Studies in Infrastructure and Control

Manish Chaturvedi
Pankesh Patel
Ramnarayan Yadav *Editors*

Recent Advancements in ICT Infrastructure and Applications

Springer

Studies in Infrastructure and Control

The book series aims to publish top-quality state-of-the-art textbooks, research monographs, edited volumes and selected conference proceedings related to infrastructure, innovation, control, and related fields. Additionally, established and emerging applications related to applied areas like smart cities, internet of things, machine learning, artificial intelligence, etc., are developed and utilized in an effort to demonstrate recent innovations in infrastructure and the possible implications of control theory therein. The study also includes areas like transportation infrastructure, building infrastructure management and seismic vibration control, and also spans a gamut of areas from renewable energy infrastructure like solar parks, wind farms, biomass power plants and related technologies, to the associated policies and related innovations and control methodologies involved.

More information about this series at https://link.springer.com/bookseries/16625

Manish Chaturvedi · Pankesh Patel ·
Ramnarayan Yadav
Editors

Recent Advancements in ICT Infrastructure and Applications

Springer

Editors
Manish Chaturvedi
Institute of Infrastructure Technology
Research and Management
Ahmedabad, India

Pankesh Patel
AI Institute
University of South Carolina
Columbia, SC, USA

Ramnarayan Yadav
Institute of Infrastructure Technology
Research and Management
Ahmedabad, India

ISSN 2730-6453 ISSN 2730-6461 (electronic)
Studies in Infrastructure and Control
ISBN 978-981-19-2373-9 ISBN 978-981-19-2374-6 (eBook)
https://doi.org/10.1007/978-981-19-2374-6

I dedicate this book to my Parents Geeta and Shivshankar Chaturvedi, my Wife Renu, and Wonderful kids Chahek and Vyom
—Manish Chaturvedi

I dedicate this book to my DA-IICT friends—Brijesh, Dampu (Harshil), Kavan, Manish, and Rahul
—Pankesh Patel

I dedicate this book to all the contributors in the field from whom I learnt
—Ramnarayan Yadav

Preface

This book is one of its own kind which covers the complete spectrum of the ICT infrastructure elements required to design, develop, and deploy the ICT applications at a large scale. Considering the focus of governments worldwide to develop smart cities with zero environmental footprint, the book is timely and enlightens the way forward to achieve the goal by addressing the technological aspects. In particular, the book provides an in-depth discussion of the sensing infrastructure, communication protocols, cybersecurity, computation frameworks, software frameworks, and data analytics. The book also presents the ICT application-related case studies in the domain of transportation, healthcare, and energy, to name a few. The book can be used as a handbook for the design, development, and large-scale deployment of ICT applications by practitioners, professionals, government officials, and engineering students.

An Internet of Things (IoT) device/sensor node consists of a microcontroller, a variety of interfaces to connect sensors or actuators, wireless transceivers, power source, etc. For the sensing of physical phenomena (e.g., velocity, acceleration, position, temperature, pressure, mass, conductivity, etc.), the sensors and data loggers are the essential components of an ICT system. A large number of microcontroller platforms such as Arduino, Raspberry PI, FPGA boards, and mobile phones are available. The book presents a detailed survey on these microcontroller and sensor platforms.

Communication is an integral part of an ICT application. Based on the application needs and the role of a particular device, it may be required to communicate with the nearby or remote devices through the wired or wireless medium. The efficient communication protocols play an important role to achieve the effective communication among devices. The book consists of the discussions on the representative communication protocols such as the RFID, Bluetooth Low Energy, Zigbee, WiFi, 5G cellular, 6LowPAN, CoAP, and MQTT.

The security infrastructure is an integral part of the ICT applications. The connected system is required to be secure. In this book, various issues related to securing the ICT infrastructure (e.g., network, servers, cloud, and devices) are discussed and the security solutions used in practice are highlighted.

With the recent advancements in connected devices and communication infrastructure, advanced computing platforms such as edge, fog, and cloud computing are becoming popular. The IoT devices are resource constrained in terms of energy, computation, and storage. They cannot process and store the huge amount of data generated locally. Currently, these limitations of IoT devices are addressed by offloading processing tasks and storage to the distributed computing paradigms such as the cloud. However, the communication of a large amount of data to the cloud results in increased bandwidth requirement, increased latency, and high energy consumption on the IoT devices. To address these issues up to a certain extent, the more advanced computing frameworks involving the edge and fog devices have been proposed. In this paradigm, the one or more layers of computing resources are made available near the data generating devices. In this book, we present a rigorous discussion on the edge-fog-cloud computing frameworks and the distributed storage systems, along with the design of a few case study applications.

Industry 4.0 refers to the 4th Industrial revolution—the recent trend of automation and data exchange in the manufacturing industry. It involves the use of technologies such as IoT devices, cloud computing, and machine learning-based data processing to improve all the aspects of the Manufacturing Executing System (MES). The devices generate a huge amount of fine-grained data which can be easily communicated, stored, and processed using the ICT infrastructure. The availability of a large amount of data with the advanced computation resources permits the use of sophisticated machine learning algorithms to support different stakeholders of a factory (e.g., factory planners and managers) in decision-making. The book presents the design of machine learning-based data-driven models that can integrate multiple sources of information and analyze them on the fly to generate the real-time data analytics.

In addition, this book incorporates the key aspects of system and software engineering and smart analytics for ICT applications, in the context of IoT, Web of Things (WoT), and Industry 4.0 domains. The book presents an early design of a novel framework that combines IoT, Semantic Web, and Big Data concepts. The book not only presents the core components to build an IoT system, but also lists existing alternatives and open standards with their merits. The potential challenges in building future IoT applications are also highlighted. The WoT is rapidly growing in popularity among all the stakeholders. The ease of building the end-to-end solutions is the key for the WoT to succeed. This book reviews the main WoT challenges, particularly highlighting the ones related to combining the heterogeneous IoT data for designing the smarter services and applications that benefit from data interoperability. The book discusses the design of Machine-to-Machine Measurement (M3) semantic engine to semantically annotate WoT data, build the logic of smarter services, and deduce meaningful knowledge by linking it to the external knowledge graphs available on the web. M3 assists application and business developers in designing interoperable Semantic Web of Things applications.

To summarize, the book presents a whole spectrum of technologies required to design, develop, and deploy the ICT applications addressing diverse needs. In the following, we present a broad outline of chapters:

Chapter 1 broadly defines the theme of the book. The frameworks and architectural components of the ICT applications are briefly described. Various kinds of wired and wireless communication technologies and the communication protocols addressing the issues related to security and reliable data transfer are presented. The infrastructure requirements are determined and the interactions among various infrastructure elements are described.

Chapters 2 and 3 discuss various aspects of sensing infrastructure and use cases. In Chap. 2, the readers are introduced to the sensing infrastructure which is the cornerstone of current ICT applications. This chapter provides a big picture of the area, presenting the basic concepts of sensing and embedded systems, the aspects of sensors technologies and features, and the data loggers. Chapter 3 discusses the use of Micro-Electro-Mechanical System (MEMS) sensors in ICT applications. The chapter investigates and analyzes the sources and origins of different MEMS sensor errors and classifies them into different categories. A machine learning-based case study on monitoring the human respiration activity using two MEMS sensors, a flow sensor and an accelerometer, is presented. The system is implemented on a Field Programmable Gate Array (FPGA) board, and various performance measurements have been carried out.

Chapters 4 and 5 focus on the computation frameworks. The smart IoT devices offload the complex large-scale data aggregation and processing, as well as the storage to different computing platforms such as edge, fog, and cloud. Chapter 4 elaborates the computing infrastructure for ICT applications. Chapter 5 presents a fog computing architecture for the large-scale deployment of smart city applications. The chapter focuses on the deployment of fog nodes and communication among them using the 5G cellular infrastructure for the data communication and processing. In these chapters, the different computing paradigms including their benefits and limitations are discussed. The challenges and future directions for research in this domain are also highlighted.

In Chap. 6, the storage elements and frameworks used in the ICT applications are studied. This chapter provides a systematic overview of distributed storage infrastructures and discusses the current state-of-the-art solutions for storage technologies using Big Data and cloud models. The chapter also investigates the foundations, tools, and open research challenges of storing data using distributed storage infrastructures.

Chapter 7 presents the design and implementation of a stream data processing system for object detection from the video data streams in real time. Generally, the camera devices do not have sufficient computation resources for the object detection. Thus, the original quality video data are sent to the remote servers for processing, which represents a significant amount of communication overhead. This chapter surveys the techniques to reduce the communication overhead and proposes the Progressive Quality Improvement (PQI) approach to further improve the performance.

Chapter 8 presents the necessary foundations for the machine learning-based ICT applications and presents the software architecture of such systems. The chapter highlights the limitations of using machine learning models as a black box in sensitive ICT applications (e.g., medical diagnosis) and advocates about the use of Explainable

Artificial Intelligence (XAI)-based models. Further, the chapter presents the XAI-based software architecture for the case studies on medical diagnosis and online Mart.

Chapters 9 and 10 focus on the security infrastructure. Chapter 9 provides a comprehensive survey on various modern security solutions across the overall spectrum of ICT applications. The modern security infrastructure is categorized into four categories which include network security, server security, cloud security, and device security infrastructures. The chapter discusses various primitives available in each of these categories in the state of the art and discusses their related security aspects. Chapter 10 discusses the security theft and privacy invasion-related issues in the biometric-based authentication systems. The methods for morphing the original biometric templates through non-invertible or irreversible transformation functions are discussed. This chapter presents an efficient template protection scheme for unimodal and multimodal biometric authentication systems. Further, a few case studies on biometric systems are presented that attain performance improvement and provide adequate security to protect original biometric data.

Finally, Chap. 11 presents a case study of ICT application in the healthcare domain. The chapter presents a cloud-based remote healthcare delivery system for remote areas and studies its impact on society. The chapter elaborates the design, development, and deployment of a modular, re-configurable, and user-friendly graphical interface-based healthcare application for use in the rural areas with none or minimal healthcare facilities. The chapter summarizes people's perceptions and views about the usability of the ICT-based healthcare system based on the surveys.

Ahmedabad, Gujarat, India — Manish Chaturvedi
Columbia, USA — Pankesh Patel
February 2022 — Ramnarayan Yadav

Contents

About the Editors

Dr. Manish Chaturvedi is currently an assistant professor of Computer Science at Institute of Infrastructure Technology Research and Management (IITRAM), Ahmedabad, Gujarat, India. Before joining IITRAM, he worked at Nirma University (2003–2016) and Pandit Deendayal Petroleum University (2017–2019) as an assistant professor of Computer Science. Manish has earned his Ph.D. degree in ICT from Dhirubhai Ambani Institute of ICT (DA-IICT), India. His research interests include the design of Intelligent Transportation Systems, Embedded Systems and IoT, Scalable protocol design for large distributed systems, and the application of ICT for solving problems of societal importance. He has academic and research interests in the domain of distributed data structures, peer to peer content sharing, and Blockchain framework. He has extensively worked on the ICT application in the domain of Intelligent Transportation Systems. His work has been published in the reputed journals and conferences such as IEEE Transactions on Intelligent Transportation Systems, Pervasive and Mobile Computing Elsevier, ITSC, ICDCN, VTC, COMSNETS to name a few. More information about him may be found at https://sites.google.com/site/manishacademics/.

Dr. Pankesh Patel is an applied research scientist and educator in the area of system and software engineering. He focuses on building software development methodologies and tools to easily develop applications in the cross section of software engineering, Internet of things, and industrial Internet of things. Currently, he is an associate professor at University of South Carolina, USA and a senior research scientist at the AI Institute, University of South Carolina, USA. He is also a Marie-Curie Fellow at SFI Confirm Center for Smart Manufacturing, Data Science Institute, NUI Galway, Ireland. Before joining this position, he was Senior Research Scientist at Fraunhofer USA—Center for Experimental Software Engineering (CESE) from August 2017 to January 2019. At Fraunhofer USA, his focus was on implementation of industrial Internet of things (IIoT) techniques and methodologies in commercial environments. He worked as Research Scientist in the Industrial Software System (ISS) group at ABB Corporate Research-India from 2014 to 2017. He obtained his Ph.D. in Computer Science from the University of Paris VI (UPMC), France, with

a “highest honors” (“Tres Honorable” in French) title. His Ph.D. was funded by the French National Institute for Research in Computer Science and Control (INRIA), Paris, France.

Dr. Ramnarayan Yadav is currently working as Assistant Professor in the area of Computer Science and Engineering at Institute of Infrastructure Technology Research and Management (IITRAM), Ahmedabad, Gujarat, India. Before joining IITRAM, He was working as Assistant Professor in the Department of Computer Science and Engineering at Indian Institute of Information Technology (IIIT), Dharwad, India (January 2019–December 2019). He received his Ph.D. in Computer Science and Engineering from Indian Institute of Technology (IIT) Patna, India in 2017. During his Ph.D. he worked on development of the opportunistic spectrum access algorithms for Cognitive Radio Networks. His research interests include algorithmic graph theory, approximation algorithms, distributed algorithms, spectrum assignments and connectivity problems in computer networks. Previously, He received M.Tech. in Information and Communications Technology (ICT) from Indian Institute of Technology (IIT) Jodhpur in 2013. He also worked as Postdoc with Prof. Andrzej Pelc at the University of Quebec (UQO), Canada (2018–2019).

