International Journal of Engineering & Technology, 7 (4.39) (2018) 555-560

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Development of Task Deployment Tool for Operating IoT Devices

Lei Hang[1], Do-Hyeun Kim*[2]

[1,2]Computer Engineering Department, Jeju National University, Jeju, 63243 Korea
**Corresponding author E-mail: kimdh@jejunu.ac.kr*

Abstract

Background/Objectives: Business processes utilize IoT data to interact with the data in physical world, to make informed decisions, improve their execution, and adjust to set context changes.
Methods/Statistical analysis: A novel IoT task deployment tool is proposed in this paper for efficiently executing and allocating the business process model to real IoT devices. The designed tool can extract the operational sequence from the task editor and interact directly with remote IoT devices to perform defined tasks.
Findings:An application in a mobile device is implemented to validate the feasibility of the proposed approach, in which CoAP protocol is used for the communication. The results show that the proposed task deployment tool can retrieve and parse tasks efficiently.
Improvements/Applications: In future, we will conduct real experiment with more IoT devices to verify the proposed model for task deployment.

Keywords: *Internet of Things, Business Process Modeling, Task Deployment, CoAP, Mobile.*

1. Introduction

The Internet of Things (IoT) is turning into an inexorably critical subject of congress both on and off the working space [1]. IoT guarantees to empower establishing various applications in areas, for example, building and home automation, smart environment, intelligent agriculture, smart transportation, and medical services [2]. Real-world devices will have the capacity to offer their functionality via SOAP-based web services (WS-*) or RESTful APIs, enabling other components to interact with them dynamically, especially in IoT [3]. A service demonstrates the activity that can be performed by a sensor node so as to be asked for meeting the involved task, which is related with the atomic service type like temperature, humidity, light, etc.[4]. The WSN applications comprise of one or more tasks, and accordingly an application might require a complex service composed of multiple atomic services. To be more precise, each of these tasks is the finest-grained and non-divisible element to establish the WSN application.
Business Process Modeling Notation (BPMN) [5] is a standardized set of notations that in a specific sequence produces a service for non-technical users to express their requirements, hence it can provide a better DIY programming platform for IoT related applications. Itcan be visualized as a collection of connected activities or tasks to achieve an authoritative objective, once finished. As of late, with the fast increasement in computing power, constrained IoT devices can participant in the execution of business logic, and in such a manner they can collect or filter the contents of data, and make processing decisions locally, by executing the business logic partially.
In our previous study, a BPMN-based task editor [6] with the intuitive drag-and-drop approach was proposed, which allowed the general public without any programming experiences to customize the services offered by remote and constrained IoT devices. The editor offers miscellaneous graphical representations of the task objects enabling general users to define the logic flow of an IoT application. This paper proposes a task deployment tool to execute the business logic and allocate IoT tasks to remote devices. The proposed deployment tool utilizes the CoAP Californium which facilitates the discovery of CoAP resources and supports libraries for generation of CoAP commands. It ingests the task profile designed from the task editor which is in the form of XML. For the execution of the business process model, the deployment tool is responsible to translate each task into RESTful web services and directly interact with remote IoT devices. The execution state or any data is returned to the deployment tool which further decides how to proceed with the further execution of the business process. In order to verify the feasibility of the designed system, a mobile application has been implemented over the proposed system. The test results denote that the proposed system is capable of evaluating the application logic of the current task and perform the further execution steps of the business process.
The remainder of this paper is structured as follows. Section 2 shortly introduces the basic principle of BPMN and gives an overview of some existing researches using business process. In Section 3, we present the system architecture of the proposed system as well as system operation and configuration. Section 4 details the implementation of the case study over the designed system and reports the experimental results with various snapshots. Section 5 demonstrates the performance of the proposed system by computing the execution time for three different business logic files with different steps. Finally, Section 6 discusses the conclusion and future work.

