Fog Computing Framework for Smart City Design

https://doi.org/10.3991/ijim.v14i01.9762

Mais Haj Qasem (✉)
The University of Jordan, Amman, Jordan
mais_hajqasem@hotmail.com

Alaa Abu-Srhan, Hutaf Natoureah
The Hashemite University, Amman, Jordan

Esra Alzaghoul
The University of Jordan, Amman, Jordan

Abstract—Fog computing is a new network architecture and computing paradigm that uses user or near-user devices (network edge) to conduct some processing tasks. Accordingly, this network architecture extends cloud computing with more flexibility than that found in ubiquitous networks. A smart city based on the concept of fog computing with flexible hierarchy is proposed in this study. The aim of the proposed design is to overcome the limitations of previous approaches, which depend on using various network architectures, such as cloud computing, autonomic network architecture, and ubiquitous network architecture. Accordingly, the proposed approach reduces the latency of data processing and transmission with enabled real-time applications, distributes processing tasks over edge devices to reduce the cost of data processing, and allows collaborative data exchange among the applications of a smart city. The design comprises five major layers, which can be increased or merged according to the amount of data processing and transmission in each application. The involved layers are as follows: connection, real-time processing, neighborhood linking, main processing, and data server layers. A case study of a novel smart public car parking, traveling, and direction advisor is implemented using iFog-Sim, and results show that the delay of real-time application, cost, and network usage are significantly reduced compared with that of a cloud computing paradigm. Moreover, the proposed approach increases the scalability and reliability of user access without considerably compromising time, cost, and network usage compared with fixed fog computing.

Keywords—Fog Computing, Smart City, Cloud, Internet of Thing.

1 Introduction

A smart city is an urban area covered by e-services, e-resource management, and e-planning using a collection of sensors, communication devices, and data-processing facilities that aim to enhance the services provided for citizens and manage available resources [1, 2]. A smart city involves various application types, such as smart water and waste management, smart energy management, smart transportation, smart traffic, disaster management, utility management, and smart building and loss management [3]. Various smart city designs have been proposed in the literature [4, 5]. The existing smart city designs differ in terms of the applications, devices, data, and access consideration. Given that smart city applications use different devices and transmit varied and large data, managing these applications in a smart city design is anything but trivial. The diversity of smart city applications creates interoperability, openness, and convergence problems [6].

A set of smart city requirements that addresses the expected needs and problems of smart cities has been established in the literature to facilitate the development of robust smart city designs. These requirements can be summarized in the following. The first requirement is the ability to support heterogeneous applications and services, which vary in terms of data type and purpose, such as decision making, resource management, and future planning [6].

The second requirement is supporting heterogeneous platforms, which include running party and profit. Smart city platforms can be a government, enterprise, or business platforms. Government platforms are run by the government and focus on critical applications and resource management; enterprise platforms emphasize large-profit applications, such as transportation; business platforms focus on products and small applications that generate revenue. These platforms require different designs with varying data collection and access permissions [5].

The third requirement is supporting a variety of technologies while ensuring scalability, privacy, access, and testbeds of the smart city network [7]. 4) The last requirement is the ability to support ubiquitous access to services and information with wide coverage, reliability, fast responsiveness, and low cost [2, 4, 8-10].

The main concepts of a smart city design, which reveal the extent to which the design covers these requirements, are the network architecture and the IoT infrastructure. A smart city domain has four different network architectures: autonomic, ubiquitous, application-layer overlay, and service-oriented network architecture [11]. The application-layer overlay, and service-oriented network architecture inherit their properties from the implemented connectivity types, which include the autonomic and the ubiquitous.

Autonomic network is developed to over-come the limitation in the internet architecture by given more flexibility to the network and provide dynamic and fully autonomous nodes formation. Accordingly, the autonomic device can communicate with pre-determined devices in specific-format. The autonomic network architecture is a good approach for smart cities, however there is devices, technologies and services limitations. Nevertheless, autonomic network has a significant feature as it “makes network devices intelligent by introducing self-management concepts that simplify network management for the network operator”. [19].

Ubiquitous network architecture aims at making services and communication available anytime, anywhere and using any device. Accordingly, the user or the autonomic device can communicate with any-other devices in any-format. The requirement for this architecture is: Internet, communication protocols and middleware. The ubiquitous network architecture consists of three layers, the task management layer, which is responsible for monitor tasks, environment management layer, which is responsible for resource monitoring and management and the environment layer, which manage the reliability of the resources. The ubiquitous network architecture is a good approach for smart cities, however the scalability, privacy and testbeds are major problems of this architecture [20]. Thus, a comparison between the autonomic and the ubiquitous architecture is presented in Table 1.

Table 1. Comparison between Network Architecture based on Smart Cities Requirements

Requirements	**Autonomic**	**Ubiquitous**
Applications and Services	Limited to applications within the covered area by the autonomic network.	Limited to services connected to the internet.
Platforms	Support all.	Support all.
Enabling Technologies	Subject to privacy expose.	Subject to access denial and slow response.
Connectivity features	Autonomic	Ubiquitous

Accordingly, standard architecture exhibits smart city-related problems. By contrast, the cloud computing paradigm solves much of this limitation, thereby allowing auto-configuration, providing reliable access, and overcoming heterogeneity. However, the cloud also comes with disadvantages, such as delay, scale, real-time processing, and cost [25].