Chapter 1
Introduction

Manish Chaturvedi, Ramnarayan Yadav, and Pankesh Patel

Abstract The ubiquitous availability of the Internet and the advancements in communication, computing, and storage technologies have enabled a large number of Information and Communication Technology (ICT) applications. In this chapter, a few use cases of ICT applications are presented and the common framework of ICT applications is proposed. The framework consists of the building blocks/architectural components that are found in the majority of ICT applications. In particular, the framework includes the sensors and controllers, communication protocols, computation models, data processing and aggregation (intelligence layer), and security protocols. We describe all these building blocks in detail along with the technologies/solutions used in practice.

1 Background

Industry 4.0 defines the technological advancements in the domain of Internet of Things (IoT), Machine Learning, and Cloud computing, and the applications enabled by using them. In recent years, we have witnessed a growing number of sensors embedded in the devices used in our day-to-day life. These devices can connect and communicate over the Internet leading to the phenomenon called the Internet of Things (IoT). The number of IoT devices has grown to more than 10 billion (a conservative number) in the year 2021.[1] The market is expected to grow at the rate of 10% or faster. The machine-to-machine connections have grown to approximately

[1] Iot-analytics.com, gartner.com, statista.com.

M. Chaturvedi (✉) · R. Yadav
Institute of Infrastructure Technology, Research And Management, Ahmedabad, India
e-mail: manishchaturvedi@iitram.ac.in

R. Yadav
e-mail: ramnarayan@iitram.ac.in

P. Patel
AI Institute, University of South Carolina, Columbia, USA
e-mail: ppankesh@mailbox.sc.edu

M. Chaturvedi et al. (eds.), *Recent Advancements in ICT Infrastructure and Applications*,
Studies in Infrastructure and Control, https://doi.org/10.1007/978-981-19-2374-6_1

50% of the total in the year 2021,[2] and more than two-thirds of the global population will have Internet connectivity by the year 2023.

The IoT devices generate detailed sensory data for phenomena. The data has high volume, high velocity (rate of data generation), and high variety, and qualifies as the Big Data. The real-time and historical data generated by the IoT devices can be processed by the machine learning algorithms to carry out intelligent operations such as decision-making, predictions, classification, and clustering. The machine learning algorithms are computationally complex. The IoT devices are generally resource-constrained and cannot store or process a large amount of data. Further, they do not have the global view of phenomena to compute useful information. The advancements in computing technology—increased computation power and the edge-fog-cloud computing frameworks—address these issues. The edge and fog computing permit the data processing near the devices, reducing the latency of communication and decision-making. Cloud computing provides a large amount of hardware and software resources for ICT applications.

Using these technologies, a large number of ICT applications have been designed. The applications such as home/industry automation, precision agriculture, virtual assistant and chat bot-based customer support, product recommendation systems, remote health monitoring and assistance, travel time prediction, and monitoring of traffic and pollution in the region have become increasingly popular. In this chapter, we study a few ICT applications and identify their architectural components to design a generic framework. These components are elaborated further along with the technologies or solutions used in practice.

2 ICT Applications

Information and Communication Technology (ICT) has enabled a large number of applications (Fig. 1). In this section, we present a few ICT applications, namely multi-modal traffic monitoring, smart grid, and health care. The aim is to identify the architectural components of the ICT applications.

2.1 Multi-modal Traffic Monitoring

Figure 2 shows the architecture of a multi-modal Intelligent Transportation System (ITS). In a multi-modal ITS, the traffic condition in a road network is monitored using a variety of explicitly deployed sensors (e.g., loop detectors and traffic cameras) and alternative sources (cellular network, GPS probes, etc.). These sensors form the Traffic Sensing layer of the ITS.

[2] Cisco Annual Internet Report 2020.

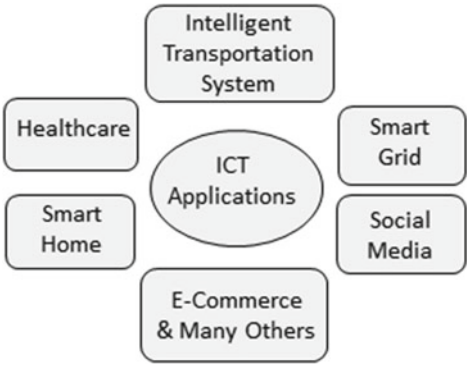


Fig. 1 ICT applications

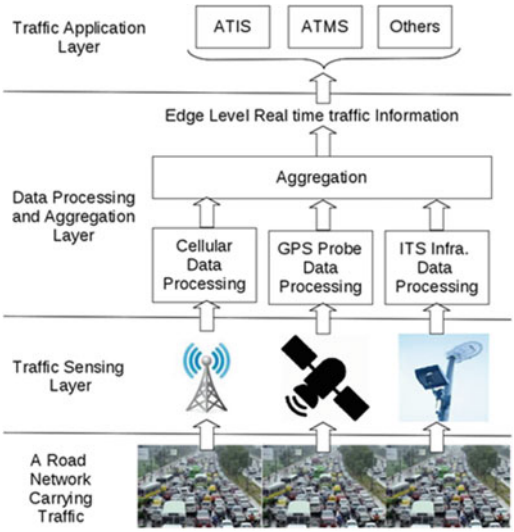


Fig. 2 Multi-modal intelligent transportation system

The raw data generated by a sensor go through sensor-specific processing. The dedicated sensors provide the fine-grained traffic information of the road segment where they are deployed, and require very limited processing. For example, the loop detectors readily provide the counts and classification of vehicles passing over them during the aggregation period. The videos/images of traffic captured by the traffic cameras are processed using sophisticated machine learning or deep learning algorithms to count/classify the vehicles. Alternative sources like cellular networks track the location of all their users for call forwarding. The location records of cellular users can be easily classified into two classes—mobile and stationary—using machine learning algorithms, e.g., Support Vector Machine, k-nearest neighbor, and decision tree. The mobile users' location records can be further processed using map-matching algorithms to compute movement trajectory of each user, which in turn can be used to compute traffic parameters of a road segment, e.g., vehicle flow, speed, and

congestion level. Similarly, the movement trajectories of Global Positioning System (GPS)-enabled vehicles can be processed to compute the traffic parameters. The aggregation process amalgamates the piecemeal data of traffic parameters collected from various sources to compute the traffic parameters for all the road segments in the region.

The spatio-temporal traffic information can be used by a variety of traffic applications. The Advanced Traveler Information System (ATIS) uses real-time traffic information and traffic forecasts to suggest trip plans to the querying commuters. The Advanced Traffic Management System (ATMS) uses real-time traffic information to enforce speed limits or diversions to reduce traffic congestion in the region. The Adaptive Traffic Signal Control uses the real-time/historical traffic information of the junctions in a local region in order to compute the cycle time and green-phase duration for every approach at a junction. The traffic information can be used by autonomous vehicles to plan their efficient and collision-free maneuver. The historical data on traffic conditions in the region can be used in the planning of infrastructure development projects, e.g., identifying junctions for the traffic signal deployment, building flyover bridges, etc.

It can be observed that the multi-modal ITS represents a classic ICT application consisting of sensors, communication hardware and protocols, the machine learning models for the data processing and aggregation, and the computation models for efficient and scalable processing of the data.

2.2 Smart Grid

Figure 3 shows the block diagram of a smart grid system. The system forecasts the electricity requirements at different times in a region using the inputs from the power grid, renewable energy sources, weather stations, and the smart meters deployed at the site of subscribers/consumers.

It is observed that the electricity consumption by a subscriber is different at different times of the day. Also, electricity consumption is affected by weather conditions. The smart meters provide fine-grained information about the energy consumption of every subscriber. The raw data of electricity usage are communicated and stored in a database. The energy production by renewable energy sources (solar, wind, etc.) is also affected by the weather condition. For example, solar panels generate more electricity during the afternoon of a hot summer day than during the nighttime or winter. The smart meters connected to renewable energy sources can provide fine-grained information about the energy generated during the different time-slots. Similarly, the weather station provides the weather forecast data on an hourly basis or with less granularity.

The aforementioned data can be communicated to a data center where it can be stored for further processing. The historical data can be processed using the centralized server or cloud-computing resources to understand the variations in the pattern of electricity generation/consumption, and the parameters (e.g., time of day,

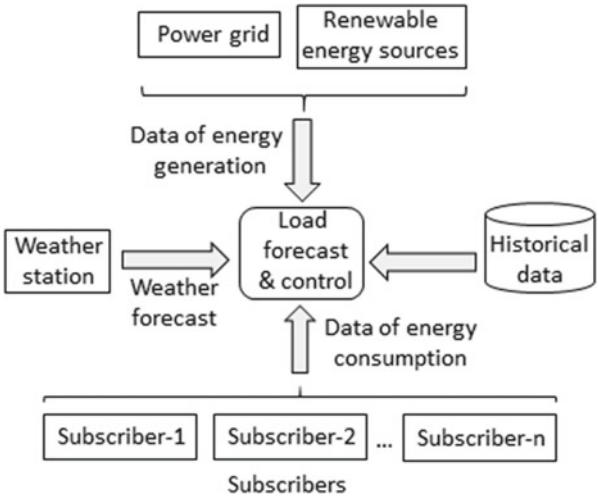


Fig. 3 Smart grid

weather conditions) causing the same. The data collected from various sources can be processed using the machine learning models to learn a mathematical model that can forecast the electricity consumption in the region and the electricity generated by renewable sources of energy. Such a forecast can be used to estimate the amount of electricity that needs to be generated using fossil fuel, and accordingly the production can be scheduled. The forecast can also be used to schedule/shift the load in order to spread the peaks/valleys of load on the electricity grid.

It can be observed that the smart grid uses sensing (smart meters and weather stations), communication and computation infrastructure, and machine learning models for improving the efficiency of electrical infrastructure.

2.3 Remote Health Monitoring System

Figure 4 shows the functioning of a remote health monitoring system. The sensing layer comprises the invasive/non-invasive/wearable devices consisting of a variety of sensors for monitoring physiological parameters (e.g., pulse rate, oxygen level, and blood pressure) of a patient. The devices permit continuous monitoring of the parameters and generate fine-grained data on a patient's health condition. The data can be locally processed and communicated using the edge computing and communication layer. Edge computing permits the processing of raw data of sensors at the local site. For example, the sensor data representing the normal condition of a patient can be aggregated using edge computing. This reduces the amount of data that gets communicated and processed by the central server. The data can be communicated locally

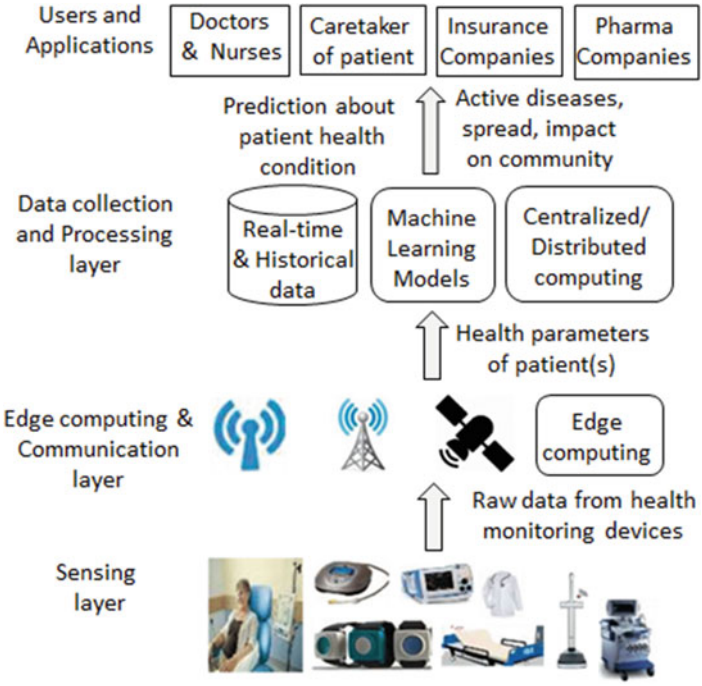


Fig. 4 Remote health monitoring system

or to a remote site using a variety of communication platforms. Zigbee, Bluetooth, or Wi-Fi can be used for local communication, e.g., from the sensor(s) to the local controller deployed in the same room/floor. The cellular infrastructure or Internet connectivity (e.g., broadband, satellite communication) can be used to communicate the data to a remote site.

The data collected at the data center can be processed in different ways. In its simple form, the physiological data can be directly provided to the doctor. The doctor makes decisions about the current health condition of the patient and suggests medication accordingly. Alternatively, the current and past data of health parameters can be processed at the data center using the time-series data analysis or machine learning models in order to understand the variations in the patient's health condition, and assist the healthcare personnel in the diagnosis. The processed information can be communicated to the doctor for his assessment and recommendation of medicines. The system can also generate the heath-status updates of a patient for perusal by the caretaker or family members.

The system permits the collection of fine-grained data of a large number of patients. The historical data can be processed to understand the overall health condition of a community, e.g., what kind of diseases are more prevailing and average life span of people. The insurance companies can use this information to

design/customize appropriate insurance plans for the community. The pharma companies can use this information to predict the requirement of certain medicines and plan their production accordingly. The research on drug discovery to treat certain medical conditions can also be guided using the information.

