SINC – An Information-Centric Approach for End-to-End IoT Cloud Resource Provisioning

Conference Paper · May 2016

2 authors, including:

Hong-Linh Truong

TU Wien

194 PUBLICATIONS **2,050** CITATIONS

SEE PROFILE

SINC – An Information-Centric Approach for End-to-End IoT Cloud Resource Provisioning

Hong-Linh Truong
Distributed Systems Group, TU Wien, Austria
truong@dsg.tuwien.ac.at

Nanjangud Narendra
Ericsson Research, Bangalore, India
nanjangud.narendra@ericsson.com

Abstract

The explosive growth of Internet of Things (IoT) has raised the need for effective provisioning and management of resources in an IoT-based system. By resource we mean any entity (physical or virtual) that makes up the IoT system, such as sensors, gateways, network elements, and cloud data stores. Typical user requests for resource provisioning demand end-to-end slices of an IoT system, and these requests are also dynamic and constantly changing. Meeting such requests using today's approaches is a time-consuming process since such end-to-end slices need to be hardwired together due to the lack of information-centric frameworks to aggregate and control diverse types of resources from IoT, network functions and cloud infrastructures.

To that end, in this conceptual paper, we propose an information-centric approach towards IoT resources provisioning and management. By virtualizing access to underlying IoT resources and leveraging APIs to manipulate those resources, our approach proposes techniques to enable IoT system designers to automate resource provisioning and management for users across IoT, network functions and clouds. In doing so, our approach focuses on novel concepts to create and manage end-to-end slices of diverse types of resources from different, distributed infrastructures. In this paper we present SINC – Slicing IoT, Network Functions, and Clouds – which enables designers to dynamically create/update end-to-end slices of the overall IoT network in order to simultaneously meet multiple user needs. We describe SINC layers and components and also outline our approach to an implementation of SINC.

1. Introduction

Driven by advances in sensor networking and mobile computing technologies, Internet of Things (IoT) systems are now becoming prevalent in practically every real-world application domain. Some estimates put the number of IoT devices in the world to about 25 billion by the year 2020 [1]. Typically, state-of-the-art IoT systems [2] comprise three major building blocks: IoT including sensors, actuators, and gateways at the edge, networking equipment and services in the middle, and backend cloud data centers. Such building blocks are also fundamental in mobile-edge-computing, fog computing and cloudlets computing models [3], [4], [5]. These need to be integrated in order to provide usable functionality to users, and are typically provisioned and configured within networks that possess high speed communication capabilities. Moreover, such networks would be large in size, comprising thousands of IoT devices (along with other IoT resources such as gateways, routers, or switches), which, through the networks, communicate with cloud services in data centers.

1.1. Motivation

Consider, for example, an emergency response scenario, such as a fire in a crowded theater resulting in several casualties. Such a scenario would require the following:

- Emergency response services would need to be composed on the fly, involving a combination of the following (but not limited to): provisioning ambulances, provisioning fire engines, alerting doctors, alerting hospitals, and identifying optimal roads/routes to be assigned for emergency vehicles.
- Victims would be identified and their conditions monitored either by wearable sensors that they are wearing, or wearable sensors that the emergency responders would attach to their bodies.
- The information from the victims, along with information sent by emergency responders, would inform the appropriate treatment protocols to be administered to the victims on the fly during their transit to the hospital.
- Additionally, traffic sensors could point to the most optimal route that emergency vehicles can take, depending on extent of traffic – although emergency vehicles always have right of way, heavy traffic on certain roads would become difficult to clear quickly enough, especially if there are traffic jams.

In such a scenario, we see the need to establish an on-demand IoT cloud system consisting of several IoT elements at the edge, network function capabilities in the middle, and cloud services at the backend. The user of such an IoT cloud system – the stakeholder who needs to perform responses to the scenario – actually wants to have an end-to-end provisioning of resources spanning from the edge, the middle, to the backend, with guaranteed quality of services, due to the short but crucial and heavy workload. With current technologies, such an implementation would require the user to provision each resource – IoT, network functions, cloud – separately, and then manually stitch them together, which is a time consuming and error-prone activity.

Similar requirements are also seen in different applications in cloudlets and mobile-edge-computing models [6], [5], such as on-demand crowd sensing in smart cities [7] and performance monitoring in sports events [8], when short, high-demand networks and computing resources are needed. Such a dynamic (and rapid) provisioning of resources for the IoT cloud system is currently not possible in current systems, since users would need to rent the various components such as sensors, network equipment and cloud storage separately, and

get them integrated via third parties. This is because (i) most IoT system modeling nowadays is host-centric [9], i.e., tied to the actual host (i.e, source) where particular functionality is available, and (ii) virtualization and dynamic provisioning techniques are investigated for IoT, network functions and cloud infrastructures individually, but not collectively. Thus, creating and managing end-to-end slices of resources spanning different types of systems is difficult, if not impossible, with current technologies.

