Things as a Service: Service model for IoT

Amit Kr Mandal[1,2], Agostino Cortesi[1], Anirban Sarkar[3], Nabendu Chaki[4]
[1]*DAIS, Università Ca' Foscari Venezia, Venice, ITALY*
[2]*Department of Computer Science & Engineering, BML Munjal University, Gurugram, INDIA*
[3]*Department of Computer Science & Engineering, National Institute of Technology, Durgapur, INDIA*
[4]*Department of Computer Science and Engineering, University of Calcutta, Kolkata, INDIA*
amitmandal.nitdgp@gmail.com, cortesi@unive.it, sarkar.anirban@gmail.com, nabendu@ieee.org

***Abstract*—Leveraging the benefits of service computing technologies for Internet of Things (IoT) can help in rapid system development, composition and deployment. But due to the massive scale, computational and communication constraints, existing software service models cannot be directly applied for IoT based systems. Service discovery and composition mechanism need to be decentralized unlike majority of other service models. Moreover, IoT services' interfaces require to be light weight and able to expose the device profile for seamless discovery onto the IoT based system infrastructure. In addition to this, the "things" data should be associated with its present context. To address these issues, this paper proposes a formal model for IoT services. The service model includes the physical property of "things" and exposes it to the user. It also associates the context with the "things" output, which in turn helps in extracting relevant information from the "things" data. To evaluate our IoT service model, a weather monitoring system and its associated services are implemented using *node.js* [31]. The service data is mapped to SSN ontology for generating context-rich RDF data. This way, the proposed IoT service model can expose the device profile to the user and incorporate relevant context information with the things data.**

Keywords—Internet of Things, Things as a Service, SSN Ontology, Web Services.

I. INTRODUCTION

Nowadays Internet technology enables "things" to share information and collaborate with other "things". This is what is usually called Internet of Things (IoT). It is envisioned as digital fabric woven with the interactive information of humans and machines, where "things" can be accessed and managed remotely as cloud services. This accelerated large scale adaption of IoT: Gartner estimates that there will be more than 20 billion IoT devices by 2020 [1]. This scenario asks for the integration of service computing concepts into the IoT to manage the increasing volume and varieties of "things" integrated into the cyber world [2]. It will greatly enhance the capability of IoT and enable an efficient utilization of IoT resources by facilitating a seamless discovery, selection and composition mechanism inherent to service computing [3].

Recent research on IoT services mainly focused on enabling technologies for various applications, ranging from home automation [4] to healthcare [5] and smart manufacturing [6]. Majority of these application specific approaches provide a significant insight of the challenges inherent to the IoT solutions. However, there is still a lack of formal mechanisms for modelling various aspects of IoT based system towards a common procedure for enabling technology irrespective of the application domain [8]. Recently several researchers [7, 9, 10] provided formal semantics of IoT based systems capable of addressing various aspects ranging from behavior modeling of the "things" to verify the correctness of network deployment of the "things". Besides, few researchers provided abstract language semantics for modelling IoT system [11, 12]. However, most of the works cited so far don't consider the service orientation of the "things"; this requires a layer of abstraction to hide most of the low-level complexities. Moreover, the IoT service model should provide users standardized access to various "things". It should also associate context along with the "things" data. For this, it should include device profile in the modeling process. Instead, most of the recent literatures mainly focuses on the middleware architecture, potential provider and consumer of IoT services and its pricing model [2, 16].

Things as a Service (TaaS) refers to the concept of delivering IoT capabilities to the end user without operation or maintenance overhead. The primary objective of TaaS is to provide "things" data as a service [2]. But in most of the recent literature IoT services are not designed from a data representational perspective, as they only rely on measured values [2, 15]. Searching, reusing, integrating, and interpreting data become difficult without the studied feature of interest. This minimizes the interoperability and analytical capabilities, falling short of fulfilling the promises of IoT towards knowledge harvesting from sensed data. To enable this, physical property of the "things" is needed to be included in the IoT service modeling. This will help in incorporating certain context in the data. To this aim, Semantic Sensor Network (SSN) Ontology [29] can be used as a meta-model. It will be useful for describing devices (sensors and actuators), their observations, the procedures involved in observation, their features of interest, and the observable properties. It will not only incorporate context in the "things" data, but it will also enhance the IoT service discovery and selection mechanism. Moreover, IoT applications not only require having "things" connected to Internet but they should be integrated into current Internet infrastructure where Web

