International Journal of Computers and Applications

ISSN: 1206-212X (Print) 1925-7074 (Online) Journal homepage: http://www.tandfonline.com/loi/tjca20

Modeling and assessing reliability of service-oriented internet of things

Ranjit Kumar Behera, K. Hemant Kumar Reddy & Diptendu Sinha Roy

INTERNATIONAL JOURNAL OF COMPUTERS AND APPLICATIONS, 2018
https://doi.org/10.1080/1206212X.2017.1420589

Modeling and assessing reliability of service-oriented internet of things

Ranjit Kumar Behera[a], K. Hemant Kumar Reddy[a] and Diptendu Sinha Roy[b]

[a]Computer Science and Engineering, National Institute of Science and Technology, Berhampur, India; [b]Computer Science and Engineering, National Institute of Technology, Shilong, India

ABSTRACT
In recent years, technological innovations in the fields of sensing, computing, and communication have seen unprecedented advancements. Particularly, the explosive progress in wireless connectivity has resulted in ubiquitous access of devices, software, and control strategies over wide geographical sprawls. The Internet of Things (IoT) is an emerging paradigm that can be seen as the consummation of all these highly disruptive set of technologies. Also, the industry has been shifting from application-oriented solutions towards more service-oriented abstractions. Since most large scale and critical systems of late have been designed with this IoT paradigm, thus, modeling and analyzing the reliability of IoTs becomes an emerging research challenge. In this paper, a reliability model for IoT has been presented that specifically emphasizes the service-oriented aspect. Cloud computing services have been employed as platform for leveraging computations and data storage required for Iot operations in the proposed model. The service-oriented paradigms of such systems have the cloud infrastructure as its focal point and consequently a Centralized, Heterogeneous IoT Service System (CHISS) model has been proposed. The paper also discusses a generalized methodology to evaluate service reliability of CHISS system. Case studies presented herewith include intelligent fire alarm system and smart health care system and the results presented herewith demonstrate the efficiency of the proposed model.

ARTICLE HISTORY
Received 2 March 2017
Accepted 20 December 2017

KEYWORDS
Internet of Things (IoT); reliability; virtual machine (VM); cloud computing

1. Introduction

The ubiquity of accessing devices by virtue of ground breaking advancements in sensing, computing, and communication technologies have brought about a highly disruptive in the way systems and services are being perceived and designed. The Internet of Things (IoT) has emerged as a technology that presents a paradigm shift in terms of conceiving systems and the functionalities they provides with the ultimate objective of societies well-being through internet and communication technology enabled effective monitoring and control mechanism of the physical world [1].

It provides a virtual representation of uniquely identifiable objects that can be remotely sensed over an Internet-like structure [2]. Besides, the range of sensing and task actuations in an IoT encompass RFIDs, sensors, actuators, Wireless Sensor Networks (WSN), and so forth [3]. IoT envisions a multitude of heterogeneous objects and interactions with the physical world. Due to the interaction among large number of heterogeneous objects involved, it poses a major challenge the design of IoT systems. The main objective of these heterogeneous objects is to provide real-world services as its basic functionalities and real-time state of the physical world.

In the present century, the primary research challenge has been to interconnect multifarious devices to form an intelligent group of embedded and networked systems to perform some system goals. Such networked embedded system are capable of sensing, computing, and communicating over short to medium distances using varied wireless technologies, like Zigbee, Wifi, Bluetooth, etc. Such system popularly referred to as environment monitoring, infrastructure planning, traffic and energy management of modern society and many more [2].

Of course to convert this vision of decision-making of such networked embedded systems to reality, the services and data need to be defined in a homogeneous and common platform to support the variegated interactions required for integration of these objects thereof. This necessitates the design of IoT services in a web-service-like fashion for easy access, monitoring and control. Since most large scale and critical systems of late have been designed with this IoT paradigm, thus, modeling and analyzing the reliability of IoTs becomes an emerging research challenge.

Service-Oriented Architecture (SOA) is one of the most potent paradigms of computing which provides services from distributed software and applications. The most capable application of service-oriented technology

CONTACT K. Hemant Kumar Reddy khemant.reddy@nist.edu

is web services. Moreover, recently, vast of literature are focusing on the integration of service-oriented architecture and IoT for the realization of distributed services. Research efforts have also focused on service modeling of IoT to represent the real-world services as web services for efficiently controlling and monitoring. Moreover, limitation of computation and storage of an IoT device is also a significant reason for recline of IoT towards service-oriented technology.

The realization of SOA-IoT necessitates a computing technology for proper control of heterogeneous data, hosting of APIs such as REST, CoRE and so on, management of requested services. However, cloud computing appears as promising computing technology for the service management of SOA-IoT. Therefore, we consider cloud-centric IoT for the realization of SAO-IoT. Cloud-centric IoT architecture has three layers that are device layer, Fog layer/cloudlets, and cloud. The real-time services are executed at the Fog/cloudlet layer. However, rest services are performed at cloud layer. The unceasing, reliable and Quality of Service (QoS) from SAO-IoT requires appropriate reliability modeling approach. In order to do so, heterogeneous IoT systems catering to distributed applications have been modeled using a number of services as their basic constructs, drawing from the service-oriented paradigm. This has been referred to as a Centralized Heterogeneous IoT Services (CHISS). Reliability analysis of such CHISS has to account for service reliability apart from failure statistics details of others IoT components. Thus, in this paper, service reliability of service-oriented IoT has been enumerated drawing from existing literature on service reliability, modeling, and assessment techniques, such as [4–6]. Reliability assessment for network infrastructure of such distributed systems has been studied for various scenarios in [7,8], etc.

Finally, this paper also presents reliability assessment of CHISS modeled IoT system using a novel generalized algorithm for distributed service reliability.

The major contributions of this paper include:

(i) Reliability modeling of service-oriented IoT as a centralized heterogeneous IoT service system (CHISS). The CHISS framework has been modeled with ubiquitous sensing at edge and cloud at the center as a platform for providing data storage and analysis capabilities. In this model, each device provides its functionalities as a service and also acts as a router that can forward data to other devices or can access data from other devices.

(ii) As devices in IoT are characterized by constrained in terms of energy, communications and computational capabilities, thus to provide better services to the users, the reliability and availability of services offered by IoT system should be focus. Therefore, this paper has also given a generalized approach to estimate the service reliability of service-oriented IoT.

The remainder of this paper is organized as follows. Section 2 presents a brief review of Internet of Things (IoT). In Section 3, this paper has proposed a service-oriented IoT with a generalized CHISS framework. Generalized reliability assessment of CHISS-based IoT has been presented in Section 4. Section 5 has presented two case studies. The results are given in Section 6. Finally, Section 7 summarizes the conclusion of the research.

