Toward a Real-Time Framework in Cloudlet-Based Architecture

O. Kotevska  
them CNRS, LIG, University Grenoble Alpes, Grenoble F-38000, France.

A. Lbath  
them CNRS, LIG, University Grenoble Alpes, Grenoble F-38000, France.

S. Bouzefrane  
them CEDRIC Lab, Conservatoire National des Arts et Metiers CNAM, Paris 75003, France.

Follow this and additional works at: https://tsinghuauniversitypress.researchcommons.org/tsinghua-science-and-technology

Part of the Computer Sciences Commons, and the Electrical and Computer Engineering Commons

Recommended Citation
O. Kotevska, A. Lbath, S. Bouzefrane. Toward a Real-Time Framework in Cloudlet-Based Architecture. Tsinghua Science and Technology 2016, 21(1): 80-88.
Toward a Real-Time Framework in Cloudlet-Based Architecture

O. Kotevska*, A. Lbath, and S. Bouzefrane

Abstract: In this study, we present a framework based on a prediction model that facilitates user access to a number of services in a smart living environment. Users must be able to access all available services continuously equipped with mobile devices or smart objects without being impacted by technical constraints such as performance or memory issues, regardless of their physical location and mobility. To achieve this goal, we propose the use of cloudlet-based architecture that serves as distributed cloud resources with specific ranges of influence and a real-time processing framework that tracks events and preferences of the end consumers, predicts their requirements, and recommends services to optimize resource utilization and service response time.

Key words: smart city; cloudlet; prediction; recommendations; framework; real-time

1 Introduction

The concept of smart environments evolves from the definition of ubiquitous computing, which, according to Mark Weiser, promotes the ideas of “a physical world that is richly and invisibly interwoven with sensors, actuators, displays, and computational elements, embedded seamlessly in the everyday objects of our lives, and connected through a continuous network”[1]. This is a general idea from which the smart city concept has evolved. The vision of the future smart city is of a city with a pervasive overlay of Information and Communication Technologies (ICT) connecting things, organizations, and people (e.g., sensors that connect cars to transportation management centers that analyze day-to-day traffic flow data and provide solutions to what-if scenarios in case of events or accidents[2]); this is known as the Internet of Things (IoT).

The IoT connects various objects, such as people, data, smart objects, and processes in networks of billions or even trillions of connections. These connections create vast amounts of data, some of which we have never been accessible before[3]. Because these objects have embedded intelligence, the device itself can filter out relevant information and even apply analytics to enable real-time decision making. Thanks to these enabling capabilities, the IoT will completely change the type of services that are offered and also how they are offered[2]. Services will be available to interact with these objects over the Internet, and query their state and any information associated with them. These new services provided over the Internet are collectively known as the Internet of Services (IoS)[3]. The main goal of IoS is to present everything on the Internet as a service, including software applications, platforms for developing and delivering these applications, and underlying infrastructures. In this scenario, cloud technology plays an important role in enabling IoS deployment, because it comprises different provisioning models for on-demand access to applications, platforms on which developers can build services and applications, and elastic computing infrastructures.

The main goal of cloud computing is to make a better use of distributed resources, combine them to achieve higher throughput, and solve large-scale computation problems. This means that a mobile device can execute a resource intensive application on a distant
high-performance server or computational cluster and support thin client user interactions with the application over the Internet. The mobile device functions as a thin client, with all complex computations being conducted in a nearby cloudlet, which is a trusted, resource-rich computer or cluster of computers that is well-connected to the Internet and available for use by nearby mobile devices[4]. Figure 1 shows the general overview of cloudlet-based architecture.

At present, the use, prioritization, and selection of resources is a major concern in the design of solutions based on emerging technologies. The focus of this study is to create a framework based on cloudlet architecture for a dynamic and real-time predictive model for mobile user potential requirements.

The rest of the paper is organized as follows. Section 2 describes related works. Section 3 presents our proposed framework for smart cities based on cloudlet architecture, while Section 4 illustrates the proposed solution under different scenarios. Section 5 is dedicated to a detailed case study and Section 6 contains the conclusions and plans for future work.

