Proceedings of the 54th Hawaii International Conference on System Sciences | 2021

MeshSOS: An IoT Based Emergency Response System

Bhavye Jain
IIT Roorkee
bjain@cs.iitr.ac.in

Kaustubh Trivedi
IIT Roorkee
ktrivedi@cs.iitr.ac.in

Swati Agarwal
BITS Pilani Goa
agrswati@ieee.org

Rahul Thakur
IIT Roorkee
rahulthakur@ieee.org

Abstract

Due to the limited mobility and technical knowledge, senior and disabled citizens of the society face difficulties during emergencies. Most hardware/software emergency assistance/response systems available in the market have a complex user interface even for the general public. Requesting help using these systems requires sharing information such as type and location of the event, which wastes precious time to respond to the event. Often, citizens end up handling the event themselves instead of waiting for someone to arrive at the event location. Hence, it is necessary to design a simple but incredibly robust system which bypasses the challenges of traditional emergency response systems. In this paper, we propose an Internet of Things (IoT) based mesh-enabled emergency response system called MeshSOS, which enables senior and disabled citizens to get assistance by simply pressing a button. Use of mesh networking along with WiFi made our system robust to network failures. We have also developed a central monitoring application for healthcare and security agencies to handle emergency events proactively. Initial field experiments and simulations show that our system has the potential to improve the robustness and response time in an energy and cost-efficient manner.

1. Introduction

Building a community with basic knowledge of delivering first-aid is important for immediate emergency response. Unfortunately, the know-how and the experience of delivering first-aid is not common, and the chances of having such a person around when needed are fairly low. A report in *The Guardian*[1] states that the lack of first-aid skills endangers up to $150,000$ lives annually. In such scenarios, timely assistance from trained personnel takes paramount importance. In most countries, the primary public safety and assistance system includes a unique phone number such as 911 (USA) and 100 (India) where citizens can call to request for assistance. Even though calling a fixed number is the simplest solution, it is certainly not the most efficient one. A citizen in distress has to share details related to his/her location and type of emergency (*security*, *medical*, *fire*, etc.) before an appropriate emergency response team can be dispatched. This delay in response time could mean the difference between life and death in case of critical emergencies[2]. The problem becomes even worse in developing countries where the emergency response services (such as *ambulance* and *police*) are inherently inefficient by design. A report of the Comptroller and Auditor General (CAG) of India shows that the ambulance service failed to respond in more than 40% ($113,632$ out of $275,243$ calls) of the cases within a stipulated time. Furthermore, these calls are not only unattended but cancelled and declined as well. As per CAG's reports [1, 2], in the state of Kerala (India), out of $1,867,508$ received calls, $17,627$ calls were cancelled while $28,102$ calls were declined. The delay in the service cannot be solely attributed to traffic delays, but also communication and management delays. A study by the National Health Systems Resource Centre (NHSRC) [3] shows that the percentage of calls for which ambulances were dispatched were as low as 1.21% of the calls received.

The need for medical and security assistance increases manifold for senior and disabled citizens of society. The response time for emergency events is even more critical when it comes to senior and disabled citizens with limited technical knowledge and/or mobility. Modern smartphone-based emergency response systems can be too complicated or intimidating for the elderly and disabled. In such systems, multiple steps are required before relevant assistance can be dispatched. Also, some systems require a lot of

[1]First aid skills deaths. `https://www.theguardian.com/society/2010/apr/12/first-aid-skills-deaths`

[2]Government of India data shows that more than 50% of heart attack cases reach hospitals late. `https://www.hindustantimes.com/story-penFdsewgGwpIwiQnRDoLJ.html`

URI: https://hdl.handle.net/10125/71089
978-0-9981331-4-0

specific configuration before they can actually work on the devices. Available hardware systems on the market are not affordable by masses and also require regular maintenance [4]. These hardware systems and applications are proprietary and hence, cannot be maintained or repaired by a third party or an end-user. Additionally, most of the existing systems rely entirely on the wide-area networking capabilities of a smartphone (or a smart device) for Internet connection via WiFi or cellular network. This makes the system unreliable since an event of a network failure will completely fail the system without any fail-safe. Therefore, it is need of the hour to develop a system that can bypass the above-mentioned challenges and minimize the points of failures while providing prompt and hassle-free emergency services.

In this paper, we introduce MeshSOS, a robust IoT-based mesh-enabled emergency response system. We have designed a simple and intuitive end-user device which allow citizens to request for *Medical* and *Security* related assistance by simply pressing a button on the device. Our system is extremely convenient for seniors and disabled citizens who prefer ”a single button press to get the job done” vision due to limited technical knowledge and mobility. Additionally, We have developed a central monitoring application which allows emergency service authorities to monitor the end-user devices and to keep track of the emergency events (location, route, status, etc.). Finally, to improve the robustness, MeshSOS uses multi-hop mesh communication which eliminates the requirement of direct connectivity to WiFi/4G networks. Field experiments and simulations show that the performance of our proposed system is better in terms of delay, message delivery ratio, robustness, energy, and cost.