2. Related work

Service-Oriented Architecture (SOA) has emerged to be one of the most potent model of computing as far as reliable, cost effective, and effortless composition of distributed applications are concerned [9]. The most capable service-oriented technology is web services. Recently, vast many researchers have focused on the integration of service-oriented technology for IoT to services. Research efforts have also focused on service modeling of IoT to represent the real-world services as web services for effective controlling and monitoring. P. Spiess et al. [10] has presented a SOA-based IoT for enterprises services. Guinard et al. [11] has presented the SOA-based IoT and has given detailed information about on-demand provisioning of web services. Wei et al. [12] proposes an environment-based approach for service modeling in IoT. Chen et al. [13] delves with WSN applies development with an emphasis on service-oriented approach. Shehu et al. in their research work 'Network aware composition for internet of thing services' considers the network characteristics to propose a model regarding the IoT services [14].

Owing to recently conceived nature of service-oriented paradigm for IoT applications, existing literature is scanty. One of the pioneering works in the field of modeling reliability and cost characteristics of IoT services composition has been reported in [15]. It presents a probabilistic approach for formalization of these parameters. Liu et al. presented a green and reliable communication modeling for industrial internet of things in which they have presented a protocol to improve the lifetime and shorten the delay on the premise of ensuring the reliability [16]. Suparna et al. [3] has presented a semantic approach for modeling various components within the IoT framework. In [17], Zhang et al. proposed a network reliability model using failure criteria of computer network. Some of the works also dedicated to improve the network performances [18]. However, explicit modeling on IoT reliability has not been carried out so far. Although [15] do present such an attempt, yet the focus in that research was more on the formal description of reliability properties for IoT services composition using the Finite State Machine (FSM) abstraction. The emphasis of our research, however, is to harbor upon a methodology for explicit reliability modeling of heterogeneous IoT systems that can help further in reliability assessment.

3. Service-oriented internet of things model

Owing to capabilities like easy locatability, publishability, and invokability across the web, web services are inherently good choice for service-oriented paradigm, having the impetus for transforming the internet from a mere information archive into distributed computing devices. In the subsequent sections, the concept and need of a cloud-based IoT has been briefly explained and also a generalized Centralized Heterogeneous IoT Service Systems (CHISS) is introduced to incorporated service-oriented paradigm.

3.1. Cloud centric IoT: a generalized model

The term Internet of Things (IoT) was first coined by Kevin Aston to represent machine to machine communication in supply-chain management context [1,19]. Internet of Things is basically the interconnection of physical devices in a manner that makes distributed device-data access and working details possible in the global context for smooth operation of such remote devices [20]. At its infancy, Radio-frequency identification (RFID) was seen as a pre-requirement for the Internet of Things [21]. But gradually other technologies like cloud computing [22], wireless sensor network (WSN) [23], FOG computing [24], and Big Data [25] are seen as backbone of IoT.

Gubbi et al. [22] has presented taxonomy of IoT, namely internet-centric IoT and things-centric IoT. In this paper, an internet-centric IoT has been modeled that enables a large No. of objects to have communication and computational capabilities to connect and interact with adjacent objects for proper monitoring and control.

Huge volumes of data/information are generated from different sources. This necessitates a computing technology for proper data management of the real-time data. Cloud computing has emerged as a promising and reliable computing paradigm for data management of internet-centric IoT [26].

In this paper, a model of internet-centric IoT has been presented that employs cloud as the platform for all data management like storage processing and transactions. Figure 1 depicts a generalized architecture of such cloud-centric IoT, having three different layers, namely device layer, Fog layer, and cloud layer. The data collected at device layer using different IoT devices like sensors, actuators, etc., will be forwarded to the Fog layer through a gateway. Fog layer is introduced by looking at the requirement of certain IoT services which require quick response. Fog computing which extends the services of cloud computing does the computing and storage at the edge of the network and can be used for a real time and dynamic environment.

As shown in Figure 1, IoT applications domain can have many categories like home, transport, community, national, etc. IoT elements are available in each application and each application is connected with internet gateways to transfer or access data from to and from the IoT cloud web portal. The cloud domain consists of three actors/

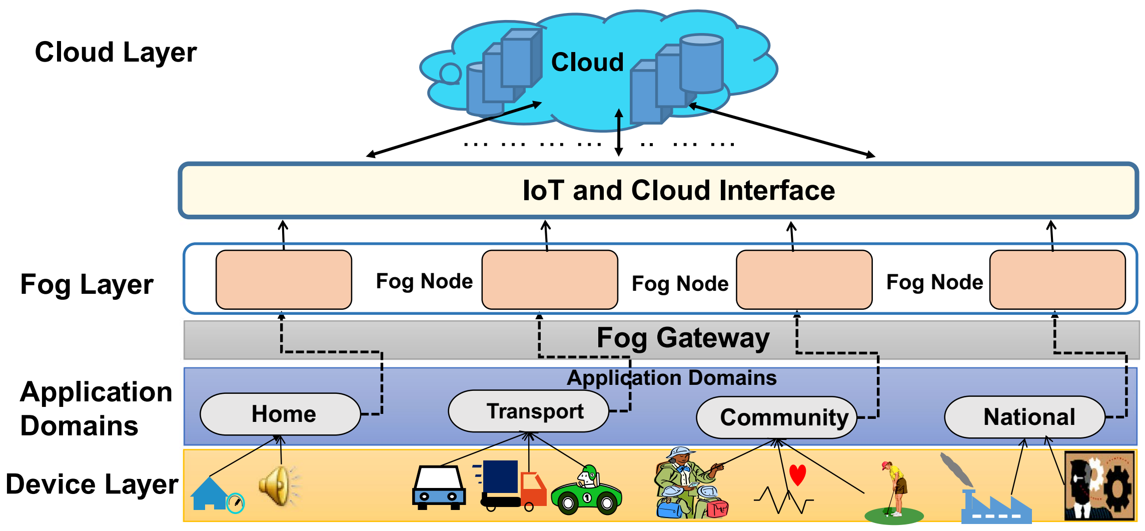

Figure 1. A generalized architecture of cloud centric IoT.

players namely the cloud providers, service providers and end users. All of these are also connected with IoT cloud web portal for accessing data for monitoring and controlling of real-time objects.

3.2. The CHISS framework

Internet of Things (IoT), being comprised of a large collection of devices lends itself naturally to a service-oriented paradigm. In such service-oriented IoT; each node delivers its functionalities as a service to a registry [15]. Such services are referred to as atomic service that act as links between orthodox service-oriented systems and the physical world. Since most networked embedded systems and IoT needs to be controlled centrally to achieve a common goal based on sensed data over a networked set of heterogeneous sensors, hence service-oriented IoT can be modeled as a Centralized Heterogeneous IoT Service Systems (CHISS).

A high-level architecture of IoT as a CHISS has been shown in Figure 2. In this architecture, IoT system consists of heterogeneous sub-systems in diverse topological networks, with diverse miniature devices, that are overseen by a control center.