2 Related Work

Recently, the concept of Smart Cities has become increasingly attractive for research, academic, and industry communities. There are factors that determine the different approaches to solve some of the challenges posed by Smart Cities, e.g., how to handle and analyze Big Data from social and physical sensors, how to make life in Smart Cities easier for users, or how users can select services that will be useful for them.

Large companies in the ICT sector, such as Fujitsu, Cisco, and IBM, are strongly involved in and have been sharing their vision for the Smart City[2,5,6]. Following are some related works in this domain.

Similar to participatory sensing, Mobile Crowd Sensing (MCS)[7] uses the power of citizens for large-scale sensing. It goes beyond participatory sensing by having implicit and explicit participations and collects crowdsourced data from both mobile sensing and mobile social network services. The authors characterized the key features of MCS, including crowd-powered data collection, cross-space data mining, and low-quality data analysis. To facilitate the development of MCS apps, they proposed a framework and discussed the efforts of balancing human and machine intelligence in MCS systems.

A semantic service composition architecture, method, and algorithm has been presented in the context of Web of Objects (WoO)[8]. Ontological relations among objects, services, and rules are described to create new services dynamically. Services are described in the knowledge-base of the ontology layer, and rules are generated to perform reasoning tasks for automatic service discovery and processing. The system is capable of natural language processing, which allows users to determine their requirements for inspection and repair. The authors also implemented an object assembler and a service composer for application to match services.

Anteater[9] is a service-oriented architecture for data mining that relies on Web services to achieve extensibility and inter-operability. It offers simple abstractions for users and supports computationally intensive processing on large amounts of data through massive parallelism.

All these approaches provide similar solutions, i.e., they process the received data offline. In this work, we expand this approach by adding a framework for a real-time prediction model. We consider user content, profiles, and trends from social sensors (social media) to create timely predictions and context-aware recommendations using machine learning and data mining techniques.

3 Proposed Approach

In order to overcome the limitations of existing recommendations, we propose a framework for a Smart Environment aimed at allowing unlimited connectivity for end consumers to their context aware services through any accessible device as well as allowing real-time prediction of user requirements.

3.1 Actors in the system

The actors in the proposed system are cloud, cloudlet,
end consumer, and user, defined as follows:

The cloud acts as a federated cloud for a defined set of cloudlets. It offers better performance than cloudlets and can be rapidly provisioned and integrated with the cloudlets with minimal management effort.

A cloudlet is the middle tier in our three-tier architecture model. It uses less memory but offers less performance than the cloud, and its function is to act as a local cloud for some territory (this means each cloudlet has an area of influence). This approach decreases requests and accesses to the federated cloud.

End consumers are devices receiving output results from the processing units. The definition includes smart objects and Cyber-Physical Systems (CPS). Smart objects are any type of user interface with the capability of receiving commands via voice or touch pad and returning the response. Examples of Smart Objects include mobile devices, Google Glass, smart watches, and smart cars. Cyber-physical systems are devices that have the capability to receive commands such as billboards and alarm systems.

User is a person who uses smart objects for their activities.

3.2 System architecture

In our proposed framework, we consider a collection of interconnected smart objects within a smart city. Based on the scope introduced in Section 1, we use a reference system architecture[10] that connects smart objects through wireless technology using cloud principles as in Fig. 2.

To meet the requirement of real-time interactive response (i.e., low latency) for the services that require it, instead of relying on a distant cloud (as measured by the latency), we might be able to address the limitations of a mobile device’s resource using a nearby, resource-rich cloudlet. If no nearby cloudlet is available, the mobile device can gracefully degrade to a fallback mode that involves a distant cloud or, in the worst case, rely solely on its own resources providing limited functionality (with full functionality and performance returning when the device subsequently discovers an available nearby cloudlet).

This system architecture is divided into four levels of communication between the actors in our system.

- The first level is the communication between user(s) and smart object(s) by employing a user interface for interaction.
- The second level comprises the communication between all types of end consumers (smart objects, CPS, and physical sensors) with various cloudlets. This level uses LAN and Wi-Fi technologies for communication.
- The third level of communication happens between cloudlets and the Federated cloud and between cloudlets. As we see, only cloudlets can communicate directly with the cloud. This level uses Internet protocols for communication.
- The fourth level of communication takes place between “social sensors” and the federated cloud, using Internet protocols. The federated cloud contains the processing unit, data management systems and mechanisms, analytic engines, and service deployment and composition.

