International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys.14, No.1 (Oct-23)

http://dx.doi.org/10.12785/ijcds/140190

A Multi-Agent System Approach for Emergency Services using QoS-Assured Cloud Computing Architecture in the Metropolitan Transportation System

Awais Qasim[1], Adeel Munawar[2,3] and Mongkut Piantanakulchai[2]

[1]*Department of Computer Science, G.C. University, Lahore - Pakistan*
[2]*Sirindhorn International Institute of Technology, Thammasat University Bangkok, Thailand*
[3]*2Departement of Computer Science, Lahore Garrison University, Lahore, 54000, Pakistan*

Received 28 Sep. 2022, Revised 23 Aug. 2023, Accepted 21 Sep. 2023, Published 1 Oct. 2023

Abstract: The paradigm of multi-agent systems has been actively applied to cloud computing to provide cost-effective, decentralized problem-solving. Rather than processing the tasks that require enormous computing resources locally, they can be processed in the cloud. In this research, we present how these two paradigms can be effectively used to handle emergency services in any metropolitan transportation system. In the proposed architecture, agents are placed at signal nodes to collect data and send it to cloud services. These agents have the ability to control the signal nodes in an emergency. They also communicate emergency information between several nodes. The cloud service is the system's backbone and employs heterogeneous agents to manage emergency services. It analyses the input data and determines the presence of emergency services. The main contribution of the research is to utilize a priority-based mechanism for handling exigency vehicles rather than using traditional timer-based signals. In addition, we provided case studies demonstrating the usefulness of the proposed framework.

Keywords: Cloud computing, Emergency services, Traffic management, Multi-agent system

1. INTRODUCTION

Multi-Agent Systems (MAS) have become increasingly significant over time due to their ability to address intricate real-life challenges.[1][2][3]. This success is because each agent can act as a decision-making entity; hence, a large problem can be subdivided among them, leading to a common universal goal[4]. MAS-aided designing of solutions can be upturned by the following terms, (a) Distributed problem architecture, (b) Dynamic environment consisting of the subsystems (c) communication flexibility of subsystem. Since the transportation system is dynamic and geographically dispersed, this makes agent-based methodology applicable [5][6]. At the same time, cloud computing is one of the emerging technologies as it provides an enormous amount of computing resources on the internet. Extensive data processing can be done these days without purchasing any hardware. With the help of cloud computing, people can rent resources to fulfill their needs [7]. Commonly there are three types of services provided by cloud computing, i.e., IAAS, PAAS, and SAAS. Cloud computing and Multi-agent systems belong to two different domains. Cloud computing services and Multi-agent integration can solve many problems in a real-time environment (RTE). Major areas of research in cloud computing are virtualization, energy efficiency, data storage, and the efficient use of the computing infrastructure [8]. Multi-agent systems can perform more efficiently by utilizing cloud computing power and storage capacity. Till now, researchers have presented numerous solutions for the automation of traffic signals using the paradigm of Multi-agent systems, but none have discussed the issue of exigency services. Emergency service refers to any vehicle requiring priority over the others, like ambulance, fire brigade, theft vehicle detection, and many more. Quality of service (QoS) assurance is a significant factor in such scenarios [9]. Many QoS attributes are related to cloud service sellers and purchasers. However, cloud service providers are primarily concerned about stability, data center running, and consistency. In our research, cloud service response time and heterogeneous-agent communication will be the highlighted factors while describing the proposed architecture. These kinds of proposed architectures are based on the Multi-agent system, which is the ultimate part of many modern applications, especially modeling and simulation. However, utilizing cloud-based resources to manage emergency vehicles is a novel concept. In this paper, we have proposed a system that consists of Multi-agent and cloud-based resources to handle the critical application of the transport system of urban areas.

E-mail address:awais@gcu.edu.pk, adeelmunawar43@gmail.com, mongkut@siit.tu.ac.th

2. RELATED WORK

In [1], a survey on the evolution and applications of multi-agent systems was presented. The paper describes the behavior and background structure of multi-agent systems. It also presents a mechanism to minimize the computation cost by early removing certain agents to solve the global optimization problem. In the study[4] presents a comprehensive view of traffic management systems based on reinforcement learning. It presented a detailed case study that showcases the developed architecture using multi-agents. Three reinforcement algorithms were evaluated for experimental purposes. It also discussed the challenges of using reinforcement learning in multi-agent systems. In [5], parallel systems for traffic control were elaborated. By juxtaposing, all modules of traditional traffic control and peer mechanism are studied. A study [10] proposed an agent with human-like communication skills and extensive destination knowledge capable of guiding tourists globally to bridge the gap left by traditional tourist facilitator systems. In [7], cloud computing-related issues and challenges are given. In [8], cloud computing architecture and its modules were discussed. It described the use of cloud computing in applications with parallel and distributed environments. It presented equivalences, differences, and possible cooperation between cloud computing and multi-agent systems. In [9], a heterogenous agent-based system is proposed that consists of physical devices and virtual resources, clouse services provision layer and cloud service management. Moreover, it supported QoS-assured cloud service provision and requests. In [11], parallel systems for traffic control were carefully been studied. The paper explained the significant modules of traditional traffic control methods and the corresponding parallel traffic control approaches. [12] emphasized the development of novel distributed team-triggered methods that combine self-triggered to let the agent decide congently based on its neighbor. In [13], microscopic simulation-based cellular automata and multi-agent systems were demonstrated. The simulation plays an imperative role in supporting the analysis of the intricacy of the traffic system and can render the present information about road capability. In [14], ACP (artificial, computational, parallel approach) played an indispensable role in system construction, integrated control, and urban transportation management. ACP methodology consists of modeling with artificial systems, operation through parallel execution, and analysis with computational experiments for appropriately controlling the complex system. [15] presented the main issues, current applications, and background effects of the parallel transport management system. It applied a data-driven approach to the decision-making and modeling of the system. Also, it considered the social complexity and engineering in its process. In [16], it was analyzed how MAS can be effectively used to route the traffic in commonly used different modes of transportation. It also addressed minor issues of the multi-agent system of flexibility and extensibility. In[17], addresses conceptually two underlying questions of emergency convoys. The first question deals with how to optimally control the normal flow of traffic, while the second question addresses the issue of controlling the flow of an emergency convoy. [18] proposes a framework with three phases, i.e., e-questioner development, grouping society process, and tactical adaptation process. In the first phase, interviews are taken with people from different areas and of different concerns in selected universities. In the next phase, data collected in a MAS environment is analyzed for decision-making. After analysis, based on the preferences of people, processed results are displayed on the web. This method of finding the preferences of different people makes a self-adaptive website that displays the contents according to their preferences. In [19], a MAS-based system is studied to control the central controlling system of an electric propulsion plant on ships. These kinds of exploration are the base of the multi-agent control system (MACS) for the complex cooling system. In [20], the concept of resolving the issues related to repairing and maintaining the equipment within the available funds. It covers all the functions required for the maintenance services. The idea compromises an agent-based approach, which regulates the Maintenance, Repair, and Operations (MRO) process, developed for repairing and maintenance purposes.[21] used the concept of multi-agents for protein synthesis. This system automatically combines the proteins by exploiting the existing DNA sources. They are further integrated into a database as long as the source of DNA exists. Biologists can utilize this type of database in a variety of fields, including medical, pathological, etc. [22] proposed the design and implementation of a multi-agent system with an International Space Station (ISS) interceptor module, which enhances the security of web applications and the Legal Aid Management System (LAMS).The II-ISM is an extension of the IIS server, which listens to HTTP requests made by the web applications and LMS Moodle. The MAS architecture was created using the Gaia methodology. Its implementation was carried out using the JADE framework. Currently, a multi-agent system can detect SQL injection and cross-site scripting attacks over any web application. These agent-based systems can be extended by adding components without redesigning the complete system.