services are predominant. IoT service model should support exposing the physical properties of "things" and event-based interaction among various other services, and Device Profile Web Service (DPWS) [28] can be considered as a standard for IoT service description. It provides specifications for describing, discovering, messaging, and eventing of services for devices. Fysarakis *et al.* [27] pointed out that the equivalent *Node.js* based implementation of DPWS is lightweight, faster and offers a large array of features compared to DPWS.

This paper proposes a formal model of the Things as a Service. The service model comprises "things" physical property and associates context to the "things" data.

In order to provide evidence of the benefits offered by this service model, a weather monitoring service is developed using *Node.js*, which is based on the devised concept and components of IoT service model and the "things" are mapped to SSN ontology for generating RDF data. This shows that the proposed IoT service model is capable of exposing "things" physical properties to the user and can incorporate relevant context with the "things" data. This will enable context-aware service discovery and composition mechanism. The RDF data can be directly used in different analytics scenarios.

II. Related Work

Service computing technologies have been extensively applied IoT applications for enabling scalable and reusable system development and integration. However, the service models of IoT significantly differ from the existing service models. Zhu *et al.* [13] and Bastani *et al.* [14] pointed out that unlike web services, IoT service discovery and composition should be de-centralized. Thus, discovery and routing protocols requires to be enhanced. To address the modeling and design issues of IoT services various researcher proposed architectural frameworks. Perera *et al.* [2] proposed a sensing as a service model comprises of four layers namely, sensors and sensor owners' layer, sensor publishers' layer, extended service providers layer and sensor data consumers layer. They provided a detailed description and functionalities of individual layers and a real-world scenario based on the model. Kantarci *et al.* [15] presented a state-of-the-art framework for IoT and Sensing-as-a-Service. They defined an aggregation framework for Wireless Sensor Networks (WSN) which provides sensing and actuation clouds as a service. Further, a Service Oriented Architecture (SOA) based sensor data exchange pattern has been discussed. Vargiu *et al.* [19] proposed an agent-oriented abstraction for design and development of IoT systems. Whereas, Alam *et al.* [20] presented an event driven sensor virtualization technique for IoT cloud. Further, Kim *et al.* [16] proposed SenseCloud, platform to addresses the challenges of virtualization, multitenancy, and dynamic provisioning. It provides a two-level virtualization mechanism. However, these architectural frameworks provide very minimal information about service modeling for IoT systems, they mainly focused on service provisioning. Thus, the major challenges of IoT service modelling such as decentralized service discovery and selection, registry structure, routing and composition of IoT services remained unaddressed [13, 14]. Further, these models do not discuss about the data formats and their interoperability.

However, the rapidly changing technology landscape induces difficulty towards selecting an approach for IoT service modeling which ensures the sustainability interoperability. For this purpose, various researcher opted for model driven development. Ciccozzi *et al.* [22] and Hassine *et al.* [18] provided a meta-model of the IoT domain. It helps in modeling and generating executable code for the "things". Although it enables technology neutral development process, but the system did not include the service aspects of the "things". Fortino et al. [20] presented a smart object based IoT meta-model for design and development of IoT services. Whereas Morin *et al.* [23] proposed an UML based approach to address the distribution and heterogenicity challenges in IoT environment. But these approaches did not discuss about service composition aspects. Besides, Zhu et al. [13] and Bastani et al. [14] presented an ontology based meta model for IoT services by extending the OWL-S ontology. Alam et al. [20] proposed an ontology-based knowledge representation framework. Seydoux et al. [24] compared the available ontologies and proposed an ontology suitable for IoT. Whereas, De et al. [25] devised an SSN ontology based IoT services. However, majority of these models did not consider device profile in service modeling and did not discuss about event-based interaction among various devices. Some of the ontology-based approach did not discussed the service orientation of the ontology models. Whereas, many approaches do not provide support the standard SSN Ontology for sensor data representation.