Cloudlets, originally motivated by narrow end-to-end latency, can play a much broader foundation role in mobile computing. A cloudlet can be viewed as a surrogate or proxy of the real cloud, located as the middle tier of a three-tier hierarchy: mobile device, cloudlet, and cloud. To meet the need for real-time interactive response with low latency, instead of relying on a distant cloud, data is transferred to a nearby resource-rich cloudlet[4]. Cloudlets are interconnected in a similar way to the topologies used in wireless sensor networks, namely mesh, star, or hub. Each cloudlet contains a service repository and outgoing logic, an end consumer register, Virtual Objects (VO) representation, processing unit(s), and a data management mechanism.

In turn, smart objects contain the user interface for interaction with users and available services.

Cloudlets, smart objects, and CPS devices are considered to be trust worthy, and the network connection between them is of sufficient bandwidth and low latency. All subjects that are included and participate use the common platform to enable
Finally, an important aspect of this architecture is that from the perspective of the end consumer everything is considered as a cloud service. This means that users do not need to be aware of or notified whether they are connected to different cloudlets, or where the services that they are using are executed (in their mobile device, in a cloudlet or in the cloud). They just need to have the opportunity to access and use their services without performance or memory constraints or limitations on time and place, and to have a quick response time. This whole process should be automatic, and complete without user intervention.

3.3 Prediction model for user requirements

As mentioned, a smart city is a complex network of people and things all interdependent, increasingly instrumented, and interconnected. In this system, this means that users will always be connected to a cloudlet, have interaction with IoT, social sensors, and use different types of services. Therefore, there will be significant amounts of data streams, which leads to questions such as:

- How to extract knowledge from data collected from physical and social sensors?
- How to identify the right services to be deployed on the cloudlets based on recommendation?
- How to predict service deployment on the cloudlets in an optimal way?

These questions are the guidelines from which this smart framework for real-time processing is derived. By using new technologies that allow the collection and analysis of data in real-time, we can make rapid decisions and smart predictions. This proposed approach for a smart framework for real-time processing provides intelligent integration, aggregation, and processing of real-time data streams based on clouds.

An important source of information for our proposed framework is social sensors where people share their opinions, experiences, events, etc., on a daily basis. Analyzing this data can give us insight into current and future actions. Especially important is the detection of current situations on social sensors: what is happening in some regions, or what are the people usually doing during a particular time of the day or year? Is there some unexpected event that people going in a given direction need to be notified? Based on these questions, our framework provides answers by collecting social sensor data streams, analyzing them in real-time, and detecting “significant trends”. In this way, when an event or situation occurs, all users affected by this event will be notified rapidly. This is especially important in the case of natural disasters (flash flood, fire, etc.) or traffic jams which require preventive action by the users. There will be knowledge-bases with known patterns classified into different categories (such as sport events, art performances, safety, etc.), but it will also be possible to identify patterns generated by unknown events or situations. Knowledge discovery in order to obtain timely information is a very important component of this proposed framework. Figure 3 shows the complete architecture of the smart framework for real-time event processing based on cloudlet architecture.

The information generated by the Pattern Discovery unit is one of the parameters used for predicting user actions and creating service recommendations. The Recommendation unit utilizes user profiles, geo-location history, and contextual information to understand what the user is interested in. When a relevant event takes place in the city (based on the information received from the Pattern Discovery unit), the user will receive recommendation notifications about it.

The Prediction unit uses the previously mentioned user data and knowledge discovery information to forecast what will be the next user action, for instance, which path they will take if there is a traffic jam, which concert they will attend, when will they continue watching an online movie, etc. Prediction actions are used to predict what the next user steps will be and to discover how the user is deciding which action to take.

The idea of this proposed framework is to be independent from communication technologies and protocols. As previously mentioned, these units have

Fig. 3 Framework architecture for real-time processing.
a service-oriented access, and can be approached as a single service.

These will be deployed to the cloudlets, and be accessible to every user.

4 General Scenarios

In this section we describe cases that would benefit from this proposed framework and explain how the user can interact and benefit from it.

