multiFIA – Multi-dimensional Future Internet Architecture

Seung-Joon Seok*, Afaq Muhammad†, Kyungbaek Kim‡, Deokjai Choi§, Youn-Hee Han¶, Woojin Seok‖, Wang-Cheol Song**
*Department of Computer Engineering, Kyungnam University, Changwon, South Korea
† **Department of Computer Engineering, Jeju National University, Jeju, South Korea
‡ §School of Electronics and Computer Engineering, Chonnam National University, Gwangju, South Korea
¶School of Computer Science and Engineering, Korea University of Technology and Education, Cheonan, South Korea
‖Korea Institute of Science and Technology Information, Daejeon, South Korea
Email: *sjseok@kyungnam.ac.kr, †afaq24@gmail.com, ‡kyungbaekkim@jnu.ac.kr, §dchoi@jnu.ac.kr, ¶yhhan@koreatech.ac.kr, ‖wjseok@gmail.com, **kingiron@gmail.com

***Abstract*—Rapid growth in Internet of Things (IoT) has resulted in the need for effective management and provisioning of resources in an IoT-based system. The resource may be an entity (virtual or physical) that constitutes the IoT system, such as network elements, gateways, cloud data stores, and sensors. The dynamic and constantly changing nature of request by a typical user for resource allocation demands end-to-end slices of an IoT system. In order to meet such requests, end-to-end slices need to be integrated, ultimately making it necessary to control and aggregate diverse types of resources from IoT, cloud infrastructures, and network functions. To that end, in this paper, we propose a framework for IoT resources management and provisioning. Our proposed approach enables to automate resource management and provisioning for users across IoT, clouds, and network functions by virtualizing access to underlying IoT resources and utilizing APIs to employ those resources. Furthermore, our proposed approach aims to establish and manage end-to-end slices of various types of resources from distributed and diverse infrastructure. In this paper, we present multiFIA – multi-dimensional Future Internet Architecture – which permits IoT system designers to accommodate multiple user demands by establishing end-to-end slices of the overall IoT network. We outline multiFIA components and layers and also describe our approach to an implementation of multiFIA.**

Keywords*—future Internet; network slicing; orchestration; resource provisioning; service abstraction; network and cloud resources.

I. Introduction

Current and future IoT resources are usually configured and provisioned within networks that are capable of high-speed communication. Such networks comprise of a huge number of IoT devices, which communicate with cloud services in datacenters through the networks. In addition, recent virtualization techniques can be used at different levels, from the cloud services in the datacenters, to the networks connecting the gateways and datacenters, and the gateways integrating sensor and actuator data and control flows [1]. Such virtualization techniques enable us to establish end-to-end slices of cloud resources, network functions, and IoT for a single application in real-time.

One of the major concerns is to manage cloud, network, and IoT resources for applications running upon IoT and their corresponding cloud services, and connected network functions. A huge number of heterogeneous devices distributed at diverse edge sites have to be handled properly because they make use of diverse models and technologies to control their IoT resource information. Hence, resource management must handle *heterogeneity* and *adaptability* of underlying cloud resources, network, and IoT models.

As a motivational example, we consider an emergency response scenario affecting multiple victims, and demanding the coordination of various agencies. Examples could be an emergency situation in a stadium with an ongoing sports event, or a multi-vehicle traffic crash on the highway. The following would be required by such a scenario:

- There would be a need to create crisis response services on the fly, which includes arranging fire engines, arranging emergency vehicles, sending alert messages to hospitals and doctors, and determining best possible routes to be assigned for ambulances.
- Identification and monitoring of victims by means of wearable sensors.
- Based on the information received from emergency responders and the victims, suitable treatment protocols would be served to the victims on the fly while taking them to the hospital.
- If there are traffic jams, the traffic sensors could determine the best possible road/route for the emergency vehicles.

Such a scenario would require creating an on-demand IoT cloud system composed of cloud service at the backend, network functions capabilities in the middle, and IoT entities at the edge. In order to meet the requirements of the user of such an IoT cloud system, an end-to-end

** Corresponding author

resource provisioning which ranges from the edge, through the middle, to the backend with Quality of Service (QoS) has to be guaranteed. Considering such an implementation based on the recent technologies, the user would have to provision each resource – cloud, network functions, and IoT – individually, and then integrate them together manually, which is an error-sensitive and time requiring task [2], [3].