A better insight about the different kind of IoT architectural model is discussed by Taivalsaari et al [17] based on various important factors such as cost, update capabilities, dynamic programmability, security, energy efficiency, and communication latency. This study shows that each model is suited for a specific application domain. Thus, a formal model of IoT service will help towards designing, development and evaluation of IoT services irrespective of the architectural framework. Buono et al. [9] proposed a formal language namely EuDroid which helps in managing the home IoT devices. Humayoun et al. [12] presented a formal task modeling language for defining and evaluating IoT scenarios called IoTGolog. It formalizes IoT system to evaluate the characteristics of the system. Cacciagrano et al. [11] presented a domain specific language called IRON, which incorporates Event-Condition-Action (ECA) rules. It also helps to prevent or report incorrect actions in the system modeling. Castiglion et al. [7] presented a process calculus for IoT system modeling. The formal notion is based on reduction semantics and existential semantics. However, majority these approaches are not designed to accommodate service orientation of the IoT system. A few of the existing formal approach enables service orientation but they did not consider the device profile into consideration. Thus, not suitable for IoT service modeling. This demands a formal model for conceptualizing various aspects of IoT based system in order to attain a common

procedure for enabling technology irrespective of the application domain supported by a suitable architectural framework.

III. IoT Service Model

The primary constituent of "things" is the device capability: sensor or actuator. The functionality of devices can be offered as a service to the end user. IoT Eclipse Working Group [30] provides a generic IoT architecture where many "things" use some form of gateway to communicate through a network to an enterprise back-end server. The proposed approach, depicted in Fig.1, extends the architecture described in [30] for the IoT services where "things" can expose the service interface via service gateways and publish it in cloud of "things".

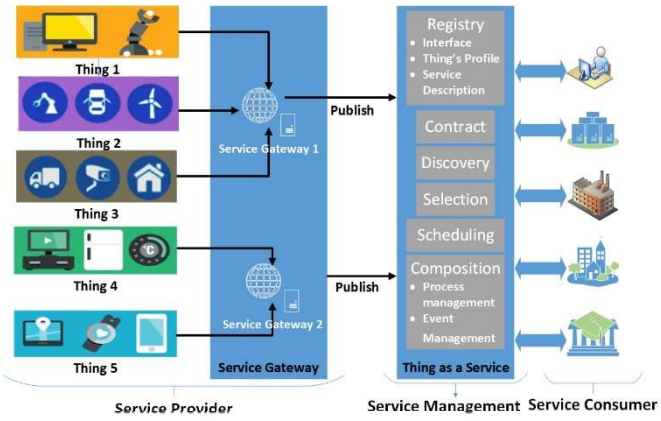

Fig. 1. Architectural Framework for Thing as a Service

The cloud maintains a registry and offers the generic functionalities of service-oriented system such as discovery, selection, contract negotiation, and composition. A formal model of IoT services is devised to conceptualize the different concepts and components of it. For this, device capabilities and physical properties is conceptualized, which are the primary constituents for "things". These "things" are then used to model the service.

A. Device:

In IoT, a device ($\mathcal{D}$) can be considered a mechanical or electronic system which can be operated by analog or digital signals. Based on the functionality it be categorized into a sensing device ($\mathcal{D}_s$) or an actuating device ($\mathcal{D}_a$). A sensing device responds to stimulus in the environment by converting the physical parameter to a signal which can be measured electrically. Whereas, an actuating device converts control signal into mechanical action. Formally, a device can be defined as,

$$\mathcal{D} =< id, f, \mu, ph, I, \mathcal{O} >$$

Here, id refers to a unique identity of the device. f represents the current *feature of interest* of the device. It is atomic and refers to the entity on which the sensor observation or actuation is performed. μ is an unalterable *observable property*. ph represents the set of *physical properties*. It is comprised of multiple system properties such as, accuracy, operating range, battery life, detection limit etc. I & $\mathcal{O}$ are the supported set of *input* and *output* respectively. For a device input can be a user input (i) and/or an event (e_i). Similarly, a device can return certain value (σ) to the user or generate an event(e_σ). Thus, input and output of a device can be defined as,

$$I = \{x | x \in i \vee x \in e_i\}, \quad where\ i \cap e_i = \emptyset$$

$$\mathcal{O} = \{x | x \in \sigma \vee x \in e_\sigma\}, \quad where\ \sigma \cap e_\sigma = \emptyset$$

