Manager Architecture for Mobile Cloud Computing-Cloudlet based

Miada Al-masre[1], Fathy Alburi[2]

[1] Faculty of Computing and Information Technology, CS Department
King Abdulaziz University, Jeddah, Saudi Arabia

[2] Faculty of Computing and Information Technology, CS Department
King Abdulaziz University, Jeddah, Saudi Arabia

ABSTRACT— ***In this paper, multi agents will be addressed as tools for managing Cloudlet, Cloudlets with rich resources are a strategy in Mobile Cloud Computing (MCC) that reduce one hop latency and the bandwidth demand of mobile users. However, cloudlets require a mechanism to manage an intensive application that needs augmented cognitive ability of users. They also need to be handled by a system capable of providing low latency results to the user. Ultimately, the importance of mobile configuration of the cloud management system is mainly to control the mobility and changes in proper ways that guarantee that the systems are not going to fail and increase the sense of reality.***

Configuration mobile management is based on agent technology that has developed during many years, but because a mobile cloud is still in its first decade with little research, it had recently merged with agent technology to manage a cloudlet. So our prototype may help to control some portable applications that need intensive processes with fewer latencies such as image processing, speech recognition, video streaming and augmented reality.

Keywords— Mobile cloud computing, cloudlet, agent, cloudlet manger, mobile user.

1. INTRODUCTION

As reported by World Mobile Applications Market the global mobile application market is expected to be worth $24.4 billion in 2015, and MCC has been introduced to be a potential technology for mobile services. This has led to the need of integration with the cloud computing into the mobile environment [1]. It has also overcome obstacles related to performance like battery life, storage, and network bandwidth [2].

MCC can provide services for customers through taking advantages of both mobile service and cloud computing. It is widely acknowledged as a concept that can greatly improve the user experience when accessing mobile services. The restrictions of mobile devices with respect to storage and computing capabilities and providing a new level of security by a centralized maintenance of security-critical software when removed is expected to find broad acceptances on the business as well as the consumer side [2].

Moreover, Multi-agent systems have attracted the researchers' interest to great extents, over the previous year's many researchers have been working on the development of mobile agents systems and their application to the areas of telecommunications and network management [3].

Firstly, this paper shows background and literature review about mobile cloud computing, and that agent technology will be present. Secondly, we introduce some related work. Thirdly, our architecture and algorithm will be proposed. Finally, we will evaluate and compare our approach with similar approaches.

2. BACKGROUND

Both mobile cloud computing (Cloudlet) and agent technologies are strong technologies that use to build and develop management architecture, so we need to review some concepts about them in addition to the concept of fragmentation a process to subtasks with a logical manner. Here we present an overview about all these concepts.

Services for customers can be provided by MCC by taking advantage of cloud computing and mobile services simultaneously. This concept is vastly acknowledged as a method that can greatly improve the user experience when portable services are being accessed. As for restrictions on computing capabilities, storage and providing a new security

level by a centralized maintenance of security-critical software when removed, will be popularly accepted on the business and consumer level [4].

MCC can be defined as a combination of mobile web and Cloud Computing, which is the most popular tool for mobile users to access applications and services on the Internet. It provides users with the data processing and storage services in clouds. However, cloud not sufficient for mobile user or mobile applications that sensitive for real time, because of cloud size and distance many researchers proposed a cloudlet concept (device consists of several resource-rich that connected to a remote cloud server) [5]. The following Figure1 is a table show some reasons that cloudlet is more suitable for MCC [6]:

	Cloudlet	**Cloud**
State	Only soft state	Hard and soft state
Management	Self-managed; little to no professional attention	Professionally administered, 24x7 operator
Environment	"Datacenter in a box" at business premises	Machine room with power conditioning and cooling
Ownership	Decentralized ownership by local business	Centralized ownership by Amazon, Yahoo!, etc.
Network	LAN latency/bandwidth	Internet latency/bandwidth
Sharing	Few users at a time	100s-1000s of users at a time

Figure 1: Why is cloudlet more suitable?

Finally, in literature review concepts the basic agent model defines is: "A computational entity, which acts on behalf of others, is autonomous, pro-active and reactive, and exhibits the capability to learn, cooperate and move" [7]. Agent has many forms such as station, mobile, intelligent, and multi-agent, etc. here we focused on a multi agent system that has mobile and stable agents.

