EDGE COMPUTING ARCHITECTURE FOR MOBILE CROWDSENSING

Seminar Report

Submitted in partial fulfillment of the requirements for the award of degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

of

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

Submitted By

PREM JYOTHI

Department of Computer Science & Engineering
Mar Athanasius College Of Engineering Kothamangalam

EDGE COMPUTING ARCHITECTURE FOR MOBILE CROWDSENSING

Seminar Report

Submitted in partial fulfillment of the requirements for the award of degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

of

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

Submitted By

PREM JYOTHI

Department of Computer Science & Engineering
Mar Athanasius College Of Engineering Kothamangalam

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
MAR ATHANASIUS COLLEGE OF ENGINEERING
KOTHAMANGALAM

CERTIFICATE

This is to certify that the report entitled **Edge Computing Architecture for Mobile Crowdsensing** *submitted by* **Ms. PREM JYOTHI, Reg. No. MAC15CS046** *towards partial fulfillment of the requirement for the award of Degree of Bachelor of Technology in Computer science and Engineering from APJ Abdul Kalam Technological University for December 2018 is a bonafide record of the seminar carried out by her under our supervision and guidance.*

....	....	....
Prof. Joby George	**Prof. Neethu Subash**	**Dr. Surekha Mariam Varghese**
Faculty Guide	***Faculty Guide***	***Head Of Department***

Date: Dept. Seal

ACKNOWLEDGEMENT

First and foremost, I sincerely thank the 'God Almighty' for his grace for the successful and timely completion of the seminar.

I express my sincere gratitude and thanks to Dr. Solly George, Principal and Dr. Surekha Mariam Varghese, Head Of the Department for providing the necessary facilities and their encouragement and support.

I owe special thanks to the staff-in-charge Prof. Joby George, Prof. Neethu Subash and Prof. Joby Anu Mathew for their corrections, suggestions and sincere efforts to co-ordinate the seminar under a tight schedule.

I express my sincere thanks to staff members in the Department of Computer Science and Engineering who have taken sincere efforts in helping me to conduct this seminar.

Finally, I would like to acknowledge the heartfelt efforts, comments, criticisms, co-operation and tremendous support given to me by my dear friends during the preparation of the seminar and also during the presentation without whose support this work would have been all the more difficult to accomplish.

ABSTRACT

Mobile crowdsensing (MCS) is a human-driven service enabling citizens to observe the phenomena of individual, community, or even societal value. They share sensor data about their environment while on the move. Typical MCS service implementations utilize cloud-based centralized architectures. They consume a lot of computational resources and generate significant network traffic. Mobile edge computing (MEC) is a natural choice to distribute MCS solutions by moving computation to network edge. An MEC-based architecture enables significant performance improvements due to the partitioning of problem space based on location. Real-time data processing and aggregation is performed close to data sources. The key MCS features are placed within the reference MEC architecture. The proposed architecture also decreases privacy threats and permits citizens to control the flow of contributed sensor data. The analysis of service overhead introduced by distributed architecture shows that this overhead is controllable and small compared with the aforementioned benefits. The proposed architecture also creates an environment for the establishment of an MCS marketplace for trading of both raw sensor data and aggregated or processed information.

Contents

List of Figures

List Of Abbreviations

MCS	Mobile Crowdsensing
MEC	Mobile Edge Computing
DNS	Domain Name System
ME	Mobile Edge
UE	User Equipment
RA	Reference Architecture
CFS	Customer Facing Service
OSS	Operations Support System
MEO	Mobile Edge Orchestrator
MGRS	Military Grid Reference System

Introduction

Mobile Crowdsensing (MCS) is a human-driven activity which leverages the pervasiveness of wireless connectivity and various mobile devices with built-in sensing capabilities as well as the inherent user mobility to create dense and dynamic data sets characterizing our environments. In other words, MCS benefits from a large user base and its potential to become a rich source of information about various phenomena in our environment.

A typical MCS scenario involves users carrying their smartphones with applications running in the background to continuously collect sensor readings, either from built-in sensors or wearables. Such data acquisition activity requires minimal user involvement and is named opportunistic sensing in literature, in contrast to participatory sensing which requires active user involvement to create sensor readings. The generated sensor data is preprocessed on mobile devices and subsequently sent to cloud servers running an MCS service. The service can potentially receive huge data volumes from a large user base and generate significant load on mobile network.

The value of MCS data is twofold: 1) offline usage: huge data sets serve as input for big data analytics to uncover hidden information about sensed phenomena in areas which are otherwise difficult to cover with dense spatio-temporal measurements, and 2) real-time usage: continuous processing of incoming data streams creates context-aware notifications and alerts, both for citizens and public administration, to be sent from the cloud to user mobile devices.

MCS service implementations are cloud-based and centralized. This architecture consumes a lot of resources since many concurrent connections to back-end cloud servers are needed for near real-time processing and storage of incoming data sets. Moreover, cloud services perform device management functions: They coordinate the sensing tasks on many user devices and keep track of device context to choose the best data sources for defined sensing tasks. Both device management and real-time data processing require significant computational resources in case of large-scale MCS deployments. In addition, user movements and frequent context changes can quickly make information obsolete, which requires efficient real-time processing of raw data to produce contextbased information followed by notification delivery with low propagation latency in real-time usage scenarios.

Mobile Edge Computing (MEC) is designed to enable third parties to run their services and applications at the edge of mobile networks, and is promoted as an enabler of advanced IoT services of massive scale, such as MCS, that would not be technically or economically feasible before the launch of 5G-like networks. MEC moves computation in the proximity of mobile devices by introducing a new intermediate layer responsible for data filtering, aggregation, processing and storage: In the context of MCS it is used to process MCS data between mobile devices and cloud services. Therefore, in addition to preprocessing of raw sensor data on mobile devices, edge resources can be used for processing/aggregation of data streams contributed by a subset of users involved in MCS tasks and can even support real-time usage scenarios autonomously without the need to contact cloud services.