B. Things:

In the context of IoT, a "thing" can be considered an entity or physical embedded system that has a unique identifier with the ability to transfer or receive data over a network. Formally it can be defined as:

$$\mathcal{T} = \{\mathcal{D}, \mathcal{P}, \mathcal{C}, \mathcal{L}, \varphi\}$$

Where, $\mathcal{D}$ is the set of devices and may consists multiple sensing ($\mathcal{D}_s$) and/or actuating ($\mathcal{D}_a$) devices,

$$\mathcal{D} = \left\{\{\mathcal{D}_{s1}, \mathcal{D}_{s2} \dots \mathcal{D}_{sp}\} \cup \{\mathcal{D}_{a1}, \mathcal{D}_{a2} \dots \mathcal{D}_{aq}\}\right\}$$

$$\text{where, } |\mathcal{D}_s \cup \mathcal{D}_a| \geq 1 \ and \ \mathcal{D}_s \cap \mathcal{D}_a = \emptyset,$$

$\mathcal{P}$ represents the physical property of the "things", which comprises of physical properties of all the sensing and actuating devices. Thus,

$$\mathcal{P} =< ph_{D1}, ph_{D2} \dots ph_{Dn} >.$$

$\mathcal{C}$ is the current set of context associated with the "things" which is a combination of its current feature of interest and observable property. Thus, context of a device d can be defined as a tuple: $\mathcal{C}_d =< f_d, \mu_d >$ and context of the "things",

$$\mathcal{C} =< \mathcal{C}_{d1}, \mathcal{C}_{d2} \dots \mathcal{C}_{dn} >$$

$\mathcal{L}$ represents the operational semantics of the "things", it comprises of procedures, communication protocols, encryptions, maximum no of client "things" can be connected to etc.

φ represents set of operational functions of the "things" which specifies how to make an observation for a specific device. The "things" use procedures to measure the observable property of the device by using a supported input format. The operational function for a device d, which takes input (I_d) to produce an output ($\mathcal{O}_d$) at timestamp t with associated context (c_d) can be defined as:

$$\varphi_d : I_d \rightarrow < \mathcal{O}_d, t, c_d >$$

C. Things as a Service (TaaS):

The TaaS enables interface-based access to the "things" capabilities over internet. All the service components of TaaS will be implemented through an interface. Formally, it ($TaaS$) can be defined as:

$$TaaS = \{addr, \mathcal{T}, \psi, port, msg, pf\}$$

Here, $addr$ is the unique interface address, $\mathcal{T}$ represents the "things" which hosting the service, a "thing" can host multiple services.

ψ is the exposed set of capabilities offered by the various devices in the "things" as functions. Thus, $\psi = \{\varphi_{d1}, \varphi_{d2} \dots \varphi_{dn}\}$.

$port$ represents a set of ports in the interface.

msg defines the information exchange patterns supported by the service interface. Where, messages consist of a unique identifier (mid), message type $(type)$, description $(mdes)$ and message fault (mf). It can be formally defined as: $msg = < mid, type, mdes, mf >$.

pf is the description of the "things" profile. It includes physical & operational properties. Here, $\mathcal{P}' \in \{\mathcal{P} \cup \mathcal{L}\}$ and it only consists of the properties of the devices responsible for providing the service.

IV. Applying the Model to IoT Service Design

To evaluate the proposed IoT service model, a weather monitoring system ("thing") and its REST based services is developed first. Later, a mechanism for mapping the "things" concepts and components to SSN ontology has been devised.