In fog computing, which was proposed by Cisco in 2012, geo-distributed application is run throughout the network [12]. Fog is a new network architecture and computing paradigm that uses user or near-user (network edge) devices to conduct some processing. Accordingly, this architecture extends cloud computing with more flexibility than that found in the ubiquitous networks. Thus, fog computing allows integration of a massive number of components and services [13]. Given that fog is based on distribution, it solves the problem of the currently used cloud-based smart city architecture while preserving all the advantages of the service-based architecture, such as reliability, interoperability, and scalability [23].

A smart city architecture based on fog computing, which comprises four levels of interconnecting devices for storage, analysis, and communication purposes, was proposed by Bar-Magen [12]. This architecture was tested and proven to be feasible for smart city applications. The goal of the proposed architecture is to provide location awareness and low-latency services for users. However, in multi-platform smart cities, enforcing such a fixed fog computing approach might cause cost problems for small applications and might be infeasible in extra-large applications. This phenomenon might be due to the less intermediate node distribution requirement by some applications, such as traffic management, while other applications with a geo-distributed nature, such as energy-consumption, require additional intermediate nodes and levels.

In this study, a non-unified hierarchical architecture is proposed for smart city design. Two layers are fixed (on the top and bottom of the hierarchy), while the rest of the layers are added based on the application and application group. The rest of this paper is organized as follows. Section 2 discusses the related work for the existing smart city designs. Section 3 presents the proposed design and detailed components and consideration. Section 4 provides a case study of a novel public parking and monitoring system with the obtained results. Section 5 finally presents the conclusion.

2 Related Work

Smart city design is controlled by requirements that are determined in advance for the designed city. Accordingly, various smart city designs with different features and criteria have been reported in the literature. Padova's smart city design [14] as illustrated in Figure 3, comprises a central server that collects data, implements the necessary processing, and allows user access. User access is provided as a web service, and protocol translation (XML-to-EXI, HTTP-to-CoAP, and IPv4/v6-to-6LoWPAN) is implemented in the gateway to overcome the interoperability problem.

Similarly, Trento's (Italy) smart city design [15] is centralized and managed by the government and provides user access as an online service. Wuxi's (China) smart city design [16] is centralized by provided services using multiple applications, unlike Trento, which implements a one-stop-shop technique [15]. Seoul's (South Korea) smart city design [17] is similar to Wuxi's (China) [16] in that it is owned and controlled by the government using a cluster of servers that stores and processes data. These designs focus on centralized processing mechanisms with multiple data collection networks. Although a central server eliminates interoperability problems, it exhibits considerable delay and cannot be used with real-time applications. Moreover, scalability issues are the main limitations of such architecture.

Amazon's smart city design [5] comprises a central server located in the clouds. However, to allow multiple platforms to be operated supported by the multi-architecture and the flexibility provided by cloud computing. Similarly, Melbourne's smart city design [10] uses cloud computing and gains the advantage of the autonomous facilities provided in the cloud and the reliability of user access. However, unlike Amazon's smart city design [5], which is built using a network architecture, Melbourne's smart city design [10] is service-oriented. Thus, using this type of design to integrate additional applications and services is easy.

Ganchev and Ji [6] proposed a generic smart city design that is based on the cloud. Similar to [5], this system has low autonomy and is run by a central station. Although cloud-based approaches solve many central server-based problems, the use of the cloud faces delay issues, and network usage and cost are major challenges in cloud-based designs.

Tang et al. [21] focused on data intensive analysis since it is the major challenge in smart city. They introduced a hierarchical distributed Fog Computing architecture in order to handle the integration between services and huge number of infrastructures in smart cities, they integrated the intelligence with fog computing to ensure the security. They used a smart pipeline monitoring system and sequential learning algorithms to analyze number of case studies for event detection. In their experiments, they worked on recognition of 12 distinct events, the results show the feasibility of the proposed work [22]. Table 2 summarizes the reviewed smart city designs.

Table 2. Existing Smart Cities

Design	Details
Padova [14]	Backend Server (data storage, management and data-access), Gateways (protocol translation and functional mapping) and IoT peripheral nodes.
Amazon [5]	Backend Server (service provider), Gateways (sensor data transmission) and IoT peripheral nodes.
Melbourne [10]	Backend Server (service provider), Gateways (sensor data transmission) and IoT peripheral nodes.
Ganchev, Ji [6]	Backend Server (service provider), Gateways (sensor data transmission) and IoT peripheral nodes.
Trento [15].	Backend Server (data storage, management and data-access).
Wuxi [16]	Backend Server (data storage, management and data-access).
Seoul [17]	Backend Server (data storage, management and data-access) and network platform.

Accordingly, these designs varied in there utilized architectures and designs based on the requirements that is established for each city. In Padova [14], interpretability was the main concern while, enabling multi-platforms was the motivation of the design in amazon [5] and the work by Ganchev, Ji [6].