For the sake of simplicity, this architecture is categorized in two abstract components, namely a control center and a number of distributed systems that collectively form the other component which consists of sensors and actuators. Thus, the second component is referred to as distributed sub-systems and this can constitute applications from diverse domains, such as healthcare, safety. For example in recent times, smart cities are being conceived that draws information from distributed sensors across cities to make informed decisions in fields such as electricity supply management, vehicular control in cities, disaster detection and recovery management. For each of these functions, smart cities draw necessary information from large variety of sources or sensors. However, there needs to be common control center decisions are taken. Naturally, such control centers would require enormous computing and data handling capabilities and thus, cloud data centers have been employed to support virtual machines to execute control algorithms.

The control center consists of many servers and data centers supported by virtual machines (VM) which form a part of cloud infrastructures. Using virtual nodes, the VMs can manage and control programs as well as data from the heterogeneous distributed IoT sub-systems, as it can mask the differences between different platforms. Virtual nodes can be seen as a kind of execution element, which only includes a CPU and memory as the fundamental unit for data execution. The components of virtual nodes and virtual machines are supported by hardware and software in the control center.

IoT systems consist of a variety of miniature devices with various programs and files connected by diverse network topologies. The exchange of data between sub-systems and virtual machine can be done through system service provider interface (SSPI) [4] as depicted in Figure 2. These sub-systems are connected to virtual nodes through router. Basically, router is a networking device, commonly hardware that forwards data packet between nodes i.e. the miniature devices. They can support to achieve the functionality of distributed IoT system. The

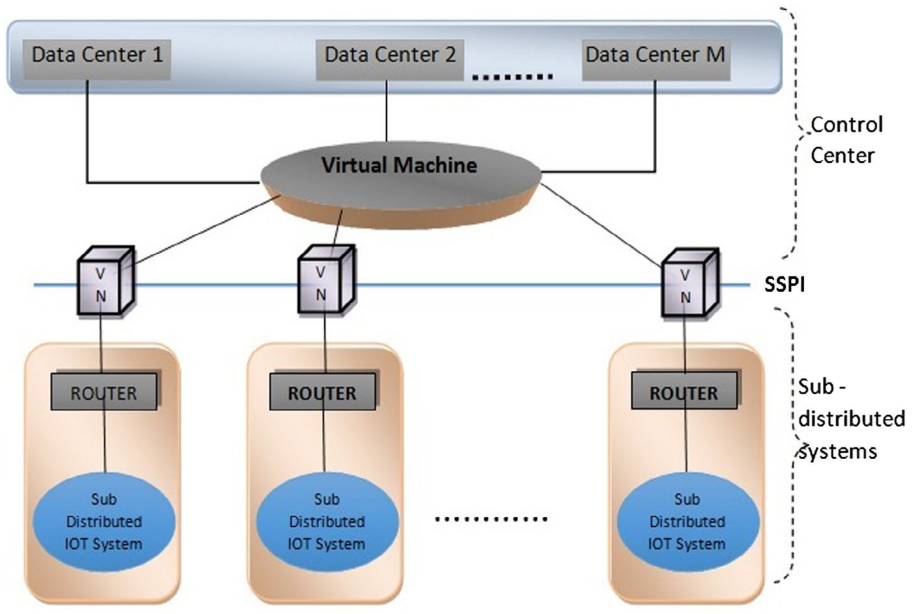
Data Center 1
Data Center 2
Data Center M
Virtual Machine
Control Center
V N
V N
V N
SSPI
ROUTER
ROUTER
ROUTER
Sub Distributed IOT System
Sub Distributed IOT System
Sub Distributed IOT System
Sub - distributed systems

Figure 2. Architecture of centralized heterogeneous service-oriented Internet of Things.

IoT sub-system can be applied on different areas such as home and personal, healthcare, transport.

Since, IoT-based services form the core of very critical services, hence while designing service-oriented IoT services, reliability of such services need to be ensured. The following section presents a reliability modeling for CHISS-based IoT.

4. Generalized service reliability assessment of CHISS-based IoT

Since, most sensors in IoT are connected over wireless channels, which are inherently susceptible to unreliability, reliability modeling of service-oriented IoT should take care of both system availability as well as system reliability to provide IoT services.

Besides, in the control center, all the actuation decisions are automated and algorithm-based. Then for every IoT application, successful operation of that IoT functionary depends on its distributed components reliabilities as well as the reliability of programs that implement the control center algorithms running on cloud data centers.

For execution of control programs in cloud data centers, there exists dependency among devices and programs. The execution of some programs in a device might require certain input files that are generated in different devices of service-oriented IoT system. So service-oriented IoT is just like distributed systems. The overall reliability of service-oriented IoT is dependent on the availability of certain things, such as programs that code the services, files required by those programs and service reliability of the sub-systems.

The service reliability of IoT system can be obtained by the program reliability of each sub-system as shown in Figure 2 and availability of the control center. For successful completion of service in IoT system, programs running in each sub-system must be successfully executed. The availability of virtual machines has to ensure that whenever any program requires some files and they should be available therein.

The time instant at which programs requires the files prepared in VM, T_f^j (where $j = 1, 2 \dots J$ denote the jth program) can be obtained using critical method [27]. The starting times when programs run in the VM, T_s^k and the consequent execution time for those programs, T_{exe}^k (where $k = 1, 2 \dots N$ denotes the kth program) can also be obtained by referring to the equations presented in [8].

Let $A(t)$ denote VM availability at an instant t. It has been assumed that program need input files at the start time, T_f^j. Thus, availability of files required maybe,

$$P_f(j) = A(T_f^j) \quad \text{where } j = 1, 2, 3, \dots J \tag{1}$$

In this paper, it has been assumed that the virtual machine is available throughout the duration for which a program runs on it. Hence, the mean program availability starting at instant T_s^k with an execution period maybe expressed as:

$$P_{pr}(k) = \int_{T_s^k}^{T_S^k + T_{exe}^k} A(t)\mathrm{d}t / T_{exe}^k \quad \text{where } k = 1, 2, 3, \dots, K \tag{2}$$

For N sub-systems as shown in Figure 2, the IoT system reliability for the jth sub-system can be expressed as ISR_j ($j = 1, 2 \dots N$). The ISR_j can be computed for each sub-system while considering the virtual machine (VM) to be a perfect node across all sub-systems and all the sub-systems are assumed to be mutually independent.

These assumptions are necessary, although they need not always be true. This shows that requests for service are independent, so as the VM failure. The availability of prepared files and executed programs in VM has not been considered while calculating ISR_j. The service reliability of service-oriented IoT can be expressed as,

$$R_s(t) = \prod_{i=1}^{N} ISR_j \prod\nolimits_{j=1}^{j} P_f(j) \prod_{k=1}^{k} P_{pr}(k) \tag{3}$$

where N is the number of sub-systems, j denotes the jth programs which require input and output files, and k denotes kth program, Equation (3) expressed the IoT service reliability function to the initial time, t.

