Service Architecture Models For Fog Computing: A Remedy for Latency Issues in Data Access from Clouds

Adnan Khalid[1] and Muhammad Shahbaz[2]
[1] Government College University, Lahore, Pakistan
[e-mail: adnan.khalid@gcu.edu.pk]
[2] University of Engineering and Technology, Lahore, Pakistan
[e-mail:m.shahbaz@uet.edu.pk]
*Corresponding author: Adnan Khalid

Received November 10, 2016; revised February 14, 2017; accepted February 15, 2017; published May 31, 2017

Abstract

With the emergence of the Internet of Things (IoT) the world is projecting towards a scenario where every object in the world (including humans) acts as a sender and receiver of data and if we were to see that concept mature we would soon be talking of billions more users of the cloud networks. The cloud technology is a very apt alternative to permanent storage when it comes to bulk storage and reporting. It has however shown weaknesses concerning real-time data accessibility and processing. The bandwidth availability of the cloud networks is limited and combined with the highly centralized storage structure and geographical vastness of the network in terms of distance from the end user the cloud just does not seem like a friendly environment for real-time IOT data. This paper aims at highlighting the importance of Flavio Bonomi's idea of Fog Computing which has been glamorized and marketed by Cisco but has not yet been given a proper service architecture that would explain how it would be used in terms of various service models i-e IaaS, PaaS and SaaS, of the Cloud. The main contribution of the paper would be models for IaaS, PaaS and SaaS for Fog environments. The paper would conclude by highlighting the importance of the presented models and giving a consolidated overview of how they would work. It would also calculate the respective latencies for fog and cloud to prove that our models would work. We have used CloudSim and iFogSim to show the effectiveness of the paradigm shift from traditional cloud architecture to our Fog architecture.

Keywords: Fog Computing; Service Architecture; Cloud Computing, Internet of Things, Distributed computing, Grid computing

1. Introduction

Computer Science has been one of the most rapidly evolving disciplines of study. While other sciences have epochal landmarks of development spread over centuries, computer science evolves in a matter of years and hence it is essential that such advances are keenly observed and developed both conceptually and theoretically to reap their benefits. The early computers were huge units of hardware connected together, spanning over several rooms but able to perform only a particular task. From the infamously famous vacuum tubes that were inevitable for conduction of electricity yet miserably poor at it, to the tiniest of microchips that now are being embedded in the smallest of gadgetry, we indeed have moved a far way forward in a span of less than a century.

1.1 Cloud computing: A ground-breaking phenomenon

As computers spread like wild fire around the world, aiding in innumerable areas of application, it became immensely important to keep a check on resource allocation, resource consumption and cost of storage and compute. The increasing personal use of computers for socializing, entertainment, manufacturing, business and so forth meant that huge infrastructures for storage and compute needed to be established and afforded. This in-turn meant that many small businesses were either incapable of fully reaping the benefits of technology to enhance their earnings and profits or were highly dependent on storage and platform providers for provision of resources. This also meant that once acquired a resource had to be purchased according to the peak usage requirement and hence one would be paying more than the needed consumption. The remedy to these problems caused by fixed storage and compute resources came in the form of a phenomenon called “Cloud Computing”. The term rapidly caught attention of the masses and became a household name.

1.1.1 The Cloud and its promises

Cloud computing is perhaps the most talked about technological advancement in the field of computing in the recent past. With the cloud emerging as the next in-thing many have tried to explain what cloud computing is. The expectations from cloud computing as the re-shaper of IT industry have caused many commentators to associate every possible service available in the computing world with the cloud [1]. If we were to define cloud computing in the light of the not-so-attractive definition by NIST that is more of an essay than a definition, we would go as follows [2],

“Cloud computing refers to the computing technology whereby resources (both hardware and software) are shared over a network typically the internet ensuring that the end user receives a highly scalable infrastructure on a pay as you go basis. The end user of the cloud may be a software-as-a-service (SaaS) or platform-as-a-service (PaaS) provider or a direct user of infrastructure. The core of cloud computing, however, lies in the fact that the infrastructure i.e. the hardware and or the platform is leased, flexible and scalable and is paid on a per usage basis.”

According to our definition, the cloud displays properties of scalability and flexibility of usage and payment of data, compute and storage resources.

1.2 Cloud computing and real-time access and processing of data

It is often the case with technologies that become house hold names that businesses use them as taglines to enhance sales and saleability of their product. The term “cloud” emerged as a unique selling point in numerous appliances ranging from cell phones and laptops to smart TVs and air conditioners. It the glitz and glamour that advertisers bestowed upon cloud computing, a few major issues remained over looked mainly because of no better alternatives.

With more applications shifting from traditional computing to cloud the overall use of bandwidth in the global cloud data centres now touches unprecedented heights. The misconception that the cloud is an infinite pool of storage space is also fast being cleared. With a hoard of real-time applications now looking to use cloud for data processing we often find that with the centrally distributed nature of the cloud data centres the delay in processing and deliverance of results is critically slow and detrimental to the whole concept of real-time applications.

1.2.1 Internet of Things (IOT) and the Cloud

Internet of Things is another catch phrase that is fast making its way to the common household with cellular companies and online businesses constantly using it as a unique selling point. What the IOT basically promotes is the concept of everything acting as receiver and sender of data. This means that gradually humans, computers, machines in general, handheld devices, manufacturing plants, buildings, power grids, vehicles and numerous other end nodes would be self-sufficient as senders and receivers of data and would communicate with the cloud networks directly for real-time and bulk processing [3].

Internet of things is an object based scenario in which objects or people are provided with unique identifiers and the ability to transfer data over a network without requiring human-to-human or human- computer intercommunication [4]. IoT has extended synchronizing wireless technologies, micro-electromechanical systems (MEMS) and the Internet [5].

1.2.2 Limitations of Cloud Computing in wake of the IOT boom

As is the case with every technology or for that matter every luxury that man enjoys, there is an opportunity cost associated with the flexible and highly scalable nature of cloud networks [6]. Since the inherent nature of the cloud is geographically segregated and far away for the end user and connectivity is through an IP based network, the data access and processing experiences a natural lag. This delay, although not significant for many applications can be hazardous for time-critical applications where a delay of millisecond can mean a risk of life. Also with the future evidently heading the way of Internet of Things we can safely predict a huge rise in the number of cloud users which would in turn cause problems in bandwidth availability and subsequently increase the lag that is experienced by the real-time application end user. The only suitable way forward seems in the direction of distributing the storage and processing of clouds into a structure that allows separate storage and processing of real-time data while also catering for bulk storage needs and analytics[7]. This structure has been coined as the Fog by Flavio Bonomi and has been endorsed by CISCO [8].

1.3 Our main contribution and the novelty of this research

Since the cloud has several limitations when it comes to dealing with real-time applications,

Fog computing is fast going to emerge as a suitable extension to the traditional clouds and will help eradicate issues of latency and network delays etc. This work simulates cloud and Fog environments to establish results that show the efficacy of the new Fog paradigm as compared to the traditional cloud networks.

2. Here comes the FOG!

Flavio Bonomi coined the term FOG computing for a distributed paradigm that allows the split between real-time data processing and bulk data processing. Fog computing is an extension of the well-employed technology cloud computing. It outspreads the resources provided by clouds at the edge of the network, nearby and closer to the user [9]. The purpose of extending the clouds to fog is to reduce latency issues faced by real time applications. The problem is that since the cloud data centers store data at the center of the cloud, the time of processing for wait-sensitive data is often more than what would ideally be appreciable. This problem can be solved by introducing a Fog layer that comprises of highly virtual machines with ephemeral storage capabilities and in close proximity to the end receivers.
A basic comparison of Fog computing with cloud computing would show how the future of Internet of Things and real time processing lies in the adoption of Fog computing.

Table 1. Comparison of Cloud Computing and Fog Computing)

	Cloud Computing	**Fog Computing**
Target User:	Common Internet users.	Mobile users
Service Type:	Global information is collected world-wide	Partial localized information services linked to exact deployment locations
Hardware:	Abundant and scalable storage space & compute power	Ephemeral, restricted storage, compute power & wireless interface.
Distance from Users:	Far-away from consumers and interconnected through IP Network.	In the physical contiguity and interconnected through single hop wireless connection
Working Environment:	Ware-house-size building with central air conditioning systems	Outdoor (parklands, streets etc.) or indoor (shopping malls, restaurants, etc.)
Deployment:	Maintained and Integrated by Google, Amazon, etc.	Distributed or Integrated in regional areas by native business (shopping Mall retailer, local telecommunication vendor etc.)

Latency	High	Low
Delay Jitter	High	Very Low
Location of Service	Within the Internet	At the edge of the local network
Distance Between Client and Server	Multiple Hops	Single Hop
Security	Undefined	Can be defined
Attack on data en-route	High probability	Very low probability
Location Awareness	No	Yes
Geo-distribution	Centralized	Distributed
No. of server nodes	Few	Very large
Support for mobility	Limited	Supported
Real-time interactions	Supported	Supported
Type of last mile activity	Leased line	Wireless

The table above is the most comprehensive comparison of Fog computing with its predecessor Cloud computing. We have tried to include as many parameters as we could think of in order to give the readers a fair picture of how big a proper extension of the cloud services to Fog can be in terms using of real-time Internet of Things applications.
Let us explain a few differences in greater detail so that the "Fog over the FOG is a little clearer".

2.1 Hardware

Fog devices are sensors and actuators whose virtual machines are used to store real-time data for ephemeral storage. This means that a fog device would store data temporarily for processing and after the results are obtained the data would be deleted. The transmission of data would be wireless in nature and would use an intranet and not the Internet. In comparison to this the cloud devices are used for bulk storage and hence are large, scalable centres of data collection available over the Internet.

2.2 Distance from the user

Since the entire issue behind the emergence of Fog computing is latency, it is obvious that the distance of the processing unit from the end user would be of paramount importance. The Fog layer is a specialized layer between the end user and the internet and its proximity is

geographically nearer to the end user. In fact the Fog devices are placed in an intranet that is bound in a geographical area and hence devices closer to that area would have rapid access to the services deployed on Fog devices. [10]

2.3 Deployment

The cloud services are provided, maintained and integrated mainly by giants like Google and Amazon and that means a certain amount of monopoly over the entire technology. Also with the control of cloud services being so centrally controlled by a big organization the access to these services is not conducive to the demands of the IoT in terms of high speed communication and processing.

