Journal of The Institute of Electronics and Information Engineers Vol.52, NO.3, March 2015 http://dx.doi.org/10.5573/ieie.2015.52.3.127
ISSN 2287-5026(Print) / ISSN 2288-159X(Online)

논문 2015-52-3-16

모바일 클라우드 컴퓨팅에서 상황인지 기반 모바일 장치 CPU사용

(Mobile Device CPU usage based Context-awareness in Mobile Cloud Computing)

조 경 희*, 조 민 호*, 전 태 웅*

(Kyunghee Cho, Minho Jo, and Taewoong Jeon©)

요 약

상황인지(Context-Aware) 모바일 클라우드 컴퓨팅은 어플리케이션이 사용자의 위치, 시간, 근처에 사용자와 장치 그리고, 사용자의 상황적인 정보를 분석함으로써, 사용자 경험을 증대시킬 수 있는 새로운 전도 유망한 패러다임이다. 본 논문에서는 Volare 미들웨어를 탑재한 상황인지 모바일 클라우드 컴퓨팅 시스템의 성능 연구를 제공한다. Volare 미들웨어는 리소스와 장치의 상황을 모니터하고, 실행시간에 클라우드 서비스 요구를 동적으로 적용한다. 이러한 접근은 실행시간에 상당한 비용절감 뿐 아니라, 더욱 효과적인 자원과 신뢰할 만한 클라우드 서비스를 제공할 것이다. 또한, 다른 QoS적용 정책에 대한 상황인지 모바일 클라우드 컴퓨팅의 성능을 연구했다. 시뮬레이션 결과는 배터리 레벨이 낮고 CPU 사용량은 높고, 사용자가 초기단계 QoS를 유지할 수 없을 때, 현 적용 정책에 의해서 서비스 비용이 줄어드는 것을 보여준다. 결과적으로, 본 논문에서 제안된 적용정책을 사용하면 상황에 따라 다르게 서비스 비용을 사용자들에게 제공할 수 있다.

Abstract

Context-aware mobile cloud computing is a new promising paradigm that allows to improve user experience by analyzing contextual information such as user location, time of the day, neighboring devices and current activity. In this paper we provide performance study of context-aware mobile cloud computing system with Volare middleware. Volare monitors the resources and context of the device. and dynamically adapts cloud service requests accordingly, at discovery time or at runtime. This approach allows for more resource-efficient and reliable cloud service discovery, as well as significant cost savings at runtime. We also have studied the performance of context-aware mobile cloud computing for different quality of service (QoS) adaptation policies. Our simulations results show that when battery level is low and CPU usage is high and user cannot maintain the initial QoS, service cost is decreased according to current adaptation policy. In conclusion, the current adaptation policy suggested in this paper may improve user experience by providing a dynamically adapted service cost according to a situation.

Keywords : context-awareness, cloud computing, mobile cloud computing, MID, quality of service

Ⅰ. Introduction

Mobile cloud computing[1, 18~19] is a new platform connecting the mobile device and cloud computing to create a new infrastructure. Users want the ease of using companies' web sites or application from

* 정회원, 고려대학교 컴퓨터정보학과
(Software Engineering Lab. Dept of Computer and Information Science. Korea University)
© Corresponding Author (E-mail: jeon@korea.ac.kr)
Received ; December 30, 2014 Revised ; February 9, 2015
Accepted ; March 6, 2015

anywhere and at anytime. Mobile devices can provide this convenience. In the past client-server centered mobile service models are not able to follow the increasing demands from mobile users by services diversity, user experience, security and privacy, and so on. This paper is written as follows. Section Ⅱ focuses on the current research related to middleware and distributed systems. Section Ⅲ shows the architecture of our proposed middleware along with the description of the interaction between its modules and components. Volare introduces a middleware for monitoring the context of a mobile device that is connected to a cloud service, and dynamically adapts the services so as to make them more resource efficient, reliable and cost efficient. It acts as an intermediary. Section Ⅳ presents an example scenario to illustrate our middleware use and describes how the middleware has been used to deal with the adaptation issue. Section V presents a simulation and evaluation of the context-aware adaptive service discovery behavior of the middleware which was run on MATLAB. before achieving our middleware use. Section VI summarizes and points to future work. This paper adds a new important context, building on a previous research advanced by P. Papakos et al. That is, CPU resource is very limited in mobile device, and thus this paper suggests to add a new CPU usage status context as a part of the context for solving this limitation.[12]