4.1 Scenario 1: Event detection

Information collected from social sensors can significantly contribute to timely inform about disruptive and preventive actions in people’s routines. Social media is a source of information for situations and facts related to the users and their social environment. When urgent situations (like a natural disaster) arise, this information needs to be collected, analyzed, and mined as soon as possible, in order to inform the population in a timely manner. In a smart city, the real-time event detection processing engine is listening for data streams gathered from different geo-locations in the city, and using this information the processing unit can then provide information about what is going on in each part of the city in real-time. If there is some social event like a concert or an art event, people with compatible interests (based on the user’s context, profile, and history) will be informed, so they can attend the events or follow the associated social media streams. Similarly, if some natural disaster occurs (like fire, earthquake, etc.), users in that area or heading in that direction (as gathered from their location and context) will be timely informed. This kind of event detection acts as preventive action for user’s safety. Figure 4 depicts the process step actions.

For this scenario, as already described, streams of data are collected in real-time from different areas and sources, then filtered, cleaned, and normalized resulting in a standard form ready for analysis. If the data shows a known pattern, it will be processed immediately. However, if it doesn’t, it will be processed by unknown pattern discovery algorithms that attempt to extract patterns. This process is integrated with a learning system that takes into account previously unknown events. Output data received from this phase is ready for semantic analysis which makes sense of the received data stream. The semantic analysis phase is very important in this step because its output is used to determine if some event happened or not in a particular area. In the case of a positive output, users affected or interested in it will be informed. A concrete example of this process would be real-time streams of Twitter data from New York City. The streams are pre-processed and forwarded to the processing phase. If this phase identifies a fire at 7th Avenue and 52nd Street, the process of notifying users in or heading to that area will be triggered. This information is later used by the recommendation and prediction units.

4.2 Scenario 2: Predicting user actions and content recommendation

In the smart city environment, where users are accessing multiple services simultaneously, it is very important to predict their actions. The system can be pre-populated with information collected by other means, such as habits, preferences, context, locations, service usage history, and event detection data. Building on this, our proposal uses these sources of information to predict the user’s actions. This functionality is part of the real-time event detection unit, shown in Fig. 5. The user profile information such as age, education, hobbies, favorite books, artists, movies, and habits are saved in a database system with respect to privacy. This information is used to construct a baseline for determining what the user preferences are. This baseline is complemented by other information sources, such as service registration (containing all the existing services and their meta-data), or the previously mentioned social sensors. Based on this information (the user’s movements, time of year, day, or week), the prediction unit uses data mining algorithms to predict the next activity or event that the user may want to take part in. For example, if a given user in the New York City area frequently reads news about tennis and the event detection unit
identifies that people are attending the US Open Tennis Championship, the system determines that the user will be interested in attending. Recommendations are especially important in this Smart City scenario where a large amount of very different services exist as it may be very difficult for end users to find appropriate services, so mechanisms are needed to help users find suitable cloud services. The Recommendation unit is also part of the real-time processing unit, shown in Fig. 5.

It uses the same source of information as the Prediction unit (user profile data and information obtained from the knowledge unit), but also takes into consideration the user’s context before providing recommendations and tuning the output scenario. Context-aware recommendations are designed to react to a user’s context without their intervention. It consists of two parts: sensing a context scenario and adapting the system to a changing scenario by providing desired services for the user. As an example let’s say that on weekdays the user requires lunch recommendations for places with quick service as the time for the lunch break is limited, but on weekends the suggestions provided should be for family friendly restaurants.

4.3 Scenario 3: Service continuity, distribution, and accessibility on cloudlets

Adding continuous data providers as services provides value-added information, with various levels of granularity, that best suit the users’ requirements and interests: concert events, sport events, transport providers, local and regional planning authorities, etc. Users are offered the option to pause the services then continue to use them later, resuming from the previous state. While this characteristic is more useful for services that provide video or music streaming or file downloads, it can also help users to monitor the progress of events, such as utility restoration after a severe thunderstorm or traffic congestion on certain roads so that alternative routes can be planned (e.g., someone going to the airport on a Sunday).

