Benefits of a fog-to-cloud approach in proximity marketing

Antonio Salis[1], Glauco Mancini[1], Roberto Bulla[1], Paolo Cocco[1], Daniele Lezzi[2], and Francesc Lordan[2]

[1] Engineering Sardegna Srl, Loc. Sa Illetta, SS195 km 2,3 I-09123 Cagliari, Italy
{antonio.salis,glauco.mancini,roberto.bulla,paolo.cocco}@eng.it
[2] Barcelona Supercomputing Center(BSC), Spain
{daniele.lezzi,francesc.lordan}@bsc.es

Abstract. The EC H2020 mF2C Project is working to the development of a software framework that enables the orchestration of resources and communication at fog level, as an extension of cloud computing and interacting with the IoT. In order to show the project functionalities and added-values three real world use cases have been chosen. This paper introduces one of the mF2C use cases: Smart Fog Hub Service (SFHS) use case, in the context of an airport, with the objective of proving that the adoption of the fog-to-cloud approach brings relevant benefits in terms of performance and optimization of resource usage, thus giving an objective evidence of the impact of the mF2C framework.

Keywords: cloud computing · fog computing · fog-to-cloud · distributed systems · IoT · proximity marketing · 3G · 4G/LTE · Wi-Fi.

1 Introduction

By 2020 the installed base of the Internet of Things (IoT) devices is forecasted to grow to almost 31 Billion worldwide, with an annual economic impact of $3.9T to $11.1T by 2025 [4]. The forecast scenario includes diverse settings and use cases including factories, cities, retail environments, and healthcare. At the same time, 50% of IoT spending will be driven by discrete manufacturing, transportation, logistics, and utilities where predictions say that IoT will have the most transformative effect on industries that are not technology-based today. The most critical success factor of all these use cases depend on secure, scalable and reliable end-to-end integration solutions that encompass on-premise, platforms [1], legacy and cloud systems. While consumer applications will attract the most attention and create significant value, Business-to-Business (B2B) applications will generate nearly 70% of potential value enabled by IoT [3]. Also, in the airport industry an increase in the number of passengers is foreseen, where more than 4 billion passengers will concentrate in the airports with an average of two connected devices for each passenger [2]. Current technology infrastructures and architectures have not been designed to process in real time the great amount of information data that is being made available from such a number of devices with a so high concentration. As one of the first technical response to such technological challenges fog computing is emerging as an architectural model that places itself between the cloud and the IoT, in the Cloud-to-Things Continuum. In this paper we present a service implemented following the fog-to-cloud approach of the mF2C project, that acts as a smart hub to provide real time information in public environments. A prototype is being implemented in the context of an airport testbed that collects data from the passengers and provides information to enable proximity marketing (shops, restaurants, etc) as well as analytics computed in the cloud. This setting could be reused in other public domains like train stations, shopping centers, etc. This paper describes the design possible scenarios, the networking particular elements of this use case, and some tests on different design scenarios that have been run. The results of these tests demonstrate that the capabilities and performance obtained by the mF2C adoption overcome the other possible choices, fulfilling the real-time requirement, enabling the distribution of processing of data, reducing traffic load and latency between cloud and hub. This paper is structured as follows. Section 2, introduces the mF2C Smart Fog Hub in an airport use case. Section 3 describes the comparison of the potential technical solutions, and Section 4 describes the experimental results and related benefits coming from the fog-to-cloud approach. Finally, Section 5 concludes the paper.

2 Fog Hub in Airport Use case

The EC Horizon 2020 program in 2016 has funded a new research initiative (mF2C)[3] bringing together relevant industry and academic players in the cloud arena, aimed at designing an open, secure, decentralized, multi-stakeholder management framework for F2C (Fog-to-Cloud) computing, including novel programming models, privacy and security, data storage techniques, service creation, brokerage solutions, SLA policies, and resource orchestration methods [7] [8]. There is an increasing demand on evaluating and identifying new market sectors and opportunities, and interest at the IoT evolution as a potential arena where current commercial cloud services offering could be enriched and differentiated. In this perspective a relevant focus in setting up hubs in public environments (e.g. airports, train stations, hospitals, malls and related parking areas) is suggested, capable of tracking the presence of people and other objects in the field, and developing added value services for proximity marketing, prediction of path/behavior of consumers, and taking real time decisions. This kind of environments can be implemented with a recommendation system, in order to produce a new and personalized pleasant experience for end-users.