Various applications in mobile-edge computing and cloudlets models [4] have identical requirements, such as sensing applications in smart cities and fitness/performance monitoring applications for athletes in sport events, when on-demand computing and network resources are required. The user would be required to lease different entities, such as network equipment, sensors, and cloud storage separately, and then link them together. Such a rapid and dynamic provisioning of resources is not attainable in recent IoT cloud systems. The two main reasons are; (i) nearly all IoT systems are bound to the actual source/host where some specific functionality exists, and (ii) dynamic provisioning methods and virtualization are examined for cloud infrastructures, network functions, and IoT individually rather than collectively. So with current technologies, establishing and controlling end-to-end slices of resources across various systems is not an easy task.

In order to address the aforementioned issues, in this paper, we present an approach towards network functions, IoT, and cloud resource provisioning, which we call multi-dimensional Future Internet Architecture (multiFIA). Referring to our motivational example, end-to-end resource slicing would need to employ different approaches for different infrastructures. So for network functions linking clouds and IoT, our approach will utilize 5G dynamic network slicing [5] to guarantee the availability of an end-to-end dedicated "slice" of the overall IoT system for particular requirements, and which will not be interfered by other existing network services provided to the users. The dynamic network slicing concept leverages NFV and SDN to create many dedicated end-to-end virtual networks. All end-to-end network slices are created and operated over a common physical infrastructure [5]. In the IoT-Cloud network paradigm, services take the form virtual network slices for the shared physical infrastructure, which can elastically and flexibally utilize network, storage, and compute resources according to dynamic service requirements. To this purpose, an end-to-end network slice orchestrator is proposed in this paper to manage all aspects of network slicing. It enables to manage the entire life cycle management of a slice, from design and creation to operation and optimization.

For cloud resources and IoT, our proposed approach will use on-demand resources provisioning methods and virtualization. Ultimately, we use our existing service abstraction model [6] which is delivered to the orchestrator for provisioning the resources systematically and dynamically.

The rest of the paper is organized as follows. In Section II, we compare our approach with related work in this area. In Section III, we discuss our approach and also present our proposed multiFIA framework and its major components, i.e. service abstraction model, resource provisioning, and network slice orchestrator. The paper concludes in Section IV with suggestions for future work.

II. Related Work

Network Virtualization and Service Abstraction: Fundamentally, both network and applications typically is subject of planning an IoT cloud system. However, any change in the network or in the applications may necessitate revision of the planning. In this case, reconfigurations in several places may be required. This drawback leads to a design in which applications are decoupled from the physical network. This is done by providing an abstract view that is independent of concrete technologies and that is also independent of the physical network topology [7]. In addition, various applications can be isolated from each other through on-demand virtualization of the network and its resources. This enables independent changes in the physical network or in applications. Creating virtual networks for applications also enables to integrate various technologies. Thus, it results in the additional advantage of providing multi-tenancy and heterogeneity in a natural way.

Furthermore, the idea of virtualizing network topologies to improve network utilization has been examined widely by means of initiatives such as OpenDayLight and OpenFlow. More precisely, the authors in [8] describe a system known as "ADVisor". It presents a network virtualization approach, which does not restrict the virtual topologies created to particular subspace of the physical topology. It also allows any two slices to share the same flowspace while isolating both from each other.

Recent network virtualization methods address the following issues by employing the path of packets between end points:

- Logical isolation of distributed end points or applications over the same shared physical infrastructure. It results in a term commonly referred to as 'substrate'.
- Restricting connectivity to parts of the network, and providing security aspects (authentication, encryption, and access control).
- Adjust various Quality of Service (QoS) dimensions such as reliability needs, guarantees, delay, and bandwidth etc. for each virtual network. This is provided by abstracting the methods involved in creating the virtual networks. Thus, applications are not aware of the underlying resource allocation and resource management.