It can be observed that the remote health monitoring system uses sensing (invasive/non-invasive sensors), communication and computation infrastructure, and machine learning models to improve the overall healthcare echo system. In the following, we present a generic framework of ICT applications. The architectural components of the ICT applications are identified using the ICT application scenarios presented here.

3 Generic Framework of ICT Applications

Figure 5 shows a generic framework of the ICT applications. Based on the aforementioned ICT application scenarios, the following architectural components are identified that are required to design an ICT application:

1. Sensors and controllers.
2. Communication protocols.
3. Computation frameworks.
4. Intelligence layer.

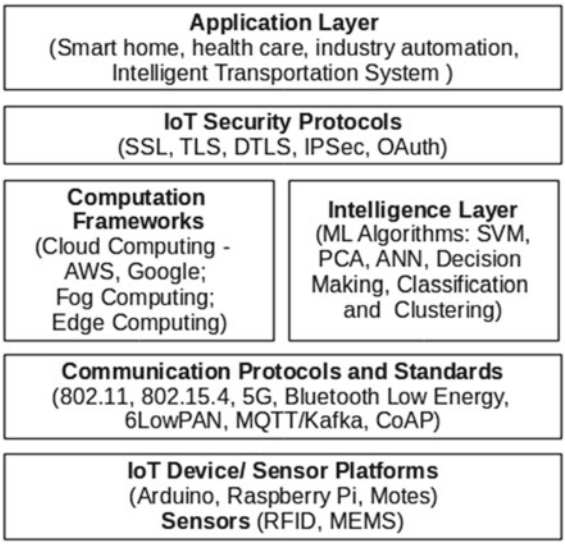


Fig. 5 A framework of ICT application

Table 1 Comparison of controller platforms

Features	Platforms			
	Arduino	Raspberry PI	Jetson nano	Smartphones
Processor	8-bit	64-bit Quad core	64-bit Quad core, 128 core GPU	64-bit Octa core with GPU support
Memory	KB	GB	GB	GB
Operating system	No	Yes (Raspbian)	Yes (Linux)	Yes (Android)
Multi tasking	No	Yes	Yes	Yes
Developer support	Arduino IDE	Python IDE (IDLE)	Jetson nano developer kit	Android studio
Application GUI	No	Yes	Yes	Yes
Programming language	C	Python	CUDA	Java
Communication	Additional hardware required	Wi-Fi	Ethernet, supports Bluetooth, Wi-Fi	Wi-Fi, Bluetooth, Cellular
Cost	Low	Moderate	Moderate+	High
Application	Hardware-based tasks with simple processing requirement	Hardware-based tasks with sophisticated processing requirements and local area wireless communication	Execution of computation-intensive image processing/deep learning models requiring local/wide area communication	Software/ Hardware-based tasks with sophisticated processing requirements and local/wide-area wireless communication

5. Application layer.
6. Security protocols.

In the following, all these components are described along with the solutions or technologies used in practice.

3.1 Sensors and Controllers

There are many microcontroller platforms available with varying capabilities and target applications. In this section, we briefly discuss the commonly available microcontroller platforms such as the Arduino, Raspberry Pi, NVIDIA Jetson, and Mobile phones (Table 1).

Arduino[3] is a simple microcontroller with limited memory, processing, interfacing, and communication capability. The sensors can be connected to the digital/analog input pins or the serial port of the controller. The programs for execution over Arduino

[3] https://www.arduino.cc.

are developed in C-like language using the Arduino IDE which also provides many library functions for accessing the data and status of the interfaced devices.

The Raspberry PI[4] is a more sophisticated microcontroller that provides the operating system interface (e.g., Raspbian), programming in a high-level language like Python, and wireless communication using the in-build Wi-Fi module. The controller has more memory and processing capability than Arduino. The controller has forty General Purpose Input Output (GPIO) pins permitting the interfacing of sensors or actuators. For video and sound communication, the controller has the High-Definition Multimedia Interface (HDMI).

The Jetson nano from NVIDIA[5] comes with 128-core GPU (Graphics processing unit) in addition to the 64-bit quad-core CPU, making it suitable for computationally intensive graphics data processing. The controller provides Internet connectivity using the Ethernet-based wired connection. The Bluetooth and Wi-Fi modules can be plugged in using the USB3.0 interface for short-range wireless communication. Considering the communication and the computation capabilities, the controller is best suited as an edge computing device requiring execution of the pre-trained deep learning models for the image/video data processing.

The smartphones come with a unique combination of processing capability (multicore processor), storage (a few GB of memory), wireless communication capability (Bluetooth, Wi-Fi for local communications, and the cellular transceivers for long-distance communications), and sensors (e.g., Global Positioning System (GPS), accelerometer, and gyroscope). An important method of data collection called crowdsourcing/participatory sensing has been enabled by the increased penetration of smartphone users. The collected data can be stored and processed locally up to a certain extent by the smartphone, and the processed information can be communicated in the local region (using Bluetooth, Wi-Fi) or over a long distance (using cellular infrastructure). These capabilities in a single box make it suitable for a variety of ICT applications, e.g., location-based services, user activity detection, and road traffic monitoring, to name a few.

3.2 Communication Infrastructure

The communication infrastructure consists of the hardware and software components. The hardware consists of the wired/wireless transceivers, network interface card, cables, connectors, interconnecting devices, and the end systems. The software consists of the protocols running at the different layers of hardware infrastructure. In this section, we discuss a few communication protocols used in the Internet of Things (IoT) systems.

[4] https://www.raspberrypi.org.

[5] https://developer.nvidia.com/embedded/jetson-nano.

3.2.1 Physical/Data Link Layer Protocols

In this section, we present a few popular physical/data link layer protocols such as RFID/NFC, 802.15.4 MAC, and Zigbee, Bluetooth Low Energy (BLE), and 5G cellular, which are enablers of many IoT applications.

Radio Frequency Identification (RFID) and Near Field Communication (NFC):

The system consists of an RFID reader and the RFID tag (active/passive). An RFID tag is designed to contain some information (e.g., Electronic Product Code or EPC of 64–96 bytes). When queried by an RFID reader, the tag responds with the stored information. The active RFID tags contain their own power source and have the communication range of up to a hundred meters. They can be used to track items/products in a warehouse. The passive RFID tags do not have their own power source. They use the electromagnetic energy transmitted by the RFID reader for communication, and support the communication range up to 25 m. The design of Near Field Communication (NFC) is motivated by the high-frequency RFID (13.56 MHZ). It permits bidirectional communication between the NFC-enabled devices. With its communication range of a few inches, the NFC is suitable for ticketing, secure payment, and data transfer.

802.15.4 MAC and Zigbee Protocol Stack:

The 802.15.4 standard defines a low-power physical layer and a Medium Access Control (MAC) protocol permitting the wireless communication at a lower data rate (250 kbps) among the resource-constrained devices. The Zigbee protocol stack uses the 802.15.4 protocol at the physical and MAC layers, and supports the star, pear-to-peer, or clustering-based topology (Fig. 6). The communication range of the Zigbee transceivers with the 1 mw of transmit power is approximately 30 m in the indoor environment and 100 m in the outdoor environment. The devices executing the Zigbee

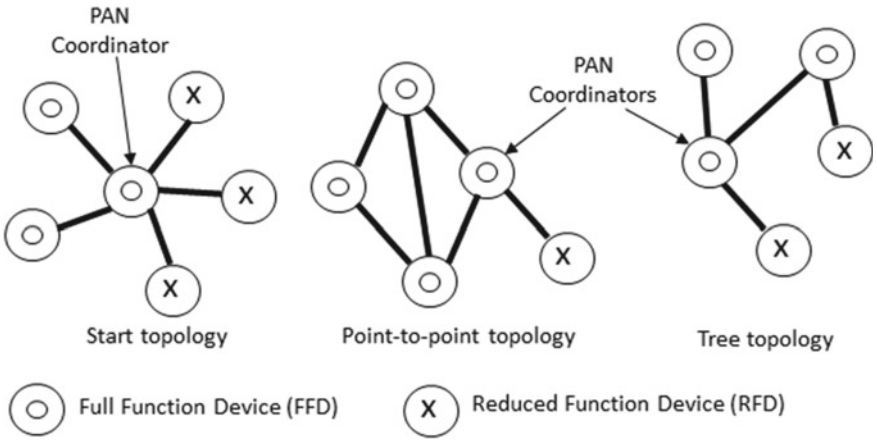


Fig. 6 Topology of Zigbee network

protocol stack are generally battery-driven low-power devices having limited storage, processing, and communication capabilities. Based on the application scenario, one or more sensors can be attached to the devices. The devices can form a Personal Area Network (PAN) in a small region to sense, process, and communicate the information.

The protocol stack supports two kinds of devices: Full Function Device (FFD) and Reduced Function Device (RFD). The RFDs have limited storage and processing capability. They can communicate only with the FFD acting as a PAN coordinator in the network, and can conduct limited functionalities of the 802.15.4 protocol. As the name suggests, the FFDs can do all the tasks defined in the 802.15.4 standard, and can take any role, e.g., the PAN coordinator or cluster head. They have more processing power and memory than RFDs.

The 802.15.4 supports the star topology, the point-to-point topology, and the cluster-tree topology of connection among the devices. In the star topology, an FFD takes the responsibility of PAN Coordinator. The other devices get associated with the PAN coordinator and communicate through it. Peer-to-peer communication can be carried out by a pair of FFDs. The FFDs can also act as cluster heads with which the other FFDs/RFDs in the communication range can associate to build the tree-like hierarchical topology.

Bluetooth Low Energy/Bluetooth Smart:

The Bluetooth Low Energy (BLE)/Bluetooth Smart has been developed by the smartphone makers and made available in most of the smartphones. The BLE uses the transmit power of 0.01–10 mw and provides a communication range up to 100 m over the 2.4 GHz frequency band. The devices can communicate at the data rate of 125–1 Mbps in the broadcast mode and the peer-to-peer mode using the protocol. In the broadcast mode, the broadcaster sends advertisement messages periodically. The observer devices in the communication range repeatedly check for the broadcast messages and receive them. In the peer-to-peer mode, the master-device coordinates the communication among peers. It scans the frequencies for the advertisement packets and initiates a connection. The connected devices exchange data periodically and sleep at other times to save energy.

5G Cellular:

The 5G cellular infrastructure is being deployed globally. It is a wireless broadband technology permitting communication over the frequency ranges < 6 GHz and >24 GHz (millimeter wave). The International Telecommunication Union (ITU) has issued the International Mobile Telecommunications (IMT)—2020 requirements for the 5G cellular networks. It is expected that the 5G cellular will provide the peak data rate of 10–20 Gbps, and the user-experienced data rate of 50–100 Mbps with a latency of 1 ms. The 5G network is expected to provide a significantly higher connection density (approximately 1 million devices in the km^2 area) compared to the 4G cellular network (approximately 2000 devices in the km^2 area). Further, the technology will provide seamless connectivity for the high mobility scenario when a user is moving at a speed up to 500 km/h. The 5G technology will provide a ten-fold

improvement in energy efficiency compared to the 4G cellular. Overall, the technology is futuristic and is expected to enable significant advancements in IoT and V2X communication-related applications.

3.2.2 Higher Layer Protocols

In this section, we present a few popular higher layer protocols such as the 6LoWPAN, MQTT, and CoAP, which are used in many IoT applications.

6LoWPAN:

The IPv6 over Low-power Wireless Personal Area Networks (6LoWPAN) allows the low-power devices running the 802.15.4 protocol to connect to the Internet. The Internet Protocol Version 6 (IPv6) uses 128-bit addresses to overcome the address scarcity-related issues that arose with the 32-bit IPv4 addresses. The RFC 49441 and 62822[6] present the specification of the 6LoWPAN protocol. The specification also describes how the header compression can be employed to support IPv6 over 802.15.4 MAC which has the frame size of 127 bytes only. The header compression is necessary due to the fact that the maximum size of the 802.15.4 MAC header is 25 bytes, AES Security overhead is 21 bytes, IPv6 header is 40 bytes, and the UDP header is 8 bytes size, leaving only 33 bytes for the application payload. If the larger size data is to be communicated, it requires segmentation at the source and reassembly at the destination. This issue of higher header overhead is addressed by the header compression mechanism of 6LoWPAN, which reduces the IPv6 header size to just 3 bytes and the UDP header to 4 bytes. During the communication, the MAC addresses are used for sending data in the 6LoWPAN network (IPv6 address of the source and destination can be derived from their MAC address). As mentioned earlier, the Reduced Function Devices (RFDs) associate with one of the Fully Functional Devices (FFDs), and communicate with the rest of the network through it. The FFDs maintain the routing table and take care of forwarding data packets toward the destination. The network also permits the LoWPAN broadcasts wherein the transmission is received by all the nodes in the network.

MQTT:

The Message Queuing-based Telemetry Transport (MQTT)[7] provides the communication between the publishers and subscribers through a broker (Fig. 7). The publishers publish the data with a specific topic to the broker. For example, a temperature sensor deployed in a room can publish its readings every minute with the topic "< sensorID:roomID >", where the < sensorID > is the unique identifier of the sensor, and the < roomID > is the unique identifier of the room. Similarly, a location tracker deployed in a vehicle can publish the periodic location updates with the topic < vehicleID:tripID >. A subscriber subscribes to the topics of interest and receives

[6] https://tools.ietf.org/html/rfc{4944, 62822}.