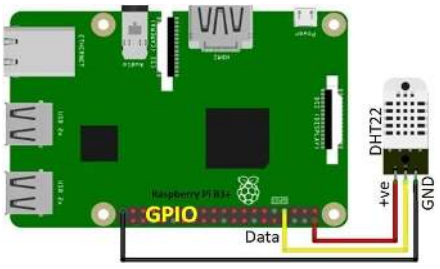

Fig. 2. Schematic Diagram of the Weather Monitoring Thing

A. Weather Monitoring Service:

The weather monitoring system provides the current temperature and humidity. The DHT22 sensor module is used for this. It is installed in a Raspberry Pi 3B+ board with 64-bit, Quad-Core, Broadcom BCM2837B0 CPU running at 1.4GHz and 1GB of LPDDR2 SDRAM. Figure 2 depicts the circuit diagram of the system. Further, *Node.js* is used to develop a REST service. The service offers device profile, current temperature and current humidity in JSON format. For simplicity device profile is described in a JSON file which includes few important parameters such as, Things ID, deployment platform, supported protocol, service type, Endurance, Up-Time and constituent devices information. The individual device is further described to provided: device ID, device type, manufacturer, accuracy, and service endpoints. The profile $(\mathcal{P})$ for the devised thing is depicted in Figure 3. Whereas, partial *Node.js* implementation is depicted in Figure 4. Here the things profile is parsed and used in the *thingProfile* endpoint to expose it to users. Again, for the DHT22 Sensor *rpi-dht-sensor* [26] library is used. The *getTemp* $(\mathcal{L})$ endpoint provides the current temperature $(\mathcal{O}_d)$, feature of interest, observable property (c_d) and timestamp (t) in JSON format. Similarly, *getHumidity* $(\mathcal{L})$endpoint provides current humidity $(\mathcal{O}_d)$ feature of interest, observable property (c_d) and timestamp (t).

```
{"ThingID": "123-456-222",
 "Device 1":
    {
      "DeviceID": "D1",
      "DeviceType": "DHT22"
      "Manufacturer":"XYZ srl"
      "accuracy ": "+/- 0.5",
      "ServiceEndpoint": " getTemp"
    },
 "Device 2":
    {
      "DeviceID": "D2",
      "DeviceType": " DHT22"
      "Manufacturer":"XYZ srl"
      "accuracy ": "+/- 0.5",
      "ServiceEndpoint": " getHumid"
    },
  "Platform": "RaspberryPiB3+",
  "protocol": "HTTP",
  "ServiceType": "REST"
  "Endurance":"96 Hrs";
  "UpTime": "38:45 Hrs"
}
```

Fig. 3. ThingsProfile.json

```
var express = require('express');
var rpiDhtSensor = require('rpi-dht-sensor');
var dateTime = require('node-datetime');
var app = express();
var fs = require("fs");
var dht = new rpiDhtSensor.DHT22(2);
var dtx = dateTime.create();
var dt = dtx.format('Y-m-d H:M:S'); //timestamp
var fi = "Venice Metropolitan Area"; //feature
of Interest
var opD1 = "temperature °C"; //observable
property
var opD2 = "Humidity %"; //observable property
const port = process.env.PORT || 3000;
const server = app.listen(port);
app.use(function (req, res, next) {
 res.header('Content-Type', 'application/json');
 next();
});
app.get('/api/thingProfile',(req, res) => {
 var contents =
fs.readFileSync("ThingsProfile.json");
 var tp = JSON.parse(contents);
  res.send(JSON.stringify(tp));
})
app.get('/api/getTemp', (req, res) => {
  var readout = dht.read();
  var tc = readout.temperature.toFixed(2);
  var res1 = {"temperature":tc,
"location":fi,"measurement":opD1, "Timestamp": dt};
  setTimeout(read, 5000);
  res.send(JSON.stringify(res1));
})
app.get('/api/getHumidity', (req, res) => {
  var readout = dht.read();
  var humid = readout. humidity.toFixed(2);
  var res2 = {" humidity ": humid,
"location":fi,"measurement":opD2, "Timestamp": dt};
```

```
34.   setTimeout(read, 5000);
35.   res.send(JSON.stringify(res2));
36. })
```