However, solutions today are configured through resource management, network management based on pre-defined rules or QoS features are either not or only basically handled. Examples include separation of different applications such VoIP, network automation and management or multi-tenant networks allowing several companies to

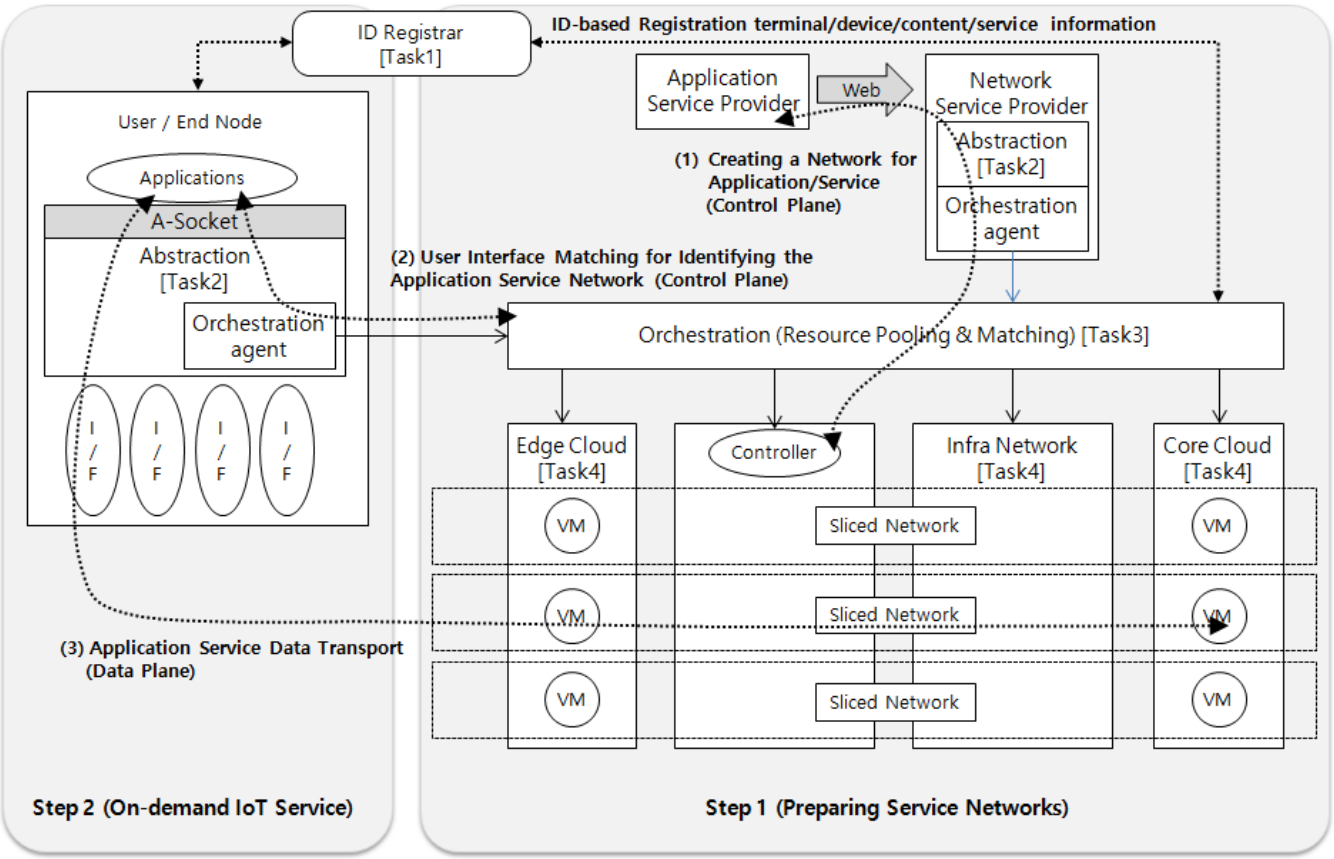

Figure 1: Overall System Architecture.

share a common infrastructure [9], [10], [11]. Our approach of network virtualization and service abstraction model would support a finer-granularity mapping of services to virtual networks.

Network Slicing and Network-Aware Service Composition: The other important aspect in managing IoT cloud systems is network slicing. The prominent principles for designing such systems are virtualization and composition of IoT entities as autonomous units. In this paper, we have taken these principles into consideration for our proposed architecture. These ideas have been influenced by recent work in network slicing; as an example, [5] describes how network slices can be established and used. Other works have shown how network slices can be established and provisioned [12]. A more recent analysis of how network slicing can be employed in 5G networks is presented in [13].

In addition network-aware service composition has been also focused in recent work; one such example is shown in [14], which describes an integrated QoS-aware composition approach that combines network services and application services together. An algorithm of the similar type has been also described in [15]. More precisely, the recent work in IoT area has focused on employing network-aware service composition to IoT services. Algorithms for IoT service composition [16] consider both latency and QoS at the IoT application layer. We will take these algorithms into consideration for implementing our proposed architecture in this paper.