The foreseen hub can be easily considered as a fog environment that embeds cloud connectivity to either process large amount of data or request extra-data, perhaps data coming from other fogs located in near sites (e.g. airport, train/main bus/ harbor station), and that could interact sharing data and cus-

[3] http://www.mf2c-project.eu

tomer behavior gathered to improve the effectiveness of marketing proposals, given that the identity of objects/customers is protected.

This scenario has been named as the Smart Fog-Hub Service (SFHS). The use case is experimental and extends the concept of a "cloud hub to a new concept of fog hub", driven by real market needs [9].

The system is under development in the Engineering Labs, and will be moved to the Cagliari Elmas Airport in 2019. In the final configuration the fog elements will be positioned in the field in order to create a grid for Wi-Fi coverage.

The field (Figure 1) includes check-in area, security control area, lounges and departure gates. Check-in and departure gates host several shops and other frequented places like bars and restaurants [4].

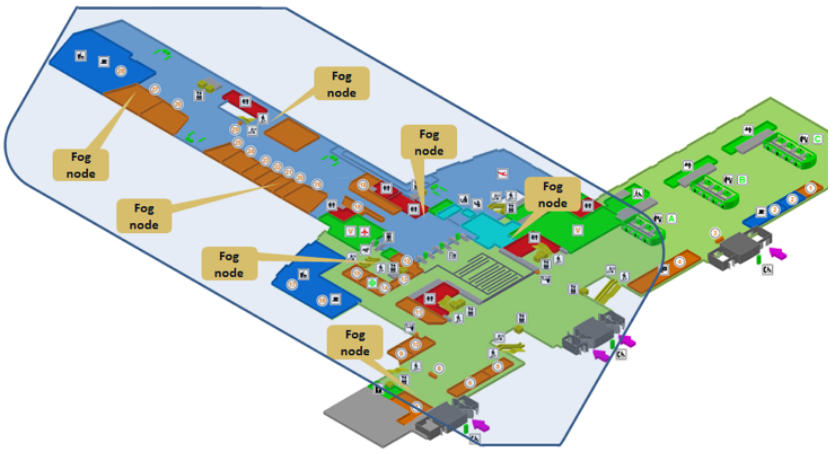

Fig. 1. Use case scenario in the airport.

According to the current architecture specifications, the system has the following elements, as depicted in Figure 3:

– A cloud layer, based on a OpenStack[5] instance, wired connected with the fog layers, that provides scalable computing power for machine learning algorithms used for the recommendation system.

[4] http://www.cagliariairport.it
[5] https://www.openstack.org

– A first fog layer, which acts as aggregator, based on a NuvlaBox mini[6], equipped with 8 GB RAM, that provides real-time computing and storage resources to the edge elements.
– A second fog layer, which acts as access node, based on six RaspberryPi3[7] with 1 GB RAM, that provide session management and fast response to the edge devices.
– Android smartphones at the edge, connected to the access node with Wi-Fi, and using an Android app to interact with the system; in this phase they are used as data generator.

A core element of the system architecture is constituted by the mF2C framework, able to manage and coordinate the orchestration of all existing and potentially available resources, from the edge up to the cloud, when executing a service, according to the service requirements and user needs. This is structured in a hierarchical architecture (Figure 2), where resources are grouped into layers, and an mF2C agent entity deploys the management functionalities in every component within the system. In practical scenarios there are different layers, from layer 0 at cloud, to layer N as closest to the edge, where the mF2C agent runs in all devices capable of supporting it, and participating in the mF2C system. In case of devices not able to run the agent, the related information is collected, processed and distributed by the software agent connecting them to the system. The clustering strategy and leadership election policy is still under development, but includes elements like spatial distance and data connectivity. Additional features of the mF2C system architecture are:

