Hierarchical Cloud Computing Architecture for Context-Aware IoT Services

Tae-Dong Lee, Byung Moo Lee, *Member, IEEE*, and Wonjong Noh, *Member, IEEE*

***Abstract*—This paper presents a new cloud computing model for context-aware Internet of Things services. The proposed computing model is hierarchically composed of two layers: a cloud control layer (CCL) and a user control layer (UCL). The CCL manages cloud resource allocation, service scheduling, service profile, and service adaptation policy from a system performance point of view. Meanwhile, the UCL manages end-to-end service connection and service context from a user performance point of view. The proposed model can support nonuniform service binding and its real-time adaptation using meta-objects. Furthermore, it supports intelligent service-context management using a supervised and reinforcement learning-based machine learning framework. We implemented a lightweight prototype of the proposed computing model. Evaluations confirm that the proposed computing model offers enhanced performance compared with legacy uniform computing models.**

***Index Terms*—Cloud computing, context-aware IoT service, hierarchical control architecture.**

I. Introduction

At present, the patterns of application services, networks, and computing are changing very rapidly. First, the rapid improvement of networks and end-systems has led to changes in services from simple applications to a variety of intelligent multimedia applications. Second, improved ubiquitous interoperability and convergence technologies have led to changes in networks, from cellular- and Wi-Fi-based networks to heterogeneous networks including all-IP, device-to-device, ad-hoc, sensor networks, and Internet of Things (IoT). Lastly, new software-defined radio, resource virtualization, and network security technologies have led to user-oriented computing platforms [1]–[4]. With these changes, various smart IoT services such as networked game, healthcare, home automation, virtual reality (VR) multimedia, and artificial intelligent (AI) robots, depicted in Fig. 1, are becoming very important.

Manuscript received April 1, 2018; accepted May 15, 2018. Date of publication June 7, 2018; date of current version June 28, 2018. This work was supported by the Basic Science Research Program through the National Research Foundation of South Korea funded by the Ministry of Education under Grant 2017R1D1A1B03036526. *(Corresponding author: Wonjong Noh.)*

T.-D. Lee is with the School of Electrical Engineering, Korea University, Seoul 02841, South Korea (e-mail: leetd@korea.ac.kr).

B. M. Lee is with the School of Intelligent Mechatronics Engineering, Sejong University, Seoul 05006, South Korea (e-mail: blee@sejong.ac.kr).

W. Noh is with the Communication and Networking Group, Samsung Electronics Company Ltd., Suwon 16677, South Korea (e-mail: wonjong.noh@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCE.2018.2844724

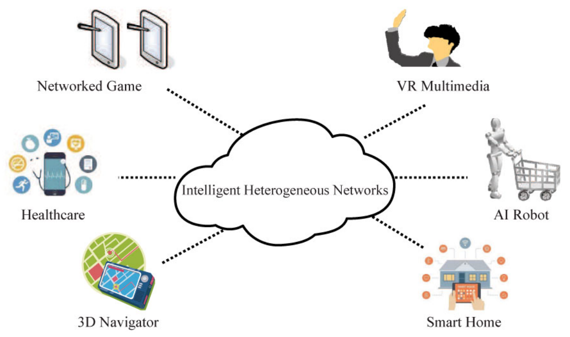


Fig. 1. Context-aware smart IoT services and computing platform.

For the best provision of smart IoT services, first, service-specific transmission should be provided. Second, the service connection should be able to dynamically change itself as the context changes. For example, as the computing user moves, there may be fluctuations in throughput and delay because the level of connectivity varies. To cope with such requirements, efficient adaptation should take place at a variety of levels in the system. Third, intelligent context awareness should be supported.

This paper proposes a new cloud computing model that satisfies these requirements. Section II summarizes recent related works and the main contributions. Section III elaborates on the proposed cloud computing architecture. Section IV describes the main control procedures and algorithms of the proposed computing platform. Performance evaluations and conclusions are presented in Sections V and VI, respectively.

II. Related Works and Main Contributions

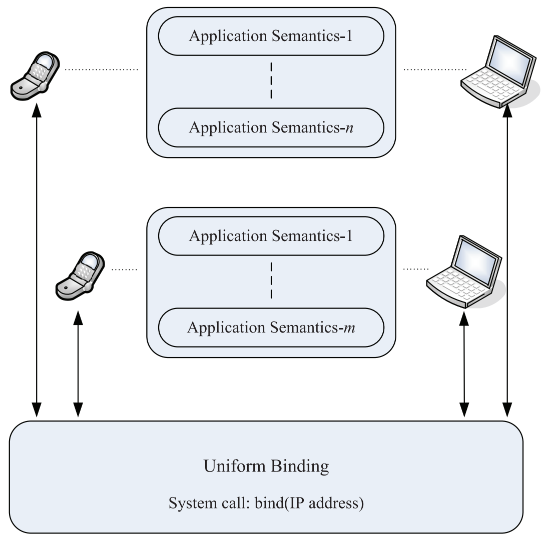
Recently, many context-aware cloud computing platforms [5]–[11] have been proposed. Xing *et al.* [5] presented a new geo-distributed cloud computing infrastructure for ad-hoc mobile network users, called MobiCloud. It supports Internet-based resource outsourcing by allowing context-aware mobile users to construct a virtual network system through virtualization and programmable networking technologies. It can provide ad-hoc mobile users with reduced service access latency and increased cloud infrastructure utilization. Haw *et al.* [6] and Guerrero-Contreras *et al.* [7] presented content-centric cloud computing architectures to improve the content delivery performance. In particular, [6] proposed content delivery framework with Software Defined Networking (SDN) and Content Centric Networking (CCN). Additionally to serve autonomic optimal services, it proposed a reinforcement learning-based context-aware content delivery scheme. Reference [7] proposed a service replication scheme together with a self-conguration approach for the activation and hibernation of the replicas of a service depending on relevant context information from the mobile system. To that end, an election algorithm has been designed and implemented. Mitra *et al.* [8] presented a mobility-efficient cloud computing architecture. It proposed multi-homed mobile IP (M-MIP) as a multi-homed mobility management protocol to facilitate soft, low latency handoffs with minimal packet loss. M-MIP enables a user to connect to several access networks simultaneously before initiating the handoff process. The user periodically probes the registered network interfaces to select a target network for handoff without disconnecting from the previous network interface, thereby minimizing network delay and packet losses during the handoff process. Zhou *et al.* [9] and Liu *et al.* [10] presented off-loading efficient cloud computing architectures. They proposed context-aware off-loading controls obtaining the lowest task execution time and energy consumption for the off-loadable tasks. The proposed control decides when it is benecial to off-load, which wireless medium is used for off-loading and which resources to use as the off-loading location considering a set of context parameters, multiple wireless medium and mobile cloud resources. Yan *et al.* [11] presented verifiable and secure cloud computing architecture to enhance the trust of cloud computing. It first proposed full homomorphic encryption technologies to process data in an encrypted form at Cloud Service Provider (CSP) in order to protect the privacy of data providers and data owners. It further deployed an auditing protocol to verify the correctness of encrypted data processing by applying a Trusted Auditing Proxy (TAP).

