Journal of
*Sensor and
Actuator Networks*

MDPI

Article

Fog Computing, Cloud Computing and IoT Environment: Advanced Broker Management System

Mohammed Al Masarweh [1,*], Tariq Alwada'n [2] and Waleed Afandi [1]

[1] Department of Management Information System, College of Business in Rabigh, King Abdulaziz University, Jeddah 25732, Saudi Arabia
[2] School of Computing, Engineering & Digital Technologies, Teesside University, Middlesbrough TS1 3BX, UK
[*] Correspondence: malmasarweh@kau.edu.sa

Abstract: Cloud computing is a massive amount of dynamic ad distributed resources that are delivered on request to clients over the Internet. Typical centralized cloud computing models may have difficulty dealing with challenges caused by IoT applications, such as network failure, latency, and capacity constraints. One of the introduced methods to solve these challenges is fog computing which makes the cloud closer to IoT devices. A system for dynamic congestion management brokerage is presented in this paper. With this proposed system, the IoT quality of service (QoS) requirements as defined by the service-level agreement (SLA) can be met as the massive amount of cloud requests come from the fog broker layer. In addition, a forwarding policy is introduced which helps the cloud service broker to select and forward the high-priority requests to the appropriate cloud resources from fog brokers and cloud users. This proposed idea is influenced by the weighted fair queuing (WFQ) Cisco queuing mechanism to simplify the management and control of the congestion that may possibly take place at the cloud service broker side. The system proposed in this paper is evaluated using iFogSim and CloudSim tools, and the results demonstrate that it improves IoT (QoS) compliance, while also avoiding cloud SLA violations.

Keywords: cloud service broker; quality of service; round robin; weighted fair queuing; fog computing; constrained application protocol; simple network management protocol

check for updates

Citation: Al Masarweh, M.; Alwada'n, T.; Afandi, W. Fog Computing, Cloud Computing and IoT Environment: Advanced Broker Management System. *J. Sens. Actuator Netw.* **2022**, *11*, 84. https://doi.org/10.3390/jsan11040084

Academic Editor: Macarena Espinilla-Estevéz

Received: 29 October 2022
Accepted: 7 December 2022
Published: 9 December 2022

1. Introduction

1.1. Cloud Computing

Cloud computing is considered as a massive dynamic and distributed resource of storage, services, and computing power available on request to users over the Internet [1,2]. Conventionally, resources in cloud computing are located in a huge data storage center and are managed and controlled by a third party, who offers computing infrastructures for the cloud users to be used by anyone from anywhere via the Internet [3]. Cloud computing presents a huge resource that can be reached on demand and distributed over the Internet [4–8]. Figure 1 illustrates the cloud computing model environment for cloud services allow users (including persons, businesses, and customers) to exploit software and hardware that are managed at remote locations by third parties. Such cloud examples include webmail, online file storage, and social networking sites [9]. Cloud computing has quickly become a very popular and widely used network system because of its cost efficiency, reducing user costs and increasing profitability for cloud providers [10–12]. Cloud computing systems can access computer resources and data from anywhere that has an Internet or even a network connection [13].

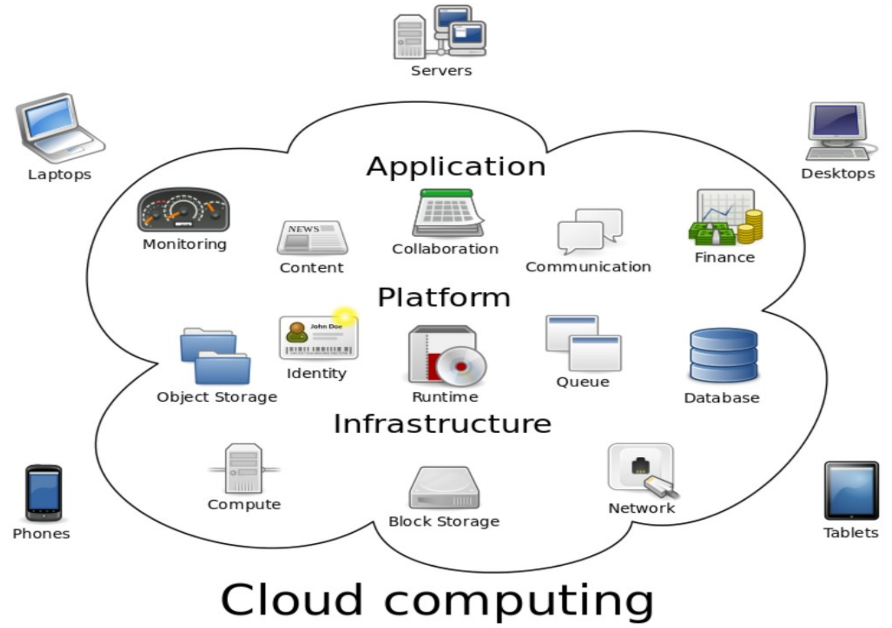

Figure 1. Cloud Computing Environment [14].

Service requesters of the cloud could pay providers for the cloud service based on specific charging types such as pay-per-use, the same as public utilities (for example, telephone, gas, power, and water), whereby customers can be charged on a monthly or quarterly basis based on their use of the fixed units [15,16].

At the current juncture, cloud computing has expanded extensively in applications and numerous usages, which is the most important part of the future generation of affordable computing infrastructure and services. Its advantageous characteristics will allow the users of the cloud to operate and develop their resources based on a pay-per-use basis, as shown in Figure 2 [17]. Cloud services have many models such as platform as a service (PaaS), infrastructure as a service (IaaS), and software as a service (SaaS), which are offered by the providers of the cloud, as described below.

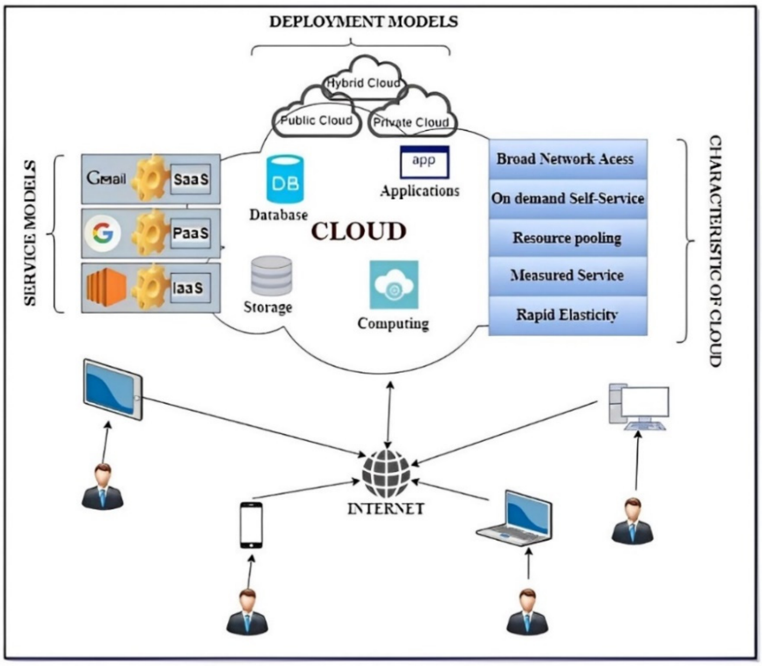

Figure 2. Cloud Computing Overview [17].

IaaS offers virtualizing resources as needed (on demand) as part of a service business model [18–20], where the provider in this model is the owner of the tools and the equipment, whose use it offers cloud users virtually instead of purchasing them (pay-per-use is used in this situation) [14]. A web-based operation management console (graphical user interface) is used for this purpose. Furthermore, this model of service allows cloud users to implement self-service once the provider's services have been utilized.

The cloud service of SaaS's model is the most widespread compared with other models, whereby the user can utilize and access the applications without the necessity to buy or download them. The provider provides the client with storage space to rent [18–21]. Additionally, the SaaS services can be used and accessed by the cloud users through a web browser.

PaaS enables cloud users to rent hardware and software infrastructures from the cloud providers, using the Internet to run their applications [18,19,22]. PaaS is mostly used by application developers to build and test their applications.