– A fog area (or cluster) is the set of nodes managed by a Leader, with election of a backup node, to be used in case of leader failure,
– Only one node acts as a leader in each fog area, and only one backup, which substitute the leader in case of failure or loose of connection,
– Only IoT devices can be connected to any of the agents in the mF2C system

The whole set of management and control functionalities of the agent is split into two main blocks, the Platform Manager (PM), and the Agent Controller (AC). The PM provides high-level functionalities, and manages the inter-agent communications, with the capacity to take decisions with a more global view. Agent Controller (AC) has a local scope, dealing with local resources and services. At run time, when a service is requested to any of the mF2C agents, the PM is responsible for deciding if this task can be executed in that agent, or forwarded down to any of the agents in the area if the agent is a leader or up to the higher hierarchical layer. If the task is forwarded, the communication is also done through the PMs of the agents. The request is passed to the AC only when an agent can execute the forwarded task, using the agents local resources.

[6] http://www.sixsq.com/products/nuvlabox/
[7] https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

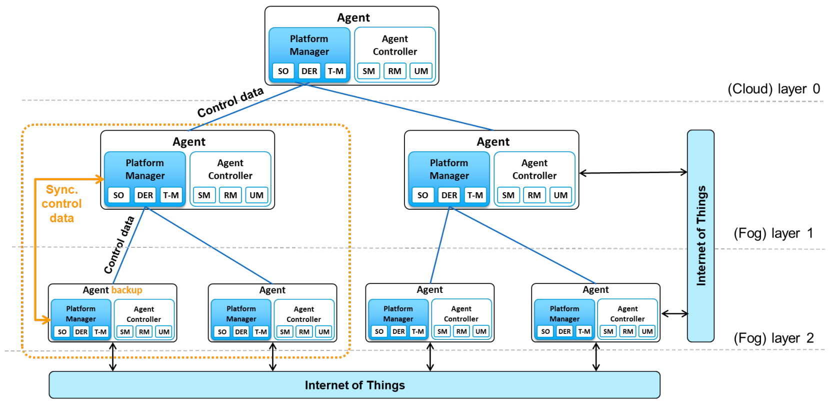

Fig. 2. mF2C architecture for IT-1.

In the use case, an mF2C agent software runs in all cloud and fog elements and provides management and control functionalities. An Android app is installed in the smartphone and implements security and privacy features to preserve managed data both at rest and in transit, with a security level comparable to the ones adopted by the mF2C agent. In particular, the Distributed Execution Runtime (DER) in the mF2C agent is responsible for optimizing services/tasks execution on the available resources. This component is based on the COMPSs [5] framework and orchestrates the execution of the requests coming from the mobile app, to optimally exploit the available computing resources. The tasks generated by the execution of the applications are distributed, in parallel, on the resources selected by other components of the mF2C platform. DataClay in the mF2C agent performs the system data management.

At application level the following business processes have been identified and under development:

– App installation and device registration.
– Position calculation, check for Points of Interest (PoI) & notification.
– Position data sync in fog & cloud.
– Airport events notification (flight call, but also invitation to move closer to the gate).
– Recommendations generation based on user similarities (and recalculations with data caching).
– Reporting (real-time and history) with the dashboard.
– Configuration of Points of Interest (PoI) and promotions (in case of shops).
– Filtering and calculating data in position data streams.

The overall idea is to track and engage all people and objects in the field and use a Collaborative Filtering [8] based recommender system to get the best possible customer experience, with suggestion on the best way to use available