Also, a number of studies [12]–[14] on binding adaptations in the cloud computing platform have been carried out. Dohndorf *et al.* [12] proposed adaptive and reliable bindings in ambient service systems. It proposed a policy-based binding management. The policy predefines the adaptation behavior of the run-time system by means of abstract guidelines that will be represented as low-level policies enforced by the management to reconfigure the system. It can be flexibly exchanged at run-time. Hsieh *et al.* [13] and Yasaki *et al.* [14] presented a dynamic reconfigurable wireless connection between smart-phone and gateway. Many IoT wireless devices would be connected to applications on smart-phones and then provide services to the end user. However, the network capability of smart-phone is limited, and it is not currently possible to connect multiple devices at one time. Therefore, when trying to connect devices beyond the network capability of smart-phone, aid from something like a gateway is helpful to continue the service. For a dynamic reconfigurable wireless connection system that migrates network connections, these works established ways to migrate devices from a smart-phone to the gateway while continuing the application on the smart-phone by introducing a driver management framework that can migrate the communication and network driver modules handling the network connection. It also proposed a software-hardware co-design and implementation that enable network adaptation operations.

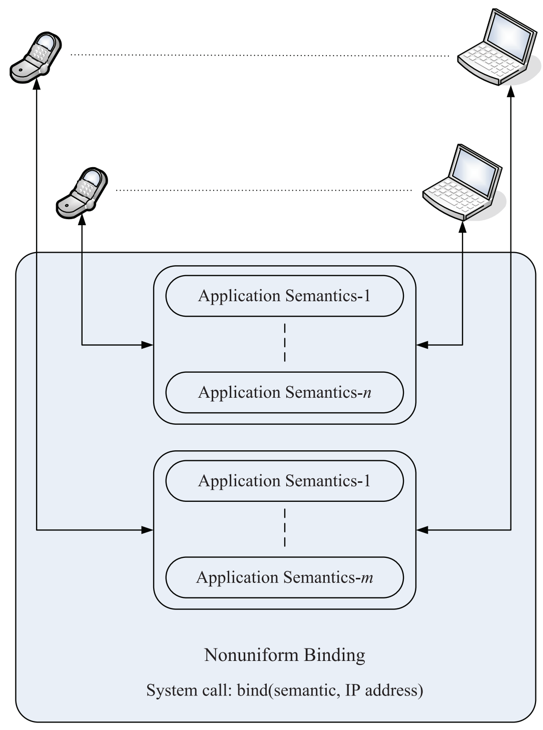
The cloud computing platforms and the adaptive bindings in [5]–[14] are all based on standard object-oriented middleware (OOM) such as the Java Remote Method Invocation (RMI) [15], the Open Groups Distributed Computing Environment (DCE) [16], the IETFs Remote Procedure Call (RPC) [17], the ISOs Reference Model of Open Distributed Processing (RM-ODP) [18], and the OMGs Common Object Request Broker Architecture (CORBA) [19]. The standard binding models have the following weaknesses. First, the standard binding model supports the uniform binding shown in Fig. 2(a). That is, the uniform binding supports only the common semantics that would be used in all applications. The uniform binding was focused mainly on interoperability and portability, and therefore it has boosted the widespread deployment of distributed applications. However, it did not support sufficient application-specific computing semantics at the platform level, so that most application-specific computing semantics were mainly processed by the end-users. This makes application coding difficult for developers and does not provide efficient performance at the platform level. Second, the standard binding model can be adapted according to a change in the service context. However, it does not perform application adaptation at the platform level, i.e., the service- or network-rebinding adaptation is only carried out using a service or network broker. This implies that the effect of the adaptation is limited.

Meanwhile, the proposed cloud computing and binding model has the following characteristics and contributions compared with previous studies.

- The proposed platform utilizes a hierarchical macroscopic and microscopic control architecture using a cloud control layer (CCL) and user control layer (UCL). The CCL controls cloud resource allocation from a system quality of service (QoS) point of view, considering all context-aware IoT services; meanwhile, the UCL controls each context-aware service from a user QoS point of view.
- The proposed platform provides a non-uniform service-binding model that provides application-specific computing environments at the platform-level as shown in Fig. 2(b), rather than at the user level.
- The proposed platform supports real-time adaptable service binding that consists of application and transmission bindings. It supports binding-wise hierarchical adaptation control. For real-time adaptation, the proposed service binding makes use of a meta-object-based reflective system.
- The proposed platform provides intelligent service-context management using a supervised and reinforcement learning-based machine learning framework.
- We implemented and evaluated a lightweight prototype of the proposed computing model. It shows enhanced performance compared with uniform binding-based legacy computing models.
- The proposed platform can be applied to environmental sensors, actuators, service-agent devices, and network devices as a core function of computing infrastructure to provide intelligent IoT services in the future. Furthermore, the proposed platform can create new



(a) Uniform binding model.



(b) Nonuniform binding model.

Fig. 2. Uniform and nonuniform service-binding models. Uniform and nonuniform bindings are established using bind(IP address) and bind(semantics, IP address) system call, respectively.

business opportunities in cloud computing where both the service providers and the network operator take co-responsibility for the performance, reliability, and scalability of the services.

III. Proposed Cloud Computing Platform Model

The overall architecture of the proposed cloud computing platform model is shown in Fig. 3.