3 Proposed Work

A smart city is developed in this work based on the advantages provided by fog computing, which is an extension of the cloud computing paradigm. Cloud paradigm implements data processing, storage, and control at one place in a server, thereby increasing cost and subjecting the design to long delays and high network traffic. Fog computing is a distributed paradigm that extends cloud computing to the edge of the network, by which the cloud becomes similar to the network architecture [12]. Fog places edge devices at the edge of an IoT network with low latency and high performance. Similar to cloud servers, these devices then provide processing, storage, and application services for end users [13]. Thus, data are processed locally, and filtered data are forwarded to a central server. Using fog computing, components of an application run in the cloud and fog. Fog devices can be routers, gateways, or dedicated devices. The fog and cloud architectures are illustrated in Figure 1.

The proposed design is developed with the following goals to overcome the limitations of previous approaches: reduce the latency of data processing and transmission for applications that require real-time response, distribute processing tasks over edge devices to reduce the cost of data processing, and allow collaborative data exchange among the applications of the smart city in real time.

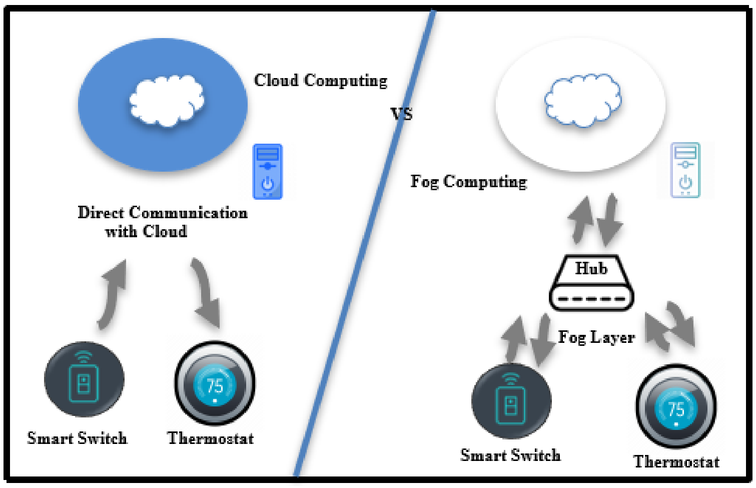

Fig. 1. Cloud-Computing vs. Fog-Computing

3.1 Design consideration

As previously mentioned, existing works on smart city design exhibit limitations with regard to access, heterogeneity, scale, real-time processing, and cost. These limitations are the core of the considerations of the developed smart city design.

- **Access**: Access should be granted at a central server through internet connection, whereas direct access is provided through edge devices within limited coverage areas to allow reliable access to provided services. For example, the number of users is pre-estimated, and the network coverage area and the suitable edge devices are designed to guarantee reliable user access
- **Heterogeneity**: As expected with IoT applications, the developed design should be flexible to incorporate gateways and protocol translations by allowing initial or incremental installation of various devices into the network as required to allow heterogeneous technologies
- **Scale**: The overall design depends on the formation of small networks that are incorporated with each other to facilitate the large expected number of connected devices with the facilities provided by the smart city. Data exchange allows users to connect to these networks and the central data servers
- **Real-time processing**: The overall design depends on fog computing, which places edge devices and the edge of the network to facilitate fast response to real-time data
- **Cost reduction**: Reducing cost is related to minimizing processing, service access, and data transmission into pre-paid cloud servers. Accordingly, the proposed design depends on the distribution of processing tasks into various edge devices

3.2 Network architecture

A hierarchical smart city design is implemented as illustrated in Figure 2 to achieve the desire goals. Specific tasks are implemented at each level using processing components and physical devices that fit with the desired goal. The design comprises five major layers, which can be increased or merged according to the amount of data processing and transmission in each application. The involved layers are as follows: connection, real-time processing, linking, main processing, and data server layers.

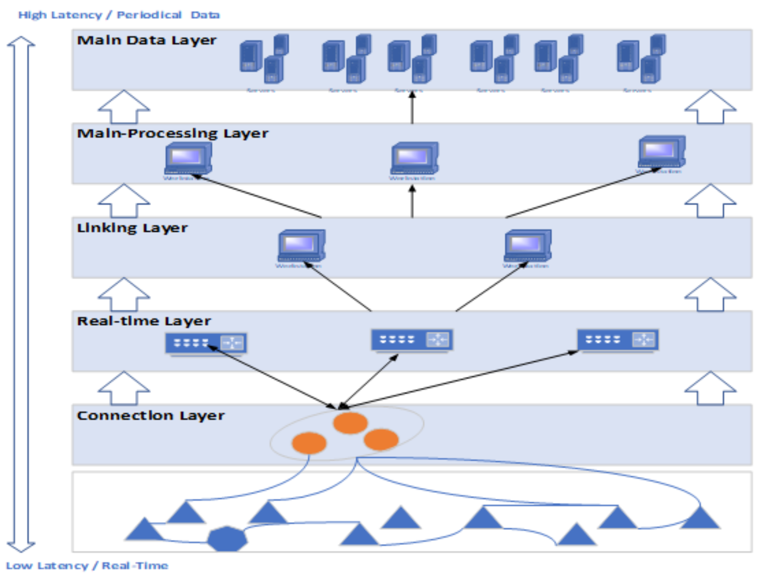

Fig. 2. The Proposed Smart City Design

In the connection layer, the collected data from the sensing devices are received and aggregated according to a specific format based on the application and the transmission protocol being used. Then, the data are transferred into the next layer using the gateway, which is the main device and commonly the sole device in this layer. In the real-time processing layer, a part of the processing task is implemented based on inputs from the previous layer, and an output is generated and sent to the linking layer, which is responsible for simple yet critical processing tasks that require real-time response.