2. Related Works and Motivation

Advancement in hardware and wireless communication technologies has accelerated the development of low-cost and reliable emergency response systems for home/office/business security, medical and road emergencies. There exist many software, hardware, and IoT based products in the market which use wireless technologies (such as WiFi, Bluetooth, and 4G) to provide emergency services. Inbuilt SOS feature in mobiles and third-party apps such as SirenGPS, Kitestring, Red Panic Button, etc. allow users to send an alert for assistance in unfavourable situations. The primary feature in all these apps is to provide an easily accessible switch/button to inform a pre-selected list of contacts or first responders about your whereabouts. Some apps, such as Kitestring take one step further to alert your friends and family if you don’t respond to a potential emergency message sent to your phone. There also exists apps and hardware devices specifically designed to tackle medical emergencies. Dedicated battery-powered lightweight hardware systems such as medical bracelets/necklaces/watches are available in the market. By pushing a switch on these wearables, the central monitoring stations can be informed about the location of a person in distress. In [5], the authors have introduced iHelp, an emergency assistance smartphone app for helping the deaf-mute and elderly citizens. This app can report an emergency using 3G/4G/WiFi Internet, which enables medics to be rushed to the scene quickly. All the above-mentioned devices and apps provide useful features but do not tackle the situations where proper WiFi or cellular networks are unavailable.

In order to deal with unreliable network connectivity, emergency response systems must use multiple wireless technologies in an efficient manner. The system must switch among these technologies without asking extra input from the end-users. In this direction, the use of wireless mesh networking (WMN) is another interesting solution which eliminates the need for ubiquitous connectivity from an access point [6]. WMNs have been receiving a great deal of attention as a broadband access alternative for a wide range of applications including transport, emergency, public-safety, carrier-access, and smart homes [7, 8]. Mesh architecture has the potential to provide easy configuration and high robustness by choosing alternative routes in case of failure/unavailability of a wireless link or access point. It also offers an economical solution to extend network coverage by eliminating the need for a large number of access points. In [9], the authors have proposed an emergency response system based on wireless mesh networks for public safety using an ARM kernel control unit system. The authors suggest that wired/wireless systems with a base station are vulnerable to destruction by natural or human agents, resulting in system collapse. Their proposed wireless mesh system performs better under the circumstances where ordinary communication technologies are unable to the Internet. In [10], the authors have explored the feasibility of using distributed wireless mesh networks for medical emergency response. The authors proposed a wireless information system for medical response in disasters situations. They presented the traffic behavioural observations made by the client-server medical emergency application tested during a large-scale homeland security drill. The authors evaluated the system on some essential requirements for an emergency response system such as geographical position, dependable and redundant backhaul links, utilization of off-the-shelf technology-based products, and robust operation in high interference situations. The results of this study have motivated us to create a system that can be deployed on a larger scale. In [11], authors have described the capabilities and architecture of a portable, interoperable, tactical operations centre communication system, which was funded by the U.S. Department of Homeland Security. The work evaluates wireless mesh network as the key solution for emergency and rural applications. The authors suggest that despite its unique challenges, the wireless mesh network is a technically suitable yet affordable approach with significant payoffs. In [12], authors have proposed a ballooned WMN to ensure communication in the disaster area promptly. The WMN devices are attached to physical balloons capable of hovering at an altitude but anchored to the ground. It is suggested that by combining multiple ballooned wireless networks, an ad-hoc network can be organized in the sky to provide urgent communication means in disaster areas.

The above-mentioned systems [9, 10, 11, 12] are interesting examples of WMN; however, they are complex, inefficient, and uneconomical for large scale deployments. For example, the WMN system in [9] uses dedicated nodes (called mesh routers) in the network for data collection, routing, and monitoring. Additionally, end-user devices collect and send audio, visual, and infrared data to the control centre via mesh networking only. On the other hand, our proposed system has a simple design where dedicated pushbuttons are used to send extremely small size emergency messages either through WiFi or mesh. This reduces the cost of each device and saves precious network bandwidth while improving connectivity. Each end-device in our system is capable of working as a router, thereby forwarding emergency messages through multi-hop communication. Another example is [12], where the system provides connectivity to devices such as mobile phones and laptops via a mesh network. Their setup requires additional equipments which are vulnerable to damage by bad weather and hence, needs periodic maintenance by emergency response authorities. Such an arrangement would work well for short deployments (during natural disasters) but won't be cost-effective for large-scale long-period deployments. On the other hand, our proposed system can be protected by simple IP67 weatherproof enclosures and does not require constant maintenance. Considering the limitations of above-mentioned systems, it is essential to develop a simple, efficient, and robust emergency response system which can minimize deployment and maintenance costs for large scale deployments.

3. Problem Formulation and Solution Description

Our problem focuses on the design of an efficient "Emergency Response System" which can provide a simple IoT-based end-device to elderly and disabled citizens for requesting emergency assistance. The system should also provide a centralized monitoring platform/application for medical and security personals to keep track of end-devices and emergency events. Finally, the system should provide reliable communication between end-device and monitoring platform in case of network failure. The contributions of our work are as follows.

- We have designed an intuitive IoT hardware prototype which allows citizens to request for Medical and Security assistance by simply pressing a button.
- We have developed a central monitoring application which allows authorities to monitor hardware prototype units and keep track of the emergency events' data (location, route, status, etc.).
- We have developed a robust mechanism for emergency message transmission from hardware prototype units to central monitoring application using multi-hop mesh and WiFi communication.
- We have analyzed the performance of our proposed system in terms of delay, message delivery ratio, robustness, energy, and cost via simulations and field experiments.

