C3F: Cross-Cloud Communication Framework for Resource Sharing amongst Cloud Networks: An Extended Study

Ahmad Waqas†, Abdul Rehman Gilal†, M. Abdul Rehman†, Qamar Uddin†, Nadeem Mahmood††
Zulkefli Muhammed Yusof ††

† Department of Computer Science, Sukkur IBA University, Sukkur, Pakistan
†† Department of Computer Science, International Islamic University Malaysia

Abstract:
This paper presents a resource sharing model to share the computing resources amongst cloud networks. The resource sharing model is developed for interconnected clouds and is tested using the cross-cloud communication framework (C3F). In many situations, clouds are overloaded or underutilized due to the varying demands of computing resources by running applications and users. The aim of the proposed resource sharing model is to manage cloud resources by sharing underutilized resources. For that, C3F is exploited for borrowing and lending the resources with mutual agreement between interconnected clouds. To illustrate, CSM and ICM are programmed to communicate with each other for requesting and allocating resources. The processes of resource lending and borrowing are explained. The paper also demonstrates an algorithm for resources sharing along with running time complexity computations.

Key words:
Cloud computing; Resource Sharing; Resource Management; Cross-cloud Communication; ReSA

1. Introduction

The fundamental object of cloud computing is to deliver computing resources over the Internet with cost-effective solutions [1]–[4]. For this purpose, cloud offers variety of resources to its clients and theoretically, cloud has unlimited resources to fulfil the user demands [5],[6]–[10] but, practically there is a limit of resources that a cloud can offer at a given time [11], [12]. Situation may arise that the requested resource may not be available as the cloud provider cannot satisfy all the requests.

This paper discusses the issue of resource unavailability and sharing of resources to cater the critical situations. It is an extension to our previous study published as "ReSA: Architecture for Resources Sharing Between Clouds" [12], [13] and also use the concepts in [14]. In order to achieve the study objective, the cross-cloud communication framework (C3F) is exploited for borrowing and lending the resources with mutual agreement between interconnected clouds. The key actor of C3F is referred as Inter-cloud Communication Manager (ICM) as illustrated in Figure 1. The ICM is a bridge between clouds to form the cloud network and facilitate the inter-cloud communication. They core responsibilities of ICM are:

- Coordination with CSM for log maintenance.
- Maintaining the Key Table that contains the entries of other connected clouds.
- Monitoring and sensing the overall performance of cloud services.
- Sending messages to all listed clouds in Key Table for resource borrowing.
- Receiving and forwarding the resource requests from cloud network.

1.1. Resources Offered by Cloud

Cloud environment offers variety of resources to its clients that include computational resources, software resources, low-level hardware, and storage resources and communication resources [15]. Cloud offers its resources as services that are Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS) and Infrastructure-as-a-Service (IaaS) that is called SPI services model. The SaaS model offers the clients with usage of on-demand software that may include the business, education and personal applications. There is no need for managing infrastructure and platform by the cloud client on which the application is running, thus it simplifies the support and maintenance. Google Apps [16], [17], Microsoft Office 365 [18], and OnLive are few examples of SaaS. In PaaS, the client is offered with a runtime environment for designing, deploying and testing applications. The Cloud Service Providers (CSPs) typically facilitate the cloud customers with a computing platform that usually include system software and programing run-time environment. Windows Azure Compute, Amazon Elastic Beanstalk, EngineYard, Cloud Foundry, Force.com, Mendix, Google App Engine, Heroku and OrangeScape are a few examples of PaaS. The basic level of cloud service model is IaaS, where consumers are provided with the virtualized computer components and resources to build and run their applications without purchasing the actual expensive

Manuscript received August 5, 2017
Manuscript revised August 20, 2017

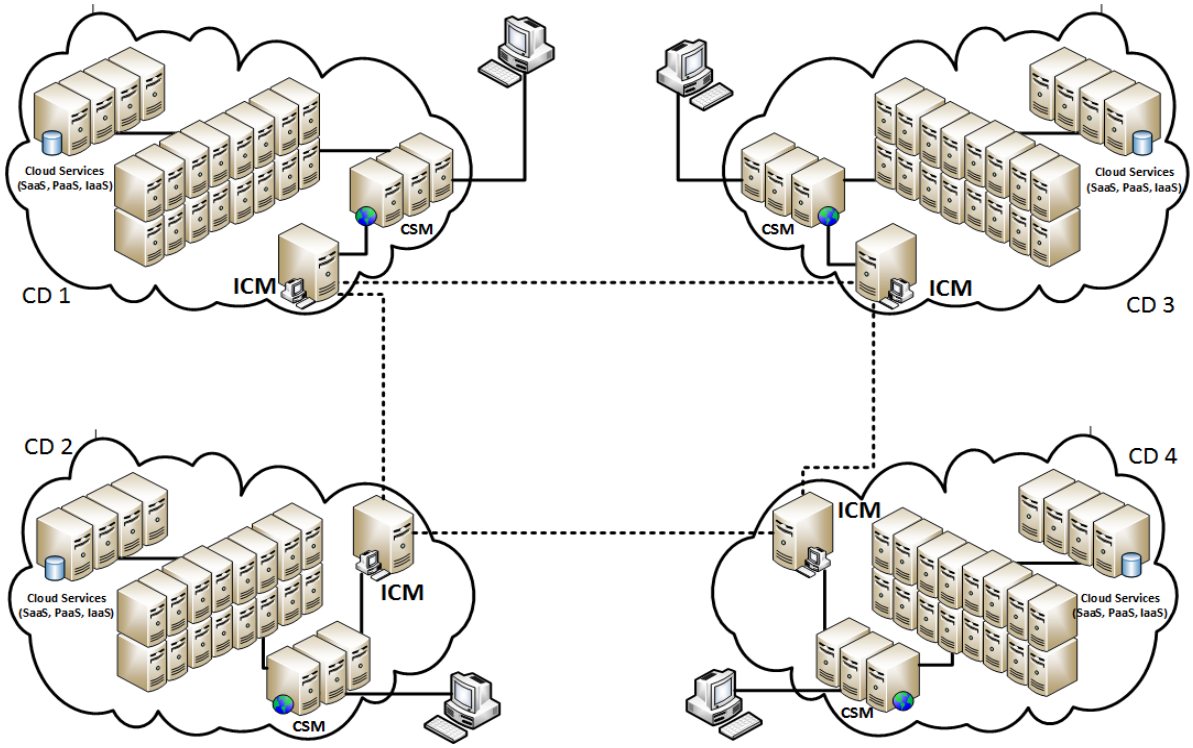

Fig. 1 Interconnected Clouds and Cross-Cloud Communication Framework

computing components. Windows Azure, Virtual Machines, Amazon CloudFormation and underlying services such as Amazon EC2, Google Compute Engine and Rackspace Cloud are examples of IaaS.