A. Proposed Context-Aware Service Model

The proposed platform provides many context-aware services, which are developed by the network operator or authorized third-party service provider. Each context-aware service provides different services depending on its service context. For example, a smart home service provides various service contexts such as sleeping, morning calls, dining, entertainment, and guarding modes, depending on the location, the time, and the activity. We denote a context-aware service and its service context as S_n and $C(S_n)$, respectively. Then, the service context can be expressed as

$$C(S_n) = f_n(c_{n1}, c_{n2}, c_{n3}, \ldots, c_{nj}, \ldots), \tag{1}$$

where c_{nj} is referred to as the j^{th} context factor that determines the service context of the service S_n; $f_n(\cdot)$ is a function that aggregates context-factor parameters and maps them into a service context for the service S_n.

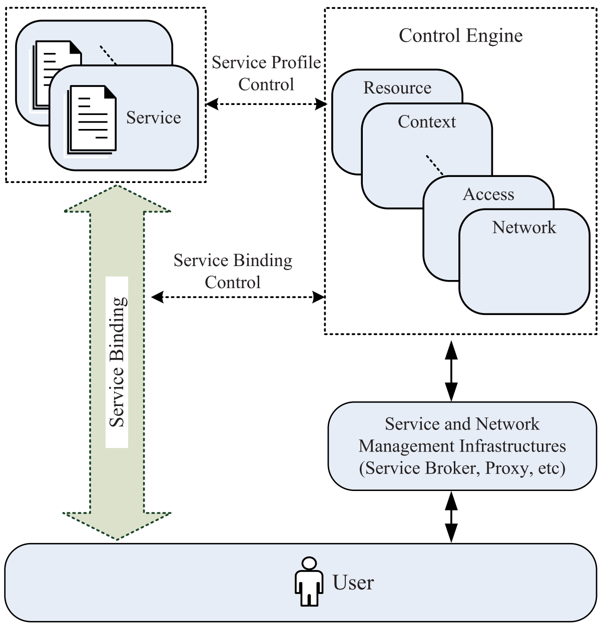


Fig. 3. The proposed computing architecture. It provides context-aware IoT services, real-time adaptable service-binding, and hierarchical control engine.

B. Proposed Service Binding Model

When an authorized user accesses a context-aware service, an association between the user and the accessed service is generated, as shown in Fig. 4, which is referred to as a service binding. The proposed service binding consists of an application-level binding and a transmission-level binding. The application-level binding represents an association among application-related objects, that is, the user interface, language, and application protocols. On the other hand, the transmission-level binding represents an association among data transmission-related objects, that is, compressors, filters, modulators, queues, caches, and transmission protocols. When a service binding is generated, it is given a binding identifier (BID), and each object in the binding is given a system object identifier (OID). These identifiers are used for platform-level control by the cloud computing platform.

The proposed computing platform adapts the service binding according to the change of service context. To make real-time binding adaptation possible, the proposed computing platform exploits a meta-object-based reflective system,

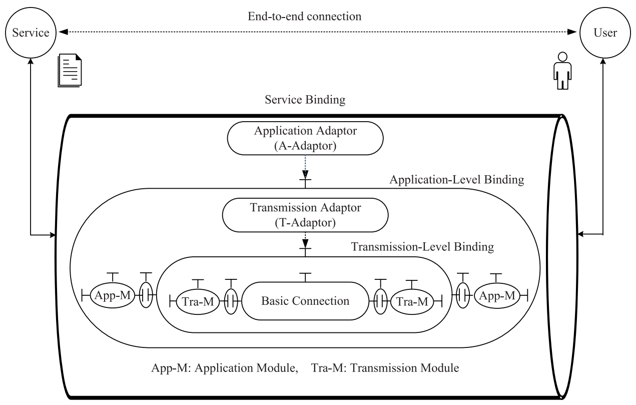


Fig. 4. The proposed service-binding model. The service-binding is composed of application binding and transmission binding hierarchically.

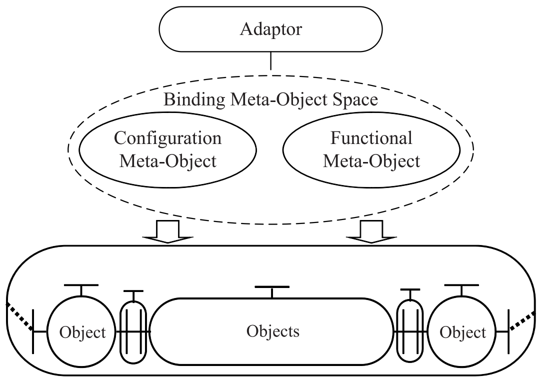


Fig. 5. Binding meta-object space. It consists of configuration and functional meta-objects.

which overcomes the limitations of the black box approach to software engineering and achieves open engineering [20]. Specifically, each application and transmission binding has its meta-object space, as shown in Fig. 5.

The meta-object space in the application and transmission binding consists of configuration and functional meta-objects. The configuration meta-object inspects and manipulates the linkage between system objects and the connection topology of the bindings. On the other hand, the functional meta-object inspects and manipulates the protocols and QoS of the bindings. In order to inspect and change the internal behavior of its binding, the meta-object has a set of the following generic functions:

- REPLACE(BID, Object/Protocol, OID, OID*), which replaces the existing system object or protocol with OID with a new one with OID* in a service-binding BID.
- INSERT(BID, Object/Protocol, OID), which inserts a new system object or protocol having OID into a service-binding BID.
- REMOVE(BID, Object/Protocol, OID), which removes the existing object or protocol having OID from a service-binding BID.
- MONITOR(BID, Object/Protocol, OID), which monitors the performance of the object or protocol having OID in a service-binding BID.