[7] mqtt.org.

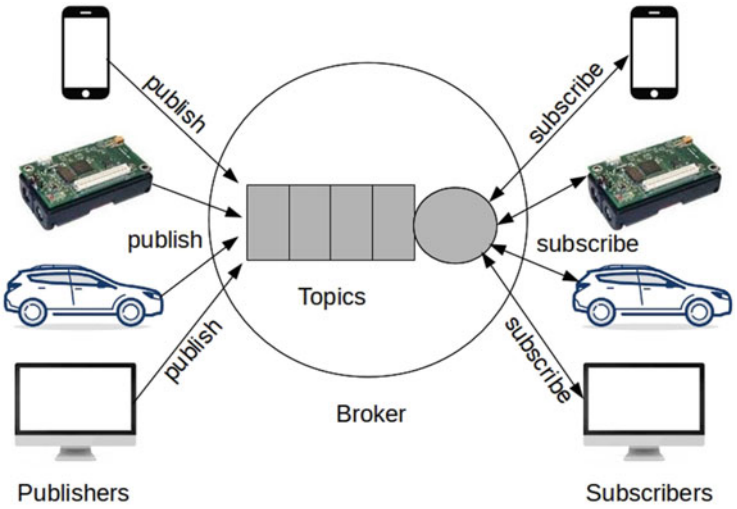


Fig. 7 MQTT-based communication

messages on the topic from the broker. For example, a smart-home controller may subscribe to the topic < sensorID:roomID > to receive the temperature data published by the sensor with the topic and activate the air conditioner to keep the room-temperature within the user-specified limits. The vehicle tracking application may subscribe to the topic < vehicleID:tripID > to receive the location updates published by the vehicle with the topic. Using the data, the tracker can determine the current location of the vehicle and also predict its arrival time at some en route location.

The presence of the broker in the system permits asynchronous communication between publishers and subscribers, i.e., the publisher and subscriber are not required to be online/connected at the same time for communicating with each other. The broker can store the published messages and make them accessible when the subscriber connects.

The MQTT protocol uses the Transmission Control Protocol (TCP) at the transport layer of the TCP/IP protocol stack and supports the three levels of Quality of Service (QoS): in QoS-0, the broker/client will deliver the message once, with no confirmation; in Qos-1, the broker/client will deliver the message at least once, and requires confirmation from the recipient; in QoS-2, the broker/client will deliver the message exactly once to the recipient using the four-step handshake. An application can select an appropriate QoS level as per the requirement.

MQTT-SN:

The MQTT-SN[8] is the publish-subscribe protocol for wireless sensor networks. It is designed to support the publisher-subscriber-based wireless communication over low-power devices, e.g., Zigbee-enabled devices. It can interface with the Application

[8] https://www.oasis-open.org/committees/download.php/66091/MQTT-SN_spec_v1.2.pdf.

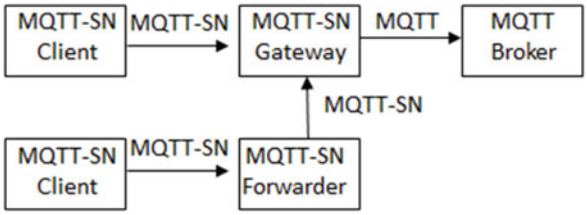


Fig. 8 MQTT-SN-based communication

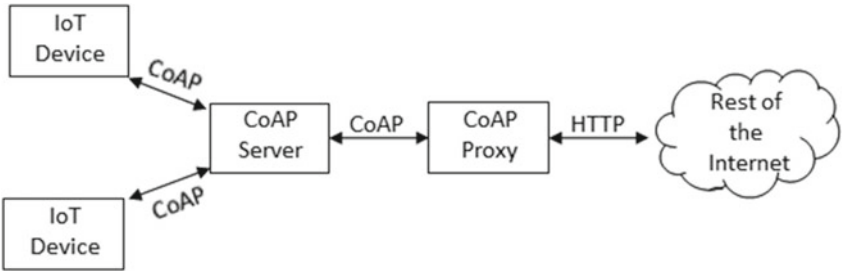


Fig. 9 CoAP-based communication

layer of the Zigbee protocol stack. Figure 8 shows a communication scenario using MQTT-SN. The MQTT-SN clients communicate with the MQTT broker through an MQTT-Forwarder or an MQTT-SN Gateway. An MQTT-SN gateway executes both, the MQTT-SN and MQTT protocols, and can communicate with the devices that execute either of these protocols. When an MQTT-SN client sends a message to the MQTT-SN gateway, it can extract message content and pack it into an MQTT message for sending it to the MQTT-broker. Similarly, the gateway can process an MQTT message received from the broker and send it to the MQTT-SN client by packing it into an MQTT-SN message. An MQTT-SN forwarder does not convert the received MQTT-SN message and just forwards it as received.

CoAP:

The Constrained Application Protocol (CoAP) is an application layer protocol for resource-constrained IoT devices. It uses the request-response model (like HTTP—Hyper Text Transfer Protocol in the World Wide Web) for communication between the IoT devices and the CoAP server (Fig. 9). The RFC 7252[9] contains the protocol specification. The CoAP runs on top of the User Datagram Protocol (UDP) at the transport layer, unlike the HTTP, which runs over the Transmission Control Protocol (TCP). UDP provides the basic service of process-to-process communication without addressing reliable communication or network congestion-related issues. The TCP is a more sophisticated connection-oriented transport layer protocol that provides reliable data transfer between the source and destination processes. It also provides

[9] https://tools.ietf.org/html/rfc7252.

congestion control in the network by monitoring the packet transfer delays and losses. It reduces the data transmission rate from the source in the event of congestion in the network. The CoAP implements the acknowledgment-based re-transmissions if the IoT application requires reliable communication.

The communications between an IoT device and the CoAP server happen using the RESTful APIs. In the REpresentational State Transfer (REST) standard, every resource is identified using the Uniform Resource Identifier (URI) just like the HTTP or other globally unique IDs. The contents are generally represented in the JSON format (key:value pairs). A CoAP proxy (possibly on the same machine running the CoAP server) is also part of the system that executes both—the CoAP and HTTP—protocols. The CoAP proxy permits the IoT devices to communicate with the rest of the Internet. It is responsible for an important task of protocol conversion. It receives the CoAP messages from the IoT devices and sends them out using the HTTP protocol; similarly, it receives the HTTP messages from the Internet and sends them to the IoT devices using the CoAP protocol. This arrangement permits the IoT devices to communicate with the rest of the Internet using the lightweight (low overhead) CoAP protocol. The CoAP proxy also caches the frequently accessed content in order to reduce the communication overhead and provide faster access.

3.3 Intelligence Layer

The IoT devices generate a large amount of data that can be processed in real time or aggregated for a longer duration. Due to the improvements in computation and communication speeds, data-driven machine learning algorithms are employed in a variety of ICT applications for predicting or classifying a scenario. In this section, we review a few machine learning algorithms for data processing. A machine learning system design involves the following steps: (1) data preprocessing, (2) model learning, and (3) model performance validation and testing. In the following, we briefly describe these steps.

Data Preprocessing:

In the data preprocessing step, we try to make the dataset usable for the subsequent steps. If there are n attributes and m data points, the dataset can be represented using an $m \times n$ matrix, where there are m rows, and each row contains a value for n attributes (n columns). It is possible that the dataset has missing values for certain attribute(s) in one or more data points. These missing values can be populated in a variety of ways without affecting the statistical properties of the attribute and the dataset. For example, one can apply the interpolation/extrapolation on the existing values of the attribute to populate the missing value. Alternatively, the mean of the existing values of the attribute can be computed and substituted for the missing value(s) of the attribute. After dealing with the missing values, we require normalization. The different attributes in a dataset may take the values from altogether different ranges of values. For example, in a dataset containing the attributes age and salary of a

person, the value of age attribute is in the range of 18–65 years, and the value of salary attribute is in the range of USD 10,000–100,000, for example. This affects the performance of a few machine learning algorithms (e.g., regression). Thus, all the attributes in the dataset can be normalized in order to map them into a similar range of values. We can do the min-max normalization, and compute the new value of the attribute as

$$x_{new}^{0-1} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{1}$$

where x is the current value of the attribute; x_{min} and x_{max} are the minimum and maximum values of the attribute, respectively. This simple method maps each attribute in the range of [0, 1] which can be further mapped to any arbitrary range by adding the min of the new range and multiplying by the size of a new range. For example, if we wish to map the attribute values in the range $[x_{low}, x_{high}]$, it can be done by multiplying x_{new}^{0-1} by $(x_{high} - x_{low})$ and adding x_{low} to the resultant.

$$x_{new} = x_{low} + x_{new}^{0-1} * (x_{low} - x_{high}) \tag{2}$$

This method of normalization is very sensitive to outliers. For example, if the value of an attribute in a particular data point is very high, it will result in $x_{new}^{0-1} \approx 0$ of the attribute for the majority of data points, as per Eq. 1. This issue is addressed by the *standard normalization*, wherein the new value of the attribute is computed as

$$x_{new} = \frac{x - \mu_x}{\sigma_x} \tag{3}$$

where μ_x and σ_x are the mean and standard deviation of attribute x, respectively, in the dataset. If the attribute values follow the Gaussian distribution, the normalized values will map in the range of $[-3, 3]$.

The outliers in a dataset represent some abnormal values for one or more attributes that deviate significantly from the *normal* values. For example, a temperature sensor deployed in a room reports the temperature in the range of 0–50° C under normal operation. At a certain time, if the sensor reports the room temperature of 200 ° C, it is generally due to an error. Such a data point should be identified and discarded. If the data is preprocessed using the standard normalization as described above, any data point having the attribute value outside the range of $[-3, 3]$ can be considered an outlier and discarded. The clustering algorithms can also be used to detect outliers. The clustering algorithms form clusters of *similar* data points. They use a variety of similarity or distance measures (e.g., Euclidean distance) to compute the similarity between a pair of data points. A data point that is not part of any cluster can be considered an outlier and discarded.

Model Learning:

In this step, the dataset is used to learn a mathematical model that defines the relationship among the attributes or the data points. The machine learning techniques can be

broadly categorized as supervised learning, unsupervised learning, semi-supervised learning, and re-enforcement learning. In the following, we briefly discuss these techniques.

The supervised learning algorithms use a dataset containing the value of input attributes and the associated output attribute for all the data points (labeled dataset). The supervised learning algorithms learn a mathematical model specifying the relationship between the input attributes and the output attribute. If the output attribute is continuous-valued (e.g., the salary of a person), the mathematical model is called a regression model. If the output attribute is discrete-valued (e.g., salary categorized as low, medium, high), the mathematical model is called a classification model. A large number of machine learning algorithms are available for regression (e.g., simple/multiple linear regression, support vector regression, and artificial neural network) and classification (e.g., logistic regression, decision tree, and random forest).

The unsupervised learning algorithms use a dataset containing the value of input attributes only (no output attribute). Such a dataset is called an unlabeled dataset. The aim of unsupervised learning algorithms is to learn the grouping/pattern among the data points. The clustering operation is an example of unsupervised learning which places similar data points in the same cluster (dissimilar or different data points in the separate clusters). As mentioned before, the clustering operation can also be used to identify and remove the outliers. Examples of unsupervised learning algorithms are k-means clustering, Self Organizing Maps (SOM), Density-Based Spatial Clustering of Applications with Noise (DBSCAN), etc.

The semi-supervised learning algorithms use a dataset where only a few data points are labeled (the value of output attribute is available) and the majority of them are unlabeled (the value of output variable is not known). The partial model—the supervised learning on the labeled data and the unsupervised learning on the unlabeled data—is used to determine the (pseudo) labels for the output variable of the unlabeled data points. The labeled data set (including the data points with pseudo labels) is processed using the semi-supervised algorithm to learn a mathematical model specifying the relationship between the input attributes and the output attribute. The examples of semi-supervised learning methods involve generative models, heuristic methods, etc.

Reinforcement learning represents an interactive environment for an agent where it learns using the trial-and-error method. The agent is permitted to take any action from a set of actions, and the penalty/reward is associated with every action possible in the given state. This permits the agent to learn the appropriate actions in a given state based on past experience. The reinforcement learning method is similar to the way human agents learn. These learning algorithms are suitable in robotics or autonomous driving scenarios wherein the complete knowledge of surrounding conditions is not available.

Model Performance Validation and Testing:

It is necessary to evaluate the performance of the designed machine learning model before deploying it in a production system. For evaluating the performance of a supervised learning model, initially the dataset is partitioned into two parts: the

training set and the test set. The relationship between the input attributes and the output attribute is learned using the data points in the training set. The performance of the trained model is evaluated using the data points in the test set. Here, the value of input attributes of every data point is provided as an input to the model, and the value of the output attribute is calculated. The calculated value is compared with the actual value of the output attribute (available in the test set) to determine the accuracy and other measures of the model.

Many a time, the k-fold cross-validation step is also introduced for the rigorous evaluation. Here, the dataset after excluding the test set is divided into k-strips, each containing an equal number of data points, and the k iterations of model training and performance evaluation are carried out as follows: in the first iteration, the first strip of data points is left out as a validation set, and the remaining strips are used for training the model. The model performance is evaluated using the validation set. In the ith iteration, the ith strip of data points is left out as a validation set, and the remaining strips are used for training the model. The performance measures obtained in all the k iterations are aggregated to compute the performance measures at the validation stage.