2. Related Work

Business process languages such as (WS-BPEL) or BPMN provides an abstraction level closer to the domain being specified. From the perspective of IoT implementations, the same definition of business process fits the processes or applications that run on top of an IoT implementation defining its behavior via the interactions of sensing and actuating agents in order to provide useful services to the end-users. The example of such an implementation would be a Smart Space, where the process running as the application would define the behavior of the Smart Space sensing (temperature, humidity, illumination, proximity etc.) and actuate (Air conditioning, heating, lightings, door locks etc.) agents based on pre-defined events (resident's arrival, time schedule, emergency state etc.).
As of late, there has been a broad proportion of research in the field of IoT behavior definition inside the business process. The authors in [7] integrate sensor networks into business processes by utilizing the BPEL language. As part of the CoBIs-project [8], a sensor network proposed to monitor the storage of chemical containers. The sensor network monitors the number of containers that are stored together as well as the combination of containers with different chemical materials. The concentration of this work is to integrate a sensor network application into existing business processes. In paper [9], a WS-BPEL model with context variables is extended to monitor IoT information as well as abstractes the set of operations interacted with IoT devices. The IoT-A project proposes some BPMN extensions to explicitly include IoT devices and their services in an IoT-aware process model, as well as some characteristics of IoT devices, such as uncertainty and availability [10]. The uBPMN project extends the BPMN by adding ubiquitous elements. New elements including a sensor, reader, collector, camera, and microphone, as well as an IoT-driven Data Object are defined as the BPMN task to represent the data transmitted from IoT devices [11]. A graphical user interface is proposed in GWELS [12] to design processes of the IoT application. The designed process can be automatically sent to IoT devices in the form of a sequence of operation calls which are web services provided by the IoT devices. It uses proprietary communication protocols to interact with IoT devices. IoT processes are provided as web services and, in this way, can also be integrated into business processes. A makeSense framework which extends the BPMN with attributes is proposed in [13]. A new intra-WSN participant is also designed, which enables IoT devices to execute some part of the process[14].
To the best of our knowledge, these systems mentioned above, are unable to define the behavior of IoT devices, they can only use services whose behavior is pre-defined. In our work, we utilize the standard BPMN to define all the business process, and IoT service is defined by a visual designer proposed in the previous study. The proposed deployment tool translates the business processes into RESTful web services, which can directly interact with IoT devices.

3. Proposed IoT Task Deployment Tool

The class diagram of the proposed task deployment tool is illustrated in Figure 1. Manager class defines the user interface to deal with the Human-Computer Interaction (HCI). Through this class,usercan choose BPMN file to load and confirm the business process intuitionally. All of the elements used in the business process is defined in the BpmnItem class. Parser class provides the functionalities to parse and extract business process items from the XML file. In addition, the Execution Engine is responsible for the deployment and allocation of the tasks created by the user. It creates a sequence for the execution of individual tasks according to the connections among various process notations as part of the graphical model along with the location of each graphical item in the model. The user can then choose to deploy the model. For the execution of the deployed process, the execution engine is responsible to send the XML representation of each task to the concerned remote IoT resource. The execution state or any data is returned to the execution engine which further decides how to proceed with the execution and scheduling of the tasks using three subclasses (Task, Gateway and Script). Task class defines the order of execution among tasks. The conditional logic of the process is implemented using the Gateway class while processor-intensive tasks and remote communication tasks which are not suitable to be executed on the remote IoT resources are represented by the Script class. The Script class has been provided with a list of scripts from which the user can choose to manipulate or process the data.