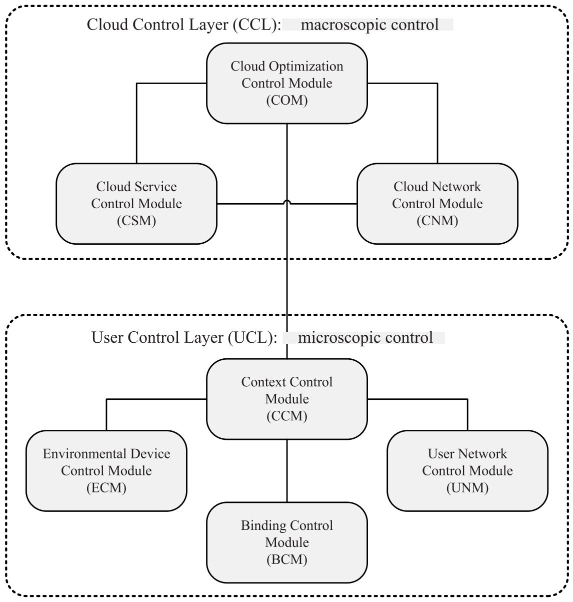

Cloud Control Layer (CCL): macroscopic control
Cloud Optimization Control Module (COM)
Cloud Service Control Module (CSM)
Cloud Network Control Module (CNM)
User Control Layer (UCL): microscopic control
Context Control Module (CCM)
Environmental Device Control Module (ECM)
User Network Control Module (UNM)
Binding Control Module (BCM)

Fig. 6. The proposed control engine. It is hierarchically composed of CCL and UCL.

C. Proposed Control Engine

The proposed control engine maintains and controls the context-aware services by performing service-context decisions, resource allocation, and service-binding adaptation. It consists of CCL and UCL hierarchically, as shown in Fig. 6.

First, the CCL is responsible for system-wide resource control and service management, which consists of three main modules: a cloud-optimization control module (COM), a cloud-service control module (CSM), and a cloud-network control module (CNM). Here, the COM controls cloud resource allocation and service scheduling using the following sub-components:

- A resource allocation component, which optimally allocates cloud resources such as network processors, storage, bandwidth, and transmission power to the services in terms of system-wide service utility maximization.
- A cloud scheduling component, which manages the activation and de-activation of the services.

The CSM controls service configuration using the following sub-components:

- A cloud service profile component, which manages how context-aware services should work according to the service contexts. It also generates new service profiles, and updates or deletes existing service profiles. Furthermore, it checks service integrity and service conflicts.
- A cloud service adaptation component, which manages the service adaptation policy, i.e., what service adaptations should be made according to the context switch.

The CNM controls cloud-related network information using the following subcomponents:

- A cloud-access component, which manages user and service authentication such as log-in, connection establishment, and connection termination. This component also shows users who pass the log-in procedure what kinds of context-aware services are available.

- A cloud security component, which protects the cloud system from unexpected users or behavior.
- A cloud network interface component, which manages a unified network access interface called the Cloud Radio Access Network (Cloud-RAN). Any network such as 3G, Long-Term Evolution (LTE), Wi-Fi, or Bluetooth can be supported by the standardized network interface,
- A cloud core-network monitoring component, which monitors system-wide network information, i.e., cloud network status and resource usage status.

Second, the UCL is responsible for context management and real-time service-binding adaptation, which consists of four main modules: a context control module (CCM), a binding control module (BCM), a user-network control module (UNM), and an environmental device control module (ECM). The CCM controls the service context using the following subcomponents:

- A context aggregate component, which collects context-factor information from environments, networks, and users.
- A context interpreting component, which translates the context-factor information to a measurable real number.
- A context reasoning component, which determines the training data and then finds the best service context for the collected context factors.

The BCM controls service-binding adaptation according to the change of the service context using the following subcomponents:

- A QoS component, which sets the target QoS into the application and transmission bindings through the A-adaptor and T-adaptor, and monitors the achieved QoS.
- A run-time code component, which manages real-time object linking and its operation through the A-adaptor and T-adaptor.

The UNM controls user-related network information using the following subcomponents:

- A mobility component, which manages the user mobility through a home location register (HLR) or visitor location register (VLR).
- An access-network monitoring component, which monitors the status of network resources that are allocated to users.

The ECM controls environmental devices such as sensors and actuators using the following subcomponents:

- A sensing component, which collects context-factor information from environmental sensors.
- An actuating component, which manages the actuation and de-actuation of the service machines.

IV. Main Control Procedures and Framework

In this section, we elaborate the service-context decision, cloud resource allocation, and real-time service-binding adaptation procedures in the proposed computing platform.

A. Service-Context Decision by CCM

We implement the service-context decision function $f_n(\cdot)$ in (1) using the machine learning framework in Fig. 7, where

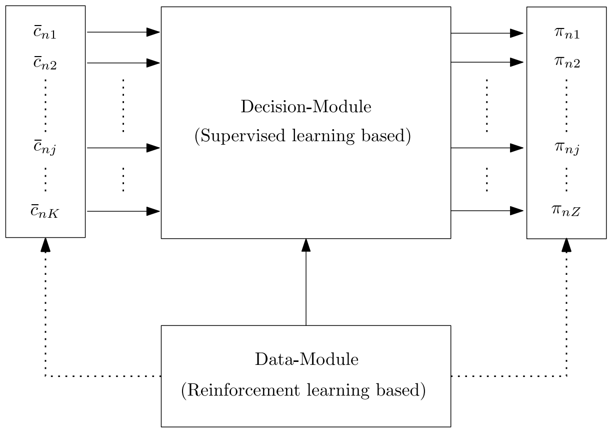


Fig. 7. Machine-learning based service-context decision. Here, K and Z are the number of context-factor and service-context of the service, respectively. It uses back-propagation supervised learning and reinforcement learning.

information $\bar{c}_{nj}$ is the measured value for the context factor c_{nj}, and π_{nj} is the probability that the service S_n is in the j^{th} service context. Initially, the data module generates the training data for the mapping between context factors and service contexts. Using these training data and supervised learning rule, the decision module learns how to determine the service context depending on the context-factor information. After the learning process, whenever the decision module receives or measures the context-factor information $\bar{c}_{nj}$, it calculates π_{nj} using the learned mapping rule and determines the service context as

$$C(S_n) = \arg \max_{j \in \{1,\ldots,Z\}} \pi_{nj}. \tag{2}$$

That is, the one with the highest probability is selected as the service context. On the other hand, the data module adjusts its training data using a reinforcement learning approach whenever a user changes its target service for certain given context factors. The overall operating procedure is as follows.

1) The ECM collects context-factor information from network sensors and reports it to the CCM.
2) The UNM collects network status information and reports it to the CCM.
3) The CCM determines the service context using the service-context decision function $f_n(\cdot)$ in the machine learning framework. In order to perform the procedure, the CCM periodically updates its training data based on the information from the ECM and UNM, and then updates the service-context decision function $f_n(\cdot)$.

B. Cloud Resource Allocation by COM

The proposed computing platform performs its resource allocation as follows.