3.4 Computing Frameworks

Many times, the data generated by the IoT devices have high volume, high velocity (how fast the data are generated), and high variety, qualifying it for the big data. The sophisticated computation models are required to process a large amount of data using computationally expensive machine learning algorithms. In this section, we discuss a few aspects of computation models such as cloud computing, edge computing, and fog computing. We also describe their impact on the parameters such as computation speed, communication overhead and latency, and storage requirement.

3.4.1 Cloud Computing

The IoT devices are generally resource-constrained and are unable to carry out computation-intensive tasks such as learning a machine learning model. Further, they do not have the required storage capacity to store a huge amount of raw data. Even though an IoT device has required resources, in a setup where a large number of sensing devices are deployed to sense a phenomenon, a particular device does not have a global view of the scenario and may not be able to generate a correct response. Hence, it is necessary to offload the computation, storage, and other responsibilities to more sophisticated cloud services. Cloud computing provides the access to remote computing, storage, and other resources. The cloud services are broadly categorized as follows:

- Infrastructure as a Service (IaaS): the cloud service provider renders virtualization, servers, storage, and networking infrastructure to an IaaS user; the user has to take care of the other requirements such as the applications, middleware, operating system, and software development platform.
- Platform as a Service (PaaS): the cloud service provider renders operating systems, virtualization, servers, storage, networking, and the software development platform to a PaaS user; the user has to take care of the application development and management, and other requirements.
- Software as a Service (SaaS): the cloud service provider renders software access over the Internet; the applications, middleware, OSes, virtualization, servers, storage, and networking all are managed by the cloud service provider.

Now, we briefly describe the features of a few popular and commercially available IoT Cloud platforms such as the Amazon Web Services IoT,[10] the IBM Watson IoT Platform,[11] Google Cloud IoT,[12] and the Microsoft Azure IoT Hub.[13] They provide connectivity and network management (IaaS), application development platforms (PaaS), and device management, data acquisition, storage, processing, and visualization (SaaS).

Amazon Web Services IoT:

It was the first to enter the market (2004). The platform provides a real-time operating system called the Amazon FreeRTOS for microcontrollers and IoT devices. The FreeRTOS provides the ease of programming, deployment, connection, and management of the devices. It also provides AWS IoT Greengrass that permits local processing of data on the devices using the pre-trained machine learning models. It also permits the filtering of data at the device level and sends only the selected data to the cloud. Secure communication among devices and synchronization with the cloud are also provided.

The AWS device management consists of the AWS IoT Core, AWS IoT Device Defender, AWS IoT Device Management, and AWS IoT Things Graph. The AWS IoT Core lets connected devices easily and securely interact with cloud applications and other devices. The AWS IoT Device Defender monitors and audits IoT configurations to ensure security practices. The AWS IoT Device Management allows monitoring and remote management of IoT devices at scale. The AWS IoT Things Graph permits connection among different devices and cloud services to build IoT applications.

The AWS Data Services consist of AWS IoT Analytics, AWS IoT Events, and AWS IoT SiteWise. The AWS IoT Analytics computes sophisticated analytics on massive volumes of IoT data. The AWS IoT Events detects and responds to events from large numbers of IoT sensors and applications. The AWS IoT SiteWise collects, structures, and searches IoT data from the industrial facility databases and uses it to analyze the equipment and process performance.

[10] https://aws.amazon.com/iot/.

[11] https://www.ibm.com/cloud/watson-iot-platform/.

[12] https://cloud.google.com/solutions/iot/.

[13] https://docs.microsoft.com/en-us/azure/iot-fundamentals/.

IBM Watson IoT Platform:

It also provides a similar set of services as the AWS IoT. The platform provides a dashboard for device registration, connectivity, risk, and security management. The data can be stored for 30 days in Cloudant NoSQL DB, and archived for a longer duration on the Cloud Object Storage or the DB2 Warehouse cloud (Data Lake) of the platform. It also supports the data analytics on real-time or historical data to derive useful information from the raw data. The Watson IoT platform on Blockchain enables the devices to participate in blockchain business networks.

Google Cloud IoT:

The platform consists of Cloud IoT Core, Cloud Pub/Sub, Cloud Data Flow, Big Query, and Cloud ML Engine. The set of services provided by the Google Cloud IoT is similar to the other cloud platforms mentioned above. The Cloud IoT Core is responsible for the registration, authentication, and authorization of the IoT devices. It uses a secure MQTT broker for data communication with the devices, e.g., send/receive configuration/data to/from devices. The Cloud Pub/Sub retrieves data from devices and stores them as durable messages for further publish/subscribe operations and access by the other cloud services. The Cloud Data Flow provides batch data processing and stream data processing. The Big Query provides the data warehouse for storing a large amount of data. It uses the Structured Query Language (SQL) for data access. The Cloud ML Engine provides the data analytics and machine learning-based processing of the real-time and historical data using the TensorFlow open-source framework.

Microsoft Azure IoT Hub:

The platform consists of IoT Central, IoT Solution Accelerator, IoT Edge, Azure Digital Twins, and Time Series Insights. Here also, the set of services provided are more or less similar to the other cloud platforms mentioned above. The IoT Central is the application software that permits the connection, monitoring, and management of IoT devices. The IoT Solution Accelerator provides the application development platform for customizing the applications using Java, .Net (back end), or Javascript (front end). The IoT Hub allows connection from IoT devices to an IoT hub, and permits the monitoring and control of a large number of IoT devices. It is the underlying service for IoT Central and IoT Solution Accelerator. The IoT Edge permits the analysis of data on the IoT devices rather than in the cloud. This reduces the number of messages and the amount of data communicated to the cloud. The Azure Digital Twins enables the creation of comprehensive models of the physical environment in the software. This unique feature permits the evaluation of various design decisions and the performance of the system in the simulation/virtual environment. The Time Series Insights can store, visualize, and query the large amount of time-series data generated by the IoT devices.

Table 2 summarizes and compares the features of the IoT cloud platforms.

Table 2 Comparison of IoT cloud platforms

Features	IoT cloud platforms			
	Amazon Web Services IoT	IBM Watson	Google Cloud IoT	Microsoft Azure IoT Hub
IoT PaaS	AWS IoT Core	Watson IoT platform service	Cloud IoT Core	Azure IoT solution accelerator
IoT SaaS	–	Watson IoT platform dashboard and analytics	Android things console	Azure IoT central
SDK	AWS IoT Device SDK	C, Java, Node.js, Python SDK Libraries	Android Things SDK	Device/Service SDKs
Protocols	HTTPS, MQTT, WebSocket	HTTPS, MQTT, TLS	HTTPS, MQTT, gRPC	HTTPS, MQTT, WebSocket, AMQP, CoAP
Embedded OS	Amazon Free RTOS	Linux, OSX	Android things	Windows 10 IoT
Analytics	IoT analytics, IoT events, IoT SiteWise, AWS data services	Watson IoT platform analytics	Google big query	Azure time series insights
Market focus	E-commerce and finance	Health care	Finance, health care, customer service	Customer service
Pricing	Per million messages	Amount of data exchanged and analyzed	Per minute	Amount of data generated by devices

3.4.2 Edge and Fog Computing

The IoT devices are resource-constrained and have limited resources for computation, communication and storage. Considering the huge amount of raw data generated by the IoT devices, it may not be feasible to communicate all of it to the cloud server for two reasons: first, the communication of raw data to the cloud represents a significant amount of communication overhead, and second, the cloud server will be overburdened with the processing of a large amount of raw data most of which would be redundant. For example, the densely deployed sensors for fire detection at some sites will generate a large amount of raw data most of which will have similar readings under normal conditions (no fire).

Edge computing permits the basic processing of raw data locally on the IoT devices or on the first-hop controller. This can significantly reduce the amount of data that is communicated over the network to cloud services. In the above example scenario, the raw sensor data can be aggregated on the device, and the processed

data can be sent to the cloud. It should be noted that the devices do not have a global view of the situation. Thus, the data processing to detect the spread area of the fire or the direction of the spread still requires cloud services. All the aforementioned commercial cloud services support edge computing on local devices.

The cloud resources are relatively far from the site and require the Internet-based communication. The communication delay is more than that in the local area communication. Also, there is a monetary cost associated with accessing the Internet and cloud services. To address these issues up to a certain extent, we try to move the processing as near as possible to the site by introducing one or more layers of resources between the IoT devices and the cloud services. This phenomenon is called fog computing. The computation tasks that cannot be carried out locally are offloaded to the layer-1 fog devices. In general, the processing tasks of an application are offloaded to the devices in layer i if they cannot be handled by the devices at the lower layer(s). Only the tasks that cannot be handled by the edge/fog layers are sent to the cloud services. This computation framework reduces the amount of communication with the cloud and reduces the associated latency. A few fog layers can be part of the local area network, reducing Internet-based communication and the associated cost.

3.5 Security Protocols

The security protocols are an integral part of any connected system. A secure system is required to provide *confidentiality*, *integrity*, and *availability*. Data confidentiality ensures that private or confidential information is not accessible to unauthorized users. In a broader term, integrity ensures that the resources (data, software, hardware) in a connected system are not tampered with or altered by unauthorized users. The accessibility of the system to a legitimate user is very important. The availability ensures that the service is never denied to an authorized user.

Any security attack in a connected system tries to affect one or more of the three attributes. In a passive attack, an attacker may try to access the information being communicated or stored in the system. This violates confidentiality and also raises privacy concerns. In an active attack, an attacker not only accesses the information, but also attempts to modify it. This affects the confidentiality and integrity of the information and system. An attacker or a set of them may send a large number of requests to a server, making it unavailable for legitimate users. Also, the data may be made inaccessible to authorized users for ransom. These types of attacks affect the availability of systems or resources to genuine users.

To address these issues, a large number of security protocols and systems have been designed. Their aim is to reduce the occurrences and the impact of such attacks. To prevent unauthorized access to content or system, organizations use password protection, file system encryption, etc. A firewall deployed in an organization ensures that all the communication with the outside world passes through it, reducing the exposure of the local systems to the outside Internet and providing a layer of security. The secure-communication protocols such as Secure Socket Layer (SSL) and

Transport Layer Security (TLS) are used to provide the end-to-end encryption of messages communicated between the programs running on source and destination. The secure keys used by various encryption algorithms for secure communication are provided by the public key infrastructure (PKI).

The IoT devices are resource-constrained and the secure-communication protocols have their own overhead. Hence, it is necessary to design lightweight security protocols that can run on these devices without compromising the level of security. The 6LoWPAN (see Sect. 3.2.2) uses the 128-bit Advanced Encryption Standard (AES) protocol for securing the communication with low-power Zigbee devices. The MQTT uses TLS whereas the CoAP uses Datagram Transport Layer Security (DTLS) protocol for securing the application layer communication.

4 Conclusion

In this chapter, a generic framework for the ICT application design is proposed. The common building blocks required for designing an ICT application are identified using a few use cases, namely multi-modal ITS, smart grid, and remote health monitoring system. The framework consists of the IoT devices and sensor platforms for sensing the phenomena and doing some local processing, the communication protocols for sending/receiving information to/from various devices, the intelligence layer for data processing using machine learning algorithms or other methods, the computation frameworks for efficient distribution and scheduling of the data processing tasks, the security protocols to provide secure and tamper-proof access to the system resources, and the application layer defining the use case and business logic. All these building blocks are described along with the recent solutions used in practice.

Chapter 2
Sensing 101

Daniel J. de Carvalho, Victor W. C. de Medeiros, and Glauco E. Gonçalves

Abstract Today decision-making in all fields demands more and more environmental data, enabling estimates improvement, fine-control of manufacturing processes, and aggregation of more value to services. Scientific research currently also depends on accurate data for testing new hypotheses and discovering new phenomena. In other words, all these fields require a complete infrastructure for automatically sensing, storing, and transmitting data. State-of-the-art Information and Communications Technology (ICT) offers an array of options for environment sensing, from high-resolution sensors to low-cost alternatives. The main objective of this chapter is to introduce the reader to the field of physical world sensing through sensors and dataloggers, which are the cornerstones of current ICT applications. We provide a big picture of the area, presenting: basic concepts on sensing and embedded systems, aspects of sensors technologies and features, the wide range of interfaces that compose current dataloggers, and finally, we end the chapter discussing future directions in the field.

When studying physical phenomena, we must have the means to measure them. Physical quantities allow us to measure this by creating a relationship between the quantity and the observed phenomenon. Quantities typically present in sensing systems are time, velocity, acceleration, position, temperature, pressure, mass, conductivity, electrical current, electrical voltage, resistance, capacitance, frequency, and viscosity. Several physical units can represent a physical quantity. Temperature, a physical quantity, can be measured using the physical units Celsius (°C), Fahrenheit (°F), or

D. J. de Carvalho (✉) · V. W. C. de Medeiros (✉)
Departamento de Estatística e Informática, Universidade Federal Rural de Pernambuco, Rua Dom Manuel de Medeiros, s/n Dois Irmãos-CEP, 52171-900 Recife, PE, Brazil
e-mail: daniel.jcarvalho@ufrpe.br

V. W. C. de Medeiros
e-mail: victor.wanderley@ufrpe.br

G. E. Gonçalves
Instituto de Tecnologia, Universidade Federal do Pará, Rua Augusto Corrêa, 01, Guamá-CEP, 66075-110 Belém, PA, Brazil
e-mail: glaucogoncalves@ufpa.br

M. Chaturvedi et al. (eds.), *Recent Advancements in ICT Infrastructure and Applications*, Studies in Infrastructure and Control, https://doi.org/10.1007/978-981-19-2374-6_2

Kelvin (K), for example. It is also possible to derive a qualitative measurement from a quantitative one. Measuring the incidence of UV radiation, for example, enables determining the actual risk for human health from very low to very high.