Services are distributed among cloudlets depending on the frequency of usage, the region specification, users’ preferences, and/or time of year. Some of the services are more attractive for some region(s) of the city during a particular time of the year. For example, if there is a sporting event, such as the aforementioned US Tennis Open in New York at the end of August and beginning of September, during this period there is a possibility that more users will use services related to the event (ranging from information about the event to traffic reports). When the event is over, some of the services will be no longer distributed on all the cloudlets, but the users will still be able to find those services.

Service accessibility, as shown in Fig. 6, means allowing access to all the services when the user requires them without constraint. Accessibility is also concerned with the mechanism of finding the right services with ease. As there are multiple services available, users need to be provided with search mechanisms that use a simple interface and timely results. By doing this, our system acknowledges and uses the inherent granularity of the architecture to optimize the interaction with the end user, allowing, among others, approximate searches, spatial synonyms, and complex information retrieval, thereby surpassing the limitations of traditional, query-based systems.

5 A Case Study: Cloud Connected Vehicles

As stated in Ref. [11], among the research work regarding smart cities, we took note of smart transportation that relies on sophisticated infrastructures to overcome situations like traffic congestion’s, emergencies, and accidents. In this paper,
we consider a vehicular infrastructure built on mobile cloud computing in order to achieve more efficient transportation.

Connected vehicles can be viewed as entities that have not only access to the cloud or cloudlet but are also equipped with sensors and actuators capable of collecting data and actuating some vehicle components. As shown in Fig. 7, each cloudlet covers an area defined by a limited distance traveled by vehicles.

In addition to the data collected by vehicle sensors being used for machine-to-machine purposes, such as maintaining a minimal distance between the vehicles in order to avoid accidents, some of this data (such as speed, location, etc.) can be sent by the vehicles to the cloudlet for it to build a view of the local traffic in real time. These data and others gathered by the cloud generate a global view of traffic in order to provide recommendations to drivers regarding the weather, congested routes, etc. In addition to the data provided by physical sensors, helpful when making recommendations to the drivers, other data may be supplied by social sensors such as providing relevant information on events related to the covered area.

Mobile devices are also useful entities that enable drivers not only to get data from the cloudlet or cloud but also to externalize services and data on the cloudlet. Because of the displacement of vehicles moving from one cloudlet coverage to another, data must be processed in real time. Hence, the data, as well as the tasks that process it, must meet temporal constraints. In fact, data are collected periodically and each must be consumed/processed by a task before generating new data, otherwise the preceding collected data becomes obsolete.

In order to manage the services and data of the vehicular infrastructure in real-time, our scenario is based on the following features.

### 5.1 Data collection by physical sensors

The data collected by the physical sensors are processed by application tasks executed within smart objects such as user mobiles or vehicle devices. Because of their limited resources, smart objects are mono-processor devices that host a real-time kernel to periodically execute application tasks because of periodic data sensing. These tasks are characterized according to temporal parameters as defined by Ref. [12]. In fact, each periodic task $T_i$ in real-time systems is defined with four parameters $(r_i, C_i, P_i, D_i)$ where:

- $r_i$ is the arrival time of $T_i$.
- $C_i$ is the execution time of $T_i$, i.e., the time used to process the collected data.
- $P_i$ is called the period, and it is the time interval that separates two successive data collections, hence two instances of $T_i$ occur.
- $D_i$ is the hard deadline of $T_i$. $dD_i = r_i + D_i$ is the absolute deadline, a time after which the execution of $T_i$ is obsolete. Generally, $D_i$ is equal to $P_i$.

Each smart object that collects distinct types of sensor data can schedule its corresponding tasks using a fixed-priority scheduling algorithm called Rate Monotonic, in which the priority is calculated using the tasks’ periods. Hence, the higher the priority of a task, the lower is its period.

According to Ref. [12], each configuration of $n$ periodic tasks, with $P_i = D_i(V_i)$, is schedulable using Rate Monotonic, if the following condition is satisfied:

$$\sum_{i=1}^{n} \frac{C_i}{P_i} \leq n(2^{\frac{1}{n}} - 1) \ (1)$$

In other words, to process data in real-time, the corresponding tasks must meet the temporal constraints dictated by Eq. (1). Another algorithm proved to be optimal is the Earliest Deadline First (EDF) which schedules real-time tasks according to their absolute deadline. In this case, the highest priority is assigned to the task with the nearest deadline. Reference [12] proved that a configuration of $n$ periodic tasks, with $P_i = D_i(V_i)$, is schedulable using EDF, if and only if, the utilization factor $U$ satisfies the following condition:

$$U = \sum_{i=1}^{n} \frac{C_i}{P_i} \leq 1 \ (2)$$

### 5.2 Data aggregation in the cloudlet

Because of the movement of vehicles, data must be processed in real-time to provide recommendations in real-time, such as suggesting an efficient direction for a vehicle regarding its desired destination.