1) We assume that services have the QoS factors of rate (r), packet loss (l), and delay (d).
2) Each QoS variable has its value as a function of the allocated resource θ : $r(\theta)$, $l(\theta)$, $d(\theta)$.
3) If the i^{th} resource is allocated to a service n as much as $\theta_{n,i}$, the utility that the service n obtains can be expressed as

$$u_n^i(r(\theta_{n,i}), l(\theta_{n,i}), d(\theta_{n,i})). \tag{3}$$

4) If there are N services and M different types of resources, then to optimally allocate the cloud resource to the context-aware services, the COM solves the following maximization problem:

$$\max \sum_{n=1}^{N} \sum_{i=1}^{M} w_n \cdot u_n^i(r(\theta_{n,i}), l(\theta_{n,i}), d(\theta_{n,i})), \tag{4}$$

where w_n denotes the weight of the service n. On the other hand, there are resource constraints as follows:

$$\sum_{n=1}^{N} \theta_{n,i} \leq \Theta_i, \quad \text{for} \quad i = 1, \ldots, M, \tag{5}$$

where Θ_i denotes the maximum available amount of a resource i. The constraints form a convex resource allocation region. On the other hand, if the utility function u_n^i is a $\log(\cdot)$ function that can reflect resource fairness among users, then the objective function also becomes a convex function. This implies that the optimal resource can be found using the Karush-Kuhn-Tucker conditions and the Lagrangian relaxation method [21], that is, the COM can determine the optimal resource allocation by solving

$$\max \sum_{n=1}^{N} \sum_{i=1}^{M} w_n \cdot u_n^i\big(r(\theta_{n,i}), l(\theta_{n,i}), d(\theta_{n,i})\big) + \sum_{i=1}^{M} \lambda_i \left(\Theta_i - \sum_{n=1}^{N} \theta_{n,i} \right). \tag{6}$$

C. Service-Binding Adaptation

We assume an example in Fig. 8, where a mobile user accesses a realistic multimedia streaming service. We also assume that the user moves around and therefore its service context fluctuates. If the service context is changed a little, e.g., from 'VR-Quality/Best' to 'VR-Quality/Good', then the service binding tries to maintain the VR-Quality service at best by inserting a 'Compression' module into its transmission binding. The service binding performs only transmission-binding adaptation. This is referred to as service-maintain-adaptation (SMA). For the adaptation, the T-adaptor inserts a new compression module into the transmission binding using the method of the configuration meta-object,

$$\text{INSERT(BID, Object, High-Compression)}, \tag{7}$$

and the T-adaptor also inserts compression protocols using the method of the functional meta-object,

$$\text{INSERT(BID, Protocol, High-Compression)}. \tag{8}$$

On the other hand, we assume that the user moves and its serving network is switched from a LTE network to a Wi-Fi network. If the service-context is changed from 'VR-Quality/Best' to '3D-Quality/Best', then the service-binding performs a application- and transmission-binding adaptation together for the best support of 3D-Full-HD service. It is referred to as service-change-adaptation (SCA). To this end, first, in the transmission-binding, the T-adaptor replaces high-rate LTE

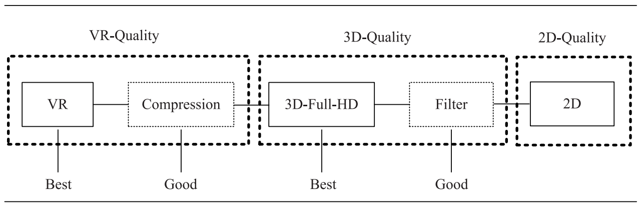


Fig. 8. Example: context-aware virtual reality multimedia service.

transmission objects with low-rate Wi-Fi transmission objects using the method of the configuration meta-object

$$\text{REPLACE (BID, Object, LTE, Wi-Fi)}, \tag{9}$$

and the T-adaptor also replaces LTE protocols with Wi-Fi protocols using the method of the functional meta-object

$$\text{REPLACE (BID, Protocol, LTE, Wi-Fi)}. \tag{10}$$

Second, in the application binding, the A-adaptor replaces VR display objects with 3D-Full-HD display objects using the method of the configuration meta-object

$$\text{REPLACE (BID, Object, VR-Display, 3D-Full-HD)}, \tag{11}$$

and the A-adaptor also replaces VR display control protocols with 3D-Full-HD control protocols using the method of the functional meta-object

$$\text{REPLACE (BID, Protocol, VR-Display, 3D-Full-HD)}. \tag{12}$$

Lastly, if the service context becomes worse than the '2D-Quality' service, then the platform performs cloud resource re-allocation as well as application and transmission binding adaptations. This is referred to as resource-change-adaptation (RCA).

The proposed platform performs the real-time service-binding adaptation as shown in Fig. 9.

① The ECM and UNM monitor their managing devices and check the environment. Whenever there is a change in the environment, they inform the changed information to the CCM.
② The CCM determines a service context based on the collected information.
③ The CCM identifies the adaptation type among SMA, SCA, and RCA. If the adaptation type is SMA, in the following, ④ is performed, if the adaptation type is SCA, ⑤ is performed, and if the adaptation type is RCA, ⑥-⑩ are performed.
④ The CCM and BCM make the real-time transmission-binding adaptation through the T-adaptor.
⑤ The CCM and BCM make the real-time application and transmission-binding adaptations through A-adaptor and T-adaptor, respectively.
⑥ The CCM sends the determined service-context information to the COM.
⑦ The COM receives service and network information from CSM and CNM, respectively.
⑧ The COM determines the new cloud resource allocation.

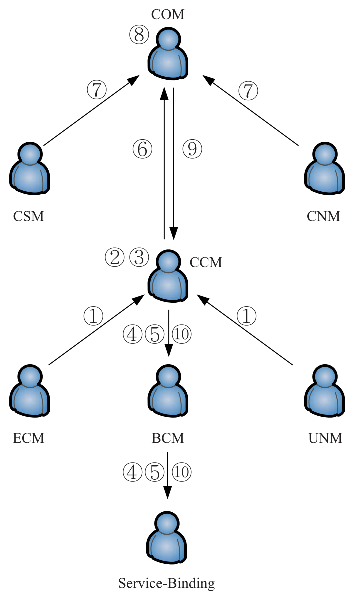


Fig. 9. Service-binding adaptation procedure. Here, binding adaptations are performed according to the change of the service context and achievable user QoS.