The steps to calculate the IoT service reliability are as follows:

(1) Identification of the CHISS structure for specific IoT function.
(2) Then finding the relationship among programs and files needed for the said structure.
(3) Using the existing model obtain the availability function.
(4) Assuming the VM a perfect one for every sub-system, calculate ISR_j ($j = 1,2, \dots, N$), which has been discussed in the following sections.
(5) Using critical path method, determine T_f^j ($j = 1,2, \dots, j$), T_s^k and T_{exe}^k ($k = 1, 2, \dots, k$).
(6) Enumerate $P_f(j)$ and $P_{pr}(k)$ by employing Equations (1) and (2).
(7) Enumerate the ISR function to initial time t from Equation (3).

4.1. The availability function

Let $\lambda(t)$ be the failure intensity of VM and μ be its repair rate. For this paper, $\mu = 0.5$ has been assumed (as has been done in [8]).

$$\lambda(t) = abexp^{(-bt)} \quad (4)$$

The values $a = 10$ and $b = 0.01$ have been assumed. Here, VM failures considered to be a Markov process with two states, namely up and down. $P_0(t)$ and $P_1(t)$ represent the probabilities of VM to be in normal working state and in failure state at time instant t respectively. The corresponding Kolmogorov's differential equations for these states can be represented by:

$$P_0'(t) = \mu P_1(t) - \lambda(t)P_0(t) \quad (4a)$$

And

$$P_1(t) = 1 - P_0(t) \quad (4b)$$

Assuming the initial conditions as $P_0(0) = 1$, $P_1(0) = 0$ drawing from [8], $P_0(t)$ can be represented as:

$$P_0(t) = \left[\left[\int_0^t u\left(\mathrm{e}^{(ux-ae^{-bx})}\mathrm{d}x + \frac{1}{e^a}\right)(e^{(-ut-b)} + ae^{-bt})\right]\right] \quad (5)$$

$P_0(t)$ is the availability function $A(t)$ of VM because it will be available when it is in normal state.

4.2. IoT System Reliability (ISR) assessment

For reliability assessment of IoT system, reliability of individual sub-systems needs to be accounted for. For instance, Figure 3 depicts a service-oriented IoT having a number of sub-systems, with a reliability of ISR_j where j represents a particular sub-system. To estimate ISR_j, this paper has further assumed that nodes never fail, probability of an edge to be working is 0.9 and programs are always available whenever requested.

Based on these assumptions, $ISR_j (j = 1,2,\ldots)$ can be evaluated using the algorithm presented in [28]. In order to evaluate ISR_j, we have employed a few data structures, drawing from literature [28,29], namely File Usage Vector,

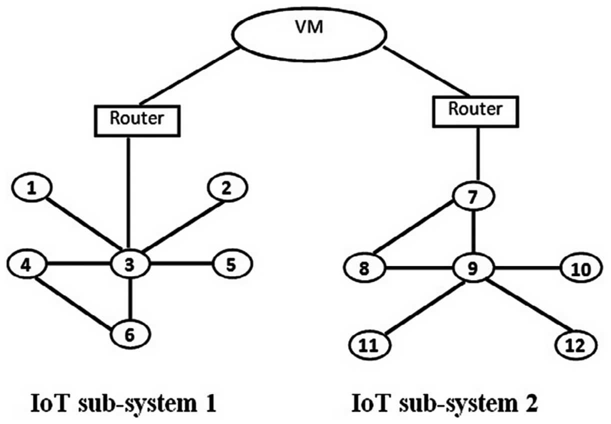

Figure 3. The CHISS for the application example.

reliability vector, Path vector, and so forth. The following subsections present a brief summary of each.

4.2.1. File Usage Vector (FUV)

A File Usage Vector (FUV) keeps tracks of file usages by all programs that run in data centers. FUV is represented as a vector having 'f' bits where 'f' is the total number of files available in the system taken as a whole. Every bit in this vector represents a particular file. A 1 means the file needed by program, whereas 0 denotes it is not needed.

If a link connects N_i to N_j and N_j has a file that is needed by program P_i then that corresponding bit in the FUV is changed to 0. In case the required file is not present, the vector is left as it is.

4.2.2. Reliability Vector (RV)

Reliability Vector (RV) keeps track of link information. The RV is a l bit vector where each bit represents the number of links in the system. RV can have one of the three values: 0, 1, or d. A 0 in the RV denotes that the link is in use but not operational, a 1 in the RV signifies that currently the link is in use and operational where as a d in the RV signifies that the node is in isolation or in other words, it is not connected to any other node.

All the terms in RV can be added to get the reliability expression, since they are disjoint. There are two rules for updating the RV, as per [26,29].

Rule 1: when the node to be expanded is the first child of the parent

$RV_{old} = RV_{predecessor}$

Rule 2: when the node to be expanded is some other node then

$RV_{old} = RV_{elder\text{-}sibling}$

RV can be updated as follows:

RV_{old} = set bit 'i' corresponding to link x_i to 1 in RV.

RV_{left} = set bit 'i' corresponding to link x_i to 0 in RV_{old}.

4.2.3. Path Vector (PV)

Path Vector (PV) keeps track of information about all the nodes that are encountered in the path while looking for files that are not locally available. The PV is an n bit vector where each bit corresponds to the number of nodes and for each bit that is traversed, that particular bit is set to 1. Other bits have default value 0.

If in a path we traverse from $Node_i$ to $Node_j$, then corresponding bit 'j' in the PV will be set to 1.

The algorithm gets terminated when ever either of the following two conditions gets satisfied:

(1) All the bits of the File Usage Vector become 0.
(2) When all the bits of the Path vector become 1.

However to compute the reliability expression, we consider only those path terminated by condition 1. The algorithm 1 presented in Section 4.2.4 provides the pseudo code for the whole process.