3. PRELIMINARIES

This section describes a few terminologies and computational models used throughout the remainder of the discussion to define the problem under analysis

A. ARTIS Agent Architecture

The architecture of the ARTIS agent was proposed in [23]. Based on the blackboard model, it is designed to work in environments of real-time restraints. Fig. 1 shows the basic skeleton of an ARTIS agent. A single agent can control several internal agents (InAgent). The role of these InAgents is to execute the delegated tasks. These InAgents are of two types i.e., Reflex and Deliberative. For tasks that require artificial intelligence techniques, we use the deliberative InAgent. For other tasks without a margin of timeline, we use reflex InAgent. Additionally, each ARTIS

agent has a Control Module that coordinates the control of all the InAgents. A basic version of an ARTIS agent is presented in Fig. 1. The usage of this architecture depends on the scenario in which we plan to use the MAS. In the planning phase, we decide the number of agents, their responsibilities and the create schedulability plan. This plan contains the details of how much time each agent will take to perform its assigned tasks. The major benefit is that we are able to predict the time our complete system will take in deployment mode.

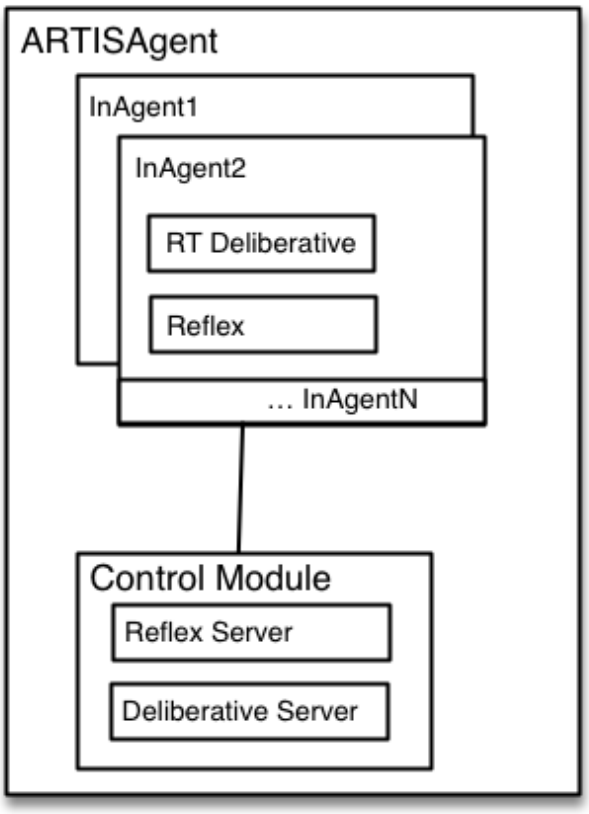


Figure 1. The basic architecture of an ARTIS agent [23]

B. FIPA Performatives

In 1995, the Foundation for Intelligent Physical Agents (FIPA) established agent communication standards. In contrast to the KQML performatives previously used in [24] [25] , FIPA only proposed 20 performatives. It was the first time a sufficient number of performatives that could specify every aspect of agent communication were standardized. Major benefit of having such a vocabulary is that no matter how many different agent architectures are proposed, we have a precise set of actions that can be used to specify their actions. Since no MAS can work without the communication between agents so we can not only specify the agent's requests but also messages sent for heartbeat like ping messages.

4. PROPOSED MULTI-AGENT SYSTEM FOR EMERGENCY SERVICES IN CLOUD ENVIRONMENT

Multi-agent System for Emergency Services in Cloud Environment (MAS-ES-CE) architecture combines MAS and cloud computing services. This combination realizes the environment and sends data to the cloud services to verify emergency service.

A. MAS Architecture

MAS architecture is shown in Fig. 2. It is based on five subsystems: traffic signals controller, agent for sensor, agent for traffic controllers, agent for communication, and IAAS cloud service.

Individually component is time-bound and features the characteristics to work in an RTE. In normal conditions, the signal controller regulates the operation of traffic signals based on the installed algorithm. In the case of a critical scenario, the traffic controller agent instructs to control the signal, and cloud services are responsible for the send a response based on the provided data. The status includes detected or rejected. Fig. 3 shows the proposed architecture of data centers on the cloud. The system's designer determines the configurations of the data centers based on desired parameters such as cost, response time, processing power, load as measured by the number of concurrent requests, etc.

1) Traffic Controller Agent

The main task of this agent is to effectively control the smooth flow of traffic, keeping in view the exigency vehicles. Since the traffic signals are microcontroller based, they send their updates to this agent periodically. After it has reached a decision of opening a signal, it communicates it with the sensor controller agent. The complete working of this agent is real-time to ensure that in case of an emergency, the system will accommodate that vehicle timely. This agent's architecture is modeled after the ARTIS agent, specifically engineered to operate in RTEs.

2) Sensor Controller Agent

The sensor controller agent, a key part of the suggested design, is in charge of keeping track of how each sensor at a traffic signal operates. These sensors are made to capture audio and video data at intervals of five seconds using security cameras with powerful microphones placed strategically on the road. The data is transferred to the cloud services for additional processing and analysis when the sensor controller agent receives it from the sensors. The communication agent plays a vital role in sending the collected data to the cloud services and receiving feedback indicating the detection or rejection of emergency services. The cloud services use advanced data processing techniques to analyze the video data and determine the presence of any emergency services. The Signal Controller agent (SCa) subsequently receives the necessary instructions from the communication agent to operate the traffic signal. The agent's architecture is based on the ARTIS agent, which is