Ⅱ. Background and Related Work

Mobile clouds can utilize the sensing abilities of their mobile devices such as location, acceleration, etc. and act as providers of context awareness information.[9] When a mobile user invokes a cloud service, the request is accompanied by his/her context information, and the most suitable service is selected based on that information. Here is a model consisting of four layers of context elements.[13~14]

Monitored context: including environmental and device settings, user preference for user-specific preference settings, situational context relating to monitored data on user location, time etc, and Service Context information such as Quality of Service (QoS).

Types of gaps: two types of gaps are identified: gap on functionality that relates to an available service and the needed service, and gap on non-functionality relating to differences in QoS values between the previous service and the current one.

Types of causes: multiple interfaces, and an interface may have different implementations, The gaps arise because of the mismatches between these: Service-level Unmatched, Service Interface-level Unmatched, Service Component-level Unmatched, and Component Instance-level Unmatched.

Adapter: refer to the remedial actions that should be taken to remove the aforementioned causes.

Based on this model, the authors propose a context aware service provisioning architecture consisting of three tiers: User Layer that is made up by the mobile devices where the applications run on, Agent Layer that adapts the services according to context, and Service Layer that deploys the services.

1. On context-aware adaptation for Web Service

CARISMA and CHISEL[7~8] were developed to deal with changing context and necessary adaptation and also use a declarative policy language to put in the reasoning rules for adaptation.

PLASTIC[2] Dynamic adaptation techniques in it were ready for Service-Oriented Architectures focusing on adaptation of entire workflows.

2. Service adaptations or service personalization based on the context information

Maamar et al.[6] suggest an approach of context

based web service composition. They first identify three types of contexts – user context, web service context, and resource context, and their relationships and to utilize these contexts in personalizing services, they clarify parameters of each context.

Seyler et al.[5] suggest an approach to adapt web service orchestrations based on context information[15~17]. They define meta-models for context-aware service composition, which are service composition meta-model, context meta-model, and combined meta-model of two previous meta-models. Based on meta-models, they show deployment-time adaptation and run-time adaptations.

Sheng et al.[4] develop a multi-agent based architecture thataims at providing a distributed, adaptive, and context-aware platform for personalized service provisioning. This architecture brings about three design principles Web Service, Agent, and Publish/Subscribe System, and consists of four layers such as User, Context, Orchestration and Service Layers.

Sell et al. show a way to adapt workflows context-sensitively by introducing a separate layer, adaptation layer[3, 10] The adaptation layer automatically calculates and semi-automatically executes the necessary workflow adaptations after interacting with a context service to subscribe context changes and an adaptive WFMS to implement the workflow adaptation.

Ⅲ. Context-aware Provisioning Architecture and Functionality

1. Context-aware Provisioning Architecture

The architecture is composed of three layers as shown in Fig. 1. User layer consists of multiple MIDs (Mobile Internet Devices), and client applications are deployed on MIDs. Agent Layer plays a role of adapting services by using context

