However, processing the various formats of data may not be a swift task. For example, in order to identify a suspicious activity in an outdoor environment, the system may integrate the video data and temperature data from different sensors and then either utilize an external third-party cloud service for the analysis processes or invoke the external database service to retrieve the related data and analyze them in the intraorganizational information system. The challenge is if such a need is on demand, how does the system generate the result in time, because it involves the data transmission time in different networks and the large volume of data processing?

Researchers addressing Big Data of BPMS4IoT can further refer to the studies in Munoz-Gama et al. [2014] and van der Aalst [2015], which provide guidance in identifying the requirement of this domain. The works have introduced a new concept, *Internet of Events (IoE)*, which represents a large volume of event stream data coming from the IoT system, and how promising process mining techniques can overcome the issues in IoE.

6. CONCLUSION AND FUTURE DIRECTIONS

The Workflow Management Coalition introduced the notion of *BPM Everywhere* [Fischer 2015] to represent the future IoT in which almost any part of the IoT system will utilize BPM. In such an IoT system, the deployment of BPMS is in the intraorganizational information systems, but it also composes the interorganizational BP activities. Specifically, the entities involved in the system include both back-end cloud services and the front-end edge network established by clusters of interconnected IoT devices and also the Fog service nodes. In order to realize such a vision, BPMS4IoT will face many specific challenges in each of its life cycle phases, which were not fully addressed in the past BPMSs.

This article has discussed the state of the art and challenges involved in each life cycle phase of BPMS. The study has shown that existing frameworks have not yet addressed many research challenges involved in BPMS4IoT. Further, most of the challenges highly relate to research in the MCC domain, which raises the opportunity

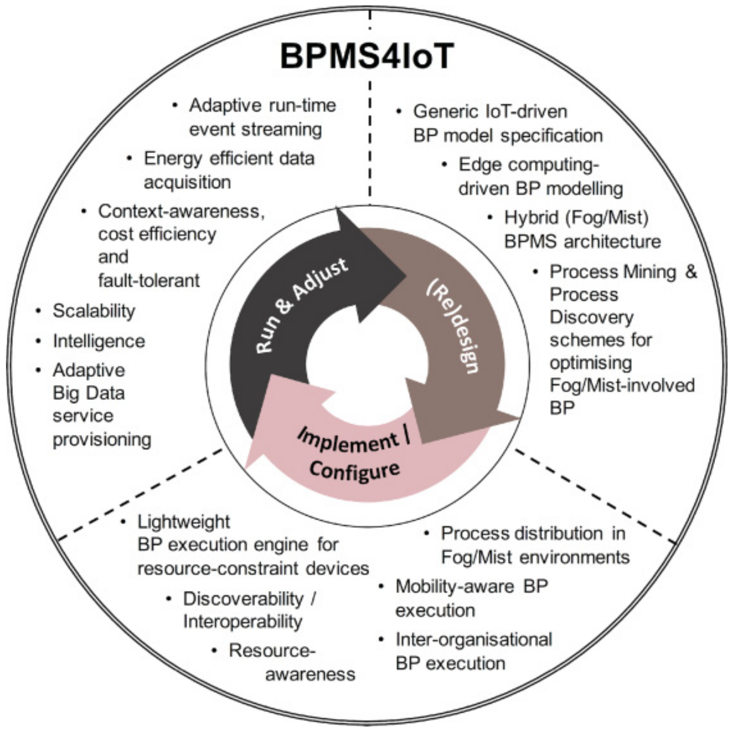


Fig. 10. Research roadmap in BPMS for IoT.

to the MCC discipline. In summary, Figure 10 illustrates a research roadmap as a future research direction in Mobile Cloud-based BPMS4IoT.

REFERENCES

Charu C. Aggarwal, Naveen Ashish, and Amit P. Sheth. 2013. The Internet of things: A survey from the data-centric perspective. In *Managing and Mining Sensor Data*. Springer US, New York, 383–428.

Sheeraz A. Alvi, Bilal Afzal, Ghalib A. Shah, Luigi Atzori, and Waqar Mahmood. 2015. Internet of multimedia things: Vision and challenges. *Ad Hoc Networks* 33 (2015), 87–111.

Stefan Appel, Sebastian Frischbier, Tobias Freudenreich, and Alejandro Buchmann. 2013. Event stream processing units in business processes. In *Business Process Management*. Springer, Berlin, 187–202.

Stefan Appel, Pascal Kleber, Sebastian Frischbier, Tobias Freudenreich, and Alejandro Buchmann. 2014. Modeling and execution of event stream processing in business processes. *Information Systems* 46 (2014), 140–156.

Liliana Ardissono, Roberto Furnari, Anna Goy, Giovanna Petrone, and Marino Segnan. 2007. Context-aware workflow management. In *Proceedings of the 7th International Conference on Web Engineering (ICWE'07) (Web Engineering)*, Luciano Baresi, Piero Fraternali, and Geert-Jan Houben (Eds.), Vol. 4607. Springer, Berlin, 47–52.

Luigi Atzori, Antonio Iera, and Giacomo Morabito. 2010. The Internet of things: A survey. *Computer Networks* 54, 15 (2010), 2787–2805.