1.2. Internet of Things (IoT)

IoT refers to the increasing number of physical objects which are being linked to the Internet at an extraordinary rate. There are many examples of these objects, which include control and monitoring systems such as air conditioning, heating, and ventilation in smart homes. The IoT allows physical objects to hear, think, and see, and to create tasks that allow them to communicate with each other in order to coordinate decisions and information sharing. The IoT converts these objects from traditional forms to smart entities by utilizing their underlying technologies. Examples of these smart objects are universal and common computing, communication technologies, embedded devices, sensor networks, and applications' Internet protocols [23]. Figure 3 shows the general conceptualization of the IoT environment.

IoT applications have spread from the home area network, smart buildings, industrial automation, and smart transportation to pervasive healthcare [24–26]. For example, smart homes will deeply count on IoT devices to identify possible gas leakages, ambient temperature, air quality, house intrusions, and many other parameters regarding homes and their residents [27].

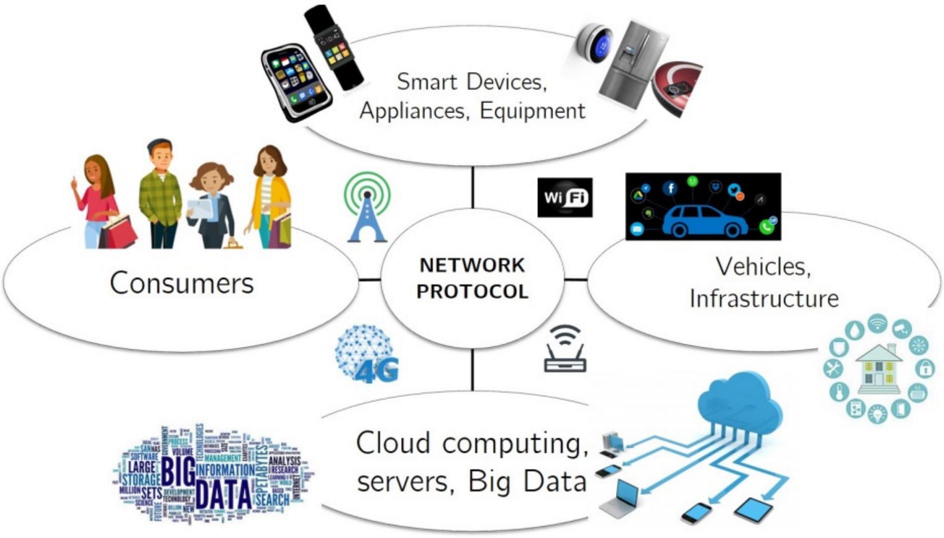

Figure 3. IoT Environment [28].

1.3. Fog Computing

As the IoT is distinguished by limited storage and processing power, it also has relatively weak performance, privacy, reliability, and security. Combining IoT and cloud networks in the Cloud of Things (CoT) is one of the best ways to overcome almost all the barriers to the IoT [29]. Moreover, the CoT can streamline the processes of IoT data collection, processing, integration, and deployment [2,29]. The creation of new IoT applications is complicated due to the huge number of IoT devices with heterogeneous paradigms [30]. Sensors and other devices provide huge amounts of data in IoT applications, generating Big Data that are later processed and analyzed to decide on suitable responsive actions. Data collected by sensors must be analyzed in the cloud, which requires a high bandwidth for the network used. Thus, these issues can be solved by using fog computing [30,31].

Several fields, especially the IoT, can benefit from fog computing, a new technology that offers many advantages. It was invented by Cisco and provides a way to process data near the information source, making it more rapidly accessible and efficient [32]. IoT users can take advantage of fog computing to access services such as storage and data processing. Just as the cloud delivers services to users, fog computing provides a way for IoT devices to connect to and use these services. Instead of transferring data to the cloud, fog computing offers local storage and processing capabilities to fog devices. Storage, computing, and networking resources are available through both cloud computing and fog computing [33]. As illustrated in Figure 4, fog computing extends the cloud to bring the environment close to the things that interact with IoT data (such as user devices).

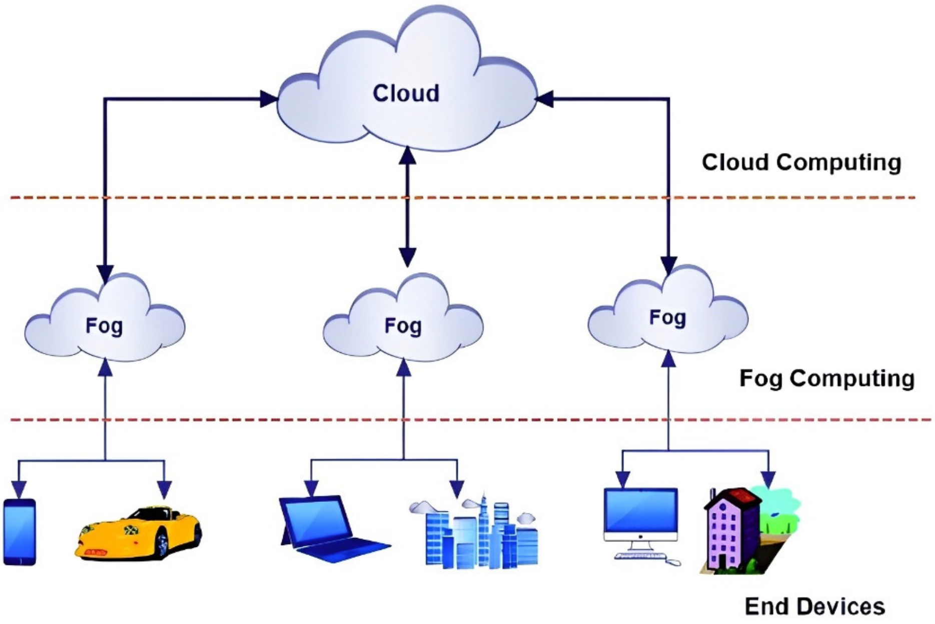

Figure 4. In fog computing, the cloud is brought closer to end user devices [31].

Fog computing can be seen as an extension of cloud computing to the edge of the network, where data are processed near the source instead of being sent to the cloud. With the increasing abundance of sensors, more and more data are being generated. In the past, these data were processed and stored in the cloud. However, this approach has several drawbacks, including latency and network congestion [34].

Fog as a service (FaaS) is a new class of service made possible by the Internet of Things and fog computing. Here, service providers create arrays of fog nodes across geographic areas, which act as ownerless resources that can be rented by businesses from many different industries. Every fog node has local computation, storage, and networking capabilities, making it an ideal solution for distributed data processing and analysis [35]. FaaS affords a new method for businesses to offer services to clients. Rather than the large companies that typically operate clouds at different scales, FaaS provides public and private computing, control, and storage services to both large and small businesses. As a result, they can meet the needs of a wide range of clients [36].

To explain how the fog works, the type of directed data from the fog should be identified first. Developers either create fog node ports or IoT applications at the edge of the network. The fog nodes that are near the network edge absorb the data collected from IoT devices, then the fog IoT application directs these data to the optimum place for analysis. The three types of data are [37]:

- The most time-sensitive data: the fog node analyzes these types of data near the things (e.g., sensor systems) that produce the data.
- Data that can be delayed for minutes or seconds for response or actions: these data are directed to an aggregation node for analysis, evaluation, and action.
- Data that are less time-sensitive to delay are forwarded to the cloud for archiving and historical analysis, long-term storage, and Big Data analytics.

Our contributions in this paper can be distilled into the following points:

1. Proposal of a new Dynamic Congestion Management Brokering (DCMB) system which is essentially adopted from Cisco queuing models. The DCMB can handle urgent requests (jobs) from the fog layer.
2. Additionally, this system can manage the enormous volume of cloud requesters while taking into account the QoS requirements of the customers as enforced by the SLA.
3. Finally, the DCMB system is used to provide cloud providers with jobs and monitor their progress.

The remainder of this paper is organized as follows. Section 2 reviews related literature concerning cloud and fog computing models. Section 3 presents the proposed system model, which is evaluated in Section 4. Finally, the conclusions are presented in Section 5.

2. Literature Review

In this section, the most popular models for cloud and fog computing are reviewed, explaining their resource monitoring and provisioning mechanisms, and it describes CSB jobs. With this information, we can better understand the strengths and weaknesses of each model and choose the one that is best suited to our needs.