2. Resource Sharing

The prime objective of cloud is to deliver on-demand resources over Internet with minimal efforts and greater scalability and transparency [10], [19]–[22]. To achieve this, cloud receives the resource requests from its clients and deliver the requested resources accordingly. Referring to the scenario of a network of four clouds illustrated in Figure 1, at the cloud service provider's end, Cloud Service Manger (CSM) is the only gateway for clients to connect with cloud for resources utilization. In order to request resources, a client needs to login with credentials for proving its identity. The CSM receives the request and grant access to clients after performing authentication with the help of authentication mechanism and authorization based on service level agreement (SLA). Once the identity of client is satisfied, CSM will check the availability of requested resources and if it is available, CSM will delegate resources to the clients. If resources are unavailable at that point in time, it may borrow the requested resource from other clouds in its network to fulfil requirement of client. For that, CSM will request to its Inter-cloud Communication Manager (ICM) for borrowing the resource as ICM is the only gateway to connect with other clouds in cloud network. Now that ICM has received request from its CSM, it will prepare a borrow message and broadcast to cloud network using Key Table as it provides the full-view of cloud network. Every ICM in cloud network will receive the borrow message from requestor ICM, and will forward it to their CSMs because resources are handled and managed by CSM. The CSM will check the requested resource availability and will respond to its ICM with available or unavailable message in either case. Consequently, if the ICM gets the available message from its CSM, it will forward it to requestor ICM otherwise it will discard the message. The requestor ICM will receives "available" message in response from one or more respondent ICMs and reply back with "accept" message only to the first respondent ICM using first come first serve (FCFS) mechanism. Upon receiving "accept" message from requestor ICM, the respondent ICM will forward it to its CSM who will allocate the requested resource to the requestor CSM that will be delivered to the client in a transparent manner. Figure 2 portrays the conceptual layered model for intra-cloud and inter-cloud communication to borrow and lend resources between clouds.

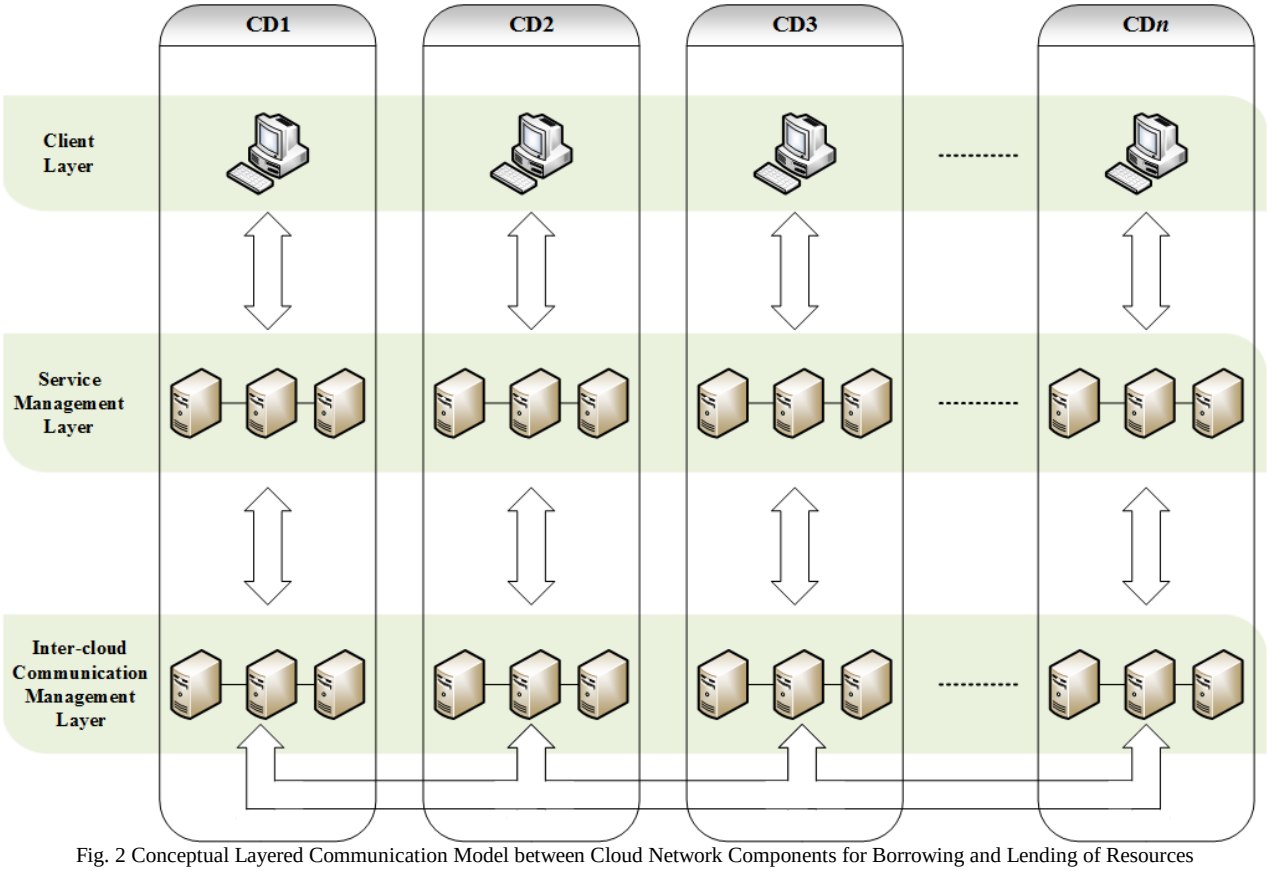

Fig. 2 Conceptual Layered Communication Model between Cloud Network Components for Borrowing and Lending of Resources

2.1. The Illustration of Resource Borrowing and Lending

As illustrated in Figure 3, a client of Cloud 1 (CD1) requested some PaaS resource P1 keeping in view the scenario of cloud network with four cloud members as described in Figure 1 and Figure 2.

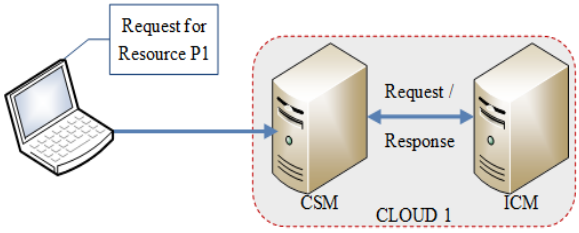

Fig. 3 Client Request for Resource P1

CSM will evaluate the availability of P1 and, if available, it will allocate P1 to client based on SLA. Suppose, P1 is not available at that point in time, CSM will request to ICM for borrowing P1 from cloud network as shown in Figure 4.

First, ICM of CD1 will prepare a "request message" and then, identify the clouds from whom it can request P1 based on SLAs. It will broadcast request to intended cloud network members as illustrated in Figure 5.

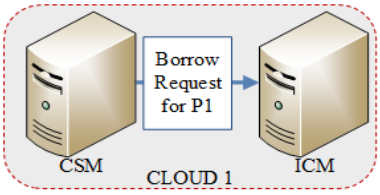

Fig. 4 CSM Borrow Request to ICM

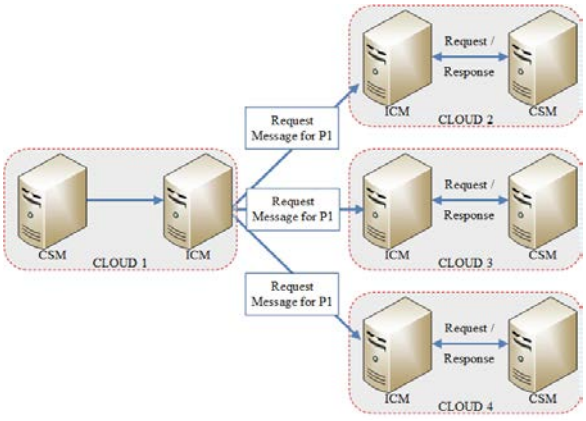

Fig. 5 ICM Broadcast Request Message to Cloud Network

The ICMs of CD2, CD3 and CD4 will receive the request and forward to their respective CSM as shown in Figure 6. Afterward, each CSM will check the availability of P1 and reply to its ICM with "available" or "unavailable" message,

respectively, whether the resource is available or not. Assume, ICM of CD2 and CD4 received "available" message and ICM of CD3 received "unavailable" message from their respective CSMs as illustrated in Figure 7. Subsequently, ICMs of CD2 and CD4 will forward "available" message to the requestor ICM of CD1 while ICM of CD3 will simply discard the message as depicted in Figure 8.