Related works

The authors bring to the forefront the necessity of defining a unifying architecture in order to address current limitations concerning developing and deploying MCS applications. To their view, architecture should enable application developers to specify their data needs in a high level language, identify common data needs across different MCS applications in order to avoid unneeded sensing and processing activities, properly schedule tasks to the most suitable set of mobile devices, ensure the desired data quality in a dynamic setting through proper reconfigurations and reuse the same local processing activities across different device platforms.

Currently, there is little consensus on the underlying system architectures with different MCS applications adopting different models, aiming to provide solutions to different problems at hand. In the following, we critically survey recent MCS systems and architectures of state-of-the-art related research efforts, in an attempt to identify open research issues and challenges and indicate potential ways of addressing them.

2.1 Sensarena

Sensarena is a three-tiered architecture presented in [1]. The first tier is the presentation tier and comprises the participants that provide collected sensor data and the requestors that submit sensing requests, equipped with SensarenaP and SensarenaR android applications, respectively. The second-tier is the business-logic tier and the third is the data tier that stores separately all user-related information, sensing task details and collected data. Sensarena platform focuses on energy-aware task assignment by excluding participants that their energy resources are below a pre-defined threshold, not allowing participation in sensing tasks and by allocating tasks to the closer participant(s) to the point of interest / measurement. The users should be registered and authenticated in order to use Sensarena platform, while they are enabled to denote their preferences on which sensor data shall be contributed.

2.2 Context-aware real-time open mobile miner

The authors propose and develop a context-aware real-time open mobile miner (CAROMM) framework to support efficient and scalable data collection for mobile crowd sensing. CAROMM integrates and correlates sensory data with social-related data from Twitter and Facebook and delivers real-time information to mobile users, answering queries pertaining to specific locations of interest. CAROMM leverages on resource-aware and energy-efficient local analytics and processing of data on the mobile device along with the relevant contextual information associated with them, reducing, thus, the amount of data being sent to the cloud and the amount of energy consumed on the mobile devices, supporting however quite accurate data collection in comparison to models of intermittent / continuous sensing and sending of information. Specifically, mobile data stream mining is employed and resource-aware clustering on sensed data is used to identify significant changes in the current context of operation in order to reduce the frequency and the amount of data transferred to the cloud, ensuring at the same time that significant information will not be lost. The sensitivity of change detection can be controlled by CAROMM framework. Finally, the cost of sending raw sensor data at pre-specified intervals to the cloud for processing vs. data aggregation and processing on the mobile devices and subsequently sending to the cloud is evaluated in terms of data transfer, energy consumption and accuracy.

2.3 Platform for remote sensing using smartphones

Platform for Remote Sensing using Smartphones (PRISM) is presented in [2]. PRISM endeavors to efficiently address the challenge of easy development and deployment of MCS applications, combining and balancing characteristics of generality, security and scalability. PRISM enables re-usage of existing code modules, adopts a push-based model sending application tasks out automatically to an appropriate set of mobile devices based on a pre-specified set of criteria, while for preserving security, untrusted applications run in a software sandbox enhanced with novel features such as sensor access control for controlling access to sensitive sensor data, forced amnesia so as to wipe out a PRISM application's state periodically and resource (in terms of energy and bandwidth) metering in avoid resource depletion.

2.4 Vita

Vita is a mobile cyber-physical system, which supports efficient development, deployment and management of multiple crowd sensing applications and tasks [3]. It is a flexible architecture, integrating service-oriented design principles with a resource optimization mechanism in order to allow for intelligent task allocation as well as for dynamic collaboration of services between mobile devices and cloud computing platform during run-time. Specifically, applicationoriented service collaboration model is introduced for intelligently allocating human-based tasks among users and computing tasks between mobile devices and the cloud computing platform efficiently and effectively. The system takes into account different parameters and criteria, such as computation power, communication capacity, remaining battery time, the number of similar tasks users have competed in the past, the number of remaining tasks, user preferences, requirements and constraints, and so on. Social related information may also be utilized in order to quantify the distance / relationship of two entities (either physical or virtual) in order to facilitate the development of human and computing tasks according to different application scenarios. The authors leverage on two intelligent computing techniques in order to provide optimized solutions for the task allocation process: genetic algorithm and K-means clustering. A service state synchronization mechanism is incorporated so as to handle potential service failures, ensure service consistency and collected data correctness. In a nutshell, Vita leverages on service computing, cloud computing and social computing, proposing a comprehensive and flexible architecture, supporting both application developers and users in the context of mobile crowd sensing.

2.5 McSense

In [4], the authors present McSense, a mobile crowdsensing platform, which enables task design and assignment, exploiting information about potential workers, regions and their context to efficiently and effectively perform the task assignment process. In particular, McSense, for each geo-localized sensing task, estimates the time required and the number of workers that are necessitated to complete it with a specific probability. To this end, it leverages information on the profile of users / region along with task related data. They have defined three different task assignment policies based on the time spent by workers in the task area in the past, and the time period since they have been in the area in the past. They have incorporated in their framework an incentive mechanism based on monetary rewards and additionally they exclude from the set of potential workers for a specific task the workers that have limited battery resources in their mobile device. In a nutshell, McSense puts task description, task assignment, and mobile sensing in a closed loop that allows more efficient and effective usage of all socio-technical resources involved.

Typical MCS service implementations are cloud-based and centralized. This architecture consumes a lot of resources since many concurrent connections to back-end cloud servers are needed for near real-time processing and storage of incoming data sets. Moreover, cloud services perform device management functions: They coordinate the sensing tasks on many user devices and keep track of device context to choose the best data sources for defined sensing tasks. Both device management and real-time data processing require significant computational resources in case of large-scale MCS deployments. In addition, user movements and frequent context changes can quickly make information obsolete, which requires efficient real-time processing of raw data to produce context based information followed by notification delivery with low propagation latency in real-time usage scenarios.

Large-scale and centralized MCS introduces the following problems:

- generates significant load on mobile network radio and backhaul,
- creates an increased traffic to cloud servers running MCS services,

- it is computationally expensive, especially for real-time usage scenarios, due to a large number of devices participating in MCS tasks with frequently changing context,
- increases the latency of data and information propagation, which is critical for real-time usage scenarios, and
- represents a threat to user privacy since all user traces are collected in a centralized manner.