Sensing physical phenomena is an essential task for all human activities. Manufacturing, services, management, and other fields employ more and more sensors to feed their processes and improve response time, product quality, and costs. Considering only a single environmental variable as relative humidity, one can see that humidity sensors have applications in several industries for reducing damage to products, environment monitoring, and maintaining employees' health. Such importance is translated in a growing market that it is expected, by 2025, to reach USD 1.6 billion [20] just for this kind of sensor. This way, wearable, environmental, and other embedded sensors can be seen today as valuable assets integrated into the information systems and the value chains. These devices that form the Internet of Things (IoT) are a centerpiece of the Industry 4.0 revolution [2, 24, 29].

Scientific research also depends on sensors for analyzing and understanding physical processes and natural phenomena. Day to day, researchers use specialized instruments to gather experimental data [22], which were impacted by the IoT with a whole set of new devices that can be integrated into information systems. Data from these experiments are sent through the world to many scientists in quasi-real-time and used on powerful simulation models [27].

The main objective of this chapter is to introduce the reader to the field of physical world sensing through sensor devices and dataloggers, which are the cornerstones of current ICT applications. We will provide a big picture of the field in six sections, which will respectively discuss: the fundamentals of sensing (Sect. 1), embedded systems (Sect. 2), sensors (Sect. 3), hardware interfaces (Sect. 4), dataloggers (Sect. 5), and further advances in the field (Sect. 6).

1 Sensing Fundamentals

Sensing consists of using devices and systems that translate the perception of a physical phenomenon that humans or machines can interpret. For example, if we want to measure the pressure exerted by a fluid in a hydraulic system, we can use a mechanical pressure gauge. This device has a diaphragm that, when subjected to a certain pressure, moves a plunger mechanically connected to a dial needle (Fig. 1). The values printed on the dial are pre-calibrated, ensuring that the value indicated by the needle under the dial is accurate to the pressure exerted by the fluid on the diaphragm. We say that devices like this are analog as they can have an infinite or continuous number of possible values. Reading from a device like this is promptly done by a human but not by computers.

As electronic devices, computers need that sensors, somehow convert the physical phenomenon to electrical signals. This conversion is not available on our mechanical gauge. In order for a computer to interpret the phenomenon, the sensor must generate electrical signals. So, how is it possible to generate these signals? There are numerous

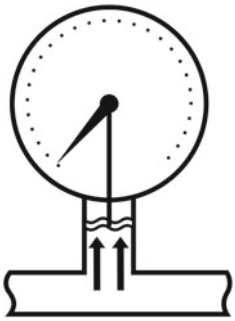


Fig. 1 Mechanical pressure gauge

ways to achieve this effect, but one must associate a variable resistance to the pressure gauge plunger. There will be a change in electrical resistance when moving the piston, implying a voltage variation over time, generating an analog signal. Although not exclusively associated with electrical signals, the term analog signal is widely used to describe them. We will discuss this sensor concept in detail in Sect. 3.

A digital computer cannot interpret the pressure gauge with proposed modifications because it still generates analog signals. Computers use the bit as an elementary unit to represent any information they need to store or process. A bit can have only two possible values, usually expressed as '0' and '1' but physically associated with the conduction or non-conduction states of the electric current through the transistors that make up processors.

All information processed and stored by computers and digital media, such as programs, data provided by a system user, and even data collected through sensors, must necessarily be represented through sequences of bits. On the other hand, our perception of the world as human beings is entirely analog. To consume information processed or stored in digital media, we need to convert them to the analog world. Thus, we conclude that for a computer to interpret the information generated by the manometer, there must be a way to convert the analog signal to a digital signal.

Let us consider another scenario. Bob, a musician, wants to record a song and make his work available on a music streaming platform. Assuming that he will record a song with only voice and guitar. His vocal folds and guitar strings' harmonic movement will generate disturbances in the air in a specific range of frequencies that we call sound waves. Our ears perceive these disturbances in the air and build our brain's perception of this physical phenomenon through biochemical and bioelectrical processes. During his performance, people close to Bob will hear the new music without the need for any other device. However, what he wants is his music to reach the ears of anyone who wants to know his work. How could we make this possible?

1.1 Transducers

We need a device capable of sensing the sound waves generated by Bob's performance for a start. This device is the microphone, technically defined as a transducer, capable of converting signals of one energetic nature to another. More specifically, a microphone is a transducer capable of converting sound waves into electrical signals. The loudspeaker and the photocells are also good examples of transducers, the first transforms electrical signals into sound waves, and the second transforms light into electrical energy. We will talk more about transducers in Sect. 3 when we discuss the working principle of some sensors.

The electrical signals generated by the microphone are still analog and thus cannot yet be processed and stored on a computer or digital media. So, how can we transform the electrical signal generated by the transducer into a digital signal?

1.2 Analog to Digital Converter—ADC

We need a device capable of converting the analog signals generated by the transducer (the microphone) into a bitstream representing this information for this task. This device is the analog-to-digital converter or ADC. In this material, we will not address the inner workings of an ADC or the technology employed in its construction. Instead, we will focus on two essential concepts in understanding the analog-to-digital conversion process: sampling and quantization.

Sampling is the process of capturing samples of the desired signal at constant time intervals. The sampling rate, usually measured in Hertz (Hz), is the frequency of these samples capture. The Nyquist–Shannon sampling theorem relates the captured signal's frequency and the sampling rate (Eq. 1). According to the theorem, the sampling rate f_s must be greater than twice the highest frequency component of the measured signal, the Nyquist frequency (f_N). In our example, we can consider that the highest frequency component we want to capture is the highest sound frequency that the human ear can perceive, something around 20 kHz. Applying the theorem directly, we have that the sampling rate of the ADC that will receive the microphone signals must be at least 40 kHz.

$$f_s > 2 \times f_N \tag{1}$$

The second important concept is quantization. This concept involves representing an analog (continuous) value through a few discrete values. The number of possible values used in representing the captured electrical signal is given by the number of bits used by the ADC. For this conversion to be possible, there is a natural rounding or truncation process of the original signal, implying a loss of information. Thus, the number of bits used impacts the quality of the discretized signal, and for this reason, it is called ADC resolution.

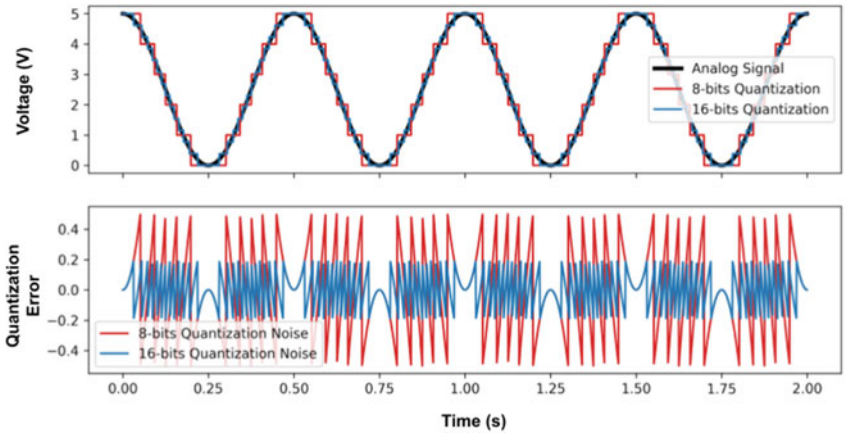


Fig. 2 Quantization and quantization noise

Let us imagine that Bob's microphone generates electrical signals ranging between 0 and 5V and that his ADC has an 8-bit resolution. With 8 bits, it is possible to represent 256 discrete values (2^8) from '0' to '255'. In this way, the smallest captured voltage value, 0V, will be represented by the integer decimal value '0'. The highest voltage value captured, 5V, will be represented by the value '255'. Applying Eq. 2 where, x_{max} and x_{min} represent the largest and smallest amplitude, respectively; and n the resolution of the ADC; we arrive at the quantization step represented by Δ. When capturing the analog data at the sampling instant, the ADC discretizes the value to the closest quantization step multiple. The difference between the analog value and the discretized digital value is called the quantization error. The variation of this error over time is called quantization noise. Note in Fig. 2 that the higher the resolution used in the ADC, the lower the quantization noise. In the example in Fig. 2, the RMS (Root Mean Square) value of the quantization error calculated for the 8- and 16-bit ADC were 0.26 and 0.11, respectively, showing us clearly the noise impact of quantization on the signal.

$$\Delta = \frac{(x_{max} - x_{min})}{2^n - 1} \tag{2}$$

It is important to emphasize that what will determine the adequate resolution of an ADC or the acceptable noise is its intended application. The vast majority of them already present satisfactory results using only 8-bit converters. Other applications, such as Bob's audio recording, require resolutions of 16 bits or more. High-fidelity sound systems (Hi-Fi), for example, commonly use ADCs with more than 16-bit resolution. These systems receive this name for being able to reproduce the original sound as faithfully as possible.

It is common to find ADC converters embedded in most microcontrollers today and with the most varied resolutions. The sampling rate is directly associated with

the operating frequency of the microcontroller used. If necessary, it is also possible to use external ADC chips. These usually have multiple channels, thus allowing the capture of multiple signals simultaneously.

1.3 Digital to Analog Converters—DAC

Now that Bob has digitized his music using the microphone and an ADC, he can digitally store and distribute it any way he likes. Since his song has been made available and accessible to Bob's fans on the internet, these fans need to make this digital file become sound in their ears. We already know that to appreciate Bob's music, it is necessary that, in some way, digital information becomes analog again. We need to do the reverse process Bob did for distributing his music in digital format. For this, we need a device called a digital-analog converter or DAC. This device can output different voltage levels over time (analog signal) by converting the digital input values. However, voltage levels are still not music. Now, we need a transducer capable of converting electrical energy into sound energy. This device is the loudspeaker. The loudspeaker consists of an electromagnet that moves a coil with varying electrical voltage. This coil is mechanically connected to the diaphragm. This thin membrane, when vibrating, generates disturbances in the air around it that will reach the ears of Bob's fans in the form of music. It is worth noting that there is usually an amplifier circuit between the DAC and the speaker responsible for raising the power of the electrical signal, allowing it to move the loudspeaker coils.

Some microcontrollers have built-in DACs, but they are less common than ADCs. Generally, projects use an external DAC integrated circuit. The DAC chip receives the digital values at a specific rate and converts them to different voltage levels. As with ADC, DAC chips have a particular resolution in bits. In our example, lower resolutions would result in a more flawed sound due to lower voltage level possibilities. Data input to the ADC can be done in parallel through n-bits equivalent to the chip resolution or through a serial interface where the n-bits are passed to the converter serially. Some serial ADCs use the I^2C interface, detailed in Sect. 4.5, as the digital data input interface. We will discuss the principles of serial and parallel communication in more detail in Sect. 4.1.

2 Embedded Systems

Embedded systems are specialized computing devices designed to perform a restricted set of activities. Unlike a conventional computer, which has easy access to power, memory, and processing power, embedded systems typically operate in resource-poor environments. They usually have modest processing power to save energy and prioritize more efficient memory use while responding quickly, some-

times in real-time. This type of system also needs to be fault-tolerant, as it is common to operate in hostile environments, such as tracking wild animals and even in outer space, on satellites.

The core of an embedded system is its **processing unit**, which can be either a microcontroller or a microprocessor. This component will control the other components and peripherals of the system, executing the functionalities defined in its firmware. Depending on the project, it may be more indicated to use a microprocessor or a microcontroller. To understand what is adequate, one must know the main differences between them.

Firmware is a software developed for a given hardware, providing a low-level control over it.

According to [13], a **microcontroller** is a small computer contained on a chip, having several built-in components, such as memory, CPU, timers, and I/O. It uses its internal memory to store the firmware that will run during the operation, and the instructions of this firmware limit its actions. Because it does not need an operating system, it tends to have a faster boot process. It has clock speeds in the tens of MHz and has a low power consumption. It is suitable for specific or repetitive tasks, such as controlling motors, turning on devices, and receiving data from sensors.

A **microprocessor** has only the integrated CPU, requiring external components for storing programs and running them, such as NAND Flash and DRAM memories, thus impacting boot time. Despite that, it has a more complex construction than the microcontroller. It needs an operating system, and it is made for general use. That is, it runs the most diverse programs created or installed by users. It features clock speeds at the GHz range, showing that a microprocessor can perform more instructions in a shorter period than a microcontroller. Consequently, it has a higher power consumption. Therefore, microprocessors are used in projects that require more processing, such as computer vision, complex calculations, and artificial intelligence.