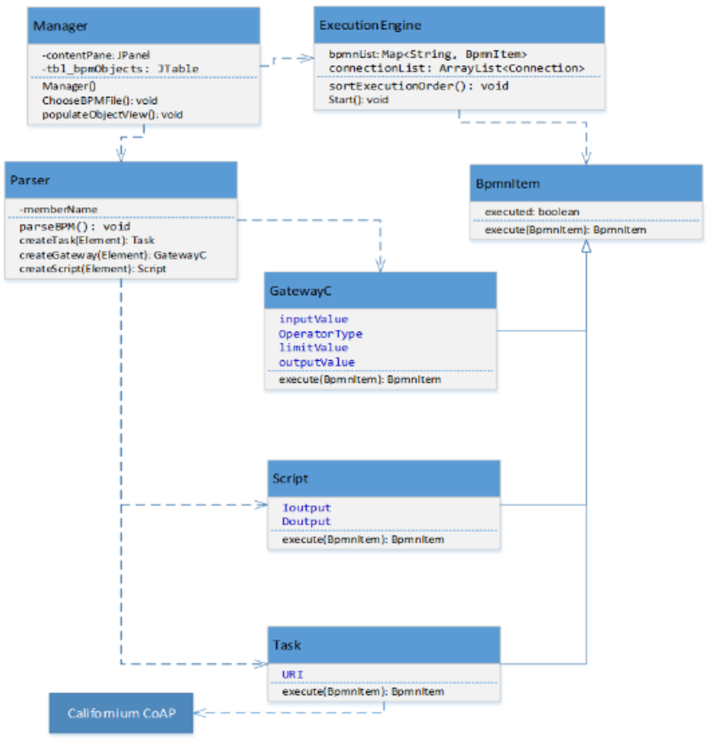

Manager
-contentPane: JPanel
-tbl_bpmObjects: JTable
Manager()
ChooseBPMFile(): void
populateObjectView(): void
ExecutionEngine
bpmnList: Map<String, BpmnItem>
connectionList: ArrayList<Connection>
sortExecutionOrder(): void
Start(): void
Parser
-memberName
parseBPM(): void
createTask(Element): Task
createGateway(Element): GatewayC
createScript(Element): Script
BpmnItem
executed: boolean
execute(BpmnItem): BpmnItem
GatewayC
inputValue
OperatorType
limitValue
outputValue
execute(BpmnItem): BpmnItem
Script
Ioutput
Doutput
execute(BpmnItem): BpmnItem
Task
URI
execute(BpmnItem): BpmnItem
Californium CoAP

Figure 1: Class Diagram of the Task Deployment Tool

Figure 2 illustrates the details of the operation of the proposed task deployment tool. The task deployment tool is responsible for the deployment and execution of the BPM models created by the users through the task editor. It presents a simple user interface which consists of a viewer for the users to view the components of the BPM that is being deployed. The viewer panel does not visualize the components of the BPM as visual notations rather it lists them in a sequential order of the textual description of these components as specified by the graphical model. This operation is performed by parsing the connections among various BPM notations as part of the graphical model along with the location of each graphical item in the model to create a sequence for the execution of individual tasks.

Once the task execution sequence is created, the list of the tasks as part of the BPM is shown to the user in the view panel. The user can then choose to deploy the model. For the execution of the deployed BPM, the execution engine is responsible to allocate tasks to the concerned remote IoT resource. The execution state or any data is returned to the execution engine which further decides how to proceed with the execution of the BPM. The BPMN Gateways provides branching and execution logic for the process and the BPMN Scripts provide functions such as data processing or network communications which are too costly for the remote IoT resources and thus are executed by the execution engine.