4.2.4. *Algorithm 1*

```
INPUT:{NV: Node vectors, AV: Adjacency vector, FUV: File Usage Vector}
OUTPUT: FRV
1. Initialize FUV, RV, PV as depicted in section 4.2.1, 4.2.2 and 4.2.3
2. for each node $n_i$: NV[], do:
        if(ni is Adj(SN)) then //SN- Source Node
        $T_{mat} \leftarrow n_i$; //Store ni in Tmat
        endif;
    endfor;
3. for each node $n_i$ : $T_{mat}$[][], do:
    Update FUV, PV, RV as in section 4.3.1, 4.3.2 and 4.3.3
    if((all bits(FUV[ ]) becomes 1) then
        FRV[k] ← RV; //—add RV to FRV (final reliability vector)
        Set $n_i$.status ← 0; //—disable $n_i$ in $T_{mat}$
    endif
    if(all bits(PV[ ]) becomes 1) then
        Set $n_i$.status←0;
    endif;
endfor
4. if (All $n_i(T_{mat})$.status = 0) then
    set FRV[i] as final output of $n_i$;
    next i;
    stop;
else do:
for each $n_i$ in $T_{mat}$, do
    $RV_{old} \leftarrow$ RV(Parent($n_i$));
    Compute the values for NewNode
    set RV of $i^{th}$ Bit (NewNode)←1
    Find all Adj(New node) & store in $CT_{mat}$
    copy $CT_{mat} \leftarrow T_{mat}$;
    for each $n_i$ in $T_{mat}$, do:
        if($i^{th}$ Bit of FUV = 1) then
            set PV of $i^{th}$ Bit (NewNode) ← 0;
        endif;
        set PV of ith Bit (NewNode) ← 1;
        set Rvleft of ith Bit (NewNode) ← 0;
        endfor;
    endfor;
while ($n_i(T_{mat})$.status != 0);
End.
```

The generalized method for reliability assessment presented in this section has been exemplified with two case studies to demonstrate how the algorithm works for different IoT systems.

5. Case study

For any case study, the foremost step for the service reliability analysis of IoT is to identify the structure of CHISS and relationship between its programs. This paper demonstrates the application of the reliability model presented in Section 4 to two sample sub-systems, namely, an intelligent fire alarm system and a smart healthcare system, one from a home/personal category, and the other from a community category.

5.1. *The CHISS structure*

An abstract view of the Centralized Heterogeneous IoT Service System (CHISS) for the above-mentioned application category is given in Figure 3. The network topologies shown in Figure 4 of two sub-systems are designed according to the required programs and files needed to provide its defined services to the end users. The network connection between devices depends on the program and files needed by other devices. The name of each node and their relationship for the sub-systems is represented in Figures 4 and 5, respectively.

5.2. *IOT sub-system 1: an intelligent fire alarm system*

Fire alarm system functionality is described as a process which starts by periodically receiving real-time information from temperature sensors and smoke sensors. An analyzer constantly checks whether the temperature sensed by the temperature sensor is above a threshold and the smoke that is detected can cause choking or not. If the information processed by the analyzer is below the threshold for both the temperature sensor and smoke detector then a ‘no danger’ signal is passed and the stage is reverted to sensor and detector. When both the conditions and even one is true, the analyzer activates the water sprinkler and the fire alarm. The sprinkler is connected to a water tank. After activation, the water sprinkler and the alarm operate concurrently. Both the sprinkler and the alarm constantly take inputs from the analyzer which in turn monitors the current temperature and smoke level. Once below the threshold, the water supply is automatically cut-off and alarm is switched off.

Based on the above explanation the intelligent fire alarm system can be represented with a common network topology with the required programs and files to provide its defined services to the end users. Figure 4 depicts 6 components of the fire alarm, denoted N_1 through N_6. PA and FA denote program available and file available,

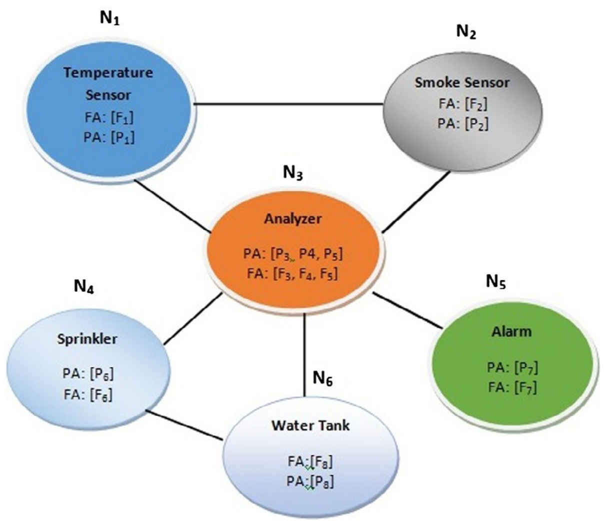

Figure 4. Detailed structure of IOT sub-system 1.

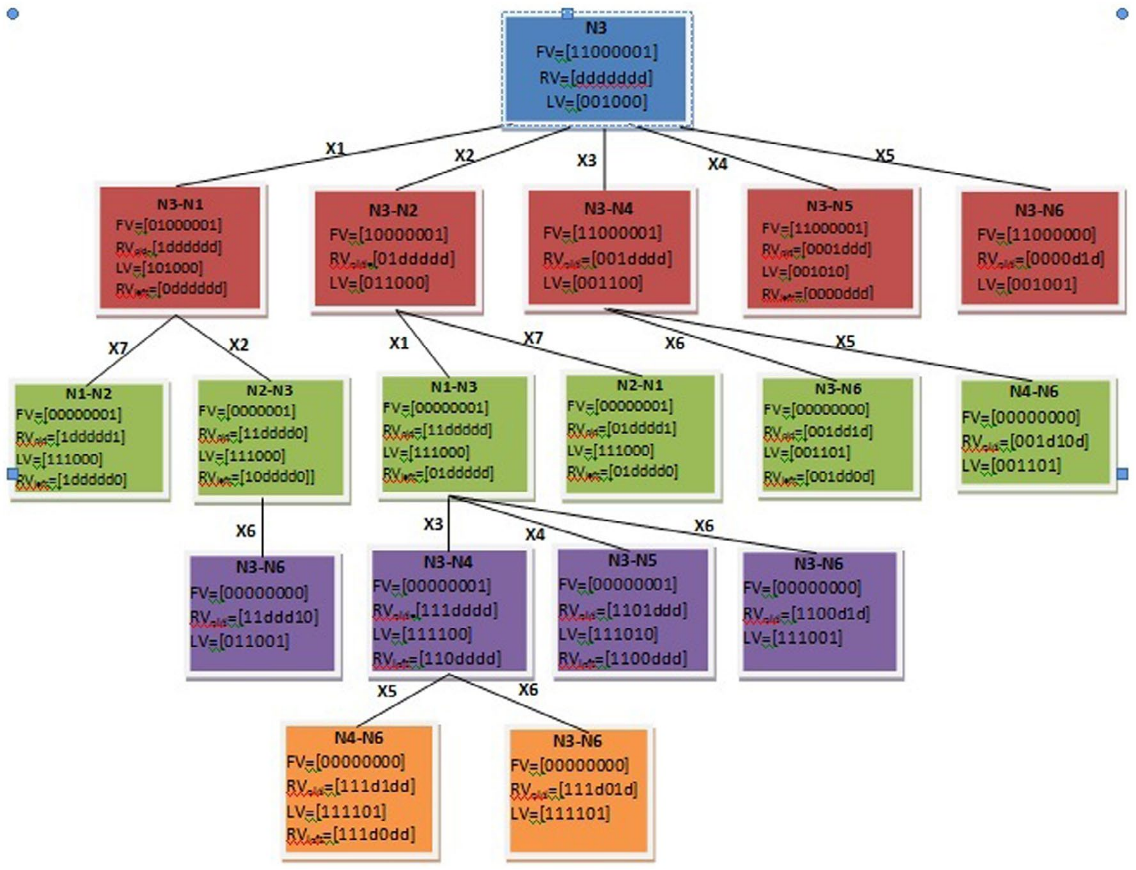

Figure 5. An execution tree for reliability of sub-system 1.

respectively for each components. Thus from Figure 4, we can say that the temperature sensors needs file F_1 and program P_1 for successful execution.