This section will cover some commonly used platforms to develop IoT projects and embedded systems, according to [21]. In Sect. 2.1 the Arduino platform is presented, being one of the leading platforms to have driven the maker community worldwide. Section 2.2 shows the ESP8266 and ESP32 platforms, highlighting the integrated connectivity and its low cost. Finally, Sect. 2.3 discusses the Raspberry Pi, a platform with a microprocessor in order to satisfy the needs of a typical user, but with appealing features for those who want to develop embedded projects with a little more processing.

2.1 Arduino

Arduino[1] is an easy-to-use open-source electronic project prototyping platform. Conceived by Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, and David Mellis, initially to help design students, it became quite popular, causing user and developer communities to emerge in several countries. It has several models, in addition to accessories that can add new features to the mainboards. With its low learning curve, we can affirm that Arduino facilitated the prototyping of embedded systems.

Figure 3 shows images of some Arduino boards. We can see shared elements between them, such as the USB connector in the upper-right corner and the external power connector in the upper-left corner. Both also have a RESET button, in red color, to reset the embedded firmware and the embedded user application. On the sides are the pins provided for the user, including the digital, analog, Vcc(5V and 3.3V), Ground pins, as well as pins that support some serial communication protocols, which will be seen in more detail in Sect. 4.

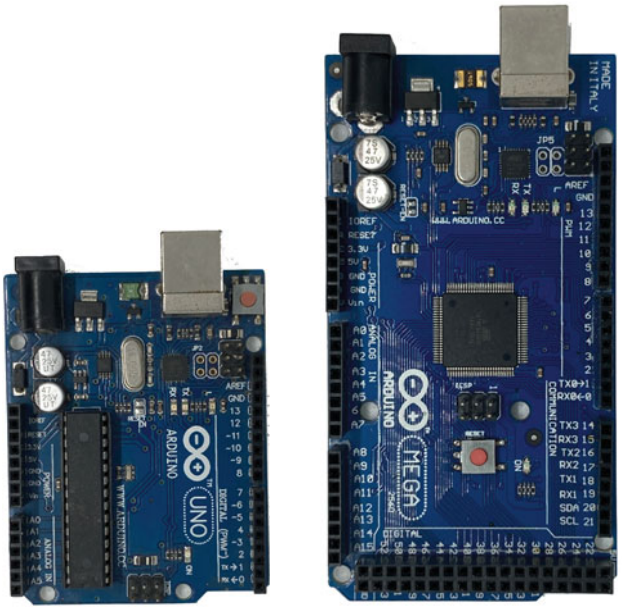
In 2005, Massimo Banzi, while professor of the Interaction Design course at an institute in Ivrea, Italy, presented the prototype of the Arduino. His main idea was to provide his students with a more suitable platform for their projects. This platform should have a shorter learning curve, not require advanced electronics and programming skills, and be accessible. The platforms available at that time were expensive and did not have the necessary computing power to carry out the students' projects.

Regarding programming, he used a project presented on Hernando Barragàn's master's thesis, the **Wiring**, which encompasses a programming language and an IDE, based on the **Processing** project by Casey Reas and Benjamin Fry. The use of these other projects allowed a more user-friendly and intuitive development environment for its users. Another very relevant issue was the adoption of an open-source model, being the first widespread open-source hardware project, according to [3]. According to [8], one of the intentions of adopting this philosophy was to make the project self-sustainable, allowing the easy replication of the boards and development of new libraries for adding more functionalities to the original project.

Trivia

Arduino was the name of a king who ruled the Ivrea region in the Middle Ages and also a bar that Banzi attended. Then came the inspiration for the platform name.

[1] https://www.arduino.cc/.



(a) A (b) B

Fig. 3 Images from Arduino boards: **a** Arduino UNO; **b** Arduino MEGA

The Arduino platform has several board models in the most different sizes and applications. It usually uses Atmel AVR microcontrollers (manufacturer acquired by Microchip in 2016 [7]), but offers models with other architectures, including ARM. The most used Arduino models are the **Arduino UNO**, **Arduino MEGA**, and **Arduino Nano**.

Arduino UNO, according to [4], is the flagship of these boards and the most used one. It has an 8-bit ATmega328P microcontroller, 16 MHz clock, 14 digital pins (where six pins are also Pulse Width Modulation (PWM) enabled), and six analog pins operating between 0 and 5V, and supporting a maximum current of 20mA per pin (50mA on a pin of 3.3V). To connect to the computer, it has a USB Type B input, where it can also be used as a power supply and an input for an external power supply between 7 and 12V.

Arduino Mega 2560 is helpful for scenarios that need one microcontroller for controlling several devices at once. It has the 8-bit ATmega2560 microcontroller, 16 MHz clock, 54 digital pins (where 15 of them are PWM enabled), and 16 analog pins, operating between 0 and 5V, supporting a maximum current of 20mA per pin (50mA on 3.3 pin V). It has a USB Type B input for communication with the computer and also as a power source. It also has a pin to connect an external power source, supporting between 7 and 12V.

The Arduino Nano is one of the tiniest boards, ideal to be used in a breadboard. It features the 8-bit ATmega328 microcontroller, 16 MHz clock, 22 digital pins (6 PWM

pins), and eight analog pins, operating between 0 and 5V, supporting a maximum current of 20mA per pin (50mA on the 3.3 V pin). Due to its small size has a Mini-B USB port for interfacing with the computer and power. It has no dedicated pin for using an external power source. It supports power between 7 and 12V.

2.2 ESP8266 and ESP32

The ESP8266 and ESP32 chips have become quite popular, even for hobbyists and makers, due to their integrated connectivity (built-in WiFi and Bluetooth modules) and affordable price. They were developed by **Espressif**,[2] a fabless manufacturer headquartered in China with branches in Singapore, India, Czech Republic, and Brazil. They are both produced in SoC and module formats, where the choice for which to use will depend on factors such as the project requirements and available budget.

! Attention

The ESP8266 and ESP32 operate with 3.3V voltage on their input pins, unlike the Arduino, which supports up to 5V. Values higher than 3.3V can damage the chips.

In Table 1 are the main characteristics of each chip. The ESP8266 has built-in Wi-Fi, general-purpose pins to interface with other peripherals such as LEDs and sensors and support for communication protocols such as UART, I^2C, and SPI. Its low energy consumption is an important feature for IoT projects. It allows interfacing with other peripherals and telemetry, connecting to local networks or the Internet, adding value to projects.

ESP32 can be considered a successor to ESP8266 but not a replacement. In addition to the features of the ESP8266, it also brings Bluetooth connectivity, including Bluetooth Low Energy (BLE), CAN interface (very common in the car industry), and embedded sensors such as capacitive touch sensors and hall effect. Its ADC has more precision than ESP8266, being interesting for projects that use this feature. The lower energy consumption in sleep mode is worth mentioning, with an electrical current of only μA. All these features increase the price of the ESP32, which currently costs an average of 3 times more than the ESP8266.

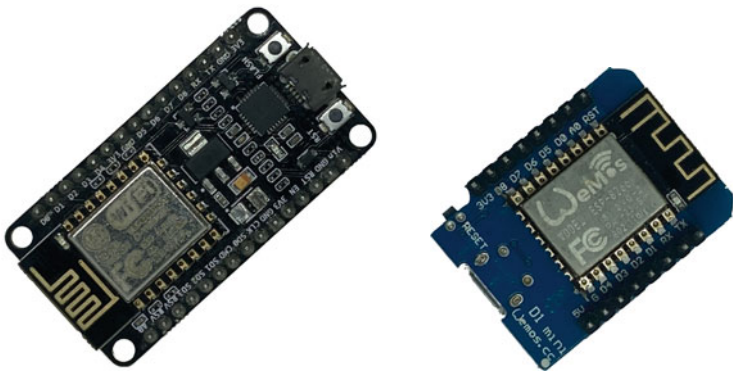
Some platforms facilitate using these microcontrollers, avoiding the need for an extra circuit to use them. In Fig. 4 are the NodeMCU and Wemos platforms. The NodeMCU boards came in versions with ESP8266 or ESP32. With it, it is possible to power up and communicate with the chip using a USB cable and programming it with Arduino's Wiring IDE, or PlatormIO,[3] an IDE with support for a considerable

[2] https://www.espressif.com.

[3] PlatformIO—https://platformio.org/.

Table 1 ESP8266/ESP32 specs

	ESP8266	ESP32
CPU	32-bit 160MHz Single Core	32-bit Single or Dual Core and adjustable frequency range clock 80MHz–240 MHz
WiFi	2.4 Ghz	2.4 GHz
Antenna	+19.5 dbm	+19.5 dbm
Sleep current	20 mA	5μA
ADC	10-bit	12-bit
Peripherals	UART, GPIO, I^2C, I2S, SDIO, PWM, ADC and SPI	Capacitive touch sensors, Hall sensor, SD card interface, Ethernet, high-speed SPI, UART, I2S, I^2C, CAN
Bluetooth	No	Classic Bluetooth and Bluetooth Low Energy (Bluetooth LE)



(a) A (b) B

Fig. 4 Images from ESP8266 prototyping platforms: **a** NodeMCU; **b**Wemos

amount of boards. It is also possible to use the Arduino libraries developed by the community, helping develop projects and prototypes. The Wemos[4] platform also has versions using ESP8266 or ESP32, and some of its models have plates with smaller dimensions than the NodeMCU.

[4] https://www.wemos.cc/.

2.3 *Raspberry Pi*

The **Raspberry Pi**[5] is a small computer, which can connect peripherals such as keyboard, mouse, and monitor, giving to its users: Internet access, office tools, software programming tools, and even some games. Officially, it uses a Debian-based GNU/Linux distribution, Raspberry OS (formerly Raspbian), but it is possible to install other Linux distributions and operating systems. Over time, other models were launched, some with different sizes and applications. One of them is the **Raspberry Pi Zero**, which is even smaller than the original model, but with less computational capacity. Fig. 5 shows some Raspberry Pi boards.

Raspberry Pi was born to facilitate and spread the teaching of computing in schools. Its origin dates back to 2006, at the University of Cambridge. The Computer scientist Eben Upton, Rob Mullins, Jack Lang, and Alan Mycroft worried about newcomers who arrived without having had much contact with technology at school. The Raspberry Pi Foundation was established, and six years later, in 2012, the first version of the board came out.

The first models were launched in 2012, but only the latest models will be presented here. Table 2 shows the main features of the Raspberry Pi models launched from 2017 until now. The latest models bring a robust configuration to their size, meeting the demands of the most basic users without any significant problems. In addition, connectivity is not a problem. Current models bring integrated support to technologies such as Bluetooth and WiFi, allowing for greater portability of the equipment in the most diverse uses and projects.

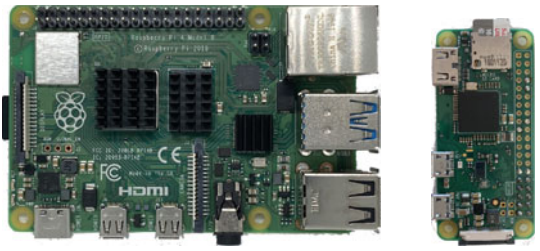
The Raspberry Pi uses a microSD card to install the operating system and store user data. If desired, an external hard drive can be connected via USB to increase storage capacity.

The Raspberry Pi has a 40-pin header that provides many features. Through them, it is possible to have access to continuous voltages of 5V and 3.3V, Ground, GPIOS, as well as pins with support for some serial communication protocols, such as SPI and I^2C (more about these protocols refer to Sect. 4). Furthermore, just like the Arduino, the Raspberry Pi has accessories that can extend its functionalities, such as LCD touchscreens, LED matrix, cameras, and even mountable shields that allow the user to weld the components they want.

Trivia

The **Raspberry Pi** has become well known for being a credit card-sized computer and costing only $35.

[5] https://www.raspberrypi.org/.



(a) A (b) B

Fig. 5 Images from Raspberry Pi boards: **A** Raspberry Pi 4 Model A; **B** Raspberry Pi Zero

Table 2 Raspberry Pi specifications

Model	RPi 3	RPi Zero W	RPi 3 B+	RPi 4 B
Launch year	2016	2017	2018	2019
CPU	Cortex-A53 64-bit	ARM1176JZF-S	Cortex-A53 (ARMv8) 64-bit	Cortex-A72 (ARM v8) 64-bit
Cores	4	1	4	4
CPU Clock	1.2 GHz	1 GHz	1.4 Ghz	1.5 GHz
GPU	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore VI
RAM	1GB	512MB	1GB	2/4/8GB
Wireless	802.11n	802.11n	802.11n 2.4/5 GHz	802.11ac 2.4/5 GHz
Bluetooth	4.1 (BLE)	4.1 (BLE)	4.2 (BLE)	5.0 (BLE)
Power ratings	350mA	150mA	500mA	600mA

Because Raspberry Pi has a microprocessor as its core, instead of a microcontroller as Arduino boards and ESP8266/ESP32-based boards, it ends up putting it in another category. As seen earlier, microprocessors do better in projects requiring more processing, so it is pretty common to use a Raspberry Pi to host a web server and run computer vision or artificial intelligence algorithms. The Arduino and ESP8266/ESP32 do better on projects with more specialized and repetitive tasks due to the lower clock rate, such as controlling motors or collecting data from sensors.