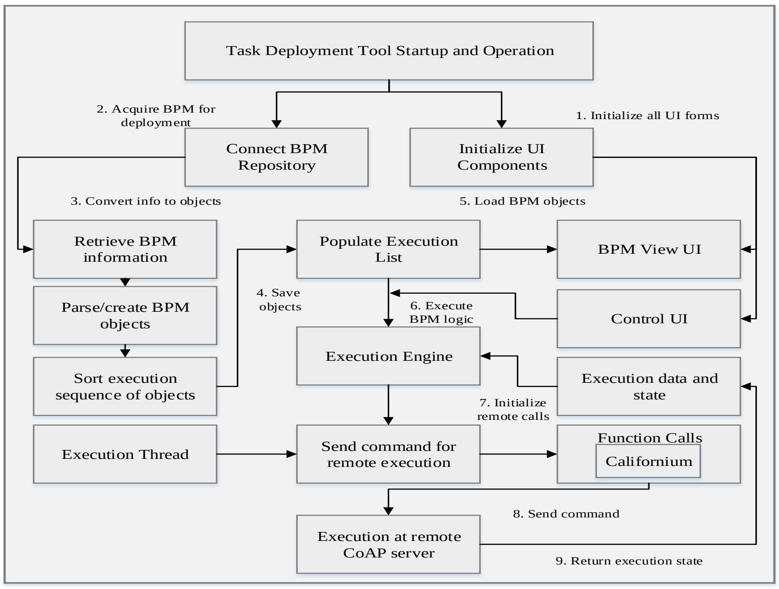


Figure 2: Operation Configuration of the Proposed Task Deployment Tool

The task deployment tool loads the BPM XML documents and parses the business process. Then it starts the process execution and sends a CoAP request using GET method to the touch resource belonged to the smart phone. The deployment manager receives the CoAP response from the phone and verifies the boolean value of the touch sensor. For example, the true value means the touch sensor is pressed by the user. The deployment manager initiates another request to the LED resource of the phone. The phone turns on the LED according to the request and sends back the response to the deployment manager, these operations are represented in Figure 3.

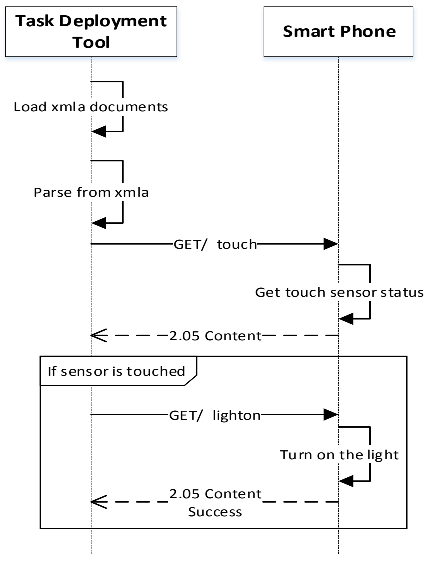


Figure 3: Sequence Diagram of the Case Study

The XML sample for the case study shown in Figure 4 representing tasks and other notations as DesignerItem objects. A DesignerItem tag in the figure completely represents the information encapsulated by a single BPM notation. In the figure first task represents a Task notation which encapsulates the complete information regarding a service object. The information includes the names of the input, output devices associated with the SO, the complete URIs of the services for both the devices and the operational conditions for the execution of the SO. The file also includes connection objects to keep track of the source and sink items in the model and hence helps in identifying the correct sequence and execution order of the process.

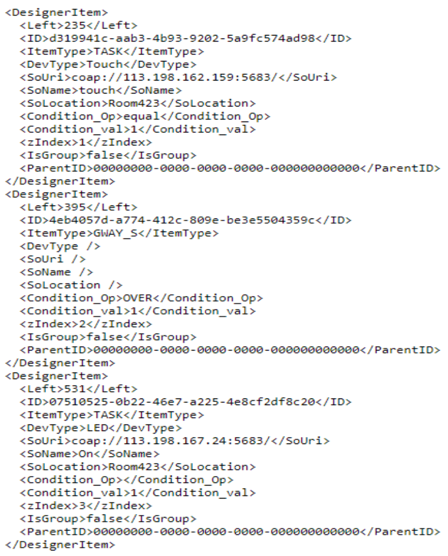

```xml
<DesignerItem>
  <Left>235</Left>
  <ID>d319941c-aab3-4b93-9202-5a9fc574ad98</ID>
  <ItemType>TASK</ItemType>
  <DevType>Touch</DevType>
  <SoUri>coap://113.198.162.159:5683/</SoUri>
  <SoName>touch</SoName>
  <SoLocation>Room423</SoLocation>
  <Condition_Op>equal</Condition_Op>
  <Condition_val>1</Condition_val>
  <zIndex>1</zIndex>
  <IsGroup>false</IsGroup>
  <ParentID>00000000-0000-0000-0000-000000000000</ParentID>
</DesignerItem>
<DesignerItem>
  <Left>395</Left>
  <ID>4eb4057d-a774-412c-809e-be3e5504359c</ID>
  <ItemType>GWAY_S</ItemType>
  <DevType />
  <SoUri />
  <SoName />
  <SoLocation />
  <Condition_Op>OVER</Condition_Op>
  <Condition_val>1</Condition_val>
  <zIndex>2</zIndex>
  <IsGroup>false</IsGroup>
  <ParentID>00000000-0000-0000-0000-000000000000</ParentID>
</DesignerItem>
<DesignerItem>
  <Left>531</Left>
  <ID>07510525-0b22-46e7-a225-4e8cf2df8c20</ID>
  <ItemType>TASK</ItemType>
  <DevType>LED</DevType>
  <SoUri>coap://113.198.167.24:5683/</SoUri>
  <SoName>On</SoName>
  <SoLocation>Room423</SoLocation>
  <Condition_Op></Condition_Op>
  <Condition_val>1</Condition_val>
  <zIndex>3</zIndex>
  <IsGroup>false</IsGroup>
  <ParentID>00000000-0000-0000-0000-000000000000</ParentID>
</DesignerItem>
```