Luigi Atzori, Antonio Iera, Giacomo Morabito, and Michele Nitti. 2012. The social Internet of things (SIoT)–when social networks meet the Internet of things: Concept, architecture and network characterization. *Computer Networks* 56, 16 (2012), 3594–3608.

Aznita Abdul Aziz, Y. Ahmet Sekercioglu, Paul Fitzpatrick, and Milosh Ivanovich. 2013. A survey on distributed topology control techniques for extending the lifetime of battery powered wireless sensor networks. *IEEE Communications Surveys & Tutorials* 15, 1 (2013), 121–144.

Soma Bandyopadhyay, Munmun Sengupta, Souvik Maiti, and Subhajit Dutta. 2011a. Role of middleware for Internet of things: A study. *International Journal of Computer Science and Engineering Survey* 2, 3 (2011), 94–105.

Soma Bandyopadhyay, Munmun Sengupta, Souvik Maiti, and Subhajit Dutta. 2011b. A survey of middleware for Internet of things. In *Recent Trends in Wireless and Mobile Networks*. Springer, Berlin, 288–296.

Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. 2012. Fog computing and its role in the Internet of things. In *Proceedings of the 1st Edition of the MCC Workshop on Mobile Cloud Computing*. ACM, New York, NY, 13–16.

Eleonora Borgia. 2014. The Internet of things vision: Key features, applications and open issues. *Computer Communications* 54 (2014), 1–31.

Alessio Botta, Walter de Donato, Valerio Persico, and Antonio Pescap. 2016. Integration of cloud computing and Internet of things: A survey. *Future Generation Computer Systems* 56 (2016), 684–700.

Maha Bouaziz and Abderrezak Rachedi. 2014. A survey on mobility management protocols in wireless sensor networks based on 6LoWPAN technology. *Computer Communications* 17 (2014), 3–15.

Cristina Cabanillas, Anne Baumgrass, Jan Mendling, Patricia Rogetzer, and Bruno Bellovoda. 2013. Towards the enhancement of business process monitoring for complex logistics chains. In *Proceedings of the 2013 International Conference on Business Process Management (Lecture Notes in Business Information Processing - Business Process Management Workshops)*, Niels Lohmann, Minseok Song, and Petia Wohed (Eds.), Vol. 171. Springer International Publishing, Switzerland, 305–317.

Zhen Cao, Jürgen Fitschen, and Panagiotis Papadimitriou. 2015. Social Wi-Fi: Hotspot sharing with online friends. In *Proceedings of the 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC'15)*. IEEE, 2132–2137.

Alexandru Caracaş and Thorsten Kramp. 2011. On the expressiveness of BPMN for modeling wireless sensor networks applications. In *Business Process Model and Notation*, Remco Dijkman, Jrg Hofstetter, and Jana Koehler (Eds.). Lecture Notes in Business Information Processing, Vol. 95. Springer, Berlin, 16–30.

Alexandru Mircea Caracaş. 2012. *Modeling, Compiling, and Efficiently Executing Business Processes on Resource-constrained Wireless Sensor Networks*. Ph.D. Dissertation. Diss., Eidgenössische Technische Hochschule ETH Zürich, Nr. 20220, 2012.

Berardina De Carolis, Stefano Ferilli, and Domenico Redavid. 2015. Incremental learning of daily routines as workflows in a smart home environment. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 4, 4 (2015), 20.

Fabio Casati, Florian Daniel, Guenadi Dantchev, Joakim Eriksson, Niclas Finne, Stamatis Karnouskos, Patricio Moreno Montera, Luca Mottola, Felix Jonathan Oppermann, Gian Pietro Picco, Antonio Quartulli, Kay Römer, Patrik Spiess, Stefano Tranquillini, and Thiemo Voigt. 2012. Towards business processes orchestrating the physical enterprise with wireless sensor networks. In *Proceedings of the 34th International Conference on Software Engineering (ICSE'12)*. IEEE, 1357–1360.

Angelo P. Castellani, Moreno Dissegna, Nicola Bui, and Michele Zorzi. 2012. WebIoT: A web application framework for the Internet of things. In *Proceedings of the 2012 Wireless Communications and Networking Conference Workshops (WCNCW'12)*. IEEE, 202–207.

Stuart Chandler. 2015. Unlocking the power of the Internet of things through BPM. In *BPM Everywhere: Internet of Things, Process of Everything*, Layna Fischer and Future Strategies (Eds.). Future Strategies, Lighthouse Point, FL, 183–189.

Chii Chang and Sea Ling. 2008. Towards a context-aware solution for device failures in service-oriented workflow. In *Proceedings of the 10th International Conference on Information Integration and Web-Based Applications & Services*. ACM, New York, NY, 77–83.

Chii Chang, Seng W. Loke, Hai Dong, Flora Salim, Satish N. Srirama, Mohan Liyanage, and Sea Ling. 2015. An energy-efficient inter-organizational wireless sensor data collection framework. In *Proceedings of the 2015 IEEE International Conference on Web Services (ICWS'15)*. IEEE, 639–646.

Chii Chang, Satish Narayana Srirama, and Sea Ling. 2012a. An adaptive mediation framework for mobile P2P social content sharing. In *Proceedings of the 10th International Conference on Service Oriented Computing*. Springer, Berlin, 374–388.