The devices in this layer can include the router, dedicated devices, or any other edge device. This layer is the first layer of the fog network. Data that do not require real-time processing are directly forwarded to the linking layer. In this layer, which is the second layer of the fog network, another processing task is implemented to prepare the data for transmission into another neighborhood application, for returning to the connection layer as real-time response for specific application type, or for transferring up as the data to be stored and processed by the center server. Processed data in the real-time processing layer are encapsulated, formatted, and transmitted without further processing.

The main processing layer is the top layer at the fog computing network, which is responsible for aggregating and putting data in a specific format, filtering unrelated data, and transmitting the filtered data into the cloud [24]. While the top and bottom layers are fixed in the developed designs because they denote data collection and data storage, the three fog layers are flexible in a way that can be embodied in a single device, distributed into three groups of devices, or extended with additional devices that aggregate and process data as required.

The proposed smart city comprises various layers responsible for data processing and transmission, which are implemented in a decentralized way with the use of edge devices. This condition allows for reduced latency of real-time response, decreases the cost of data processing, and allows collaborative data exchange among the applications of the smart city in real time.

3.3 Components

According to the developed design, the components of the proposed smart city are as follows:

- **IoT devices**: Connected devices that send data to the fog network and then eventually to the cloud. IoT is designed in a network connected with a gateway to facilitate common sensing paradigms, such as Radio-frequency identification (RFID), wireless sensor network (WSN), and participatory sensing. The presence of a network with a gateway for connecting devices within an area facilitates low cost and real-time sensing of the environment. Given that the collected devices can have mobility features, gateways are placed in a fixed infrastructure
- **For network**: The network receives data from IoT devices in real time, runs translation and processing, and allows user access and transmission tasks
- **Cloud**: The cloud receives and saves periodical data from fog and allows user access

3.4 Data processing and application design

Each application is designed as a set of modules required for time processing and differentiated from modules that are not required for real-time application. Similarly, modules that process data to be exchanged with other applications are differentiated from data that have never been exchanged. Accordingly, each application is divided into modules, which are arranged in the fog and cloud devices to eliminate delays in real-time response, enhance the performance, and reduce the network usage and cost of the application as a whole. An example of application modeling is presented in the following section.

4 Case Study

A simulation of the parking system is implemented to prove the feasibility of the proposed design. The literature presents two ways to implement smart parking: private and public. The private parking system depends on associating each parking slot with a sensor, which records and transmits the status of the parking slot. Meanwhile, the public system depends on a set of cameras, which observe and analyze the monitored area as a whole, followed by processing and allocating spaces. The private system requires less processing and data transmission, thus allowing for easy implementation and integration with the smart city design. However, this system cannot be used with public parking areas, in which installation and connection of slots using sensors is not a choice.

4.1 Design

In this paper, a case study of the proposed design embodied in a novel public parking system monitoring and controlling is implemented with additional tasks compared with those of the common applications proposed in the literature. The tasks of the implemented system are as follows.

- Advise users on less crowded places at a specific time by analyzing the parking spaces in multiple areas, each of which is covered by its own system. Accordingly, this task does not require real-time analysis because historical results can serve the purpose and the crowding state of any place does not change immediately
- Direct each user within an area covered by a single system on available parking slot to be used. This task requires real-time analysis because the user asks for direction whenever he/she is within the area. Moreover, the status of the parking slots can rapidly change
- Report to the government on crowded areas for future planning using historical data
- Forward information for other applications, such as traffic monitoring.
- Use the cameras and some functionalities with the surveillance system to reduce devices, data transmission, and delay

The proposed system comprises multiple cameras installed in specific areas, where public parking slots are designed on the edge of the street, to accomplish these goals. Each camera is associated with a motion sensor. Cameras are connected to a gateway with a router to transferred traffic into the processing devices at the fog layers and the fog devices are connected to the clouds. The design of the novel public system is illustrated in Figure 3, and the network hierarchy of the designed parking system is given in Figure 4.