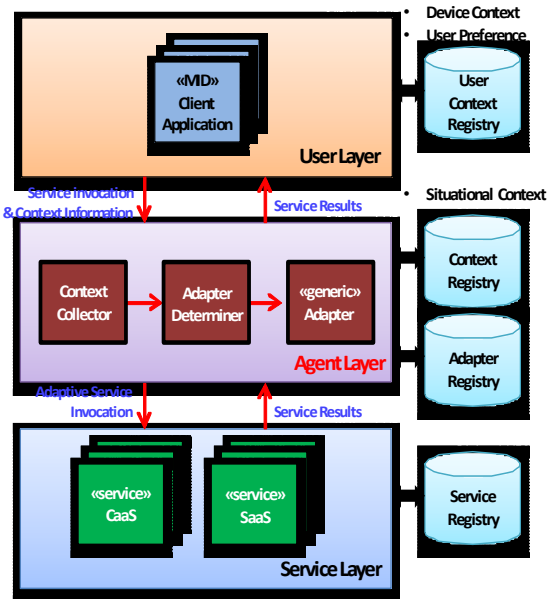


그림 1. 상황인지 예비단계 아키텍처
Fig 1. The architecture of the context aware provisioning[11].

information including user preferences.

1) Context Collector, Adapter Determiner and Adapter

Service layer deploys multiple services such as CaaS and SaaS. Fig. 2 shows three kinds of configurations

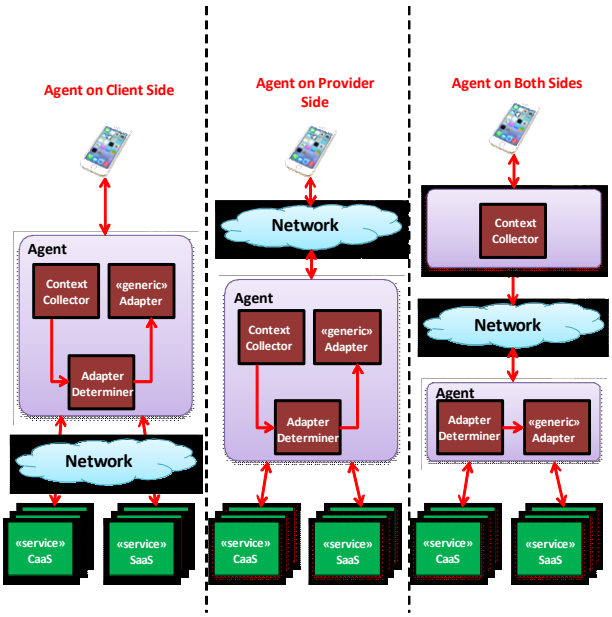


그림 2. 세 가지 실행 가능한 시스템 구성
Fig. 2. Three Possible Configurations[11].

as a result of allocating three layers to client and provider. The last configuration is where Agent Layer is distributed to client and provider side by considering functionality distribution. At runtime, when a service consumer calls for services with user and device context information, Context Collector gathers additional context information related to a service and its runtime environment. Then, Adapter Determiner is to discover several candidate services meeting functionalities required by the consumer, generic adapter is to make a service personalized to a service consumer and is specialized to more specific formswhich will be covered in the following section.

2) VOLARE middleware

VOLARE middleware is active, monitoring and adding context data through the Context Monitoring Module at a specified recheck rate (meaning the rate by which the middleware rechecks for changes in the context). The VOLARE middleware is situated on the client device, as depicted in Fig. 3. VOLARE's architecture follows a modular approach, consisting of several independent modules.

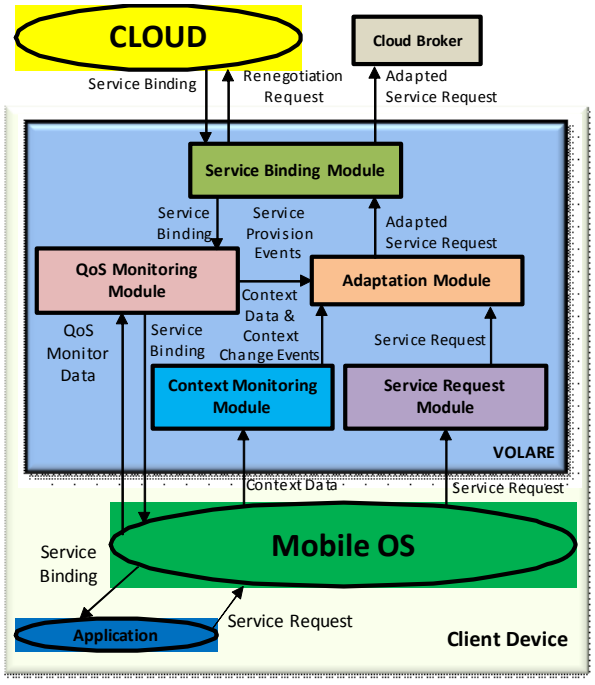


그림 3. VOLARE 미들웨어 모듈
Fig. 3. The VOLARE middleware modules[12].