Chii Chang, S. N. Srirama, and J. Mass. 2015b. A middleware for discovering proximity-based service-oriented industrial Internet of things. In *Proceedings of the 2015 IEEE International Conference on Services Computing (SCC'15)*. IEEE, 130–137.

Chii Chang, Satish Narayana Srirama, and Sea Ling. 2012b. An adaptive mediation framework for mobile P2P social content sharing. In *Proceedings of the 10th International Conference on Service-Oriented Computing (ICSOC'12) (Lecture Notes in Computer Science)*, Chengfei Liu, Heiko Ludwig, Farouk Toumani, and Qi Yu (Eds.), Vol. 7636. Springer, Berlin, 374–388.

Chii Chang, Satish Narayana Srirama, and Sea Ling. 2014a. SPiCa: A social private cloud computing application framework. In *Proceedings of the 13th International Conference on Mobile and Ubiquitous Multimedia*. ACM, New York, NY, 30–39.

Chii Chang, Satish Narayana Srirama, and Sea Ling. 2014b. Towards an adaptive mediation framework for mobile social network in proximity. *Pervasive and Mobile Computing* 12 (2014), 179–196.

Chii Chang, S. N. Srirama, and M. Liyanage. 2015a. A service-oriented mobile cloud middleware framework for provisioning mobile sensing as a service. In *Proceedings of the 21st IEEE International Conference on Parallel and Distributed Systems (ICPADS'15)*. IEEE, 124–131.

Moumena A. Chaqfeh and Nader Mohamed. 2012. Challenges in middleware solutions for the Internet of things. In *Proceedings of the International Conference on Collaboration Technologies and Systems (CTS'12)*. IEEE, 21–26.

Wei-Chih Chen and Chi-Sheng Shih. 2011. ERWF: Embedded real-time workflow engine for user-centric cyber-physical systems. In *Proceedings of the 17th IEEE International Conference on Parallel and Distributed Systems (ICPADS'11)*. IEEE, 713–720.

Ting-Shuo Chou, S. Y. Chang, Y. F. Lu, Y. C. Wang, M. K. Ouyang, C. S. Shih, T. W. Kuo, J. S. Hu, and J. W. S. Liu. 2009. EMWF for flexible automation and assistive devices. In *Proceedings of the 15th IEEE Real-Time and Embedded Technology and Applications Symposium*. IEEE, 243–252.

Marco Conti, Sajal K. Das, Chatschik Bisdikian, Mohan Kumar, Lionel M. Ni, Andrea Passarella, George Roussos, Gerhard Trster, Gene Tsudik, and Franco Zambonelli. 2012. Looking ahead in pervasive computing: Challenges and opportunities in the era of cyberphysical convergence. *Pervasive and Mobile Computing* 8, 1 (2012), 2–21.

Li Da Xu, Wu He, and Shancang Li. 2014. Internet of things in industries: A survey. *IEEE Transactions on Industrial Informatics* 10, 4 (2014), 2233–2243.

Kashif Dar, Amir Taherkordi, Harun Baraki, Frank Eliassen, and Kurt Geihs. 2015. A resource oriented integration architecture for the Internet of things: A business process perspective. *Pervasive and Mobile Computing* 20 (2015), 145–159.

Kashif Dar, Amirhosein Taherkordi, Romain Rouvoy, and Frank Eliassen. 2011. Adaptable service composition for very-large-scale Internet of things systems. In *Proceedings of the 8th Middleware Doctoral Symposium (MDS'11)*. ACM, New York, NY, Article 2, 6 pages.

Anind K. Dey. 2001. Understanding and using context. *Personal and Ubiquitous Computing* 5, 1 (2001), 4–7.

Salvatore Distefano, Giovanni Merlino, and Antonio Puliafito. 2015. A utility paradigm for IoT: The sensing cloud. *Pervasive and Mobile Computing* 20 (2015), 127–144.

Mari Carmen Domingo. 2012. An overview of the Internet of things for people with disabilities. *Journal of Network and Computer Applications* 35, 2 (2012), 584–596.

Dan Driscoll, Antoine Mensch, Toby Nixon, and Alain Regnier. 2009. Devices profile for web services version 1.1. *OASIS Standard* 1 (2009), 4.

Marlon Dumas, Marcello La Rosa, Jan Mendling, and Hajo A. Reijers. 2013. *Fundamentals of Business Process Management*. Springer-Verlag, Berlin.

Dave Evans. 2011. The Internet of things: How the next evolution of the Internet is changing everything. *CISCO White Paper* 1 (2011), 14.

Sergei Evdokimov, Benjamin Fabian, Steffen Kunz, and Nina Schoenemann. 2010. Comparison of discovery service architectures for the Internet of things. In *Proceedings of the 2010 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*. IEEE, 237–244.

Nick Feamster, Lixin Gao, and Jennifer Rexford. 2007. How to lease the Internet in your spare time. *ACM SIGCOMM Computer Communication Review* 37, 1 (2007), 61–64.

Niroshinie Fernando, Seng W. Loke, and Wenny Rahayu. 2013. Mobile cloud computing: A survey. *Future Generation Computer Systems* 29, 1 (2013), 84–106. Including Special section: AIRCC-NetCoM 2009 and Special section: Clouds and Service-Oriented Architectures.