1.2. Contributions

To address the above-mentioned issues, in this paper, we present an *information-centric* approach [10], [11], [12] towards IoT (sensors and actuators), network functions and cloud resource provisioning, which we call *information-centric slicing and provisioning*. Such an approach would enable users to, for example, provision a combination of the following for emergency response crews: dedicated access to data from specified sensors/wearables, dedicated network bandwidth, and reserved compute and storage space in cloud data centers, for as long as they need to complete providing emergency medical assistance to the fire victims. End-to-end resource slicing will need to utilize different techniques for different infrastructures. Thus, for network functions connecting IoT and clouds, our approach will use 5G network slicing [13], [14] to ensure that an end-to-end dedicated "slice" of the overall IoT system (encompassing sensors & actuators, network infrastructure, as well as cloud resources such as compute & storage) is available for specific requirements (e.g., the emergency response crews), and which will not face interference from other other existing networking services offered to the users. Indeed, the approach in [14] shows how network slices can be created by enabling the declarative specification of customer's network requirements, which are then translated into actual physical resource allocation at level of networking equipment.

For IoT and cloud resources, our approach will use virtualization and on-demand resources provisioning techniques. Ultimately, we do approach end-to-end resource slicing by information-centric resource naming, slicing and routing through the overlay network abstracting and integrating host-centric resource infrastructures.

In this paper, we elaborate our approach by contributing the SINC framework (Slicing IoT, Network functions, and Clouds) and discuss main building blocks for SINC to support the users to create and manage their end-to-end slices of resources.

1.3. Paper Structure

The rest of our paper is organized as follows. In Section 2, we discuss some of the research challenges for provisioning end-to-end slices of resources in IoT systems. In Section 3 we discuss our information-centric approach for IoT system provisioning and also present our proposed SINC framework for the same. Section 4 positions our approach against related work in this area, and the paper concludes in Section 5 with suggestions for future work.

2. Research Challenges

Many application scenarios require techniques to provision resources spanning IoT sites, networks and clouds. In these scenarios, users want to have an end-to-end view of resources so that they can configure and control resources to make sure that resources fulfill the requirements given specific contexts, such as, high performance, high availability, and high throughput. However, considering the diversity and complexity of resources, users do not want to deal with host-centric information (e.g., low-level sensor identifier and network routers).

Dynamic provisioning of IoT devices (sensors and actuators), edge-to-cloud network resources, and cloud data center resources as envisioned in our paper needs to address the following research challenges before it becomes a reality:

- *Modeling Distributed IoT, Network Functions and Cloud Resource Capabilities*: resources are of varied types, such as sensors, actuators, network elements (routers, gateways, and switches), backend cloud resources (compute and storage). In the traditional host-centric networking paradigm, each of these resources is identified individually and modeled in an information model. However, information-centric provisioning requires capability modeling at higher abstraction levels. Referring to our running example, emergency crews may want to determine the vehicular traffic flow through a particular highway, for the purpose of determining if there are any blockages. This would require integrated information from multiple traffic sensors (road sensors, camera sensors, etc.) along the highway, which needs to be presented to the user in summarized form.
- *End-to-End Networks Slicing*: Network slicing is already present in implementations such as OpenFlow [15]. However, the end-to-end slicing of *all* resources across the edge, the middle and the back-end required by implementations such as our running example is still an open problem. This would require special techniques for: modeling the slice, representing it in memory, reserving resources at various levels from different infrastructure-as-a-service in order to provision the slice (e.g., sensors, network bandwidth, virtual machines, storage), and automatically updating/releasing them in response to users' (constantly changing) needs.
- *Resource Composition in Network Slices*: After end-to-end network slicing is implemented, IoT resources have to be "composed", i.e., provisioned together in a particular sequence as per the user's needs. For example, accessing data from a set of sensors would require the following resources to be provisioned, perhaps in order: network access to intermediate gateway, gateway access to "virtual" sensor (that collates data from multiple sensors), "virtual" sensor access to appropriate set of sensors. All

the while, this will have to be executed over virtualized network infrastructure such as OpenFlow, which would require implementing multiple path forwarding actions across the network slice.
- *Re-configuring Composed Resources*: After the above resource composition is completed, issues could still arise. Referring to our running example, a network outage could occur in the network slice, perhaps due to a hardware fault in some network equipment. This would require dynamically reconfiguring and recomposing the network slice by appropriately rerouting network flows away from the faulty equipment.

Addressing these issues will create basic building blocks for us to provision an end-to-end slice of IoT, network functions and cloud resources.

3. SINC Framework

We approach the above-mentioned issues by designing a framework named Slicing IoT, Network Functions and Clouds (SINC). The goal of the framework is to provide fundamental building blocks to enable the creation, management and adaption end-to-end slices of diverse types of resources from IoT, network functions and clouds for particular customers based on their functional and non-functional requirements.

3.1. Conceptual Architecture

The overall conceptual architecture of our framework is as depicted in Fig. 1. We envisage the existence of a user request provisioning component at the *Application* layer (the top layer in Fig. 1), which is essentially a placeholder for a component that processes user requests and invokes the appropriate components in our SINC framework. Typical applications for our framework would be emergency application, on-demand sensing, sport event management, to name just a few. We also envisage the availability of different types of *IoT Networks*, *Network Function Services* and *Clouds* infrastructures, shown at the bottom of Fig. 1. These types of infrastructures can be simple or complex, e.g., an IoT network might just consist of a set of public sensors that can access directly or a set of pay-per-use sensors, actuators and gateways offered under the service model [16]. Similarly, Network Function Services might be provided by an on-demand micro data center [17], [18] connected to telco networks or complex network function virtualization services offered by telcos in the fog computing model [19]. These infrastructures provide different types of resources and these resources will be accessible and controllable through well-defined APIs.