The system of DRPM has been presented by Al-Ayyoub et al. [38]. This multi-agent system was designed to control the cloud provider's resources while considering the customers quality of service requirements, which are controlled by the SLA. In the event of physical machine overload, the DRPM system's host fault detection (HFD) algorithm determines virtual machines to mitigate the issue. This is achieved by taking into account the source of the overload and making an informed decision accordingly. The system of DRPM has been evaluated and tested by the CloudSim tool, in order to utilize the resource enhancement, mitigate the power consumption, and avoid violations of the SLA.

The Aneka system was designed by Vecchiola et. [39] which provides a PaaS for cloud environments by the platform of NET-based applications. This system makes it easy to manage, deploy, and develop applications in the cloud. APIs and runtime environment applications are presented by this system, which can be used on public and private cloud computing paradigms, such as GoGrid and Amazon EC2. Furthermore, the Aneka model similarly proposes techniques of resource monitoring according to SLA guidance. This method helps ensure that new jobs from cloud users are completed on time and within the parameters set by SLAs. It does this by estimating the time required to complete these tasks and matching them with available resources. The system keeps running if the calculated completion time for the new workloads matches; else, it keeps running cloud resources to stay within SLAs.

Fog computing has been described by Peter [33] along with its real-time applications, and it has been shown that fog computing can run and handle Big Data produced by IoT devices. Additionally, it was demonstrated that fog computing may address latency and congestion problems. This method works by estimating the time required to complete new tasks from cloud users and then matching them up with the available resources and the SLA deadline time.

Chiang and Zhang [36] showed that the IoT and fog computing are two important emerging technologies that are beginning to see more integration. They give an overview of some of the challenges and concerns that come with evolving IoT systems, and how fog computing can help to address them. Additionally, they discussed how to develop new business opportunities by using modern storage, computing, and networking architectures. In addition to reviewing the characteristics and advantages of this fog architecture, the authors also recommended solutions for several IoT problems.

Other related papers have reviewed fog computing architectures, such as Dastjerdi et al. [40]. Instead of using the cloud, their model operates the IoT in the local fog. In the suggested architecture, fog services are positioned within a software-defined resource management layer [41]. By using cloud-based middleware, fog cells are analyzed, arranged, and provisioned.

Kong et al. [42] analyzed how fog computing is accompanied and spread by cloud computing and examined how the former integrates with the IoT. Using traffic lights and wind farms as use cases, their architecture was evaluated.

Agarwal et al. [43] described the CSB architecture, which is shown in Figure 5, where the CSB architecture is split into four parts:

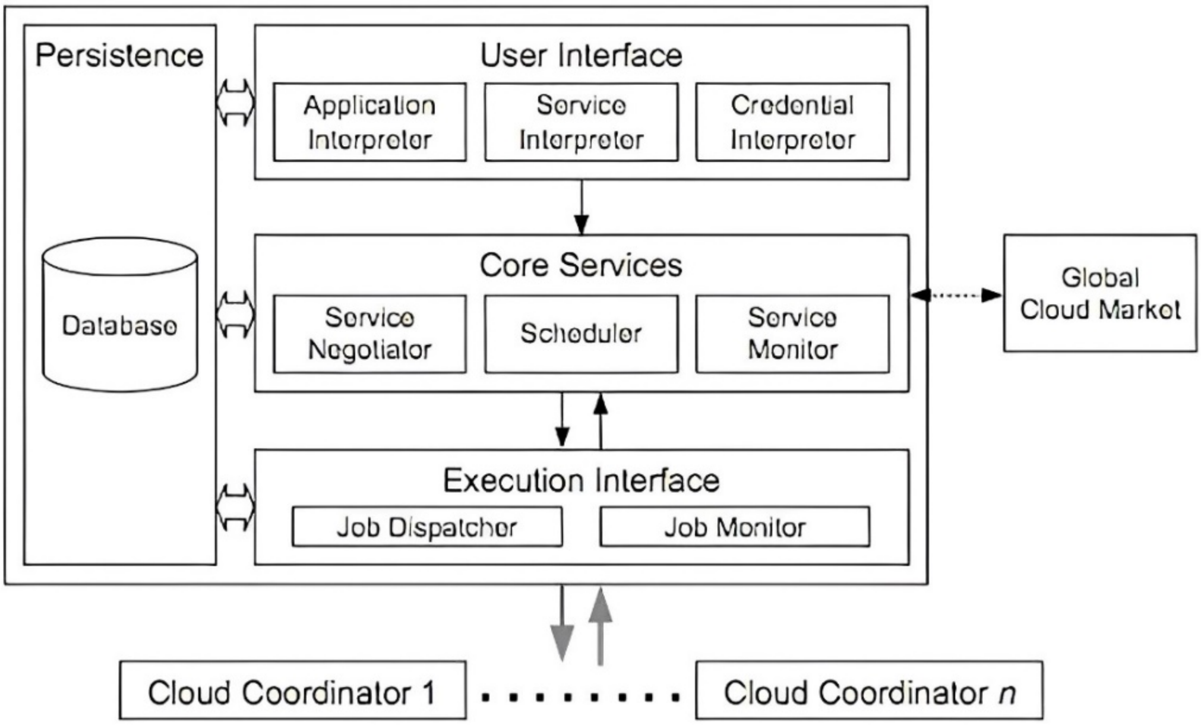

Figure 5. Cloud Service Broker Architecture [43].

A: The User Interface between the user and the CSB serves as a link between the two. An Application Interpreter describes how tasks should be performed, what QoS should be provided, and what is to be implemented by the user. Moreover, the service requisites needed for execution are identified by the Service Interpreter. A few other essential services are included in these requirements, such as the type of service and the location of the service. To gain access to required services, the Credential Interpreter examines the credentials.

B: The Core Services layer is used by the CSB to match user requests with the most appropriate cloud services. The Service Negotiator gathers requirements from the User Interface and passes them on to the Scheduler, which then uses these requirements to identify which cloud services will best suit the user's needs. The Service Monitor is a constantly running program to check the cloud services' availability and to find available new services. This allows tracking of the state of the cloud and ensures that services are always up and running.

C: The Execution Interface is where the Job Dispatcher executes user applications, by combining data files with the user application code, and sending the resultant packages to cloud resources. This layer provides the necessary infrastructure to make this happen. The Job Monitor is an essential tool for tracking the progress of a job and ensuring that the results are delivered to the customer upon completion.

D: The Persistence Layer is key to maintaining the User Interface state, Execution Interface, and Core Services in the event of broker failure.

Alwada'n et al. [44] presented a new system called Dynamic Congestion Management (DCM) that uses multiple agents to consider various elements when making decisions, such as cloud providers' limitations and the number and types of resources available, which are essential factors in decision-making, customers' QoS requirements, and customers' conditions (service demands). Figure 6 shows the Dynamic Congestion Management (DCM) system architecture proposed in [44]. The system is partitioned into three parts.

A: Each client is given a local agent by the Cloud Service Users section, who responds to queries in order of priority. Additionally, the agent is responsible for sending the marked requests to the Classifier inside the CSB.

B: In the case of job congestion, the Cloud Service Broker Classifier identifies the jobs' priority determining which should be run first. When cloud users continue sending job requests to the CSB in large numbers, the CSB might randomly reject some of these requests. The Classifier in the DCM system seeks to deploy the a priori task categorizations of the local agent to manage the requests flowing to the CSB in order to control the random rejection of requests. If the CSB is not overloaded, the classifier will implement the first-in, first-out (FIFO) order for the incoming requests. Conversely, the weighted fair queuing (WFQ) mechanism will be applied by the Classifier to resolve the congestion.

C: The Service Provider's physical and virtual machines, including bandwidth, CPU, storage, RAM, etc., receive job requirements from the CSB via agents for every single resource, which report to the Job Monitor with a periodic update message regarding the job status.

Due to restricted resources (memory, energy, etc.), the calculation of shared data can be laborious and typically cannot be performed by IoT devices themselves [45]. Companies have been able to overcome this capacity restriction by outsourcing intense computer processes to the cloud thanks to the adoption and use of the cloud paradigm. However, this offloading comes at a cost to the quality of service (QoS) provided, including increased latency brought on by the distance between the cloud and the end devices, network overhead, and increased security and privacy risk. Over the past few years, the edge computing paradigm has been advocated as a way to lessen this penalty. Edge computing enables the transfer of computing tasks to nodes located closer to end devices (at one hop from them). As a result, these duties are closer to the data source and the data consumer, improving the quality of service provided [46].