2.4 Recovery Schemes for Fog computing failure

Fog computing paradigm comes with the danger of downtime of the Fog network as a result of hardware failure, software failure or Fog resource overflow. Two schemes are presented by Dimas Satria et al. to cater these problems. [11]Fog computing is based on bringing services to the edge of the network. If properly described a Fog network is a localized network of devices in a restricted geographical area that can be accessed by end devices that can quickly access and process data without having to interact with the Internet.

The shift from a traditional cloud based setup to a distributed Fog environment is going to aid fast processing of time-critical data. One example of the difference that this shift can cause is in the field of medicine.

Body area networks for Non-communicable diseases

A prime use of the above mentioned deployment is the use of sensors and actuators for generating signals from the brain when a stroke is detected in a human body. If a Fog network is deployed within the confines of a human body i.e Fog devices form a body area network then we can quickly process any dangerous signal from the body and send it to a response mechanism. This is a proposed area that we feel is the future of both health sciences and cognitive sciences.

2.5 Workload balancing and scheduling in the Fog environment

Workload balancing is one of the major concerns that arises when we decide to shift from a cloud environment to a Fog environment and there is considerable commendable work that is available in order to achieve a balance between the data that resides on clouds and the one that is tranferred to Fog. Deng et al. provides a very comprehensive workload optimiazation mechanism in their work in this domain. [12] In another paper they also address consumption delay tradeoffs resulting from effcient load balancing.[13]

2.5.1 Internet of Things (IoT) and the Cloud

The advent of the Internet of Things concept has rendered the Cloud storage mechanism as inefficient as the number of nodes interacting with the Cloud has increased exponentially. This puts a huge amount of burden on the Cloud network and exposes the limitations of bandwidth available to Cloud networks. This calls for data profiling through which data can be identified as critical, latency sensitive and “hot”. The Fog computing paradigm is a great leap towards a truly IOT driven environment as discussed by various researchers[14]. Datta et al. have given an architecture to enable consumer centric IOT[15]

2.5.2 Data Profiling

In order to migrate the right amount and kind of data from Cloud to Fog and vice versa we need to differentiate data according to some parameters. In case of Fog computing the main concern with regards to access and processing of data is latency. This means that one parameter that must be considered when decided what data to move to the Fog layer would be latency sensitivity. Another aspect of data that needs consideration in this regard is its temperature.

2.5.2.1 Data Temperature

Data can be deemed "hot" or "cold" depending on its usage. A "hot" data would typically be data that is to be processed within a given time frame. As the time passes this "hot" data becomes less critical and hence its temperature is said to have cooled down resulting in the term "cold" data. At a particular instance the determination of "hot-spots" in the stored data can be done by IO profiling whereby IOPS-intensive data is shifted to the faster Fog layer. In this regard a key challenge arises from the fact that hot-spots in data continue to move over time i.e. previously cold data that is seldom accessed would suddenly or gradually become hot due to it being frequently accessed or becoming performance critical in response to a certain event. Such fluctuations in the nature of data require us to devise an adaptive mechanism that can assess the future needs of the system and migrate appropriate chunk of data from Cloud to Fog and vice versa.

2.5.2.2 Deadline

Data selection for migration requires preemptive determination of hot or latency-sensitive data. Such determination requires screening data for certain indications such as criticality and IOPS-sensitivity. One major aspect of this screening is the determination of a reasonable deadline within which the migration has to be completed in order to meet the deadline of the workload to which the "hot" data is associated.

2.5.3 Energy and delay management and scheduling

To minimize the energy consumption in the Fog environment we need to perform the following actions, (i) admission control; (ii) dispatching of the admitted workload; (iii) flow control of the inter-VM TCP/IP connections; (iv) queue control; (v) up/down scaling of the processing frequencies of the instantiated VMs; and, (vi) adaptive joint consolidation of both physical servers and TCP/IP connections[16]. All these can be managed by a very accomplished scheduler called the Q* scheduler [17].

3. Our Service Architecture for Fog computing

Although there has been much talk on the importance of Fog computing for IoT applications and different areas of use are identified there is still no service architecture that has been presented for the establishment of Fog Networks. The cloud service architecture for Infrastructure as a Service, Platform as a Service and Software as a Service show a pathway as to how models can be made for Fog services. In this paper we are going to make an effort to provide service architecture models that would guide any future deployment of Fog Services.

3.1 Infrastructure as a Service (IaaS)

We provided an IaaS model for cloud computing in a paper published in 2013 that at that time gave a guideline as to how Infrastructure could be acquired by a cloud user. The cloud IaaS that we gave was as follows,

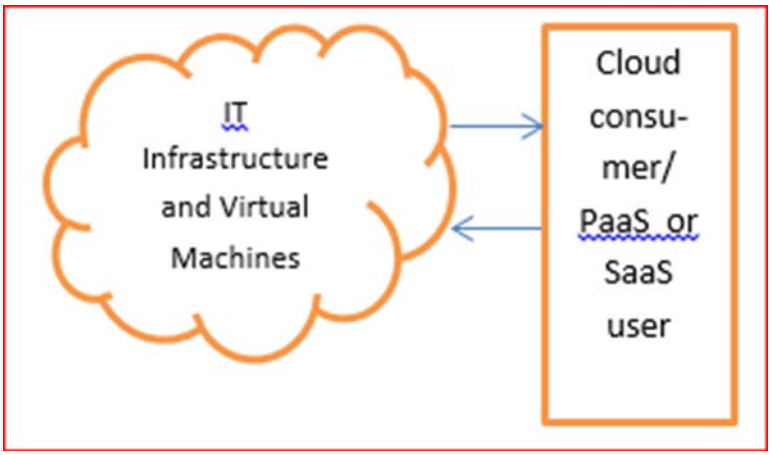


Fig. 1. Infrastructure as a Service (Cloud)) [10]

Now with the advent of Fog computing the IaaS for Fog would have to be presented in a distributed structure that would ensure a split between real-time data processing and analytical processing also insuring bulk storage and ephemeral storage of data separately. Our proposed model is as follows and it is the first of its kind.

Fig. 2 clearly shows that with the highly real time nature of use of the IoT applications the Fog services would have to insure the availability of two kinds of networks for storage. The Fog layer would provide ephemeral storage that would mean that the real time latency sensitive data would be stored temporarily and processed quickly with a response time that would make sure that the access is truly real time.

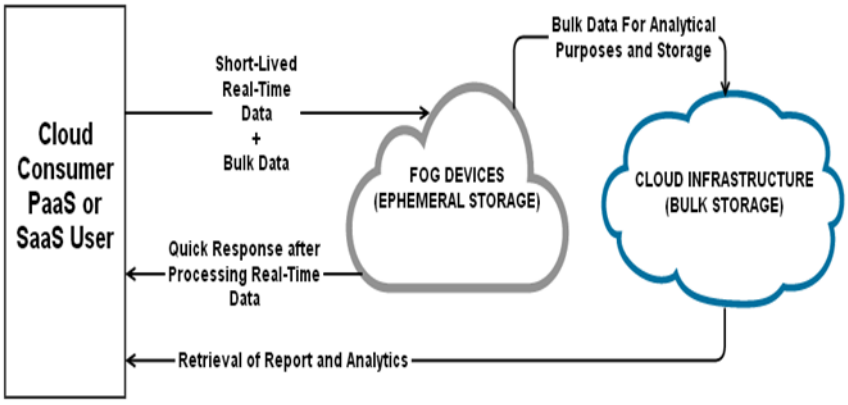


Fig. 2. Infrastructure as a Service (IaaS) for Fog)

A cloud layer would support bulk storage of data and would have heavy storage devices capable of storing large amounts of data. This data would also be used for generating reports for analytical purposes.

3.2 Platform as a Service (PaaS)

In the paper in 2013 we proposed the following PaaS model that gave a view of how simple cloud services would be accessed by users. The model looked as follows,

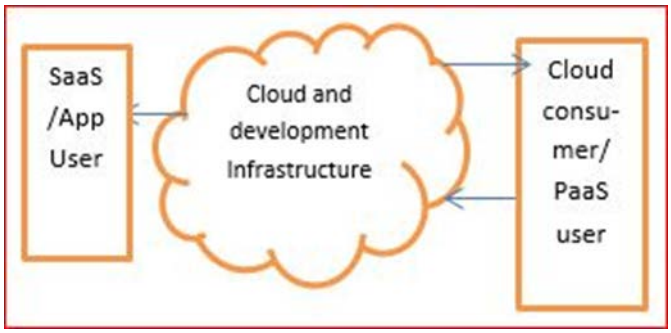


Fig. 3. Platform as a Service (PaaS) for Cloud [10]

The nature of Platform as a Service would also have to be distributed for development of both real-time and analytical parts of the application. The application developer would have to use the PaaS in a way that the real-time modules of the applications are built on the Fog Devices and the analytical modules are built using the platforms available at the cloud datacentres.

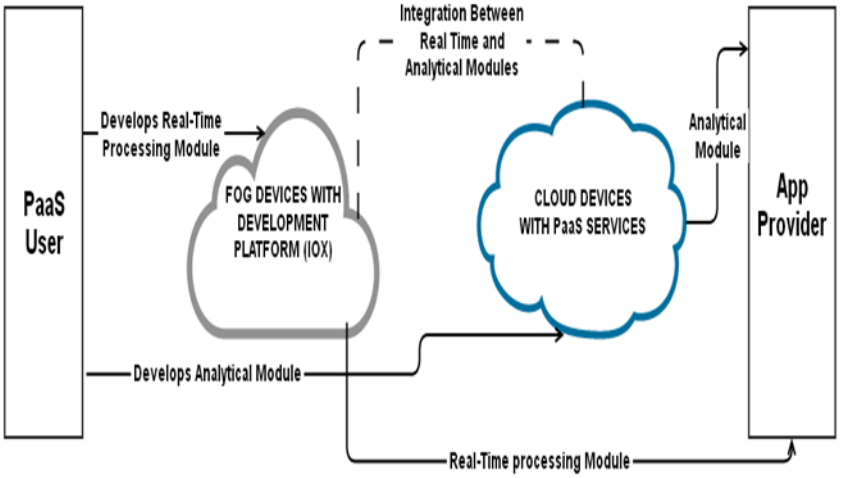


Fig. 4. Platform as a Service (Paas) for Fog

Fig. 4 shows how application developers would have to use two platforms to make real-time applications available to the end users. The application developer called the PaaS user in the **Fig. 4** would construct the real-time processing module of the application using IOX based platform available at the local intranet of Fog layer. This module will process all the latency sensitive data that can be either tagged or intelligently identified depending on the nature and scope of the application. The developer or PaaS user will develop an analytical part of the application for bulk storage and analysis of data on the PaaS facilities available over the traditional Internet based cloud. An integration team would bridge the local modules at the Fog layer with the modules at the Internet based cloud layer to provide connectivity between

the Fog and the Cloud. An application provider who provides the application as Software as a Service (SaaS) will have to access both the real-time and analytical parts separately, in fact there may be separate application providers for both parts. We are however assuming a single vendor for application provision in this particular model.