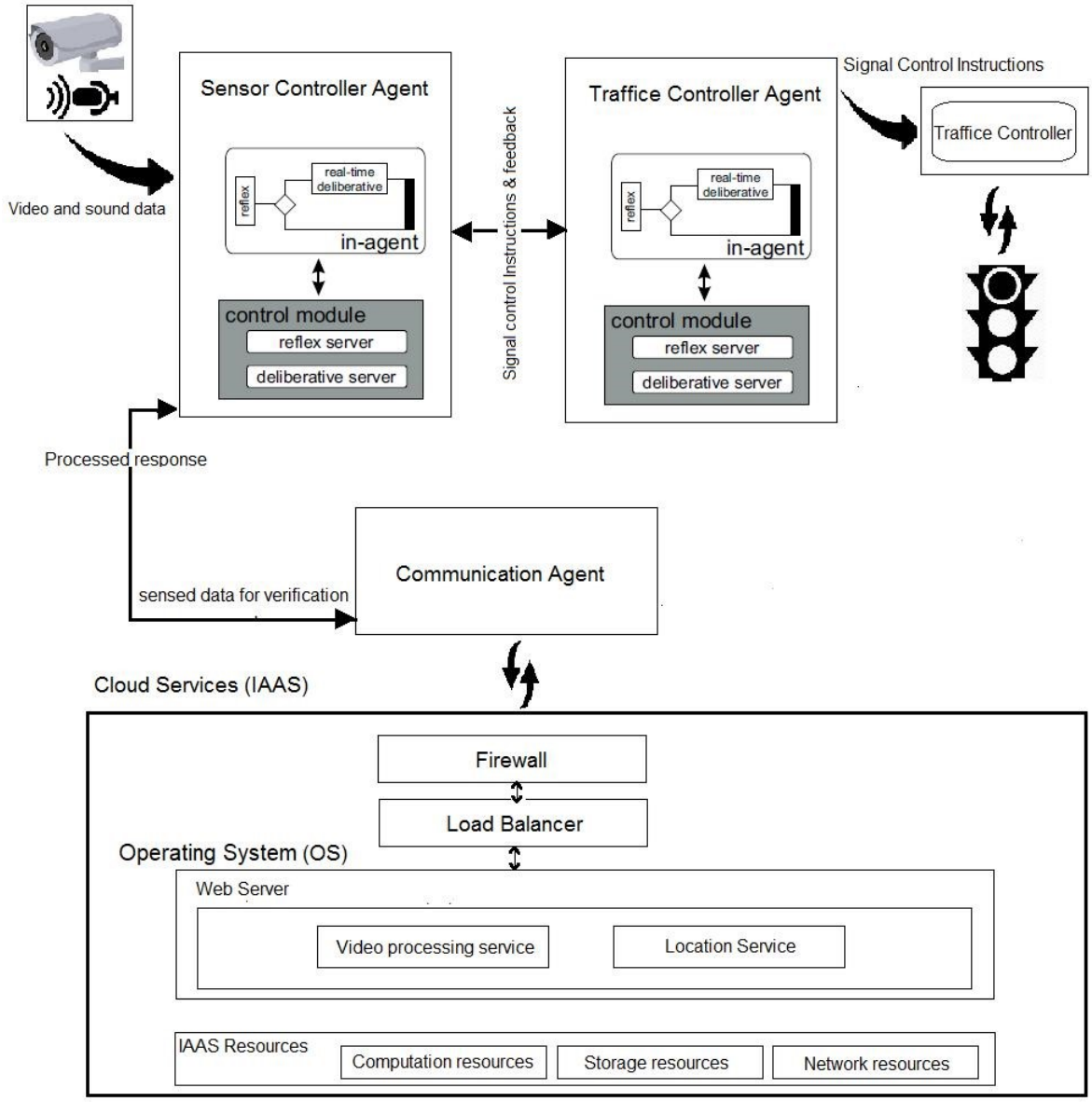


Figure 2. Proposed MAS architecture

designed to meet the needs of RTE traffic automation. The implementation of this suggested architecture reduces the delays experienced by the fire and ambulance departments, which frequently result in fatalities. Together with other agents, the sensor controller agent makes sure that traffic signals are managed smoothly and effectively and that emergency services are promptly accessible.

3) Communication Agent

This agent supports communication with cloud services and inter-node communication across various signals, providing coordinated traffic flow and effective response to shifting traffic patterns. It uses UDP for quick and efficient data transmission.

4) IAAS Cloud services

AAS provides the essential infrastructure, and users can deploy and configure the services and applications according to the requirement. IAAS offers three types of services: image processing, sound processing, and location service. Image processing service processes the video frames and detects the objects, which give the emergency scenario information with the help of image processing algorithms. Sound service recognizes the emergency siren sound. Finally, location service provides the location information about the nearest hospitals.

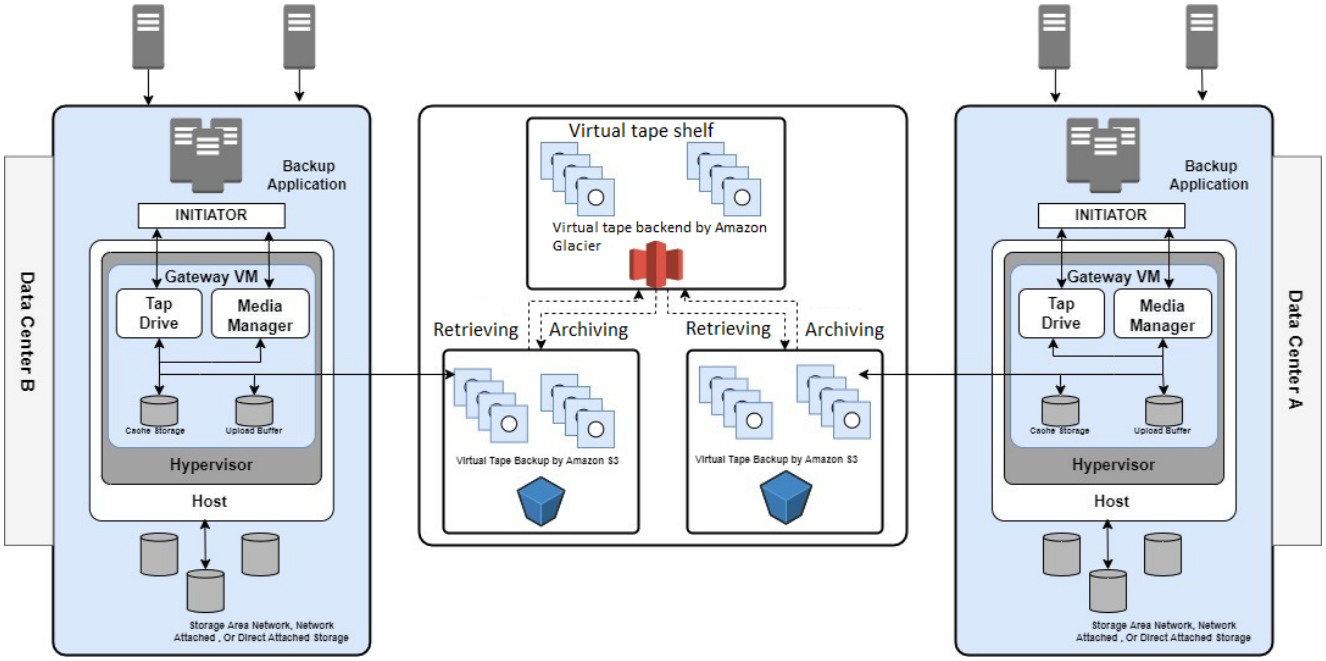


Figure 3. Proposed data centers architecture

B. MAS Operation

MAS operation is divided into the following phases, i.e., detection, recognition, and execution.