End-to-end Resource Provisioning: In [17], an end-to-end perspective on provisioning, managing, and controlling of cloud and IoT resources is presented. However, it does not examine network functions, although it includes methods and ideas for establishing end-to-end slices of resources and does not cover end-to-end clouds, network functions, and IoT.

III. MultiFIA Framework

We address the aforementioned issues by designing a framework named multi-dimensional Future Internet Architecture (multiFIA). The aim of the framework is to provide major components in order to establish, manage, and adapt end-to-end slices of various types of resources from clouds, network functions, and IoT for specific users based on their requirements.

A. Proposed Architecture

The overall architecture of our proposed framework is illustrated in Figure 1. It includes three fundamental building blocks: (i) *Service Abstraction Model*, (ii) *Resource Provisioning* and (iii) *Slice Orchestration*. We use our existing *Service Abstraction Model* [6] for the orchestrator, with which the orchestrator would understand the requirements of the requested services and provision the resources

for the services. The *Slice Orchestration* is responsible for creating, managing, and adapting end-to-end slices of different types of resources for particular users based on their requirements.

The overall architecture has been separated into two planes, i.e. control plane and data plane. The major components of the control plane include Application Service Provider (ASP), Network Service Provider (NSP), and Orchestrator. In general, ASP provides access to the users at different levels with applications and related services over the Internet. It offers multiple services over the 5G network to the users at individual or at enterprise level. ASP interact with NSP through Web API for the management of services at the application level. NSP consists of further two sub-components named as Abstraction and an Orchestration agent, orchestration agent is for interacting with the Orchestrator. Orchestrator is responsible for dynamic resource pooling and matching, and communication with Edge Cloud, Controller, Infra Network and Core Cloud.

The data plane on the other hand consists of User/End module. It also contains an Orchestration agent for interacting with the orchestrator that is located in the control plane. End user would be able to request a service/application, which will be provided through multiple networks between the User/End node and the Core Cloud based on the requested service/application requirements.

The other important component of this architecture is ID Registrar, which is responsible for storing and keeping track of IDs and the location of the application service available over the network. The functionality of ID Registrar is shared between both control and data planes of the architecture. Initially, all the information related to devices/contents/services based on their ID is registered in the ID Registrar. Based on the location and ID of the application in Core Cloud, the available network resources are registered for the user request. After the registration of resources, the user can request the application/service from the list of available applications. The Orchestrator then creates an end-to-end dedicated slice to meet user requirements after getting the latest update of the resources from the resource pool. The data plane of the overall architecture is depicted in Figure 2.

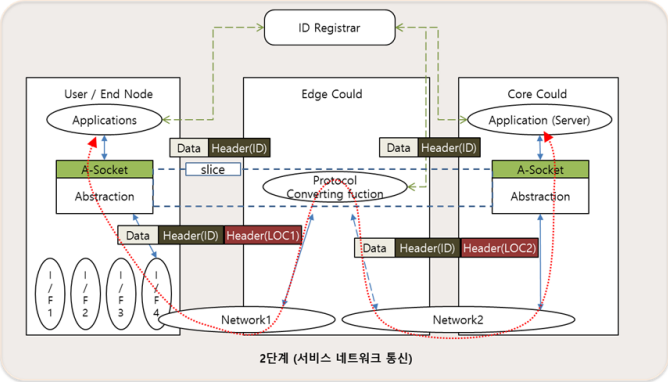

Figure 2: Data Plane of Overall Architecture.

B. Service Abstraction Model

With our existing service abstraction model [6], the orchestrator gets the abstracted requirements of the requested services, and maintains/provisions networks performance to meet the requested service requirements. The orchestrator deals with the dynamic nature of the services by easily changing the networks accordingly. For instance, the abstraction model describes any change occurred in the number of viewers of the video service, whereas the orchestrator manages the required bandwidth for this change. Our service abstraction model abstracts the requested service parameters to support future and legacy network services. The overall concept of the service abstraction model is depicted in Figure 3.