Layna Fischer (Ed.). 2015. *BPM Everywhere: Internet of Things, Process of Everything*. Future Strategies, Lighthouse Point, FL.

Pantelis A. Frangoudis, George C. Polyzos, and Vasileios P. Kemerlis. 2011. Wireless community networks: An alternative approach for nomadic broadband network access. *IEEE Communications Magazine* 49, 5 (2011), 206–213.

Karol Furdik, Gabriel Lukac, Tomas Sabol, and Peter Kostelnik. 2013. The network architecture designed for an adaptable IoT-based smart office solution. *International Journal of Computer Networks and Communications Security* 1, 6 (2013), 216–224.

Bo Gao, Ligang He, Limin Liu, Kenli Li, and Stephen A. Jarvis. 2012. From mobiles to clouds: Developing energy-aware offloading strategies for workflows. In *Proceedings of the 13th ACM/IEEE International Conference on Grid Computing*. IEEE, 139–146.

Krishna C. Garikipati and Kang G. Shin. 2013. Distributed association control in shared wireless networks. In *Proceedings of the 10th IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks*. IEEE, 362–370.

Kerstin Gerke, Alexander Claus, and Jan Mendling. 2009. Process mining of RFID-based supply chains. In *Proceedings of the 2009 IEEE Conference on Commerce and Enterprise Computing*. IEEE, 285–292.

Pau Giner, Carlos Cetina, Joan Fons, and Vicente Pelechano. 2010. Developing mobile workflow support in the Internet of things. *IEEE Pervasive Computing* 9, 2 (April 2010), 18–26.

Nils Glombitza, Sebastian Ebers, Dennis Pfisterer, and Stefan Fischer. 2011. Using BPEL to realize business processes for an Internet of things. In *Ad-hoc, Mobile, and Wireless Networks*, Hannes Frey, Xu Li, and Stefan Ruehrup (Eds.). Lecture Notes in Computer Science, Vol. 6811. Springer, Berlin, 294–307.

Jorge Granjal, Edmundo Monteiro, and Jorge Sá Silva. 2015. Security in the integration of low-power wireless sensor networks with the Internet: A survey. *Ad Hoc Networks* 24 (2015), 264–287.

Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. 2013. Internet of things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems* 29, 7 (2013), 1645–1660.

Dominique Guinard, Vlad Trifa, and Erik Wilde. 2010. A resource oriented architecture for the web of things. In *Proceedings of the Internet of Things 2010 Conference (IoT'10)*. IEEE, 1–8.

Bin Guo, Daqing Zhang, Zhiwen Yu, Yunji Liang, Zhu Wang, and Xingshe Zhou. 2013. From the Internet of things to embedded intelligence. *World Wide Web* 16, 4 (2013), 399–420.

Gregory Hackmann, Mart Haitjema, Christopher Gill, and Gruia-Catalin Roman. 2006. Sliver: A BPEL workflow process execution engine for mobile devices. In *Service-Oriented Computing*, Asit Dan and Winfried Lamersdorf (Eds.). Lecture Notes in Computer Science, Vol. 4294. Springer, Berlin, 503–508.

Marc J. Hadley. 2006. *Web Application Description Language (WADL)*. Technical Report. Sun Microsystems, Mountain View, CA.

Constantin Houy, Peter Fettke, Peter Loos, Wil M. P. Aalst, and John Krogstie. 2010. BPM-in-the-large – towards a higher level of abstraction in business process management. In *E-Government, E-Services and Global Processes: Joint IFIP TC 8 and TC 6 International Conferences (EGES'10 and GISP'10), Held as Part of WCC 2010*. Springer, Berlin, 233–244.

S. M. Riazul Islam, Daehan Kwak, Md. Humaun Kabir, Mahmud Hossain, and Kyung-Sup Kwak. 2015. The Internet of things for health care: A comprehensive survey. *IEEE Access* 3 (2015), 678–708.

Valrie Issarny, Nikolaos Georgantas, Sara Hachem, Apostolos Zarras, Panos Vassiliadist, Marco Autili, MarcoAurlio Gerosa, and AmiraBen Hamida. 2011. Service-oriented middleware for the future Internet: State of the art and research directions. *Journal of Internet Services and Applications* 2, 1 (2011), 23–45.

Zakwan Jaroucheh, Xiaodong Liu, and Sally Smith. 2011. Recognize contextual situation in pervasive environments using process mining techniques. *Journal of Ambient Intelligence and Humanized Computing* 2, 1 (2011), 53–69.

Cullen Jennings, Jari Arkko, and Zach Shelby. 2013. Media Types for Sensor Markup Language (SENML) draft-jennings-senml-10. Retrieved from https://tools.ietf.org/html/draft-jennings-senml-10.

Diane Jordan, John Evdemon, Alexandre Alves, Assaf Arkin, Sid Askary, Charlton Barreto, Ben Bloch, Francisco Curbera, Mark Ford, Yaron Goland, Alejandro Guízar, Neelakantan Kartha, Canyang Kevin Liu, Rania Khalaf, Dieter König, Mike Marin, Vinkesh Mehta, Satish Thatte, Danny van der Rijn, Prasad Yendluri, and Alex Yiu. 2007. Web services business process execution language version 2.0. *OASIS Standard* 11, 120 (2007), 5.