The proposed method

Mobile Edge Computing (MEC) is designed to enable third parties to run their services and applications at the edge of mobile networks, and is promoted as an enabler of advanced IoT services of massive scale, such as MCS, that would not be technically or economically feasible before the launch of 5G-like networks. MEC moves computation in the proximity of mobile devices by introducing a new intermediate layer responsible for data ltering, aggregation, processing and storage[5]: In the context of MCS it is used to process MCS data between mobile devices and cloud services, as shown in Fig 3.1. Therefore, in addition to preprocessing of raw sensor data on mobile devices, edge resources can be used for processing/aggregation of data streams contributed by a subset of users involved in MCS tasks and can even support real-time usage scenarios autonomously without the need to contact cloud services.

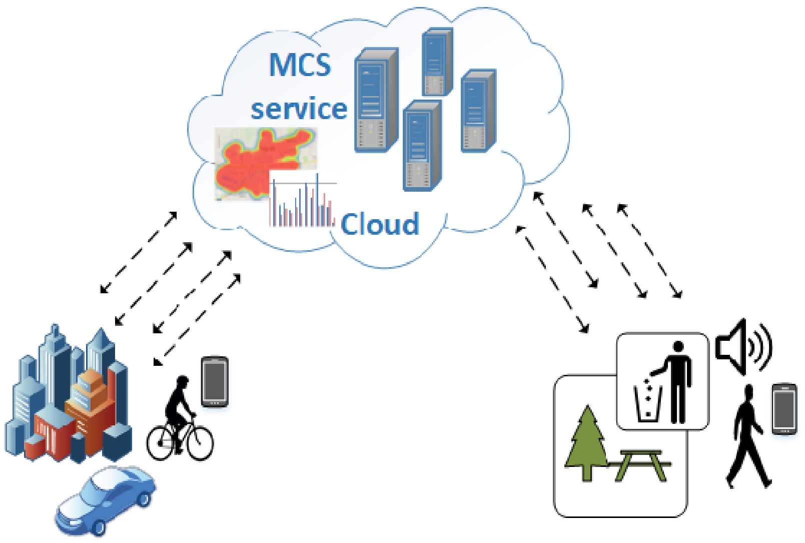

Figure 3.1: Mobile crowdsensing deployments- MCS architecture powered by MEC infrastructure

The main benefits of MEC in the context of MCS are the following: parallelization and partitioning of problem space based on location, where MEC servers naturally take the responsibility for controlling the sensing process on mobile devices located within their deployment area and manage MCS tasks within the same area,

- computation offloading from both mobile devices and cloud servers, since certain data processing tasks can be performed at the edge,
- reduced computational complexity of centralized cloud services, which in turn increases the complexity of service orchestration,
- decreased latency when disseminating notifications and alerts to mobile devices in case of real-time MCS usage scenarios, and
- reduced privacy threats since sensitive data is partitioned and distributed across MEC servers.

3.1 Mobile crowdsensing characteristic features

To model a functional architecture suitable for MCS, while exploiting the full potential of MEC technology, we identify a characteristic set of MCS features focusing on these that benet from functionalities natively provided by the MEC technology.

3.1.1 Data preprocessing

Data (pre)processing refers to the processing performed close to sensor data production place, mostly to reduce the load on edge and application servers as well as to save smartphone resources. An MCS service performs basic preprocessing of raw data on a smartphone before transmitting data further to the cloud, e.g., time-series processing techniques can be used to reduce the size of the data that is communicated. An ME server can aggregate received data streams before sending a final result to the cloud for a longterm storage. Such incremental and hierarchical processing is adequate to reduce network traffic and save energy on user devices, thus increasing MCS service performance.

3.1.2 Sampling mode

Sampling mode distinguishes continuous sampling which enables monitoring of a value during its lifetime, and triggered sampling which indicates the occurrence of a signi ficant event. Since the ME technology can infer user availability based on network traffic, this information can be used to control the data acquisition process on UE (i.e., continuous sampling can be triggered in the background when appropriate, while the user is occupied with another activity). Of course, it is vital to have user consent for such scenarios.

3.1.3 Sensing scope

Sensing scope describes outreach of an MCS service, grouping it in three categories: personal, group and community. The personal scope is focused on individuals, the group scope involves more individuals who share a common goal, while the community scope indicates that a topic is of interest to a wide-scale population. Both group and community dissemination scope, which is location-aware, are benefiting from the MEC technology, since an ME server possesses knowledge about users grouped within the same geographical location.

3.1.4 Device discovery

Device discovery is one of the mandatory non-functional requirements because an MCS service needs to efficiently match a task with the best workers. A network provider running an ME host has the knowledge about available UEs which are candidates to fulfill an active MCS task and can choose the most appropriate workers based on their location, capabilities and reputation.

3.1.5 Mobility support

Mobility support relates to handling of a large user base and frequent location changes. The functionality is vital to detect patterns of user movement, with emphasis on identifying dense and sparsely populated areas, so that an MCS service can devise measures to efficiently handle both cases.

3.1.6 Energy management

Energy management is responsible to minimize energy consumption by optimizing the set of UEs that are needed for MCS tasks, while others are on stand-by until their involvement is required. Smart energy management ensures that a user is more willing to be involved in MCS tasks, while the final goal is to increase task accomplishments and to satisfy the required sensing quality levels. Since an ME host is responsible only for managing a subset of devices, it can perform optimizations on a smaller scale with better results compared to the cloud approach which optimizes energy consumption on a global scale.

3.1.7 User and sensor node reputation

User (and sensor node) reputation provides a measurement of trustworthiness related to a contributed data set which is vital for the quality of contributed data, since the service is driven by data collected from sources that are not known a priori. There is no single method to calculate reputation because each service adapts the reputation mechanism to its particular needs. In MEC environments, novel distributed mechanisms for calculating trustworthiness, which should not jeopardize user privacy, are needed.

3.1.8 Real-time processing

Real-time processing is essential for MCS services, both in terms of real-time data processing and information dissemination to interested parties. Support for such high reactivity is expected because of a highly dynamic environment, which involves numerous users frequently changing their context, while the service needs to send only up-to-date information. The MEC infrastructure can fulfill ultra-low latency requirements (less than 1 ms) since it enables hosting of MCS services at the network edge and therefore supports data processing close to the users. It provides the shortest path between the application and MCS service to significantly reduce the information retrieval time.