The SINC framework will include three layers: (i) *API Integration and Communication*, (ii) *Naming, Slicing and Routing*, and (iii) *Resource Management, Configuration, and Adaptation*. We devise *API Integration and Communication* for integrating and communicating existing infrastructures of sensors, edge-to-cloud and cloud resources, each from different vendors, whereas *Resource Management, Configuration and Adaptation* is meant for specific operations on provisioning end-to-end slices for the applications. The *Naming, Slicing and Routing* layer comprises components for information-centric provisioning partially derived from [20]. The Naming component is responsible for assigning identities to virtual resources, such as virtual sensors or virtual networks. It also maintains mappings to the actual underlying physical resources. The Slicing component performs the actual task of virtualizing the network resources as well as defining and provisioning the network slice to meet the user needs. During actual operation, it invokes the Routing component to route data through the network slice components. The actual routing and transport is performed by *API Integration and Communication* components, which interface with network layer components such as gateways, routers and virtual sensors. In the next subsections, we will elaborate these layers and components and outline our effort towards an implementation of SINC.

3.2. API Integration and Resource Management

3.2.1. API Integration. The first step in our framework is the capability to integrate different APIs from existing infrastructures so that we can access diverse resource capabilities and manage these capabilities. In this sense, in the *API Integration and Communication* layer, *API Integration* will include a collection of APIs for accessing and managing resources so that we can negotiate and procure resource capabilities and invoke and monitor them. It is important to note that there are many APIs from different infrastructures (and their providers), however, we do not need to consider all of them. In the SINC framework, only needed APIs will be included. In this view, we could leverage API management services [21], [22], [23] within a SINC implementation to support the integration with existing infrastructures.

3.2.2. Communication Middleware. Since we will have requests and commands from high-level components to concrete resources in different, distributed infrastructures, *Communication Middleware* is needed to support different types of communication. This middleware works together with API integration to invoke suitable APIs to transfer (meta)data about resources and commands. However, it has to deal with different types of interactions. Therefore, this requires us to develop a loosely coupled communication middleware based on matured technologies for inter-cloud and IoT communication, such as AMQP [24], CoAP [25], and MQTT [26]. In SINC, the *Communication Middleware* is for managing resources. Therefore, we envisage different middleware for transferring actual data produced by sensors to cloud services.

3.2.3. Resource Grid. Through *API Integration* and *Communication Middleware* we could have an instant view on available resources from existing infrastructures. Such available resources might be just a subset of existing resources in these infrastructures (since the providers might not make all resources available for the application). We need *Resource Grid*