Mobile Agent (MA) is a small program containing both code and data that move around and executing at different nodes in the network. Search engine is a good example about MA application. Figure 2 shows a sample of MA life cycle architecture (Aglet perspective) when the agent move from one environment to another with full responsibility to adaptive and achieves his task with a dynamical behavior, such as consensus, convergence, consensus decision-making, and so on [8].

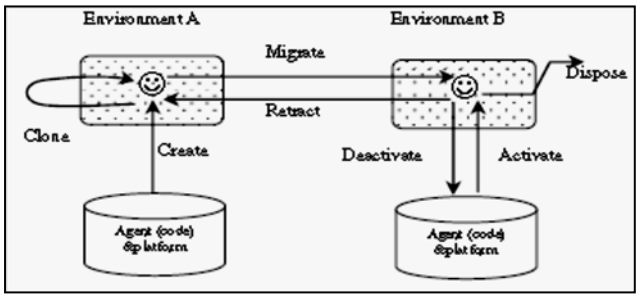

Figure 2: Mobile Agent life cycle

3. RELATED WORK

In 2012, a paper was published with title "Cloudlets: Bring the cloud to the mobile user". They presented a new dynamic cloudlet architecture figure3, where applications are managed on a component level. The architecture consists of three elements first one is Execution Environment (EE) run on top of an OS; in virtualizes or real hardware, the second is a component that provides an interface and node agent is the last one that mange all the EEs and monitors the recourses [9].

In 2013, paper with title "MobSched: Customizable Scheduler for Mobile Cloud Computing" analyze the reason why Hadoop's performance is poor in MANET, and not aware of mobility and multi-hop nature of MANET. They discuss two ways to deploy mobile cloud computing in an efficient manner: MobSched as a customizable job scheduler "it is based on a linear programming formulation" and a mobile friendly MapReduce framework [10].

In addition at the same year, a paper was published with title "Migrate Or Not? Explosion dynamic task migration in mobile cloud computing systems" the authors presented a migration mechanism that effectively brings the computing power of the cloud closer to the mobile user. The Virtual Machine (VM) migration mechanism enabled cloud providers

to adjust the load at each server at will. They proposed to send part of VM not all the data (Overlay VM = Base VM - Launch VM) then the overlay VM migrates from cloudlet to be other with mobile user as figure4 shows. This new trend makes VM migration techniques more powerful [11], [12].

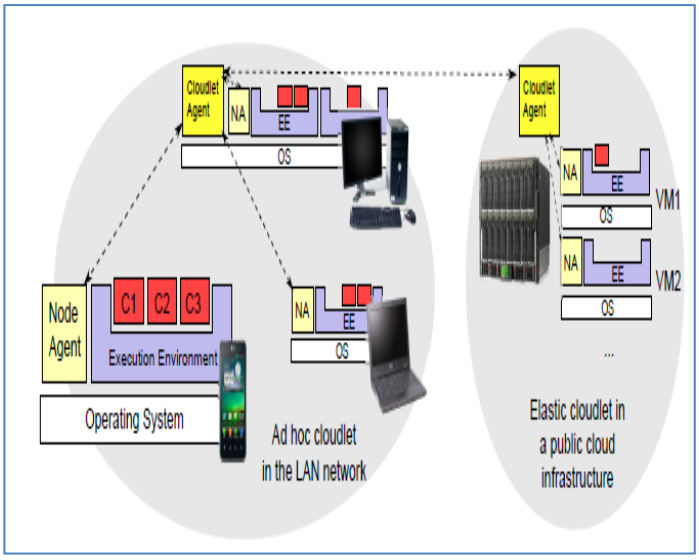

Figure 3: Distributed components in two cloudlets.

Figure 4: Virtual Machine Overlay process

4. HEADINGS AND FOOTNOTES

In this article, we proposed a set of agent teams to address the individual functions and then define specialized software agents operating within these teams, the below figure (level one proposal system) illustrates the main operations in the cloudlet management function and the layers that explained.