3.3 Software as a Service (SaaS)

Similar to the above two models we gave a model for SaaS in clouds in 2013 as well. A lot of models for SaaS have been presented before and after the model we gave but we take pride in simplifying the way things work in clouds and now in Fog and hence this model was in its time the simplest you could find.

Fig. 5. Software as a Service (SaaS) for Clouds [10]

As is evident from the **Fig. 5** the cloud based SaaS is very straight forward and there is nothing too fancy to explain now that the Cloud is a well read and well-practiced domain. However, when one shifts to Fog and considers a distributed environment where the Software will have to be provided through two different services one from the Cloud network via Internet and the other from a local network via an Intranet connected wirelessly, a more complex model needs to be presented. I hope this model, along with the two above it would seem quite simple to someone looking at them three years from now because that sort of familiarity and acceptance would mean that our work was worth the effort.

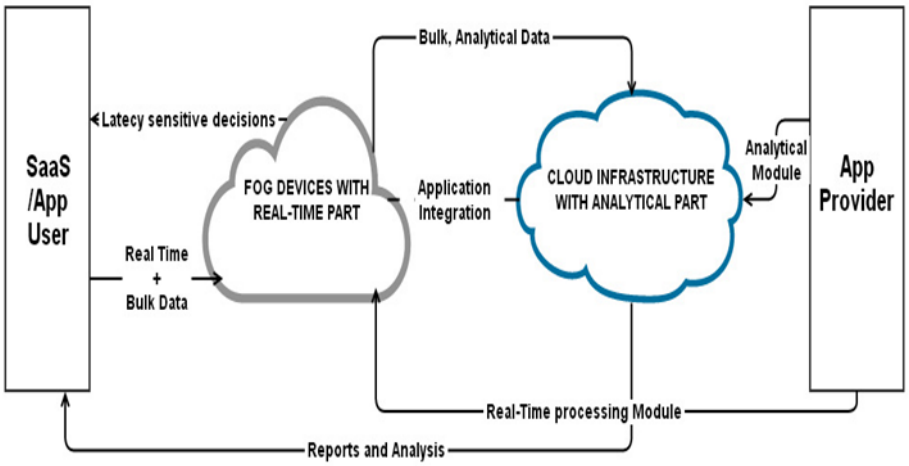


Fig. 6. Software as a Service (SaaS) for Fog

Fig. 6 shows that the application provider would provide the real-time modules of application at the Software-as-a-Service portals available at the Fog networks. The application provider can either provide a downloadable real-time module that can be deployed at the Fog devices or have a portal hired at the Fog devices. Both these formulae can be used to provide a distributed access to the applications. The analytical processing module of the applications would be accessible from the cloud infrastructure on which the application provider would deploy it on the cloud infrastructure.

4. Our main performance metric – Service Latency

When an application runs on an IOT device its service latency is basically its response time, and is calculated as the sum of the transmission latency and the processing latency that occurs as the consequence of the request. Let us assume that communication between multiple FIs (Fog instances) at tier 2 and between different cloud data-centers at tier 3 takes place over high-bandwidth channel hence leading to very little delay.

If d_{tf} and d_{fc} are the delays in transfer of a single data packet from a TN (terminal node) to the corresponding FI and from an FI to the cloud data-center. The mean transmission latency, d_{fog}, for the data packets of N_i application instances running within V_i (V represents a collection of Tns) is hence given by

$$\boldsymbol{d_{fog} = [d_{tf}P_i + d_{fc}p_i + d_{fc}b_r^i + d_{tf}\, b_r^i + d_{tf}\,(P_i - p_i)^r]/P_i} \quad (1)$$

where, P_i and p_i ($P_i > p_i$) are the total number of packets sent by N_i application instances to F_i and from F_i to the cloud data centers. b_r is the cumulative number of data packets that are sent as a response to b requests. The mean transmission latency for an application instance (D_{fog}^{tr}) request is given by

$$\boldsymbol{D_{fog}^{tr} = \frac{d_{tf}\sum_{i=1}^{w}[P_i + p_i^r + (P_i - p_i)^r] + d_{fc}\sum_{i=1}^{w}[P_i + p_i^r]}{\sum_{i=1}^{w}P_i}} \quad (2)$$

In traditional cloud computing the same would be represented as

$$\boldsymbol{D_{cloud}^{tr} = \frac{d_{fc}\sum_{i=1}^{w}[P_i + p_i^r]}{\sum_{i=1}^{w}P_i}} \quad (3)$$

Latency for an application instance request is calculated by taking into account the number of requests that are processed at the server end before it is processed. Let us assume that at time t, N_i (number of application instances) are running within V_i. Thus, for a total of w VCs the total number of application instances processed simultaneously is $\boldsymbol{N = \sum_{i=1}^{w} N_i}$.

$\Delta(Vi, Ii)$ is the service delay of an application instance, Ii running in Vi, served by Fi. Out of Ni application instances it is assumed that ni ($Ni > ni$) instances are redirected to the core cloud computing module for servicing. The total number of application instances processing at the cloud side at time t is $\boldsymbol{n = \sum_{i=1}^{w} n_i}$, and the processing latency at the cloud side for each of these n application instances is denoted as $\nabla(n)$. The mean processing latency of an application instance running within Vi is calculated as

$$\boldsymbol{\bar{\Delta}_{fog}(Vi, Ii) = [N_i\Delta(Vi, Ii) + n_i\nabla(n)]/N_i} \quad (4)$$

As we consider all the VCs (V1, . . . , Vw) present in tier 1, the mean service latency D_{fog}^{sr} can be represented as

$$D_{fog}^{sr} = \frac{\sum_{i=1}^{w} \bar{\Delta}_{fog}(Vi,Ii)}{w} \quad (5)$$

On the contrary, in a generic cloud model, all N application instances running at the user side, directly interact with the core computing module and require it to be constantly involved. The mean processing latency of an application instance request (D_{cloud}^{sr}) here is given by:

$$D_{cloud}^{sr} = \frac{\sum_{i=1}^{w} N_i}{N} \mathbf{N} = \nabla(\mathbf{N}) \quad (6)$$

5. Performance Evaluation

The latency faced when a data packet is transmitted is based on the round trip between two terminals, and is computed as rtt(ms) = 0.03 × distance(km) + 5 [18]. We change the percentage of applications which require access to the cloud core. Plotting the cumulative transmission latency for all the nodes within a VC against variable number of TNs as shown in **Fig. 7** we see that with the increase of the number of TNs present at the lowest tier the cumulative transmission latency increases as shown by a linear slope.
Furthermore, as the percentage of applications routed towards the core increases the transmission latency is observed to increase.

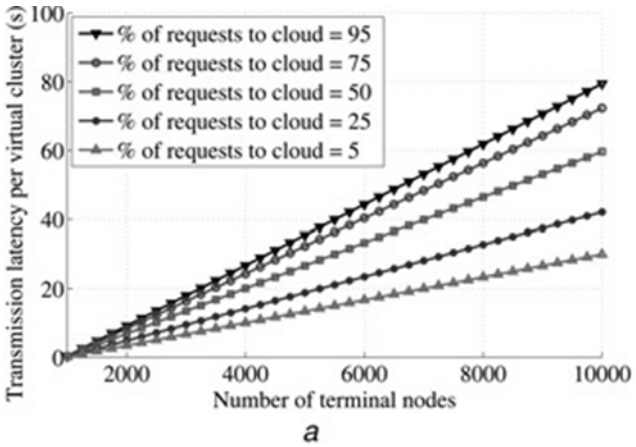


Fig. 7. Transmission Latency VS Number of requests to the cloud

In **Fig. 8**, we observe that as the number of FIs decreases the service latency increases and except for one case where FIs = 1. In all other cases the service latency is found to be less than what it would be in a traditional cloud computing environment.

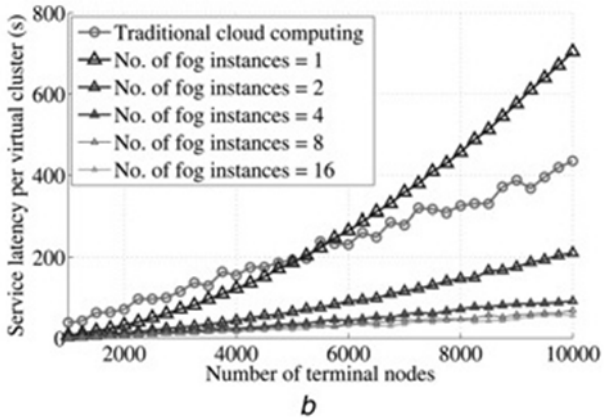


Fig. 8. Service latency of FIs in comparison to traditional clouds

However, in case of a single FI present in the system, a bi-layered cloud computing architecture manifests itself and yields higher service latency as compared to that in conventional cloud computing due to the additional layer overhead.

A thorough comparison of the service latency for processes running in fog computing and cloud computing environments can be given by plotting the transmission and processing latencies along with the total service latency for both these environments as shown in **Fig. 9**.

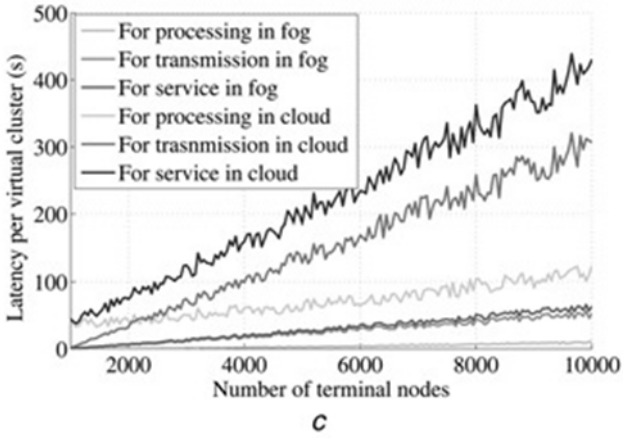


Fig. 9. comparison of the service latency for processes running in fog computing and cloud computing environments

6. CloudSim simulations to demonstrate the efficacy of proposed Fog service architecture

6.1 Scenario 1 for simulation

We simulated a scenario that requires time sensitive operations. The scenario is a traditional patient-medicine scenario in which it is critical for patient to receive his medicine at a proper time. A device is conFig.d to inject patients at a particular time based on its interaction with the cloud. If high latency occurs in this scenario in terms of response from the cloud the results can be disastrous and may lead to life threatening consequences. We can simulate the efficiency of the fog over cloud in our infrastructure in terms of

- Response Time
- Network usage
- Latency
- Cost
- Energy

6.1.1 Simulation Environment

In this scenario there are a number of patients for whom it is critical that their medicine is given on time. A couple of seconds of latency and it could cost a patient's life. For this application we have four main modules each module is responsible for performing a task and each module represents a separate task.