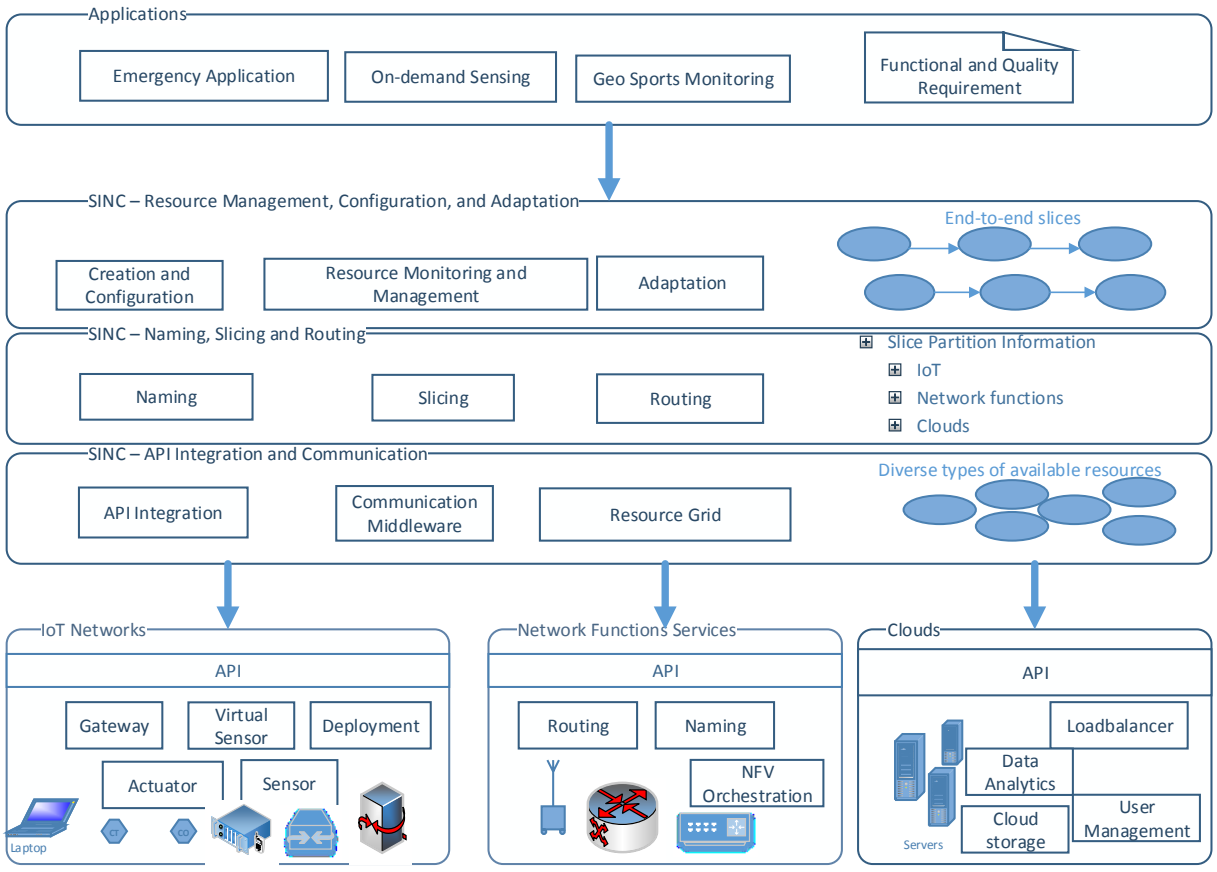

Fig. 1. Slicing IoT, Network Functions, and Clouds (SINC) – Conceptual Architecture

to provide a map to underlying resources that are available to applications using the end-to-end slice approach. *Resource Grid* presents resources to components in the *Naming, Slicing and Routing* layer by harmonizing diverse types of information representations. Depending on the implementation, *Resource Grid* may just mirror all available resources through APIs – so it may just perform the transformation and harmonization of information through API calls – or it may replicate information about resources in SINC by querying, pulling and caching information from existing infrastructures. Overall, *Resource Grid* just keeps a portion of resources that individual providers make available for end-to-end slicing.

3.3. Naming, Slicing and Routing

3.3.1. Naming. As enunciated in [20], the information-centric approach enables the use of application-specific names. However, this naming scheme should be broad enough so that it can be understood by users spanning various application domains. The naming hierarchy should also be explicitly specified, e.g., $/city\_name/bangalore/street\_name/cunningham\_road/location/20/corner/east/sensor\_type/video/sensorID/1$, may refer to video camera sensor #1 stationed 20 feet from the beginning of Cunningham Road in the city of Bangalore. Multiple such video camera sensors could be positioned along Cunningham Road, and would form part of a virtual sensor with name $/city\_name/bangalore/street\_name/cunningham\_road/sensor\_type/virtual/sensorID/A$.

In our approach, we will utilize this naming strategy for all resources in the IoT system, including IoT devices (sensors & actuators), network services & equipment, and cloud services. Indeed, if many (cloud) computing and/or storage functions are moved to the edge as described in [27], this naming strategy would be needed. Using this naming, we can easily describe and identify the data, computing and network capabilities of diverse types of resources. However, a challenge is that the composition of these resources for an end-to-end provisioning would also rely on many other types of information, such as geographical distribution, connection performance, availability, and security and privacy attributes. In this case, naming for resources gathered from underlying systems will also link to such (meta and monitoring information) for determining optimal compositions.

3.3.2. End to End Slicing. : As stated earlier, and as depicted in Fig. 1, network slicing in our case encompasses the *entire* IoT system. Typical proposed network slicing approaches as in [14] limit themselves to only the core network equipment infrastructure, with some integration to backend cloud infrastructure. Indeed, [14] does present a prototype implementation on Openstack [28]. However, our approach would need to provide a higher-level service orchestration mechanism through the utilization of specific mechanisms for *partitions* of IoT, network functions, and clouds as well as through the new development of such combined mechanism. In our approach, slicing techniques for IoT and cloud partitions would focus on mechanisms to provision guaranteeing data and computing capabilities (e.g., dedicated sensors, gateways, and virtual machines). Slicing techniques for network function partitions will focus on network capabilities between (i) existing nodes of IoT subnetworks connecting to access nodes of the network functions in the middle and (ii)existing nodes of the network functions in the middle and the access node of the cloud data centers. As a result of slicing techniques, resources in end-to-end slices can be described, deployed and configured by techniques such as TOSCA [29], and service chaining [30], the latter for instantiating and interconnecting virtual network functions to form a service.

3.3.3. Routing. Typical routing infrastructure in ICN comprises the following components [20], [10]: (i) a Content Store (CS) for temporary caching of incoming Data packets; (ii) a routing table called Forwarding Information Base (FIB) used to guide the Interests towards Data; and (iii) a Pending Interest Table (PIT), which keeps track of the forwarded Interest(s) that are not yet satisfied with a returned data packet. Our goal is to utilize these concepts for routing data and requests among resources provisioned for slices. This means that we need to carefully investigate how to implement these concepts at the level of (virtualized) resources in slices. Essentially, as we have different partitions of resources identified by the *Naming* component that can be accessed and controlled through the *Communication Middleware*, all of these concepts will be implemented atop an overlay network of resources. Hence, routing can be two-way, i.e., the FIB can also be used to route commands to sensors and actuators, for implementing actuation commands, e.g., changing the data rate of a sensor, or changing the unit of a body temperature sensor from Centrigrade to Fahrenheit. The key issue in the design here is that the routing has to be accomplished within certain limits (based on QoS considerations), and has to be accomplished reliably. At the higher-level, this, however, depends on the capabilities of the *Communication Middleware* which carry requests and data. At the lower level, this depends on the properties of the selected end-to-end network slice, in particular, its bandwidth capacity, its topology, and the latency of data transmission within the slice. Since multiple slices may have to be implemented simultaneously based on user needs, this becomes a dynamic optimization problem that needs to be solved within the constraints of the selected network slice.