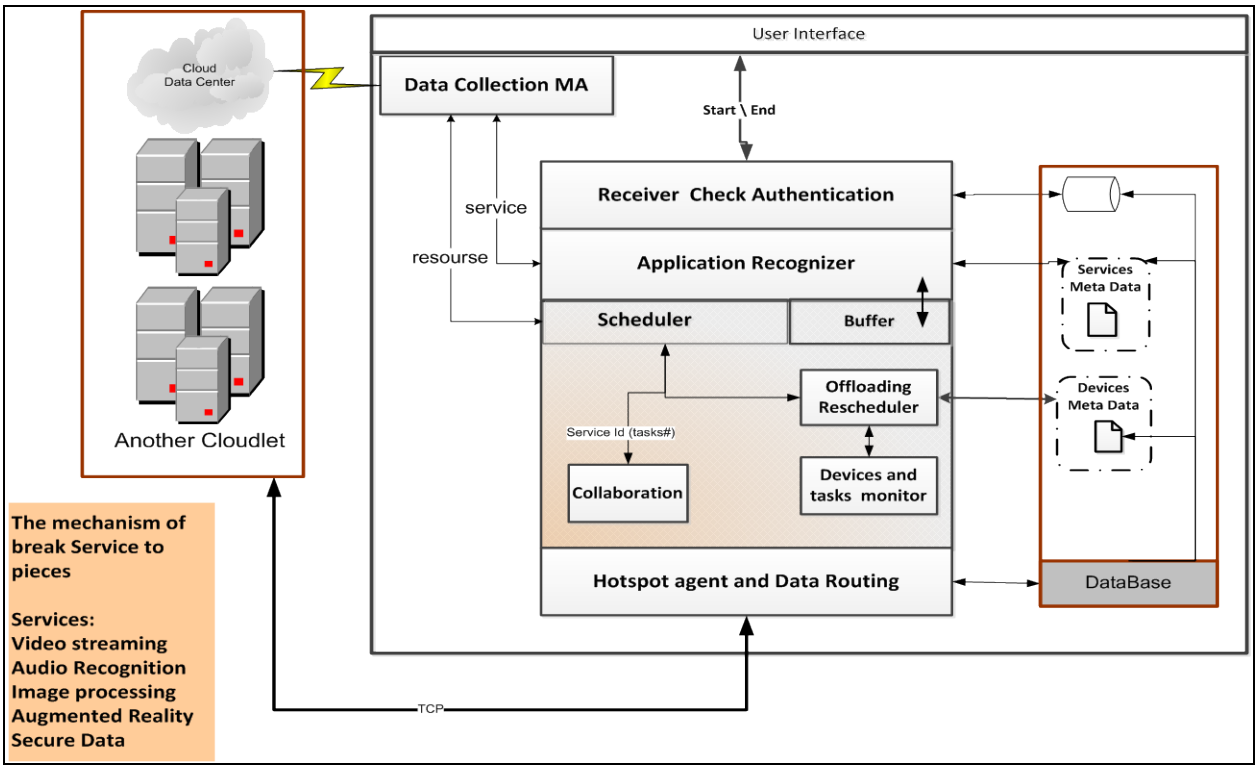

Figure 5: Proposed Architecture elements

These working agents include:

Authentication agent: that responsible to supports all mobile users who have a permeation to access the cloudlet because they are trust members in the main cloud using a privet key.

Application recognizer agent that responsible to divided one service to many tasks with suitable fragmentation method and based on some specifications.1 The main functions of this layer are split service to tasks, encapsulate each task with an essential data, put all tasks in a buffer that scheduler can access it and send the request to the scheduler to start work and when the scheduler finish this layer must reassembles fragmented tasks to become an integrated service.

However, object oriented needs a specific algorithm that analyze data and detect a certain information from the user request, But web service or component based system is easier to categorize and recognize so if the job is a service we will make a call service or component and if the job integrated (group of web services), then each one becomes a job with one call.

Hot spot agent and data routing this agent responsible for many network communication functions between cloud and cloudlets using wire or wireless connection and the two main function are routing data (IP address or privet keys) and routing service (exchange all information about the available services on all cloudlets that belong to the same cloud).

Scheduler: this agent has many basic functions that divided on several executors as follows:

Offloading agent: match each task to device using hash function and if any failure happened this agent responsible to reschedule and transfer the task to another device (the task should transfer from the last status that arrived).

Device and task monitor agent: It is responsible for monitoring tasks on devices, so must contact with operating systems of each device to check the device status (memory, buttery, task pointer...)