Figure 4: XML Representation of Business Logic for the Case Study

4. Implementation Details

This section introduces the IDEs, hardware, and technologies used for the case study development. The use case consists of two main components as shown in Figure 3 so that two tables are made to illustrate the development environment for each module separately.
Table 1 represents the technology stacks used for developing the task deployment tool. The implementation is performed on Intel Core i3-3220 CPU at 3.30GHz with 12 GB memory and 64-bit operating system. This application is developed with the Eclipse Luna (version 2.3) in Java language. The Swing is a Java-written toolkit that is used to create window-based applications. In order to support the CoAP communication, Californium framework has been used.

Table 1: Development Environment of Task Deployment Tool

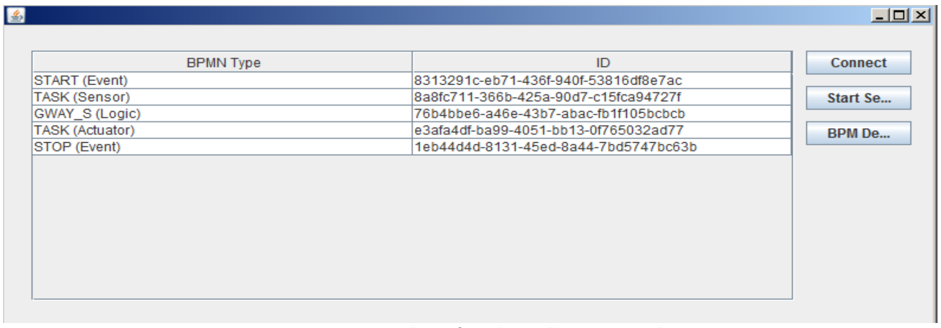
Task Deployment Tool	
CPU	Intel(R) Core i3-3220 3.30GHz
OS	Windows 7 Ultimate 64bits
Memory	12GB
Development Tool	Eclipse Luna (4.2)
Communication Protocol	CoAP Protocol
Language	Java
Library and Framework	CoAP Californium, Swing

Table 2 represents the development environment of the mobile application in smart phone. This application is developed with the Android Studio IDE (version 2.3) in Java and XML language. The application has been tested in the Vivo X5 Pro in Android 5.0 with API 23. CoAP Californium framework is used to implement the CoAP server and these mobile sensors such as LED and touch sensor are implemented into resources as part of the server, and each of them is assigned with a unique URI in order to be identified by the server. This mobile application listens for the request on the appointed port and performs the operation on the corresponding resources accordlingly.