Jae-Yoon Jung, Pablo Rosales, Kyuhyup Oh, and Kyuri Kim. 2012. edUFlow: An event-driven ubiquitous flow management system. In *Business Process Management Workshops*, Florian Daniel, Kamel Barkaoui, and Schahram Dustdar (Eds.). Lecture Notes in Business Information Processing, Vol. 99. Springer, Berlin, 427–432.

Percy J. Igei Kaneshiro, Pari Delir Haghighi, and Sea Ling. 2014. Situation-aware adaptation to optimise energy consumption in intelligent buildings using coloured petri nets. In *Proceedings of the IEEE 9th Conference on Industrial Electronics and Applications (ICIEA'14)*. IEEE, 231–236.

Nikos Kefalakis, John Soldatos, Nikolaos Konstantinou, and Neeli R. Prasad. 2011. APDL: A reference XML schema for process-centered definition of RFID solutions. *Journal of Systems and Software* 84, 7 (2011), 1244–1259.

Ryan K. L. Ko, Stephen S. G. Lee, and Eng Wah Lee. 2009. Business process management (BPM) standards: A survey. *Business Process Management Journal* 15, 5 (2009), 744–791.

Gerd Kortuem, Fahim Kawsar, Vasughi Sundramoorthy, and Daniel Fitton. 2010. Smart objects as building blocks for the Internet of things. *IEEE Internet Computing* 14, 1 (Jan. 2010), 44–51.

Pavel Kucherbaev, Stefano Tranquillini, Florian Daniel, Fabio Casati, Maurizio Marchese, Marco Brambilla, and Piero Fraternali. 2013. Business processes for the crowd computer. In *Proceedings of the 2013 Business Process Management Workshops*. Springer, Berlin, 256–267.

Mohan Liyanage, Chii Chang, and Satish Narayana Srirama. 2015. Lightweight mobile web service provisioning for sensor mediation. In *Proceedings of the 2015 IEEE International Conference on Mobile Services (MS'15)*. IEEE, New York, 57–64.

Seng Wai Loke. 2003. Service-oriented device ecology workflows. In *Proceedings of the 1st International Conference on Service-Oriented Computing (ICSOC'03) (Lecture Notes in Computer Science)*, Maria E. Orlowska, Sanjiva Weerawarana, Michael P. Papazoglou, and Jian Yang (Eds.), Vol. 2910. Springer, Berlin, 559–574.

Seng Wai Loke. 2015. On crowdsourcing information maps: Cornucopia of the commons for the city. In *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers (UbiComp/ISWC'15 Adjunct)*. ACM, New York, NY, 1527–1533.

Seng Wai Loke, Keegan Napier, Abdulaziz Alali, Niroshinie Fernando, and Wenny Rahayu. 2015. Mobile computations with surrounding devices: Proximity sensing and multilayered work stealing. *ACM Transactions on Embedded Computing Systems (TECS)* 14, 2, Article 22 (Feb. 2015), 25 pages.

Denise Lund, Carrie MacGillivray, Vernon Turner, and Mario Morales. 2014. *Worldwide and Regional Internet of Things (IoT) 2014–2020 Forecast: A Virtuous Circle of Proven Value and Demand*. Technical Report. IDC.

Luca Mainetti, Luigi Patrono, and Antonio Vilei. 2011. Evolution of wireless sensor networks towards the Internet of things: A survey. In *Proceedings of the 19th International Conference onSoftware, Telecommunications and Computer Networks (SoftCOM'11)*. IEEE, 1–6.

James Manyika, Michael Chui, Jacques Bughin, Richard Dobbs, Peter Bisson, and Alex Marrs. 2013. *Disruptive Technologies: Advances That Will Transform Life, Business, and the Global Economy*. Technical Report. Chrysalix.

Michael J. Martin. 2015. Cloud, Fog, and Now, Mist Computing. https://www.linkedin.com/pulse/cloud-computing-fog-now-mist-martin-ma-mba-med-gdm-scpm-pmp.

Ibrahim Mashal, Osama Alsaryrah, Tein-Yaw Chung, Cheng-Zen Yang, Wen-Hsing Kuo, and Dharma P. Agrawal. 2015. Choices for interaction with things on Internet and underlying issues. *Ad Hoc Networks* 28 (2015), 68–90.

Sonja Meyer, Andreas Ruppen, and Lorenz Hilty. 2015. The things of the Internet of things in BPMN. In *Proceedings of the 2015 Advanced Information Systems Engineering Workshops*. Springer International Publishing, Switzerland, 285–297.

Sonja Meyer, Andreas Ruppen, and Carsten Magerkurth. 2013. Internet of things-aware process modeling: Integrating IoT devices as business process resources. In *Proceedings of the Advanced Information Systems Engineering: 25th International Conference (CAiSE'13)*. Springer, Berlin, 84–98.

Sonja Meyer, Klaus Sperner, Carsten Magerkurth, and Jacques Pasquier. 2011. Towards modeling real-world aware business processes. In *Proceedings of the 2nd International Workshop on Web of Things (WoT'11)*. ACM, New York, NY, Article 8, 6 pages.