Timer module is responsible for invoking the checker module at appropriate time. Checker module checks for patients group for which medicine is intended. Potency calculator module is responsible for calculating the potency of the medicine. Finally the selector module selects which individual patient this medicine is intended for. In the end our actuator injector injects the medicine to the appropriate patient. This scenario was chosen because latency and proximity of network are crucial parameters for simulation of this scenario.

6.1.2 Simulator

The simulator we used for this purpose was cloudsim and its extension that we have worked on is iFogSim. iFogSim has the ability to model applications in terms of geographical proximities by adding them to geo coverage map. It has also the ability to measure parameters related to fog simulation e.g cost, energy etc.

6.1.3 Steps for simulation

Following are the important steps and parameters we needed to set in order to execute our simulation

Step 1:

iFogSim is based on Cloud Sim and uses CloudSim as its main engine for performing tasks such as creation of datacenters that are core part of our simulation. So we initialize cloudsim with the following parameters. Number of users for this simulation is set to one. We initialize calendar for keeping the current instance so that it can conclude at the end when simulation starts. In the end we initialize trace flag to "false" so that details log which is not relevant to our simulation is not shown.

Step 2:

We initialize fog broker based on data center broker. A data center broker class coordinates between users and Cloud Service providers based on their requirements related to Quality of Service requirements. A fog broker that helps users create tuples runs on fog. Tuples are extended from cloudlets class in CloudSim that models task in cloudsim so does tuples in IFogSim and represents tasks for fog going up and down the modules.

Step 3:

We create cloud data center and fog data center with the following characteristics. Both fog and cloud data centers are created with separate characteristics. As in real case we set characteristics of our fog device so that it is less powerful than our cloud processing device and has less storage than cloud data center.

The characteristics for our relevant cloud devices are depicted in **Table 2**. Below it is the table for our fog devices and their characteristics (Fog is also traditionally a data center which is less powerful but much closer to the origin where data is generated).

Table 2. Cloud Data Center Characteristics

Name of device	**Cloud**
Million instructions per second	**44880**
Ram	**40 Gb**
Uploading Bandwidth	**100 Mbits/sec**
Downloading Bandwidth	**10 Gbits/sec**
Parent level	**0(top of the topology)**
Rate per processing usage	**$ 0.01**
Busy power	**1648 w**
Idle power	**1332 w**

Table 3. Fog Data Center Characteristics

Name of device	**Proxy server**
Million instructions per second	**2800 mips**
Ram	**4 Gb**
Uploading Bandwidth	**10 Gbits/sec**
Downloading Bandwidth	**10 Gbits/sec**
Parent level	**1(Child of Cloud)**
Rate per processing usage	**$ 0.01**
Busy power	**102 w**
Idle power	**80 w**
Latency between cloud and proxy server	**100 ms**

6.1.4 Gateway devices

At the second last level of the hierarchy we create gateway devices. These gateway devices are part of our fog layer and responsible for communicating with our proxy server and cloud devices. Here are the characteristics of our gateway devices.

Table 4. Gateway device characteristics

Million inst per second	**2800 mips**
Ram	**4 Gb**
Upbw	**10 Gbits**
Downbw	**10 Gbits**
Level	**2(below proxy server)**
Busy power	**102 w**
Idle power	**79 w**
Latency b/w gateway and proxy server	**4 ms**

6.1.5 Observer devices

We then create observer devices for our scenario which observe the patients with following characteristics.

Observer devices characteristics are as follows

Table 5. Observer Device Characteristics

Million inst. Per second	**1000 mips**
Ram	**1 Gb**
Up bandwidth	**10 Gbits/sec**
Down bandwidth	**270 Mbits/sec**
Level	**3(below gateways)**
Busy power	**85 w**
Idle power	**79 w**

6.1.6 Sensors and Actuators

As our actual device model is based on IoT devices generating huge amounts of data that needs to be processed, each device involved in our scenario for patient monitoring purpose has a sensor and an actuator attached to it. The purpose of sensor 'timer' is to identify the patient or group of patients for whom the medicine should be given and purpose of actuator is to inject medicine into patients at the time the response comes from the server for the specific patient which is identified by the selector module on the server.

6.1.7 Module to module interaction

As our communication is between modules of the application that interact with each other in order to perform the tasks and collect the results. Tuples are sent from one module to the other in order to interact from one module to other. The tuples sent up to the fog or cloud for processing, are identified as TuplesUp and tuples that are sent downward from one module to the other are known as TupleDown. Also tuples are mapped to modules using tuplemapping technique defined in IFogSim.

6.1.8 Simulation Execution

Finally we run the scenario on iFogSim for different configurations. The configurations are as follows.

6.1.8.1 One Gateway and two observer devices

In this topology configuration one gateway is attached with two devices. Each mobile shown in the diagram acts as an observer and has an actuator and sensor which are responsible for sending data to the server and acting on response from the server.

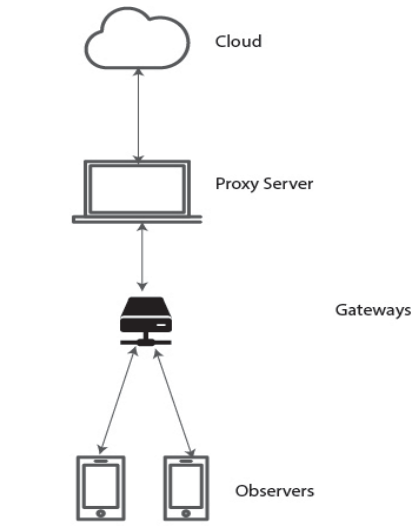
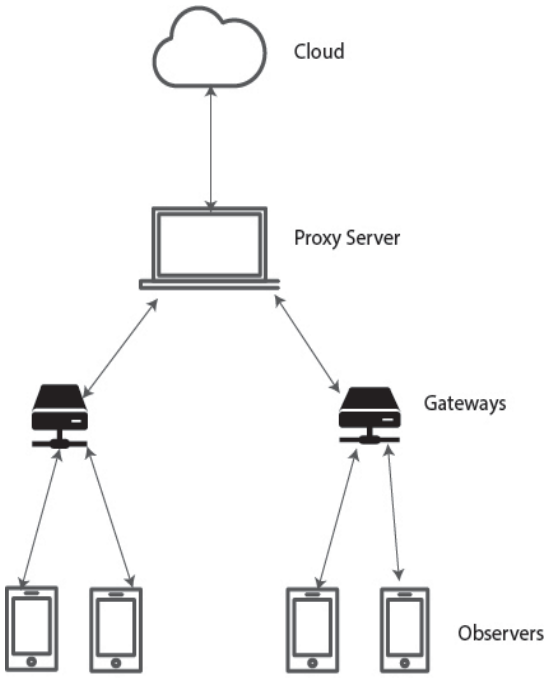


Fig. 10. One gateway and two observers

6.1.8.2 Two Gateways and two observer devices

In this topology, two gateways connect with two observer devices per each gateway and proxy server represents fog processing unit.

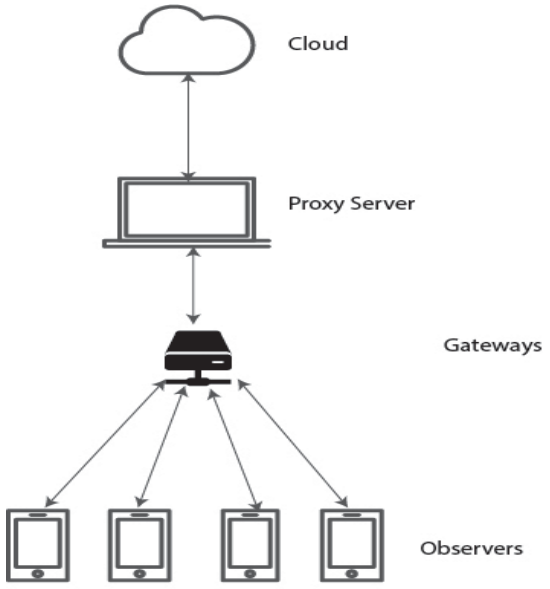


Two Gateways and two observers topology

Fig. 11. Two gateways and two observers

6.1.8.3 One gateway and four devices

In this topology one gateway connect with four observer devices each. As a result of this configuration network congestion increases so does latency and in time sensitive environments it is critical for devices to send data and act upon response data quickly.



One Gateway and four observers topology

Fig. 12. One gateway and four observers

6.1.9 Results

We now communicate results of our simulation.

6.1.9.1 Latency (Average)

The average latency for cloud based configurations and fog based configuration are as follows. Results are in milliseconds.

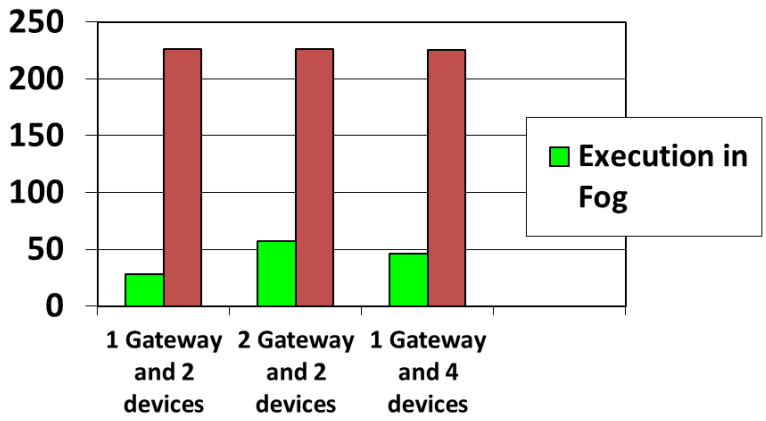


Fig. 13. Latency on fog and cloud)

All response times are in milliseconds. Latency is calculated using module to module latency and then average of them is taken which is shown here as the graph clearly depicts latency is much higher when application modules are executed on fog and when executed on cloud.