Table 2: Development Environment of Mobile Application in Smart Phone

Mobile Application	
Development Tool	Android Studio 2.3
Communication Protocol	CoAP Protocol
Language	Java, XML
Library and Framework	CoAP Californium
Hardware	Vivo X5 Pro (Android 5.0, API 23)
Memory	2GB
Resources	LED, Touch sensor

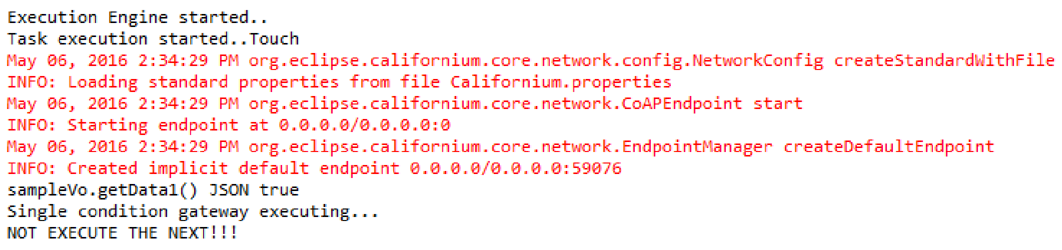
Task deployment tool provides the execution engine for the BPM created via the task editor. Figure 5 shows the main interface for the task deployment tool. For this specific prototype as the physical devices are CoAP based IoT resources in a mobile phone, the task deployment tool utilizes the Californium framework to communicate with the remote IoT resources in order to execute the process represented via the BPM. The interface is intentionally kept very simple with a viewer for the user to visualize the steps of the deployed BPM and a few controls to enable the user to load BPM from the repository and to control the execution of the BPM. As mentioned earlier, that a process model representing the interaction among various IoT resources is created by the user via the task editor and stored as an XML document. These XML documents can be loaded in the task deployment tool to be executed. The BPM in its XML format is loaded into the task deployment tool. The file is first parsed to extract all the executable entities as represented by the BPM notations. Based on the connection between the notations, the executable entities are sorted and sequenced so that the final execution of the process is in sync with the original graphical BPM created by the user. The execution steps are then displayed in the viewer part of the interface.



BPMN Type	ID
START (Event)	8313291c-eb71-436f-940f-53816df8e7ac
TASK (Sensor)	8a8fc711-366b-425a-90d7-c15fca94727f
GWAY_S (Logic)	76b4bbe6-a46e-43b7-abac-fb1f105bcbcb
TASK (Actuator)	e3afa4df-ba99-4051-bb13-0f765032ad77
STOP (Event)	1eb44d4d-8131-45ed-8a44-7bd5747bc63b

Figure 5: Snapshot of Task Deployment Tool

The entities are stored in a list which is sorted in accordance with the flow of the process model. The list is then iterated and each entity is executed based on its attributes and behavior. For task-related to remote IoT resources, the URI of the resource extracted from the XML representation is sent to the corresponding CoAP server. This transfer is done through a CoAP function call service of the task deployment tool.

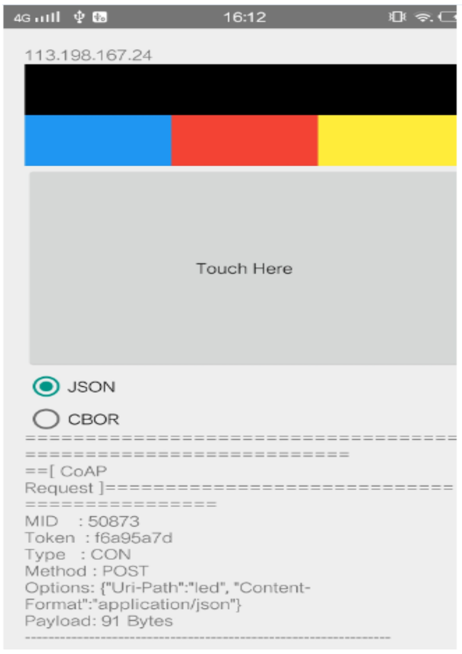


```
Execution Engine started..
Task execution started..Touch
May 06, 2016 2:34:29 PM org.eclipse.californium.core.network.config.NetworkConfig createStandardWithFile
INFO: Loading standard properties from file Californium.properties
May 06, 2016 2:34:29 PM org.eclipse.californium.core.network.CoAPEndpoint start
INFO: Starting endpoint at 0.0.0.0/0.0.0.0:0
May 06, 2016 2:34:29 PM org.eclipse.californium.core.network.EndpointManager createDefaultEndpoint
INFO: Created implicit default endpoint 0.0.0.0/0.0.0.0:59076
sampleVo.getData1() JSON true
Single condition gateway executing...
NOT EXECUTE THE NEXT!!!
```