[8] http://recommender-systems.org/collaborative-filtering/

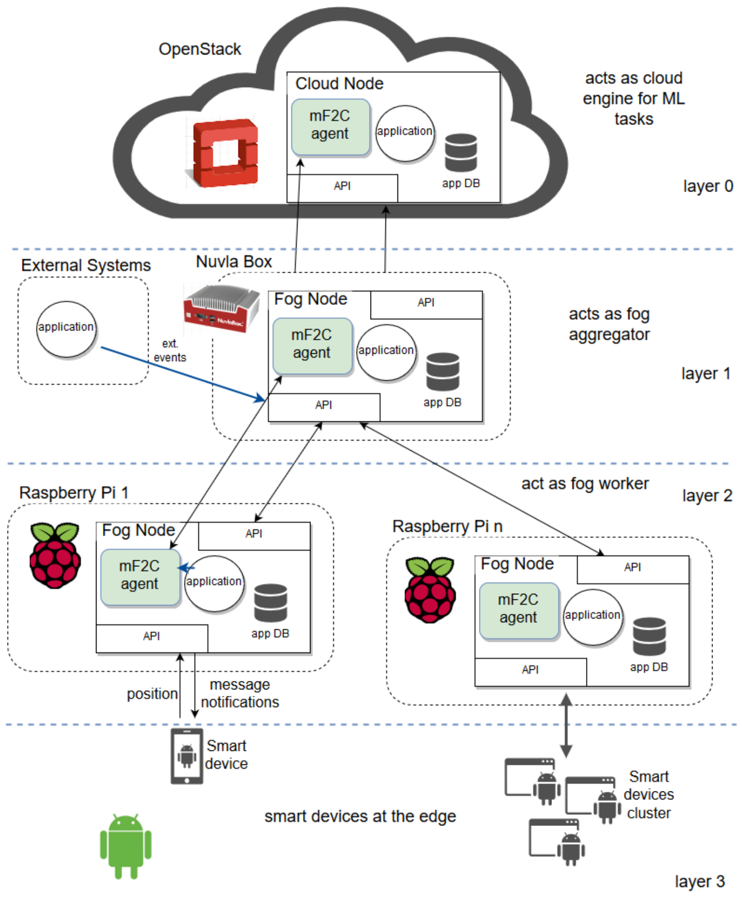

Fig. 3. use case 3 System architecture.

services, e.g. suggest the moment for shorter waiting times in Security Control to departing people, to move close to the gate or notify the final call, or recommend relevant proposals and offerings in shops close to the user. All these suggestions can be refined according to behavior and choices done by passengers.

The recommender system will play a major role for the personalization of the traveler's experience, machine learning based features like users similarity will be used to suggest items that other users liked but the current user has not interacted with yet. A particular care has been provided for the privacy and security of personal data: the recommender system uses algorithms that works perfectly without any personal information.

3 Architecture evaluation

Given the nature of the chosen use case and the business processes listed above, some characteristics emerge in terms of processing demand:

– Real-time requirements for position calculation and check for PoIs nearby.
– Massive calculation in the case of machine learning algorithms for Collaborative Filtering.

While the processing demand from the machine learning can be offloaded using the Fog-to-Cloud approach to ask for more computing power when needed and keep the latest data, the real-time requirements for position tracking and engaging (with notifications) need to be managed immediately, starting from the smartphone owned by the users.

The typical loop is the following:

– Calculate PoIs in proximity, given object position (x, y)
– Notify the user in case of PoIs nearby

This is a small processing that every object runs every few seconds with immediate response, but that requires much higher processing capacity as the number of devices in the field grows.

Since the smartphone is the selected device, the possible scenarios of implementation are:

1. Wi-Fi connectivity and mF2C support.
2. 3G connectivity and direct connection to a public cloud

The first case corresponds to the architecture shown in the previous section, with Wi-Fi connection between the smartphone and RaspberryPi3 devices, with wired connections between the other layers. In this scenario the only network link that requires attention is the Wi-Fi, that will be used in open space; further copper or fiber connections can be ignored. In the other case the smartphone uses 3G networking to connect directly to the public cloud that hosts the relevant services. Here 3G connectivity is the most critical link, while intra cloud communications could be ignored.

Both communication means try to satisfy the growing expectation of ubiquitous connectivity for a broad range of services. At the same time, wireless transmission means are highly variable in terms of bandwidth, latency, battery usage.

The following are the main features of Wi-Fi communications with respect to the proximity processing:

– IEEE 802.11g /n /ac standards are quite adequate to support the real-time requirement and sustain the fast growing number of devices to be managed in the field;

- The newest standards based on 5 Ghz frequency and MIMO (multiple input multiple output) features enable wider bandwidth and faster response times
- Slow start events like first hop do not affect the overall speed of the communication channel;
- The Wi-Fi protocol is the preferred communication protocol for indoor communication and adequate for the proximity processing request;
- Wi-Fi connections are more battery efficient than 3G/4G.