Fig. 4. Node.js REST Service for Weather Monitoring Thing

B. *Compatibility with SSN Ontology*

The semantic sensor ontology is widely accepted as standard for sensor data representation. The service developed using the TaaS model can easily be mapped to SSN ontology. On successful invocation of the services, the user gets the output in JSON format. The users get "things" profile as

```
Method ParseSensorData( ){
 String gettemp = getTemp(http://host:3000/api/
getTemp)
 JSONObject tempobj = new JSONObject("gettemp");
 for(each attribute value in tempobj){
  if(CObject.getsrting("location")!=NULL)
    sosa:FeatureOfInterest =
CObject.getsrting("location")
if(CObject.getsrting("temperature")!=NULL){
    sosa:Result =
CObject.getsrting("temperature")
    sosa:resultTime =
CObject.getsrting("Timestamp")
   }
  …//parse other attributes
 }
}
Method ParseDeviceProfile( ){
 String dp =
getDeviceProfile(http://host:3000/api/thingsProf
ile)
 JSONObject obj = new JSONObject("dp");
  for(each nested device attribute in obj){
  JSONObject CObject = obj.get("Device1");
    for(each attribute value in CObject){
     if(CObject.getsrting("DeviceID")!=NULL)
       ssn:sensor =
CObject.getsrting("DeviceID")
     if(CObject.getsrting("DeviceType ")!=NULL)
       ssn:property =
CObject.getsrting("DeviceType")
 …//parse other attributes
    }
  }
  if(CObject.getsrting("DeviceType ")=NULL)
    ssn-system:BatteryLifetime =
CObject.getsrting("Endurance")
  if(CObject.getsrting("UpTime")!=NULL)
    ssn-system:BatteryLifetime =
CObject.getsrting("UpTime")
  …//parse other attributes
}
Method GenerateTTL("temperature.ttl"){
 Create("<sensor/DHT22>","sosa:Sensor "){
 add values for:
sosa:observes,sosa:madeObservation
 }
 Create("<Venice/temperature °C>","
sosa:ObservableProperty"){
  add values for: sosa:isObservedBy
 }
 for(each observation: i){
  Create("<Observation/i> ","sosa:Observation
"){
   add values for: sosa:observedProperty,
sosa:madeBySensor, sosa:hasResult
  }
 }
}
```

Fig. 5: Parsing JSON output to generate SSN Ontology data

```
@prefix sosa: <http://www.w3.org/ns/sosa/>.
@prefix ssn:  <http://www.w3.org/ns/ssn/> .
@prefix xsd:
<http://www.w3.org/2001/XMLSchema#>.
@prefix qudt-1-1:
<http://qudt.org/1.1/schema/qudt#>.
@prefix qudt-unit-1-1:
http://qudt.org/1.1/vocab/unit#>.
@base <http://myproj.org/data/>

<sensor/DHT22> rdf:type sosa:Sensor ;
 sosa:observes  <Venice/temperature °C>;
 sosa:madeObservation
<Observation/1>,<Observation/2>.
<Venice/temperature °C> rdf:type
sosa:ObservableProperty ;
 sosa:isObservedBy <sensor/DHT22>.
<Observation/1> rdf:type sosa:Observation ;
 sosa:observedProperty <Venice/temperature °C>;
 sosa:madeBySensor <sensor/DHT22> ;
 sosa:hasResult [
 rdf:type qudt-1-1:QuantityValue ;
 qudt-1-1:numericValue "6.8"^^xsd:double ;
 qudt-1-1:unit qudt-unit-1-1:temperature °C];
 sosa:resultTime "2019-02-02
10:34:47.048"^^xsd:dateTimeStamp.
 <Observation/2> ...//similar to observation 1
```

Fig. 6: Sample TTL output

defined in the Figure 3, whereas the output format of *getTemp* and *getHumid* is defined in line 26 (opD1) and line 33 (opD2) of the code snippet depicted in Figure 4. The resultant JSON file is parsed and then mapped to the various SSN ontology classes. Figure 5 depicts the partial pseudocode for parsing the "things" output and stored as SSN Ontology RDF data. The method ParseSensorData*( )* of the decided algorithm parses device output and maps it to various SSN ontology classes such as: sosa:FeatureOfInterest, sosa:Result etc. Similarly *ParseDeviceProfile ( )* parses the devise profile data and assigns values for SSN ontology classes such as, ssn:sensor, ssn:property etc. Whereas, *GenerateTTL( )* method uses these values to populate the RDF data and saves it as *".ttl"* file. The sample output of the devised mechanism is shown in Figure 6.