6.1.9.2 Operational Cost

Total cost is a combination of different parameters that depicts what operational cost is intended for the fog and cloud based deployments.

Fig. 14. Total Operational Cost)

Operational cost is a combined result of different parameters of the system like processing power, energy etc. As we can clearly see here the difference between execution in cloud and fog is evident from the results here as cost for fog is much less than cloud. The cost is not in terms of dollars but rather it is an abstract representation of cost for operational parameters of a system.

6.1.9.3 Network Usage

Network usage is the overall network usage for the system. Network usage is represented in kilobytes. Here we can clearly see from the results that as number of devices increase so does the network usage, which significantly poses a difference in fog, and cloud based deployments. In fog-based environments network usage is much less as compared to fog based environments.

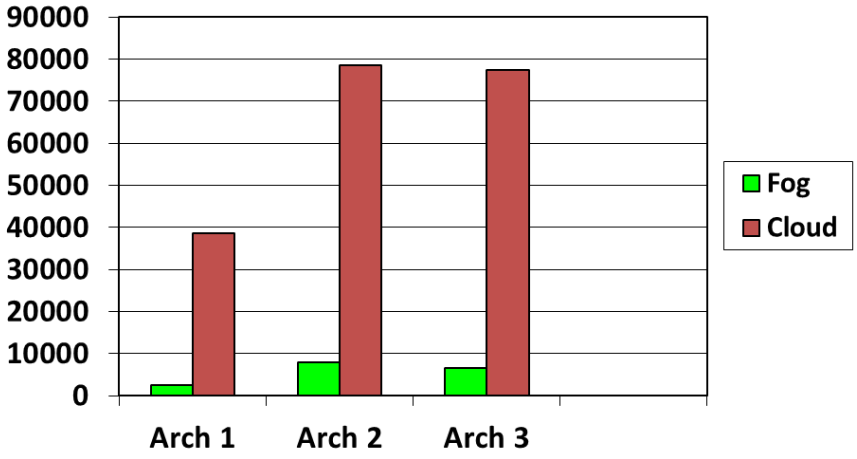


Fig. 15. Network Usage

6.1.9.4 Energy Consumption

Finally the energy consumption for fog based architectures in comparison to traditional clouds is presented as follows.

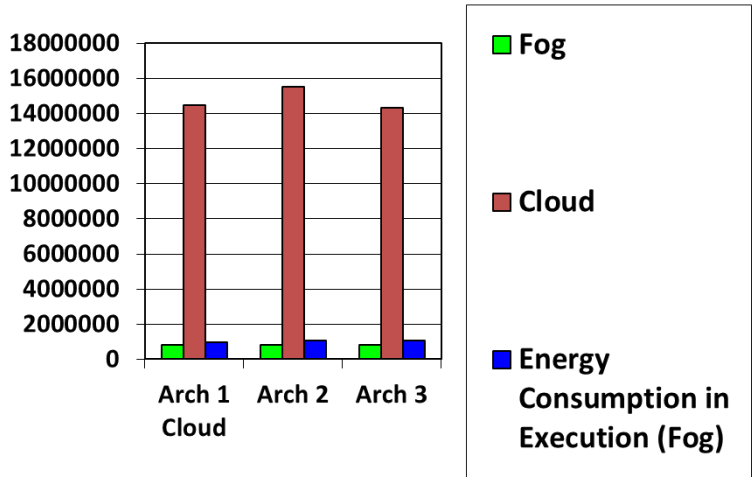


Fig. 16. Energy Consumption in Fog Based configurations

6.2 Scenario 2 for Simulation

Body sensor networks aims at sensing human conditions and environment. It has wide range of applications in current era. An extensive network of sensors connected to human body in a community generates enormous volume of circumstantial data. These large amounts of data need an efficient approach for storage and processing which is scalable according to requirement. Cloud computing can provide this platform, an infrastructure for processing and storage of data both offline and online.

Fog computing also comes into picture as it entails characteristics which are required by body sensor networks for instance dynamic resource allocation and management, run-time decision making according to changing conditions, handling heterogeneous sensor nodes which are de-centralized. The dispersed network of fog makes it easier to collect large amounts of sensed data from multiple hops. Furthermore its ability to provide location awareness with application helps in predicting the current location of host body which is necessary in case of health oriented implementations. It can definitely provide scalable platform to perform data mining of data streams generated by body sensor data. Furthermore, it has the ability of supporting online analytics. BSN nodes have low energy, power and bandwidth which make it a suitable candidate of fog infrastructure. Fog computing integrated with body sensor networks incorporates fundamental characteristics of the system such as; flexibility/scalability of service, sensor nodes heterogeneity, low latency dynamic distribution and administration of health applications in a large community.

6.2.1 Architecture

Body Sensor networks is one of the principle component in healthcare applications. Internet of things (IoT) generates bio-signals which include body temperature, SpO2, blood pressure, blood glucose, electrocardiogram (ECG), electroencephalogram (EEG), electromyography (EMG) and galvanic skin responses from wearable/implantable sensor nodes. Context awareness information is also required in BSN and can be gathered by GPS. BSN is great support in disease management and prevention.

Fog computing has been introduced recently which outspreads the cloud to the verge of network. Fog computing has the capability of supporting real-time interactions among large number of fog nodes which can be sensor in BSN implemented with fog infrastructure. It also provides low latency, context awareness and online analytics which are beneficial in case of emergency where dynamic decision making is required. Fog can be potential platform as it suits the requirements of BSN networks.

In this section we propose architecture for BSN implemented with clouds, introducing a middle layer with fog infrastructure. Deploying this architecture, we establish the efficiency of fog computing in healthcare applications in terms of latency, energy utilization, cost of execution and cost of processing at various nodes. This architecture utilizes fog gateway in addition to cloud infrastructure. In this research work architecture has been introduced. The distinct feature is additional layer of fog in addition with cloud. An Architecture diagram of BSN implemented with Fog is shown in **Fig. 17**.

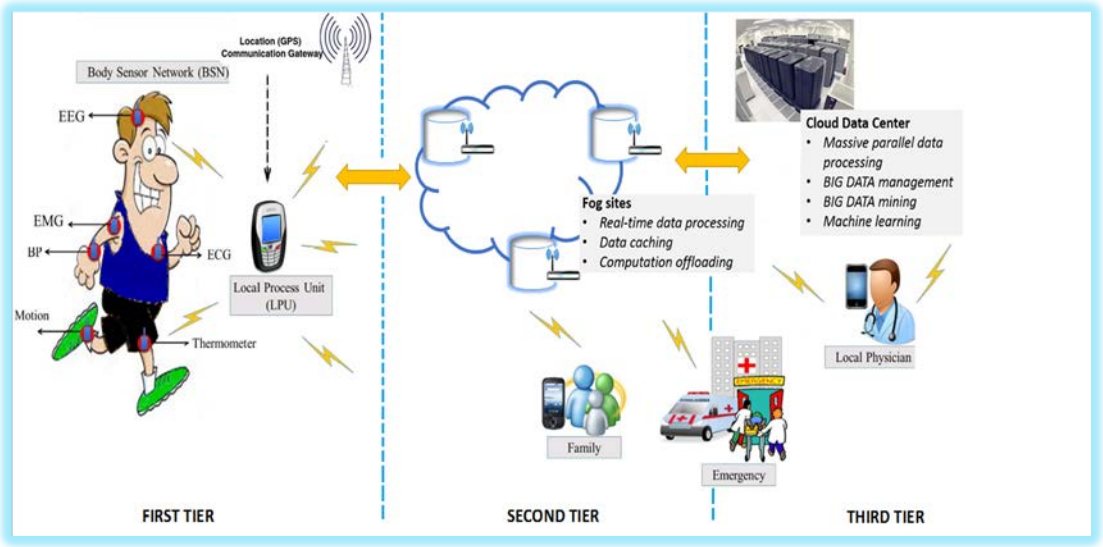


Fig. 17. Architectural View of BSN implemented with FOG Layer

1st tier : Data Acquisition:

BSN architecture (**Fig. 17**) consists of wearable/implantable sensors. These sensor nodes are embedded with bio-sensors (EEG, ECG, EMG, Blood Pressure, body temperature etc.). Sensors gather physiological parameters and transfer them to Local Processing Unit (LPU). It also gathers contextual data such as location .LPU is supposed to stay with the human body implanted with BSN and it can be PDA or smartphone. It works as a router between sensor nodes and fog gateway, transmitting data using communication protocols such as Wi-Fi/Zigbee or Bluetooth. It is also responsible for immediately alerting abnormalities to the person associated with sensor nodes. This is data acquisition in which data is collected, convert raw readings into meaningful values and transport it to designated modules.

2nd tier: Data Management/Application Execution/ Real-time Analytics

2nd tier consist of fog gateway and distributed databases. It differs from conventional gateway as these have intelligence of fog computing and has various responsibilities. First of all data consistency is ensured from multiple sensor nodes. It is extremely important to validate the data collected by sensors and maintain the quality. Data management includes missing data notification, data transmission, automatic data correction and data segregation. It also responsible for detecting any changes in data collected (change state). Application execution which includes data analysis for decision making, health monitoring at a continuous rate is also performed at this layer. This layer has a significant role in this architecture as it is responsible for informing consult parties in case of any emergency situations with the patient. Real- time notification is the essence of this tier. It can help in many adverse situations. Database should be able to store high rate data coming from sensors at dispersed locations.

3rd tier : Long-time Result Processing and Storage

This specific tier encompasses cloud deployment. In this architecture it's responsible for storing large amounts of data, processing output streams for analysis of data collected over long period of time. This tier informs physician about the health of the patient and what is long term condition. It is connected with physician of the concerned patient.

6.2.2 Application Model

This section explains application model of the system. It clarifies the general requirements of the system. There are various sensors signals gathered for human well-being and diagnosis but in this case study we will be concentrating on ECG sensor for experimentation purpose. ECG is one of the fundamental tests of health monitoring system. Overall general requirements of the system are elaborated as follows:

- Provide the functionality of receiving and managing sensor data in unified manner from body sensor networks.
- Setting up scalable/extensible infrastructure for managing multiple streams of sensor data.
- Continuous processing and storage on short term and long term basis for analysis and decision making.
- System should provide low –latency. If patients need immediate treatment system should be able to respond quickly and alert related parties.
- System should able to manage voluminous data coming from multiple hops. Sensors generate continuous signal data which can result in traffic congesting network.
- System should be able to process substantial data processing on long term. This is required for performing overall analysis of patient's health over long period of time.
- Real-time analytics is required in this system for responding in adverse situations.