3.1.9 Communication protocol

Communication protocol can be used to distinguish two groups of devices based on their communication capabilities. Resource constrained devices without IP-connectivity use one of (wireless) close proximity protocols, such as Bluetooth or ZigBee, to communicate with a smartphone which serves as a gateway. Devices with IP-connectivity use one of protocols suited for IoT and MCS environments, such as CoAP or MQTT. The MEC can act as a gateway between devices which interact through diverse communication technologies serving as an aggregation point towards the cloud layer.

3.1.10 Security and privacy

Security and privacy is a key requirement since users contribute their location data or data enriched with sensitive contextual information, and thus users would like to ensure that sensitive data is not subjected to misuse. Approaches to address this requirement span from anonymization to endto- end data protection along the communication path. MEC technology can potentially improve security and privacy protection of end users, since there is no need to forward usersensitive data in its original form to the cloud.

In addition to the previously listed features, MCS services also possess domain-specific characteristics which relate to the type of sensing nodes, user involvement (opportunistic or participatory), incentives motivating users for active involvement, and quality of service measuring overall user satisfaction with an MCS service. Since the previously mentioned features do not have significant influence on MCS architecture, they are not further discussed in detail.

3.2 Functional mobile crowdsensing architecture

The four-layered functional architecture for MCS derived from the key MEC properties and MCS features in given in Figure 3.2.

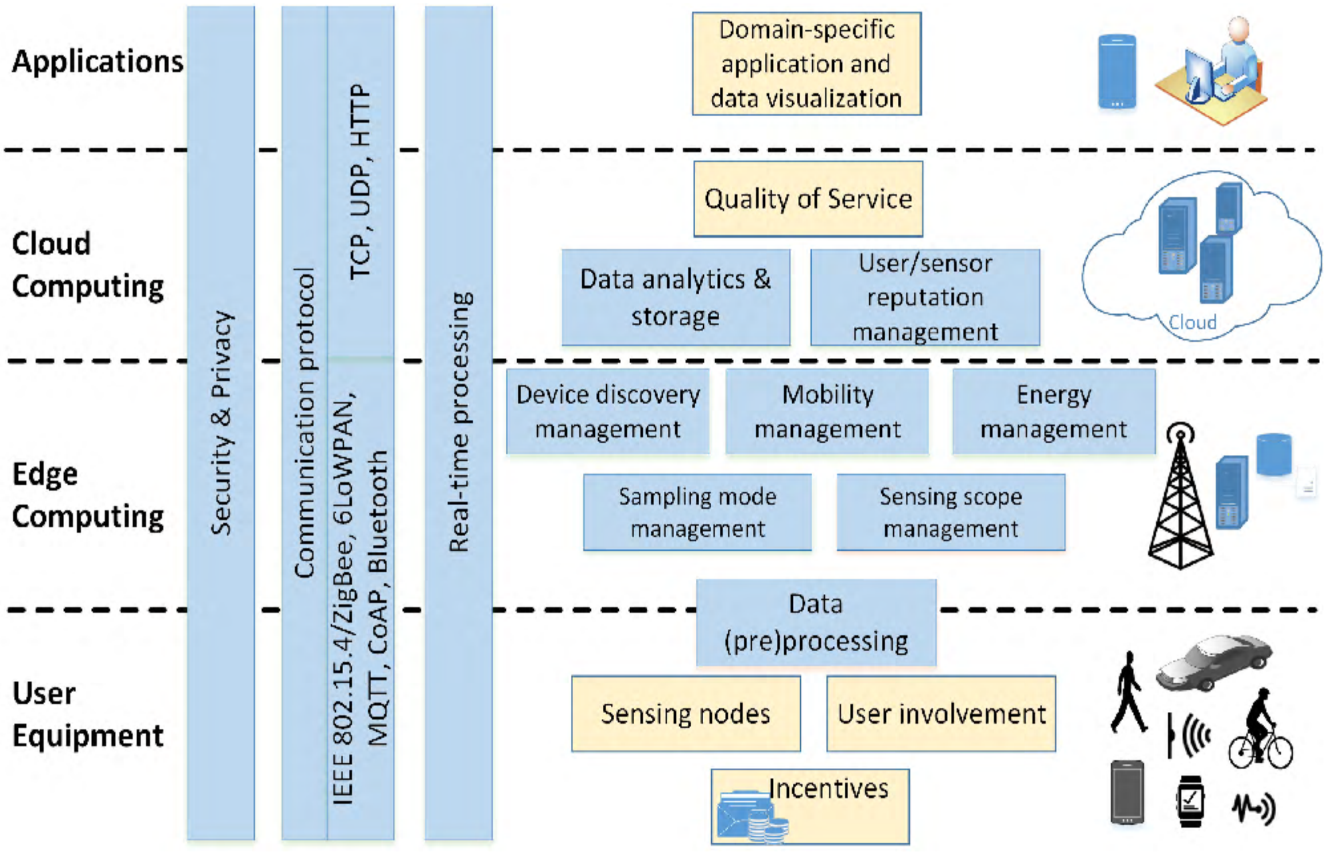

Figure 3.2: The four-layered functional architecture for MCS.

3.2.1 User equipment

User Equipment comprises both wearable sensors and smartphones with different sensing capabilities, as well as human-driven sensing, where incentive mechanisms have great impact on participant's involvement in MCS tasks. Initial data processing is typically performed on a smartphone to reduce bandwidth and energy consumption before sensor data is transmitted to an edge server.

3.2.2 Edge computing layer

Edge Computing layer is located in close vicinity of users, between physical sensing devices and the cloud. It is responsible for the management of workers in its location area, data gathering from active workers, processing and filtering of sensor data, as well as data aggregation. Available information about UEs on ME servers can be reused by MCS service to infer user context, detect new available workers, monitor user mobility, and to push notications created at the edge to the UE with low latency.