V. Future Research Direction

It is clear from the literature review that the massive scale of the IoT services poses many crucial challenges. This work primarily addressed issues related to the service design from the perspective of "things" and associated context with "things" data. However, many other challenges need to be overcome to foster wide spread adaption of IoT services:

a) Service Registry Structure: A "thing" may provide multiple service, which should be published in the registry to facilitate discovery. The registry structure can be centralized, hierarchical, cache based or combination of these. Moreover, it must provide efficient mechanism to support mobile "things". In addition to these, it also possess challenges of publishing REST services with minimum overhead.

b) Service Discovery and Composition: The computational-constrains and mobility measures of "things" resulted in highly dynamic environment for IoT services, which demands context-aware decentralized service discovery mechanism.

This in turn brings new challenges for service composition in IoT.

c) Routing and Scheduling: IoT services uses different types of communication protocols which can be intra-domain or inter-domain, single-hop or multi-hop. Further, such heterogeneity of IoT demands certain level of intelligence in the communication mechanism. This will enhance the ability of "things" to be aware of the settings in which it is functioning and collaborate with the other devices. Further the collaboration should be supported by an efficient scheduling mechanism.

VI. CONCLUSION

In this paper, a formal model for IoT services is devised. The service model comprises "things" physical property and incorporated context to the "things" data which in turn helps in extracting relevant information. In order to evaluate the devised IoT service model, a weather monitoring service using node.js is devised and later the service data is mapped to SSN ontology to generate context-rich RDF data. It shows that the proposed IoT service model can expose the device profile to the user. This will enable context-aware IoT service discovery and composition mechanism. Further, the proposed model incorporates relevant context with the "things" data, which will help in various analytics scenarios. Moreover, based on the studied literature some crucial challenges for the IoT services is outlined.

Besides the raised research agendas, the future work includes the refinement of the IoT service model to add operational semantics for IoT service composition.

REFERENCES

[1] M. Hung, "Leading the IoT, Gartner Insights on How to Lead in a Connected World." Gartner Research (2017), pp. 1 – 29, 2017.

[2] C. Perera, A. Zaslavsky, P. Christen, D. Georgakopoulos, "Sensing as a service model for smart cities supported by internet of things", Transactions on Emerging Telecommunications Technologies, Vol. 25, no. 1, pp. 81 – 93, 2014.

[3] L. H. Nunes, J. C. Estrella, C. Perera, S. Reiff-Marganiec, A. C. Delbem. "The elimination-selection based algorithm for efficient resource discovery in Internet of Things environments." In Consumer Communications & Networking Conference (CCNC), 2018 15th IEEE Annual, pp. 1-7. IEEE, 2018.

[4] V. Vladimir, M. Maksimović, "Raspberry Pi as a Sensor Web node for home automation" Computers & Electrical Engineering Vol. 44, pp. 153 – 171, 2015.

[5] P. A. Laplante, N. Laplante. "The internet of things in healthcare: Potential applications and challenges." IT Professional 3, pp. 2 - 4, 2016.

[6] F. Tao, Q. Qi, "New IT driven service-oriented smart manufacturing: framework and characteristics." IEEE Transactions on Systems, Man, and Cybernetics: Systems, Vol. 49, No. 1, pp. 81 – 91, 2017

[7] V. Castiglioni, R. Lanotte, M. Merro. "A Semantic Theory of the Internet of Things." arXiv preprint arXiv:1510.04854 (2015).

[8] I. Lanese, L. Bedogni, M. Di Felice, "An Operational Semantics for a Calculus for Wireless Systems", Theoretical Computer Science, Vol. 411, No. 19, pp. 1928 – 1948, 2010.

[9] P. Buono, F. Cassano, A. Legretto, A. Piccinno. "EUDroid: a formal language specifying the behaviour of IoT devices", IET Software, Vol. 12, No. 5, pp. 425 – 429, 2018.

[10] C. Mahmoudi, F. Mourlin, A. Battou, "Formal definition of edge computing: An emphasis on mobile cloud and IoT composition", Third International Conference on Fog and Mobile Edge Computing (FMEC), pp. 34 – 42, IEEE 2018.

[11] D. R. Cacciagrano, R. Culmone, "Formal semantics of an IoT-specific language", 2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 579 – 584, IEEE, 2018.