1) Detection of Emergency Service

Every 5 seconds, the sensor controller agent at the traffic light records audio and visual data. The communication agent sends this data to the cloud services, using the UDP protocol to send it effectively. This method equips the traffic management system with the knowledge necessary to make timely judgments in the face of shifting traffic patterns. The agent is essential for streamlining traffic, decreasing congestion, and raising road safety standards.

2) Recognition of Emergency Service

The recognition process is carried out by cloud services, with Infrastructure as a Service (IAAS) seamlessly integrated into the cloud environment. This IAAS framework furnishes the essential hardware components, encompassing storage, data processing capabilities, and a robust communication network infrastructure. Operating within this framework, Windows Server and Internet Information Services (IIS) serve as the foundational operating system and web server, respectively. The following services are deployed on this web server.

- Video processing service.
- Location service.

a) Video processing service

It extracts multiple frames from the input video and processes them using two algorithms. As all the emergency vehicles have a siren fixed on the roof. The siren could be of red or blue color or a combination of both colors. It periodically emits light. The first algorithm processes the video frames and extracts the area with red and blue light intensity. It also detects the repetition of this pattern in multiple frames. After completion of the processing, it provides a threshold value for detecting an emergency vehicle. The second algorithm converts the frames into grayscale images. It draws an object using the edge detection technique where the intense red and blue light gets detected. The detected object is compared with the patron of emergency vehicles saved already. It provides the type of emergency service as a result. This emergency service could be an ambulance, a fire brigade vehicle, a police vehicle, or a non-detected vehicle. It also processes sound and determines the presence of emergency services by using an efficient algorithm. This algorithm can detect the emergency vehicle's siren with the help of the frequency and pitch of the sound. Emergency vehicle siren has a defined range of frequencies. It lies between 392 Hz and 660 Hz. The siren produces different sounds for different frequencies in this range. It produces sound with these frequencies periodically. The algorithm detects an emergency vehicle by comparing the pitch of the sound and the repetition of this patron. As a result, it returns a threshold value to specify the percentage detection of emergency vehicle siren sounds.

b) Location service

This service uses Google map services to show emergency service locations on the map. After detecting an emergency service by the sound and image processing service, the location service gives the detected location. Then, it updates the Google map with the given location to intimate the nearest hospitals. A Website will be deployed

with this location service for the hospitals to view the emergency service location on the map.

3) Execution of Emergency Service

When the detection process is complete, the cloud services inform the sensor controller agent whether the event was accepted or rejected. If an emergency is found, the sensor controller agent notifies the SCa, who subsequently changes the signal for the emergency vehicle to green. The communication agent subsequently tells the closest signal controller nodes of the emergency vehicle's arrival, allowing easy traffic flow and the vehicle's safe passage.

4) Working of the Algorithm

The proposed architecture includes three different types of emergency vehicles, each with a unique priority based on their criticality level. Priority is given to the ambulance, then the fire brigade, and lastly, the police vehicle. If two vehicles of the same priority arrive simultaneously, the system prioritizes the one detected first. If the system cannot recognize the type of emergency vehicle, it gives it the ambulance's highest priority. Upon detection of an emergency, all signals, except for the emergency vehicle's path signal, are obstructed. In the event of multiple vehicle detection, two scenarios arise. In the normal case, the vehicles have diverse priorities on different signals. The agent will open that signal first, whose priority is high. In the rare case, vehicles have the same priority on different signals. In this case, the agent will use the first-come-first-serve approach.

5) Multi-Agents Communication Mechanism

The communication between the agents in our framework will be FIPA ACL based. In addition, the agents use multiple performatives for information sharing. Agent communication at different levels is described below.

a) Communication and sensor controller agent

The communication agent receives the video and audio data from the sensor controller agent for processing. The 'inform' FIPA performative is used to initiate communication, and then the 'inform if' performative is used by the communication agent to convey the processed data as well as a rejection or detection reaction. If the emergency service is identified, the 'inform if' performative provides signal control instructions to the SCa and informs the sensor controller agent to stop collecting data.

b) Signal nodes communication

The 'request' FIPA performative is used by a signal node to inform other surrounding nodes of the arrival of an emergency service vehicle when it leaves the node.

5. PROPOSED MAS ARCHITECTURE-BASED APPLICATION

To give a broader perspective of MAS architecture-based applications, some use cases are discussed in this section. Fig 4 presents a representation of the system's internal operation. The CB_SEN_Ag records video data and sends it to Google Cloud via an agent communication channel using the UDP protocol. In the cloud, it is received by the Zscalar firewall, which is deployed for security purposes. It prevents the system from hacking attacks. We are working with bulk data processing, which could slow down the system.

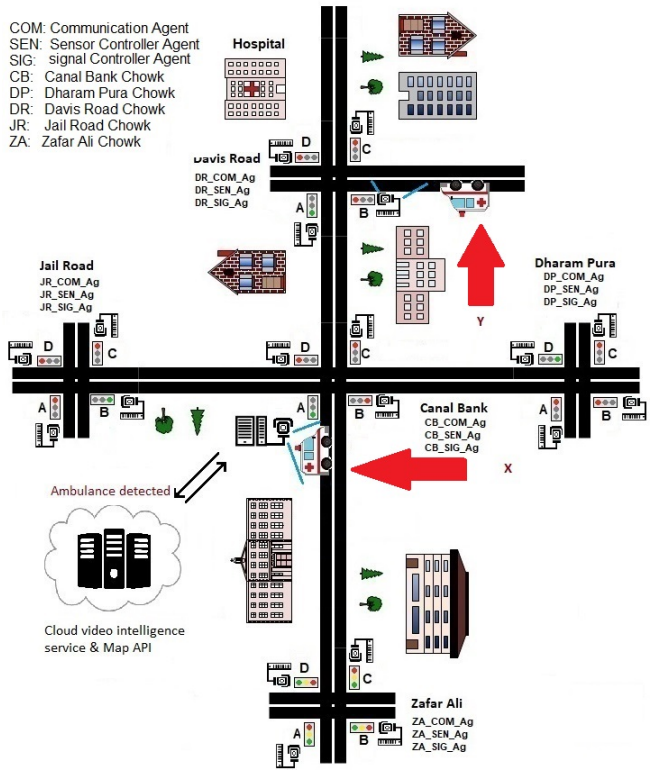


Figure 4. Ambulance X is detected at the canal bank, and ambulance Y is approaching at the Davis Road Intersection