Table 1 summarizes the acronyms used in Figure 4.

Table 2 denotes which node in Figure 4 contains which files and run which programs.

To find out the reliability of the intelligent fire alarm system which is denoted as ISR_1, we can apply the algorithm discussed in the previous section. In the following section we have given the demonstration of how it can be applied to calculate the reliability and the detailed execution tree for program 3 starting at node 3 (analyzer) is shown in Figure 6.

Initial File Usage Vector (FUV) values (P_3 at Node 3): 11000001.

In Figure 6, the root has FUV_3 = [11000001] and the link x_1 connects it to N_1. At N_1, F_1 is locally available hence FUV at box N_3–N_1 will become FUV = [01000001].

Similarly while moving from node 1 to node 2, file F_2 is locally available. Hence, FUV at box N_1–N_2 through link x_7 becomes FUV = [00000001] (see Tables 3 and 4).

Initial Reliability Vector (RV) values (P_3 at Node 3): ddddddd.

For updating RV, we can apply the rules mentioned in Section 4.2.3.

Box N_3–N_1 in the Figure 6 is the first child of node N_3. Therefore RV_{old} = [1dddddd] as link x_1 connects N_3 to N_3–N_1.

Box N_3–N_2 is the second child of N_3 or first sibling of N_1. Therefore RV_{old} = [01ddddd] and RV_{left} = [00ddddd].

Initial Path Vector (PV) values (P_3 at Node 3): 001000.

PV for node 3 = [001000]. For the box N_3–N_1, PV = [101000].

5.3. IoT sub-systems 2: smart healthcare

Smart healthcare functionality is described as a process with the collection of devices which are: Glucometer, Blood pressure monitor, Weight scale, Medical experts, a patient, and a health server. These devices, periodically receiving real-time information from the patient when the person comes for a regular checkup. The data generated after the check up by each of the devices is sent to the health server. The health server in turn has programs that check if there is any anomaly in the data like if the blood pressure is above normal or blood sugar level is at a level

Table 1. Descriptions and requirements of files by various programs presented in Figure 4.

Acronyms	Description	File Needed
FA	Files available locally in the node	
FN I	Files needed by program I	
PA	Programs available	
F_1	Sense temperature details	
F_2	Sense smoke details	
F_3	Analysis done for temperature sensor	
F_4	Analysis done for smoke sensor	
F_5	Analysis done for water tank	
F_6	Log files of sprinkler	
F_7	Log files of alarm	
F_8	Water level details	
P_1	Take temperature input	FN_1: F_1
P_2	Take smoke input	FN_2: F_2
P_3	Program to check whether temperature above or below threshold	FN_3: F_1, F_2, F_3
P_4	Program to check whether smoke above or below threshold	FN_4: F_1, F_2, F_4
P_5	Program to check whether water level above or below threshold	FN_5: F_5, F_8
P_6	Activate and Run the sprinkler	FN_6: F_3, F_4, F_6
P_7	Activate and Run the alarm	FN_7: F_3, F_4, F_7
P_8	Program to stop water supply	FN_8: F_5, F_8

Table 2. Programs and prepared files in different nodes of sub-system 1.

Nodes	1	2	3	4	5	6
Programs	P_1	P_2	P_3, P_4, P_5	P_6	P_7	P_8
Files	F_1	F_2	F_3, F_4, F_5	F_6	F_7	F_8

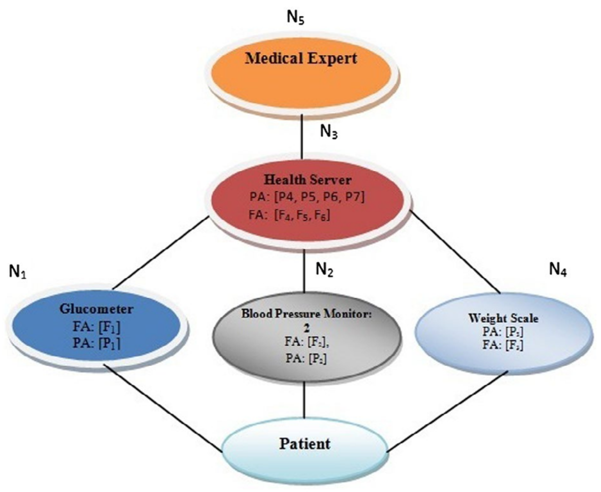

Figure 6. Detailed structure of IoT subsystem 2.

higher than that considered safe. After this check up, the data for the patient is sent to the 'medical experts.' Here, all the information is processed and the related and necessary medical actions are taken by the medical experts and the information is conveyed to the patient.

The detailed execution tree for program 4 starting at node 3 (health server) is shown in Figure 7.

Table 3. Summarizes the acronyms used in Figure 6.

Acronyms	Description	File Needed
FA	Files available locally in the node	
FN I	Files needed by program I	
PA	Programs available	
F_1	Measure glucose details	
F_2	Measure blood pressure details	
F_3	Measure weight scale details	
F_4	Log files of glucometer	
F_5	Log files of blood pressure monitor	
F_6	Log files of weight scale measurement	
F_7	Patient complete details	
P_1	Take glucose level as input	FN_1: F_1
P_2	Take blood pressure as input	FN_2: F_2
P_3	Take weight as input	FN_4:F_3
P_4	Program to check whether blood pressure above 140 mm Hg	FN_3: F_2
P_5	Program to check whether glucose level above 100	FN_3: F_1
P_6	Program to check whether weight satisfies the height-weight-gender relation	FN_3: F_3
P_7	Update data for the patient	FN_3: F_1, F_2, F_3

Table 4. Programs and prepared files in different nodes of sub-system 2.