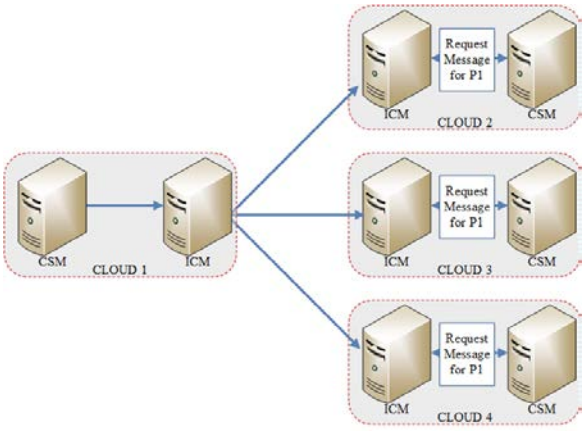

Fig. 6 ICMs Forward Request Message to their CSM

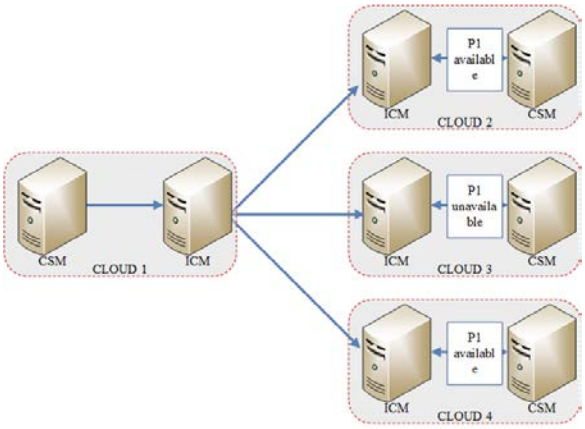

Fig. 7 CSM Reply after Resource Availability Check

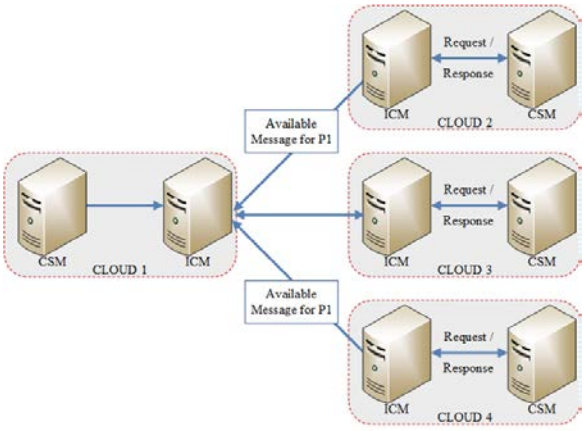

Fig. 8 ICM Response to Requestor Cloud

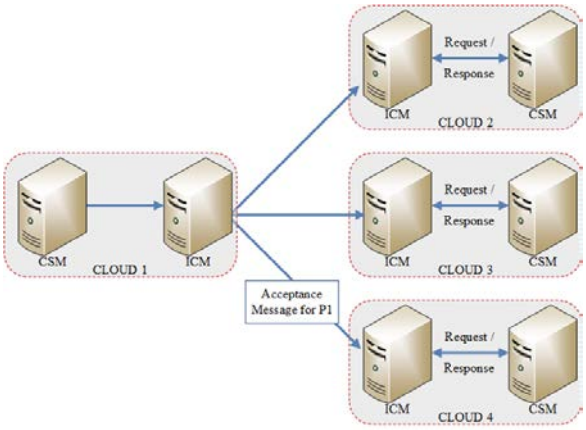

Fig. 9 Acceptance Message based on FCFS

The ICM of CD1 may receive the "available" message from a number of cloud network members or none. After a time limit, if it receives no message it will send "unavailable" message to its CSM. And if ICM receives more than one message, it will reply with "acceptance" message from the list of respondents using FCFS mechanism and discard rest of the messages. For example, it received the first message from CD4, the ICM of CD1 will reply with "acceptance" message to CD4 and will discard the message of CD2 as shown in Figure 9.

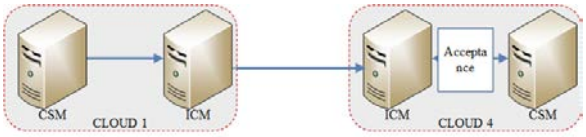

Fig. 10 ICM Forwarded Acceptance Message to CSM

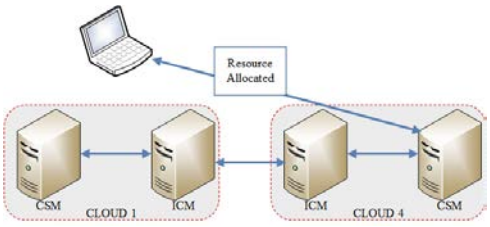

Fig. 11 Resource Allocation from CD4

Once the ICM of CD4 receives "acceptance" message from ICM of CD1, it will be forwarded to its CSM as shown in Figure 10. Then, CSM of CD4 will allocate the resource P1 to CSM of CD1 which will be delivered to client of CD1 as illustrated in Figure 11. The CD1 seamlessly fulfilled the request of its client by borrowing the resource P1 from its network member.

2.2. The Processes of Resource Sharing

Resource sharing between clouds involves two major processes termed as "Borrowing" and "Lending" along with many sub-processes. It includes CSM and ICM to accomplish these processes. Both CSM and ICM are

responsible to receive resource requests and respond accordingly. To achieve this, all cloud actors communicate with each other that are classified in three tiers for simplification and management. These tiers of communication are: 1) client to CSM, 2) CSM to ICM and 3) ICM to other ICMs in cloud network. Table 1 lists the major processes performed by respective actors that are involved in resource sharing between clouds.

Table 1: Actors and their Process for Resource Sharing

Actor	***Process***
CSM	• Handle resource requests and respond accordingly • Requests for resources
ICM	• Handle resource (lending) requests • Request (borrow) resource

2.2.1. Managing Resource Request Received at CSM

When the CSM receives a request for some resource, it records the entry and updates log file in database as described in Figure 12. CSM may receive resource request either from a client or it's ICM. If the request is from client, it means that its own client want to utilize the resource and if the request is from ICM, it means some other cloud wants to borrow resource. The CSM then evaluates the cloud resources and check the availability of requested resource. If the resource is available and requested by client, it will allocate requested resource to client and update the log file, client's status and resource status in database. Further, it will start a sub-process of metering by using SLA information to record the utilization of resource for billing and payment invoice. In case that client has requested the resource and resource is not available, then it will send resource request message to its ICM to borrow it from cloud network members and update the log entries. The CSM will be waiting to receive the response from ICM and put the client on hold for delivery of requested resource.