To overcome this problem, we propose the Rackspace load balancer. It can handle 20 G.B. per second network throughput. It receives all the requests and transfers them to multiple servers for further processing. The Cloud Video Intelligence Service is used by the Google Cloud platform to analyze video data. This service analyzes every frame of the video and identifies the likelihood of relevant content occurrence. The presence/detection of an emergency vehicle is then notified to the sensor controller agent through the communication agent. It is assumed that the agent is able to retrieve the nearest destination, like the hospital, in the case of an ambulance using an API like Google Map. The sensor controller agent at Canal bank road send information to the SCa ($CANAL_BANK_AG$) about the emergency service by using the FIPA's 'inform' performative. After that, the sensor controller agent will then use the devised strategy and inform the agent deployed at the other nearest signal. The demonstrated scenario is given below:

1) Depicting Simultaneously Two Ambulances

Tab. 1 shows the emergency situation in which a signal A canal bank intersection detects the presence of an am-

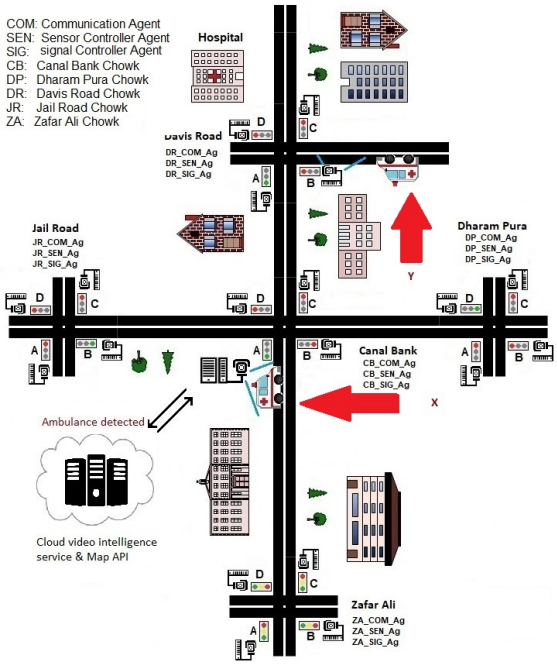


Figure 5. Ambulance X is detected at the canal bank, and ambulance Y is approaching at the Davis Road Intersection

bulance X. The corresponding signals at the nearby signal intersections open for the detected ambulance. Ambulance Y starts the emergency vehicle detection process as it moves toward signal B at the Davis Road junction at the same time. The recorded video is transferred by the sensor controller agent (DAVIS_ROAD_SEN_CON_AG) to the communication agent (DAVIS_ROAD_COMM_AG), which then sends the data to the cloud services for verification. The cloud services detect the ambulances in the given data and provide the information to the communication agent regarding the detection. After that, the sensor controller agent informs the SCa (DAVIS_ROAD_SIG_CON_AG) to block traffic for C, D, and A sides while opening traffic for the B site. The sensor controller agent also informs the SCa at Qartba intersection (QARTABA_INTER_AG), Canal Bank (CANAL_BANK_AG), and Katchery intersection (KATCH_INTER_AG) about the emergency service's arrival. Ambulance X passes through the Canal Bank intersection and heads towards the Davis road intersection. As shown in Fig. 3, Ambulance X passes through the Canal Bank intersection on its way to the Davis Road intersection. The ambulance finally arrives at the hospital after passing the Davis Road crossing. Eventually, the detection procedure is repeated when ambulance X approaches signal A at the Davis Road intersection, allowing the ambulance X to continue and arrive at the hospital, as shown in Fig. 5 and Fig. 6.

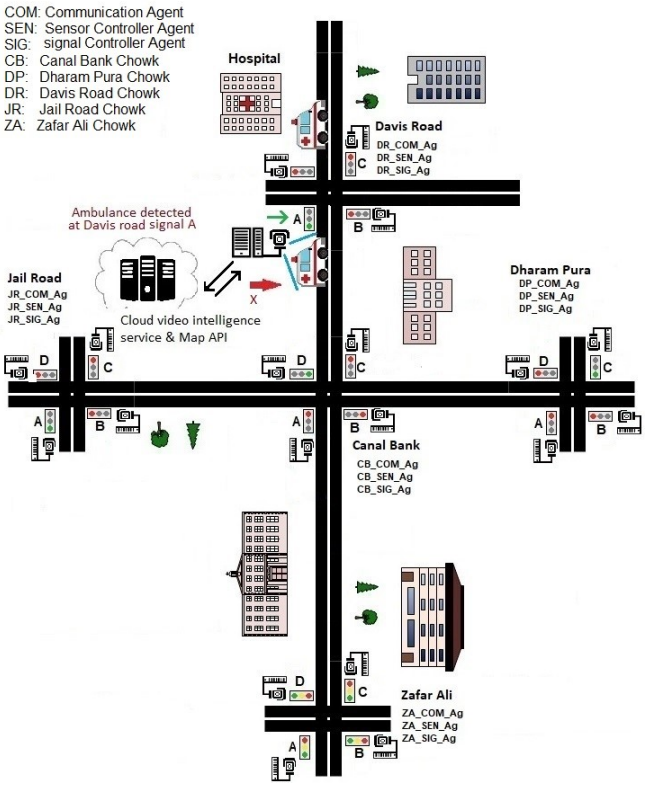


Figure 6. Ambulance X is detected at Davis Road Intersection, and ambulance Y is approaching at the hospital

2) FIPA Messages

Agent's communication in the form of FIPA messages is presented in Table I. The FIPA message communication protocol is widely used in multi-agent systems for exchanging information and coordinating activities among agents. Table I provides a comprehensive overview of the different types of FIPA messages used by the agents, including their purpose and format. By utilizing FIPA messages, agents can effectively communicate and collaborate with one another, ultimately leading to more efficient and effective decision-making in complex environments.