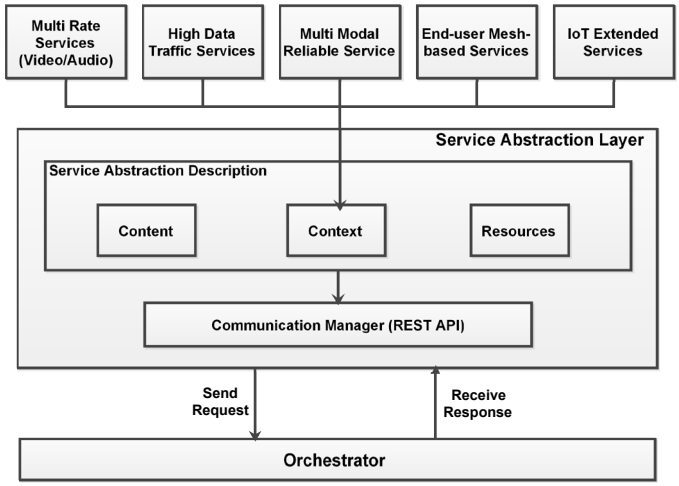

Figure 3: The Service Abstraction Model.

The service requirements are described in the service abstraction description of service abstraction layer. The requirements are classified into three categories, i.e., Resources, Context, and Content. The service-related parameters are provided by Content. They may be standard and resolution of a video, QoS, or audio bit-rate. The user-related parameters are covered by Context. They may be schedule and location of the service, or interest of the user. The requirements of the infrastructural resources are provided by Resource. All these three sets of parameters are converted into XML format, and are parsed at service abstraction description module. After that, the communication manager module transfers these requirements of a service to the orchestrator. Finally, they are processed by the orchestrator, and an appropriate network is generated for the user to access the requested service.

The service parameters are received by the service abstraction layer either from the applications or inputted by the users as shown in Figure 2. Based on these parameters, the service is categorized by the service abstraction layer, which then generates a suitable service abstraction model in XML format and delivers it to the orchestrator.

C. Resource Provisioning

As depicted in Figure 1, the orchestrator also communicates with the Infra Network and SDN controller in order to manage cloud and network infrastructure resources. More precisely, Network Resource Communicator (NRC) module is responsible for managing the network infrastructure resources, whereas Cloud Resource Communicator (CRC) manages storage and compute resources in the cloud according to the user requirements. Infra Network Monitor is a module meant for collecting cloud and network infrastructure-related information and storing it in an internal database server for use by the orchestrator. As illustrated in Figure 4, NRC interacts with the SDN controller in order to get the details of the available resources, and instruct the changes/rules to manage the network. CRC on the other hand exploits APIs provided by the cloud provider in order to control/manage/allocate storage and compute resources based on the application/service requirements.

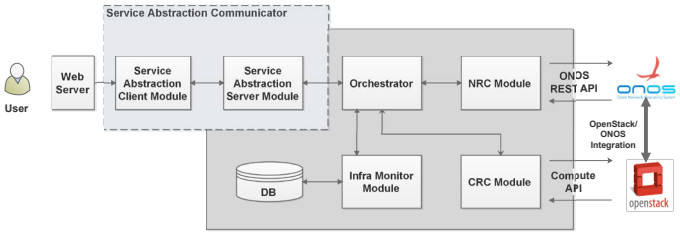

Figure 4: Network and Cloud Resource Communicator Modules.

When a user requests for a service/application, it is forwarded to the web server. The web server forwards this request to service abstraction client module, which abstracts the requirements such as service type and quality, and forwards it to the service abstraction server module. It parses the request message and forwards it to the orchestrator, which then forwards set of links and bandwidth for embedding virtual topology to NRC module, and Node and Service to CRC module. Hence, the orchestrator handles the network requirements by means of SDN Controller and service requirements via Cloud Manager.

D. Slice Management in Orchestrator

The orchestrator in our proposed architecture is responsible for the creation, management, and adaption of end-to-end slices. The slice management part of the orchestrator, as shown in Figure 5, has three main components, i.e. Sliced Network Manager (SNM), Resource Matching Function for Sliced Network (RMFS), and End node Matching functions (ENMF). SNM manages the end-to-end network slices that are created with dedicated or shared resources. Application Service Provider interacts with the SNM through the Network Service Provider and RMFS. Sliced networks are created based on the application/service requirements requested by the user/end node. The functions in RMFS and ENMF are used to adapt an end-to-end slice if there is a change in the application/service requirements that is requested by the user/end node.