However, edge computing, according to some researchers, addresses the offloading of computing work from the cloud to the last hop before smart devices, while others claim it just includes the collection of devices at the end of the computer chain, including IoT devices, etc. [47]. It becomes essential to adopt technology that will allow for real-time data collecting and analysis while also ensuring process and product quality [48]. Edge computing (EC) must thus be introduced in order to guarantee the quality of the production process and final goods (zero defect manufacturing) [48].

Additionally, some studies advise the use of edge computing for various domains, applications with stringent reaction times, and for those who wish to lower the cost of their infrastructure or control their privacy better. The particular coverage, chosen areas, and advantages of edge computing are therefore not presently agreed upon [49].

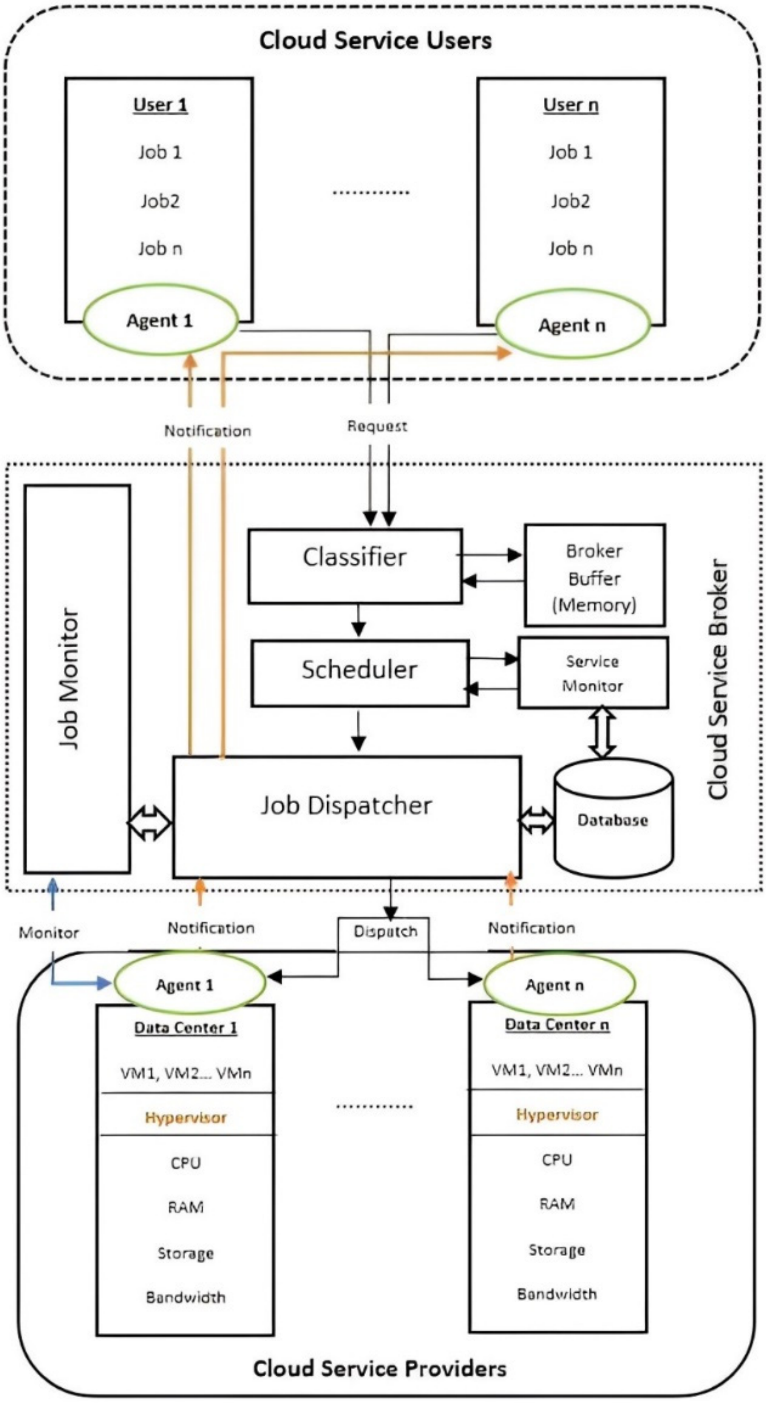

Figure 6. Architecture of Dynamic Congestion Management (DCM) System [44].

3. System Model

The proposed multi-agent DCMB system is designed to take into account several factors when making decisions, utilizing different agents to make informed decisions. This system was created in order to improve traffic conditions by reducing congestion and improving the flow of traffic. Some of the factors that are considered when making decisions include urgent requests (jobs) coming from the fog layer, the huge amount of cloud requesters, customer technical and QoS requirements, and the scope and limitations of cloud provider resources. This system is essentially adopted from Cisco's queuing algorithms [50] and the DCM system proposed by Alwada'n et al. [44].

Figure 7 shows the DCMB system's architecture, which comprises four main sections: the Cloud Users, Cloud Service Broker, Fog Service Broker, and Cloud Service Provider sections.

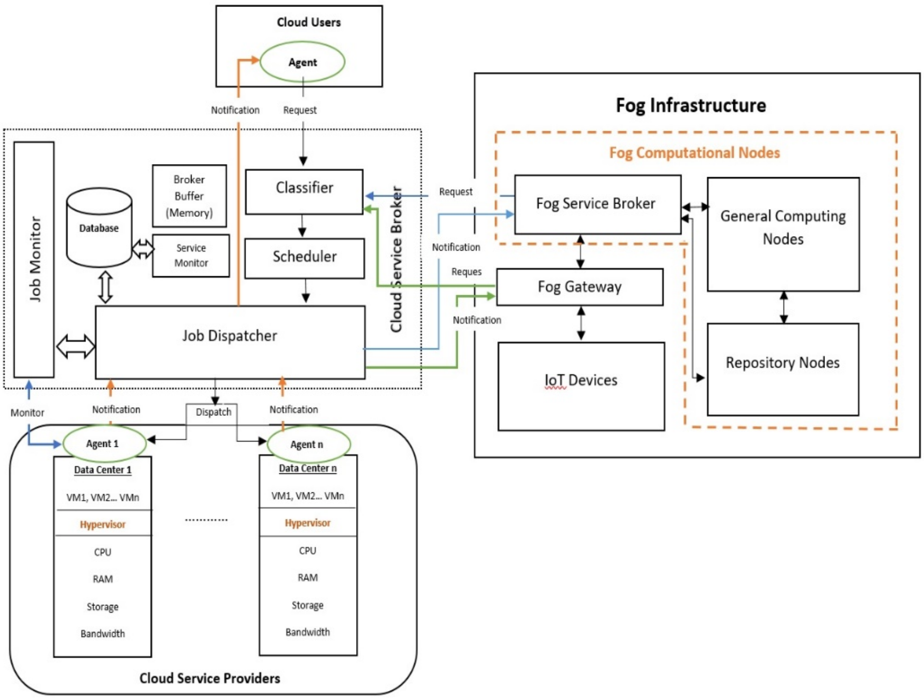

Figure 7. Architecture of Dynamic Congestion Management Brokering (DCMB) System.

3.1. Part 1: The Cloud Users

This part assigns a local agent to each customer, which remarks on customers' requests according to their priority, and forwards the distinguished requests to the CSB (specifically to its Classifier entity). The marking that is carried out by the local agent is considered as an additional paid service, whereby clients can purchase priority cloud use. If this service is not chosen, the level of priority will be assigned as a law by the local agent. User requests contain job conditions (specifications), including in terms of hardware specifications, required software, and virtual machine (VM) type. The local agent marks the users' requests as low, medium, or high priority, which supports the user to address some demanding jobs when required.