Figure 6: Task Deployment Tool Execution Results

Once the CoAP server receives the request from the task deployment tool, the CoAP services, are executed using the Californium framework. The response based on the complete execution of the task is then sent back to the task deployment tool, where it is utilized to evaluate the conditional gateways or provided as inputs to the other BPM notations directly connected

with the specific task notation. Scripts implemented as part of the Deployment Manager are executed by the manager itself while the data is provided by other entities such as remote IoT resources. Figure 6 shows the execution of a business process model via the task deployment tool. The responses received from the touch resource has been shown before utilized to further execute the process. In this case, the LED will be turned on if the touch sensor is pressed according to the defined business process in Figure 4.
Figure 7 represents the snapshot of the execution result for the Led resource in the phone. The three color LED is turned on as the touch sensor is pressed. Then the LED resource returns a CoAP response which indicates the task is successfully performed.



```
==================================
============================
==[ CoAP
Request ]=============================
================
MID    : 50873
Token  : f6a95a7d
Type   : CON
Method : POST
Options: {"Uri-Path":"led", "Content-
Format":"application/json"}
Payload: 91 Bytes
------------------------------------------------------------
```

Figure 7: Snapshot of LED Resource Execution Result in Smart Phone

5. Performance Evaluation

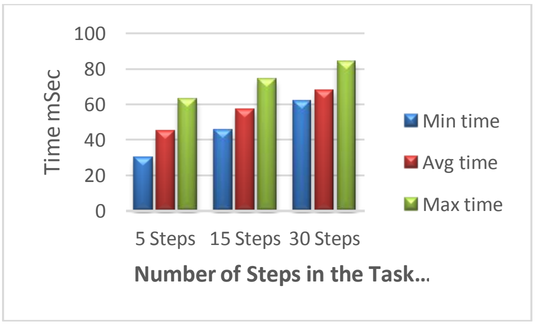


Figure 8: Performance Analysis Graph for Task Deployment Tool

The task deployment tool acquires the process object from the task editor in order to execute it. It first parses that information to create relevant executable objects and then sequence them according to the order set in the graphical business logic created by the user. This process has been analyzed for performance and the results have been represented in Figure 8. Three different business logic files with 5, 15 and 30 steps process were provided to the task deployment tool for this analysis, and each file was allowed to be parsed by the task deployment tool twenty times at random system resource levels. Three measurement methods: the minimum, average and maximum time in milliseconds taken by the task deployment tool are used to compute the time parsing the business logic into executable objects and adjusting the sequence of execution accordingly. For the 5 steps testing, the minimum time taken in the ten iterations was recorded to be 30 milliseconds, averaging at the 45.4 milliseconds and the maximum delay was recorded to be 62 milliseconds. For the 15 steps testing, the minimum time taken in the ten iterations was recorded to be 45 milliseconds, averaging at the 57.2 milliseconds and the maximum delay was recorded to be 68 milliseconds. Lastly, for the 30 steps testing, the minimum time taken in the ten iterations was recorded to be 63 milliseconds, averaging at the 74.4 milliseconds and the maximum delay was recorded to be 84 milliseconds. According to the experimental results, the execution time cost of the task deployment tool is maintained in an optimal level which can be even disregarded and thus, it will not impact the user's experience.

6. Conclusion

The vision of IoT is a global network of diverse sensing and actuating devices for the provision of useful services via data acquisition, communication, data sharing, andactuation. Task allocation on IoT devices is very challenging in a sense that efficient task deployment is interlinked inherently on the network problem and energy consumption. This study proposes the novel idea of an IoT task deployment tool for executing the business process model. Business process modeling has been at the core of software requirement analysis and specification processes. A case study based on mobile devices have been developed to demonstrate the feasibility of the design system. A basic information retrieval and parsing based performance analysis have been performed and the results indicate that the designed system has a good application prospect.