Node	1	2	3	4	5	6
Prog	P_1	P_2	P_4, P_5, P_6, P_7	P_3		
Files	F_1	F_2	F_4, F_5, F_6	F_3	F_7	

6. Result and discussion

IoT system reliability can be evaluated using algorithm 1 as has been entailed in Section 4.2.4. Therefore, for any sub-system *j*, the IoT service reliability can be depicted as $ISR_j = \sum_{j=1}^{Nt} pr\{FRV[j]\}$. Each row in FRV contains 1 sub network that satisfies the file requirements for the given programs. Table 2 shows the program availability at each node along with required files and programs of each sub-system. The CHISS structure for program P_3 at node 3 for IoT sub-system 1 has been shown in Figure 4. The corresponding execution tree is shown in Figure 6. The reliability expression of the aforementioned sub-system extracted from Figure 6 can be represented as:

$$\begin{aligned} \mathrm{ISR}_1 &= q_1q_2p_3p_6 + q_1q_2p_3p_5 + p_1p_2q_3q_4p_6 \\ &\quad + p_1p_2p3p_5 + p_1p_2p_3q_5p_6 \end{aligned} \tag{6}$$

The reliability expression of program P_4 at node 3 for sub-system 2 can similarly be extracted from the corresponding execution tree depicted in Figure 7 as follows:

$$\mathrm{ISR}_2 = p_1 + q_1p_2p_5p_6 + q_1q_2p_3p_4p_6 \tag{7}$$

The obtained results for ISR_1 and ISR_2 are 0.7452 and 0.98019, respectively, with the assumed values of $p = 0.9$ and $q = 0.1$ as has been presumed in Section 4.2.

For evaluating the overall system reliability, we used six different components and its success and failure percentages varied from 0.98 to 0.88 and 0.02 to 0.12, respectively.

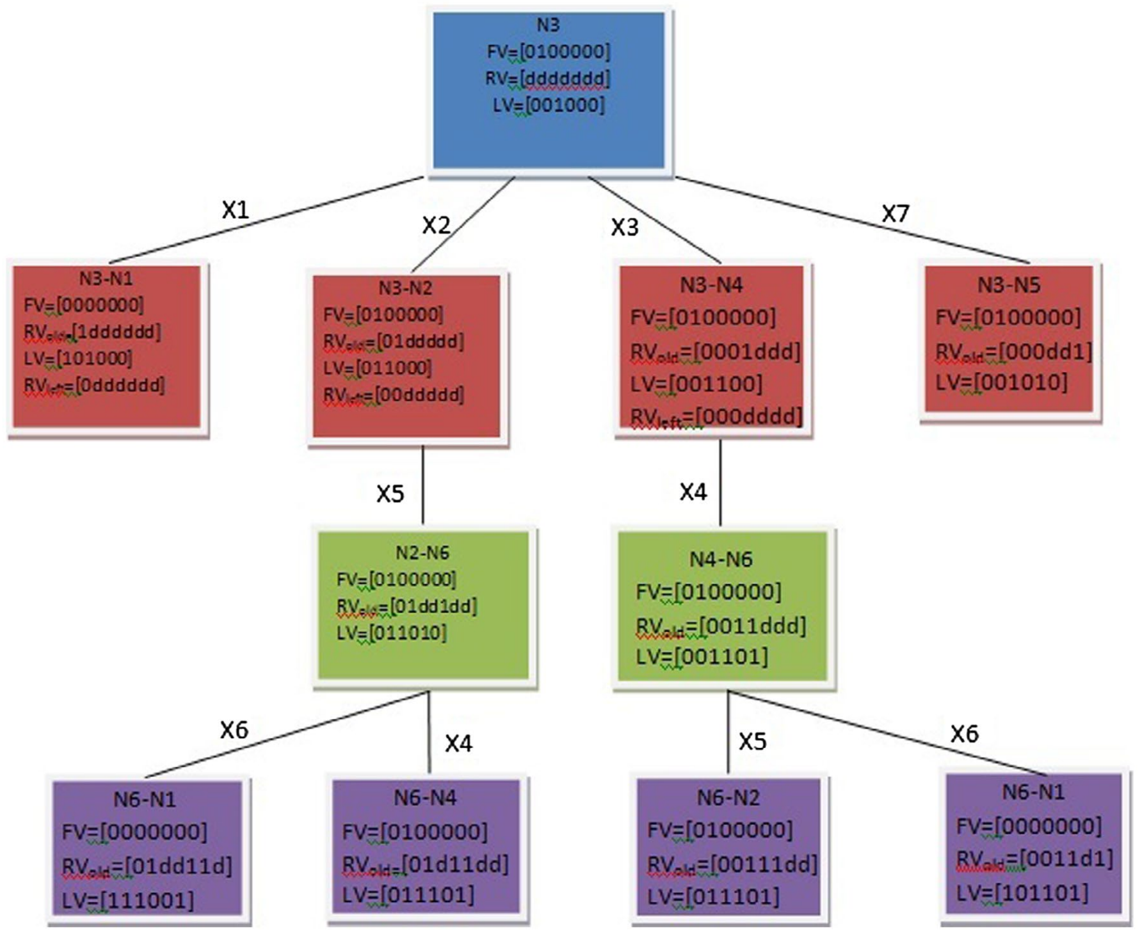

Figure 7. An execution tree for reliability of sub-system 2.

Based on these components success and failure percentages ISR_1 and ISR_2 are calculated using above formula (6) and (7), respectively. As IoT systems are subjected to change its state and in order to build the model more realistic to IoT, we set percentage of success of six components in decreasing from 0.98 to 0.88 and percentage of failure of components increasing from 0.02 to 0.12. Table 5 depicts the details of components success and failure percentage.

Assuming that $P(ISR_1)$, $P(ISR_2)$, …, $P(ISR_n)$ represent the probability of successful operation of n IoT sub-systems and $\overline{P(ISR_1)}$, $\overline{P(ISR_2)}$, …, $\overline{P(ISR_n)}$ their respective unsuccessful operation probability and the fact that the failure of these sub-systems are uncorrelated (independent of each other), we have evaluated the reliability of aggregate IoT sub-system accounting for the availability of needed files and programs as:

$$ISR = 1 - \left[1 - PISR\left(ISR_1\right)\right] x \left[1 - P\left(ISR_2\right)\right]$$

ISR = 1–[1–0.8790522752]*[1–0.9958152192] = 0.999493860283453.

Table 5. ISR Result for different values of P and Q.

P: Probability of success	Q: Probability of failure	ISR_1	ISR_2	ISR
0.98	0.02	0.879052275	0.995815	0.999494
0.96	0.04			
0.94	0.06			
0.92	0.08			
0.9	0.1			
0.88	0.12			

Hence, putting the values of ISR, P_{pr} and P_f in Equation (3), the service reliability,

$R(s) = 0.999493860283453 \times 1 \times 1 = 0.999493860283453$, assuming that required programs and files are always available.

Thus, we arrive at IoT system reliability with two sub-systems following the methodology presented in this paper. The method presented in this paper is a generalized one that can be applied to any service-oriented IoT.