4. Design Science Research Framework

Before introducing the MeshSOS system, we briefly discuss the research methodology adapted behind its development using the Design Science Research (DSR) framework [13, 14]. Figure 1 shows six main steps in our framework viz., Problem Motivation and Identification, Objectives, Design and Development, Experimental Setup, Performance Evaluation, and Communication. As shown in Figure 1, these steps can be further represented as five possible research phases. In the first phase, we identify the research problem, its motivation, and challenges. In the second phase, we define the objective as the design and development of a simple and robust emergency response system for elderly and disabled citizens. We have already discussed the first two phases in Sections 1 and 2. In the third phase, we design and develop the IoT

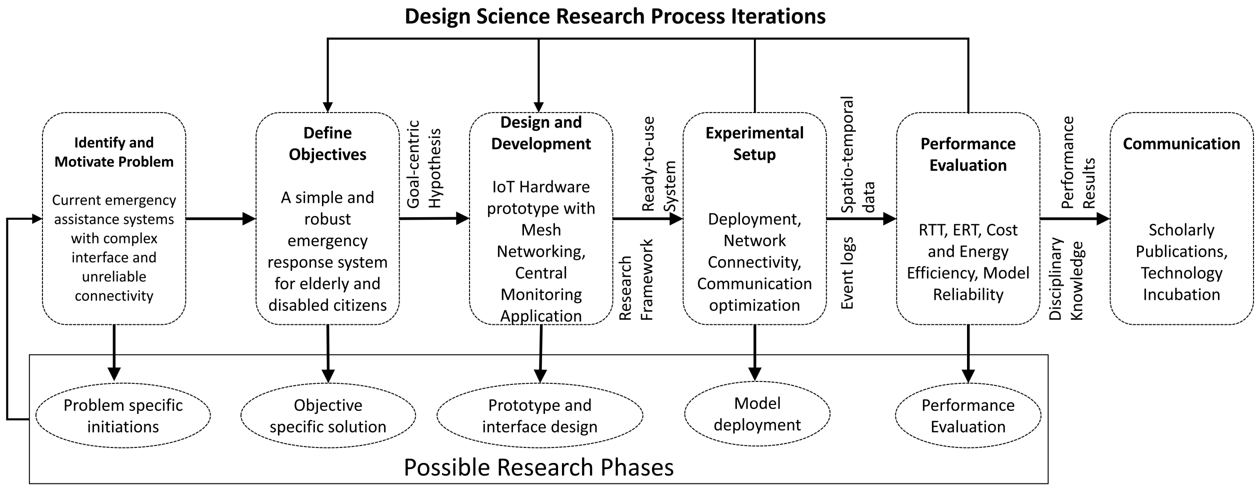

Design Science Research Process Iterations
Identify and Motivate Problem
Current emergency assistance systems with complex interface and unreliable connectivity
Define Objectives
A simple and robust emergency response system for elderly and disabled citizens
Goal-centric Hypothesis
Design and Development
IoT Hardware prototype with Mesh Networking, Central Monitoring Application
Ready-to-use System
Research Framework
Experimental Setup
Deployment, Network Connectivity, Communication optimization
Spatio-temporal data
Event logs
Performance Evaluation
RTT, ERT, Cost and Energy Efficiency, Model Reliability
Performance Results
Disciplinary Knowledge
Communication
Scholarly Publications, Technology Incubation
Problem specific initiations
Objective specific solution
Prototype and interface design
Model deployment
Performance Evaluation
Possible Research Phases

Figure 1: Design Science Research Framework

hardware prototype (discussed in Section 5.1 and 5.2) and a central monitoring application (Section 5.5). In the fourth phase, we perform field experiments (refer to Section 5.4) based on ready-to-use system and research framework acquired from the previous state. In the fifth phase, we evaluate the performance of our proposed system in terms of delay and message delivery ratio (discussed in Section 6). Finally, based on the domain knowledge acquired from experimental results, we aim for the scholarly publications and technology incubation at our organization.

Our DSR framework follows an iterative process where we iteratively update the objective, design and development of our proposed system to improve its performance further. We are finished with the first iteration, and the first version of hardware prototype and central monitoring application is finalized. We have evaluated the performance of our system via simulations and field experiments. Currently, we are in the final step (Communication) with the review process of this research article. Based on the results of our simulations and field experiment, we are now planning to make changes in the design of the prototype unit and monitoring application for future large-scale deployments.

5. Emergency Response System

Our proposed emergency response system has four major components: 1) IoT Hardware Prototype 2) Software and Services 3) Networking and Data Collection and 4) Central Monitoring Application. In this section, we discuss these components in detail.

5.1. IoT Hardware Prototype

We have designed an IoT based hardware prototype which allows its user to request for emergency assistance by merely pushing a button. The prototype is designed using an Arduino compatible microprocessor board called Argon from Particle Inc [15]. Argon boards support multiple wireless technologies such as Bluetooth 5.0 and 802.11 b/g/n WiFi. Additionally, these boards have out-of-the-box 802.15.4 mesh networking capabilities which can be configured using Google OpenThread[3]; an open-source implementation of the Thread networking protocol. Due to these integrated wireless technologies, an Argon board in a mesh network can act as an end-node (thereby collecting end-user input/data) or a relay (thereby forwarding traffic to other boards using 802.15.4 mesh) or a gateway (thereby connected to the Internet using WiFi). The schematic of our hardware prototype is shown in the Figure 2 which includes two momentary push buttons, an 8 dBm external antenna, USB power supply, on-board RGB status LED, and an external Li-Po 3.7V battery. Two push buttons are configured to request for two different types of emergencies viz., Medical and Security. An external antenna is added to provide long-range transmission capabilities for direct WiFi and mesh communication. The prototype can be powered using a USB power supply or using an external battery in case of a power outage. Note that, an equivalent IoT prototype can also be designed using much cheaper ESP-32 microprocessor boards with additional hardware modules for the external antenna, RGB LED, Li-Po