3.2.3 Cloud computing layer

Cloud Computing layer offers resources for complex data analysis and long-term storage. Furthermore, this layer enables interactions between multiple MCS services, including their collaboration and data exchange.

3.2.4 Applications

Applications are developed on top of the cloud services and provided to requesters specifying MCS tasks who want to analyze results of data analysis performed on top of collected data sets. They are domain-specific and may include different interfaces which enable data visualization and knowledge sharing among users, both via web and mobile applications with real-time user notifications.

Security and privacy, support for diverse communication protocols and real-time processing are generic features needed across all architectural layers to ensure data protection during the execution of an MCS service, and to enable interactions and data dissemination between multiple functional components.

3.3 Mobile edge computing paradigm

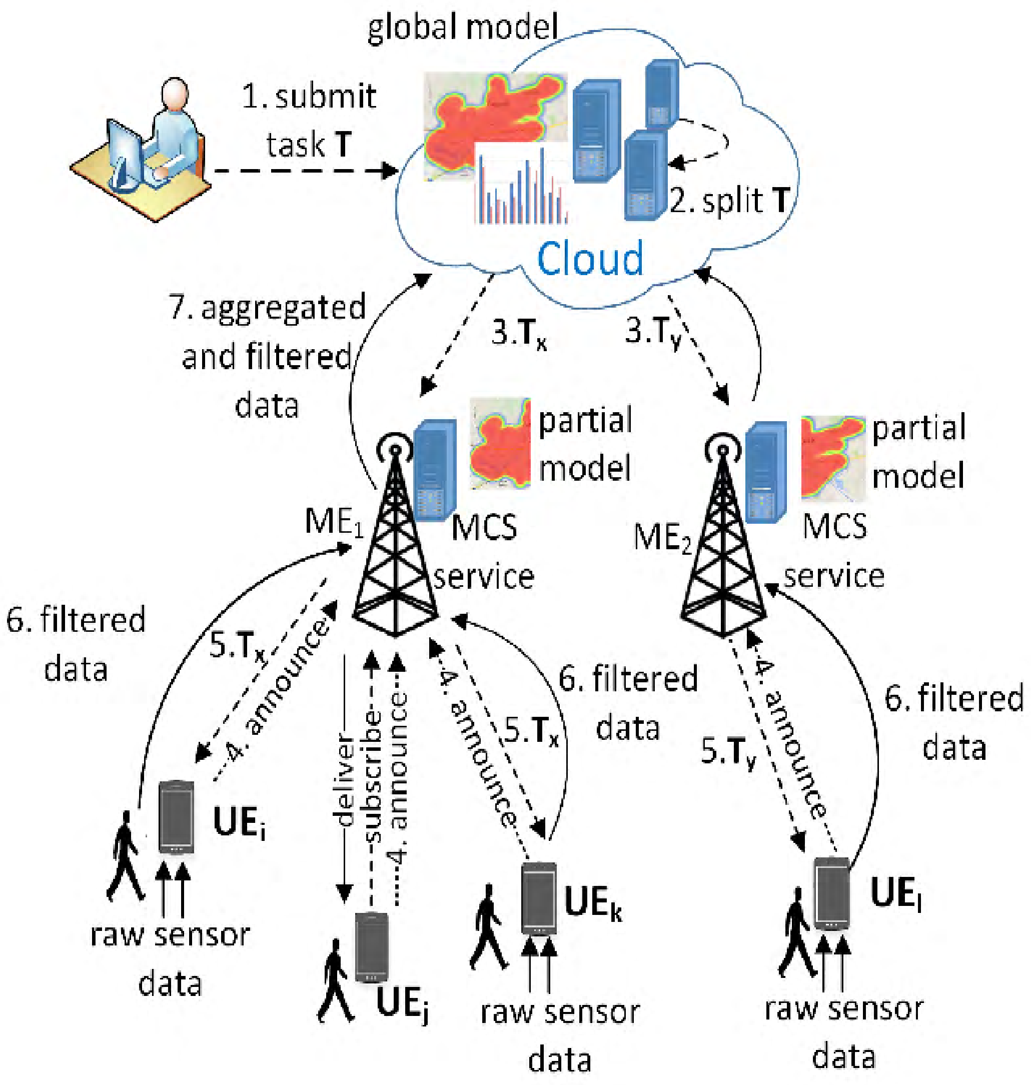

Figure 3.3: Communication flow in an MCS ecosystem deployed on the MEC infrastructure.

The MEC paradigm has the power to reduce the load of MCS data processing and service execution by distributing MCS tasks to multiple Mobile Edge (ME) servers deployed at network edge. ME servers become responsible for a subset of tasks and workers in their close vicinity, as shown in Figure 3.3. When a new task T (e.g., for monitoring air quality in a specific region) is submitted to the system (1), the cloud server splits T in two subtasks Tx and Ty (2), based on required location areas, and forwards them to the corresponding ME servers (3). An MCS service running on the ME server is aware of all potential workers in its area and their capabilities, since workers announce (4) the type of data they can and are willing to measure with their user equipment (UE) when entering the area under the responsibility of an MCS service. Thus, only tasks which can be potentially executed by workers are forwarded to their UEs, which prevents device overload, in accordance with the publish/ subscribe messaging paradigm that proves to be suitable for MCS ecosystems, as we have already shown in [6]. For example, in Figure 3 the MCS service deployed onME1 sends task Tx only to workers i and k with equipment UEi and UEk (5). Workers start to produce raw sensor data, which is rst ltered on their mobile devices and subsequently delivered to the MCS service running on ME1 (6). Since ME servers have substantial computational power compared to mobile devices, they can perform data aggregation and processing of all data produced by workers under their responsibility. Thus, an MCS service builds a partial data model for its area and sends the aggregated model to the cloud (7). The MCS cloud server subsequently creates a global data model based on partial models received from multiple ME servers and performs different analytics to extract knowledge about the sensed phenomena for a larger area. Additionally, it can also correlate different sensor data. In case of real-time MCS usage, users who express interest in the data produced in their close vicinity can receive alerts with low propagation delay from their edgeMCSservice. For example, user j sends a subscribe message to the MCS service on ME1 server to express his/her interest in the air quality for the area in which he/she currently resides. If the MCS service posses data regarding air quality, it delivers real-time notifications to the user j, as long as the user is interested in the data. Note that an event subscribe can happen at any time, while deliver occurs when an MCS service deployed on the ME server finds a positive match between user subscription and published data.