In case CSM received the resource request from its ICM and the resource is not available, it will reply back with "Unavailable" message and terminate process after recording log entry. On the contrary, if it received request from ICM and requested resource is available, it will send "available" message to its ICM and initiate a sub-process

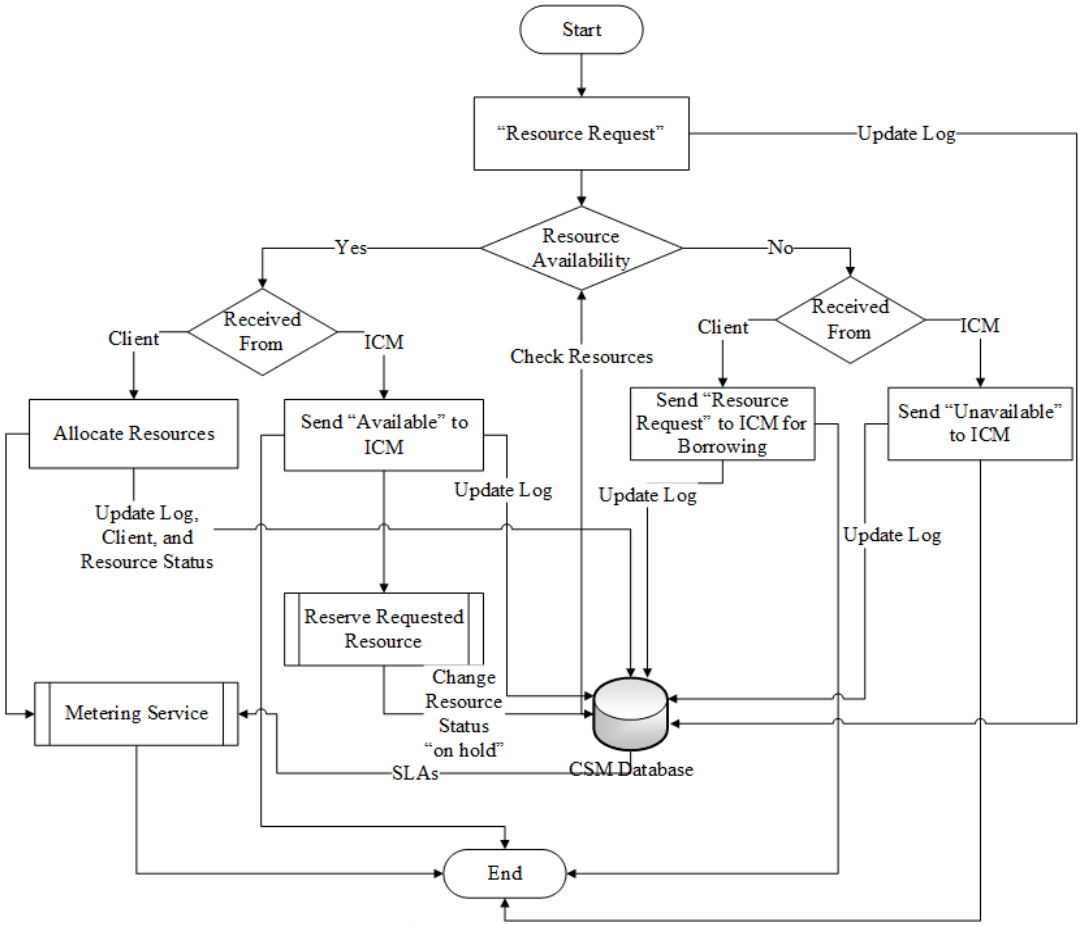

Fig. 12 Flowchart for Managing Resource Request Received at CSM

to put the resource on hold. The log file will be updated with new entry regarding communication between CSM and ICM. Figure 12 elaborates a detailed flow diagram to handle the resource request by CSM.

2.2.2. Managing Resource Request Received at ICM

The inter-cloud communication manager (ICM) receives resource request either from its CSM or the ICM of some other cloud in cloud network as shown in Figure 13. If it receives the request from CSM, it implies that it has to borrow resource from some other cloud. In this case, first, it will prepare a "Borrow Message" along with its credentials including IP and MAC addresses. This message also states which resource is required and for how long it will be utilized, nevertheless, it may be scaled according to need. It will short-list the clouds to request for borrowing resource by using SLA information. After that, it will broadcast the request message to short-listed clouds and update the log entries. The ICM contains the information of connected clouds and cloud network in the Key Table.
In case of receiving request from other ICM from cloud network, the ICM forward the request message to its CSM in order to check the resource availability. Further, it will update the log file entries and will wait for the CSM response.

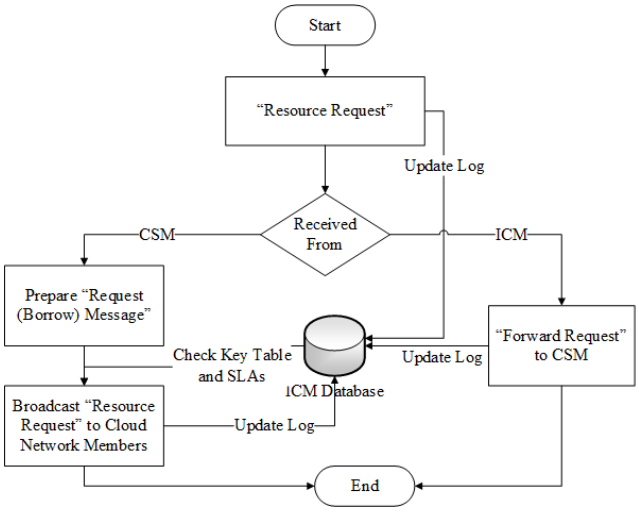

Fig. 2 Flowchart for Managing Resource Request at ICM

2.2.3. Managing Request Response Received at ICM

The ICM when request a resource as discussed in previous section, it receives response accordingly either from its CSM or from other ICMs in cloud network. It receives request response from its CSM when it has forwarded the resource request that was received from cloud network.

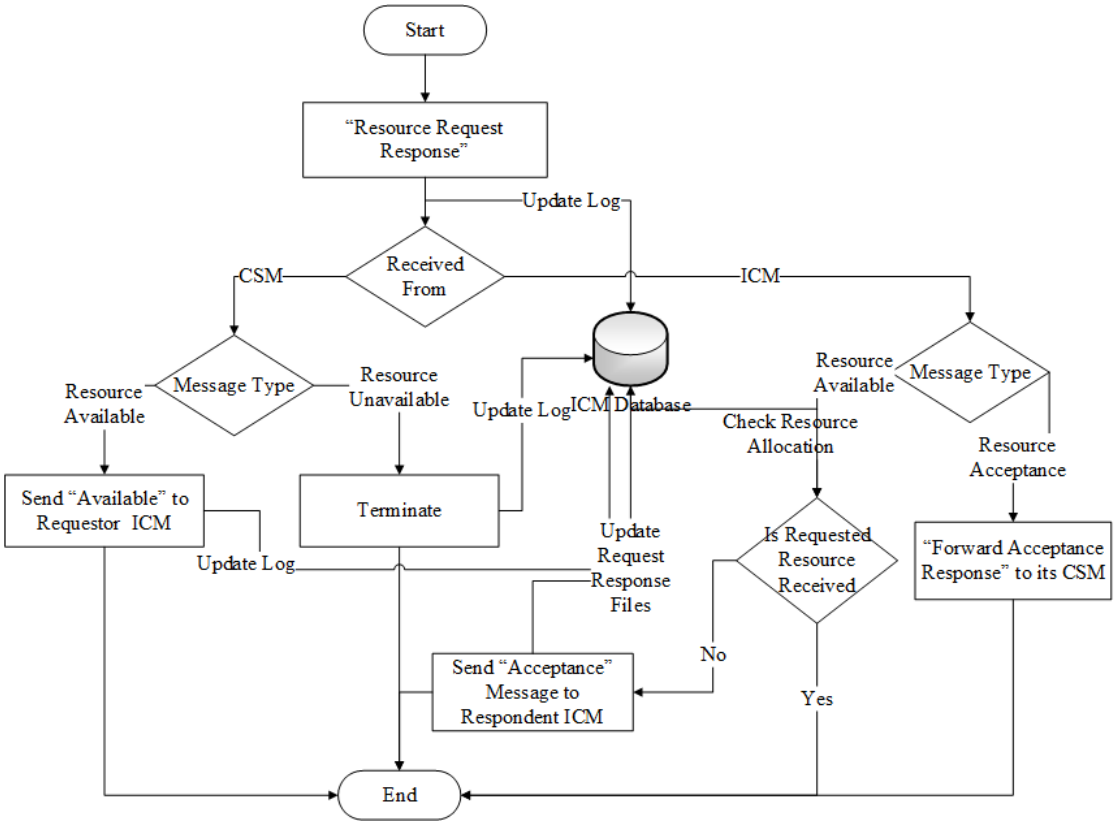

Fig. 14 Flowchart for Managing Request Response Received by ICM

And, it receives the response from ICMs of cloud network when it has requested to borrow the resource. ICM updates the log file as soon as it receives the request response.