3.2. Part 2: The Broker of Cloud Service

The cloud center is considered in this part. Cloud users (including IoT devices and fog brokers) submit jobs to the CSB, which is the core of the DCMB system. During congestion, the Classifier determines which job should be operated first. Due to the large number of requests that the CSB receives, it may be forced to delay or reject some requests. Instead of delaying or rejecting the requests as per FIFO order, the Classifier can utilize the request marking previously applied by the agent in both the user part and the fog broker in the fog layer, to classify the requests in a way that allows for the customers' QoS requirements. The FIFO algorithm is employed by the Classifier in order to process the ingress requests if there is no congestion in the CSB. In the event of congestion, the Classifier relies on the WFQ to resolve it. Streams are sorted into queues automatically by WFQ, which assigns a weight to each queue according to its priority. Initially, the process is a round robin, where time slices are allotted to each process in equal percentages, but once the round is completed, the traffic flow is passed by the processer, depending on its weight. The WFQ mechanism solves the starvation issue that may arise when round robin is used alone. Figure 8 explains the job of the Classifier while receiving cloud users' requests, the IoT devices' requests, and the fog broker requests by using the WFQ mechanism. Furthermore, the Broker Buffer is used by the Classifier, if the requests need an additional classification, and kept on hold.

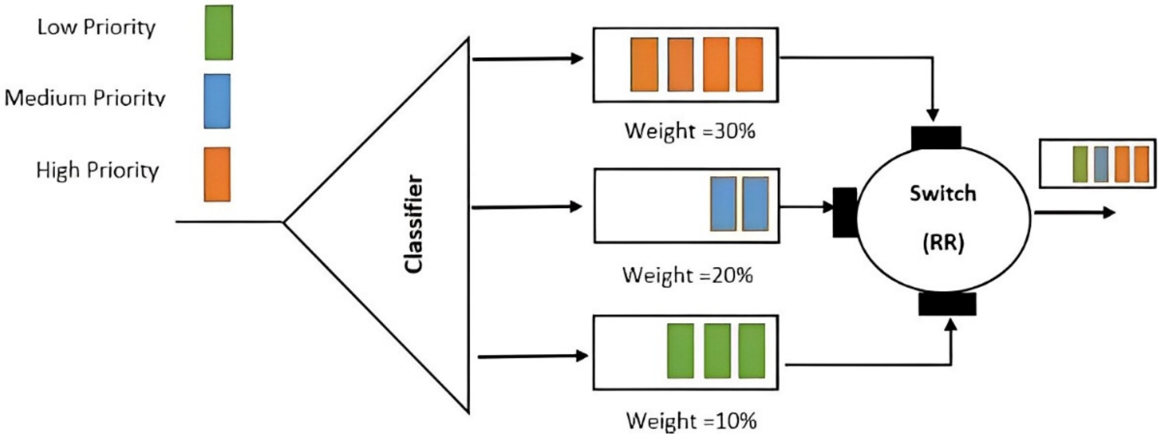

Figure 8. Classifier Mechanism [44].

In the next step, the Scheduler runs the requests the classifier selected to run first. In response to a user request, the Scheduler selects the most appropriate cloud service based on the request's application and requirements. Cloud service status is monitored by the Service Monitor and the new cloud services are periodically checked. Subsequently, the Job Dispatcher obtains the users' jobs and puts together the user application and data files which are sent to a particular cloud resource in order to execute them. The job running status is monitored by the Job Monitor in order to send the job outcomes again to the client once the jobs are completed. The job status can be viewed and managed by using an agent which is linked together with the cloud resource. In cases where the broker fails for any reason (e.g., the broker's system is down or has crashed), the state of every CSB entity is saved in a database in order to be recovered once the failure is fixed. Finally, once the task is finished, the agent connected to the resource directs the results back to the Job Dispatcher, which reports that to the Job Monitor while simultaneously sending the outputs back to the local agent at the cloud user side.

3.3. Part 3: The Broker of Fog Service

This part describes the high level of integration IoT with the edge-fog and their relationship with the cloud. Some of the components in this part are adapted from the model developed by Tuli et al. [51], which was clarified in a notably accomplished way. IoT devices observe the external environment and translate any given directive into physical actions. IoT devices are connected with nearby gateways by wired or wireless communication protocols. The Fog Gateway offers user interfaces for applications to facilitate users' credentials, report service expectations, access the backend program, receive service results, request resources from computing infrastructure according to availability, and manage IoT devices. It employs either the Constrained Application Protocol (CoAP) or Simple Network Management Protocol (SNMP) for communication. General Computing Nodes are responsible for data storage, application execution, and other operational management functions. Repository Nodes store the infrastructure and user credentials, as well as data from IoT devices; back up the application catalog; and increase the cloud storage to assist in data management operations.

The Fog Service Broker (FSB) is responsible for operating and managing the fog layer operations, including:

A: Finding suitable computing nodes to run the most time-sensitive data requiring a quick response from the fog layer, which cannot wait for feedback from the cloud.

B: Finding the suitable computing node(s) or the suitable repository node to run or store data that can be delayed for minutes or seconds for response or actions.

C: Sending less time-sensitive data to the cloud on a delayed basis, for archiving and historical analysis, long-term storage, and Big Data analytics.

D: In case there is a need to process very urgent operations, and the FSB cannot find suitable resources for these operations, it sends the request to the CSB for an urgent process. However, before sending this request, it informs the CSB of this urgent request by marking the request a high priority job. The CSB in turn applies the WFQ mechanism for classification, which helps to accelerate the processing of prioritized FSB requests.

Another point that should be noted is that in some cases the Fog Gateway may have the ability to communicate directly with the CSB without the need to communicate with the FSB first. This situation can take place in the case of a job that is less time-sensitive to delay.

3.4. Part 4: The Cloud Service Providers

This part includes the virtual and physical machines (CPU, storage, RAM, bandwidth, etc.). Every physical machine can have more than one VM and every VM has an agent that is accountable for job attributes received from the cloud broker and informing the Job Monitor by frequent messages of the condition of the job. In other words, the agent is responsible for monitoring the execution of jobs at the cloud providers.

4. Experiments and Results

To evaluate our proposed system, iFogSim [52] and CloudSim [53] were utilized. iFogSim is one of the major simulation platforms for the fog computing environments, and CloudSim is one of the main tools for simulating the cloud environment.

4.1. Requirements

Tables 1 and 2 show the settings for the physical and the virtual machines involved in the simulation. The simulation was divided into two sections with identical settings. The first simulation was split into two sections. For both, 500 jobs from 5 users from fog and cloud systems (100 each) were sent to the CSB. The simulation assumed four machines (three VMs built on one physical machine) were used as providers. Up to 300 cloudlets may be accepted by the CSB, but requests above that limit are rejected. The simulation period was 24 h, based on 300 s scheduling intervals. Table 3 shows the characteristics of the cloudlets (jobs) that were sent to the CSB.

Table 1. VM Attributes.

Attribute	Size/Quantity
Quantity	3
VM types	4
Computer’s raw CPU processing power	1000
No. PEs	1
RAM space	512
Frequency range	100 Mbit/s
Storage size	5 GB

Table 2. Physical Machine Attributes.

Attribute	Size/Quantity
No. hosts	1
VM types	4
Computer’s raw CPU processing power	1000
No. PEs	2
RAM space	2048
Frequency	1 Gbit/s
Storage size	10 GB

Table 3. Job Attributes.

Attribute	Size/Quantity
No. cloudlets	400
Job length	1000
File volume	300
Output volume	300
Number of PEs	1

The cloud or fog users did not provide any paid services for high or medium priority in the first section of the simulation. Every cloud user is represented by a local agent (4 users), and the FSB (1 user) assigns a "low" priority for each job. The characteristics of the FSB and IoT devices (sensors) used in the simulation are shown Tables 4–6.

Table 4. FSB Characteristics.

Attribute	Size/Quantity
Level	0
Uplink bandwidth	1000
Downlink bandwidth	1200
MIPS of CPU	5000
RAM capacity	45,000
Rata/MIPS	1000

Table 5. Sensor 1 Characteristics.

Attribute	Size/Quantity
Type	EEG
Distribution type	Normal
Mean	10
StdDev	5

Table 6. Sensor 2 Characteristics.

Attribute	Size/Quantity
Distribution type	Uniform
Min	20
Max	100
Distribution type	Uniform

"High" priority was assigned to the FSB's jobs in the second section, while no priority was assigned to users of the cloud. Each test measured the total processing time by sending all jobs simultaneously to the CSB. In consequence of the high volume of requests, the number of delayed or rejected jobs was also tested.