The Service Request Module intercepts the service request from the client application and forwards it to the Adaptation Module. This module implementation is, by necessity, platform dependent. The Context Monitoring Module monitors the context of the device, providing context data to the Adaptation Module. When significant deviations of the context from the currently bound service are detected at runtime, the context monitor alerts the Adaptation Module to re-evaluate whether the presently bound service satisfies the requirements of the client based on its new context. This module implementation is also, by necessity, platform dependent.

The Adaptation Module at discovery time handles the adaptation of the service request according to context and resource data. or further adaptation based on the results of the initial search to re-evaluate and possibly rebind to a more appropriate service according to the context and QoS level.

The QoS Monitoring Module periodically checks if the QoS levels of service offered by the service provider deviate significantly from the agreed requirements.

The Service Binding Module initiates service discovery or rebinding sending the adapted service request to the Broker. When an appropriate service is discovered, the Service Binding Module forwards the service binding to the mobile OS through the QoS Monitoring Module.

In this paper we present an example scenario to explain VOLARE's use. Following that, we can show the VOLARE functionality and architecture and the policy language. The VOLARE middleware works at two levels. At service discovery time, it intercepts service requests from the client application while also monitoring the context of the mobile device. This may include hardware resources such as battery consumption, CPU, Memory usage etc., environmental variables like network bandwidth and user preferences like Low Cost Binding, Low Power

Operation etc. It then proceeds to adapt each service request according to the context data, to better reflect the current QoS requirements of the client. At runtime, the middleware monitors existing cloud bindings and the changing context of the device; if the context of the client device or the provided QoS level changes beyond specified thresholds then VOLARE negotiates a dynamic adaptation of the QoS levels provided by the cloud service itself or initiates service discovery and rebinding to a more appropriate service.

Ⅳ. Example Scenario

In the scenario, there are a mobile user, service provider, and cloud provider (e.g., video streaming application). If a Bandwidth drop happens due to mobility, a user does not obtain an initially requested QoS. while the binding cost and the device resources spent will correspond to the initially requested high bandwidth service, with current bandwidth possibly being only a fraction of the requested high bandwidth. User satisfaction is low and a Service Provider has to pay more to a Cloud Provider.

How can a User and a Service Provider obtain at any time the best possible QoS level and cost of binding that the current context permits?

They can do so with the VOLARE middleware installed on the mobile and consequently context-aware adaptive service discovery and binding on cloud commercial services.

Any time a context parameter changes beyond specified limits, then openly to the User and the application, the middleware will search and bind to a service providing the QoS level supported by the current context.

Ⅴ. Simulation & Evaluation

We create different adaptive policies and criteria. NormalCost is the price that a user pays when there is no adaptation policy. Whether the QoS decreases from the provider or not, the user still has to pay the full 100% of the contract that they signed for.

In Table 1, Compressed1 is a compressed video streaming. This policy happens when the battery level is in between 40% and 60% and also the CPU status also in the range of 75% and 85%. The cost for this policy decreases by the amount of the battery level and CPU usage status.

Compressed2 is a more compressed video streaming than the previous policy. This policy starts when the battery level is less than 40% and the CPU status is greater than 85%. The cost for this policy decreases by the amount of the battery level and CPU usage status.

The last one is PlainDefault. This is the default policy when the above policy types do not meet the criteria specified for each of them. It considers the device is working in a normal condition and the user can get a quality service.