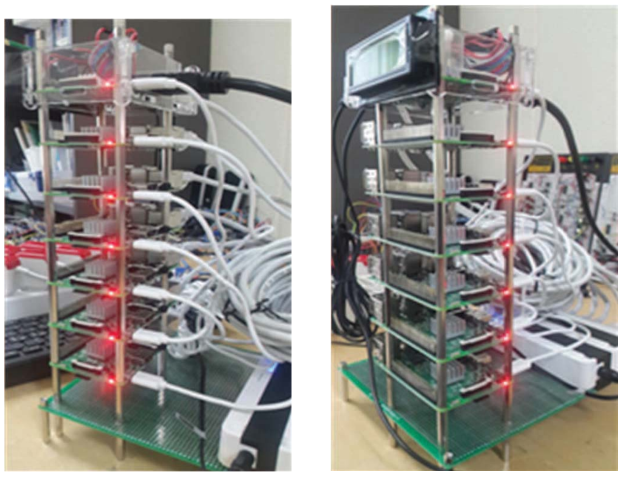


Fig. 10. Prototype of the proposed computing server.

⑨ The COM conveys the resource allocation and adaptation script to the CCM.

⑩ The CCM and BCM control the application and transmission bindings through the A-adaptor and the T-adaptor, respectively.

V. Implementation and Evaluations

We implemented a lightweight prototype server for the proposed computing model. The server consists of 7 sub-modules that are in charge of COM, CSM, CNM, CCM, ECM, UNM, and BCM, respectively. Each sub-module is equipped with a 1.2-GHz 64-bit quad core processor, 1-GB RAM, 802.11n Wireless LAN/Ethernet, and a Linux-compatible OS. They are connected using a 5-GHz IP hub.

To verify the performance of the proposed computing platform, a private testbed network has been deployed, as shown

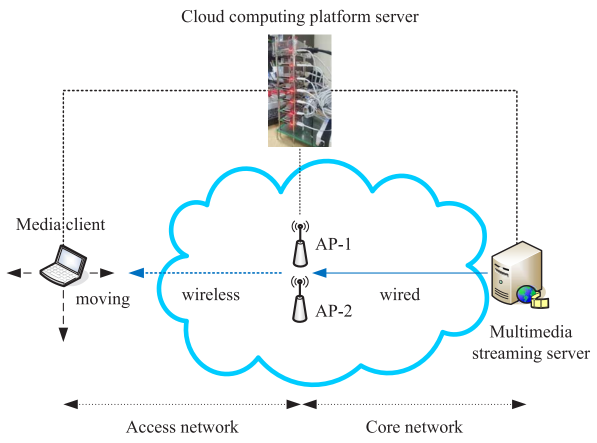


Fig. 11. Testbed network.

TABLE I
SINR to Adaptive Modulation and Coding Mapping Rule

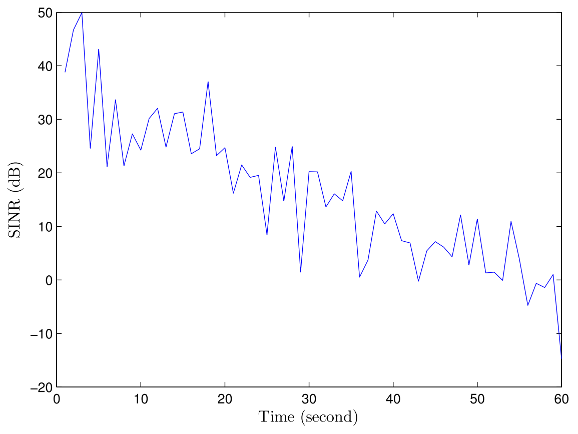
Service Context	SINR[dB]	Modulation	Code rate
Best	19.1 - 100	64-QAM	5/6
Good	17.6 - 19.1	64-QAM	3/4
	15.6 - 17.6	64-QAM	2/3
	13.2 - 15.6	16-QAM	5/6
Normal	12.2 - 13.2	64-QAM	1/2
	12.0 - 12.2	16-QAM	3/4
	10.3 - 12.0	16-QAM	2/3
	7.73 - 10.3	16-QAM	1/2
	6.18 - 7.73	QPSK	5/6
Bad	5.17 - 6.18	QPSK	3/4
	3.71 - 5.17	QPSK	2/3
	1.72 - 3.71	QPSK	1/2
Worst	-100 - 1.72	BPSK	1/4

in Fig. 11. We assume a multimedia streaming service as a context-aware service. A media client receives its streaming data from a media server through an access point (AP). Initially, the client is associated with AP-1. The transmission power and bandwidth of the APs are set to 30 dBm and 5 MHz, respectively. Moreover, the client moves randomly with a certain mobility speed.

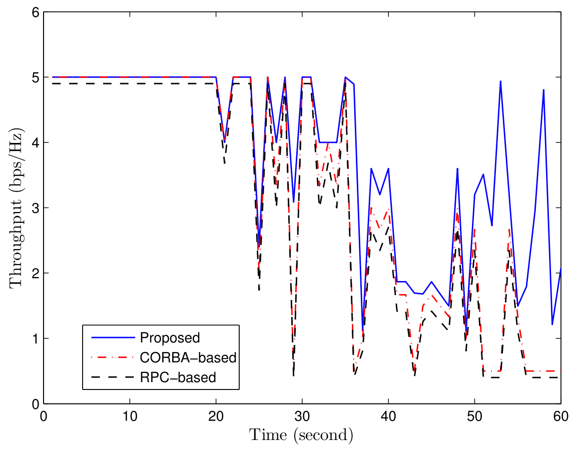
Here, we assume that the service context is determined only by the network status, that is, in terms of the signal-to-interference-plus-noise ratio (SINR), as seen in Table I. The moving client experiences the fluctuation of the SINR, i.e., the service-context change. At this time, for effective data transmission, the proposed computing platform first employs the modulation and coding rate as given in Table I.