We compared and evaluated both sections of the experiment on several factors, including how long each took to complete and how many jobs were rejected.

4.2. Simulation Details

Section 1: Four cloud users sent 500 jobs to the CSB, and one cloud user (User 5) sent 100 jobs to the FSB. No priority was assigned at this stage, and FIFO requests were received and processed by the CSB. Regardless of the order in which the user requested the cloudlets, the first 300 cloudlets from the first three users were processed without any rejections. As the CSB reached the bounds of taking up jobs (in accordance with its policy), the fourth and fifth users' jobs were rejected. Both users re-sent their jobs later in order to be processed. Figure 9 shows the processing time and Figure 10 shows the jobs' rejection for the simulation in the first section.

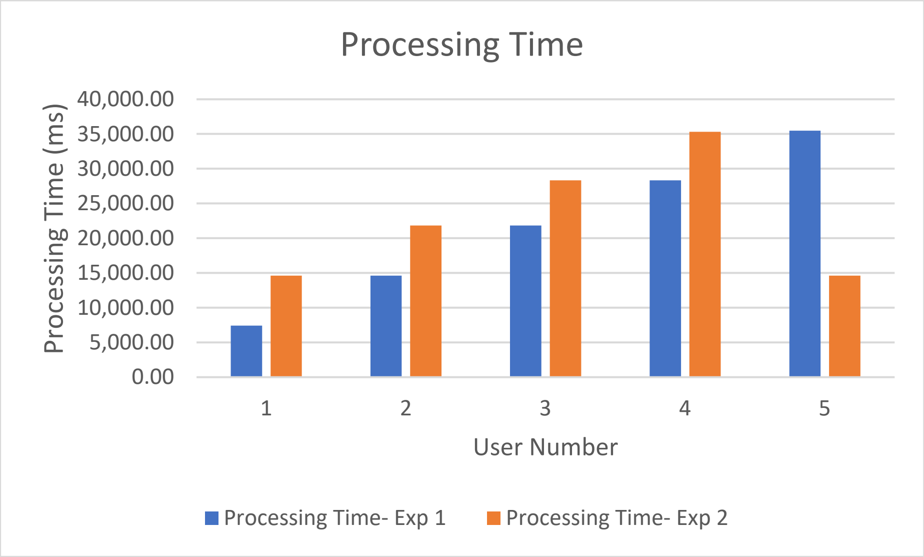

Figure 9. Time Required to Process Both sections—Experiment 1.

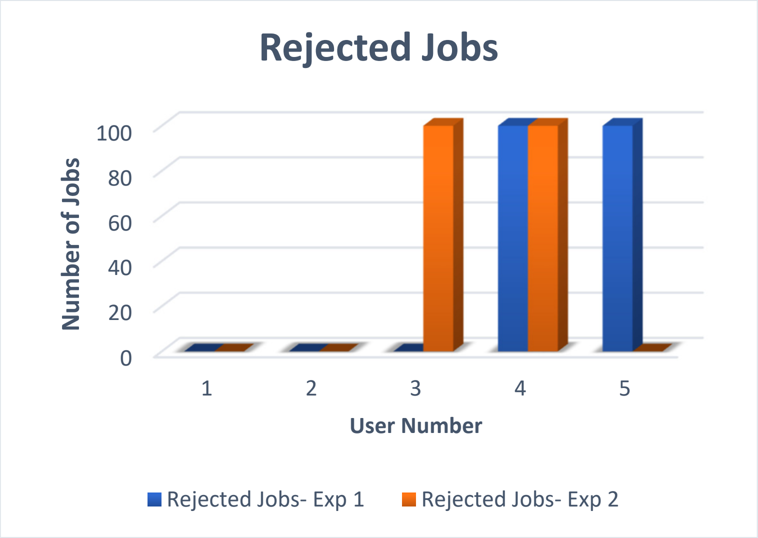

Figure 10. Job Rejections for Both sections—Experiment 1.

Section 2: Another 500 jobs were sent to the CSB from the same source, but the FSB marked its jobs for users as high priority (paid service) this time around. Other cloud users retained normal priority. Figure 9 shows the processing time and Figure 10 shows the rejected jobs for the second section of the simulation.

All the tasks were processed for User 5 by the FSB, illustrating the advantages of the priority service used. In addition, in comparison to the first section, the processing time was very short. Moreover, there were no rejected jobs for the FSB. Using the DCMB system gave a guarantee of finishing all the jobs in a short time. Based on CSB's FIFO mechanism, Users 1 and 2 still had no rejected jobs, while Users 3 and 4 had all of their jobs rejected.

Another experiment was conducted, with the same settings for the physical machine, the virtual machines, the characteristics of the cloudlets (jobs), and the characteristics of the FSB and IoT devices (sensors) used in the first simulation (Tables 1–6). Five hundred jobs were sent, including 100 each from four cloud users, and 100 from the FSB. Additionally, up to 300 cloudlets were accepted by the CSB, and any requests above that were rejected. An interval of 300 s was used in the simulation time of 24 h.

Once more, the simulation was divided into two sections, which were evaluated in terms of several factors, such as the completion time and rejection rate for jobs.

4.3. Simulation Details

Section 1: The CSB received 400 jobs from cloud Users 1 to 4, and 100 from the FSB (User 5), with no priority being assigned. The CSB received the jobs and managed them as FIFO. The first 300 jobs coming from the first three users were processed without any job rejection. As per its policy, when the fourth and fifth users' jobs came to the bounds of the limited jobs level, they were rejected by the CSB. It took two users longer to process their jobs after they re-sent them. Figures 11 and 12 show the time processing and the jobs' rejection for the experiment in the first section.

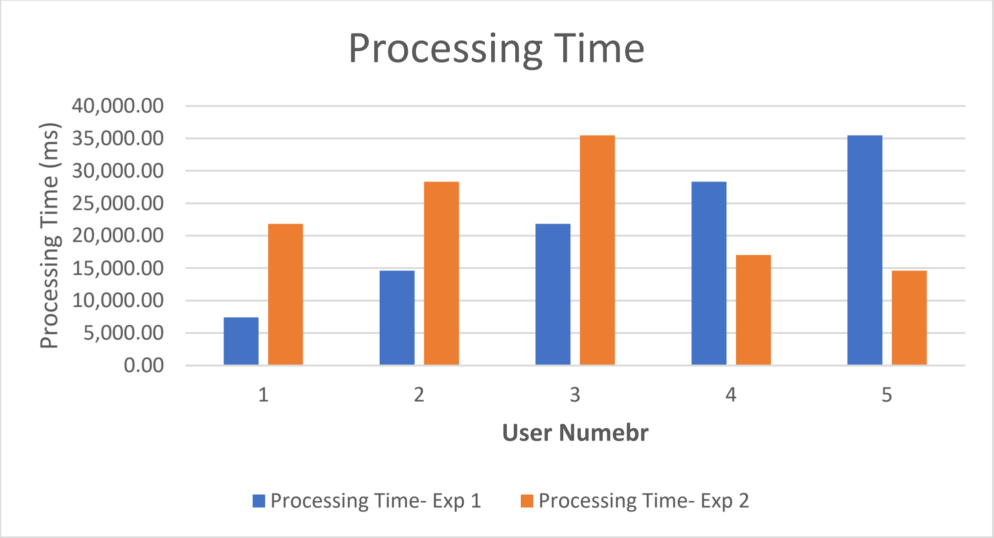

Figure 11. Time Required to Process Both sections—Experiment 2.

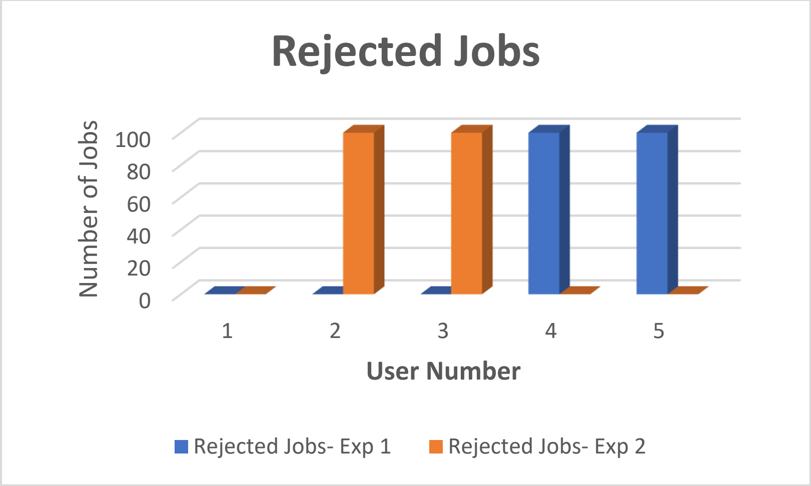

Figure 12. Job Rejections for Both sections—Experiment 2.