In case response is received from CSM, the ICM will check the type of message whether it is “available” or “unavailable” message as shown in Figure 14. If the received response is “unavailable”, it simply terminates the process after updating the log entries. But if the response is “available”, it sends the “available” message to the requestor ICM to inform that the requested resource is available and can be shared. It then updates the log file and set a time window of T = 30 milliseconds to wait the acceptance of resource by requestor ICM. The time window is used to wait the acceptance from requestor ICM because the requestor ICM will only reply to one of the respondent ICMs. Further, it is required to release the resource whose status was set to “on hold” by CSM during response to requestor ICM.

On the other hand, if response is received from some ICM from cloud network it will check whether it is a result of request (available), or acceptance for lending the resource. It receives the “resource available” if it has broadcast borrow message earlier. The ICM will make sure whether requested resource is received already from some other cloud or not. If the resource is already received it will terminate the process otherwise it will reply with “acceptance message”.

ICM when receives the “acceptance message”, it means resource will be shared mutually among clouds and it is going to deliver the requested resource. In this case, it forwards the “acceptance message” to its CSM so that it can deliver the requested resource to the requestor cloud. CSM will allocate the resource to the requester cloud and the requester CSM will allocate the resource to its client. The log entries are updated at each step.

2.2.4. Request Response at CSM

The CSM receives the response of resource request in terms of borrow and lending as shown in Figure 15. If it is a borrow response, it means the CSM has requested the resource from cloud network and now it has received the response that resource is available and ready to be delivered. In this case, CSM receives the resource and deliver to its client along with starting the metering for charges. In addition, it updates log entries at each step and also update the clients and resources status in database.

Likewise, if CSM receives lending response, it means the other cloud is agreed to receive resource from it in response of “available” message. First, it checks the time window and if it is expired, it sends “deny” message to its ICM because the resource status was changed from “on hold” to “available” as the time limit exceeds. If the time window is still alive, it allocates the requested resource to requester CSM. It updates resource status from “on hold”

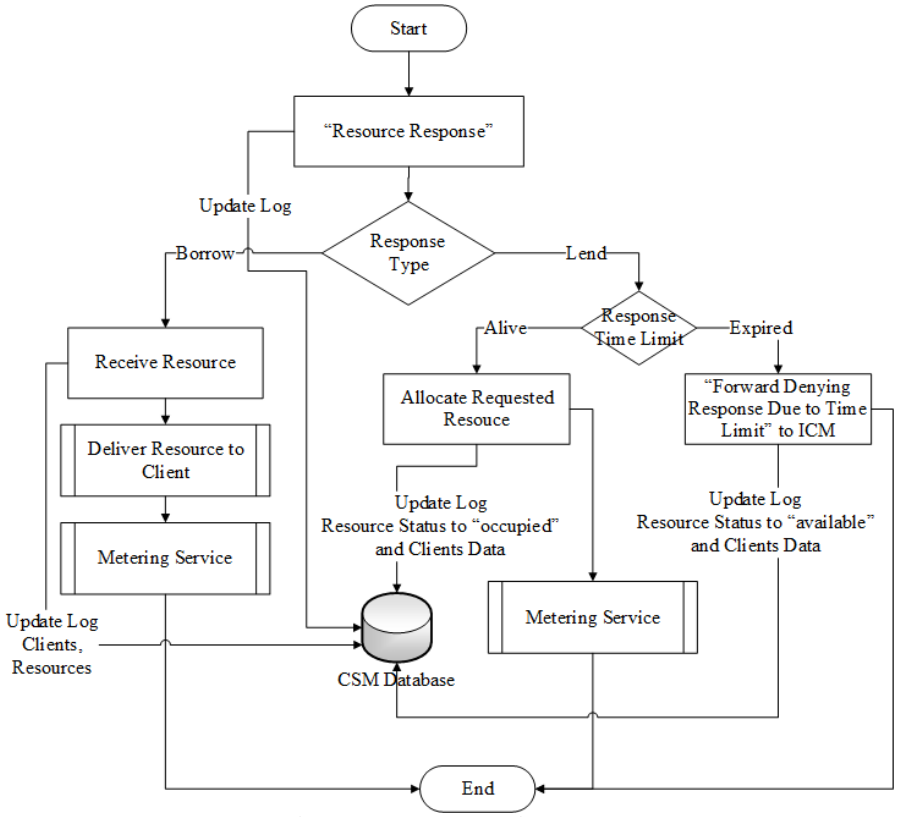

Fig. 15 Flowchart for Managing Response of Request Received at CSM

to "occupied" and updates the client's data. It also updates log entries at each step.

2.3. Algorithm for Resource Sharing between Clouds

Algorithm 1 is the step-by-step process of sharing the resources between clouds. As explained in the previous sections, client can request a resource from CSM only. For that client call the CSM process ResRequest(x,s) by indicating the resource x and its (sender's) credentials. The availability of resource is checked by the method availability(x) by CSM and if the value of y is 1, it means resource is available and will be allocated to client by invoking the method allocate(x,s). But if the value of y is 0 it means requested resource is not available. In this case, CSM will ask ICM to borrow the resource by calling the ICM function CMRequest(x). This method invokes the CMForReq(x) at each ICM in cloud network that are listed in Key Table until it receives y=1. As soon as availability of resource is ensured it will stop sending requests to other ICMs and exits the loop and if it does not receive 1 till the loop ends, it means the resource is not available in the whole cloud network. If the CMRequest(x) exits loop with y=1, it looks up the received ID (Identity of ICM who agreed to lend resource) and initiate the method accept(x,CloudID) and it will invoke the function allocate(x) and the resource will be allocated to requestor CSM.

Algorithm 1: Algorithm for Resource Sharing Between Clouds

```
ResRequest (x,s)
1:      do if y ← availability(x)
2:          do if s ∈ clients
3:              allocate(x,s)
4:      else
5:          ICMRequest(x)

ICMRequest(x)
1:      for ∀ICMs ∈ KeyTable
2:          y,ID ← ICMFwdReq(x)
3:          if y = 1
4:              exit loop
5:      lookup(ID) from KeyTable
6:          accept(x,CloudID)

ICMFwdReq(x)
1:      do if y ← availability(x)
2:          status (x) ← 2
3:          return y,ID

accept(x,CloudID)
1:      allocate(x,CloudID)

availability(x)
1:      do for ∀R = x
2:          if status(R) = 1
3:              y ← 1
4:              exit for loop
5:          else
6:              y ← 0
7:      return y

allocate(x,s)
1:      status(x) = 0
2:      meter(x,s)
```

0 = unavailable 1 = available 2 = on-hold

2.3.1. Analysis of the Resource Sharing Algorithm

Complexity analysis of an algorithm is a measure of the amount of time and space (memory) required by an algorithm for an input of a given size (n). In analysing an algorithm, rather than a piece of code, it is to be predicted that how many number of times "the principle activity" of that algorithm is performed. For the algorithm of resource sharing between clouds (Algorithm 1), the principal activity is to lookup the available resource either locally or from the cloud network and deliver it to client. Here are some assumptions for model machine to calculate the running time complexity of given algorithm.