Catherine A. Middleton and Amelia Bryne. 2011. An exploration of user-generated wireless broadband infrastructures in digital cities. *Telematics and Informatics* 28, 3 (2011), 163–175.

Peter Middleton, Peter Kjeldsen, and Jim Tully. 2013. *Forecast: The Internet of Things, Worldwide, 2013*. Technical Report. Gartner.

Daniele Miorandi, Sabrina Sicari, Francesco De Pellegrini, and Imrich Chlamtac. 2012. Internet of things: Vision, applications and research challenges. *Ad Hoc Networks* 10, 7 (2012), 1497–1516.

Jorge Munoz-Gama, Josep Carmona, and Wil M. P. Van Der Aalst. 2014. Single-entry single-exit decomposed conformance checking. *Information Systems* 46 (2014), 102–122.

Kim Thuat Nguyen, Maryline Laurent, and Nouha Oualha. 2015. Survey on secure communication protocols for the Internet of things. *Ad Hoc Networks* 32 (2015), 17–31.

Object Management Group. 2011. Business Process Model and Notation (BPMN) version 2.0.

Oracle. 2016. ORACLE SOA SUITE. Retrieved from http://www.forzaconsulting.eu/dienst/applicatie-integratie/oracle-soa-suite/.

Lasse Pajunen and Suresh Chande. 2007. Developing workflow engine for mobile devices. In *Proceedings of the 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC'07)*. IEEE, 279–279.

Maria Rita Palattella, Nicola Accettura, Xavier Vilajosana, Thomas Watteyne, Luigi Alfredo Grieco, Gennaro Boggia, and Mischa Dohler. 2013. Standardized protocol stack for the Internet of (important) things. *IEEE Communications Surveys & Tutorials* 15, 3 (2013), 1389–1406.

Milan Patel, Yunchao Hu, Patrice Hédé, Jerome Joubert, Chris Thornton, Brian Naughton, Julian Roldan Ramos, Caroline Chan, Valerie Young, Soo Jin Tan, Daniel Lynch, Nurit Sprecher, Torsten Musiol, Carlos Manzanares, Uwe Rauschenbach, Sadayuki Abeta, Lan Chen, Kenji Shimizu, Adrian Neal, Peter Cosimini, Adam Pollard, and Guenter Klas. 2014. *Mobile-Edge Computing Introductory Technical White Paper*. Technical Report. ETSI.

Tao Peng, Marco Ronchetti, Jovan Stevovic, Annamaria Chiasera, and Giampaolo Armellin. 2014. Business process assignment and execution from cloud to mobile. In *Proceedings of the 2014 Business Process Management Workshops*. Springer International Publishing, Switzerland, 264–276.

Paolo Pileggi, Alejandro Rivero-Rodriguez, and Ossi Nykanen. 2015a. Towards traditional simulation models of context using process mining. In *Proceedings of the 7th International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN'15)*. IEEE, 70–75.

Paolo Pileggi, Alejandro Rivero-Rodriquez, and Ossi Nykänen. 2015b. Using context overlays to analyse the role of a priori information with process mining. In *Proceedings of the 9th Annual IEEE International Systems Conference (SysCon'15)*. IEEE, 639–644.

Jurgo S. Preden, Kalle Tammemae, Axel Jantsch, Mairo Leier, Andri Riid, and Emine Calis. 2015. The benefits of self-awareness and attention in fog and mist computing. *Computer* 48, 7 (2015), 37–45.

Rüdiger Pryss, Manfred Reichert, Alexander Bachmeier, and Johann Albach. 2015. BPM to go: Supporting business processes in a mobile and sensing world. In *BPM Everywhere: Internet of Things, Process of Everything*, Layna Fischer (Ed.). Future Strategies, FL, 167–182.

Rüdiger Pryss, Julian Tiedeken, Ulrich Kreher, and Manfred Reichert. 2011. Towards flexible process support on mobile devices. In *Information Systems Evolution*, Pnina Soffer and Erik Proper (Eds.). Lecture Notes in Business Information Processing, Vol. 72. Springer, Berlin, 150–165.

Petri Pulli, Olli Martikainen, Ye Zhang, Valeriy Naumov, Zeeshan Asghar, and Antti Pitkänen. 2011. Augmented processes: A case study in healthcare. In *Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies*. ACM, New York, NY, 137:1–137:6.

Muhannad Quwaider and Yaser Jararweh. 2016. A cloud supported model for efficient community health awareness. *Pervasive and Mobile Computing* 28 (2016), 35–50.

Rodrigo Roman, Jianying Zhou, and Javier Lopez. 2013. On the features and challenges of security and privacy in distributed Internet of things. *Computer Networks* 57, 10 (2013), 2266–2279.

Flora Salim and Usman Haque. 2015. Urban computing in the wild: A survey on large scale participation and citizen engagement with ubiquitous computing, cyber physical systems, and Internet of things. *International Journal of Human-Computer Studies* 81 (2015), 31–48.

Mahadev Satyanarayanan, Paramvir Bahl, Ramon Caceres, and Nigel Davies. 2009. The case for VM-based cloudlets in mobile computing. *IEEE Pervasive Computing* 8, 4 (Oct. 2009), 14–23.