Application model defines the modules, edges, tuple (input-output relationship of modules) of the system. This model is created in context with the requirements and it maps to implementation of the system. The basic modules are interface, sensor analyzer and cloud storage. There is one ECG sensor input and two Display actuators. Edges are named in reference to their relationship with modules. **Fig. 14** elaborates application domain of BSN implemented with fog architecture.

As demonstrated in **Fig. 18**, the application model of BSN implemented with fog architecture has one sensor node, two actuators, three major modules performing processing. There are edges which are named and are meant to connect modules. They depict input-output relationship between modules. The functionality of each module is described as follows:

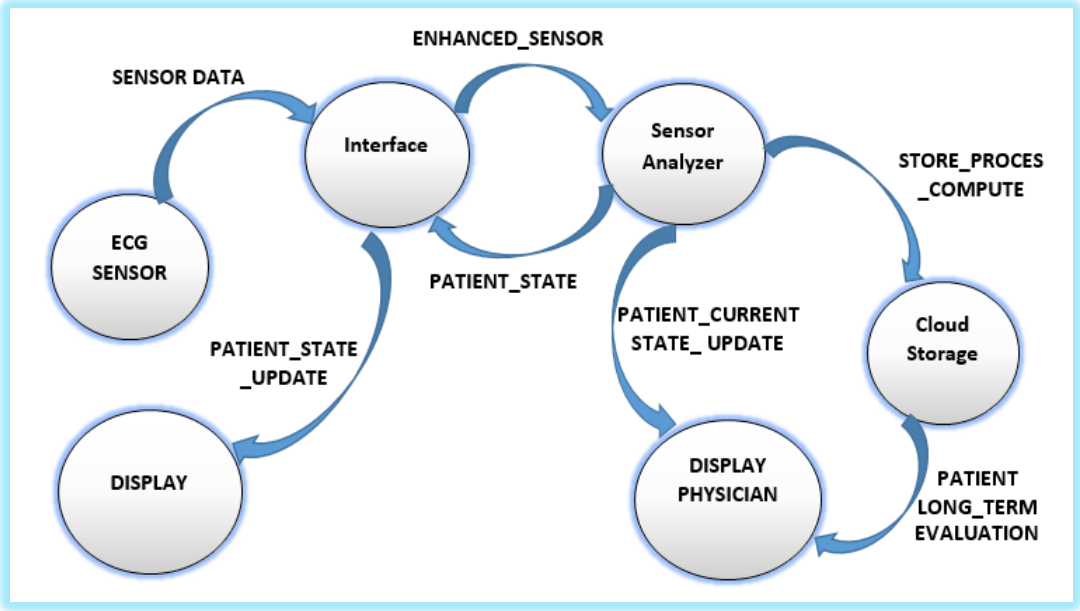


Fig. 18. Application Model of ECG Case Study

Interface: This module is responsible for receiving raw ECG signals generated by sensor nodes. It validates signals, discard any discrepancy and check consistency of data. Data management is done at this stage and includes missing data notification, data transmission, automatic data correction and data segregation. After this data is transmitted to sensor analyzer module for further processing and analysis. Interface module also updates patients about his/her current ECG report.

Sensor Analyzer: The responsibility of sensor analyzer is to process incoming data from interface module. It perform analysis on ECG extracted values and updates patients about his/her state. This module also inform physician about any immediate assistance required in case of abnormal values of ECG. It responds to interface module and it updates patients about his current ECG situation. Sensor analyzer is also connected to DISPLAY PHYSICIAN actuator so that physician can get current and continuous update of patient ECG value.

Cloud Storage: This module is responsible for data processing and storage at cloud level. This involves large amounts of data; long term patient history is being stored and processed here. This module provides long term analysis of patient health over long period of time. It interfaces with physician in order to deliver patient history.

Edges connect these modules so they can communicate and perform data processing at various phases. These edges carry tuples of data which vary in characteristics. **Table 6** shows tuples carried by these edges, there capacity of carrying data, network length, source and destination. There CPU length and network length depends on the source and destination.

Table 6. Description of edges in ECG application Model

Tuple Type	Source	Destination	CPU Length (MIPS)	Network Length
SENSOR DATA	ECG	Interface	2000	500
ENHANCED_SENSOR	Interface	Sensor Analyzer	3500	500
PATIENT_STATE	Sensor Analyzer	Interface	14	500
PATIENT_STATE_UPDATE	Interface	DISPLAY	1000	500
PATIENT_CURRENT _STATE UPDATE	Sensor Analyzer	DISPLY PHYSICIAN	14	500
STORE_PROCESS _COMPUTE	Sensor Analyzer	Cloud Storage	1000	1000
PATIENT_LONG_TERM EVALUATION	Cloud Storage	DISPLY PHYSICIAN	1000	1000

6.2.3 Physical Network Topology

This section depicts the physical infrastructure of the BSN system implemented with Fog. Physical topology shows the pattern of nodes and devices in network. Physical entities are created, their competence, capability and configurations are specified. These entities include sensor, actuators, gateways and cloud VM. The links between these entities and their configuration are also established.

Physical network topology is important to understand the pattern of the network, how various network devices are organized and how they communicate with each other. There configurations and capacity determine the load a network can tolerate, amount of data it can transfer. **Table 7** shows devices and their capacity which are being utilized in the network.

Table 7. Device Configuration in Network topology

	Cloud	Proxy Server	Area	Local Processing
"ratePerMips"	0.001	0.0	0.0	0.0
"downBw"	10000	10000	10000	270
"level"	0	1	2	3
"upBw"	100	10000	10000	10000
"ram"	40000	4000	4000	1000
"name"	"cloud"	" proxy-server"	"Area"	"LPU"
"mips"	20000	2000	2000	1400
"type"	"CLOUD_	"FOG_	"FOG	FOG_DEVICE"

Table 8 demonstrations configuration for sensors. ECG sensor is being fed in the LPU for further processing. The sensor mean value and deviation is specified so we can test the application model.

Table 8. ECG Sensor Configuration

Configuration	**Sensor**
"sensortype"	"ECG"
"name"	Sensor
"value"	10.0
"type"	"SENSOR"
"distribution"	2

Fig. 19 illustrates the network topology illustrating the organization of network devices. Two actuators and one sensor are controlled by one local processor unit. Actuators are display devices to update current and long-term status of patient health. Various LPU comes under one Area and then multiple area report to clouds.

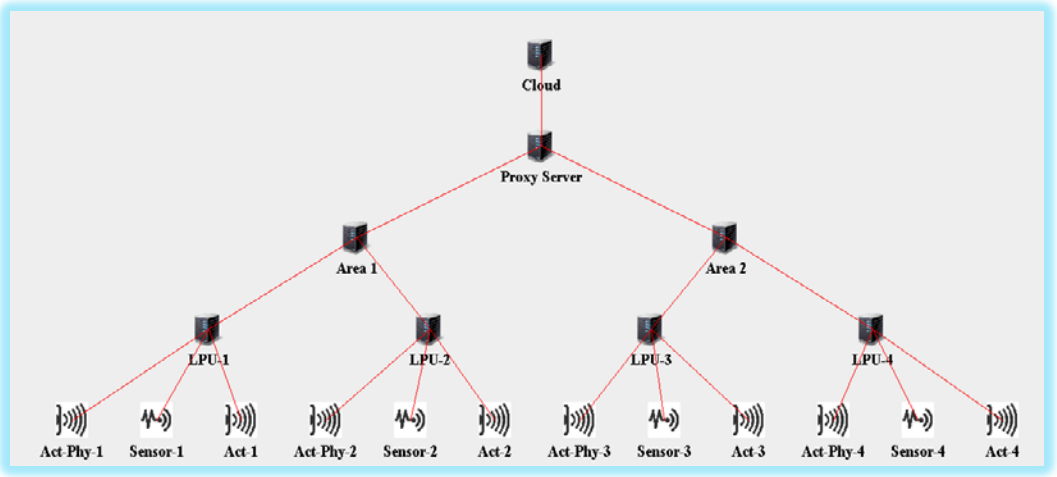


Fig. 19. Physical Network Topology of ECG application

6.2.4 Simulation Setup

In above sections we have proposed a novel BSN architecture implemented with Fog layer. We have considered the rationale behind introducing fog layer, and have recognized several potential benefits of the approach. Architecture has been illustrated of the system, along with application model and physical network topology demonstrating arrangement of network devices with their specific configurations.

This section focuses on simulation of above proposed system so that we can evaluate system on various parameters. A toolkit iFogSim is being used for simulation. This simulation tool provides basis for testing application model of the systems employed with fog architecture. Performance and resource management is evaluated and it also measures the efficiency in terms of network usage, latency, energy consumption and cost.
Various classes are depicted in iFogSim to model application. The application model demonstrated in **Fig. 21** has three significant modules; interface, sensor analyzer and cloud storage. These are implemented using AppModule class, data dependencies or edges are signified by AppEdge class. Control loop in application model (ECG →interface → sensor analyzer→interface→DISPLAY) and (ECG→Interface →sensor analyzer→cloud storage→DISPLAY PHYSICIAN) are implemented using AppLoop class. Measuring the time in these control loops helps us determine latency in terms of cloud and fog architecture. Instances of cloud devices, fog devices, sensors and actuators are created with specific configurations for simulation. Physical topology demonstrated in **Fig. 19** is also created in iFogSim specifying arrangement of network devices.

Once the application is modeled and control loops determined we tested implementation with cloud and fog architecture. There are two placement mechanisms for modules, cloud-only and edge ward. In our application sensor analyzer is the fundamental testing module. In one analysis we placed this module on cloud device and in second strategy it is placed on fog device with different configuration as compared to first one. The application is then run with complete cycle starting from sending ECG signal, data management and purification at interface module and then transportation to sensor analyzer module for analyzing condition of parent. At end sending useful information (patient status) is conveyed to cloud storage, family members and physician respectively. The application results such as latency, network usage,

total execution time, cost and energy consumption with both placement techniques is collected and then compared for evaluating performance at cloud level and fog level. Algorithm and code extracts from the application simulation are shown below for further clarity of the simulation performed.

6.2.4.1 Algorithm

Algorithm shows the inputs and desired outputs of the simulation. It specifies steps briefly to perform simulation.