3.4 Mobile edge computing reference architecture

We position MCS functional components within the MEC reference architecture (RA). More specifically, we focus only on the functionalities that we have identified as part of the Edge Computing layer, and place them in the MEC RA. Other functionalities are placed either above MEC infrastructure in the cloud layer responsible for complex data analysis and long-term storage, or below MEC infrastructure in UE layer which contains different data sources.

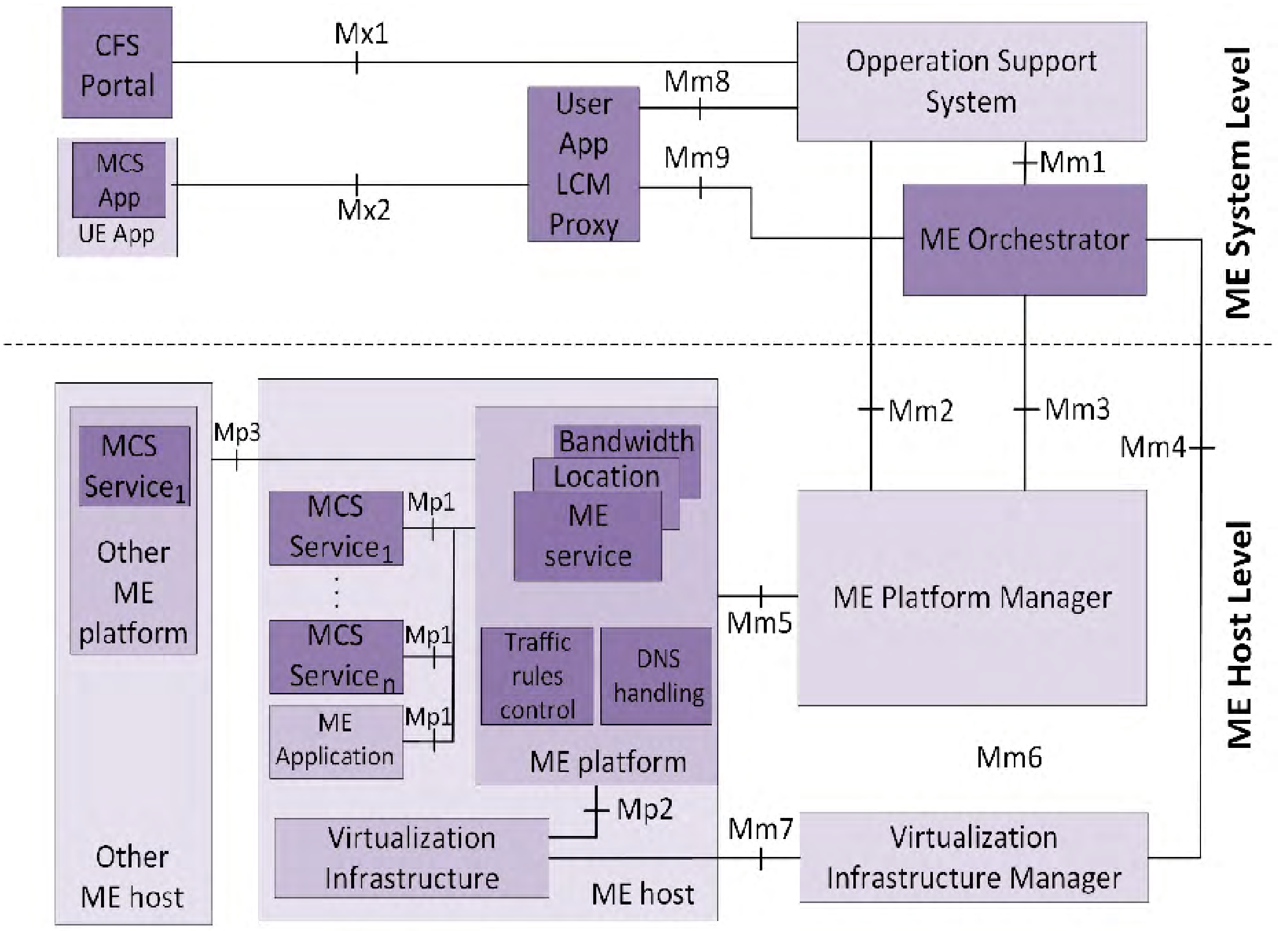

Figure 3.4: Mapping between MCS and MEC reference architecture

MEC RA identifies the functional entities at the host and system level, and defines reference points enabling communication between those entities. Different MCS services can be deployed on an ME host as ME applications, in accordance with MEC RA, as shown in Figure 3.4. The ME host runs the ME platform and the Virtualization Infrastructure to provide computing, storage, and network resources for the ME applications. The ME platform maintains valuable network information for MCS. In particular, the MCS service interacts with the ME platform over the Mp1 reference point to consume network operator data useful for managing the distribution of MCS tasks and worker involvement. Also, the ME platform can be used to direct user traffic to a specific ME application deployed on a corresponding ME host since it supports configuration of the local domain name system (DNS) proxy/server. The Mp3 reference point enables direct communication between two parallel deployments of MCS services in different ME hosts, which can be utilized for direct communication between neighboring ME hosts.

The interaction flow in MEC RA is as follows: a new MCS task is submitted by a requester to the system via the customer-facing service (CFS) portal. It is delivered to the Operations Support System (OSS), the highest level management system responsible for the instantiation or termination of ME applications. The OSS forwards this task to the ME Orchestrator (MEO), a central function in the ME system which has visibility over the resources and capabilities of the entire ME network. Since MEO keeps track of all deployed ME hosts, their available services and resources, as well as already instantiated applications and network topology, it can split the received task in several subtasks, based on the location area, and assign each subtask to a corresponding ME host. Also, it can instantiate a new MCS service if needed. In parallel, an MCS application which runs on a UE can use Mx2 reference point to communicate directly with an MCS service deployed on the ME host responsible for the UE in a particular area. The user application lifecycle management proxy (User AppLCMProxy) will authorize the received request and forward it to the OSS and ME Orchestrator for further processing. Other components of the MEC architecture, as shown in Figure, such as the ME Platform Manager and Virtualization normal work operation of the ME host, i.e., they handle the management of specific functionalities on a particular ME host and all applications running on it. In the context of MCS, they are responsible for orchestrating the MCS services on ME hosts in accordance with defined tasks and available workers.