- Multi-processor machines.
- 64-bit architecture.
- Sequential execution.
- c=1 unit constant time for arithmetical and logical operations.
- c=1 unit constant time for assignment, calling and return.
- c=1 unit constant time for computing conditional if with reading and writing from memory and/or disk.
- c=1 unit constant time for data transfer over high-speed local area network with CAT6 twisted pair cable.

In the best case, a requested resource is delivered to client from the local pool of resources. For that, the run-time complexity of function ResReq(x) is linear with $\Omega(n)$ where n represents the number of operations performed to deliver the requested resource. This is because the ResReq(x) calls another method availability(x) that contains a loop to check the status of each resource that is x.

As an average case, the requested resource is not locally available and borrowed from some other cloud in network.

For that, the run-time complexity of function ResReq(x) is quadratic with $\Theta(n^2)$ where n represents the number of operations performed to deliver the requested resource including borrow and lending processes. This is because the ResReq(x) calls another method ICMRequest(x) that contains a loop and it calls the ICMFwdReq(x) function that forwards the request to CSMs of cloud network. This function again calls another function availability(x) to check the status of each resource at other clouds. The worst case running time complexity of Algorithm 1 is then to lookup for the resource at each cloud in the network. It is computed using the same method that is also quadratic with $O(n^2)$.

3. Results and Discussion

The experiments were performed to test and evaluate the algorithm of resource sharing among cloud network (Algorithm 1). The experiments were performed using CloudWeb that is a Web-based Prototype for simulation of Cross-Cloud Communication [23]. The results showed that allocation of requested resources to clients from local cloud and by borrowing it from foreign clouds in network is possible with minimal allocation time. Moreover, it maintained the transparency of resource allocation with minimal management issues even though new management policies and SLA are required to be designed for cloud service providers for mutual interests and agreements. The transparency refers to the allocation of resources after borrowing from foreign connected clouds in a seamless manner.

3.1. Performance Evaluation Metrics

There are different performance matrices to evaluate the resource allocation in cloud environment. Most of them includes response time, availability, security and throughput [11], [24], [25] for both static and dynamic allocation techniques. Therefore, in this study, the cost for resource sharing is evaluated in terms of success rate and resource allocation time.

Success Rate: Most significantly, success rate refers to the successful allocation of requested resource when it is available either from local cloud or foreign cloud, but more specifically, after borrowing the resource from cloud network.

Allocation Time: Resource allocation time measures the time taken to deliver a resource after the request is received. The resource allocation time is computed in microseconds and the total performance cost T_c of requested resources allocation is computed by (1).

$$T_c = \sum_{i=1}^{n} (\tau \times \psi) + \sum_{j=1}^{m} (\tau \times \phi) \quad (1)$$

Where,

n = The number of requests with successful allocation

m = The number of requests with unsuccessful allocation

τ = The total time taken for processing request

ψ = The success rate

ϕ = The unsuccessful rate

3.2. Success Rate for Resource Allocation

Success rate refers to the rate of successful allocation for requested resource. For the experimentation scenario, the client requested the resource P1 from CD1 and CSM received the request. CSM then checked the availability of P1. The resource was available, therefore it was allocated to client from local cloud. Table 2 (Row 1) shows that P1 was requested from Cloud 1 (CD1) and the status was available, hence it was successfully allocated to client.

Table 2: Resource Allocation Success Rate for the Same Request

Requested Resource		CD1	CD2	CD3	CD4
P1	*Status*	Available	Available	Available	Available
	Allocated	Yes	-	-	-
P1	*Status*	Unavailable	Available	Available	Available
	Allocated	No	-	-	Yes
P1	*Status*	Unavailable	Available	Available	Unavailable
	Allocated	No	Yes	-	No
P1	*Status*	Unavailable	Unavailable	Available	Unavailable
	Allocated	No	No	Yes	No
P1	*Status*	Unavailable	Unavailable	Unavailable	Unavailable
	Allocated	No	No	No	No

In another experiment, the CSM of CD1 received request for resource P1 and checked the availability and it was unavailable at that point in time but was available on all other clouds as shown in Table 2 (Row 2). In this case, according to resource sharing algorithm, CSM requested ICM to borrow the resource from some other cloud in network. For that, ICM prepared the borrow request and broadcasted to all members of cloud network. On the other hand, all ICMs of cloud network received borrow request and checked with their respective CSMs whether the resource can be allocated to CD1 or not. This check was performed on the basis of SLA for agreed services and availability of resource. In this case, when requested resource was available at all other clouds and they have agreed to share resource P1, the ICMs replied CD1 with "available" message. The ICM of CD1 replied with

"accept" message to CD4 as it received "available" message from CD4 first as it uses FCFS mechanism. Hence, resource P1 was allocated and process was finished according to algorithm (Table 2, Row 2).
The experiment was repeated once again to request resource P1 when it was not available at CD1 and CD4. This time, ICM of CD1 replied with "accept" message to ICM of CD2 as it received "available" message first from it and the resource was allocated successfully (Table 2, Row 3).
Once again same experiment was repeated by requesting resource P1 from CD1 while only CD3 was having it available. Now, CD1 accepted the resource from CD3 as it only received "available" message from it. Therefore, the resource was successfully allocated once again (Table 2, Row 4).
The experiment was repeated fifth time while none of the cloud in network was having resource available (Table 2, Row 5). Likewise, all the respective processes for borrowing resource were repeated but CD1 did not get any response from any cloud so the process was terminated without allocation of resource. It is to be noted that each cloud checked the "agreed services" from SLA database before lending the resource, and if it was agreed to be shared then only it was allocated.

3.2.1. Test Case – I

In the above discussed experimentations, for all situations where the P1 was available at any of the cloud, it was successfully allocated to client. This gave the 100% success rate for borrowing and allocating resource to client when it was not available at local cloud. CSM of CD1 only refused the client when requested resource was not available at any of the cloud in cloud network.
To test the system when a cloud has no resource available for allocation, multiple requests were generated to allocate same resources. For instance, cloud 1 was not having any resource available and client requested for the resources. Whereas, cloud 2, and cloud 3 were having resources available and they agreed to share resources. Cloud 1 borrowed the "agreed" resources from cloud 2 and cloud 3 and allocated to its client successfully.

3.2.2. Test Case – II

In order to test the framework rigorously, 500 different requests for different resources were generated using a single client system that was connected to CD1. These requests were containing the single and multiple resources requisition. It was observed that 100% requests were entertained by CSM although 5.6% requests were not fulfilled because of the requested resource unavailability at all clouds in the cloud network. However, 94.4% requests were successfully completed and resources were allocated.