[3]OpenThread. `https://openthread.io/`

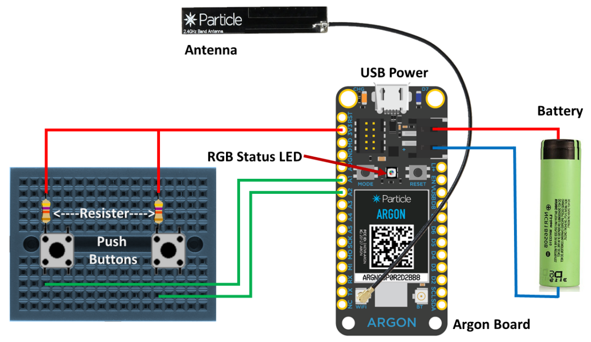


Figure 2: IoT Hardware Prototype

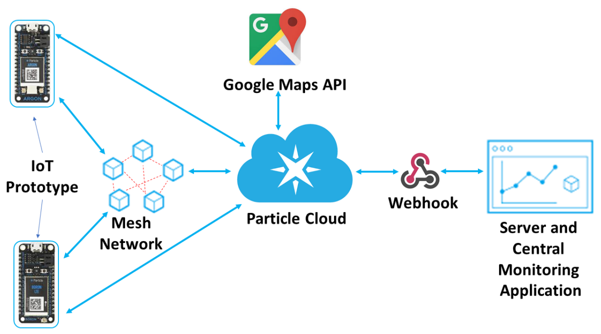


Figure 3: Software Components and Services

charging circuit, and battery connector [16]. However, to avoid the complexity of connecting multiple modules together with unreliable wired connections, we chose to use all-in-one Argon boards.

5.2. Software Components and Services

In this section, we describe all the software components and services used in the development of our proposed emergency response system as shown in Figure 3.

5.2.1. Particle Workbench. To code, compile, and flash the Argon board with the source code, we make use of Particle Workbench [17]. This workbench provides various tools, libraries, and extensions for offline and online (cloud) compilation of the source code and supports OTA (Over-The-Air) interface for future updates. Additionally, a command-line interface (CLI) is also available which can be used to flash individual Argon boards and to accept serial outputs via a USB cable.

5.2.2. Particle Cloud. Particle Inc provides a device cloud to manage different types of Particle IoT devices/boards (such as Argon, Boron, Photon) in a secure, scalable, and reliable manner [18]. The cloud communicates with the IoT boards using a publish-subscribe model. The cloud also supports the integration of third-party APIs such as Google Maps and HTTP webhooks. The main objective of Particle cloud in our emergency response system is to process the emergency messages and deliver them to the central server. The cloud is also responsible for publishing response/acknowledgement messages to the Argon boards. The Particle cloud essentially binds all the software and hardware components in our system.

5.2.3. Google Maps API. Our prototype keeps track of its GPS location using Google Map's Geolocation service [19]. The Argon boards should be connected to the Internet using WiFi to use Geolocation service. Apart from providing GPS location (with 30-40 meter accuracy), Geolocation service also computes an optimal route to reach the board/event location. Since each prototype unit is expected to be fixed within a home/office region, constant calculation of its precise location is not necessary (hence, avoiding additional GPS/GLONASS hardware modules).

5.2.4. HTTP Webhook. Webhooks are user-defined HTTP callbacks that enable an application to provide data to other applications efficiently in real-time [20]. Webhooks eliminate the need for constant polling by an application to fetch data. Our system requires that emergency messages are forwarded to the central server as soon as the Particle cloud receives it from a board. Thus, it is necessary to link the server to the Particle cloud using a webhook that updates the server immediately. The support for webhooks is provided in the form of Particle cloud Integration. When the cloud receives a message from a board, the webhook fires an HTTP POST request with the message in JSON format to an API endpoint on our server. The server checks for the validity of the data and appends an acknowledgement in response to the request.

5.3. Workflow

When a user presses a button on the prototype unit, an emergency message is generated which we refer to as SOS message from now on. We use the term 'Source' for the Argon board, which generates the SOS message. The term 'Relay' is used for the Argon boards which