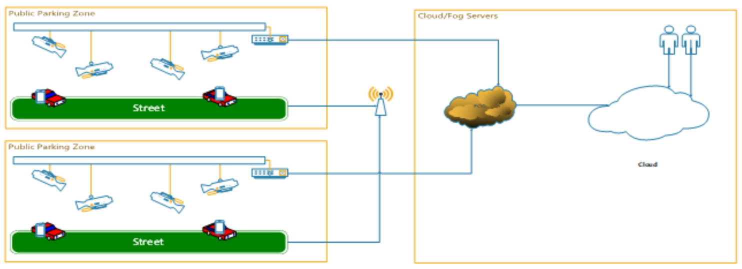

Fig. 3. Network Architecture of the Proposed Parking System

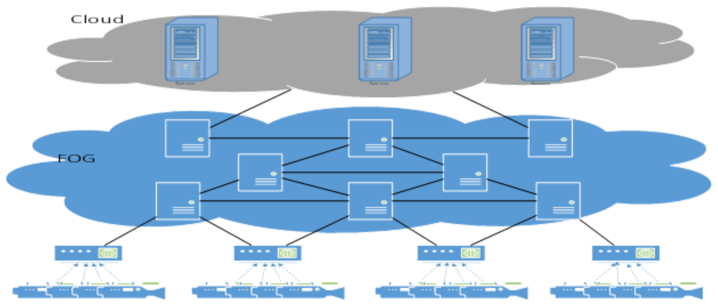

Fig. 4. Hierarchical Device Connectivity of the Proposed Parking System

The developed application comprises six modules, as illustrated in Figure 5. Four of these modules are responsible for internal processes that monitor motion, allocate car and parking spaces, and analyze crowd, while the other two modules can interact with user's on-demand. These modules are as follows:

- **Motion detection**: Responsible for capturing car motion in the covered area and sends motion notification to the other modules
- **Space allocation**: Responsible for storing, maintaining, and updating a map of the available parking slots in the covered area; updating is implemented as the module is notified with the car motion and its location
- **Zone analysis**: Responsible for storing, maintaining, and updating a map that is identical to that in space allocation; receives periodical updates regarding the changes over the map and periodically analyzes the map to calculate the crowd in the underlying zone
- **Direction calculation**: Responsible for receiving car location from the motion detection and slot location from the space allocation, calculates the best route, and directs the user to the space
- **User interface**: A website located at the cloud that receives the zone analysis data from multiple networks and advises users on less crowded areas
- **User application**: A mobile application connected to the fog device in a specific network that receives parking directions from the direction calculation modules

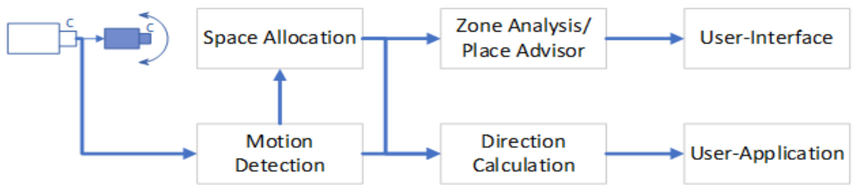

Fig. 5. Application Modules of the Proposed Parking System

4.2 Implementation

The proposed parking system is implemented using IFogSim to verify its applicability and efficiency [18]. IFogSim is a simulation tool that "models IoT and fog environments and measures the impact of resource management techniques in terms of latency, network congestion, energy consumption, and cost." IFogSim is implemented in the following classes, which forms the basics of any application simulation:

- **Fog device:** Model hardware characteristics and their connectivity, accessible memory, processor, storage, and uplink and downlink bandwidth
- **Sense:** model sensor connectivity and output attribute, its gateway and latency between the sensor and the gateway, and tuple inter-transmission or inter-arrival time
- **Actuator:** Model actuator connectivity, its gateway and latency between the actuator and the gateway, tuple inter-arrival time, and a method to perform action on tuple arrival from an application module
- **Tuple:** Communication unit between entities and involves the following parameters: type, source, destination, application module, processing requirements, and length of data
- **Application:** Directed graph of modules and dependencies

A set of devices with specific parameters, such as Million Instructions Per Second (MIPS), Rapid Eye Movement (REM), and bandwidth, should be simulated to simulate the proposed system. The parameters of the simulated devices are given in Table 3.

Table 3. Parameters of the Devices utilized in the Parking System

Parameter	Cloud	Proxy	Gateway/Router	Camera
MIPS	44,800	2,800	2,800	500
RAM	4,000	4,000	4,000	1,000
Uplink bandwidth	100	10,000	10,000	10,000
Downlink Bandwidth	10,000	10,000	10,000	10,000
Level hierarchy	0	1	2	3
Rate per MIPS	0.01	0	0	0
Busy Power	16*103	107.339	107.339	87,53
Ideal Power	16*83.25	83.433	83.433	82.44
Parent	-1	Cloud	Proxy	Gateway
Uplink Latency	None	100	2	2

The proposed system is implemented using 1-, 2-, and 3-level fog to prove the concept of the flexible hierarchical model, and the results of these models are compared with each other and with that of cloud architecture. The application modules are distributed among the available devices in each architecture because it maximizes the benefits of the available resources as shown in Table 4.

Table 4. Modules Allocation in Different Architectures

Level	CLOUD	1-Level FOG	2-Levels FOG	3-Levels FOG
Motion Detection	Camera	Camera	Camera	Camera
Space Allocation	Cloud	Fog-Device (L-1)	Fog-Device (L-1)	Fog-Device (L-1)
Direction Calculation	Cloud	Fog-Device (L-1)	Fog-Device (L-2)	Fog-Device (L-2)
Zone Analysis	Cloud	Fog-Device (L-1)	Fog-Device (L-2)	Fog-Device (L-3)
User-Application	Cloud	Fog-Device (L-1)	Fog-Device (L-2)	Fog-Device (L-3)
User-Interface	Cloud	Cloud	Cloud	Cloud

4.3 Results and discussion

The delays at two user-interactive modules, namely, user application and user interface, are represented in Figures 6 and 7, respectively. Each delay represents the loop delay of the process in the compared module. Therefore, the delay of user application represents the delay of the loop {Motion-Detector, Space-Allocator, Direction-Calculator, User-Application}, whereas the delay of user interface represents the delay of the loop {Motion-Detector, Space-Allocator, Zone-Analysis, User-Interface}.