When the CPU usage status increases and Battery level decreases the device cannot handle the original QoS. So it decreases the cost the user has to pay by

표 1. Compressed1, Compressed2 and PlainDefault에 관한 정책과 적용기준
Table 1. Policy and criteria on Compressed1, Compressed2, and PlainDefault.

	Policy	Criteria
Compressed 1	NormalCost = NormalCost ((Bandwidth / BaseBandwidth) - 0.2 (Battery) - 0.1 (1 - CPU))	Battery >=40% and Battery < 60%; CPU > 75% and CPU <=85%;
Compressed 2	NormalCost = NormalPrice ((Bandwidth / BaseBandwidth) - 0.2 (Battery) - 0.25 (1 - CPU))	Battery <40%; CPU > 85%;
PlainDefault	NormalCost = NormalCost	Default:

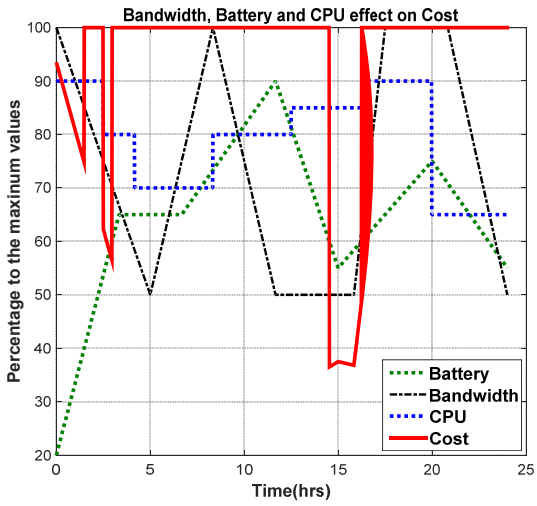


그림 4. 시뮬레이션 결과
Fig 4. Simulation Output.

some numerical factor. Each cost has to be decreased by the amount of the battery and CPU status with some multiplying factor.

As it is shown in the cost calculation the multiplying factors increase as the status of the device becomes highly difficult to handle the request. When there was no condition that satisfy the two compression policies the cost shown in red line is 100%, it means the user will pay the total amount of money to the service provider.

When there was a condition that satisfies the second policy (Battery <40% and CPU > 85%) or first policy (Battery >=40%, Battery < 60%, CPU > 75% and CPU <=85%) the cost decreases as shown in the simulation output.

Ⅵ. Conclusion and Future work

We observe a potential positive connection between Cloud Computing(CC)[20] and Mobile Internet Device(MID), which is to providing services for MIDs. That's why problems caused by resource limitation of MID can be resolved by deploying functionalities on the provider side in the form of service. Here is a way of technical issues to resolve; monitoring user contexts and provisioning services for the monitored contexts.

VOLARE middleware monitors the context of the mobile device and then carries on to dynamically adapt cloud service requests accordingly. This produces an improvement on service discovery of cloud services and binding, resource efficiency and cost-effectiveness by choosing the most appropriate services conforming to the current needs of the client. Monitoring of the QoS levels and available resources will allow for rebinding during runtime, which will increase system reliability and reduce resource and binding costs. The middleware functionality is also searched for extending to discover and bind to more than one service on the cloud. An open issue that remains to be addressed is to determine automatically reasonable adaptation policies and recheck rate without need for a manual adaptation policy from the developer, using the context history of the device to anticipate context changes in the future and produce or modify adaptation policies accordingly.

REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica and M. Zaharia, "Above the clouds: A Berkeley View of cloud Computing," UCB/EECS-2009-28, 2009.

[2] A. Bertolino, W. Emmerich, P. Inverardi, V. Issarny, F. Liotopoulos and P. Plaza, "PLASTIC: Providing Lightweight Adaptable Service Technology for Pervasive Information & Communication," in Proc. 23rd ASE IEEE/ACM International Conference on Bertolino. 2009.

[3] C. Sell and T. Springer, "Context-Sensitive Adaptation of Workflows," In Proc. of the doctoral symposium for ESEC/FSE on Doctoral symposium, pp. 1-4, 2009.