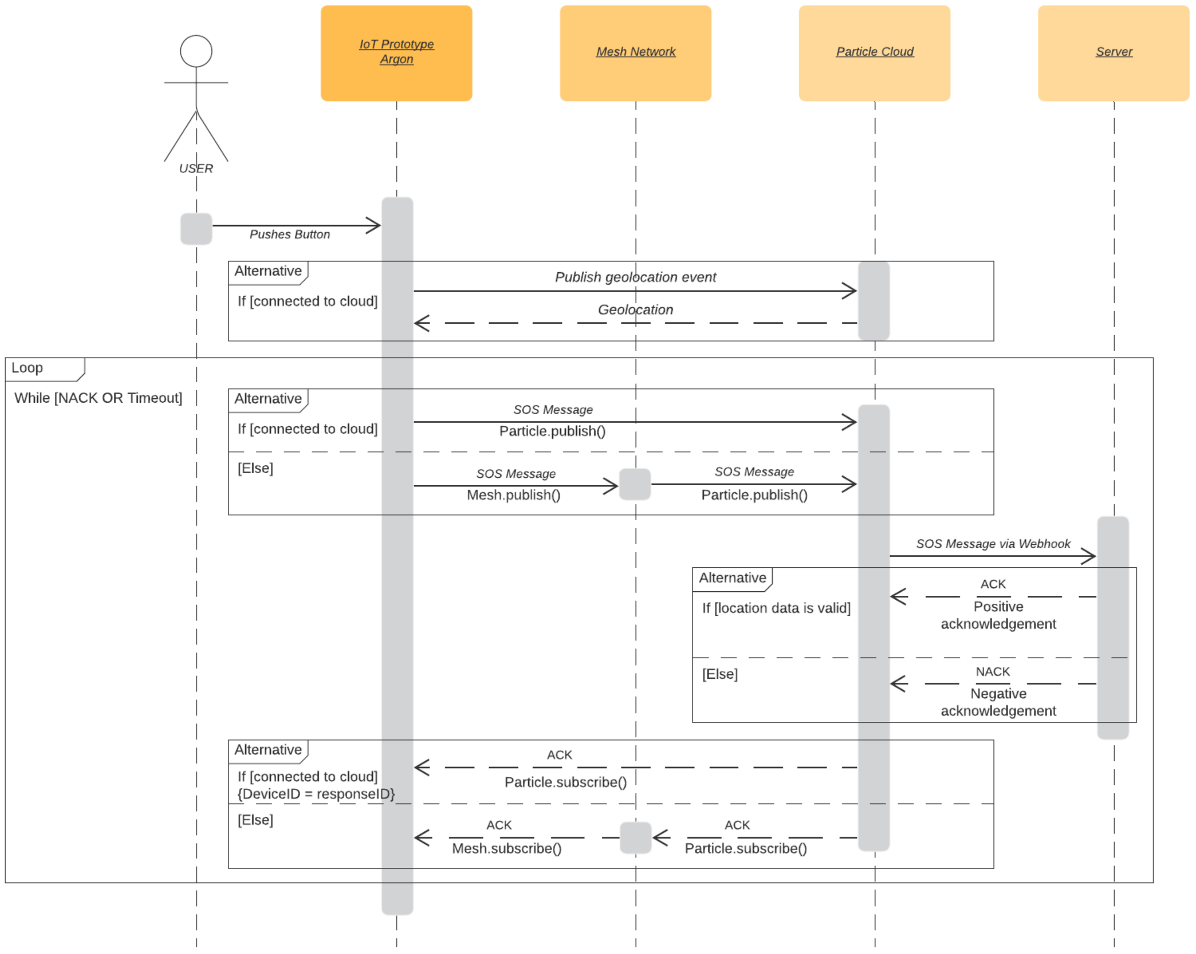


Figure 4: Sequence Diagram

forwards/relays the SOS message to Particle cloud or to other boards in the mesh network. The sequence diagram in Figure 4 summarizes the workflow of our proposed emergency response system. Broadly, there can be two possibilities based on whether source Argon board has Internet connectivity or not:

1. If the source board is connected to the Internet: In this case, the board first fetches its geographic location from the Particle cloud. Then, the board publishes an SOS message to the Particle cloud directly using the WiFi Internet connection. Finally, the webhook integrated into the Particle cloud sends the data to a central server for data storage and analytics. If the server successfully processes the SOS message, a positive acknowledgement (ACK) is sent from the server to the Particle cloud. If the server is unable to process the SOS message due to any reason (For example, Message corruption) and then a negative acknowledgement (NACK) is sent. Based on the received ACK (or NACK), Particle cloud sends the ACK (or NACK) to the source board. On receiving the ACK, the status of successful delivery is shown to the user by blinking the RGB status LED. In the case of NACK or timeout, the process is repeated until ACK is not received. Note that, in this case, the mesh network is not involved in SOS message transmission.

2. If the source board does not have an active Internet connection: In this case, the source board floods the SOS message to all the neighbouring Argon boards (in the communication range) using the mesh network. After receiving the SOS message, neighbouring boards without an active Internet connection floods the SOS message to its own neighbours and so on. We put a limit on the number of times an SOS message can be flooded in the mesh to avoid network congestion. If any neighbouring board have an active Internet connection, it can publish the SOS message to the Particle cloud. This further leads to 2 sub-cases:

- The source board has valid but outdated location information: In this case, the relay board publishes the SOS message received from the source board to the cloud. Then, integrated webhook sends the data from the cloud to a central server.
- The source board does not have any location information: In this case, the source board attaches a flag to the SOS message. Relay board upon receiving the flagged SOS message, append its own location information to the SOS message and publishes it to the Particle cloud. Then, integrated webhook sends the data from the cloud to a central server.

Upon successful delivery of the SOS message, ACK is sent from the server to cloud, then from cloud to relay board, and finally from relay board to the source board. In case of an unsuccessful delivery, NACK follows the same path from server to the source board, and the entire process is repeated until an ACK is received or a timeout occurs. The boards in the mesh network and the server use the publish-subscribe model to communicate. As soon as an SOS message is generated and successfully published to the Particle cloud, a webhook is fired, which sends the data to the central server. This ensures that the data is forwarded instantaneously and no time is wasted in waiting for the server to start the synchronization process. Acknowledgements from the Particle cloud are sent the source boards via relay unless the acknowledgement was meant for the relay board itself.

