Mobile Clouds for Smart Cities

https://doi.org/10.3991/ijoe.v13i01.6320

Yousef Ibrahim Daradkeh
Prince Sattam bin Abdulaziz University, Wadi Aldawaser, KSA
daradkehy@yahoo.ca

Mujahed ALdhaifallah
Prince Sattam bin Abdulaziz University, Wadi Aldawaser, KSA
m.aldhaifallah@psau.edu.sa

Dmitry Namiot
Lomonosov Moscow State University, Moscow, Russia
dnamiot@gmail.com

Abstract—This paper is devoted to mobile cloud services in Smart City projects. As per mobile cloud computing paradigm, the data processing and storage are moved from the mobile device to a cloud. In the same time, Smart City services typically contain a set of applications with data sharing options. Most of the services in Smart Cities are actually mashups combined data from several sources. This means that access to all available data is vital to the services. And the mobile cloud is vital because the mobile terminals are one of the main sources for data gathering. In our work, we discuss criteria for selecting mobile cloud services.

Keywords—A Smart Cities; mobile cloud; grid services; NoSQL.

1 Introduction

In this article, we would like to focus on selecting solutions for mobile cloud services in the smart city. We discuss criteria for selection and rationale for architectural decisions in the pilot project for a mobile operator in Moscow.

As per [1], mobile cloud computing is a paradigm for mobile applications proposes the movement of the data processing and storage from the mobile device to some centralized computing platforms. And this centralized platform should be cloud-based (in other words, located in clouds). These centralized applications are then accessed over the wireless connection based on a thin native client or web browser the mobile devices. So, it is an infrastructure where both the data storage and data processing happen outside of the mobile device.

As per definition from British Standard Institute [2], Smart City is an effective integration of physical, digital and human systems in the built environment to deliver a sustainable, prosperous and inclusive future for its citizens.

A key feature of smart cities is the ability of the component systems to interoperate. The above-mentioned PAS-182 [2] defines a concept model and gives guidance to decision makers on applying it to promote interoperability for data created, used, and maintained by a city across all sectors.

So, it is a vital issue for services in Smart Cities to share data. Even more, we can conclude that most of the services for Smart Cities are mashups and collect (proceed) data from many sources [3]. In the above-mentioned paper, we show that most services could be classified via Data Program Interfaces (DPI), rather than via Application Program Interfaces (API).

As per BSI specification, data is a resource that can transform the capability of a city. This resource backs the development of Smart City services. And the centralized store (cloud at the whole is centralized) is the convenient way for access to different data from mashups. In general, Smart City consists of organizations across all sectors, facilitated by the sharing of data, based on a common framework of its meaning, and consistent use of identifiers and classifications [2].

Why should it be a mobile cloud? Mobile devices play an important role in gathering data in Smart Services. It is so-called crowd-sensing (or mobile crowd-sensing) [4].

There is so-called crowd-sourcing as a form of cooperation of a group of users (crowd), where all single users are performing small subtasks of a bigger task. It lets handle complex problems with many co-working users. Crowd-sensing is a subtype of crowd-sourcing where the actually outsourced job is a complex sensing task [5].

Crowd-sensing could be used in Smart Cities alone and alongside with sensor networks [6]. It is an additional technology which involves moving sensors on mobile devices. In Smart Cities, many crowd-sensing applications target such areas as noise and pollution measurements, urban transportation systems, tracking of public buses and trams, etc. The typical tasks for mobile tracking are circled about various forms of monitoring [7].

Of course, we need to save data from mobile sensors. And data should be available for mashups too. It means that mobile cloud should be an important part of Smart City platform. It could bring the following benefits for Smart Cities [2]:

- reduced cost as the need to recollect and verify data is removed;
- integrated city systems and data-driven services;
- a common understanding of the needs of communities;
- shared objectives, collaboratively developed and evidenced using data;
- engaged and enabled citizens and communities;
- transparency in decision-making models;
- consequently improved quality of life for citizens.

The rest of the paper is organized as follows. In Section II, we discuss related works. In Section III, we discuss criteria for selecting mobile cloud solutions in Smart Cities. In Section IV, we present our architecture for mobile cloud in Smart Cities.

2 Related Works

With the original idea to eliminate the constraints of weakness in computing power in mobile devices, mobile cloud computing is an attractive topic in scientific research as well as in practical implementations.

Mobile cloud computing has various service models [8]. In the upper layers, we have the following paradigms: Data centers layers, Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) (Figure 1).