Algorithm for BSN implemented with Fog

Input:

- ☐ Integer Number of Areas
- ☐ Integer Number of Local Processing Units
- ☐ Boolean Cloud

Output:

- ☐ Execution Time
- ☐ Execution cost
- ☐ Network Usage
- ☐ Control Loop Latency
- ☐ Energy Consumption

Steps:

1. Call function “createApplication”
2. Call function “CreateFogDevices”
3. Initialize module mapping
4. Add module cloud storage to device cloud
5. For fog devices stating with “L”

Add module interface to device LPU

6. If cloud is true

Add module sensor analyzer to device cloud

Else

Add module sensor analyzer to device Area

7. Submit application

Place module on cloud and on fog devices at edge

Add sensors and actuators

8. Start simulation
9. End simulation
10. Define function createFogDevices

Call function createFogDevice (Create fog device “cloud”)

Call function createFogDevice (Create fog device “proxy server”)

Set link latency (proxy server→cloud)

For total numbers of Areas

Call function addGw

11. Define function createFogDevice

Set characteristics of fog devices

Add host configuration to fog devices

Set level of each fog device

12. Define function addGw

Call function createFogDevice (Create fog device “Area”)

Set link latency (Area → proxy server)
For total number of Local Processing Unit
Call function addLpu
Set link latency (LPU →Area)
13. Define function addLpu
Call function createFogDevice (Create fog device "LPU")
Add sensor
Add actuator Display
Add actuator DisplayPhysicain
Set LPU as gateway to Sensor
Set link latency (sensor → LPU)
Set LPU as gateway to Dispaly
Set link latency (Displayr →LPU)
Set LPU as gateway to DisplayPhysician
Set link latency (DisplayPhysician →LPU)
14. Define function createApplication
Add modules of application
Add edges to connect modules
Add tuples to each edge
Add control loops

6.2.5 Simulation Results

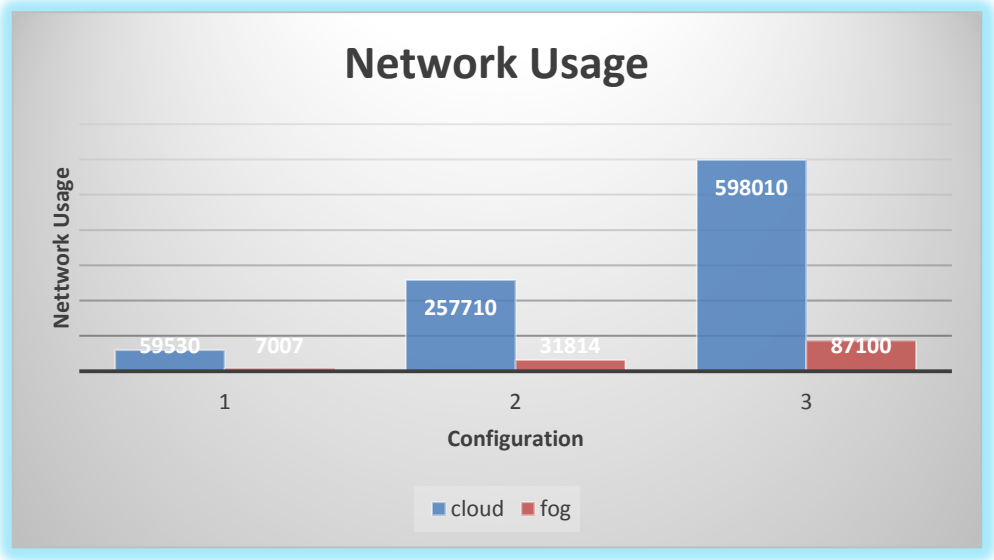
This application is evaluated both ways, with cloud architecture and with additional fog layer introduced in between. Sensor analyzer is placed on cloud for first testing phase and in second it is placed on Area gateway which is conFig.d as fog device. Efficiency of both placement strategies is assessed. Various parameters such as network usage, latency, cost of execution, energy consumption and execution time all are recorded and the compared. Three configurations are employed with varying number of area and local processing unit so that consistent pattern could be extracted. Results are demonstrated.

6.2.5.1 Network Usage

This parameter defines the usage of network resources. The more the network is used expenditure increases. Efficient network topologies prefer to use minimal network. It reduces network traffic and expenditure in terms of resource usage. Network usage in cloud placement and fog placement is shown below in **Table 9** with corresponding graph.

Table 9. Network Usage with Cloud and Fog Architectures

Configuration	Number of Areas	Number of LPU	Cloud	Fog
1	1	2	59530	7007
2	2	4	257710	31814
3	3	6	598010	87100



The results clearly demonstrate that network usage in fog architecture is considerably low as compared to cloud only placement. This proves that fog increases the efficiency because data does not have to be transported to cloud often rather it is processed at fog nodes. This decreases network usage.

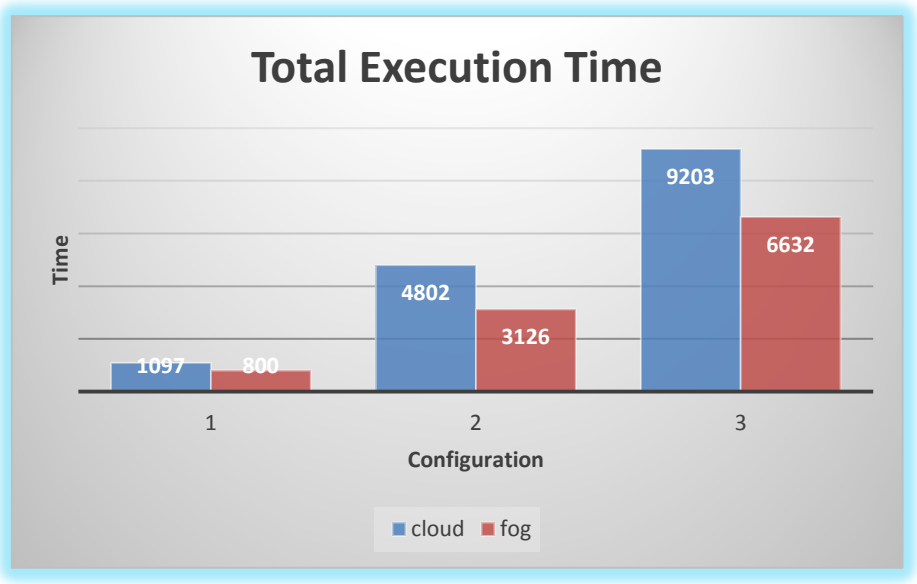
6.2.5.2 Execution Time

This parameter represents the total execution time of the tuples carried by edges. It calculates total time taken or sensor transportation, data transportation, processing and conveying the results. Execution time with clouds and fog architecture vary considerable. **Table 10** shows result.

Same amount of workload is tested with both architectures. Cloud results show greater number as compared to fog architecture. The reason behind is very obvious. Fog nodes are placed near to end users. This makes cost less as resources being used to transport data over long distances are not required.

Table 10. Execution Time with Cloud and Fog Architecture

Configuration	Number of Areas	Number of LPU	cloud	Fog
1	1	2	1097	800
2	2	4	4802	3126
3	3	6	9203	6632

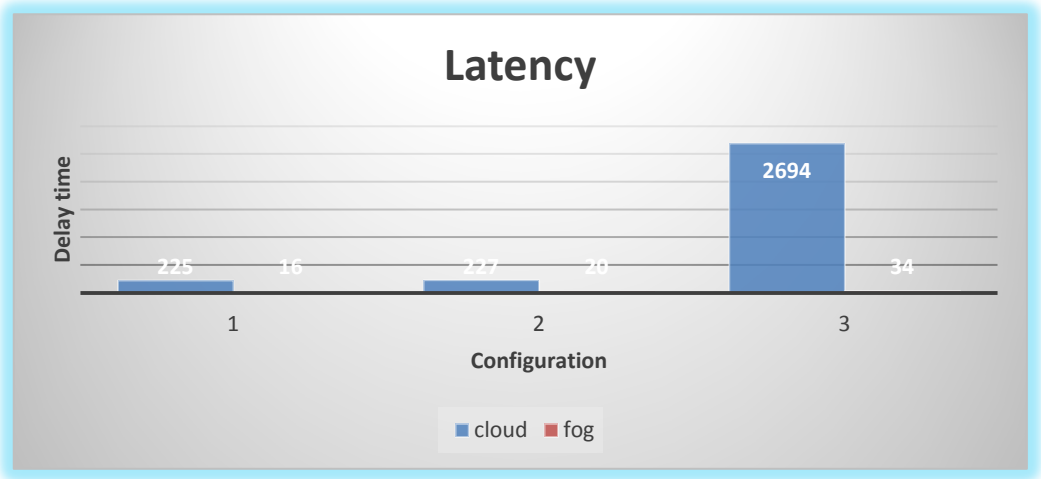


6.2.5.3 Latency

Health applications are very time sensitive. Results cannot be delayed. For instance if patient's ECG is demonstrating some kind of abnormality it has to be immediately reported to concerned parties. Delay cannot be afforded as it can lead to very negative consequences. This Delay is calculated by implementing control loop (ECG→ interface→sensor analyzer→interface→DISPLAY). Results are demonstrated in **Table 11**.

Table 11. Latency in Cloud and Fog Architectures

Configuration	Number of Areas	Number of LPU	Cloud	Fog
1	1	2	225	16
2	2	4	227	20
3	3	6	2694	34



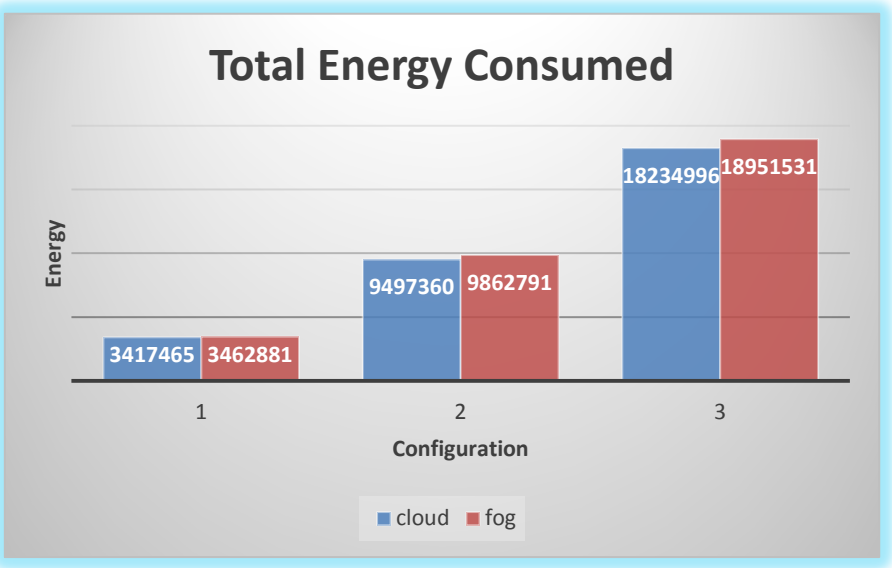
Latency results depict substantial differences between cloud and fog setups. It can be seen that with cloud architecture latency is very high. Secondly with increasing number of local processing units and areas it is dramatically increasing starting from 225 →227→2694. This means that as number of users increase, latency issue will become problematic. Fog depicts very low numbers. This ensures that there is considerably less delay.