Table 1 summarizes the main features of Wi-Fi current technology.

Table 1. Wi-Fi release history and main features

802.11 protocol	Freq (Ghz)	Bandwidth (Mhz)	Data rate (Mbit/s)	Max MIMO streams	median Latency (msec)
b	2.4	20	1, 2, 5,5, 11	1	6,22
g	2.4	20	6, 9, 12, 18, 24, 36, 48, 54	1	6,22
n	2.4	20	7,2, 14,4, 21,7, 28,9, 43,3, 57, 8, 65, 72, 2	4	6,22
n	5.0	40	15, 30, 45, 60, 90, 120, 135, 150	4	0,90
ac	5.0	20, 40, 80, 160	Up to 866,7	8	0,90

The following are the main features of 3G/4G radio communications [9] with respect to the proximity processing:

- Intermittent network access, like polling, is a performance waste on mobile networks, as it uses the packet based communication mean in the worst way;
- Real time analytics like proximity processing demand high battery usage against all battery optimizations implemented in 3G/4G;
- Intermittent network access carries a large latency cost due to the Radio Resource Control (RRC) state transitions
- In case of an HTTP request like in our scenario, also the DNS, TCP, TLS and HTTP protocols can increase the overall latency in the communication;
- TCP implementation on top of 3G shows poor performances due to the 3-way handshaking nature of TCP and related slow start, in practice it triples response times, compared with native TCP implementation;
- 3G and 4G/LTE, in order to make the best use of bandwidth, have a kernel that waits for data (buffer more requests) to have bigger packets instead of sending smaller packets immediately

Table 2 summarizes the latency of a single HTTP request on 3G/4G network:

From the data presented above Wi-Fi seems to be much better than 3G, only the latest 4G promise to compete with Wi-Fi over peak throughput and latency, use battery in a more efficient way and be more adequate in indoor environments.

4 Experimental results and benefits from mF2C

All information reported seems to determine that the Wi-Fi network connection, together with the fog-to-cloud approach, brings the best performance, thus being

[9] Ilya Grigorik, High Performance Browser Networking, OReilly, 2013

Table 2. Latency overhead of a HTTP request

	3G	4G/LTE
RRC Control plane	200 2500 ms	50 100 ms
DNS lookup	200 ms	100 ms
TCP handshake	200 ms	100 ms
TLS handshake	200 400 ms	100 200 ms
HTTP request	200 ms	100 ms
Total Latency overhead	200 3500 ms	100 600 ms

the preferred solution for the Smart Fog Hub in the Airport for the proximity calculations.

In order to validate this assumption an experimental benchmarking has been defined with a client-server sample with the following characteristics:

– The client simply calls a request with a position (x, y) asking for the list of available Points of Interest within 5 meters,
– The server receives the request, calculates the available PoIs and return a JSON array with them.

This client-server sample has been run in the following environments:

– (Client) XIAOMI REDMI NOTE2 smartphone with Android 5.0.2, and app Rest Api Client for HTTP requests
– (Server 1. option) RaspberryPi3 with ARM Cortex A53 64-bit cpu and 1GB Ram
– (Server 2. option) VM running in a Public cloud on OVH-Paris, with 4 cpu and 4GB Ram, with minimum load. The average ping from the RaspberryPi3 to the VM is about 35 msec.

A VM running a service that provides the PoIs search has been prepared and deployed on both server targets. The available Wi-Fi connectivity is based on 802.11g at 2.4 Ghz.

The following scenarios have been prepared for tests:
(A) Smartphone connects with Wi-Fi to RaspberryPi3, the "Rest Api" app calls the service locally.
(B) Smartphone connects with Wi-Fi to RaspberryPi3, the "Rest Api" app calls the service in the public cloud.
(C) Smartphone connects with 3G to the public cloud and the "Rest Api" app calls the service.
(D) Smartphone connects with 4G/LTE to the public cloud and the "Rest Api" app calls the service.