Section 2: The CSB received 400 jobs from cloud Users 1 to 4, and 100 jobs from the FSB (User 5). For the fourth user, the local agent marked the jobs as "medium" priority (as per a paid service), and for User 5, the FSB marked them as "high" priority. Other users retained normal (i.e., "low") priority. Figure 11 shows the processing time and Figure 12 shows the jobs' rejection for the simulation in the first section.

The results in Figures 11 and 12 show that Users 5 (FSB) and 4 (cloud user), who made use of the paid priority service introduced by the DCMB system, had all of their jobs processed, and this was achieved in a shorter time compared to the first stage of the experiment. For those two users, there were no delayed jobs. This is because the CSB processed their jobs according to FIFO. Users 1, 2, and 3 had 100 rejected jobs each, while User 1 had no rejected jobs.

5. Conclusions

IoT applications are rapidly transforming traditional cloud computing models, which are facing challenges such as capacity, latency, and network failures with the fast development of IoT applications and escalating demands and expectations from users. Fog computing usually processes and stores data locally among IoT devices instead of sending them to the cloud. However, in cases where there is a need to send some sensitive data requiring a fast response time to the cloud in order to process them, a huge amount of cloud requesters coming from the fog broker layer can overload latent resources and incur technical problems and reduced QoS. The paper presents a (DCMB) system that satisfies the IoT SLA requirements for QoS. The results in our experiment show that the user from the fog environment and the other users from the cloud environment who made use of the paid priority service introduced by the DCMB system had all of their jobs processed, and this was achieved in a shorter time (15,000 ms) compared to the first stage of the experiment (35,000 ms). As part of the job running, it also provisioned and monitored the cloud providers' work. The proposed DCMB system was evaluated by using iFogSim and CloudSim, and the results indicated that using DCMB enhanced the degree to which the IoT's QoS requirements were met during resource utilization and avoided the cloud SLAs being violated. The experiments show that the processing time needed by the DCMB system was very short compared to the traditional cloud system and the number of rejected jobs which resulted from using the DCMB system is lower than the traditional cloud system. Our suggested system has some limitations, such as not having been tested with a large number of jobs. Only 500 jobs were used in the experiments. In order to check its usability and performance, we plan on extending our proposed system and testing it using a large number of jobs.

Author Contributions: Conceptualization, M.A.M. and T.A.; methodology, M.A.M.; validation, M.A.M. and A.T; formal analysis, T.A.; investigation, T.A.; resources, M.A.M. and W.A.; data curation, M.A.M. and T.A.; writing—original draft preparation, M.A.M., T.A. and W.A.; writing—review and editing, M.A.M. and T.A.; visualization, T.A.; supervision, M.A.M. and T.A.; funding acquisition, M.A.M. and W.A.; All authors have read and agreed to the published version of the manuscript.

Funding: The Deanship of Scientific Research (DSR) at King Abdulaziz University, Jeddah, under grant no. G: 365-849-1443.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This project was funded by the Deanship of Scientific Research (DSR) at King Abdulaziz University, Jeddah, under grant no. G: 365-849-1443. The authors, therefore, acknowledge with thanks DSR for technical and financial support.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Alouffi, B.; Hasnain, M.; Alharbi, A.; Alosaimi, W.; Alyami, H.; Ayaz, M. A Systematic Literature Review on Cloud Computing Security: Threats and Mitigation Strategies. *IEEE Access* **2021**, *9*, 57792–57807. [CrossRef]
2. Alwada'n, T.; Al-Zitawi, O.; Khawaldeh, S.; Almasarweh, M. Privacy and Control in Mobile Cloud Systems. *Int. J. Comput. Appl.* **2015**, *113*, 12–15.
3. Stergiou, C.; Psannis, K.E.; Gupta, B.B.; Ishibashi, Y. Security, privacy & efficiency of sustainable Cloud Computing for Big Data & IoT. *Sustain. Comput. Inform. Syst.* **2018**, *19*, 174–184. [CrossRef]
4. Mbarek, N. Service Level Management in the Cloud. *Serv. Level Manag. Emerg. Environ.* **2021**, *20*, 45–81.
5. Washizaki, H.; Xia, T.; Kamata, N.; Fukazawa, Y.; Kanuka, H.; Kato, T.; Yoshino, M.; Okubo, T.; Ogata, S.; Kaiya, H.; et al. Systematic Literature Review of Security Pattern Research. *Information* **2021**, *12*, 36. [CrossRef]
6. Nayyar, A. *Handbook of Cloud Computing: Basic to Advance Research on the Concepts and Design of Cloud Computing*; BPB Publications: Noida, India, 2019.
7. Butpheng, C.; Yeh, K.H.; Xiong, H. Security and privacy in IoT-cloud-based e-health systems—A comprehensive re-view. *Symmetry* **2020**, *12*, 1191. [CrossRef]
8. De Sousa, N.F.; Perez, D.A.; Rosa, R.V.; Santos, M.A.; Rothenberg, C.E. Network service orchestration: A survey. *Comput. Commun.* **2019**, *142*, 69–94. [CrossRef]
9. Alwada'n, T. Mobility in Cloud Systems. *Int. J. Comput. Trends Technol. (IJCTT)* **2015**, *11*, 202–205. [CrossRef]
10. Bello, S.A.; Oyedele, L.O.; Akinade, O.O.; Bilal, M.; Delgado, J.M.D.; Akanbi, L.A.; Ajayi, A.O.; Owolabi, H.A. Cloud computing in construction industry: Use cases, benefits and challenges. *Autom. Constr.* **2021**, *122*, 103441. [CrossRef]
11. Services, A.W. Aws Economics Center. May 2012. Available online: https://aws.amazon.com/economics/ (accessed on 28 October 2022).
12. Sanger, A.K.; Johari, R. Survey of Security Issues in Cloud. In Proceedings of the 2022 International Mobile and Embedded Technology Conference (MECON), Noida, India, 10 March 2022; pp. 490–493.
13. The Office of the Privacy Commissioner of Canada, O.: Introduction to Cloud Computing. October 2011. Available online: https://www.priv.gc.ca/resource/fs-fi/02_05_d_51_cc_e.asp (accessed on 28 October 2022).
14. Chawla, I. Cloud Computing Environment: A Review. *J. Int. J. Comput. Technol.* **2018**, *17*. [CrossRef]
15. Jamsa, K. *Cloud Computing*; Jones & Bartlett Learning Publisher: Burlington, VT, USA, 2022; ISBN 9781284233971.
16. Alwada'n, T.; Al-Zitawi, O.; Atoum, J. Cloud Computing: Privacy, Mobility and Resources Utilization. *Int. J. Comput. Trends Technol. (IJCTT)* **2016**, *41*, 29–36. [CrossRef]
17. Mohammad, H.; Zaman, K.R. A Systematic Review on Cloud Computing. *Int. J. Comput. Sci. Eng.* **2018**, *6*, 632–639.
18. Alwada'n, T. Cloud Computing Topology: Towards Enhancing the Performance. *Int. J. Comput. Sci. Inf. Secur.* **2016**, *14*, 654–658.
19. Alwada'n, T. Cloud Computing and Multi-Agent System: Monitoring and Services. *J. Theor. Appl. Inf. Technol.* **2018**, *96*, 2435–2444.
20. Ahluwalia, J.K.; Mouradian, C.; Alam, M.N.; Glitho, R. A Cloud Infrastructure as a Service for an Efficient Usage of Sensing and Actuation Capabilities in Internet of Things. In Proceedings of the NOMS 2022–2022 IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, 25–29 April 2022; pp. 1–6.
21. Quattra Inc. *Quattra Cloud Vision and Frame Work Value*. March 2017, pp. 1–5. Available online: https://silo.tips/download/quattra-s-cloud-vision-framework-value (accessed on 8 December 2022).
22. Yasrab, R. Platform-as-a-Service (PaaS): The Next Hype of Cloud Computing. *arXiv* **2018**, arXiv:1804.10811.
23. Al-Fuqaha, A.; Guizani, M.; Mohammadi, M.; Aledhari, M.; Ayyash, M. Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Commun. Surv. Tutorials* **2015**, *17*, 2347–2376. [CrossRef]
24. Nalin, M.; Baroni, I.; Mazzara, M. A Holistic Infrastructure to Support Elderlies' Independent Living. In *Encyclopedia of E-Health and Telemedicine*; IGI Global: Hershey, PA, USA, 2016.
25. Salikhov, D.; Khanda, K.; Gusmanov, K.; Mazzara, M.; Mavridis, N. Microservice-based IoT for Smart Buildings. In Proceedings of the 31st International Conference on Advanced Information Networking and Applications Workshops (WAINA), Taipei, Taiwan, 27–29 March 2017.
26. Salikhov, D.; Khanda, K.; Gusmanov, K.; Mazzara, M.; Mavridis, N. Jolie Good Buildings: Internet of things for smart building infrastructure supporting concurrent apps utilizing distributed microservices. In Proceedings of the 1st International Conference on Convergent Cognitive Information Technologies, Moscow, Russia, 25–26 November 2016; pp. 48–53.
27. De Donno, M.; Giaretta, A.; Dragoni, N.; Bucchiarone, A.; Mazzara, M. Cyber-Storms Come from Clouds: Security of Cloud Computing in the IoT Era. *Futur. Internet* **2019**, *11*, 127. [CrossRef]
28. Yashodhan, A.; Sridhar, K. A Device-Independent Efficient Actigraphy Signal-Encoding System for Applications in Monitoring Daily Human Activities and Health. *Sensors* **2018**, *18*, 2966. [CrossRef]
29. Atlam, H.F.; Alenezi, A.; Alharthi, A.; Walters, R.; Wills, G. Integration of cloud computing with Internet of things: Challenges and open issues. In Proceedings of the 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Exeter, UK, 21–23 June 2017; pp. 670–675.
30. Ai, Y.; Peng, M.; Zhang, K. Edge cloud computing technologies for Internet of things: A primer. *Digit. Commun. Netw.* 2017; *in press*.
31. Atlam, H.F.; Walters, R.J.; Wills, G.B. Fog Computing and the Internet of Things: A Review. *Big Data Cogn. Comput.* **2018**, *2*, 10. [CrossRef]