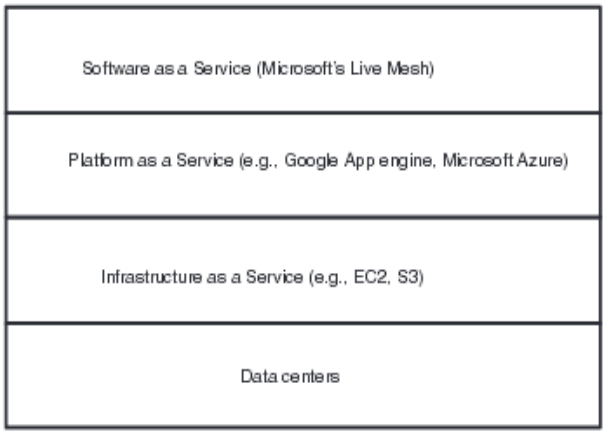

Fig. 1. Cloud computing paradigm

Data centers layer provides the hardware facility and infrastructure for clouds. IaaS is built on top of the data center layer and enables the provision of storage, hardware, servers, and networking components. The canonical examples of IaaS are Amazon Elastic Cloud Computing and Simple Storage Service (S3) [9]. PaaS offers an advanced integrated environment for building, testing, and deploying custom applications. The examples of PaaS are Amazon Map Reduce/Simple Storage Service, Google App Engine or Microsoft Azure [10]. SaaS supports a software distribution with specific requirements. The canonical examples are Salesforce or Google Apps [11].

For smart cities at the present time, we have a very agile structure of services, which is continuously added new services and removed the old ones. In addition, the idea of smart cities implies an increase in the number of developers from many different areas (we need mashups). It means that PaaS should be the most suitable layer for Smart Cities backends unless we will get a stable set of services and move them to SaaS layer.

What is the main difference of mobile clouds from the general model? We have to have some level of support for mobile devices (mobile operations systems). So, for the last years, we can see the emergence of a new technology (and new acronym), Mobile Backend as a Service (MBaaS) [12].

MBaaS (backend as a service - BaaS), is a model for providing the web and mobile app developers with a way to link their applications to backend cloud storage [13]. MBaaS exposes APIs and custom software development kits (SDKs) for mobile developers and also provides features such as user management, push notifications, and integration with social networking services. Usually, MBaaS providers offer a different set of backend tools and resources. As the common services provided by the majority of providers, we can mention file storage, file sharing, push notifications, location services, chat and messaging, integration with social networks such as Facebook and Twitter, usage analysis tools [14]. MBaaS are gaining mainstream traction with enterprise consumers being vendor-agnostic and suitable for novice developers.

For example, MBaaS Convertigo [15], supports a wide set of features. Developers can connect to enterprise data using a wide range of connectors such as SQL or Web Services. MBaaS supports cross-platform development – desktop and mobile apps on multiple devices (iOS, Android), as well as server-side business logic. Convertigo supports push notifications and test driven development, integrated version control (GIT, SVN), etc.

The common features for MBaaS include also support for programming device features (e.g., plugins, APIs) such as cameras or sensors, support for development environment (e.g., integrated version control or GIT), multiple operational systems support, cloud deployment, testing support and encrypted transactions. An important feature for all mobile backends is activity monitoring. It lets monitor system activities such as connected devices, server's request or response's time and logging. This monitor should also support a search for activity logs with rich tracking and filtering options. Of course, MBaaS should support user authentication (e.g., LDAP, Facebook Connect), mobile applications management, visual development and provide task scheduler. The scheduler is mandatory at least for push notifications planning [16].

As per [17], MBaaS offerings are positioned between the existing platform-as-a-service vendors and the full end-to-end solution space, occupied by mobile enterprise/consumer application platforms (Figure 2).

As a typical example here we can mention FI-WARE [18] mobile cloud. It is based on OpenStack functionality. We are mentioning FI-WARE because it is a "standard" offer for Smart City projects, supported by European Commission.

We've mentioned already, that in our opinion FI-WARE is over-engineered and unnecessary complexity [3]. In Figure 4, we illustrate the MBaaS platform from EMC [20]

It is more "service" oriented and directly enlists proposed services as push notifications, analytics, file store, etc. When we use MBaaS services, the amount of code will be less. Most of the mobile specific functionality will be managed by the MBaaS service. In the same time, FI-WARE offers a generic approach and leaves details to so-called enablers. Actually, it is a very important point of view. Excessive generalization loses more practical solutions.