In every test run a sequence of calls has been performed with a delay of 5 seconds between consecutive calls.

Test results have confirmed the statements of the previous chapter: the mF2C (A) with Wi-Fi connectivity approach over-performed 3G and 4G/LTE network communications. Figure 4 presents the summary of response times collected.

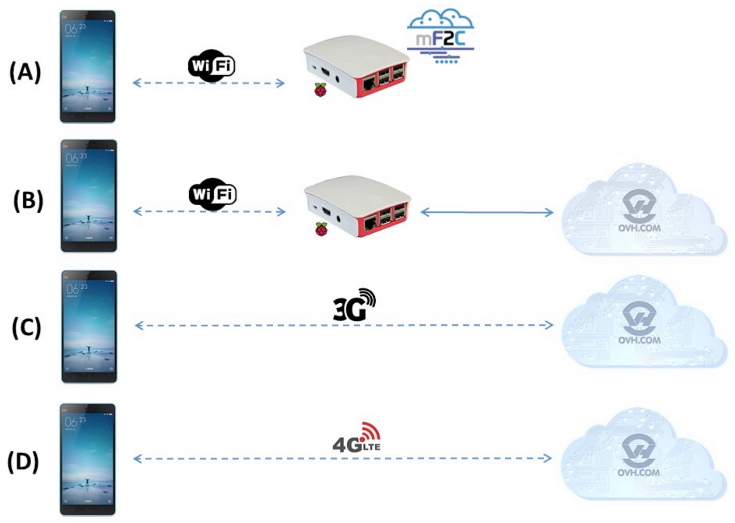

Fig. 4. Test environment in the different approaches.

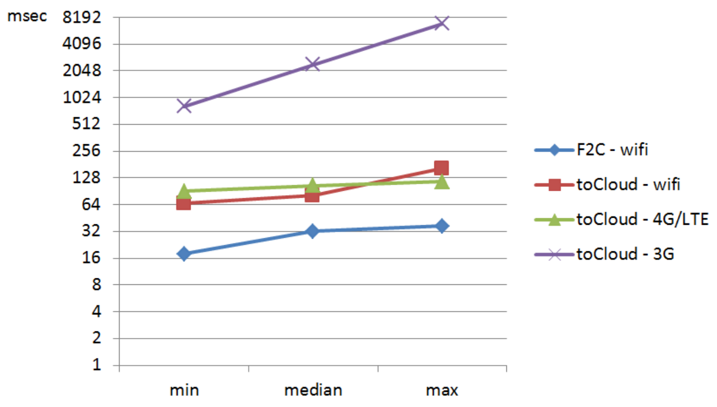

Fig. 5. Comparison of response times with different approaches.

The (A) scenario, currently used for the mF2C project, performed the best, with a pretty stable and fast response. Times are in line with the real-time requirement, but they could be improved with the use of 5.0 Ghz frequencies and latest version that offer both a much wider frequency range and is still largely interference free in most environments. The (B) scenario returned good values, since it benefited from the four-cpu configuration and the minimum cpu load, but with a high latency between the RaspberryPi3 and the Public cloud. Times are 2X compared with the (A) scenario, but still within the real-time requirement. The (C) scenario scored the worst, with a median response time near 3 seconds, which does not fulfil the real-time proximity requirement. The (D) scenario scored the third, just within the limits of the real-time requirement. The sampling showed a very stable response time, coming both from the low cpu load of the cloud and low traffic on the 4G/LTE channel. Times are 3.5X compared with the (A) scenario.

As final evaluation both theoretical aspects and experimental results confirm that Wi-Fi must be preferred to radio communications like 3G or 4G/LTE, as it fulfills completely the real-time requirement. As second aspect the fog-to-cloud approach of the (A) scenario presents the best performance, thus it would clearly highlight the benefits of the adoption of the fog support. The ability to move close to the edge the computation makes the difference: even if with a smaller computing element like a RaspberryPi3 the overall performance is fully respondent to the requirement and leaves room for additional load that could come from the adoption of augmented reality features.