3.4. Managing End to End Slices

As we have seen above, the end to end network slice, encompassing sensors, network functions and cloud resources, would become the most crucial entity of our IoT network. Hence it needs to be managed effectively. This involves the following: creation and deployment of the slice, distributed monitoring and management of the resources that make up the slice, and adaptation of the slice in response to changing user requests and/or changing network environment.

3.4.1. Creation and Deployment. The process of creation of a slice would start with the user specifying their requirements for the following: data needed from sensors, minimum network bandwidth needed, backend compute and storage needed. For example, the user may ask for data from a set of traffic sensors every 5 minutes, at least network bandwidth of 16 MB per second, backend compute capacity of at least 16 GB of memory, and backed storage of at least 1 TB. The user may also request this functionality for 4 hours, for e.g., for the time it would take to complete evacuating the victims in our running example, providing them first aid en route to hospitals, and monitoring their condition in the hospitals until they are declared stable and out of danger.

There are also scenarios where such user requests may be placed well in advance [14], e.g., a rock concert or football match, whose organizers may request dedicated network bandwidth so that attendees can obtain live information feeds on their smartphones. Hence in a situation such as our emergency response scenario, if network bandwidth or compute capability happens to be insufficient, then lower priority user requests (rock concert) may have to be de-provisioned in favor of a higher priority request (emergency response).

Fulfilling such a request would require the development of a dedicated service that can integrate with NFV orchestrators, virtual sensors, gateways, cloud APIs and SDN controllers. Such a service also has to deal with different resource provisioning models imposed by underlying infrastructures. For example, it is possible that underlying infrastructures offer pay-per-use resources with or without negotiation as well as opportunistic resources through spot sensors and machine instances. Similarly, deployment for slices would need to address different deployment models for IoT and clouds, which are quite different.

3.4.2. Distributed Resource Monitoring and Management. Once the slices have been created, the next step is to monitor the resources that make up the slice and ensuring that they are performing as intended. There are two ways in which this monitoring can be implemented - pull and push. In the former approach, the monitoring application at the resource management layer of Fig. 1 directly polls the resources – sensors, network elements, cloud resources – via the API integration layer. The polling frequencies, as well as extent of status data to be sent by the IoT resources, are detailed

design choices that need to be made depending on the slice in question.

Naturally, we foresee the need to integrate different monitoring systems for IoT, network functions and clouds [31], [32], [33]. From the monitoring data, using information from naming, we will correlate monitoring information from different sources to provide an end-to-end view on performance of slices. In addition to solving the complexity of monitoring metric spaces (IoT, network, cloud metrics), we also have to deal with the complexity of measurement granularity of metrics.

3.4.3. Runtime Slice Adaptation. As is the case with any complex software-defined system, issues will constantly arise during execution, that demand runtime adaptation. In our case, we can identify two key issues: changes in user requirements, and changes in the underlying infrastructures. These can be handled in either of two ways: either the change can be regarded as a new requirement, for which a fresh slice is provisioned (as described above); or the slice can be reconfigured at runtime. In particular, changes in the underlying infrastructures – such as unavailability of network equipment, breakdown in underlying network or sensor hardware, etc. – would require runtime slice reconfiguration to ensure continuity.

A running slice has many points, existing in different slice partitions, at which reconfiguration can be implemented, such as: provisioning additional compute and/or storage at the cloud backend; provisioning additional network bandwidth; rerouting network traffic using SDN controllers to ensure adherence to customer needs in the presence of network equipment failure, redefining virtual sensor data in the presence of failure of sensors. The last example is particularly interesting; consider, for example, a set of load sensors on a bridge providing load data. If a subset of them fail, the virtual sensor that collates this data needs to perform suitable data interpolation in order to compensate for the lost sensor data.

Besides performance, reliability and availability adaptation for slices, an important aspect of monitoring and management of slices is to deal with uncertainties [34] which are not well studied in an integrated virtualized IoT, network functions and cloud-based environment. On the one hand, adaptation techniques have to deal with monitoring data and actuation uncertainties [35]. On the other hand, they must deal with the uncertainties in resource behavior. To deal with this problem, further uncertainties inherent in slices must be studied.

3.5. Towards an Implementation

We envisage different approaches to the implementation of the SINC conceptual framework due to the complexity and diversity of the underlying IoT, network function services and clouds[1]. We are currently looking into three key aspects for our concrete implementation. The first aspect is to integrate the diverse IoT infrastructures and providers at sensor, network function and cloud levels. We currently make the assumption that all providers would expose common interfaces and protocols via well-understood technologies, such as REST API for service interfaces, MQTT and AMQP for communication, and dockers and OpenStack for virtualization, and TOSCA/HOT for deployment and configuration. To that end, we are currently working on a high-level distributed IoT resource information model and communication middleware, with initial emphasis on IoT resource monitoring and management [36].