3.5 Challenges

3.5.1 Interoperability

Interoperability between MCS deployments covering different domains and phenomena is vital to reuse the same user base for various MCS tasks. In traditional MCS ecosystems, collaboration is possible only between back-end cloud systems, without sharing the same user base, while MEC technology enables collaboration on a ME host level, where MCS services can share the same user base. However, to enable cooperation and data sharing between multiple MCS services deployed on ME hosts coordinated from the cloud, all MCS services should use unified interfaces, open communication protocols and standardized information models. More specifically, interoperability between MCS deployments relates to two aspects of service coexistence in MCS environments enabled by the MEC technology. The first aspect enables the sharing of user-generated data between MCS services running in MEC environment. Semantic interoperability which enables various MCS solutions to understand the data being shared is vital here, with a potential to use the concepts of data bartering between different solutions that in this context have new incentives to share their user base. The second aspect relates to data sharing between a network/ME host operator and an MCS service. Since the network operator maintains valuable data characterizing potential workers, an MCS service would greatly benefit from such information, but needs to properly identify users and understand the data provided by the mobile network.

3.5.2 Enriched user context

Enriched user context would represent a user with a single profile collecting all user-related information from both the network operator and MCS service domain. Such user profiles would provide a deeper understanding of user behavior for the benefit of all involved parties. The user is involved in MCS tasks with minimum overhead, a network operator acquires better understanding of user needs and can provide personalized customer services, while an MCS service can optimize the management of MCS tasks and consumption of required resources.

3.5.3 Privacy and security

Privacy and security is a delicate issue both for MCS and MEC because both environments are inextricably connected to user context. The MEC technology offers positive benefits to the security of MCS data, mostly due to distributed storage and its aggregation at the edge so that a single breach would not expose all user data. In addition, such distributed operation is beneficial from the privacy perspective, but is also challenged by the previous open issue requiring a single user profile. An open and yet vital issue is to ensure that users have the control over their data with protection of sensitive data at all service levels, so that contextual user data are not misused or used by third-parties without user consent.

3.5.4 Orchestration of mobile crowdsensing services

Orchestration of MCS services needs to be enabled by the MEC virtualization infrastructure which will initiate MCS services on adequate edge resources to answer the needs of MCS tasks and take into account available workers. Furthermore, it needs to regulate an MCS service life-cycle, setting the responsibilities regarding service execution which provides location for data storage and defines rules for data ownership and access rights.

3.5.5 Task scheduling and optimization

Task scheduling and optimization between users connected to the same edge server strongly depends on their geographical locations, mobility patterns and reputation levels as well as requirements of active MCS tasks. Different algorithms can be deployed on ME servers to minimize energy consumption of mobile devices, cost and execution time, while satisfying sensing coverage and ensuring QoS. Furthermore, a geographical distribution of MCS tasks and the types of MCS services deployed on ME servers will also have influence on the performance of the proposed system. It is possible to achieve significant energy savings and reduce latency because tasks are processed at an ME server and there is no need to send all data to a back-end cloud.

3.6 Performance evaluation and comparison

Hereafter we present the evaluation of the proposed MEC architecture for MCS to investigate the overhead incurred due to the need to orchestrate MCS services at network edge. This overhead depends highly on the movement pattern of users involved in MCS tasks, and thus a realistic data set is needed to investigate the distribution of MCS services and the need for their reconguration in a MEC environment. For this purpose we are using the data set collected in SouthKorea from March 2011 to September 2012, where 85 participants actively contributed their location information by using a smartphone application for tagging places. The application was used 79 days on average by a single user. Altogether, users have tagged 13,500 distinct places with information obtained from cellular network providers, smartphone GPS sensor, wireless module, microphone and camera. The data set was originally used for autonomous place detection, and thus we needed to process it to match our need of simulating movements of a large user base which is adequate for MCS deployments.

The data set was first filtered to remove entries where user check-ins are not associated to exact location or timestamp. The filtered data set contains 151,649 user check-ins from 67 unique users. Next, we split the filtered set based on a user identifier and date because the number of unique users is too small to investigate a realistic MCS scenario. Thus, we have created multiple virtual user traces for a single day from the trace of a single real user. Each slice (i.e., user-day slice is a set with user check-ins during a single day) represented a movement pattern of a virtual user during one day. The total of 7,724 virtual users and associated user-day traces is used further on in our evaluation. Since the data set does not contain user location information with high sampling frequency as expected in MCS, we created additional check-ins with a sampling frequency of 1 minute by interpolating expected user location between two consecutive check-ins. This creates a realistic MCS data trace with the total of 4.7 million userlocation entries.

The analysis of the created MCS data set reveals that user behavior in urban areas highly depends on the time slots in which the analysis is performed and geographical grouping of users in cells which represent a single location context. For geographical grouping, i.e., determining unique cells over wide urban areas, we used the military grid reference system (MGRS)[7]. MGRS is a reference system for locating points on Earth. It supports precision in the range from 1 meter to 100 kilometers and is suitable to be used both in urban and rural areas. We use cells of 10 square kilometers in our evaluation, where a single MGRS cell is associated to one MEC instance, i.e., we assume that a single ME host is deployed per each MGRS cell covering several cellular antennas that can be densely distributed in urban environments. The second important parameter influencing the deployment of MCS services in corresponding MEC cells is the time period in which user behavior is observed. Long time periods are not suitable for MCS services since the distribution of users in urban areas is volatile during a single day. In particular, different periods within a single day exhibit different characteristics . We decided to analyze the behavior of MCS service in the MEC environment during the course of a day by varying the observation time slot from 15 minutes to 24 hours. We assume that all time slots are independent and an MCS service deployed on a ME host is active during the entire slot if there is at least one user contributing MCS data during this period within a MEC cell.