TABLE I. COMMUNICATION BETWEEN AGENTS IN THE GIVEN SCENARIO

Sr. No.	Performative	Request Agent	Response Agent	Information
1.	Inform	DAVIS_ROAD_SEN_CON_AG	DAVIS_ROAD_COMM_AG	Video and sound data for emergency service X
2.	Inform	DAVIS_ROAD_COMM_AG	DAVIS_ROAD_SEN_CON_AG	Cloud Service response with ambulance detected status
3.	Inform	DAVIS_ROAD_SEN_CON_AG	DAVIS_ROAD_SIG_CON_AG	Open signal for ambulance X
4.	Request	DAVIS_ROAD_SEN_CON_AG	QARTABA_INTER_AG	Signal Opening Request B>> for Ambulance
5.	Request	DAVIS_ROAD_SEN_CON_AG	KATCH_INTER_AG	Signal Opening Request A>> for Ambulance
6.	Request	DAVIS_ROAD_SEN_CON_AG	CANAL_BANK_AG	Signal Opening Request C >> for Ambulance
7.	Agree	QARTABA_INTER_AG	DAVIS_ROAD_SEN_CON_AG	Request Agreed
8.	Agree	KATCH_INTER_AG	DAVIS_ROAD_SEN_CON_AG	Request Agreed
9.	Agree	CANAL_BANK_AG	DAVIS_ROAD_SEN_CON_AG	Request Agreed
10.	Inform	DAVIS_ROAD_SEN_CON_AG	DAVIS_ROAD_COMM_AG	Video and sound data for emergency service Y
11.	Inform	DAVIS_ROAD_COMM_AG	DAVIS_ROAD_SEN_CON_AG	Cloud Service response with ambulance detected status
12.	Inform	DAVIS_ROAD_SEN_CON_AG	DAVIS_ROAD_SIG_CON_AG	Open signal for ambulance Y
13.	Request	DAVIS_ROAD_SEN_CON_AG	QARTABA_INTER_AG	Signal Opening Request B>> for Ambulance
14.	Request	DAVIS_ROAD_SEN_CON_AG	KATCH_INTER_AG	Signal Opening Request A >> for Ambulance
15.	Agree	QARTABA_INTER_AG	DAVIS_ROAD_SEN_CON_AG	Request Agreed
16.	Agree	KATCH_INTER_AG	DAVIS_ROAD_SEN_CON_AG	Request Agreed

TABLE II. FIPA MESSAGES

basicstyle= [language=] 1. Performative: Inform Request-Agent: $DAVIS_ROAD_SEN_CON_AGResponse - Agent : DAVIS_ROAD_COMM_AGOntology : Exigency - ServicesContent : (Action : VerifyEmergencyVerviceData : Streaming - video)$	[language=] 2. Performative: Inform Request-Agent: $DAVIS_ROAD_COMM_AGResponse - Agent : DAVIS_ROAD_SEN_CON_AGOntology : Exigency - ServicesContent : (Action : EmergencyServiceDetectedVType : AMB)$	[language=] 3. Performative: Inform Request-Agent: $(DAVIS_ROAD_SEN_CON_AG)Response - Agent : (DR_SlG_Ag)Ontology : Exigency - ServicesContent : (Action : Open - signalVType : AMB)$
[language=] 4. Performative: Request Request-Agent: $DAVIS_ROAD_SIG_CON_AGResponse - Agent : QARTABA_INTER_AGOntology : Exigency - ServicesContent : (Acton : Open - signalVType : AMB)$	[language=] 5. Performative: Request Request-Agent: $DAVIS_ROAD_SIG_CON_AGResponse - Agent : KATCH_INTER_AGOntology : Exigency-ServicesContent : (Acton : Open - signalVType : AMB)$	[language=] 6. Performative: Request Request-Agent: $DAVIS_ROAD_SIG_CON_AGResponse - Agent : KATCH_INTER_AGOntology : Exigency-ServicesContent : (Acton : Open - signalVType : AMB)$
[language=] 7. Performative: Agree Request-Agent: $KATCH_INTER_AGResponse - Agent : DAVIS_ROAD_SIG_CON_AGOntology : Exigency - ServicesContent : (VType : AMB)$	[language=] 8. Performative: Agree Request-Agent: $QARTABA_INTER_AGResponse - Agent : DAVIS_ROAD_SIG_CON_AGOntology : Exigency - ServicesContent : (VType : AMB)$	[language=] 9. Performative: Agree Request-Agent: $CANAL_BANK_AGResponse - Agent : DR_SlG_AgOntology : Exigency - ServicesContent : (VType : AMB)$
[language=] 10. (Performative: Inform Request-Agent: $DAVIS_ROAD_SEN_CON_AGResponse - Agent : DAVIS_ROAD_COMM_AGOntology : Exigency - ServicesContent : (Action : VerifyEmergencyServiceData : Streaming - video)$	[language=] 11. (Performative: Inform Request-Agent: $DAVIS_ROAD_SEN_CON_AGResponse - Agent : DAVIS_ROAD_COMM_AGOntology : Exigency - ServicesContent : (Action : VerifyEmergencyServiceData : Streaming - video)$	[language=] 12. Performative: Inform Request-Agent: $DAVIS_ROAD_COMM_AGResponse - Agent : DAVIS_ROAD_SEN_CON_AGOntology : Exigency - ServicesContent : (Action : VerifyEmergencyServiceVType : AMB)$
[language=] 13. Performative: Inform Request-Agent: $(DAVIS_ROAD_SEN_CON_AG)Response - Agent : (DAVIS_ROAD_SIG_CON_AG)Ontology : Exigency - ServicesContent : (ActonOpen - signalVType : AMB)$	[language=] 14. Performative: Request Request-Agent: $(DAVIS_ROAD_SIG_CON_AG)Response - Agent : (QARTABA_INTER_AG)Ontology : Exigency - ServicesContent : (Action : Open - signalVType : AMB)$	[language=] 15. Performative: Request Request-Agent: $(DAVIS_ROAD_SIG_CON_AG)Response - Agent : (KATCH_INTER_AG)Ontology : Exigency - ServicesContent : (Acton : Open - signalVType : AMB)$
[language=] 16. Performative: Agree Request-Agent: $(KAC_SlG_Ag)Response - Agent : (DAVIS_ROAD_SIG_CON_AG)Ontology : Exigency - ServicesContent : (VType : AMB)$	[language=] 17. Performative: Agree Request-Agent: $(QARTABA_INTER_AG)Response - Agent : (DAVIS_ROAD_SIG_CON_AG)Ontology : Exigency - ServicesContent : (VType : AMB)$	

Overall Response Time Summary

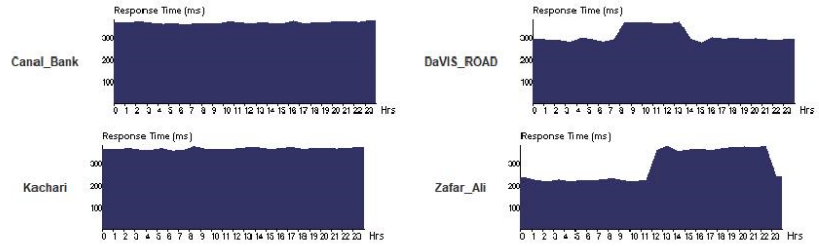
Total Response time	255.43 (min)	Total Requests Processed	43396.0
Total processing time	219.53 (min)		

	Average (ms)	Minimum (ms)	Maximum (ms)
Overall Response Time:	353.16	206.17	1080.19
Data Center Processing Time:	303.53	170.67	1021.68

Response Time By Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
Canal_Bank	374.485	211.921	1,080.188
DaVIS_ROAD	320.859	207.424	900.113
Kachari	373.281	212.173	1,079.437
Zafar_Ali	325.326	206.173	1,069.949

User Base Hourly Average Response Times



Data Center Request Servicing Times

Data Center	Avg (ms)	Min (ms)	Max (ms)
Pak_Cloud_Data_Center	303.527	170.668	1,021.683