Collaborate agent: responsible to re-assembly of small tasks in a logical manner

The scheduling mechanism in our proposed architecture is linear programming formulation like (MobSched) that descried in the previous related work, can efficiently optimize multiple objectives such as power and (or) throughput.

Database in this architecture must contain all privet keys or IP about trusted cloudlets, meta data about the available services and devices in all cloudlets. The system supports some time sensitive services, and the mobile user needs it through his mobility and needs intensive processing on powerful, and these services have priority based on the effects that it could have happened through time change, and this table shows this priority put by IEEE 802.16e standard [13].

Table 1 : Mobile Network Qos Service Classes

Priority	Service	Definition	Example
1	Unsolicited Grant Service	Real-time data streams comprising fixed-size data packets issued at periodic intervals	T1/E1 transport
2	Extended Real-time Polling Service.	Real-time service flows that generate variable-sized data packets on a periodic basis	VoIP
3	Real-time Polling Service.	Real-time data streams comprising variable-sized data packets that are issued at periodic intervals	MPEG Video
4	Non-real-time Polling Service.	Delay-tolerant data streams comprising variable-sized data packets for which a minimum data rate is required	FTP with guaranteed minimum throughput
5	Best Effort.	Data streams for which no minimum service level is required and therefore may be handled on a space-available basis	HTTP

4.1 Algorithm of Manger

This proposed architecture need's pre-requests on each cloudlets such as the user becomes trusted when he can access the main cloud and routing table with trusted cloudlets' IP should be existed using (TCP protocol). This algorithm describes the functional implementation:

[1] in this layer we can put a static roles to split the service or using a tool to split

1. Receive request (service).
2. Check (User Privet Key)
3. If (Not Trusted) : display (Error message).
4. Else:
   - Update routing data();
   - bandwidth = detect bandwidth();
   - Service Type = read Service request ();
   - Service Info = Service [Service Type].Full info \\ ***service ID, Sequence Number, default delay, resources needs, default cost, priority.***
5. **Call Application Recognition Procedure (Service type, Service Info, bandwidth);**
6. **Service Info** = **Service Info + tasks Info**;
7. If (Buffer & j) then **Call Scheduling Procedure (Service Info, Buffer)**.
8. If (Buffer)
9. Reassemble the service (tasks).
10. Reply user();

The flowing all sub procedures that used in the previous algorithm

Structure Application Recognition Procedure \\ **Return Structure [**Buffer , j, Task info]
Structure Buffer; int j=0;
If (Service Type exist) then
    Select fragment methodology(Service Type);
    Tasks = extract tasks();
    Task info = for each task Encapsulate (TasksID. service info)
Else,
    Launch a mobile agent(Trip bath, necessary Service Type information);
    Update database ();
Return Buffer = Fill buffer (tasks, j=1)

Structure Scheduling Procedure:
Structure schedule, Buffer;
schedule =Extract Scheduling info.Access buffer (Service Info, Fill Buffer);
Send_to_collaboration agent (service Id, number of tasks)
Send_to_ offloading agent (all_tasks).
Set i = number of tasks
Set task [] = all_tasks;
cost = Cost_offloading (latency, bandwidth)
    If (cost > main cloud or close cloudlet cost) then reply "perform tasks on the main cloud".
    Else, While (i > 0) Start_ achieve task(i)
        Check recourses need = Open task();
        Selected device = Access devices database();
        schedule = Hash function (task (i) , Selected device);
        End while (i= i-1);
    SendTo_monitor device agent(schedule);
    All Tasks = **Call Check-Status procedure**(task i, device i)
    Buffer =Fill Buffer (**All Tasks** has same service ID)
    If cloudlet has not enough resources.
        Hot spot agent. Lunch agent Looking Out(trip, recourses needs) back to step 5;
Return Buffer;

VAR Check-Status procedure:
Task Status= for each device (Communicate MobileOS(Back task status)
Device Status= for each device (Communicate MobileOS(Back device status)
for all tasks i=1 to i= tasks number

1. If (Task Status =FIN) and (Device Status=OK), then
   - Send_to_collaboration agent (task i)
   - If (collaboration agent (Arrived all tasks has same serviceId)),then send to main scheduler.
2. Else (failure happened)
   - Reallocate (task at arrived statues) to another device.