For naming and routing we are, at an early stage, investigating integration of techniques from [20]; in particular, we will be leveraging the concept of application-specific naming schemes in the Naming component, and name-based routing in the Routing component.

For creation and management, we are investigating TOSCA [29] for describing resources in a slice and existing deployment and configuration tools slice deployment[37], [38]. We will also be leveraging our ongoing work [36] and extending our monitoring work [33] for slice monitoring, and we will be enhancing it for slice adaptation. We are also investigating Calvin [39] as a modeling language for representing slices as compositions of atomic services.

4. Related Work

Information-Centric Networking for IoT: The concept of virtualizing network topologies in order to enhance efficiency of network utilization has been researched extensively via initiatives such as OpenFlow and OpenDayLight. In particular, in [15], the authors presented an innovative system called "ADVisor". This system, which extends an earlier network virtualization approach, provides the following features: (i) the virtual topologies created are not restricted to specific subsets of the physical topology, and (ii) enables any two slices to share the same flowspace while preserving isolation between them. Works such as [15] have paved the way for taking a relook at the overall network from an information-centric viewpoint, as presnted earlier in our paper. These are collectively known as Information-Centric Networking or ICN.

Many ICN applications to IoT have been presented. For example, the citation [11] presents a vision for a global, all-encompassing IoT realized through an integrating architecture relying on information and its identifiers/names. They also argue that ICN is the ideal candidate architecture for realizing IoT. They also present several research challenges that need to be overcome before this vision can be realized, viz., naming, efficient and contextual information retrieval, trust modeling, privacy and access control, and information forwarding. A more detailed elaboration of ICN architectures for IoT is presented in [20], which presents a 3-tier architecture comprising application, management+data and thing layers. We have leveraged a few ideas from [20] in our paper.

A case study implementation of ICN for IoT in a smart home scenario has been presented in [10]. In that paper, the authors present a realization of their architecture from [20], with

1. Up-to-date information about SINC implementation is available at http://sincconcept.github.io.

emphasis on naming scheme, service model, and strategies for multi-party communications. They also present a preliminary evaluation to provide a rough estimate of ICN based packet communications.

An excellent exposition of ICN and its applicability to IoT is presented in [9]. Apart from discussing the well-known research challenges of ICN, viz., naming, distributed caching and decoupling senders and receivers of data, that paper also discussing specific design choices to be made when ICN is to be used to build IoT systems. A few key choices are: coexistence with existing Internet protocols, using ICN network only for directly addressable data, mandating that IoT uses only immutable data objects, naming data streams so that they can be traced and integrated (e.g., video streams), and representing time-related data streams. We will be considering the design recommendations from [9] for our future papers in this area.

Network Slicing and Network-Aware Service Composition: The other side of the coin in managing IoT systems is network slicing. In our earlier work, we had proposed several principles for engineering cloud-based IoT systems, prominent among them being recommendations for virtualization and composition of IoT components as self-contained units. In this paper, we have incorporated these principles into our proposed conceptual architecture in Fig. 1. These ideas have been leveraged from existing work in network slicing; one such example is described in [14], which shows how network slices can be created and used. Other works have described in more detail how slices can be created and provisioned [40], [41]. A more recent exposition of how network slicing can be implemented in 5G networks is detailed in [42].

Recent work has also focused on network-aware service composition [43], which presents an integrated QoS-aware composition method that integrates application services and network services together. A similar algorithm was also presented earlier in [44]. Extending to the IoT area, recent work has focused on applying network service composition to IoT services. In particular, the citation [45] presents algorithms for IoT service composition that consider not only quality of service (QoS) but also network latency at the IoT application layer. We will be considering these algorithms for implementing our conceptual architecture presented in Fig. 1.

End-to-end Resource Provisioning: In [46], an end-to-end view on controlling, management and provisioning of IoT and cloud resources is discussed. However, it does not cover network functions, although it does outline concepts and techniques for creating end-to-end slices of resources. The work in [47] discusses the need to have end-to-end network service provisioning across different metro and core network domains. However, it does not focus on end-to-end IoT, network functions and clouds.

5. Conclusions and Future Work

In this paper, we have considered the crucial research issue of end-to-end resource provisioning and management in IoT cloud systems that is of paramount importance for several applications, such as emergency responses and on-demand sensing. Considering the dynamism and heterogeneity inherent in such systems and leveraging virtualization, API management and service models, we have proposed SINC, an information-centric approach and conceptual architecture that abstracts from lower-level details to provide techniques for slicing IoT, network function services and clouds. Such an approach would not only enable users to declaratively specify their needs from the system, it would also enable designers to compose resources at various levels of the system in order to provide the integrated functionality that users would require. We also discussed the research challenges that would arise when trying to make SINC a reality.

Future work would involve detailing out the components of SINC conceptual architecture, developing a proof of concept prototype, and evaluating the prototype on various realistic usage scenarios.

Acknowledgments

Nanjangud Narendra wishes to thank his colleagues at Ericsson Research Bangalore for their feedback. This work is partially supported by the EU H2020 U-Test project, under grant No. 645463, w.r.t. modeling and provisioning cyber-physical systems for uncertainties testing.