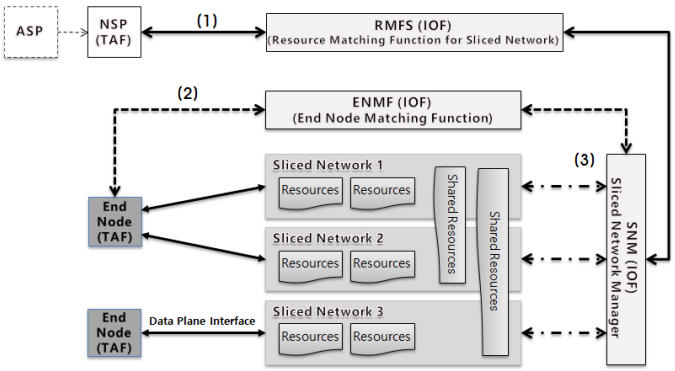

Figure 5: Slice Management in Orchestrator.

1) Slice Orchestration: An end-to-end slice creation is illustrated in Figure 6. In the first step, ASP interacts with control plane's TAF (Abstraction) interface through NSPs (TAF or Abstraction Interface). ASP can have direct access to the RMFS for the matched resources in the resource pool. Resources in the resource pool include edge computing resources, core computing resources, function resources, access network resources, core network resources, and end node resources. Second step in slice orchestration is matching the resources for a slice network. Since these resources are provided to the user through a sliced network, the third step is sliced network generation after interacting with Infra interface component in the fourth step. Network slicing is then performed in the fifth step through edge clouds, core clouds, core networks and access networks. Sliced networks are managed by the SNM in the sixth step, and in finally in the seventh step the sliced network instances are available to access the service/application requested by the user though an end-to-end slice.

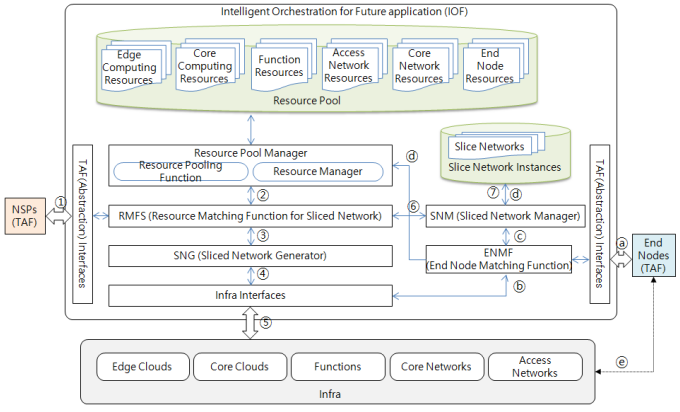

Figure 6: Slice Orchestration.

E. Towards an Implementation

We envision various approaches to implement the multiFIA framework because of the diversity and complexity of the underlying clouds, network function services, and IoT. For a concrete implementation of our framework, we

focus on three major aspects. The first aspect is to develop a thin abstraction layer to arbitrate between complex and dynamic requirements of services, and flexible and diverse network requirements. We aim to improve the service abstraction model for a variety of future services with different dynamic requirements. We also plan to design a framework for the service abstraction layer to deliver the service abstraction model to both of SDN/NFV orchestrator and intended users of requested services.

Currently, to manage network resources, we have used ONOS SDN controller to communicate with orchestrator's NRC module, and to manage cloud resources, we have used OpenStack to communicate with orchestrator's CRC module. We have integrated ONOS SDN controller with OpenStack to manage and monitor the resources more efficiently. The resources are allocated in the form of VMs in order to meet the requirements of the requested applications/services. To this purpose, we implemented a resource allocator module inside the orchestrator, which determines what VMs are best to be allocated. We have addressed this issue by means of an algorithm called maximum RAM minimum CPU utilization (MRMC) algorithm, which first selects the VMs with the maximum amount of RAM, and then out of these selected VMs, the VM with the minimum CPU utilization averaged over the last n quantifications is selected for allocation.

For creation and management of sliced networks, we are working on our proposed slice orchestration framework presented in Section III. We are also investigating a slice-based 5G architecture, combined with many existing paradigms such as SDN, NFV, and cloud computing, that can efficiently manage network slices. We aim to work on the concept of "Network Store in a programmable cloud "allowing for dynamic network slicing [13].