Figure 7. Statistics for the cloud server

Overall Response Time Summary

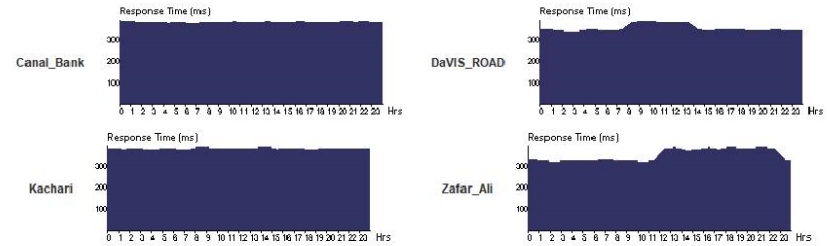
Total Response time	269.30 (min)	Total Requests Processed	43396.0
Total processing time	88.47 (min)		

	Average (ms)	Minimum (ms)	Maximum (ms)
Overall Response Time:	372.33	238.40	805.10
Data Center Processing Time:	122.32	56.89	504.10

Response Time By Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
Canal_Bank	382.325	258.78	738.599
DaVIS_ROAD	356.444	242.9	657.722
Kachari	380.979	246.402	805.104
Zafar_Ali	361.962	238.399	680.465

User Base Hourly Average Response Times



Data Center Request Servicing Times

Data Center	Avg (ms)	Min (ms)	Max (ms)
UK_Cloud_Data_Center	122.318	56.891	504.102

Figure 8. Statistics for the foreign cloud

6. DISCUSSION

In an automated traffic environment, it is very difficult to handle and regulate all services related to a critical situation like an emergency scene. Although it is tried to implement a system, which can detect ambulances based on their siren, however, its efficiency remains a question. These solutions are less effective and inefficient because of noise and daylight. To cope with such issues, we proposed a Multi-agent System for Emergency Services in Cloud Environment (MAS-ES-CE) solution.

In this system, the agent is responsible for the identification of data from a video. This video contains information about the external environment of vehicles. Then, the collected data is sent to the cloud services for further processing. The cloud services provide the most satisfactory solution for Bulk data processing. MAS decides the emergency service based on information returned by the cloud services. To show the utility of our work, we have implemented a simulation for the cloud environment. The tool used for simulation is the cloud analyst based on the Cloud Sim library developed at the GRIDS laboratory at the University of Melbourne. It is designed to analyze the behavior of large-scale applications in the cloud environment. In our simulation environment, we have four base stations (Canal Bank, Davis road, Katchery, Zafar Ali), a dedicated server (Pak data center), and two cloud servers (Pak cloud data center, U.K. cloud data center). The total simulation time is 24 hours. As response time is the most critical factor in client-server environments, we calculate response time for the given scenarios. In the given scenarios, we have two comparisons, dedicated server vs. local cloud environment and local cloud environment vs. foreign cloud environment, as shown in Fig.7. In an RTE, slow data communication is a major issue. There are two ways to approach this issue. First, communication speed could be increased by enhancing the communication protocols. In addition, data compression techniques can reduce the data's size, facilitating effective communication.

To provide a comparative advantage of our proposed scheme we assume that a single agent deployed on a signal can detect ten exigency vehicles per hour. This includes the vehicles detected using the image and video data. This amounts to a total of 240 vehicles in 24 hours. In a four-way intersection, we will deploy four agents so the total vehicles at max can be nine hundred and sixty. Considering we have a single cpu per agent and each vehicle recognition requires the execution of 1 million instructions then a single agent will need to execute a total of 240 million instructions in a single day. For 4 agents we need to execute a total of 960 million instructions in a day. It is to be noted that we have chosen instructions per second and not the cpu speed as it varies. If we refer to Fig 8 then we can see that Canal_Bank and Kachari has a consistent flow of traffic but Davis_Road and Zafar_Ali road does not. This means if we increase the local infrastructure then at some signals it might go wasted as it will not be fully utilized. Similarly, if we do not increase the processing capacity then at peak hours the number of undetected exigency vehicles will increase. Not to mention that the peak hour's traffic is subject to change at any time. This is where our proposed solution shows its effectiveness. Rather than deploying separate infrastructure for each agent, we can use any of the cloud service providers. This will provide us with the ability to increase and decrease the number of virtual machines as per need. Also, this option is cost effective as we pay for the number of instructions that will be executed. The scalability aspect of cloud is what makes it most suitable to be used in dynamic situations like real-time traffic monitoring or as in our case real-time detection of exigency vehicles.

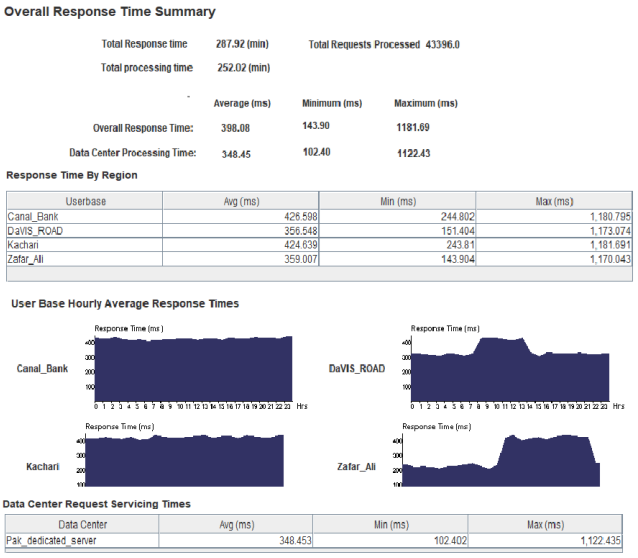


Figure 9. Statistics for the dedicated web server

7. CONCLUSION

In this research, we presented how Multi-agent technology and cloud services can manage emergency services in automated urban transportation. We have demonstrated how the automation of traffic signals can be achieved through the coordination of multiple agents using FIPA messages. The ARTIS architecture, which was developed to satisfy the RTE's criteria for traffic automation, is used to generate the agents. Infrastructure as a service is implemented in the cloud to process the massive amount of data sent by signal nodes. A case study was conducted on local transportation to demonstrate the proposed architecture's effectiveness. Cloud analysis tools were utilized to process the data and showcase the practicality of the suggested cloud services. These services were implemented on both dedicated and cloud servers.

Study is significant because it can potentially decrease delays for emergency services like the fire brigade and ambulance, which have been known to result in fatalities. Furthermore, we proposed a new direction in which com-

puting could efficiently enhance automated transportation systems' smartness and effectiveness.

References

[1] A. Byrski, R. Dreżewski, L. Siwik, and M. K. Dorohinicki, "Evolutionary multi-agent systems," *The Knowledge Engineering Review*, vol. 30, no. 2, pp. 171–186, 2015.