Acknowledgment

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2017-0-00756, Development of interoperability and management technology of IoT system with heterogeneous ID mechanism), and this work was supported by Institute for Information & communicationsTechnology Promotion(IITP) grant funded by the Korea government(MSIT)(No.2018-0-01456, AutoMaTa: Autonomous Management framework based on artificial intelligent Technology for adaptive and disposable IoT)), Any correspondence related to this paper should be addressed to DoHyeun Kim.

References

[1] Gubbi J, Buyya R, Marusic S, Palaniswami M. Internet of Things (IoT): A vision, architectural elements, and future directions. Future generation computer systems. 2013 Sep 1;29(7):1645-60.
[2] Hang L, Jin W, Yoon H, Hong Y, Kim D. Design and Implementation of a Sensor-Cloud Platform for Physical Sensor Management on CoT Environments. Electronics. 2018 Aug 7;7(8):140.
[3] Guinard D, Trifa V, Karnouskos S, Spiess P, Savio D. Interacting with the soa-based internet of things: Discovery, query, selection, and on-demand provisioning of web services. IEEE transactions on Services Computing. 2010 Feb 10(3):223-35.
[4] Li W, Delicato FC, Pires PF, Lee YC, Zomaya AY, Miceli C, Pirmez L. Efficient allocation of resources in multiple heterogeneous wireless sensor networks. Journal of Parallel and Distributed Computing. 2014 Jan 1;74(1):1775-88.
[5] Domingos D, Martins F, Cândido C, Martinho R. Internet of Things Aware WS-BPEL Business Processes Context Variables and Expected Exceptions. J. UCS. 2014 Aug 1;20(8):1109-29.
[6] Ahmad S, Hang L, Kim DH. Design and Implementation of Cloud-Centric Configuration Repository for DIY IoT Applications. Sensors. 2018 Feb 6;18(2):474.
[7] Spiess P, Vogt H, Jutting H. Integrating sensor networks with business processes. InReal-World Sensor Networks Workshop at ACM MobiSys 2006 Jun 19.
[8] Decker C, Riedel T, Beigl M, De Souza LM, Spiess P, Muller J, Haller S. Collaborative business items.

[9] George AA, Ward PA. An architecture for providing context in WS-BPEL processes. InProceedings of the 2008 conference of the center for advanced studies on collaborative research: meeting of minds 2008 Oct 27 (p. 22). ACM.

[10] Meyer S, Sperner K, Magerkurth C, Pasquier J. Towards modeling real-world aware business processes. InProceedings of the Second International Workshop on Web of Things 2011 Jun 12 (p. 8). ACM.

[11] Yousfi A, Bauer C, Saidi R, Dey AK. uBPMN: A BPMN extension for modeling ubiquitous business processes. Information and Software Technology. 2016 Jun 1;74:55-68.

[12] Glombitza N, Lipphardt M, Werner C, Fischer S. Using graphical process modeling for realizing SOA programming paradigms in sensor networks. InWireless On-Demand Network Systems and Services, 2009. WONS 2009. Sixth International Conference on 2009 Feb 2 (pp. 61-70). IEEE.

[13] Casati F, Daniel F, Dantchev G, Eriksson J, Finne N, Karnouskos S, Montero PM, Mottola L, Oppermann FJ, Picco GP, Quartulli A. Towards business processes orchestrating the physical enterprise with wireless sensor networks. InProceedings of the 34th International Conference on Software Engineering 2012 Jun 2 (pp. 1357-1360). IEEE Press.

[14] Nallapaneni Manoj Kumar, Pradeep Kumar Mallick," The Internet of Things: Insights into the building blocks, component interactions, and architecture layers", Elsevier Procedia Computer Science Journal , Volume 132, Pages 109-117, 2018, ISSN:1877-0509, UGC Sl No: 46138 and 48229, DOI: https://doi.org/10.1016/j.procs.2018.05.170.