The Cloud Standards Customer Council (CSCC) [21] has published the Customer Cloud Architecture for Mobile whitepaper. This paper describes vendor-neutral best practices for hosting the services and components required to support mobile applications using cloud computing. It provides the reference architecture for mobile cloud [22].

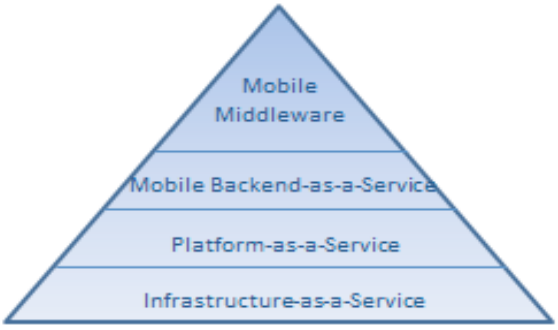

Fig. 2. MBaaS triangle [17]

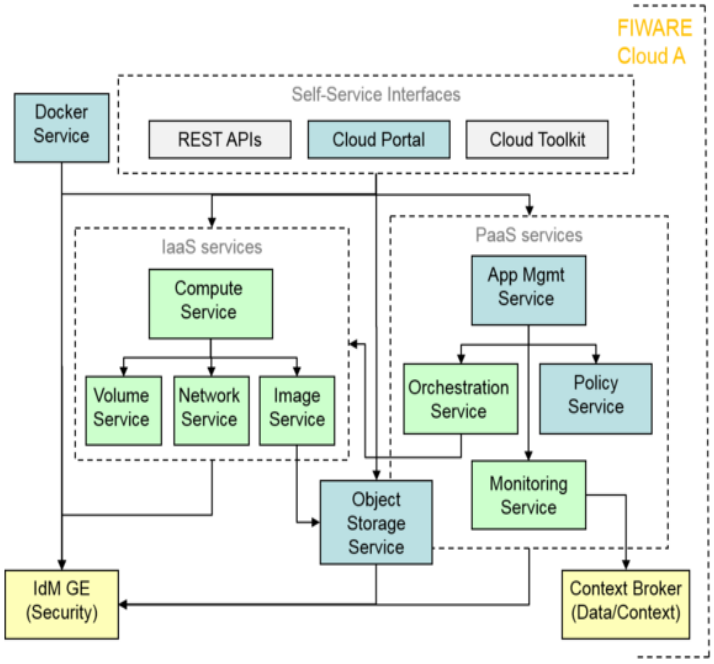

Fig. 3. FI-WARE mobile cloud [19]

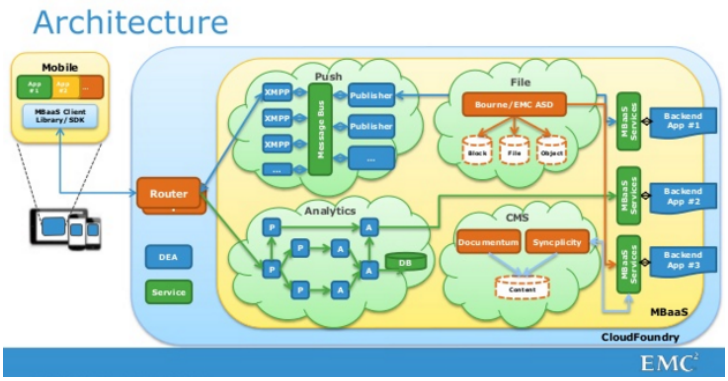

Fig. 4. EMC MBaaS [20]

Hyrax [23] is a platform derived from Hadoop that supports cloud computing on Android smartphones. Hyrax allows client applications to conveniently utilize data and execute computing jobs on networks of smart-phones and heterogeneous networks of phones and servers. Hyrax supports a Hadoop [24] cluster which is configured to run on Android phones. Running Hadoop on a cluster of phones is analogous to running Hadoop on a cluster of servers. It is a very interesting approach, but we this it is not suitable for Smart Cities applications, where data processing should be mode "centralized". Decentralized systems attract a lot of attention nowadays, but as per our experience the decentralized computing model was prohibited by city's authority.