[2] A. Qasim, S. A. R. Kazmi, and I. Fakhir, "Formal specification and verification of real-time multi-agent systems using timed-arc petri nets," *Advances in Electrical and Computer Engineering*, vol. 15, no. 3, pp. 73–78, 2015.

[3] A. Qasim and S. A. R. Kazmi, "Mape-k interfaces for formal modeling of real-time self-adaptive multi-agent systems," *IEEE Access*, vol. 4, no. 1, pp. 4946–4958, 2016.

[4] M. Fiosins, B. Friedrich, J. Görmer, D. Mattfeld, J. P. Müller, and H. Tchouankem, "A multi-agent approach to modeling autonomic road transport support systems," in *Autonomic road transport support systems*. Birkhäuser, Cham, 2016, pp. 67–85.

[5] L. Li and D. Wen, "Parallel systems for traffic control: a rethinking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1179–1182, 2015.

[6] M. N. Ali and B.-S. Kim, "Intersection management system for autonomous vehicles using a fuzzy inference system," *IEIE Transactions on Smart Processing & Computing*, vol. 11, no. 3, pp. 199–212, 2022.

[7] D. Puthal, B. P. Sahoo, S. Mishra, and S. Swain, "Cloud computing features, issues, and challenges: a big picture," in *2015 International Conference on Computational Intelligence and Networks*, 2015, pp. 116–123.

[8] D. Talia, "Cloud computing and software agents: towards cloud intelligent services," in *WOA*, vol. 11, Rende, Italy, 2011, pp. 2–6.

[9] B.-Q. Cao, B. Li, and Q.-M. Xia, "A service-oriented qos-assured and multi-agent cloud computing architecture," in *IEEE International Conference on Cloud Computing*, Berlin, Heidelberg, 2009, pp. 644–649.

[10] A. Munawar, S. A. Raza, and A. Qasim, "Design and development of ai-based tourist facilitator and information agent," *Applied Computer Systems*, vol. 25, no. 2, pp. 124–133, 2020.

[11] K. H. Bui and J.-J. Jung, "Internet of agents framework for connected vehicles: a case study on distributed traffic control system," *Journal of Parallel and Distributed Computing*, vol. 116, no. 1, pp. 89–95, 2018.

[12] S. L. Bowman, C. Nowzari, and G. J. Pappas, "Coordination of multi-agent systems via asynchronous cloud communication," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 2215–2220.

[13] M. Chmielewska, M. Kotlarz, and J. Was, "Computer simulation of traffic flow based on cellular automata and multi-agent system," in *Parallel Processing and Applied Mathematics*. Springer, 2016, pp. 517–527.

[14] F. Wang, "Toward a revolution in transportation operations: ai for complex systems," *IEEE Intelligent Systems*, vol. 23, no. 6, pp. 8–13, 2008.

[15] ——, "Parallel control and management for intelligent transportation systems: concepts, architectures, and applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 630–638, 2010.

[16] B. Chen and H. Cheng, "A review of the applications of agent technology in traffic and transportation systems," *IEEE Transactions on intelligent transportation systems*, vol. 11, no. 2, pp. 485–497, 2010.

[17] M. Benalla, B. Achchab, and H. Hrimech, "A distributed intelligent system for emergency convoy," *International Journal of Interactive Multimedia & Artificial Intelligence*, vol. 4, no. 1, pp. 42–45, 2016.

[18] M. A. Mahmoud and M. S. Ahmad, "A self-adaptive customer-oriented framework for intelligent strategic marketing: a multi-agent system approach to website development for learning institutions," in *2015 International Symposium on Agents, Multi-Agent Systems and Robotics (ISAMSR)*, Putrajaya, Malaysia, 2015, pp. 1–5.

[19] Y. Shao, Y. Wu, and Y. Chen, "Design and research of multi-agent control system for central cooling system," in *2014 IEEE 3rd International Conference on Cloud Computing and Intelligence Systems*, Shenzhen, 2014, pp. 218–221.

[20] V. V. Panteleev, V. A. Kamaev, A. V. Kizim, and A. V. Matokhina, "Using of multi-agent system to model a process of maintenance service and repair of equipment of a service company," in *2016 IEEE 10th International Conference on Application of Information and Communication Technologies (AICT)*, Baku, Azerbaijan, 2016, pp. 1–3.

[21] M. Kermani and Z. Boufaida, "A modeling of a multi-agent system for the protein synthesis," in *2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA)*, Marrakech, Morocco, 2015, pp. 1–7.

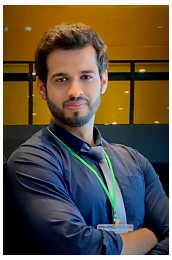
[22] F. El Hajj, A. El Hajj, and R. Chehade, "Multi-agent system vulnerability detector for a secured e-learning environment," in *2016 Sixth International Conference on Digital Information Processing and Communications (ICDIPC)*, Beirut, Lebanon, 2016, pp. 113–118.

[23] V. Botti, C. Carrascosa, V. Julián, and J. Soler, "Modelling agents in hard real-time environments," in *European Workshop on Modelling Autonomous Agents in a Multi-Agent World*. Springer, 1999, pp. 63–76.

[24] P. D. O'Brien and R. C. Nicol, "Fipa—towards a standard for software agents," *BT Technology Journal*, vol. 16, no. 3, pp. 51–59, 1998.

[25] N. Dragoni and M. Gaspari, "Performative patterns for designing verifiable acls," in *International Workshop on Cooperative Information Agents*. Springer, 2006, pp. 375–387.

Awais Qasim received the M.S. degree in Computer Science from Lahore University of Management Sciences (LUMS), Lahore, Pakistan, in 2011. After that, he worked as a Software Engineer in Industry and developed a number of iPhone and Android applications. He is working as Assistant Professor in the Computer Science Department, Government College University. Currently, he is working as post-doc researcher at the School of Science, Engineering and Environment, University of Salford, U.K. His research interests include model checking, multi-agent systems, real-time systems, self-adaptive systems, and robotics for disaster resilience.



Adeel Munawar received the M.S. degree in computer science from Government College University (GCU), Lahore, Pakistan, in 2018. His research interests encompass multi-agent systems, Artificial Intelligence, real-time systems, machine learning, and Natural Language Processing. Presently, he serves as a senior lecturer in Computer Science at Lahore Garrison University, actively engaged in research activities to promote research advancements. Additionally, he is currently pursuing a Ph.D. degree from Sirindhorn International Institute of Technology, Thammasat University, Thailand.

Mongkut Piantanakulchai received M.Eng. degree in transportation from the Asian Institute of Technology (AIT), Thailand. He received a Ph.D. degree in transportation from Tohoku University, Japan. Currently, he is working as an associate professor at Sirindhorn International Institute of Technology (SIIT). His research interest includes Intelligent transportation systems (ITS), multi-criteria decision-making in transportation planning, activity-based travel demand modeling and fuzzy decision making methods.