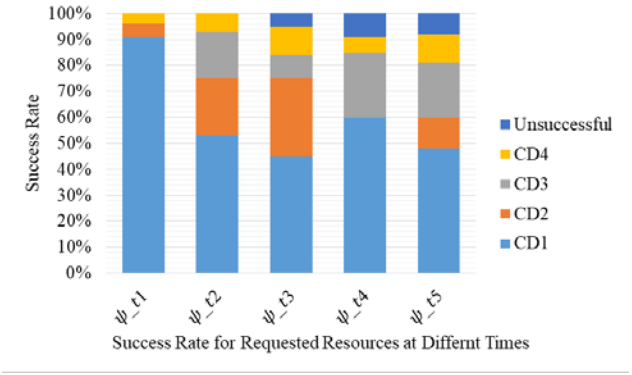

Fig. 17 Success Rate for Various Resource Requests

Furthermore, different requests for resources were generated at different times. Figure 17 shows that at time t_1, 100% requests were fulfilled and 91% resources were allocated from local cloud CD1, 5% resources were allocated from CD2 and 4% resources were allocated from CD4. It is because CD1 borrowed resources from CD2 and CD4 after fulfilling 91% of requests as the requested resources were no more available at CD1. Similarly, the resources were accepted from CD2 and CD4 as the "available" messages were received from these clouds. In the same vein, at time 〖 t〗 _2, 53% of resources were allocated from local cloud CD1 and rest of resources were borrowed and successfully allocated from CD2, CD3 and CD4. On the contrary, at time 〖 t〗 _3, t_4 and t_5, 100% requests were not fulfilled even though CD1 borrowed resources from all other four clouds and 5%, 9% and, 8% requests were unsuccessful respectively. It was because of the flood of requests and limit of resources while the resources were allocated to other clients at all systems.
Experimental results showed that cross-cloud communication benefits for borrowing and lending the resources and fulfilling the client's requests. As the case discussed above, at time t_3 CD1 was having less than 50% requested resources even though it managed to allocate overall 95% resources after borrowing from other clouds in network.

3.3. Resource Allocation Time

The mean time taken to allocate the resource P1 is 12 microseconds when it was available at local cloud CD1. This time includes the authentication process, checking the availability of resource and updating the status of resource, client and writing logs and was computed using the UNIX timestamps. The exhaustive requests for several resources were generated in order to test the allocation of requested resource from local cloud CD1. The requests also included

the requisition of same resource for multiple usages and the distinct resources too. Table 3 shows a sample of resources allocation times both from local cloud and connected foreign clouds. The mean resource delivery time is less than 12 microseconds when the resource is allocated from local cloud and the maximum difference for allocating different resources is 5 microseconds. This difference may have been caused by the state of system at the time when it received client request because the system may be busy to perform other tasks. Client can also request multiple resources at the same time within a single request, so the multiple resources was requested to test the performance. For instance, resources P1, S1 and I1 were requested simultaneously in a single request message. It was observed that same allocation time was taken for the said requested resources when they were allocated from the local cloud. It is because the system state did not change while handling the request and allocating the resource from local pool.

The CSM tried to allocate resource from its local cloud first that take fewer mean allocation time (less than 12 microseconds) as discussed above. But, acquiring resource from other clouds consumed more time because it involved communication between clouds. This communication speed depends on the channel and network characteristics. As, prototype of the study used LAN based connections with layer 2 switch, that is why abundant difference in allocation time from other clouds was not noticed. Table 3 lists the experimental results related to allocation time after the request has received from client.

It is confirmed that if the resource is unavailable at local cloud, it borrows it from other clouds in network that sometimes take much higher time than allocating from local cloud. So, it is concluded that it takes less time to allocate resource from local cloud and the time increases as the request is forwarded to other clouds for allocation. Furthermore, requesting multiple resources in same request also gives almost same results. For instance, client requested P1, S1 and I1 where P1 allocated from local cloud, S1 allocated after acquiring from CD2 and I1 allocated after acquiring from CD4. The mean allocation time is almost same for requesting single resource or multiple resources.

Table 3: Resource Allocation Time (microseconds)

Resource Request	***Local Cloud***	***Resource Borrowed From***		
		CD2	***CD3***	***CD4***
P1	15	20	25	20
P2	10	30	20	25
P3	15	30	25	20
S1	15	20	25	20
S2	10	25	20	30
S3	10	20	30	25
I1	10	25	20	20
I2	15	20	30	25
I3	10	25	20	30

In another experiment, ten resources were requested from CD1 and Figure 18 shows the trend of time taken to allocate these resources by four different clouds. The resource allocation time ranges between 10 and 15 microseconds when the resources were allocated from CD1. Whereas, the resource allocation time ranges between 20 and 30 microseconds when CD1 borrowed them from any of the other clouds. It is observed that CD1 took fewer time to allocate resources while other three clouds consumed more time for allocation of the same resources. It is because, these resources were borrowed from other clouds and the trend was observed that other clouds took more time than CD1 with some variations. Figure 19 shows that mean allocation time for allocation of resources after borrowing form other clouds is higher than local cloud. Furthermore, it also portrays that mean allocation time for borrowing and allocating resource from all other clouds ranges between 20 and 30 microseconds.

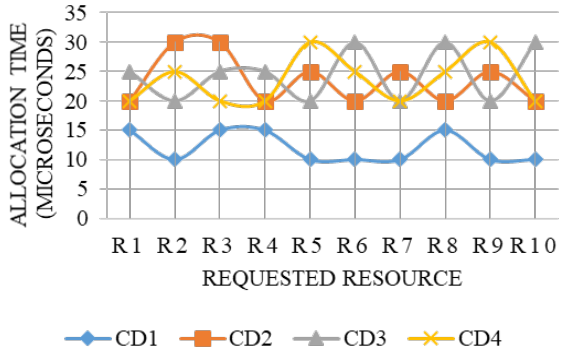

Fig. 18 Resource Allocation Time Trend for Various Requests

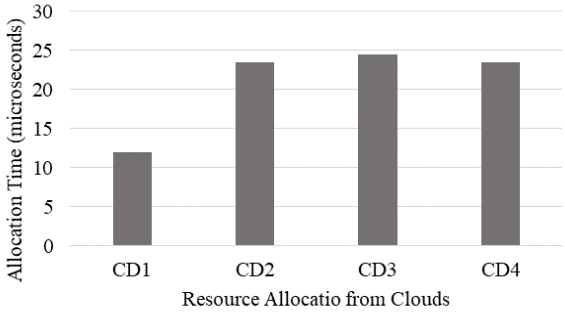

Fig. 19 Mean Resource Allocation Time for Requests

4. Conclusion

This paper discussed the process of resource sharing between interconnected clouds. The process is explained with the help of case study with technical and technological details. To illustrate, the resource sharing process involves three layers (i) client to CSM, (ii) CSM to ICM and (iii) ICM to CSM and these architectural components have their significant activities at each layer. The major activities include requesting resources, evaluating resources and responding to requests accordingly. The details of activities for CSM and ICM are discussed in terms of resource request and request response. In addition, the algorithm for resource borrowing and lending is discussed and explained with dry-run to validate the functionality. The algorithm is analysed in terms of asymptotic running time complexity for best, average and worst cases. It is observed that the algorithm behaviour is linear in best case and denoted by $\Omega(n)$. Whereas, the asymptotic running time behaviour of algorithm in average and worst cases is quadratic and denoted with $\Theta(n^2)$ and $O(n^2)$ respectively. It is concluded with experiments that the resource sharing among interconnected clouds for borrowing and lending resources can be performed by utilizing cross-cloud communication framework. The quadratic running time complexity suggests that the algorithm is reasonably acceptable.

The results showed the successful communication between clouds for borrowing and lending resources. The resource sharing is evaluated in two dimensions: success rate and allocation time. It was tested and concluded that 94.4% of the time, client's request was successfully fulfilled by borrowing the resource from other clouds. The mean allocation time was calculated 12 microseconds when resource is allocated from local cloud. The increase in allocation time is noticed when it is borrowed from other connected clouds. Whereas, the mean allocation time for borrowing and allocating resource from all connected clouds showed minor difference.

References

[1] L. Badger, T. Grance, R. Patt Corner, and J. Voas, "Cloud Computing Synopsis and Recommendations," 2011.

[2] F. Liu et al., "NIST Cloud Computing Reference Architecture: Recommendations of the National Institute of Standards and Technology," in NIST Special Publication 500-292, Cloud Computing Program Information Technology Laboratory National Institute of Standards and Technology Gaithersburg, MD 20899-8930, 2011, pp. 1–35.