5.4. Networking and Data Collection

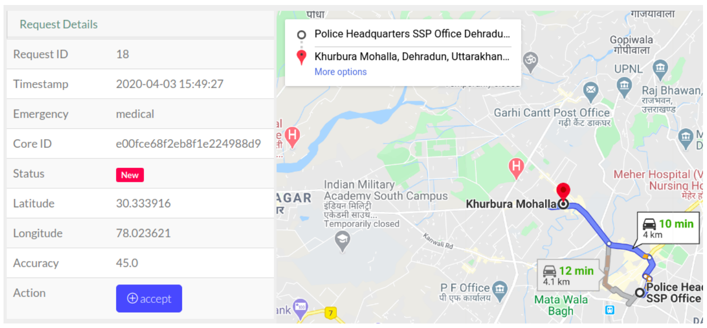
The primary motivation for using the mesh network along with WiFi is to provide redundancy for guaranteed message delivery. If a user presses a button, then SOS message must be delivered to the server either through WiFi or mesh network. Not only that, the acknowledgement of the SOS message should also be delivered to the source board. Keeping this in mind, our IoT prototype is programmed to constantly look for neighbouring boards in its communication range (for mesh networking) as soon it is powered up. Additionally, in the background, the IoT prototype constantly looks for available WiFi connections and tries to connect to a known access point. Once a WiFi connection is established, the board maintains its connection to the Particle cloud. This ensures that the board does not stall the boot process due to unavailability of WiFi Internet connection and that the board can publish SOS messages in the mesh network at all times.

The central server receives data related to the emergency events from the Particle cloud. It is the responsibility of the central server to handle duplicate SOS messages pointing to a single emergency event, which occur due to publish of an SOS message from multiple relay boards in the mesh network (a redundancy by design). The server maintains logs of all the emergency SOS messages and store related data in an SQLite database. The server also provides REST API endpoints to connect with the Particle cloud and to synchronize the monitoring application with the database.

5.5. Central Monitoring Application

The Central Monitoring Application enables the administrator to view all the details related to emergency events. The application also provides Spatio-temporal information of the events. The application is designed to be simplistic, informative with a focus on efficient operation management. The application can list all the emergency logs at once or show them filtered into three different categories viz., medical and security. After receiving an SOS message, the administrator can click on the log ID to view the details of individual events and process them, if necessary (Figure 5a). The timestamp field shows the time at which the event occurred, and the emergency field displays the type of incident that occurred. The status contains labels to show message category wiz., New, Processing, and Resolved. The action field allows the administrator to take necessary actions. Additionally, the application displays the location of the incident on a map and the shortest route to the location of the event from a fixed location (Hospital or Police Station).

Figure 5b represents the *Analytics* tab which provides a 3D visualization of the data events on a map. The height (and colour gradient) of the columns represent the number of cases in a geographical region. The administrator can hover the mouse pointer over a column to view the number of events in the region. The administrator has the option to modify the location of the emergency response centre, and this updated information reflects across the system immediately. Additionally, the map will centre around the updated emergency response centre location, and the starting point of the shortest routes to the event location will be changed instantaneously.

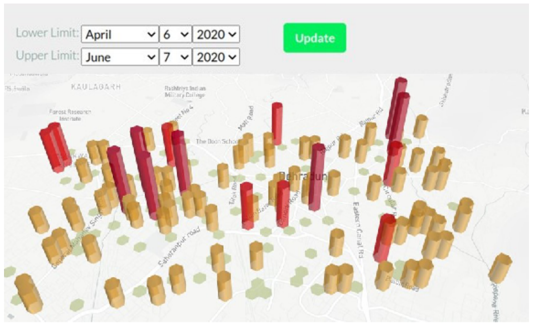


Request Details

Request ID	18
Timestamp	2020-04-03 15:49:27
Emergency	medical
Core ID	e00fce68f2eb8f1e224988d9
Status	New
Latitude	30.333916
Longitude	78.023621
Accuracy	45.0
Action	⊕accept

Police Headquarters SSP Office Dehradu...
Khurbura Mohalla, Dehradun, Uttarakhan...

(a) Event Details and Route Planner



Lower Limit: April 6 2020
Upper Limit: June 7 2020

(b) Spatio-Temporal Information

Figure 5: Central Monitor Application Visualization

6. Performance Evaluation

To evaluate the performance of our proposed system, we have performed field experiments by deploying multiple prototype units in a geographical region. For diversity, we have used both broadband WiFi and mobile hotspots to provide wireless connectivity to the prototype units. To further diversify the environment, we have varied the network bandwidth from 120 Kbps to 50 Mbps. The firmware of Particle Argon boards is modified to automate the message sending for field experiments. A total of 15,000 events are triggered and data is recorded at the central monitoring application. We evaluate the performance of our emergency response system in terms of Round Trip Delay (RTD), Event Resolution Time (ERT), and Delivery Failure Ratio (DFR). RTD is defined as the time required for an SOS message to travel from source board to server plus the time needed for an acknowledgement (ACK or NACK) to travel from server to the source board. In case of unreliable network connectivity, prototype units may take multiple attempts to successfully deliver the SOS message at the central monitoring application or mark delivery as failed. ERT is the sum of RTD of all the attempts (successful or failed). DFR defined as the percentage of emergency messages fail to deliver at the central monitoring application.