References

[1] http://www.gartner.com/newsroom/id/2905717, last access: 28 Mar 2016.

[2] A. Botta, W. de Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things," *Future Gener. Comput. Syst.*, vol. 56, no. C, pp. 684–700, Mar. 2016. [Online]. Available: http://dx.doi.org/10.1016/j.future.2015.09.021

[3] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, Oct. 2009. [Online]. Available: http://dx.doi.org/10.1109/MPRV.2009.82

[4] M. R. Rahimi, J. Ren, C. H. Liu, A. V. Vasilakos, and N. Venkatasubramanian, "Mobile cloud computing: A survey, state of art and future directions," *Mob. Netw. Appl.*, vol. 19, no. 2, pp. 133–143, Apr. 2014. [Online]. Available: http://dx.doi.org/10.1007/s11036-013-0477-4

[5] "Mobile-Edge Computing – Introductory Technical White Paper," https://portal.etsi.org/portals/0/tbpages/mec/docs/mobile-edge_computing_-_introductory_technical_white_paper_v1%2018-09-14.pdf, September 2014.

[6] M. Satyanarayanan, G. A. Lewis, E. J. Morris, S. Simanta, J. Boleng, and K. Ha, "The role of cloudlets in hostile environments," *IEEE Pervasive Computing*, vol. 12, no. 4, pp. 40–49, 2013. [Online]. Available: http://dx.doi.org/10.1109/MPRV.2013.77

[7] D. AG, "Crowd analytics archives," http://www.dfrc.ch/tag/crowd-analytics/, last access: 30 Mar 2016.

[8] "U-test geo sports case study."

[9] A. Lindgren, F. B. Abdesslem, B. Ahlgren, O. Schelen, and A. Malik, "Applicability and tradeoffs of information-centric networking for efficient iot," Internet-Draft, Tech. Rep., 2015.

[10] M. Amadeo, C. Campolo, A. Iera, and A. Molinaro, "Information centric networking in iot scenarios: The case of a smart home," in *Communications (ICC), 2015 IEEE International Conference on*. IEEE, 2015, pp. 648–653.

[11] N. Fotiou and G. C. Polyzos, "Realizing the internet of things using information-centric networking," in *Heterogeneous Networking for Quality, Reliability, Security and Robustness (QShine), 2014 10th International Conference on*. IEEE, 2014, pp. 193–194.

[12] E. Baccelli, C. Mehlis, O. Hahm, T. C. Schmidt, and M. Wählisch, "Information centric networking in the iot: experiments with NDN in the wild," in *1st International Conference on Information-Centric Networking, ICN'14, Paris, France, September 24-26, 2014*, G. Carofiglio, L. Muscariello, L. Zhang, D. Kutscher, and L. Muscariello, Eds. ACM, 2014, pp. 77–86. [Online]. Available: http://doi.acm.org/10.1145/2660129.2660144

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

[14] R. Inam, A. Karapantelakis, K. Vandikas, L. Mokrushin, A. V. Feljan, and E. Fersman, "Towards automated service-oriented lifecycle management for 5g networks," in *Emerging Technologies & Factory Automation (ETFA), 2015 IEEE 20th Conference on*. IEEE, 2015, pp. 1–8.

[15] E. Salvadori, R. Doriguzzi Corin, A. Broglio, and M. Gerola, "Generalizing virtual network topologies in openflow-based networks," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*. IEEE, 2011, pp. 1–6.

[16] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Sensing as a service model for smart cities supported by internet of things," *Trans. Emerg. Telecommun. Technol.*, vol. 25, no. 1, pp. 81–93, Jan. 2014. [Online]. Available: http://dx.doi.org/10.1002/ett.2704

[17] N. Laoutaris, P. Rodriguez, and L. Massoulie, "Echos: Edge capacity hosting overlays of nano data centers," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 1, pp. 51–54, Jan. 2008. [Online]. Available: http://doi.acm.org/10.1145/1341431.1341442

[18] S. Echeverría, J. Root, B. Bradshaw, and G. A. Lewis, "On-demand VM provisioning for cloudlet-based cyber-foraging in resource-constrained environments," in *6th International Conference on Mobile Computing, Applications and Services, MobiCASE 2014, Austin, TX, USA, November 6-7, 2014*, C. Julien, N. D. Lane, and S. Mishra, Eds. IEEE, 2014, pp. 116–124. [Online]. Available: http://dx.doi.org/10.4108/icst.mobicase.2014.257768

[19] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. ACM, 2012, pp. 13–16.

[20] M. Amadeo, C. Campolo, A. Iera, and A. Molinaro, "Named data networking for iot: an architectural perspective," in *Networks and Communications (EuCNC), 2014 European Conference on*. IEEE, 2014, pp. 1–5.