Stefan Schulte, Philipp Hoenisch, Christoph Hochreiner, Schahram Dustdar, Matthias Klusch, and Dieter Schuller. 2014. Towards process support for cloud manufacturing. In *Proceedings of the 18th IEEE International Enterprise Distributed Object Computing Conference (EDOC'14)*. IEEE, 142–149.

Stefan Schulte, Christian Janiesch, Srikumar Venugopal, Ingo Weber, and Philipp Hoenisch. 2015. Elastic business process management: State of the art and open challenges for BPM in the cloud. *Future Generation Computer Systems* 46 (2015), 36–50.

Rohan Sen, Gruia-Catalin Roman, and Christopher Gill. 2008. CiAN: A workflow engine for MANETs. In *Coordination Models and Languages*, Doug Lea and Gianluigi Zavattaro (Eds.). Lecture Notes in Computer Science, Vol. 5052. Springer, Berlin, 280–295.

Zach Shelby and Carsten Bormann. 2011. *6LoWPAN: The Wireless Embedded Internet*. Vol. 43. John Wiley & Sons, United Kingdom.

Quan Z. Sheng, Xiaoqiang Qiao, Athanasios V. Vasilakos, Claudia Szabo, Scott Bourne, and Xiaofei Xu. 2014. Web services composition: A decades overview. *Information Sciences* 280 (2014), 218–238.

Zhengguo Sheng, Shusen Yang, Yifan Yu, Athanasios Vasilakos, J. Mccann, and Kin Leung. 2013. A survey on the ietf protocol suite for the Internet of things: Standards, challenges, and opportunities. *IEEE Wireless Communications* 20, 6 (2013), 91–98.

Sabrina Sicari, Alessandra Rizzardi, Luigi Alfredo Grieco, and Alberto Coen-Porisini. 2015. Security, privacy and trust in Internet of things: The road ahead. *Computer Networks* 76 (2015), 146–164.

Ricardo Silva, Jorge Sá Silva, and Fernando Boavida. 2014. Mobility in wireless sensor networks–survey and proposal. *Computer Communications* 52 (2014), 1–20.

Sucha Smanchat, Sea Ling, and Maria Indrawan. 2008. A survey on context-aware workflow adaptations. In *Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia*. ACM, New York, NY, 414–417.

Rute Sofia and Paulo Mendes. 2008. User-provided networks: Consumer as provider. *IEEE Communications Magazine* 46, 12 (2008), 86–91.

Klaus Sperner, Sonja Meyer, and Carsten Magerkurth. 2011. Introducing entity-based concepts to business process modeling. In *Proceedings of the Business Process Model and Notation: 3rd International Workshop (BPMN'11)*. Springer, Berlin, 166–171.

Satish Narayana Srirama, Matthias Jarke, and Wolfgang Prinz. 2006. Mobile web service provisioning. In *Proceedings of the 2006 Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT-ICIW'06)*. IEEE, 120–125.

C. Timurhan Sungur, Patrik Spiess, Nina Oertel, and Oliver Kopp. 2013. Extending BPMN for wireless sensor networks. In *Proceedings of the 15th IEEE Conference on Business Informatics (CBI'13)*. IEEE, 109–116.

Feilong Tang, Minyi Guo, Mianxiong Dong, Minglu Li, and Hu Guan. 2008. Towards context-aware workflow management for ubiquitous computing. In *Proceedings of the 2008 International Conference on Embedded Software and Systems (ICESS'08)*. IEEE, 221–228.

Thiago Teixeira, Sara Hachem, Valérie Issarny, and Nikolaos Georgantas. 2011. Service oriented middleware for the Internet of things: A perspective. In *Towards a Service-Based Internet*, Witold Abramowicz, IgnacioM. Llorente, Mike Surridge, Andrea Zisman, and Julien Vayssire (Eds.). Lecture Notes in Computer Science, Vol. 6994. Springer, Berlin, 220–229.

Robert A. Thacker, Kevin R. Jones, Chris J. Myers, and Hao Zheng. 2010. Automatic abstraction for verification of cyber-physical systems. In *Proceedings of the 1st ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS'10)*. ACM, New York, NY, 12–21.

Anne-Sophie Tonneau, Nathalie Mitton, and Julien Vandaele. 2015. How to choose an experimentation platform for wireless sensor networks? A survey on static and mobile wireless sensor network experimentation facilities. *Ad Hoc Networks* 30 (2015), 115–127.

Stefano Tranquillini, Florian Daniel, Pavel Kucherbaev, and Fabio Casati. 2015. BPMN task instance streaming for efficient micro-task crowdsourcing processes. In *Business Process Management*, Hamid Reza Motahari-Nezhad, Jan Recker, and Matthias Weidlich (Eds.). Lecture Notes in Computer Science, Vol. 9253. Springer International Publishing, Switzerland, 333–349.

Stefano Tranquillini, Patrik Spieß, Florian Daniel, Stamatis Karnouskos, Fabio Casati, Nina Oertel, Luca Mottola, Felix Jonathan Oppermann, Gian Pietro Picco, Kay Römer, and Thiemo Voigt. 2012. Process-based design and integration of wireless sensor network applications. In *Business Process Management*. Springer, Berlin, 134–149.

Chun-Wei Tsai, Chin-Feng Lai, Ming-Chao Chiang, and Laurence T. Yang. 2014. Data mining for Internet of things: A survey. *IEEE Communications Surveys & Tutorials* 16, 1 (2014), 77–97.