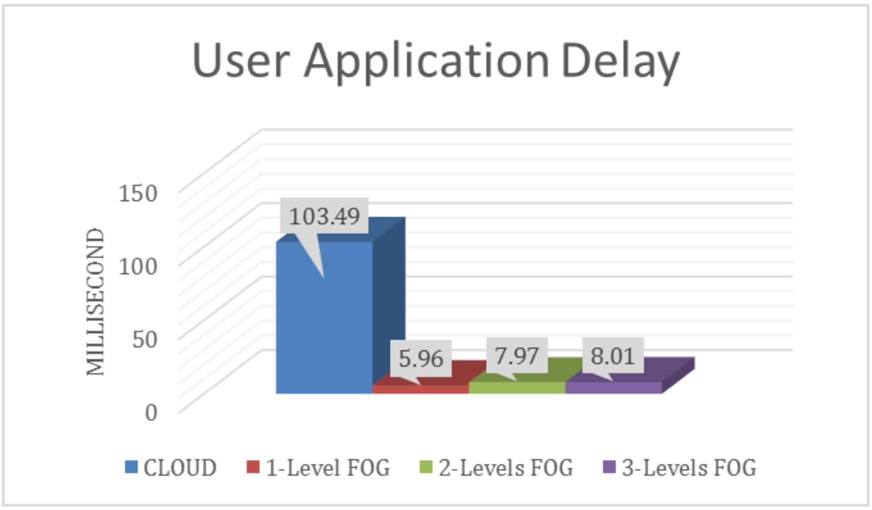

Fig. 6. Application Delay of Multiple Architecture of Proposed Parking System Compared to Cloud-Computing Paradigm

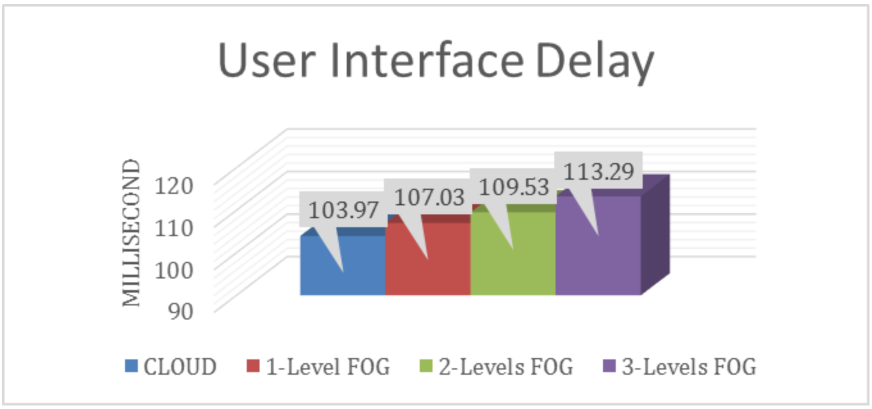

Fig. 7. Interface Delay of Multiple Architecture of Proposed Parking System Compared to Cloud-Computing Paradigm

Compared with cloud, the proposed architecture significantly reduces the delay of the user application and accordingly implements fast response to real-time application. Moreover, increasing the number of devices does not increase the delay but gains other benefits, such as smooth data exchange among applications and reliable access for users. By contrast, for user interface, which does not require real-time processing, the cloud provides better delay compared with that of the proposed fog-based architecture. However, this performance is not considered as an advantage for cloud computing because data delay in such a case is not an important issue.

The network usage and the cost of using the cloud of 10 users with 0.01 cost-per-processing are given in Figures 8 and 9, respectively. Compared with the cloud, the proposed architecture significantly reduces the network usage and cost. Moreover, increasing the number of devices does not increase the cost nor usage but gains other benefits as previously mentioned.

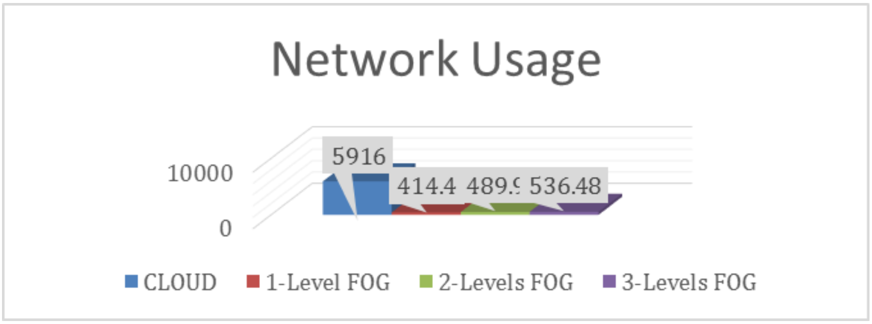

Fig. 8. Network Usage of Multiple Architecture of Proposed Parking System Compared to Cloud-Computing Paradigm