[21] M. Vukovic, "Internet programmable iot: On the role of apis in iot: The internet of things (ubiquity symposium)," *Ubiquity*, vol. 2015, no. November, pp. 3:1–3:10, Nov. 2015. [Online]. Available: http://doi.acm.org/10.1145/2822873

[22] Y. Cho, J. Choi, and J. Choi, "An integrated management system of virtual resources based on virtualization api and data distribution service," in *Proceedings of the 2013 ACM Cloud and Autonomic Computing Conference*, ser. CAC '13. New York, NY, USA: ACM, 2013, pp. 26:1–26:7. [Online]. Available: http://doi.acm.org/10.1145/2494621.2494648

[23] M. Vukovic, J. Laredo, V. Muthusamy, A. Slominski, R. Vaculin, W. Tan, V. Naik, I. Silva-Lepe, A. Kumar, B. Srivastava, and J. W. Branch, "Riding and thriving on the api hype cycle," *Commun. ACM*, vol. 59, no. 3, pp. 35–37, Feb. 2016. [Online]. Available: http://doi.acm.org/10.1145/2816812

[24] "Advanced message queuing protocol," https://www.amqp.org/, last access: 30 Mar 2016.

[25] "CoAP: RFC 7252 Constrained Application Protocol," http://coap.technology/, last access: 30 Mar 2016.

[26] "Message queue telemetry transport," http://mqtt.org/, last access: 30 Mar 2016.

[27] A. M. Haubenwaller and K. Vandikas, "Computations on the edge in the internet of things," *Procedia Computer Science*, vol. 52, pp. 29–34, 2015.

[28] "Openstack," www.openstack.org, last access: 30 Mar 2016.

[29] "Topology and orchestration specification for cloud applications (tosca)," https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=tosca, last access: 30 Mar 2016.

[30] "Network functions virtualisation update white pap er," https://portal.etsi.org/NFV/NFV_White_Paper2.pdf, October 2013.

[31] D. Trihinas, G. Pallis, and M. D. Dikaiakos, "Adam: An adaptive monitoring framework for sampling and filtering on iot devices," in *2015 IEEE International Conference on Big Data, Big Data 2015, Santa Clara, CA, USA, October 29 - November 1, 2015*. IEEE, 2015, pp. 717–726. [Online]. Available: http://dx.doi.org/10.1109/BigData.2015.7363816

[32] G. Aceto, A. Botta, W. De Donato, and A. Pescapè, "Survey cloud monitoring: A survey," *Comput. Netw.*, vol. 57, no. 9, pp. 2093–2115, Jun. 2013. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2013.04.001

[33] D. Moldovan, G. Copil, H. L. Truong, and S. Dustdar, "MELA: elasticity analytics for cloud services," *IJBDI*, vol. 2, no. 1, pp. 45–62, 2015. [Online]. Available: http://dx.doi.org/10.1504/IJBDI.2015.067569

[34] S. Ali and T. Yue, "U-test: Evolving, modelling and testing realistic uncertain behaviours of cyber-physical systems," in *8th IEEE International Conference on Software Testing, Verification and Validation, ICST 2015, Graz, Austria, April 13-17, 2015*. IEEE, 2015, pp. 1–2. [Online]. Available: http://dx.doi.org/10.1109/ICST.2015.7102637

[35] S. Nastic, G. Copil, H. L. Truong, and S. Dustdar, "Governing elastic iot cloud systems under uncertainty," in *7th IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2015, Vancouver, BC, Canada, November 30 - Dec. 3, 2015*. IEEE, 2015, pp. 131–138. [Online]. Available: http://dx.doi.org/10.1109/CloudCom.2015.77

[36] D.-H. Le, N. Narendra, and H.-L. Truong, "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.

[37] D.-H. Le, H.-L. Truong, G. Copil, S. Nastic, and S. Dustdar, "SALSA: A Framework for Dynamic Configuration of Cloud Services," in *6th International Conference on Cloud Computing Technology and Science*, Dec 2014.

[38] M. Vögler, J. M. Schleicher, C. Inzinger, and S. Dustdar, "A scalable framework for provisioning large-scale iot deployments," *ACM Trans. Internet Technol.*, vol. 16, no. 2, pp. 11:1–11:20, Mar. 2016. [Online]. Available: http://doi.acm.org/10.1145/2850416

[39] P. Persson and O. Angelsmark, "Calvin merging cloud and iot," *Procedia Computer Science*, vol. 52, pp. 210 – 217, 2015, the 6th International Conference on Ambient Systems, Networks and Technologies (ANT-2015), the 5th International Conference on Sustainable Energy Information Technology (SEIT-2015). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050915008595

[40] S. Gutz, A. Story, C. Schlesinger, and N. Foster, "Splendid isolation: A slice abstraction for software-defined networks," in *Proceedings of the first workshop on Hot topics in software defined networks*. ACM, 2012, pp. 79–84.

[41] Y. Yiakoumis, K.-K. Yap, S. Katti, G. Parulkar, and N. McKeown, "Slicing home networks," in *Proceedings of the 2nd ACM SIGCOMM workshop on Home networks*. ACM, 2011, pp. 1–6.

[42] 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.

[43] 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.

[44] 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. [Online]. Available: http://doi.acm.org/10.1145/2187836.2187965

[45] 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.

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

[47] F. Muoz, R. Muoz, J. Rodrguez, V. Lopez, O. G. de Dios, and J. P. Fernndez-Palacios, "End-to-end service provisioning across mpls and ip/wdm domains," in *Smart Communications in Network Technologies (SaCoNeT), 2013 International Conference on*, vol. 02, June 2013, pp. 1–5.