7. Conclusions

This paper presents a novel reliability model for Internet of Things that are composed using service-oriented paradigm. To that end, a detailed framework, namely the Centralized Heterogeneous IoT Service System (CHISS) for IoT systems has been employed where sensing functions are executed in distributed fashion and control functions are done in a centralized fashion using state-of-the-art cloud data centers. This paper also proposes a generalized reliability assessment algorithm for CHISS-based IoT system. To demonstrate the application of this reliability model and reliability assessment algorithm, two case studies of CHISS-based IoT have been employed in this paper. We seek to extend this work in future by relaxing certain assumptions that have been employed in this paper regarding availability of distributed files and programs.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

Ranjit Kumar Behera was born in India and received his BTech and MTech from Biju Patnaik University of Technology, ODISHA in 2004 and 2010, respectively. He is currently with the National Institute of Science and Technology, Berhampur, India as an assistant professor. His research interests include reliability modeling, IoT and cloud computing.

K. Hemant Kumar Reddy was born in India and received his MTech and PhD from Berhampur University in 2008 and 2014, respectively. He is currently with the National Institute of Science and Technology, Berhampur, India as an assistant professor. His research interests include grid computing, cloud computing, Fog Computing, IoT and service-oriented architectures, and optimization in engineering.

Diptendu Sinha Roy was born in India. He received his BTech from Kalyani University in 2003 and subsequently his MTech and PhD from Birla Institute of Technology, Mesra, India in 2005 and 2010, respectively. He is currently with the National Institute Technology, Meghalay, India as an associate professor. His research interests include distributed, grid computing, cloud computing, IoT, Edge computing, software reliability, optimization in engineering. He also works towards design and analysis of distributed infrastructure of power systems.

ORCID

Ranjit Kumar Behera http://orcid.org/0000-0002-7408-4900
K. Hemant Kumar Reddy http://orcid.org/0000-0003-2492-3312

References

[1] Ashton K. That ‘Internet of Things’ thing. RFiD J. 2009;22(7):97–114.
[2] Akyildiz IF, Su W, Sankarasubramaniam Y, et al. Wireless sensor networks: a survey. Comput Netw. 2002;38(4):393–422.
[3] De S, Barnaghi P, Bauer M, et al. Service modelling for the Internet of Things. 2011 Federated Conference on Computer Science and Information Systems (FedCSIS). IEEE; 2011.
[4] Dai Y-S, Xie M, Poh KL, et al. A study of service reliability and availability for distributed systems. Reliab Eng Syst Saf. 2003;79(1):103–112.
[5] Dai Y-S, Yang B, Dongarra J, et al. Cloud service reliability: modeling and analysis. 15th IEEE Pacific Rim International Symposium on Dependable Computing; 2009.
[6] Dai Y-S, Pan Y, Zou X. A hierarchical modeling and analysis for grid service reliability. IEEE Trans Comput. 2007;56(5):681–691.
[7] Lin, Y-K, Huang C-F. Assessing reliability within error rate and time constraint for a stochastic node-imperfect computer network. Proc Inst Mech Eng Part O: J Risk Reliab. 2013. DOI:10.1177/1748006X12468685.
[8] Zhang Y, Xu Z, Wang X, et al. Single minimal path based backup path for multi-state network. Proc Inst Mech Eng Part O: J Risk Reliab; 2014;228(2):152–165. DOI:10.1177/1748006X13502953
[9] Papazoglou MP, Traverso P, Dustdar S, et al. Service-oriented computing: state of the art and research challenges. Computer. 2007;40(11):38–45.
[10] Spiess P, Karnouskos S, Guinard D, et al. SOA-based integration of the internet of things in enterprise services. IEEE International Conference on Web Services, 2009. ICWS 2009. IEEE; 2009.
[11] Guinard D, Trifa V, Karnouskos S, et al. Interacting with the SOA-based internet of things: discovery, query, selection, and on-demand provisioning of web services. IEEE Trans Serv Comput. 2010;3(3):223–235.
[12] Wei Q, Jin Z, Li L, et al. Lightweight semantic service modelling for IoT: an environment-based approach. Int J Embedded Syst. 2016;8(2/3):164–173.
[13] Cañete E, Chen J, Díaz M, et al. A service-oriented approach to facilitate WSAN application development. Ad Hoc Netw. 2011;9(3):430–452.
[14] Shehu, UG, Ali Safdar G, Epiphaniou G. Network aware composition for internet of thing services. Trans Netw Commun. 2015;3(1):45.
[15] Li L, Jin Z, Li G, et al. Modeling and analyzing the reliability and cost of service composition in the IoT: a probabilistic approach. 2012 IEEE 19th International Conference on Web Services (ICWS); IEEE; 2012.
[16] Liu A, Zhang Q, Li Z, et al. A green and reliable communication modeling for industrial internet of things. Comput Electr Eng. 2017;58:364–381.
[17] Zhang F. Research on reliability analysis of computer network based on intelligent cloud computing method. Int J Comput Appl. 2017;1–6.
[18] Li G, Liu Q, Bai S. A novel model to improve network performance. Int J Comput Appl. 2017;1–6.
[19] Li X, Xu G, Li L. RFID based smart home architecture for improving lives. ASID 2nd International Conference; 2008.
[20] Buckley J, editor. The Internet of Things: from RFID to the next-generation pervasive networked systems. New York (NY): Auerbach Publications; 2006.
[21] Giusto D, Iera A, Morabito G, et al, editors. The Internet of Things. Springer; 2010. ISBN 978-1-4419-1673-0.
[22] Gubbi J, Buyya R, Marusic S, et al. Internet of Things (IoT): a vision, architectural elements, and future directions. Fut Gener Comput Syst. 2013;29(7):1645–1660.

[23] Bellavista P, Cardone G, Corradi A, et al. Convergence of MANET and WSN in IoT urban scenarios. IEEE Sens J. 2013;13(10):3558–3567.

[24] Bonomi F, Milito R, Zhu J, et al. Fog computing and its role in the internet of things. Proceedings of the first edition of the MCC workshop on Mobile cloud computing, ACM; 2012.

[25] Reddy KHK, Sinha RD. DPPACS: a novel data partitioning and placement aware computation scheduling scheme for data-intensive cloud applications. Comput J. 2015;59(1):64–82.

[26] Caceres R, Friday A. Ubicomp systems at 20: progress, opportunities, and challenges. IEEE Pervasive Comput. 2012;11(1):14–21.

[27] Hillier FS, Lieberman GJ, Hillier F, et al. MP introduction to operations research. 2004.

[28] Kumar A, Agrawal DP. A generalized algorithm for evaluating distributed-program reliability. IEEE Trans Reliab. 1993;42(3):416–426.

[29] Kumar, A, Agrawal DP. Computer network reliability evaluation from application's point of view. Proceedings of the Second IEEE Symposium on Parallel and Distributed Processing, 1990. IEEE; 1990.