It is remarkable to notice the benefits coming from the distributed processing supported by the COMPSs and DER runtime support embedded into the mF2C agent, that enables the optimization of computing, moving and balancing the load at different levels of the fog hierarchy. Another benefit coming from the mF2C support is related to the distributed data management: DataClay [6] mechanisms such as replication guarantee that nodes requesting some data will have a way to access it, even if the originating node is not available at a given time. At the same time these replicas can follow different synchronization policies depending on the particular data, while avoiding single points of failure. All the mentioned benefits of orchestration, distribution and optimization in the resource usage would not be possible with the direct link of the edge devices to a centralized cloud-based architecture.

5 Conclusions

The relevant increase in the number of IoT devices is going to generate a huge amount of data. While consumer market will attract most attention, B2B applications will generate more revenues. But the successful development of business depends on critical factors like security, reliability and fast response of proposed solutions. So the arise of the fog computing concept as an architectural model that makes the glue between the cloud and the IoT, extends cloud computing and services to IoT objects to the ends of the network.

The experimental use case on Smart Fog Hub Service (SFHS) has been described, detailing the main business needs and the objectives of using proximity marketing and a personalized customer centric approach based on the use of Collaborative Filtering and a recommendation system. Both the system and application architecture defined for iteration-1 have been presented, with a detailed vision of the mF2C framework, its hierarchical architecture, resource clustering in layers and agent splitting into PM and AC blocks.

Then some critical requirements of the use case have been analyzed in depth, particularly the real-time calculation of object positions and check for Points of Interests nearby. For this reason two different approaches have been defined, one with the fog-to-cloud support, the other with direct cloud connection. At the same time different wireless communications means have been evaluated with their characteristics.

Finally, a benchmark has been defined for the experimental evaluation of different scenarios, where results have confirmed that the mF2C approach together with Wi-Fi network connection brings the best performance, compared with other approaches based on direct cloud connection and 3G/4G usage.

Acknowledgments This work is supported by the H2020 mF2C project (730929).

References

1. Boston Consulting. Internet of things market to reach $267b by 2020.
2. IATA. 2036 forecast reveals air passengers will nearly double to 7.8 billion, 2017.
3. McKinsey Global Institute. The internet of things: mapping the value.
4. McKinsey Global Institute. What's new with the internet of things?
5. Francesc Lordan, Daniele Lezzi, Jorge Ejarque, and Rosa M. Badia. An architecture for programming distributed applications on fog to cloud systems. In *Euro-Par 2017: Parallel Processing Workshops*, pages 325–337. Springer International Publishing, 2018.
6. Jonathan Mart, Anna Queralt, Daniel Gasull, Alex Barcel, Juan Jos Costa, and Toni Cortes. Dataclay: A distributed data store for effective inter-player data sharing. *Journal of Systems and Software*, 131:129 – 145, 2017.
7. Xavi Masip-Bruin, Eva Marín-Tordera, Jordi García, Ana Juan Ferrer, Anna Queralt, Daniele Lezzi, Admela Jukan, Alexander Leckey, Antonio Salis, Matija Cankar, Denis Guilhot, Cristovao Cordeiro, and Jens Jensen. mf2c: Towards a coordinated management of the iot-fog-cloud continuum. In *Proceedings of the ACM Smartobjects'18 (MobiHoc) Conference*, page To appear, 2018.
8. Wilson Ramirez, Xavier Masip-Bruin, Eva Marin-Tordera, Vitor Souza, Admela Jukan, Guang-jie Ren, and Oscar Gonzalez de Dios. Evaluating the benefits of combined and continuous fog-to-cloud architectures. *Computer Communications*, 113:43 – 52, 2017.
9. Antonio Salis and Glauco Mancini. Making use of a smart fog hub to develop new services in airports. In Dora B. Heras, Luc Bougé, Gabriele Mencagli, Emmanuel Jeannot, Rizos Sakellariou, Rosa M. Badia, Jorge G. Barbosa, Laura Ricci, Stephen L. Scott, Stefan Lankes, and Josef Weidendorfer, editors, *Euro-Par 2017: Parallel Processing Workshops*, pages 338–347. Springer International Publishing, 2018.