Mobile-Edge Computing (MEC) [25] provides cloud-computing capabilities at the edge of the mobile network. The main features are ultra-low latency, high bandwidth as well as real-time access to radio network information that can be leveraged by applications. Technically, it is a next step in the convergence of IT and telecommunications networking. As per ETSI vision, use cases include video analytics and Internet of Things. In other words, it is more than applicable for Smart Cities.

The key element of MEC is its application server, which is placed with the base station. This server provides computing capabilities, storage capacity, and connectivity. It looks like a localized cloud infrastructure. Via own Application Program Interfaces (API), this server provides access to the traffic and radio network information. This option lets application developers to tune their services. The above-mentioned API also allows real-time analytics. We think that MEC will be a mandatory part of 5G deployment and could be a part of Smart City technological stack. At this moment, it looks just as a promising technology from ETSI.

Cloudlet model [26] is similar to MEC. A cloudlet is a cloud datacenter that is located at the edge of the Internet. It is the same idea to support resource-intensive and interactive mobile applications by providing computing capabilities and storage capacity to mobile devices with lower latency.

As per its original idea [27], a cloudlet is a new architectural element that arises from the convergence of mobile computing and cloud computing. It represents the middle tier of a 3-tier hierarchy:

mobile device -> cloudlet -> cloud.

Some authors call it as a data center in a box with the main goal to bring the cloud closer [28]. The key features, usually listed in scientific papers are self-managing, good connectivity to the cloud, a logical proximity to the associated mobile devices, a usage of standard cloud technologies.

The logical proximity could be defined as low latency and high bandwidth. In terms of physical proximity it could be, for example, combined with Wi-Fi access point.

3 On Selection Criteria

In this part, we would like to discuss the criteria for mobile cloud selection in Smart City projects. In our opinion, Smart Cities development adds own specific to this process.

Let us return back to the basic ideas behind MBaaS. Actually, they are very simple:

- mobile applications need a back-end
- back-end services are complex to build and test
- reuse the same back-end service can decrease time-to-market value for mobile applications.

So, the time to market is the key criteria here. Actually, the same is true for any API's pretended to the "standard". The standard API should simplify the development.

By the commercial reasons, Smart City core applications (Smart City platforms, Smart City SDK, etc.) in the most cases rely on open solutions and cannot use the commercial products without limitations.

In some countries (Russia is an example), collected data should be saved locally (cannot cross the borders). This requirement closes the usage Amazon and other popu-

lar public clouds. Technically, this restriction covers so-called personal data. But because many public measurements in Smart Cities are collected with smartphones, they fall into this category (smartphone's ID – IMEI, for example, links measurements to the owner).

The set of services for Smart Cities is not stable. We have to deal with the constantly updated list of services. Some old of them become irrelevant while new specifications require the newest development. Cities often involve non-professional developers from various areas into service (software) development. This once again emphasizes the requirement for simplicity and speed of development. Creating services should be cheap, as well as the abandonment of existing services should not be very expensive.

Most of the services for Smart Cities, at least, in the current vision, could be presented as on-demand Internet of Things (IoT) system. This IoT system contains several elements (e.g., sensors) at the edge, network function capabilities (middle tier), cloud services (backend). The typical situation for Smart Cities is a short but heavy workload. It is crucial to support data gathering in such cases and provide an end-to-end provisioning. And obviously, Smart Cities should avoid provisioning the above-mentioned elements separately and manually. A dynamic provisioning of resources for IoT systems requires so-called information-centric design [29]. An opposite approach is so-called host-centric architecture, which is tied to the actual host where particular functionality is available [30].

The information-centric approach leverages such functions as in-network caching, multiparty communication via replication, and (most important) interaction models where senders and receivers are decoupled. We can see several information-centric models. For example [31]:

- Data-Oriented Network Architecture (DONA) [32]
- Content-Centric Networking (CCN) [33],
- Publish-Subscribe Internet Routing Paradigm (PSIRP) [34]
- Design for the Future Internet and Scalable and Adaptive Internet Solutions [35]

In our paper, we discuss an experimental design for mobile cloudlets with a publish-subscribe model.

4 Messaging Cloudlets

In this section, we describe our experimental design for information-centric architecture.