End for

Figure6 below presents the methods and parameters that used in the proposed architecture and system needs.

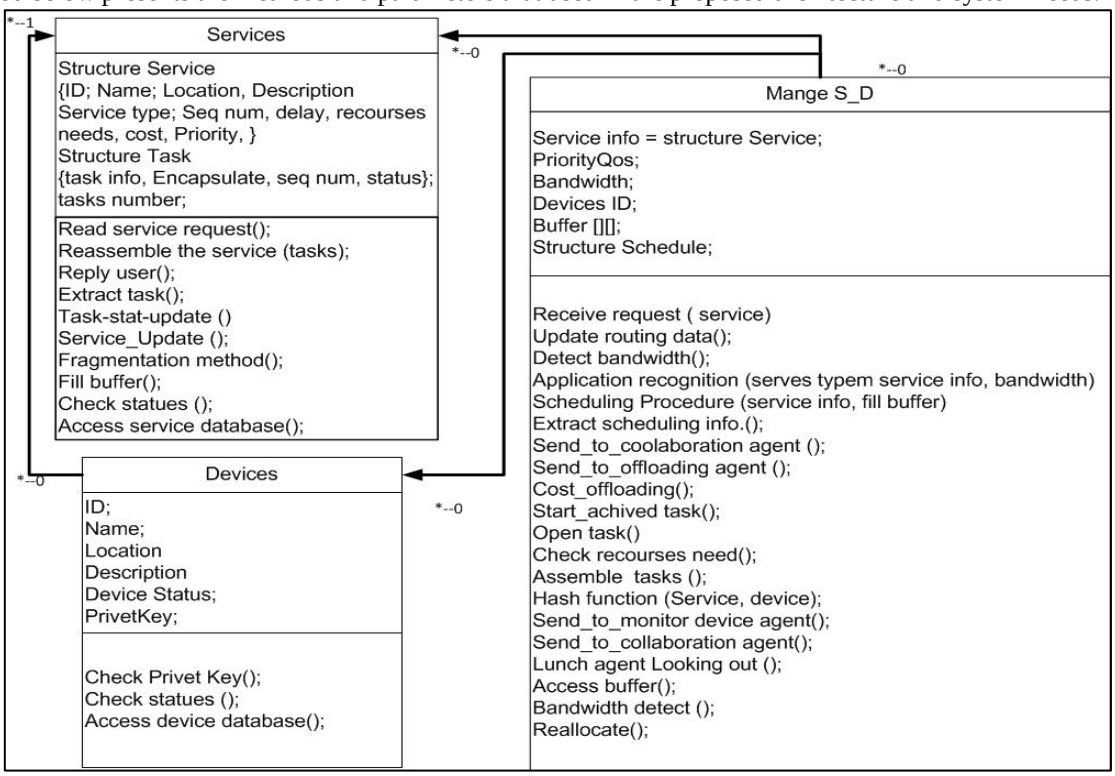

Figure 6: Class Diagram

5. EVALUATION AND DISCUSSION

Our proposed architecture comparison with others is a better solution for MCC because agent technology has many strong features that component based technology such as Mobility and less communication cost. Moreover, with providing of many service's type that categorized based on QoS attributes give the architecture more importance and make our proposal architecture more suitable for MCC.

MobScheduler that has a linear programming formulation comparison with other schedulers such as uniform load balancing, FIFO, and clustering types show that the MobScheduler performs best when it comes to optimizing for specific criteria such as total power consumption within reasonable latency, so we can be focusing to use it as a scheduling mechanism.

The proposed system provides quality attributes like scalability and security; that's because we can load each manager on a cloudlet and when we need more devices, we can add a new cloudlet server. The security achieved because the authentication agent used a privet key for each device that can access the cloudlet.

The following table presents a comparative study between our proposed architecture, and the two related work that we showed previously.

.