6.2.5.4 Energy Consumption

Energy consumption is one of the most thought out domain in current era. We want applications which are energy friendly and consume less energy. This is the reason we compared energy consumption with both the architectures. Results are summarized in **Table 12**.

Table 12. Energy Consumed with Cloud and Fog Architectures

Configuration	Number of Areas	Number of LPU	cloud	Fog
1	1	2	1.3	1.3
2	2	4	1.6	1.3
3	3	6	1.8	1.3



Energy consumed by both architecture is almost same, rather its bit higher with fog architecture. The reason behind this rise is increasing number of fog devices dispersed over geographical area. They consume more energy as compared to one centralized cloud device, although fog devices are said to be energy efficient. In order to manage the energy consumption more efficiently we can also take help from the stable election protocol names prolong SEP that is designed specifically for wireless sensor networks[19].

6.2.5.5 Execution Cost

The execution cost includes transferring data to and fro to network devices and processing it. It includes cost of computing, storage and communication. We have tested same workloads with both the architectures. Results are shown in **Table 13**.

Table 13. Execution Cost with Cloud and Fog Architecture

Configuration	Number of Areas	Number of LPU	Cloud	Fog
1	1	2	293384	9456
2	2	4	3946184	25567
3	3	6	4197088	52419

Execution cost is intensely high with cloud infrastructure. There is great difference between values of both architectures. Results clearly demonstrate that fog renders much lower cost in terms of storage, computation and communication although we cannot utilize fog for bulk data or extensive calculations. Edge nodes with low competency can be used for lighter calculation and time sensitive queries could be processed at fog. Clouds are suitable for storing large amounts of data and performing complex calculations.

7. Discussion and Conclusion

In light of the above results presented to prove the higher efficiency of Fog networks over the traditional clouds we can safely conclude that Fog computing is an able extension to clouds and has immense potential as a remedy to problems of latency, network delays and geographical distribution that are inherent vices of the cloud computing paradigm. The models presented in this paper to facilitate a shift from the cloud computing environments to the more distributed Fog computing paradigm are molded to accommodate a real-time, less latent and quick response mechanism that is operational in the closest possible geographical environment of the end user. In addition, the assistance and support of the cloud component of the models insures that bulk storage of data and report generation from big data is not an issue.

One of the major issues with clouds has been the reliance on private clouds for a secure less latent use of the technology. This in turn means that fixed resources and expense would be necessary and hence the true benefits of cloud computing would be diminished. As can be seen with the Fog computing models, the entire flow of the business process in terms of the creation, storage and deployment of applications is solely done through the hired resources and through the Fog computing environment that we have proposed. The local nature of real-time processing unit insures greater security and control over data and hence there is no need for a private cloud. In fact, the private cloud is now provided by the name of a Fog network by a

local Fog vendor registered with local authorities. This gives a traceable and secure alternative to a very expensive private cloud model. [20]

The software developer would develop software with two main modules, one for the Fog network and other for the Cloud network. For this, he would use a PaaS at the Fog device with IOX as the main developing environment to develop a module that would process real-time data and a PaaS at the cloud data-centre to process bulk data for reporting [21]. These modules would be purchasable or downloadable via SaaS providers at the Fog and the Cloud networks. The storage requirements for the real-time processing of data would be catered by Fog devices that would have ephemeral storage and would be near to the end user and the bulk storage would be in the far away data centres at the cloud network. The entire Fog environment would be self-sufficient and would require no fixed investment, hence providing a truly beneficial pay as you go service.

Furthermore we Fig.d out that for a system serving a significant number of real-time, low latency IoT applications, the service latency over a fog environment would be significantly lower as compared to that of cloud computing. Moreover, the degree of energy emissions resulting from transmission of data to the computing cores was reasonably low.

We substantially justified the efficacy, soundness and importance of our proposed models by simulating a scenario on CloudSim and iFogSim and showing that judged over several parameters, our proposed shift of paradigm from cloud to fog would be beneficial in terms of reducing latency in real-time data processing over the cloud networks.

Fogging or fog computing is not an alternative to cloud computing. In fact, fog computing combined with the traditional cloud platform, will serve as an optimal computing platform in the fast emerging IoT environment.

8. Future Work

As mentioned earlier these models for service architecture for Fog computing are the first of their kind. Future work for their improvement, efficacy, validation and verification is going to be the key to exploring the true potential of Fog computing in the IoT environments. Some key areas that would really help in understanding these models and their importance could be in the field of identification of real-time data. We are currently relying on manual tagging of data rather than intelligent decision at the Fog device level. Efforts can be made to help the receiving Fog device identify which data to process and which to send to the cloud intelligently. Security is an issue as important in the Fog as it is in the cloud. Although methods such as decoy and Asymmetric encryption exist, we still need some concrete steps in this direction to evade threats such as the man in the middle attacks.

Finally, our future works would try to characterize fog computing concerning resource management and virtualization.

References

[1] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A. & Zaharia, M., "A view of cloud computing," *Communications of the ACM*, 53(4), 50-58, 2010. Article (CrossRef Link)

[2] Jacobson, I., Booch, G., Rumbaugh, J., Rumbaugh, J., & Booch, G., "The unified software development process," *Reading: Addison-wesley*, Vol. 1, 1999.

[3] Bal, S. N., "Clouds for different services," *IJCSI International Journal of Computer Science Issues*, 9(4), 273-277, 2012.

[4] Favela, J., & Peña-Mora, F., "An experience in collaborative software engineering education," *IEEE Software*, 18(2), 47-53, 2011. Article (CrossRef Link)

[5] Dillon, T., Wu, C., & Chang, E., "Cloud computing: issues and challenges," in *Proc. of 2010 24th IEEE international conference on advanced information networking and applications*, pp. 27-33, April 2010. Article (CrossRef Link)

[6] Hammersley, M. (Ed.), Educational Research: Volume One: Current Issues (Vol. 1). Sage, 1993.

[7] Naranjo, P. G. V., Shojafar, M., Vaca-Cardenas, L., Canali, C., Lancellotti, R., & Baccarelli, E., Big Data Over SmartGrid-A Fog Computing Perspective.

[8] Bonomi, F., Milito, R., Zhu, J., & Addepalli, S., "Fog computing and its role in the internet of things," in *Proc. of the first edition of the MCC workshop on Mobile cloud computing*, pp. 13-16, August 2012.

[9] Kruchten, P. B., "The 4+ 1 view model of architecture," *IEEE software*, 12(6), 42-50, 1995. Article (CrossRef Link)

[10] Khalid, A., & Shahbaz, M., "Cloud computing technology: services and opportunities," *Pakistan Journal of Science*, 65(3), 348-351, 2013.

[11] D. Satria, D. Park & M.Jo, "Recovery for Overloaded Mobile Edge Computing," *Future Generation Computer Systems*, Dec. 2016, Article (CrossRef Link)

[12] Deng, R., Lu, R., Lai, C., Luan, T. H., & Liang, H., "Optimal workload allocation in fog-cloud computing towards balanced delay and power consumption," *IEEE Internet of Things Journal*, 2016. Article (CrossRef Link)

[13] Deng et al., "Towards power consumption-delay tradeoff by workload allocation in cloud-fog computing," *ICC*, 2015. Article (CrossRef Link)

[14] Sarkar, S., Chatterjee, S., & Misra, S., "Assessment of the Suitability of Fog Computing in the Context of Internet of Things," *IEEE Transactions on Cloud Computing*, 2015. Article (CrossRef Link)

[15] Datta, S. K., Bonnet, C., & Haerri, J., "Fog Computing architecture to enable consumer centric Internet of Things services," in *Proc. of Consumer Electronics (ISCE), 2015 IEEE International Symposium on*, pp. 1-2, June 2015. Article (CrossRef Link)

[16] Shojafar, M., Cordeschi, N., & Baccarelli, E., "Energy-efficient adaptive resource management for real-time vehicular cloud services," *IEEE Transactions on Cloud computing*, 2016. Article (CrossRef Link)

[17] Baccarelli, E., Naranjo, P. G. V., Shojafar, M., & Scarpiniti, M., "Q*: Energy and delay-efficient dynamic queue management in TCP/IP virtualized data centers," *Computer Communications*, 2016. Article (CrossRef Link)

[18]Misra, Sudip, & Subhadeep Sarkar, "Theoretical modelling of fog computing: a green computing paradigm to support IoT applications," *IET Networks*, 2016.

[19] Naranjo, Paola G. Vinueza, Mohammad Shojafar, Habib Mostafaei, Zahra Pooranian and Enzo Baccarelli, "P-SEP: a prolong stable election routing algorithm for energy-limited heterogeneous fog-supported wireless sensor networks," *The Journal of Supercomputing*, Volume 73, Issue 2, pp 733–755. 2017. Article (CrossRef Link)

[20]Babar, S., Stango, A., Prasad, N., Sen, J., & Prasad, R., "Proposed embedded security framework for internet of things (iot)," in *Proc. of Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology (Wireless VITAE), 2011 2nd International Conference on*, pp. 1-5, February 2011.

[21]Vaquero, L. M., & Rodero-Merino, L., "Finding your way in the fog: Towards a comprehensive definition of fog computing," *ACM SIGCOMM Computer Communication Review*, 44(5), 27-32, 2014. Article (CrossRef Link)

Adnan Khalid is a lecturer at the prestigious Government College University Lahore. He has an M.Phil. in Computer Science and is pursuing a Ph.D. in Cloud Computing. His area of research is Fog Computing and he intends to highlight the benefits of this relatively novel field of research. Adnan Khalid teaches research methods and software engineering at the undergraduate and postgraduate level. He has four scholarly publications in local HEC recognized journals.

Dr. Muhammad Shahbaz is the Head of department of one of the most prestigious engineering and technology universities in Asia, University of Engineering and Technology, Lahore. He has a Ph.D in Computer Science specializing in Data Mining and Artificial Intelligence. He is an approved Ph.D supervisor and a full Professor of Computer Science.