Our idea is to deploy publish-subscribe message system as cloudlet. As the particular example of high-throughput distributed messaging system, we choose Kafka system [36]. Kafka is a distributed message broker. Technically, message brokers are used for a variety of reasons. The two most important arguments in our case are: to decouple processing from data producers and to buffer unprocessed messages. We think these two features are very suitable for the short and heavy workloads, mentioned in Section III. The original use case for Kafka was to be able to rebuild a user activity tracking pipeline as a set of real-time publish-subscribe feeds [37]. These feeds are available for subscription for a range of use cases including real-time processing, real-time monitoring, and loading into databases for offline processing and reporting. Actually, it is a perfect example of the measurements in Smart City applications.

In general, Kafka is a distributed publish-subscribe messaging system that is designed to be fast, scalable, and durable. At its core, Kafka maintains feeds of messages in topics. Kafka treats each topic partition as a log (an ordered set of messages). Producers (e.g., mobile applications) write data to topics and consumers (data proceedings) read from topics. Kafka is a distributed system, so, topics are partitioned and replicated across multiple nodes. The key abstraction in Kafka is a structured commit log of updates (Figure 5).

Messages in Kafka are simply byte arrays. It is possible to attach a key to each message, in which case the producer guarantees that all messages with the same key will arrive to the same partition.

Kafka does not attempt to track which messages were read by each consumer. Kafka retains all messages for a predefined amount of time, and consumers are responsible for tracking their location in each log. In our case, Smart City's data proceeding applications have got some predefined time to read measurements and react. By these principles, Kafka can support a large number of consumers and retain large amounts of data with very little overhead.

In our case, we are planning to use Kafka cluster (Figure 6). When communicating with a Kafka cluster, all messages are sent to the partition's leader. The leader is responsible for writing the message to its own in sync replica and, once that message has been committed, is responsible for propagating the message to additional replicas on different brokers.

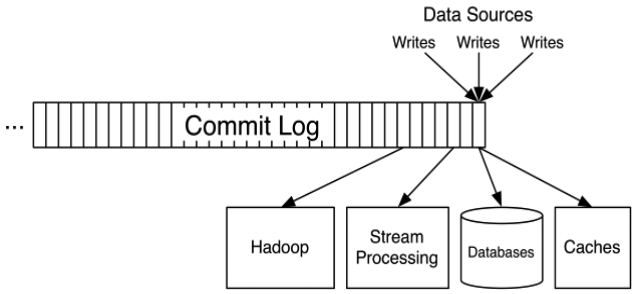

Fig. 5. Kafka structure [38]

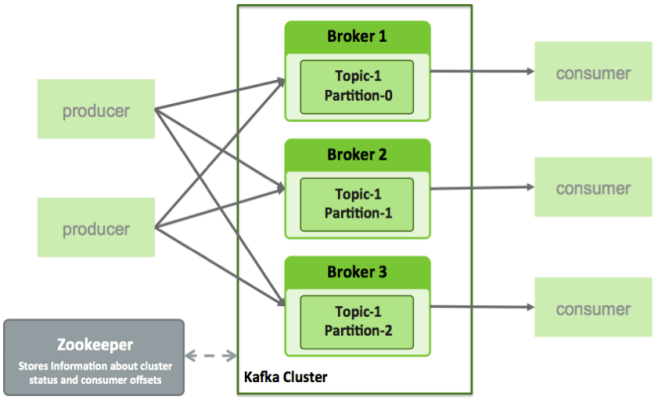

Fig. 6. Kafka cluster [39]

The topic here corresponds to the particular service within Smart City, presented by the producer (e.g. mobile application).

Mobile clients (e.g., crowdsensing applications [40]) communicate with Kafka through REST proxy, based on Nginx. This proxy allows more flexibility for developers, and it significantly broadens the number of systems and languages that can access cloudlet (Kafka cluster).

As the backend data store, we propose Cassandra [41]. In our opinion, it is the best choice for time series data (typical representation for the measurements. And cloud-based Cassandra installation is a widely used choice [42].

5 Conclusion

In this paper, we discuss mobile cloud development in deployment for Smart Cities application. We have discussed several opportunities as well selection criteria for cloud support in Smart City applications. We propose a design for mobile cloud solution based on the cloudlet approach. In our design, we use Kafka cluster as messaging based cloudlet, REST proxy for mobile clients and Cassandra as cloud data store. In our opinion, it has got advantages over existing proposals like FI-WARE due to its simplicity and usage of standard open source solutions in the background.

6 Acknowledgment

The project was supported by the deanship of scientific research at Prince Sattam bin Abdulaziz University (Kingdom of Saudi Arabia) under the research project # 2014/1/863. We would like to thank people from Open Information Technologies Lab in Lomonosov Moscow State University for the valuable discussions. We would like to thank the anonymous reviewers helped us to improve the early version posted in Arxiv preprint service [43].