IV. Conclusions and Future Work

In this paper, we have presented an important research issue of end-to-end resource management and provisioning in IoT systems that is crucial for many diverse applications, such as on-demand sensing and disaster responses. Taking into account the heterogeneous and dynamic nature of such systems and leveraging service abstraction models, resource pooling and matching, orchestration, and SDN, we have proposed multiFIA, a multi-dimensional future Internet architecture that offers techniques clouds, network function services, and IoT. Our approach would not only facilitate users to define their requirements from the system, it would also allow system designers to devise resources at different levels of the system to specify the integrated functionality required by the user. We also discussed our steps taken towards implementation to make multiFIA a reality.

In the future, we aim to provide further details about the components of multiFIA architecture. We would also develop a prototype, and evaluate it on diverse practical use cases.

Acknowledgment

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2016R1D1A1B01016322).

References

[1] H.-L. Truong and S. Dustdar, "Principles for engineering iot cloud systems," Cloud Computing, IEEE, vol. 2, no. 2, pp. 68–76, Mar 2015.

[2] H.-L. Truong and N. Narendra, "Sinc – an information-centric approach for end-to-end iot cloud resource provisioning," in International Conference on Cloud Computing Research and Innovation 2016, CloudAsia 2016, May 2016.

[3] D. H. Le, et al. "Hinc harmonizing diverse resource information across iot, network functions and clouds," in Proceedings of the 2014 International Conference on Future Internet of Things and Cloud, ser. FICLOUD '16, 2016, submitted.

[4] M. Satyanarayanan, et al., "The role of cloudlets in hostile environments," IEEE Pervasive Computing, vol. 12, no. 4, pp. 40–49, 2013.

[5] A. Hellemans, "Why IoT Needs 5G," http://spectrum.ieee.org/tech-talk/computing/networks/5g-taking-stock, May 2015, last access: 28 Mar 2016.

[6] Gde Dharma N., et al. "Design of service abstraction model for enhancing network provision in future network." In Network Operations and Management Symposium (APNOMS), 2016 18th Asia-Pacific, pp. 1-4. IEEE, 2016.

[7] Huth, Hans-Peter, and Amine M. Houyou. "Resource-aware virtualization for industrial networks." In 4th international conference on data communication networking (DCNET 2013), Reykjavik, Iceland. 2013.

[8] E. Salvadori, et al. "Generalizing virtual network topologies in openflow-based networks," in Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE. IEEE, 2011, pp. 1–6.

[9] GA Mazhin, et al. "Multi-layer architecture for realization of network virtualization using MPLS technology". ICT Express, 2016.

[10] N Omnes, et al. "A programmable and virtualized network and it infrastructure for the internet of things: How can nfv and sdn help for facing the upcoming challenges." Intelligence in Next Generation Networks (ICIN), 2015 18th International Conference on. IEEE, 2015.

[11] N Bizanis, et al. "SDN and Virtualization Solutions for the Internet of Things: A Survey." IEEE Access 4 (2016): 5591-5606.

[12] S. Gutz, et al. "Splendid isolation: A slice abstraction for SDN," in Proceedings of the first workshop on Hot topics in software defined networks. ACM, 2012, pp. 79–84.

[13] N. Nikaein, E. Schiller, R. Favraud, K. Katsalis, D. Stavropoulos, I. Alyafawi, Z. Zhao, T. Braun, and T. Korakis, "Network store: Exploring slicing in future 5g networks," in Proceedings of the 10th International Workshop on Mobility in the Evolving Internet Architecture. ACM, 2015, pp. 8–13.

[14] J. Huang, G. Liu, Q. Duan, and Y. Yan, "Qos-aware service composition for converged network-cloud service provisioning," in Services Computing (SCC), 2014 IEEE International Conference on. IEEE, 2014, pp. 67–74.

[15] A. Klein, F. Ishikawa, and S. Honiden, "Towards network-aware service composition in the cloud," in Proceedings of the 21st International Conference on World Wide Web, ser. WWW '12. New York, NY, USA: ACM, 2012, pp. 959–968.

[16] U. G. Shehu, G. A. Safdar, and G. Epiphaniou, "Network aware composition for internet of thing services," Transactions on Networks and Communications, vol. 3, no. 1, p. 45, 2015.

[17] H.-L. Truong and S. Dustdar, "Principles for engineering iot cloud systems," Cloud Computing, IEEE, vol. 2, no. 2, pp. 68–76, Mar 2015.