[4] Q. Z. Sheng, B. Benatallah and Z. Maamar, "User-Centric Services Provisioning in Wireless Environments," Communications of the ACM, vol. 51, no. 11, pp. 130-135, 2008.

[5] F. Seyler and C. Taconet, "Context Adaptation of Web Service Orchestrations," In Proc. of the

16th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, pp. 351-356, 2007.
[6] Z. Maamar, S. K. Mostefaoui and Q. H. Mahmoud, "Context for Personalized Web Services," In Proc. of the 28th Hawaii International Conference on System Sciences, pp. 66-73, 2005.
[7] L. Capra, W. Emmerich and C. Mascolo, "CARISMA: context- aware reflective middleware system for mobile applications," IEEE Transactions on Software Engineering, vol. 29, no. 10, pp. 929 - 945, 2003.
[8] J. Keeney and V. Cahill, "CHISEL: Policy-Driven, Context-Aware, Dynamic Adaptation Framework," 4th IEEE International Workshop on Policies for Distributed Systems and Networks, pp. 3-14, 2003.
[9] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith and P. Steggles, "Towards a Better Understanding of Context and Context-Awareness," In Proc. of the 1st International Symposium on Handheld and Ubiquitous Computing, Lecture Notes in Computer Science 1707, pp. 304-307, 1999.
[10] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, "RTP: A transport protocol for real-time applications," IETFRFC, 1996.
[11] H. La and S. Kim, "A Conceptual Framework for Provisioning Context-aware Mobile Cloud Services," In Proc. of the 3rd IEEE International Conference on Cloud Computing, pp. 466-473, 2010.
[12] P. Papakos, L. Capra and D. S. Rosenblum, "VOLARE: Context-Aware Adaptive Cloud Service Discovery for Mobile Systems," In Proc. of the 9th International Workshop on Adaptive and Reflective Middleware, pp. 32-38, 2010.
[13] N. Fernando, S. W. Loke and W. Rahayu, "Mobile cloud computing: A survey," Future Generation Computer Systems, vol. 29, no. 1, pp. 84-106, 2013.
[14] H. T. Dinh, C. Lee, D. Niyato and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," Wireless Communications and Mobile Computing, vol. 13, no 18, pp. 1587-1611, 2013.
[15] A. Klein, C. Mannweiler, J. Schneider and H. D. Schotten, "Access Schemes for Mobile Cloud Computing," in Proc. Of the 11th International Conference on Mobile Data Management, pp. 387-392, Kansas City, USA, May 2010.
[16] D. Kovachev and R. Klamma, "Context-aware Mobile Multimedia Services in the Cloud," in Proc. Of the 10th International Workshop of the Multimedia Metadata Community on Semantic Multimedia Database Technologies, Graz, Austria, December 2009.
[17] K. Akherfi and H. Harroud, "Mobile Cloud Middleware: A New Service for Mobile Users," International Journal of Computer, Information, Systems and Control Engineering, vol. 8, no. 3, pp. 27-31, 2014.
[18] W. Song and X. Su, "Review of Mobile cloud computing," in Proc. Of the 3rd International Conference on Communication Software and Networks, pp. 1-4, Xían, China, May 2011.
[19] H. Qi and A. Gani, "Research on Mobile Cloud Computing: Review, Trend and Perspectives," in Proc. Of the 2nd International Conference on Digital Information and Communication Technology and its Applications, pp. 195-202, 2012.
[20] G. Pallis, "Cloud Computing: The New Frontier of Internet Computing," IEEE Internet Computing, vol. 14, no. 5, pp. 70-73, 2010.