Figure 6 represents the RTD and ERT for the case when the source board is connected to the Particle cloud using a WiFi connection. Each point on the horizontal X-axis corresponds to an emergency event (Button press). Y-axis represents the time in milliseconds. Note that, to avoid clutter in the graphs, we have plotted only 200 points which are randomly selected from 15,000 data points collected during a field experiment. Figure 6 represent the RTD and ERT values for emergency messages when prototype units are connected to the Internet. In some occasions, an erroneous SOS message gets delivered to the server and NACK is returned to the source board. The sharp peaks in the ERT represent such occasions where multiple re-transmissions are required before receiving an ACK from the server. This causes ERT to have a higher value than RTD. The observed DFR for this case is 1.2% resulted from timeouts happened during location fetch operations.

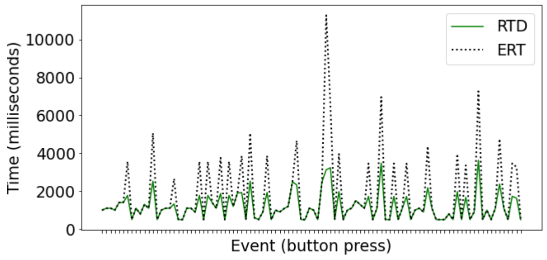


Figure 6: Source board is connected to the Internet

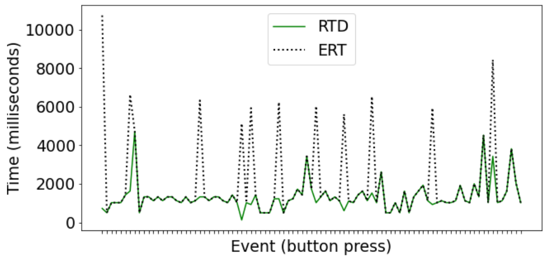


Figure 7: Source board is not connected to the Internet but has the location information

Figure 7 represents the RTD and ERT values for the case when the source board has the location information but does not have an active Internet connection to update the location to the cloud. In this case, avoiding location fetch (as shown in the sequence diagram) saves some

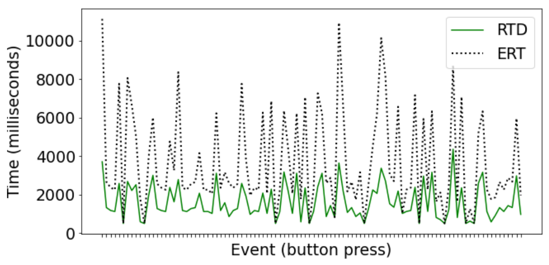


Figure 8: Source board neither has Internet connectivity nor the location information

time. However, message delivery takes extra time as it travels through the mesh network to reach the server. The observed DFR for this case is 0.08%, which is the lowest among all the cases. This is due to the fact that location fetch operation is not required before SOS message transmission. In case of absence of Internet connectivity and location information, the values of ERT are almost double of RTD, as shown in Figure 8. This is due to the fact that when the first SOS message sent to the server without any location information, a NACK is returned. In the second attempt, the relay board appends its own location information, and the server returns an ACK. Due to this second attempt, the ERT values are always higher then RTD values. The DFR for this case is around 1.68% which is the highest among all cases.

The average RTD and ERT values for all three cases are shown in Table 1. As can be observed that the average RTD values are always lower than average ERT values for all the cases. Additionally, average ERT values are the highest when the source board neither has the Internet connectivity nor the location information.

Case	**RTD**	**ERT**
Internet access	1348.68	1781.15
No Internet access with location info.	1395.84	2077.88
No Internet access without location info.	1784.67	3779.15

Table 1: Average RTD and ERT values

6.1. Real-World Deployment Analysis and Implications

To analyze long-term feasibility and profitability of our proposed system in real-world deployments, it is necessary to examine its performance in terms of robustness, energy, and cost. Due to the rarity of emergency events, and a limited number of prototype units and WiFi access points, we have performed a numerical simulation along with field experiments. Simulation region is logically partitioned into 20 blocks, creating a grid of size 5×4. All the prototype units within a block are connected to a WiFi access point deployed at the centre that block. Prototype units not connected to WiFi can use mesh technology, if available. On average, seven prototype units are randomly distributed in each block. The communication range of WiFi is considered to be four times the range of single-hop mesh communication. We analyze the robustness of the system in terms of percentage of prototype units connected to the Particle cloud (and consequently, the central monitoring application) either via WiFi or mesh networking.

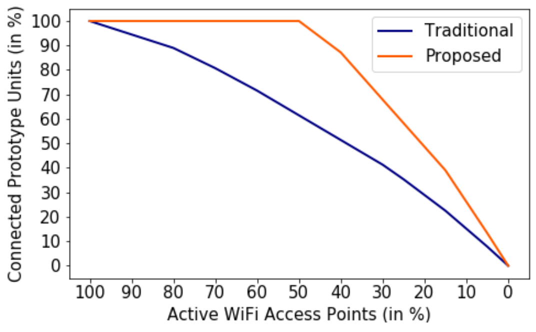


Figure 9: Connected Prototype Units

For performance evaluation, we have compared two different emergency response systems. First is the *Traditional* system where prototype units are equipped with WiFi only. The second is the *Proposed* system where prototype units are equipped with both WiFi and mesh to support multi-hop (max five hops) transmission of SOS message from prototype units to the Particle cloud. The simulation starts with 20 *active* WiFi access points. In the beginning, both traditional and proposed systems perform the same because all the prototype units are connected to the Particle cloud via WiFi. As the number of active WiFi access points decreases, the number of connected prototype units keeps decreasing for the traditional system, as shown in Figure 9. However, for the proposed system, the number of connected prototype units remains constant when at least 50% WiFi access points are active. This is due to the fact that even with a lower number of WiFi access points, mesh communication guarantees network connectivity to prototype units via multi-hop communication. Deactivating more than 50% WiFi access points reduces the number of connected prototype units for both the systems. However, our proposed system is always providing connectivity to a higher number of prototype units, and hence, has higher robustness compared to the traditional system.