To supply such recommendations to the driver, either tasks or data can be offloaded from vehicles to the cloudlet in order to be aggregated with other data from

![Fig. 7 Vehicle infrastructures.](image-url)
other vehicles, thereby building a local view of the covered area in terms of vehicle-to-vehicle interactions and moving-vehicle states. Data issued by the in-vehicle sensors can be enriched with social data provided by the cloud. To manage these distinct data types, the cloudlet needs to host an ordinary guest operating systems such as Windows or Linux, as well as a real-time guest OS. The real-time VM hosts a Real-Time Operating System (RTOS) to handle real-time tasks while other VMs are dedicated to general-purpose services such as those related to social networks. Hence, tasks that process the sensing data must be scheduled and executed by the real-time VM to allow the tasks to meet their timing constraints. The real-time VM handles real-time applications by executing the application tasks according to a real-time strategy such as EDF described above. However, the deadline cannot be guaranteed to be met unless the cloudlet hosts a real-time hypervisor. The use of this type of hypervisor is motivated by the fact it schedules VMs according to a real-time scheduling algorithm based on a fixed priority. The highest priority is assigned to the VM that hosts a real-time guest OS while lower priorities are assigned to the other VMs. Hence, an ordinary VM is launched by the real-time hypervisor only if there are no real-time tasks to execute within the real-time VM.

Consequently, there are two scheduling levels:

- The first level concerns the real-time hypervisor that schedules VMs and determines the privileges of the real-time VM (that hosts RTOS) among the others. The scheduling strategy of the hypervisor can be based on a fixed-priority scheduler.
- The second level is related to the scheduling algorithm provided by RTOS that can use EDF to schedule tasks. According to the criticality of the manipulated data, we can distinguish two categories of real-time tasks:
  - Hard real-time tasks that must process data on time because a missed deadline could trigger a system disaster such as an accident.
  - Firm real-time tasks that can miss their deadline infrequently, but degrade the system’s quality of service.

Regarding this classification, we can consider all the tasks that deal with in-vehicle sensing data as hard tasks while tasks related to social sensors can be viewed as firm tasks. The hard tasks can be either executed periodically by the vehicle’s smart objects or externalized to the nearest cloudlet. Whichever execution environment is used, these tasks must be handled using a real-time scheduling strategy. In contrast, the firm real-time tasks can be defined according to an aperiodic task model which processes social data provided by the cloud to the cloudlet.

Each aperiodic task \( A_i \) is modeled thanks to a triple \( (r_i, C_i, D_i) \) where \( r_i \) is the time that \( A_i \) is ready to execute, \( C_i \) is the execution time of \( A_i \), and \( D_i \) is its deadline.

Aperiodic tasks handled by the cloudlet can either be executed within the real-time VM (RTOS) during processor slack time or within an ordinary VM. To verify if RTOS can execute the periodic and the aperiodic tasks while meeting their temporal constraints, the utilization rate of the processor must be lower than 1 as in Eq. (3).

\[
U = \sum_{1}^{n} \frac{C_i}{P_i} \leq 1, \quad \forall T_i \tag{3}
\]

where \( T_i \) is a periodic or aperiodic task.

5.3 Data mining and learning methods on the cloud

The cloud can gather data from distinct cloudlets regarding vehicle movement and other events like congestion and accidents to design a global view of the traffic. This view is also enriched with other data obtained from social networks. Learning methods, such as context recognition, can be used to compare current situations with previous ones and help the system be more adaptive and reactive.

5.4 Recommendations

The system, based on its distributed infrastructure as well as its cooperative tasks for getting accurate information on traffic, can issue some directives to drivers in terms of the most efficient direction to take or a travel route to avoid social events around a recommended itinerary, etc.