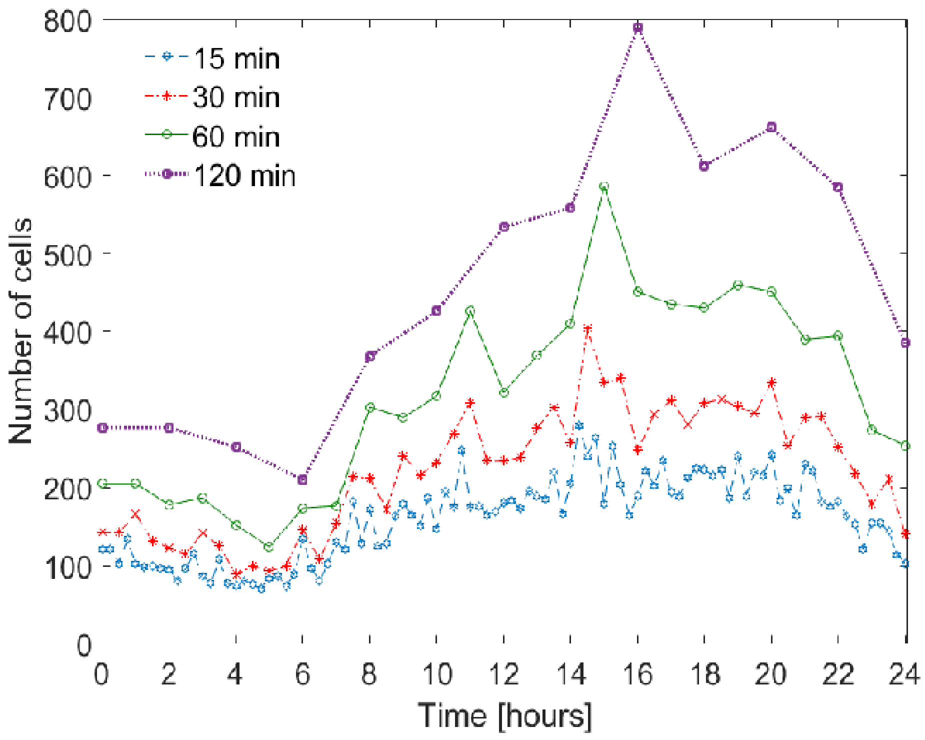

Figure 3.5: Number of active MEC cells during a single day for different time slots.

Figure 3.5 analyzes the number of active MEC cells during a single day depending on the time slot duration. We can see that, as expected, the number of active cells grows for longer time slots since mobility of users is increased (i.e., users travel longer distances) and they can cross over multiple MEC cells in which an MCS service instance remains active during the whole time slot even when there are no users generating data for the particular service. For example, when an MCS service is reconfigured in intervals of 120 minutes, the number of active MEC cells during peek hours (between 2 p.m. and 4 p.m.) is almost 800, while for the 15 minute intervals, the number of MEC cells which are active in parallel never exceeds 300. This indicates that the activity period of an MCS service should be rather short, but each reconfiguration of MCS services is also costly and thus should be carefully selected.

Conclusion

Mobile crowdsensing is a human-driven paradigm empowered by ordinary citizens who contribute and share sensor data by means of mobile devices and wearables. It is a true example of a sharing economy where generated sensor data represents a shared resource, while scalable and interoperable technical solutions are needed to create the next-generation marketplace for MCS involving a huge number of users and various stakeholders.

This paper presents a reference architecture for hierarchical and large-scale deployments of MCS services which assumes the usage of edge computing resources to decentralize MCS services and improve their performance. Mobile Edge Computing brings computation and storage to the edge of mobile network providing MCS services in close proximity of users. The goal of such architecture is to simplify service execution and increase the quality of service, primarily by reducing the latency and complexity of data processing.

Our analysis of the service overhead introduced by the distributed architecture which requires reconfiguration of edge MCS services shows that this overhead is controllable and small, especially for shorter reconfiguration periods of MCS services at network edge. Thus, it represents a promising approach for enabling the next-generation MCS marketplace of massive scale.

REFERENCES

[1] Rim Ben Messaoud, Zeineb Rejiba, Yacine Gharmi-Doubane, “An Energy-aware End-to-End Crowdsensing Platform: Sensarena”, in Proc. of the 13th IEEE Annual Consumer Communications Networking Conference, pp. 284-285, Las Vegas, USA, 2016.

[2]] Tathagata Das, Prashanth Mohan, Venkata N. Padmanabhan, Ramachandran Ramjee, Asankhaya Sharma, “PRISM: Platform for Remote Sensing using Smartphones”, in Proc. of the 8th International Conference on Mobile systems, applications, and services, pp. 63-76, San Francisco, CA, USA, 2010.

[3]] Xiping Hu, Terry H.S. Chu, Henry C.B. Chan, and Victor C.M. Leung, “Vita: A Crowdsensing-Oriented Mobile Cyber-Physical System”, IEEE Transactions on Emerging Topics in Computing, vol. 1, issue 1, pp. 148- 165, 2013.

[4] Giuseppe Cardone, Luca Foschini, Paolo Bellavista, Antonio Corradi, Cristian Borcea, Manoop Talasila, and Reza Curtmola “Fostering ParticipAction in Smart Cities: A Geo-Social Crowdsensing Platform”, IEEE Communications Magazine, vol. 51, issue 6, pp. 112-119, 2013.

[5] P. Mach and Z. Becvar, “Mobile edge computing: A survey on architecture and computation ofoading,” IEEE Commun. Surveys Tuts., vol. 19, no. 3, pp. 16281656, 3rd Quart., 2017.

[6] A. Antoni¢, M. Marjanovi¢, K. Pripuºi¢, and I. P. arko, “A mobile crowd sensing ecosystem enabled by CUPUS: Cloud-based publish/subscribe middleware for the Internet of Things,” Future Generat. Comput. Syst., vol. 56, pp. 607622, Mar. 2016.

[7] T. D’Roza and G. Bilchev, “An overview of location-based services,” BT Technol. J., vol. 21, no. 1, pp. 2027, 2003.