32. Wang, X.; Han, Y.; Leung, V.C.M.; Niyato, D.; Yan, X.; Chen, X. Convergence of Edge Computing and Deep Learning: A Comprehensive Survey. *IEEE Commun. Surv. Tutorials* **2020**, *22*, 869–904. [CrossRef]
33. Peter, N. FOG Computing and Its Real Time Applications. *Int. J. Emerg. Technol. Adv. Eng.* **2015**, *5*, 266–269.
34. Wen, Z.; Yang, R.; Garraghan, P.; Lin, T.; Xu, J.; Rovatsos, M. Fog Orchestration for Internet of Things Services. *IEEE Internet Comput.* **2017**, *21*, 16–24. [CrossRef]
35. Yang, Y. FA2ST: Fog as a Service Technology. In Proceedings of the 2017 IEEE 41st IEEE Annual Computer Software and Applications Conference, Turin, Italy, 4–8 July 2017; p. 708.
36. Chiang, M.; Zhang, T. Fog and IoT: An Overview of Research Opportunities. *IEEE Internet Things J.* **2016**, *3*, 854–864. [CrossRef]
37. Cisco, Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are. April 2015. Available online: https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf (accessed on 14 July 2020).
38. Al-Ayyoub, M.; Jararweh, Y.; Daraghmeh, M.; Althebyan, Q. Multi-agent based dynamic resource provisioning and monitoring for cloud computing systems infrastructure. *Clust. Comput.* **2015**, *18*, 919–932. [CrossRef]
39. Goudarzi, M.; Ilager, S.; Buyya, R. Cloud Computing and Internet of Things: Recent Trends and Directions. In *New Frontiers in Cloud Computing and Internet of Things*; Springer International Publishing: Cham, Switzerland, 2022; pp. 3–29. ISBN 978-3-031-05528-7.
40. Dastjerdi, A.V.; Gupta, H.; Calheiros, R.N.; Ghosh, S.K.; Buyya, R. Fog Computing: Principles, architectures, and applications. In *Internet of Things: Principles and Paradigms*; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 2016; pp. 61–75.
41. Suárez-Albela, M.; Fernández-Caramés, T.M.; Fraga-Lamas, P.; Castedo, L. A practical evaluation of a high-security ener-gy-efficient gateway for IoT fog computing applications. *Sensors* **2017**, *17*, 1978. [CrossRef] [PubMed]
42. Kong, L.; Tan, J.; Huang, J.; Chen, G.; Wang, S.; Jin, X.; Zeng, P.; Khan, M.K.; Das, S.K. Edge-Computing-Driven Internet of Things: A Survey. *ACM Comput. Surv.* 2022; Just Accepted. [CrossRef]
43. Agarwal, K.; Singh, S. A Survey on Infrastructure Platform Issues in Cloud Computing. *Int. J. Sci. Eng. Res.* **2012**, *3*.
44. Alwada'n, T.; Al-Tamimi, A.; Mohammad, A.H.; Salem, M.; Muhammad, Y. Dynamic Congestion Management System for Cloud Service Broker. *Int. J. Electr. Comput. Eng. (IJECE). Febr.* **2023**, *13*, 872–883. [CrossRef]
45. Bellavista, P.; Berrocal, J.; Corradi, A.; Das, S.K.; Foschini, L.; Zanni, A. A survey on fog computing for the internet of things. *Pervasive Mob. Comput.* **2019**, *52*, 71–99. [CrossRef]
46. Pérez, J.; Díaz, J.; Berrocal, J.; López-Viana, R.; González-Prieto, Á. Edge computing. *Computing* **2022**, *104*, 2711–2747. [CrossRef]
47. Yousefpour, A.; Fung, C.; Nguyen, T.; Kadiyala, K.; Jalali, F.; Niakanlahiji, A.; Kong, J.; Jue, J.P. All one needs to know about fog computing and related edge computing paradigms: A complete survey. *J. Syst. Archit.* **2019**, *98*, 289–330. [CrossRef]
48. Kubiak, K.; Dec, G.; Stadnicka, D. Possible Applications of Edge Computing in the Manufacturing Industry—Systematic Literature Review. *Sensors* **2022**, *22*, 2445. [CrossRef] [PubMed]
49. Papageorgiou, E.I.; Theodosiou, T.; Margetis, G.; Dimitriou, N.; Charalampous, P.; Tzovaras, D.; Samakovlis, I. Short Survey of Artificial Intelligent Technologies for Defect Detection in Manufacturing. In Proceedings of the 2021 12th International Conference on Information, Intelligence, Systems & Applications (IISA), Chania Crete, Greece, 12–14 July 2021; pp. 1–7. [CrossRef]
50. Singh, H. *Implementing Cisco Networking Solutions: Configure, Implement, and Manage Complex Network Designs*; Packt Publishing Ltd.: Birmingham, UK, 2017.
51. Tuli, S.; Mahmud, R.; Tuli, S.; Buyya, R. FogBus: A Blockchain-based Lightweight Framework for Edge and Fog Computing. *J. Syst. Softw.* **2019**, *154*, 22–36. [CrossRef]
52. Mahmud, R.; Buyya, R. Modeling and Simulation of Fog and Edge Computing Environments Using iFogSim Toolkit. In *Fog and Edge Computing: Principles and Paradigms*; Wiley: New York, NY, USA, 2019; pp. 433–465. [CrossRef]
53. Mahmud, R.; Pallewatta, S.; Goudarzi, M.; Buyya, R. iFogSim2: An extended iFogSim simulator for mobility, clustering, and microservice management in edge and fog computing environments. *J. Syst. Softw.* **2022**, *190*, 111351. [CrossRef]