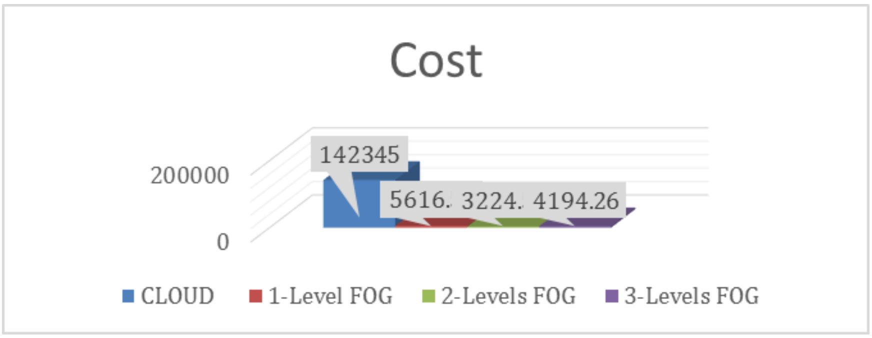

Fig. 9. Cloud Cost of Multiple Architecture of Proposed Parking System Compared to Cloud-Computing Paradigm

Finally, the only disadvantage of the proposed architecture and fog is energy consumption because additional energy is required when more devices are installed. Energy consumption on the cloud and each fog proxy for 10 users is given in Figures 10 and 11, respectively.

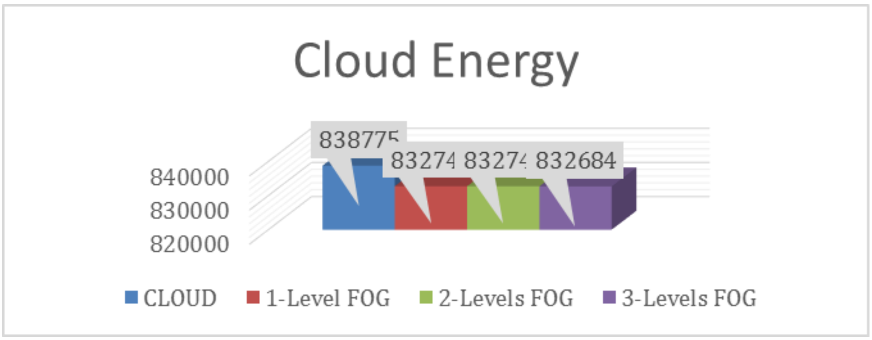

Fig. 10. Cloud Energy Consumption of Multiple Architecture of Proposed Parking System Compared to Cloud-Computing Paradigm

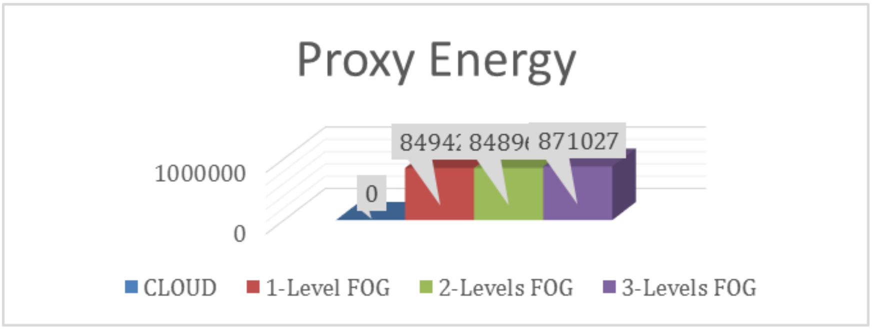

Fig. 11. Proxy Energy Consumption of Multiple Architecture of Proposed Parking System Compared to Cloud-Computing Paradigm

5 Conclusion

With advancements in communication, less attention has been provided to real-time analysis. Fortunately, using fog computing can reduce delay and enable real-time data analysis. Fog generally reduces network traffic due to its low latency and high scalability. Moreover, fog enables data management and monitoring and is suitable for IoT tasks. A novel smart city design using fog computing is proposed in this study. The proposed design comprises five major layers, which can be increased or merged according to the amount of data processing and transmission in each application. The involved layers are as follows: connection, real-time processing, linking, main processing, and data server layers.

A case study of a novel public parking and monitoring system implemented using iFogSim shows that the proposed architecture significantly reduces the delay of user application and accordingly implements fast response to real-time application in comparison with cloud computing. Moreover, this system remarkably reduces cost and network usage. Increasing the number of devices does not increase the delay but gains other benefits, such as smooth data exchange among applications and reliable access for users. Future research will focus on the design and implementation of other applications to prove the applicability and efficiency of the proposed design.