In our proposed system, to provide connectivity to all prototype units, we require a lower number of WiFi access points which directly translates into a better cost and energy efficiency. For example, Figure 9 reveals that the proposed system reduces the one-time deployment cost (i.e., cost of WiFi access points) and recurrent operational cost (i.e., electricity charges) to half by deploying only 50% of WiFi access points as compared to the traditional systems. This will accelerate the large-scale deployment of emergency response systems to realize the smart city initiative.

7. Conclusion

In this paper, we proposed an emergency response system for handling medical and security-related emergencies. Our proposed system consists of mesh-enabled IoT prototype units which allow users to request for assistance even in the absence of an Internet connection. Additionally, a central monitoring application is developed to provide a simple and efficient event management portal to the relevant authorities. In the future, we plan to integrate LTE/LoRaWAN communication capability into prototype units to provide long-range communication capabilities.

Acknowledgement

This research work is supported by the Faculty Initiation Grant (FIG) of Indian Institute of Technology Roorkee, INDIA.

References

[1] Comptroller and Auditor General, "Compliance and performance audit on general and social sector of government of kerala." [Online]. `https://bit.ly/2DOMBZr`, 2015.

[2] V. Venugopal and S. Yilmaz, "Decentralization in kerala: Panchayat government discretion and accountability," *Public Administration and Development: The International Journal of Management Research and Practice*, vol. 29, no. 4, pp. 316–329, 2009.

[3] National Health Systems Resource Centre, "Study of emergency response service - emri model." `https://bit.ly/36p0cCH`, 2010. [Online].

[4] Y. T. Michelle, B. Ramesh, R. V. Srinivasan, V. Arun, and H. P. TAN, "Aging-in-place: The journey ahead," tech. rep., Tata Consultancy Services Limited, October 2020. `https://on.tcs.com/3cZDqmj`.

[5] L. Chen, C. Tsai, W. Chang, Y. Cheng, and K. S. Li, "A real-time mobile emergency assistance system for helping deaf-mute people/elderly singletons," in *IEEE International Conference on Consumer Electronics (ICCE)*, pp. 45–46, 2016.

[6] Y. Liu, K.-F. Tong, X. Qiu, Y. Liu, and X. Ding, "Wireless mesh networks in iot networks," in *2017 International Workshop on Electromagnetics: Applications and Student Innovation Competition*, pp. 183–185, IEEE, 2017.

[7] R. K. Kodali, M. Azman, and J. G. Panicker, "Smart control system solution for smart cities," in *2018 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pp. 89–893, 2018.

[8] M. Khalil and A. A. Alvi, "Utilizing wireless networks for public safety: An overview of special use cases," in *4th International Conference on Emerging Trends in Engineering, Sciences and Technology (ICEEST)*, pp. 1–6, 2019.

[9] M. Gao, F. Zhang, and J. Tian, "Wireless mesh network for emergency response system based on embedded system," in *International Conference on Embedded Software and Systems Symposia*, pp. 361–365, 2008.

[10] Braunstein and B. et al, "Feasibility of using distributed wireless mesh networks for medical emergency response," in *AMIA Annual Symposium*, pp. 86–90, 2006.

[11] Yarali, A. . Ahsant, B. . Rahman, and Saifur, "Wireless mesh networking: A key solution for emergency & rural applications. advances in mesh networks," in *Second International Conference on Advances in Mesh Networks*, pp. 143–149, 2009.

[12] Y. Shibata, Y. Sato, N. Ogasawara, and G. Chiba, "Ballooned wireless mesh network for emergency information system," in *22nd International Conference on Advanced Information Networking and Applications - Workshops*, pp. 1118–1122, 2008.

[13] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of management information systems*, vol. 24, no. 3, pp. 45–77, 2007.

[14] J. Venable, J. Pries-Heje, and R. Baskerville, "A comprehensive framework for evaluation in design science research," in *International Conference on Design Science Research in Information Systems*, pp. 423–438, Springer, 2012.

[15] "Particle Argon: Wi-Fi + Bluetooth." `https://docs.particle.io/argon/`. [Online].

[16] "ESP32: A feature-rich MCU with integrated Wi-Fi and Bluetooth connectivity for a wide-range of applications." `https://www.espressif.com/en/products/socs/esp32`. [Online].

[17] "Partcle Workbench: Professional IoT development on Windows, macOS, Linux, powered by Visual Studio Code." `https://www.particle.io/workbench/`. [Online].

[18] "Device Cloud: A secure, scalable, and reliable cloud platform to manage your fleet of IoT devices." `https://www.particle.io/workbench/`. [Online].

[19] "Google Maps Platform Documentation." `https://developers.google.com/maps/documentation`. [Online].

[20] "Webhooks." `https://docs.particle.io/tutorials/device-cloud/webhooks/`. [Online].