7 References

1. Dinh, Hoang T., et al. "A survey of mobile cloud computing: architecture, applications, and approaches." Wireless communications and mobile computing 13.18 (2013): 1587-1611. https://doi.org/10.1002/wcm.1203
2. PAS 182 Smart city concept model http://www.bsigroup.com/en-GB/smart-cities/Smart-Cities-Standards-and-Publication/PAS-182-smart-cities-data-concept-model/ Retrieved: May, 2016
3. Namiot, Dmitry, and Manfred Sneps-Sneppe. "On software standards for smart cities: API or DPI." ITU Kaleidoscope Academic Conference: Living in a converged world-Impossible without standards?, Proceedings of the 2014. IEEE, 2014. https://doi.org/10.1109/kaleidoscope.2014.6858494
4. Guo, Bin, et al. "From participatory sensing to mobile crowd sensing." Pervasive Computing and Communications Workshops (PERCOM Workshops), 2014 IEEE International Conference on. IEEE, 2014. https://doi.org/10.1109/percomw.2014.6815273
5. Petkovics. A, et al. " Crowdsensing Solutions in Smart Cities towards a Networked Society." EAI Endorsed Transactions on Internet of Things 15(1): e6 (2015).

6. Perera, Charith, et al. "Sensing as a service model for smart cities supported by internet of things." Transactions on Emerging Telecommunications Technologies 25.1 (2014): 81-93. https://doi.org/10.1002/ett.2704
7. Smart City Use Cases and Requirements, CityPulse Project http://www.ict-citypulse.eu/page/content/smart-city-use-cases-and-requirements Retrieved: May, 2016
8. D. Kovachev, Y.Cao, and R. Klamma, "Mobile Cloud Computing: A Comparison of Application Models", International Journal of Engineering Technology & Management Research, Vol. 1, Issue. 1, Feb. 2013, pp. 20-25.
9. Wang, Lizhe, et al. "Cloud computing: a perspective study." New Generation Computing 28.2 (2010): 137-146. https://doi.org/10.1007/s00354-008-0081-5
10. Luo, Jun-Zhou, et al. "Cloud computing: architecture and key technologies." Journal of China Institute of Communications 32.7 (2011): 3-21.
11. Pallickara, Shrideep. "Some Recent Advances in Utility and Cloud Computing." Future Generation Computer Systems 56.C (2016): 315-316. https://doi.org/10.1016/j.future.2015.11.018
12. Rogers, Michael P., and Bill Siever. "Solving the Cloud Computing Impasse with MBaaS." Proceedings of the 47th ACM Technical Symposium on Computing Science Education. ACM, 2016.
13. Gheith, A., et al. "IBM Bluemix Mobile Cloud Services." IBM Journal of Research and Development 60.2-3 (2016): 7-1.
14. Barker, Adam, Blesson Varghese, and Long Thai. "Cloud Services Brokerage: A Survey and Research Roadmap." arXiv preprint arXiv:1506.00485 (2015).
15. Convertigo http://www.convertigo.com Retrieved: May, 2016
16. Sneps-Sneppe, Manfred, and Dmitry Namiot. "Spotique: A new approach to local messaging." Wired/Wireless Internet Communication. Springer Berlin Heidelberg, 2013. 192-203.
17. Michael Facemire Mobile Backend-As-A-Service: The New Lightweight Middleware? http://blogs.forrester.com/michael_facemire/12-04-25-mobile_backend_as_a_service_the_new_lightweight_middleware Retrieved: May, 2016
18. Ramparany, Fano, et al. "Handling smart environment devices, data and services at the semantic level with the FI-WARE core platform." Big Data (Big Data), 2014 IEEE International Conference on. IEEE, 2014. https://doi.org/10.1109/bigdata.2014.7004417
19. FI-WARE Cloud Hosting https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/Cloud_Hosting_Architecture Retrieved: May, 2016
20. Mobile Backend as a Service via Cloud Foundry http://www.slideshare.net/WangTianqing/mobile-backend-as-a-service-via-cloud-foundry
21. The Cloud Standards Customer Council (CSCC) http://www.cloud-council.org/ Retrieved: May, 2016
22. Cloud Customer Architecture for Mobile http://www.cloud-council.org/deliverables/CSCC-Cloud-Customer-Architecture-for-Mobile.pdf Retrieved: May, 2016
23. Marinelli, Eugene E. Hyrax: cloud computing on mobile devices using MapReduce. No. CMU-CS-09-164. Carnegie-mellon univ Pittsburgh PA school of computer science, 2009.
24. Lei, L. E. I. "Towards a High Performance Virtual Hadoop Cluster." Journal of Convergence Information Technology 7.6 (2012).
25. Hu, Y. C., et al. "Mobile Edge Computing—A Key Technology Towards 5G." ETSI White Paper 11 (2015).
26. Ha, Kiryong, and Mahadev Satyanarayanan. "OpenStack++ for Cloudlet Deployment." School of Computer Science Carnegie Mellon University Pittsburgh (2015).
27. Elijah Cloudlet-based Mobile Computing http://elijah.cs.cmu.edu/ Retrieved: May, 2016