Moreover, it also performs service-binding adaptations as follows:

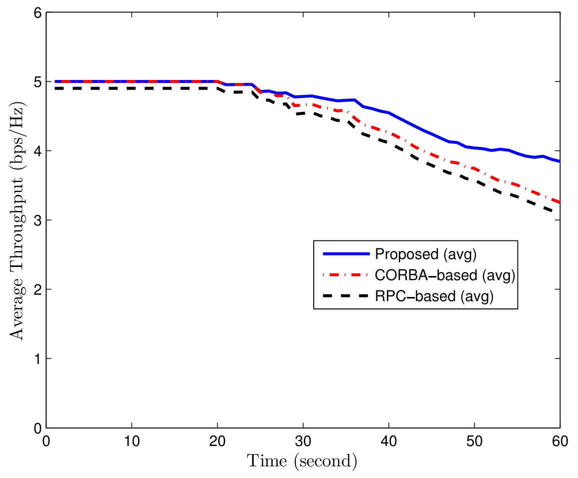
- When the service-context is 'Good,' the service-binding employs compression, caching, and a filter module to maintain its service rate.
- When the service-context is 'Normal,' the service binding employs a multiple antenna-based diversity transmission scheme to maintain its service rate.
- When the service context is 'Bad,' the service binding employs a cooperative transmission scheme such as



(a) An example of channel realization.



(b) Instantaneous throughput



(c) Average throughput

Fig. 12. Achievable instantaneous and average throughput.

opportunistic AP switching between AP-1 and AP-2 to maintain its service rate.

- When the service context is ‘Worst,’ the service binding cannot provide the minimum rate to the users. Therefore, the service binding requests RCA adaptation. That is, the COM reallocates the wireless and computing resources to the service user, which enhances the performance.

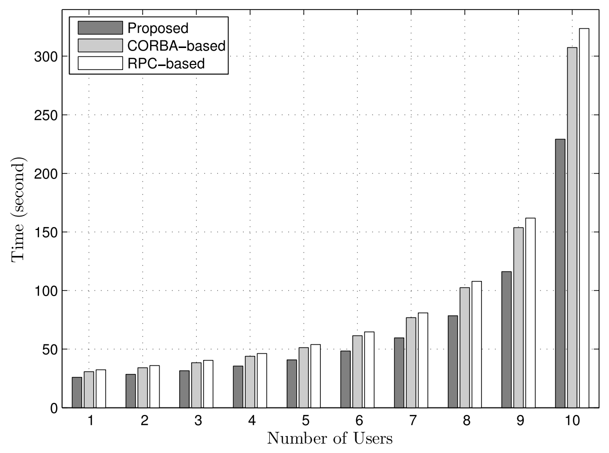


Fig. 13. File downloading time with respect to the number of users.

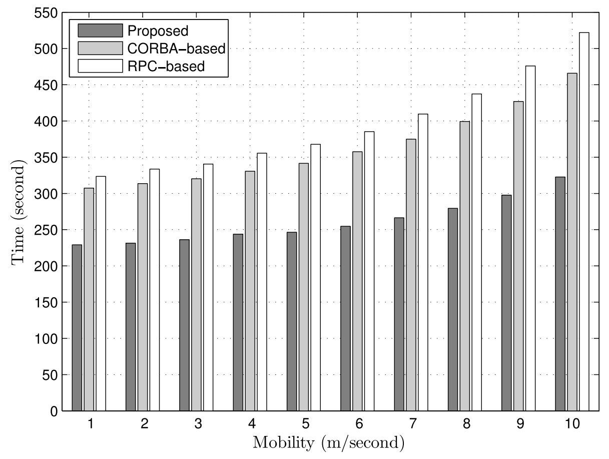


Fig. 14. File download time with respect to mobility.

In this evaluation, we compare the proposed platform with CORBA- and RPC-based platforms that are used in [5]–[14]. Here, we assume that the CORBA- and RPC-based computing platform employs only the modulation and coding rate control given in Table I.

First, Fig. 12 compares instantaneous and average throughput when a user randomly moves at 1 m/s. Here, we assume a channel realization in Fig. 12(a). This implies that the service context changes with time. Under this channel realization, the achievable instantaneous and average throughput are shown in Fig. 12(b) and (c). From the simulation, we can observe that the proposed computing platform has better instantaneous and average throughput. As the network status deteriorates, the throughput of the proposed platform drops. However, it is insignificant compared to that of the legacy CORBA- and RPC-based platform models.

Second, Fig. 13 compares the file download time for an 1-GB streaming file, increasing the number of clients to 10. The clients are uniformly deployed in the testbed of Fig. 11, and they randomly move at 1 m/s. Here, the clients are initially associated with AP-1. From the simulation, we can observe that the proposed computing platform operates more efficiently as the number of users increases. That is, the file downloading time for the proposed platform increases more slowly than the legacy platforms. When 10 clients try to download the file, the proposed computing platform takes 232 seconds, whereas the CORBA- and RPC-based platforms take 307 seconds and 323 seconds, respectively. The proposed computing platform provides approximately 24% and 29% shorter downloading time than CORBA- and RPC-based platforms, respectively.

Third, Fig. 14 compares the file download time for the same 1-GB streaming file, increasing the average moving speed. Here, 10 clients are uniformly deployed and randomly move. From the simulation, we observe that the proposed computing platform operates more efficiently as the mobility speed increases. That is, the file downloading time for the proposed platform increases more slowly than the legacy platforms. Under 10-m/s average moving speed, the proposed platform yields approximately 30% and 38% shorter downloading time than CORBA- and RPC-based platforms, respectively.

VI. Conclusion

This paper presented a hierarchical cloud computing model for context-aware IoT services. It supports nonuniform service binding, real-time service-binding adaptation, and intelligent service-context management. We implemented a lightweight prototype of the proposed computing model and confirmed that the proposed model offers enhanced performance in terms of system throughput as compared with legacy uniform binding based computing models. The proposed computing model can be deployed to all information technology consumer devices and network entities as a key infrastructure. In future work, we will investigate advanced service-binding adaptation, cloud resource control, and mobility management frameworks to enhance the utilization of the proposed computing platform.

References

[1] T.-A. N. Pham, X. Li, G. Cong, and Z. Zhang, "A general recommendation model for heterogeneous networks," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 12, pp. 3140–3153, Dec. 2016.

[2] S. M. R. Islam, M. N. Uddin, and K. S. Kwak, "The IoT: Exciting possibilities for bettering lives: Special application scenarios," *IEEE Consum. Electron. Mag.*, vol. 5, no. 2, pp. 49–57, Apr. 2016.