6 References

[1] Miorandi, D., et al., Internet of things: Vision, applications and research challenges. Ad hoc networks, 2012. 10(7): p. 1497-1516. https://doi.org/10.1016/j.adhoc.2012.02.016
[2] Chourabi, H., et al. Understanding smart cities: An integrative framework. in System Science (HICSS), 2012 45th Hawaii International Conference on. 2012. IEEE. https://doi.org/10.1109/hicss.2012.615
[3] Pan, G., et al., Trace analysis and mining for smart cities: issues, methods, and applications. IEEE Communications Magazine, 2013. 51(6): p. 120-126. https://doi.org/10.1109/mcom.2013.6525604
[4] Vlacheas, P., et al., Enabling smart cities through a cognitive management framework for the internet of things. IEEE communications magazine, 2013. 51(6): p. 102-111. https://doi.org/10.1109/mcom.2013.6525602
[5] Lea, R. and M. Blackstock. City hub: A cloud-based iot platform for smart cities. in Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on. 2014. IEEE. https://doi.org/10.1109/cloudcom.2014.65
[6] Ganchev, I., Z. Ji, and M. O'Droma, A generic IoT architecture for smart cities. 2014.
[7] Ruiz-Romero, S., et al., Integration of distributed generation in the power distribution network: The need for smart grid control systems, communication and equipment for a smart city—Use cases. Renewable and sustainable energy reviews, 2014. 38: p. 223-234. https://doi.org/10.1016/j.rser.2014.05.082
[8] Gardašević, G., et al., The IoT architectural framework, design issues and application domains. Wireless Personal Communications, 2017. 92(1): p. 127-148. https://doi.org/10.1007/s11277-016-3842-3
[9] Balakrishna, C. Enabling technologies for smart city services and applications. in Next Generation Mobile Applications, Services and Technologies (NGMAST), 2012 6th International Conference on. 2012. IEEE. https://doi.org/10.1109/ngmast.2012.51
[10] Jin, J., et al., An information framework for creating a smart city through internet of things. IEEE Internet of Things Journal, 2014. 1(2): p. 112-121. https://doi.org/10.1109/jiot.2013.2296516
[11] Jin, J., et al. Network architecture and QoS issues in the internet of things for a smart city. in Communications and Information Technologies (ISCIT), 2012 International Symposium on. 2012. IEEE. https://doi.org/10.1109/iscit.2012.6381043
[12] Bar-Magen, J. Fog computing: introduction to a new cloud evolution. in Escrituras silenciadas: paisaje como historiografía. 2013. Servicio de Publicaciones.
[13] Yi, S., C. Li, and Q. Li. A survey of fog computing: concepts, applications and issues. in Proceedings of the 2015 Workshop on Mobile Big Data. 2015. ACM. https://doi.org/10.1145/2757384.2757397
[14] Zanella, A., et al., Internet of things for smart cities. IEEE Internet of Things journal, 2014. 1(1): p. 22-32. https://doi.org/10.1109/jiot.2014.2306328
[15] Fioroni, G., et al., Smart Government-Toward an Innovative Concept of a "One-Stop Shop" for Interactive Online Services. IEEE-TN Smart Cities White Paper, Trento, Italy, 2014.
[16] Mao, L., W. Fang, and X.-J. Wu, Smart City of Wuxi: Current Situation and Future Plan: IEEE smart city. 2016.

[17] Lee, J.H., M.G. Hancock, and M.-C. Hu, Towards an effective framework for building smart cities: Lessons from Seoul and San Francisco. Technological Forecasting and Social Change, 2014. 89: p. 80-99. https://doi.org/10.1016/j.techfore.2013.08.033
[18] Gupta, H., et al., iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. Software: Practice and Experience, 2017. 47(9): p. 1275-1296. https://doi.org/10.1002/spe.2509
[19] IOS, C. Autonomic Networking Configuration Guide, Cisco IOS Release 15E - Autonomic Networking. 2014.
[20] Otto, C., et al., System architecture of a wireless body area sensor network for ubiquitous health monitoring. Journal of mobile multimedia, 2006. 1(4): p. 307-326.
[21] Tang, B., Chen, Z., Hefferman, G., Pei, S., Wei, T., He, H., & Yang, Q. (2017). Incorporating intelligence in fog computing for big data analysis in smart cities. IEEE Transactions on Industrial informatics, 13(5), 2140-2150. https://doi.org/10.1109/tii.2017.2679740
[22] Tang, B., Chen, Z., Hefferman, G., Wei, T., He, H., & Yang, Q. (2015, October). A hierarchical distributed fog computing architecture for big dataanalysis in smart cities. In Proceedings of the ASE BigData &SocialInformatics 2015 (p. 28). ACM
[23] Alzaghoul, E., & Bahsoon, R. (2013, May). CloudMTD: Using real options to manage technical debt in cloud-based service selection. In 2013 4th International Workshop on Managing Technical Debt (MTD) (pp. 55-62). IEEE. https://doi.org/10.1109/mtd.2013.6608680
[24] Alzaghoul, E., & Bahsoon, R. (2013, December). Economics-driven approach for managing technical debt in cloud-based architectures. In Proceedings of the 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing (pp. 239-242). IEEE Computer Society. https://doi.org/10.1109/ucc.2013.49
[25] Alzaghoul, E., & Bahsoon, R. (2014, April). Evaluating technical debt in cloud-based architectures using real options. In 2014 23rd Australian Software Engineering Conference (pp. 1-10). IEEE. https://doi.org/10.1109/aswec.2014.27

7 Authors

Mais Haj Qasem and **Esra Alzaghoul** work for The University of Jordan at Amman in Jordan

Alaa Abu-Srhan and **Hutaf Natoureah** work for The Hashemite University at Amman in Jordan

Article submitted 2018-10-27. Resubmitted 2019-10-22. Final acceptance 2019-10-22. Final version published as submitted by the authors.