APPENDIX: Simulation Code in MATLAB

Input: ?
Output: ?
Method:

```
% For every minute 24hr * 60min = 1440min
maxbandwidth = 256
% Battery status in percentage
battery=zeros(1,1441)
% Bandwidth status in percentage
bandwidth=zeros(1,1441)
cpu=zeros(1,1441) % CPU status in percentage
cost=zeros(1,1441)
y=0:1:100
x=0:1:1440

% Battery
% A random battery percentage status
battery([1:200]) = (0.225)*x(1:200)+ 20
battery([201:400]) = 65
battery([401:700]) = 31.6667+((0.08333)*x(401:700))
```

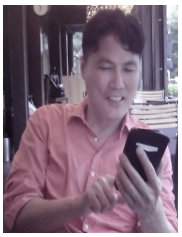
```
    battery([701:900]) = 212.5-(0.175)*x(701:900)
10: battery([901:1200]) = (-5 +((0.0667)*x(901:1200)))
11: battery([1201:1441]) = 175-((0.08333)*x(1201:1441))
12:
    % Bandwidth
    % Assuming a fluctuating bandwidth between
      128kbs and 256kbs
    bandwidth([1:300]) = (((-0.42667)*x(1:300) +
      256)/maxbandwidth)*100
13: bandwidth([301:500]) = (((0.64)*x(301:500) -
      64)/maxbandwidth)*100
14: bandwidth([501:700]) = (((-0.64)*x(501:700) +
      576)/maxbandwidth)*100
15: bandwidth([701:950]) = ((128)/maxbandwidth)*100
    bandwidth([951:1050]) = (((1.28)*x(951:1050) -
16:     1088)/maxbandwidth)*100
    bandwidth([1051:1250])= ((256)/maxb
17: andwidth)*100
    bandwidth([1251:1441]) = (((-0.6737)*x(1251:1441)
18:     + 1098.1053)/maxbandwidth)*100

19: % CPU
    % Assuming a fluctuating CPU status
    CPU([1:150]) = 90
    CPU([151:250]) = 80
    CPU([251:500]) = 70
    CPU([501:750]) = 80
20: CPU([751:1000]) = 85
21: CPU([1000:1200]) = 90
22: CPU([1200:1441]) = 65
23:
24: % Cost
25: % From the Policy and the Criteria.
26: for c = 1:1441
      % Compressed1 Video streaming
      if battery(c)>=40 & battery(c)<60 &
        CPU(c)>75 & CPU(c)<=85
27:     cost(c) =
          bandwidth(c)-0.2*battery(c)-0.1*(100-CPU
28:       (c))
      % Compressed2 Video streaming
29:   elseif battery(c)<40 & CPU(c)>85
        cost(c) =
          bandwidth(c)-0.2*battery(c)-0.25*(100-CP
          U(c))
30:
31:
      % Default Video streaming
      else
        cost(c) = 100
32:   end
33: end
34:
35: % to make it into hrs
    x=x/60
    plot(x,battery,'g:','LineWidth',1.5)
36: hold on
37: plot(x,bandwidth,'k-.','LineWidth',1)
38: hold on
39: plot(x,CPU,'b:','LineWidth',1.5)
40: hold on
41: plot(x,cost,'r-','LineWidth',1)
42: xlabel('Time(hrs)')
43: ylabel('Percentage to the maxinum values')
44: title('Bandwidth, Battery and CPU efect on Cost')
45: legend('Battery','Bandwidth','CPU', 'Cost')
46:

47:
```

저 자 소 개

조 경 희(정회원)
1994년 성결대학교 영어영문과 학사 졸업.
1997년 연세대학교 교육대학원 영어교육학 석사졸업.
2013년~현재 고려대학교 컴퓨터 정보학과 박사과정
<주관심분야: Software, cloud computing, 통신, 컴퓨터>



조 민 호(정회원)
1984년 조선대학교 전자공학과 학사 졸업.
1994년 Ph.D, Lehigh University, Industrial and Systems
<주관심분야 : Cognitive Radio, IoT, Mobile cloud computing>

전 태 웅(정회원)
1981년 서울대학교 컴퓨터통계 학과 학사 졸업.
1983년 서울대학교 컴퓨터통계 학과 석사졸업.
1992년 Ph.D. Illinois Institute of Technology, Computer Science
<주관심분야 : Software Engineering, 통신, 컴퓨터>