[3] F. Hao, T. V. Lakshman, S. Mukherjee, and H. Song, "Enhancing dynamic cloud-based services using network virtualization," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 1, pp. 67–74, Jan. 2010.

[4] D. Diaz-Sanchez, F. Almenarez, A. Marin, D. Proserpio, and P. A. Cabarcos, "Media cloud: An open cloud computing middleware for content management," *IEEE Trans. Consum. Electron.*, vol. 57, no. 2, pp. 970–978, May 2011.

[5] T. Xing, D. Huang, S. Ata, and D. Medhi, "MobiCloud: A geo-distributed mobile cloud computing platform," in *Proc. IEEE CNSM*, Las Vegas, NV, USA, 2012, pp. 164–168.

[6] R. Haw, M. G. R. Alarm, and C. S. Hong, "A context-aware content delivery framework for QoS in mobile cloud," in *Proc. IEEE NOMS*, Hsinchu, Taiwan, 2014, pp. 1–6.

[7] G. Guerrero-Contreras, J. L. Garrido, S. Balderas-Díaz, and C. Rodríguez-Domínguez, "A context-aware architecture supporting service availability in mobile cloud computing," *IEEE Trans. Services Comput.*, vol. 10, no. 6, pp. 956–968, Nov./Dec. 2017.

[8] K. Mitra, S. Saguna, C. Ahlund, and D. G. Luleå, "M^2C^2: A mobility management system for mobile cloud computing," in *Proc. IEEE WCNC*, New Orleans, LA, USA, 2015, pp. 1608–1613.

[9] B. Zhou, A. V. Dastjerdi, R. N. Calheiros, S. N. Srirama, and R. Buyya, "mCloud: A context-aware offloading framework for heterogeneous mobile cloud," *IEEE Trans. Services Comput.*, vol. 10, no. 5, pp. 797–810, Sep./Oct. 2017.

[10] Z. Liu *et al.*, "Framework for context-aware computation offloading in mobile cloud computing," in *Proc. IEEE ISPDC*, Fuzhou, China, 2016, pp. 172–177.

[11] Z. Yan, X. Yu, and W. Ding, "Context-aware verifiable cloud computing," *IEEE Access*, vol. 5, pp. 2211–2227, 2017.

[12] O. Dohndorf *et al.*, "Adaptive and reliable binding in ambient service systems," in *Proc. IEEE ETFA*, Toulouse, France, 2011, pp. 1–8.

[13] C.-W. Hsieh, K.-H. Chi, J.-H. Jiang, and C. C. Ho, "Adaptive binding of wireless devices for home automation," *IEEE Wireless Commun.*, vol. 21, no. 5, pp. 62–69, Oct. 2014.

[14] K. Yasaki, H. Ito, and K. Nimura, "Dynamic reconfigurable wireless connection between smartphone and gateway," in *Proc. IEEE COMPSAC*, Taichung, Taiwan, 2015, pp. 228–233.

[15] A. Wollrath, R. Riggs, and J. Waldo, "A distributed object model for the Java system," in *Proc. USENIX*, Toronto, ON, Canada, 1996, pp. 219–232.

[16] (Jun. 1997). *TINA-C: Service Architecture (ver. 5.0), TINA Specification 1.0.* [Online]. Available: http://www.tinac.com/specifications/specifications.htm

[17] "RPC: Remote procedure call protocol," IETF, Fremont, CA, USA, RFC 5531, 2009.

[18] P. Linington, Z. Milosevic, A. Tanaka, and A. Vallecillo, *Building Enterprise Systems With ODP: An Introduction to Open Distributed Processing*, 1st ed. Boca Raton, FL, USA: Chapman & Hall, 2011, pp. 125–181.

[19] (Oct. 2012). *Object Management Group: Common Object Request Broker Architecture, CORBA Specification 3.3*. [Online]. Available: https://www.omg.org/spec/CORBA/3.3/

[20] P. Maes, "Concepts and experiments in computational reflection," in *Proc. ACM OOPSLA*, Orlando, FL, USA, 1987, pp. 147–155.

[21] S. P. Boyd and L. Vandenberghe, *Convex Optimization*, 1st ed. New York, NY, USA: Cambridge Univ. Press, 2004, pp. 215–223.

Tae-Dong Lee received the B.S., M.S., and Ph.D. degrees in electronics engineering from Korea University, Seoul, South Korea in 1996, 2001, and 2006, respectively. Since 2006, he had been researching for Visual Display Division, Samsung Electronics, Suwon, South Korea, for ten years. He joined as a Research Professor with Information and Communication Technology Institute, Korea University for one year. He is currently the Co-Founder for a SW Company, and researching as CTO. His research interests are cloud and distributed computing, peer-to-peer computing IoT system, visual processing, and mobile operating systems.

Byung Moo Lee received the Ph.D. degree in electrical and computer engineering from the University of California at Irvine, Irvine, CA, USA, in 2006. He had ten years of industry experience including research positions with Samsung Electronics, Seoul Research and Development Center, Samsung Advanced Institute of Technology, and Korea Telecom Research and Development Center. He is currently an Assistant Professor with the School of Intelligent Mechatronics Engineering, Sejong University, Seoul, South Korea. During his industry experience, he participated in IEEE 802.16/11, Wi-Fi Alliance, and 3GPP LTE standardizations, and also participated in Mobile VCE and Green Touch Research Consortiums, where he made numerous contributions and filed a number of related patents. His research interests are in the areas of wireless communications, signal processing, and machine learning applications. He served as the Vice Chairman of the Wi-Fi Alliance Display MTG from 2015 to 2016.

Wonjong Noh (S'02–M'08) received the B.S., M.S., and Ph.D. degrees from the Department of Electronics Engineering, Korea University, Seoul, South Korea, in 1998, 2000, and 2005, respectively. From 2005 to 2008, he conducted his Post-Doctoral Research with Purdue University, West Lafayette, IN, USA and the University of California at Irvine, Irvine, CA, USA. Since 2008, he has been researching for Samsung Electronics, Suwon, South Korea, as a Principal Engineer. His current research interests are in wireless communication networks. He is especially interested in studying issues related to performance modeling and analysis, network capacity, cloud computing, and machine learning-based intelligent IoT systems control in 5G/6G wireless communication and networks. He was a recipient of the Samsung Best Paper Award in 2010.