Table 2: Comparison dissociation between Some Mangers Methods in related work

	Paper1	***Paper2***	***Proposed Architecture***
	Mangers based on Virtual Machine	***Mangers based on Component***	***Mangers based on Agent***
Communication cost	**High communication cost** because: When a new task is generated by the user, many stages will happen on the cloud like upload all data required for initialization VM, task assignment, execution. And migration down loaded.	**High communication cost** Need continues a connection for channel of communication.	Strong Mobility With less communication cost
Complexity	Each services need a new requirement to generate VM.	High complexity with execution environment needs	Need middleware for agent execution environment. Scheduling with linear formulation
Services and QoS	Any service can demand from main cloud but no concern about the security issue or QoS.	One service AR with fixed fragmentation method for one application but no concern about the security issue or QoS	Security: privet key used for trusted user. Many services that classified as IEEE 802.16e (Five QoS classes), that provided by many service's type categorized based on QoS attributes.
Scalability	Scalable with load on the main data center		More scalability with more users we add new cloudlet server

6. CONCLUSION

Architecture has been proposed help to manage mobile user request based on MCC (cloudlet)

It provides many services that need intensive process and the proposed architecture using agent technology has many advantages not only in reducing time of communication and presenting a lot of ready-made services, which make the system more powerful and professional, but also with mobility issues. Like this system, many problems will be solved such as Internet latency, management integrity in one place, many quality attributes achieved based this architecture as following:

- Scalability is achieved when each manger loaded on cloudlet and when we need more devices, we can add a new cloudlet server and security attributes achieved when an authentication agent work using privet key.

Finally, because developers need to easily split up services or applications in different components (tasks), we must be thinking in a (semi-) automatic way and with as less burden on the developer as possible. Moreover, these will make creating a virtual machine to each service easier.

Future work enhances the architecture by providing a mechanism of job detection or recognition for object oriented request. In addition, studying how we can minimize the response time by providing a virtual machine overlay for each service..

7. REFERENCES

[1] MMA Mobile Marketing Association, “Mobile Location Based Services Marketing whitepaper”, Journal of mmaglobal, 2011.

[2] Hoang T. Dinsh, Chonho Lee, Dusit Niato, ping Wang, “Survey of mobile cloud computing: architecture, applications and approaches”, Wireless Communications and Mobile Computing, Wily online Library, 2011.

[3] Jinhu L.u, Guanrong Chen, Xinghuo Yu. “Modelling, Analysis and Control of Multi-Agent Systems: A Brief Overview”, IEEE Computer Society, 2012.

[4] Debessay Fesehaye, Yunlong Gao, Klara Nahrstedt, Guijun Wang, “Impact of Cloudlets on Interactive Mobile Cloud Applications”, IEEE 16th International Enterprise Distributed Object Computing Conference, 2012.

[5] Naif Aljabri, Fathy Essa, “Scheduling Manager for Mobile Cloud Using Multi-Agents”, International Journal of Computer and Information Technology, 2013.

[6] Mahadev Satyanarayanan, Paramvir Bahl, Ramón Cáceres, Nigel Davies, "The Case for VM-Based Cloudlets in Mobile Computing", PERVASIVE computing, IEEE CS.2012
[7] Adel Aneiba "Mobile Agents Technology And Mobility", Faculty of Computing, Engineering and Technology,Staffordshire University, 2013.
[8] Paulo Marques, Paulo Simões, Luís Silva, Fernando Boavida, João Silva. "Providing Applications with Mobile Agent Technology", IEEE Computer Society, 2012.
[9] Tim Verbelen, Pieter Simoens, Filip Turck, Bart Dhoedt, "Cloudlets: Bringing the cloud to the mobile user". Ghent University - IBBT, Department of Information Technology, 2012.
[10] Suraj Sindia, Song Gao, Bobby Black, Alvin S. Lim, Vishwani Agrawal, Prathima Agrawal, "MobSched: Customizable Scheduler for Mobile Cloud Computing", IEEE, Computer society,2013.
[11] Azaros Gkatzikis, Iordanis Koutsopoulos, "Migrate or Not? Exploiting Dynamic Task Migration In Mobile Cloud Computing Systems", IEEE Wireless Communications, 2013.
[12] Sarah Clinch, Jan Harkes, Adrian Friday, Nigel Davies, Mahadev Satyanarayanan. "How Close is Close Enough ? Understanding the Role of Cloudlets in Supporting Display Appropriation by Mobile Users", IEEE International Conference on Pervasive Computing and Communications, 2013.
[13] Chakchai So-In, Raj Jain, Abdel-Karim Tamimi, "Scheduling in IEEE 802.16e Mobile WiMAX Networks: Key Issues and a Survey", IEEE Journal on Selected Areas in Communications, 2009.