28. Soyata, Tolga, et al. "Accelerating mobile cloud computing: A survey." Communication Infrastructures for Cloud Computing (2013): 175-197.
29. Trossen, Dirk, and George Parisis. "Designing and realizing an information-centric internet." Communications Magazine, IEEE 50.7 (2012): 60-67. https://doi.org/10.1109/MCOM.2012.6231280
30. Dannewitz, Christian. "Netinf: An information-centric design for the future internet." Proc. 3rd GI/ITG KuVS Workshop on The Future Internet. 2009.
31. Ahlgren, Bengt, et al. "A survey of information-centric networking." Communications Magazine, IEEE 50.7 (2012): 26-36. https://doi.org/10.1109/MCOM.2012.6231276
32. Koponen, Teemu, et al. "A data-oriented (and beyond) network architecture." ACM SIGCOMM Computer Communication Review. Vol. 37. No. 4. ACM, 2007. https://doi.org/10.1145/1282380.1282402
33. Oehlmann, Fabian. "Content-centric networking." Network 43 (2013).
34. Lagutin, Dmitrij, Kari Visala, and Sasu Tarkoma. "Publish/Subscribe for Internet: PSIRP Perspective." Future internet assembly 84 (2010).
35. Ko, Bong Jun, et al. "An information-centric architecture for data center networks." Proceedings of the second edition of the ICN workshop on Information-centric networking. ACM, 2012. https://doi.org/10.1145/2342488.2342506
36. Ranjan, Rajiv. "Streaming big data processing in datacenter clouds." IEEE Cloud Computing 1 (2014): 78-83. https://doi.org/10.1109/MCC.2014.22
37. Apache Kafka documentastion http://kafka.apache.org/documentation.html#uses Retrieved: May, 2016
38. Putting Apache Kafka To Use: A Practical Guide to Building a Stream Data Platform http://www.confluent.io/blog/stream-data-platform-1/ Retrieved: May, 2016
39. What Kafka does http://hortonworks.com/apache/kafka/ Retrieved: May, 2016
40. Namiot, Dmitry, and Manfred Sneps-Sneppe. "On Open Source Mobile Sensing." Internet of Things, Smart Spaces, and Next Generation Networks and Systems. Springer International Publishing, 2014. 82-94.
41. Lakshman, Avinash, and Prashant Malik. "Cassandra: a decentralized structured storage system." ACM SIGOPS Operating Systems Review 44.2 (2010): 35-40. https://doi.org/10.1145/1773912.1773922
42. Khan, Zaheer, et al. "Towards cloud based big data analytics for smart future cities." Journal of Cloud Computing 4.1 (2015): 1-11. https://doi.org/10.1186/s13677-015-0026-8
43. Sneps-Sneppe M., Namiot D. On Mobile Cloud for Smart City Applications //arXiv preprint arXiv:1605.02886. – 2016.

8 Authors

Dr. Yousef Daradkeh is Associate Professor with the Prince Sattam bin Abdulaziz University, College of Engineering at Wadi Aldawaser, 18611, KSA (e-mail: daradkehy@yahoo.ca).

Dr.Mujahed ALdhaifallah is Dean of College of Engineering at Wadi Aldawaser 18611, with the Prince Sattam bin Abdulaziz University, KSA (e-mail: m.aldhaifallah@psau.edu.sa:).

Professor Dmitry Namiot is now with Lomonosov Moscow State University, Moscow, Russia (e-mail: dnamiot@gmail.com).

Submitted, 7 October 2016. Published as resubmitted by the authors on 01 Dec 2016.