6 Conclusions and Future Work

In the Smart City context, integrating information from different sources is a necessity in real-time to make smarter decisions for individual citizens and for the city as a whole.

In this paper, we propose an approach for a framework, based on prediction models, that provides integration, aggregation, and processing of real-time tracking data based on cloudlet architecture and cloud principles. By adopting a service-oriented approach when collecting, integrating, storing, and intelligently analyzing social and user data, we achieve better granularity, accessibility, recommendation, and prediction of user requirements. In addition, our
approach provides service continuity capabilities that allow the system to provide value-added information that suits the users’ requirements at any time and place.

The next steps involve the implementation of the proposed framework by realizing the scenarios presented here and using existing standards in the field.

In order to evaluate this implementation, we will build a prototype to conduct user performance requirements, using multiple metrics that measure request-response time, performance, processing time, and other relevant measures.

Acknowledgements

This work was supported by the National Institute of Standards and Technologies (NIST), and conducted within a collaboration under Information Technology Laboratory, Advanced Network Technologies Division (ANTD) and the Universities of Grenoble and CNAM. Our special thanks go to Dr. Abdella Battou, ANTD division chief for his support and advises.

References

[1] M. Weiser, R. Gold, and J. S. Brown, The origins of ubiquitous computing research at PARC in the late 1980s, *IBM Systems Journal*, vol. 38, no. 4, pp. 693–696, 1999.

[2] R. Y. Clarke, Smart cities and the internet of everything: The foundation for delivering next-generation citizen services, IDC Government Insights #GI243955, 2013.

[3] L. Heuser, C. Alsdorf, and D. Woods, eds. *International Journal of Distributed Sensor Networks*, vol. 20, no. 4, pp. 145–151, 2014.

[4] C. L. Liu and J. W. Layland, Scheduling algorithms for multiprogramming in a hard real-time environment, *Journal of the ACM*, vol. 20, no. 1, pp. 46–61, 1973.

[5] D. Guedes, W. Meira, and R. Ferreira, Anteater: A service-oriented architecture for high-performance data mining, *IEEE Internet Computing*, vol. 10, pp. 36–43, 2006.

[6] S. S. Ara, Z. U. Shamszaman, and I. Chong, Web-of-texts based user-centric semantic service composition methodology in the internet of things, *International Journal of Distributed Sensor Networks*, vol. 2014, p. 482873, 2014.

[7] S. Dirks, C. Gurdgiev, and M. Keeling, Smarter cities for smarter growth, IBM Global Business Services, 2010.

[8] O. Kotevksa is a PhD student at Joseph Fourier University (UJF) in Grenoble, France and Guest Researcher at ITL Lab NIST, Washington DC Metro, USA. She received the MSc and BSc degrees from the University “Ss Cyril and Methodius” in Skopje, Macedonia in 2013 and 2008, respectively. Her research is in the field of text mining, machine learning, cyber physical systems, and software design.

[9] S. Bouzefrane is an associate professor and has the accreditation to conduct research (HDR) at the Conservatoire National des Arts et Metiers of Paris. She received her PhD degree in computer science from the University of Poitiers (France) in 1998. After four years at the University of Le Havre (France), she joined in 2002 the CEDRIC Lab of CNAM where she worked on real-time and embedded systems. She is the co-author of many books (Operating Systems, Smart Cards, and Identity Management Systems). Her current research areas cover service architecture and security in mobile cloud computing, internet of things, and virtualization.

Ahmed Lbath is a full professor of computer science at University of Grenoble1 (MRIM/LIG Laboratory), and he is also conducting research in collaboration with ITL Lab NIST in Washington DC metro area where he carried out research activities as visiting professor. He is the IUT Deputy Director, former Head of CNS Department, and former Director of R&D in a French company. He received his PhD degree in computer science from the University of Lyon and hold an “Habilitation Diriger des Recherches”. He is currently acting as project manager coordinating scientific collaborations in the domain of Cyber Physical Systems and smart cities. His research interests include cyber physical human systems, smart cities, mobile cloud computing, recommendation systems, web services, GIS, and software design. He published several patents, papers in books, journals, and conferences and he regularly serves as co-chair and/or member of several committees of International conferences, journals, and research programs.