Wil M. P. van der Aalst. 2013. Business process management: A comprehensive survey. *ISRN Software Engineering* 2013 (2013), 37.

Wil M. P. van der Aalst. 2015. Extracting event data from databases to unleash process mining. In *BPM - Driving Innovation in a Digital World*. Springer International Publishing, Switzerland, 105–128.

Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, and Mathias Weske. 2003. Business process management: A survey. In *Business Process Management*, Wil M. P. van der Aalst and Mathias Weske (Eds.). Lecture Notes in Computer Science, Vol. 2678. Springer, Berlin, 1–12.

Wil M. P. van der Aalst and Mathias Weske. 2001. The P2P approach to interorganizational workflows. In *Proceedings of Advanced Information Systems Engineering: 13th International Conference (CAiSE'01)*. Springer, Berlin, 140–156.

Ovidiu Vermesan, Peter Friess, Patrick Guillemin, Sergio Gusmeroli, Harald Sundmaeker, Alessandro Bassi, Ignacio Soler Jubert, Margaretha Mazura, Mark Harrison, Markus Eisenhauer, and Pat Doody. 2011. Internet of things strategic research roadmap. In *Internet of Things - Global Technological and Societal Trends from Smart Environments and Spaces to Green Ict*. River Publishers, 9–52.

Berta Carballido Villaverde, Rodolfo De Paz Alberola, Antonio J. Jara, Szymon Fedor, Sajal K. Das, and Dirk Pesch. 2014. Service discovery protocols for constrained machine-to-machine communications. *IEEE Communications Surveys & Tutorials* 16, 1 (2014), 41–60.

Berta Carballido Villaverde, Dirk Pesch, Rodolfo De Paz Alberola, Szymon Fedor, and Menouer Boubekeur. 2012. Constrained application protocol for low power embedded networks: A survey. In *Proceedings of the 6th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS'12)*. IEEE, 702–707.

VITAL-IoT. 2016. EU FP7-ICT VITAL Project. Retrieved from http://vital-iot.eu/.

Fei-Yue Wang, Kathleen M. Carley, Daniel Zeng, and Wenji Mao. 2007. Social computing: From social informatics to social intelligence. *IEEE Intelligent Systems* 22, 2 (March 2007), 79–83.

Mathias Weske, Wil M. P. van der Aalst, and H. M. W. Verbeek. 2004. Advances in business process management. *Data & Knowledge Engineering* 50, 1 (2004), 1–8.

Workflow Management Coalition. 2012. XML Process Definition Language version 2.2. The Workflow Management Coalition Specification. August 30, 2012.

WSO2. 2016. [WSO2Con EU 2016] WSO2 IoT Server: Your Foundation for the Internet of Things. Retrieved from http://wso2.com/library/conference/2016/06/wso2con-europe-2016-wso2-iot-server-your-foundation-for-the-internet-of-things/.

Zhenyu Wu, Timo Itälä, Tingan Tang, Chunhong Zhang, Yang Ji, Matti Hämäläinen, and Yunjie Liu. 2012. Gateway as a service: A cloud computing framework for web of things. In *Proceedings of the 19th International Conference on Telecommunications (ICT'12)*. IEEE, 1–6.

Zhiqiang Yan, Remco Dijkman, and Paul Grefen. 2012. Business process model repositories–framework and survey. *Information and Software Technology* 54, 4 (2012), 380–395.

Zheng Yan, Peng Zhang, and Athanasios V. Vasilakos. 2014. A survey on trust management for Internet of things. *Journal of Network and Computer Applications* 42 (2014), 120–134.

Chao Yang, Nengcheng Chen, and Liping Di. 2012. RESTFul based heterogeneous geoprocessing workflow interoperation for sensor web service. *Computers & Geosciences* 47 (2012), 102–110.

Alaaeddine Yousfi, Christine Bauer, Rajaa Saidi, and Anind K. Dey. 2016. uBPMN: A BPMN extension for modeling ubiquitous business processes. *Information and Software Technology* 74 (2016), 55–68.

Alaaeddine Yousfi, Adrian de Freitas, Anind K. Dey, and Rajaa Saidi. 2015. The use of ubiquitous computing for business process improvement. *IEEE Transactions on Services Computing* 9, 4 (2015), 621–632.

Andrea Zanella, Nicola Bui, Angelo Castellani, Lorenzo Vangelista, and Michele Zorzi. 2014. Internet of things for smart cities. *IEEE Internet of Things Journal* 1, 1 (2014), 22–32.

Deze Zeng, Song Guo, and Zixue Cheng. 2011. The web of things: A survey. *Journal of Communications* 6, 6 (2011), 424–438.

Ye Zhang, Olli Martikainen, Petri Pulli, and Valeriy Naumov. 2012. *Developing a Real-Time Process Data Acquisition System for Automatic Process Measurement*. Springer, Berlin, 115–124.

Michele Zorzi, Alexander Gluhak, Sebastian Lange, and Alessandro Bassi. 2010. From today's intranet of things to a future Internet of things: A wireless-and mobility-related view. *IEEE Wireless Communications* 17, 6 (2010), 44–51.

Received March 2016; revised August 2016; accepted October 2016