[12] S. R. Humayoun, Y. Dubinsky, R. AlTarawneh. "Using IoTGolog to formalize IoT scenarios", 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), pp. 234 – 238, IEEE, 2015.

[13] W. Zhu, G. Zhou, I. Yen, F. Bastani. "A pt-soa model for cps/iot services." 2015 IEEE International Conference on Web Services (ICWS), pp. 64 – 654. IEEE 2015.

[14] F. Bastani, W. Zhu, H. Moeini, S. Hwang, Y. Zhang. "Service-Oriented IoT Modeling and Its Deviation from Software Services", IEEE Symposium on Service-Oriented System Engineering, pp. 40-47, 2018.

[15] B. Kantarci, H. T. Mouftah, "Sensing as a Service in Cloud-Centric Internet of Things Architecture", Enabling Real-Time Mobile Cloud Computing through Emerging Technologies, pp. 83-115, IGI Global, 2015.

[16] M. Kim, M. Asthana, S. Bhargava, K. K. Iyyer, R. Tangadpalliwar, J. Gao, "Developing an on-demand cloud-based sensing-as-a-service system for internet of things." Journal of Computer Networks and Communications, URL: https://www.hindawi.com/journals/jcnc/2016/3292783 /cta/ [Accessed On: January 28, 2019], Hindwai, 2016.

[17] A. Taivalsaari and T. Mikkonen, "A Taxonomy of IoT Client Architectures," in IEEE Software, vol. 35, no. 3, pp. 83-88, 2018.

[18] T. B, Hassine, O. Khayati, H. B. Ghezala. "An IoT domain meta-model and an approach to software development of IoT solutions", 2017 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC) , pp. 32-37. IEEE, 2017.

[19] E. Vargiu, F. Zambonelli. "Agent abstractions for engineering IoT systems: A case study in smart healthcare", In 14th IEEE International Conference on Networking, Sensing and Control, pp. 667-672, 2017.

[20] S. Alam, M. M. R Chowdhury, J. Noll. "Senaas: An event-driven sensor virtualization approach for internet of things cloud", 2010 IEEE International Conference on Networked Embedded Systems for Enterprise Applications (NESEA), pp. 1-6. IEEE, 2010.

[21] G. Fortino, A. Guerrieri, W. Russo, C. Savaglio, "Towards a development methodology for smart object-oriented IoT systems: a metamodel approach", 2015 IEEE International Conference on Systems, Man, and Cybernetics (SMC), , pp. 1297-1302. IEEE, 2015.

[22] F. Ciccozzi, I. Crnkovic, D. Di Ruscio, I. Malavolta, P. Pelliccione, R. Spalazzese. "Model-driven engineering for mission-critical iot systems", IEEE Software 1 (2017): 46-53, 2017.

[23] B. Morin, N. Harrand, F. Fleurey, "Model-based software engineering to tame the iot jungle", IEEE Software Vol. 34, No. 1, pp. 30-36, 2017.

[24] N. Seydoux, K. Drira, N. Hernandez, T. Monteil. "IoT-O, a core-domain IoT ontology to represent connected devices networks", European Knowledge Acquisition Workshop, pp. 561-576, 2016.

[25] S. De, P. Barnaghi, M. Bauer, S. Meissner, "Service modelling for the Internet of Things", 2011 Federated Conference on Computer Science and Information Systems (FedCSIS), pp. 949-955. IEEE, 2011.

[26] rpi-dht-sensor, URL: https://github.com/roland-vachter/rpi-dht-sensor[Accessed On: January 28, 2019].

[27] K. Fysarakis, D. Mylonakis, C. Manifavas, I. Papaefstathiou, "Node. dpws: Efficient web services for the internet of things", IEEE Software, Col. 33, No. 3 pp. 60-67, 2016.

[28] Devices Profile for Web Services, URL: http://ws4d.org/technology /dpws/ [Accessed On: January 28, 2019].

[29] Semantic Sensor Network Ontology, URL: https://www.w3.org/TR /vocab-ssn/ [Accessed On: January 28, 2019].

[30] Eclipse IoT Working Group. "The three software stacks required for IoT architectures", (latest update: December 2017).

[31] Node.js, URL: https://nodejs.org/en/, [Accessed On: January 28, 2019].