[3] P. Mell and T. Grance, "The NIST Definition of Cloud Computing (Draft) Recommendations of the National Institute of Standards and Technology," in NIST Special Publication 800-145 (Draft), Computer Security Division, Information Technology Laboratory (ITL), National Institute of Standards and Technology (NIST), U.S. Department of Commerce, Gaithersburg, MD, USA., 2011.

[4] A. Waqas, Z. Muhammed Yusof, and A. Shah, "Fault Tolerant Cloud Auditing," in 4th International Conference on Information and Communication Technology for Muslim World (ICT4M), 2013, pp. 1–5.

[5] M. Armbrust et al., "Above the Clouds : A Berkeley View of Cloud Computing Cloud Computing," 2009.

[6] C. L. Li and L. Y. Li, "Optimal resource provisioning for cloud computing environment," J. Supercomput., vol. 62, pp. 989–1022, 2012.

[7] L. Wu, S. K. Garg, and R. Buyya, "SLA-Based Resource Allocation for Software as a Service Provider (SaaS) in Cloud Computing Environments," in 2011 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2011, pp. 195–204.

[8] A. García-García, I. Blanquer Espert, and V. Hernández García, "SLA-driven dynamic cloud resource management," Futur. Gener. Comput. Syst., vol. 31, no. 1, pp. 1–11, 2014.

[9] T. B. Winans and J. S. Brown, "Cloud computing: A collection of working papers," Deloitte LLC, 2009.

[10] B. Furht, "Cloud Computing Fundamentals," in Handbook of Cloud Computing, USA: Springer US, 2010, pp. 3–19.

[11] A. Nathani, S. Chaudhary, and G. Somani, "Policy based resource allocation in IaaS cloud," Futur. Gener. Comput. Syst., vol. 28, no. 1, pp. 94–103, 2012.

[12] A. Waqas, Z. M. Yusof, A. Shah, and M. A. Khan, "ReSA : Architecture for Resources Sharing Between Clouds," in Conference on Information Assurance and Cyber Security (CIACS2014), 2014, pp. 23–28.

[13] A. Waqas, Z. M. Yusof, A. Shah, Z. Bhatti, and N. Mahmood, "Simulation of Resource Sharing Architecture between Clouds (ReSA) using Java Programming," in 2014 International Conference on Information and Communication Technology for Muslim World (ICT4M), 2014, pp. 5–10.

[14] A. Waqas, Z. M. Yusof, A. Shah, and N. Mahmood, "Sharing of Attacks Information across Clouds for Improving Security : A Conceptual Framework," in IEEE 2014 International Conference on Computer, Communication, and Control Technology (I4CT 2014), 2014, pp. 255–260.

[15] A. Waqas, Z. M. Yusof, and A. Shah, "A security-based survey and classification of Cloud Architectures, State of Art and Future Directions," in 2nd International Conference on Advanced Computer Science Applications and Technologies – ACSAT2013, 2013, pp. 284–289.

[16] Google, "Google Apps for Work," Access Online, 2008. [Online]. Available: https://www.google.com/work/apps/business/. [Accessed: 18-Apr-2015].

[17] GoogleCloud, "Google Cloud Platform," 2008. [Online]. Available: https://cloud.google.com/. [Accessed: 30-Apr-2015].

[18] Microsoft, "Office when and where you need it," Access Online. [Online]. Available: https://products.office.com/en-us/business/explore-office-365-for-business. [Accessed: 18-Apr-2015].

[19] L. O. Brien, "Towards a Taxonomy of Performance Evaluation of Commercial Cloud Services," in IEEE Fifth

International Conference on Cloud Computing, 2012, pp. 344–351.
[20] W.-T. Tsai, X. Sun, and J. Balasooriya, "Service-Oriented Cloud Computing Architecture," in Seventh International Conference on Information Technology, 2010, pp. 684–689.
[21] G. S. Thakur, R. Gupta, and S. Mukharjee, "A Survey on Cloud Computing and its Services," Int. J. Adv. Res. Comput. Sci. Electron. Eng., vol. 1, no. 5, pp. 121–124, 2012.
[22] M. Armbrust et al., "A view of cloud computing," Communications of the ACM, vol. 53, no. 4. p. 50, 2010.
[23] A. Waqas, "A CLOUD COMPUTING FRAMEWORK FOR SHARING OF CLOUD RESOURCES AND ATTACKS INFORMATION AMONGST CLOUD NETWORKS," International Islamic University Malaysia, 2016.
[24] B. B H and H. S. Guruprasad, "A survey on various resource allocation policies in cloud computing environment," COMPUSOFT, Int. J. Adv. Comput. Technol., vol. 3, no. 6, pp. 893–899, 2014.
[25] P. Endo et al., "Resource allocation for distributed cloud: concepts and research challenges," IEEE Netw., vol. 25, no. 4, pp. 42–46, 2011.

Ahmad Waqas received his PhD from the Department of Computer Science, International Islamic University Malaysia in 2016. He is working as Assistant Professor in the Department of Computer Science, Sukkur IBA University, Pakistan. He has been involved in teaching and research at graduate and post graduate level in the field of computer science for the last eleven years. His area of interest is Distributed Computing, Cloud Computing, Complex Networks, Network Security and Auditing, Computing architectures, theoretical computer science, Data structure and algorithms. He has published many research papers in renowned international journals and conference proceedings.

Abdul Rehman Gilal is a faculty member of Computer Science department at Sukkur IBA University, Pakistan. He has earned a PhD in Information Technology from Universiti Teknologi Petronas (UTP), Malaysia. He has been mainly researching in the field of software project management for finding the effective methods of composing software development teams. Based on his research publication track record, he has contributed in the areas of human factor in software development, complex networks, databases and data mining, programming and cloud computing.

Dr. Abdul Rehman is an Associate Professor of Computer Science at Sukkur IBA University. He received his Ph.D. in Computer Science from the University of Bayreuth Germany. Prior to joining the Sukkur IBA in 2011, he held an appointment as an Assistant Professor at the Department of Information Technology, Quad-e-Awam University of Engineering, Science and Technology, and prior to that as a Database Administrator at AOL Pakistan Ltd. Dr. Rehman's research interests focus on the broad topic of 'Management of Scientific Data' which essentially includes the data logistic (retrieval, movement, and storage), data conversion (diverse data formats), structural and semantic data interoperability (integration and transformation). Dr. Rehman chairs the Department of Computer Science and serves as an Associate Director ORIC at Sukkur IBA. He has published several research papers in world's renowned peer-reviewed journals as well as presented his research work at various well-known international/national conferences.

Qamar Uddin Khand received the PhD degree in Production and Information Systems Engineering from Muroran Institute of Technology, Hokkaido, Japan in 2005. He is currently an Associate Professor in Department of Computer Science, Sukkur IBA University, Sukkur, Pakistan. His current research interests include Embedded Control Systems, Sketch-Based Interfaces and Modeling, Soft Computing, Computer Graphics and Software Engineering.

Dr. Nadeem Mahmood is affiliated with the Department of Computer Science, University of Karachi as an Associate Professor. He has been involved in teaching and research at graduate and post graduate level for the last eighteen years. He completed Post-Doctoral Research from KICT, International Islamic University Malaysia in 2014. He has obtained his MCS and Ph.D. in Computer Science from University of Karachi in 1996 and 2010 respectively. His area of interest is advanced database systems, health information systems, artificial intelligence and assistive technology. He is working as member national standard committee for IT/ICT, PSQCA, Ministry of Science and Technology. He has published more than 30 research papers in international journals and conference proceedings (ISI, IEEE and ACM). He presented papers and invited talks in various international conferences and seminars. He is working as technical committees' member and program committees' member for many international journals and conferences.