3 Sensors

Sensors are devices used to measure or capture various physical phenomena and translate them into a human- or machine-readable format. The working mechanisms of these sensors are many, and in this Section, we will describe some of them, focusing on sensors built to be interpreted by machines. Most electronic components provide all of their technical specifications in a document called a datasheet. With sensors, it

is no different. Datasheets are essential in constructing projects involving sensors and will be mentioned frequently throughout the text. Websites like alldatasheets.com[6] can be used for finding datasheets.

3.1 Resistor-Based Sensors

These sensors have the principle of changing their resistance according to the variation of the physical magnitude that one wishes to observe. There are resistor-based sensors capable of measuring temperature, humidity, pressure, electrical conductivity, light intensity, etc. Integrating these sensors to a microcontroller involves the use of a voltage divider as illustrated in Fig. 6. A resistance change of the sensor, represented in the Figure by R_L, necessarily implies a variation in the value of the output voltage V_{OUT}. It is possible to relate and interpret the voltage value obtained and measure the observed physical phenomenon applying an analog-to-digital converter and a characteristic curve, available in the datasheet. The Eq. 3 gives the value of V_{OUT} in this circuit. With the circuit arrange in Fig. 6, a decrease in the resistance R_L implies a reduction in the output voltage V_{OUT}. If we desire the inverse behavior, it is necessary to interchange the resistors R_L and R_S.

The LDR sensor, the acronym for Light-Dependent Resistor, is a component consisting of a high-resistance semiconductor. When receiving numerous photons from incident light, it absorbs electrons, improving its conductivity and reduces its electrical resistance. In this way, an LDR placed in the position of the resistor R_L in Fig. 6 would present the behavior of reducing the value of the voltage V_{OUT} with increasing light intensity.

Another example of resistor-based sensors is thermistors. These components have the property of varying their resistance depending on the environment's temperature. In general, these components have a nominal resistance that is associated with a reference temperature (room temperature), usually 25°C. Each component has its operating range and a curve that correlates the resistance value to the temperature. Thermistors come in two main classes: NTC, widely used as temperature sensors, which decreases their resistance with increasing temperature; PTC, usually used as inrush current limiters and overcurrent protectors, present the inverse of NTC behavior.

$$V_{OUT} = \frac{R_L}{R_S + R_L} \times V_{IN} \tag{3}$$

[6] https://www.alldatasheet.com/.

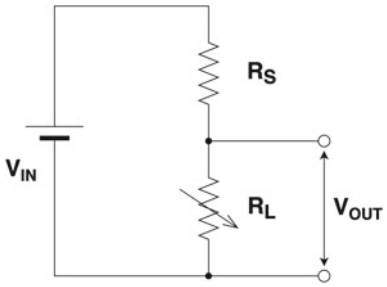


Fig. 6 Voltage divider circuit

3.2 Hall-Effect Sensors

The phenomenon identified by Edwin Hall in 1889 is the principle of the Hall-effect sensors. A potential difference is created by applying a magnetic field perpendicular to a conductor, and an electric current flow, proportional to the applied magnetic field, is generated.

It is possible to build different types of sensors based on this principle. For example, linear position sensors, widely used in industry, relate their position to the strength of the observed magnetic field (ratiometric). Another common practice is associating a hall-sensor to a hysteresis circuit generating pulses when the intensity of a magnetic field increases or decreases. This strategy allows treating the pulses as digital information. Using a series of sensors and varying their positions to the magnetic field, it is also possible to build solutions capable of determining the speed and direction of movement of various mechanisms, such as brushless motor encoders. Another common application is the measurement of electrical current without the need to break the circuit. When flowing through a conductor, an electrical current also generates a magnetic field. It is possible to measure this magnetic field and deduce the electrical current intensity by positioning a Hall-effect sensor perpendicular to this conductor.

Some rain gauges are built-up on the hall-effect. A rain gauge consists of a funnel that conducts rainwater into a tipping-bucket mechanism. Each bucket is calibrated so that it tips when a settled amount of water accumulates. When this happens, the water spills out from one bucket, and another one starts to receive the water flow. Once it reaches its capacity, the process will repeat itself regularly like a seesaw during the rain. There is a permanent magnet and a hall-effect sensor associated with the tipping-bucket mechanism in a way that generates a pulse whenever it tips. By counting these pulses, it is possible to determine the amount of rain in a given period.

One of the significant advantages of Hall-effect sensors is to allow the construction of less invasive systems, avoiding direct mechanical contact and reducing wear, which in many applications is essential.

3.3 Integrated Circuit Sensors

Various sensors are available, as we have seen so far, and each has its operating principle. With the advance of microelectronics in the last decades, sensors are becoming more complex and cheaper. There is a great tendency to build them up as integrated circuits of varying complexity. The TMP36,[7] for example, is a low-cost integrated-circuit temperature sensor with an output voltage linearly proportional to the centigrade temperature. It has a scaling factor of 10mV/°C. Unlike the NTC thermistor mentioned in Sect. 3.1, there is no need to implement a voltage divider circuit for the temperature to be measured. We only need to power the IC up and connect the linear voltage output of the sensor to the microcontroller's ADC. In general, the accuracy of this sensor is greater than that of a conventional thermistor. There are several types of similar temperature sensors on the market with different operating characteristics—for example, the LM34,[8] a sensor that is also linear but calibrated for temperatures in Fahrenheit.

There are even more complex sensors. An example is the MPU-6050,[9] a triaxial accelerometer and gyroscope combined with a digital motion processor (DMP) on the same integrated circuit. The DMP can be configured, for example, to detect no motion or free-fall motion through accelerometers' and gyroscopes' raw data and generate an interrupt to a microcontroller when these situations are verified. These features enable the use of these sensors in the most diverse ways and applications. The sensor market continues to expand, and the trend is to have even more diverse, accurate and cheaper sensors.

3.4 Counters and Timers

Counters are digital electronic devices designed to count events. We find them as integrated circuits dedicated to this function or as peripherals inside microcontrollers. Let us consider the 74HC590 integrated circuit from Nexperia[10] as an example. It is an 8-bit binary counter with the interface shown in Fig. 7. The bars over the signal names indicate that these signals are low-level active. The $\overline{OE}$ (output enable) signal, when active (LOW), enables all eight Q outputs removing them from a high-impedance state. The counter has two synchronism or clock signals. The first one, the CPC (counter clock), will increment by one the eight-bit internal binary counter value whenever it receives a positive edge trigger. This counter is associated with an eight-

[7] TMP36 datasheet—https://www.analog.com/media/en/technical-documentation/data-sheets/TMP35_36_37.pdf.

[8] LM34 datasheet—https://www.ti.com/lit/ds/symlink/lm34.pdf.

[9] MPU-6050 datasheet— https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf.

[10] Nexperia 74HC590 Datasheet—https://assets.nexperia.com/documents/data-sheet/74HC590.pdf.

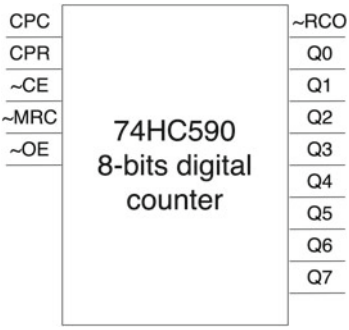


Fig. 7 74HC590 logic block

bit storage register that will keep its state until the CPR (register clock) signal is activated through a positive edge trigger. When CPR is active, the storage register and Q outputs are updated with the counter value. It is possible to merge the entries CPC and CPR so that they are activated simultaneously. When configured in this way, the counter will always be one count ahead of the register. The input signal $\overline{CE}$ (count enable) inhibits counts by ignoring eventual activations of CPC and CPR, preserving the current state of the counter. The $\overline{MRC}$ (master reset counter) signal is responsible for clearing the counter by resetting its count to zero. Finally, the signal $\overline{RCO}$ (ripple carry output) allows the construction of binary counters with a larger number of bits, connecting them in cascade. For this setup, we connect the $\overline{RCO}$ of a 74HC590 to the CPC of the next one in a chain. Table 3 summarizes the functioning of the counter.

Most of the counter chips available on the market have interfaces very similar to the 74HC590 and the feature of cascading them together to increase the counting capacity of a project. Another widespread type of counter is the decade counter. This counter has the characteristic of counting from zero to nine only. When the counter reaches nine, it returns to zero in the next cycle. By connecting decade counters in a cascade, we can do a decimal-based count, increasingly associating them with units, tens, hundreds, and so on.

Counters are of great importance when reading some sensors. Let us consider the rain gauge presented as an example of the use of Hall-sensors in Sect. 3.2. The tipping-bucket mechanism is calibrated so that it tips whenever a settled amount of rain accumulates. When this occurs, a pulse is generated to counter's input. If we desire to compute the amount of rainfall in one hour, we only have to multiply the counter value in this time window by the calibrated capacity of the bucket. Another common application of counters is to integrate them with motor encoders. As the motor turns, the encoders generate pulses, and the counter counts them. This count, when associated with the time interval, determines the speed at which the motor turns.

In microcontrollers, counters are usually present in the form of peripheral timers. Timers have the exact working mechanism as counters with the difference that they input periodic events from a clock. Imagine that we have a clock that generates an

Table 3 74HC590 function table

$\overline{OE}$	CPR	$\overline{MRC}$	$\overline{CE}$	CPC	Description
HIGH	X	X	X	X	Q outputs disable
LOW	X	X	X	X	Q outputs enable
X	PE	X	X	X	Counter data stored into register
X	NE	X	X	X	Register stage is not changed
X	X	LOW	X	X	Counter clear
X	X	HIGH	LOW	PE	Advance one count
X	X	HIGH	LOW	NE	No count
X	X	HIGH	HIGH	X	No count

X = don't care; PE = positive edge trigger; NE = positive edge trigger

event every second for a counter input. If we count '3600' events in this counter, we know that the one-hour interval has passed. Timers are also essential when reading sensor data. Many sensors need to be read periodically, and we use timers to signal this reading period. We will discuss in Sect. 3.5 how we use microcontroller timer interrupts to implement periodic sensor readings. Like counters, it is also possible to cascade timers to measure longer time intervals.

3.5 Polling, Interruption and DMA

One of the most critical steps in sensing is collecting sensor data to store and process it on the computing device. The data captured by the sensor can be available to the processor of a microcontroller in different ways: in a general-purpose digital pin, it can assume 'LOW' and 'HIGH' values only; available in some peripheral register, such as analog to digital converters; or by reading a bus that connects both devices, such as the I^2C. In all cases, we necessarily need to read these ports, peripherals, or buses. The reading task is critical because, depending on how we implement it, it can result in data loss, which in many applications can be unacceptable. The three most common ways to fetch this data are polling, interrupt and direct memory access, DMA.

In polling, the processor is primarily responsible for fetching the data. The programmer's code execution flow determines the moment of data collection. Therefore, the entire responsibility to ensure not losing data is in his hands when evaluating the workload imposed on the processor. If the processor takes more time to perform another task, it will undoubtedly lose sensor data. We can make the analogy of the

polling process with that of checking a mailbox. Bob has several duties to do during the day. One of them is going to his mailbox and pick up all available mail. Realize that He needs to stop what he is doing to go there and pick up his mail. The same thing happens to the processor. He starts executing a reading routine at the expense of running another task. Polling works very well when we have events that do not have a very high frequency. Otherwise, the processor will use all its valuable time reading a single piece of data. Note that in applications that are not periodic, the processor will eventually stop what it was doing to read data and find nothing available.

An alternative to polling is interruptions. In this case, the microcontroller will associate an event with an interrupt service routine (ISR) call. When an interrupt occurs, the processor stops instantly, and the execution flow switches to this ISR. Different from polling, an interrupt guarantees that there is available data for fetching. It is a good programming practice to keep ISRs as short as possible, as this could seriously impact the system's main execution flow. To keep ISRs shorter, the programmer usually only registers that event in some internal data structure so that the processor can handle it more appropriately. One interrupt may also occur while serving another one generating what we call nested interrupts. There are different ways to handle nested interrupts. The simplest way is to disable interrupts when running an ISR. This solution is quite simple but can lead to events loss. A second approach present in many microcontrollers is the possibility of not preempting the execution of the ISR and automatically registering the occurrence of a new interrupt on an interrupt vector. The microcontroller can then handle the interrupt vector as soon as it completes the previous ISR. This solution solves interrupt loss but does not allow higher priority interrupts, standard in critical systems, to be immediately serviced by the processor. In this case, some microcontrollers enable the implementation of interrupt priorities. Such a feature gives excellent flexibility to time-critical applications. Naturally, the more critical the application is, the more complex the interrupt handling mechanism will be. The handling of interrupts is undoubtedly one of the most challenging topics in developing embedded systems. Because of its non-deterministic nature, it is almost impossible to predict the behavior of a system. Hence, it is an excellent practice to adjust the interrupt mechanism to the needs of the target application, always keeping it as simple as possible.

For a better understanding, we can make the analogy of an interruption system with phone calls. When the phone rings, we usually immediately stop what we are doing to answer that call. If what we were doing before the phone rang was very important, and the call took too long, we could be in an uncomfortable situation. One solution is to write down the caller's number and indicate that we will call back as soon as possible without seriously affecting the task we were performing before. The phone call analogy also applies to nested interrupts. The simplest solution is to signal a new caller that we are already on a call through the busy tone. In this case, we would lose the ring. The second option would be to route the caller to a voice mailbox, allowing them to leave a message so that we can return later, similar to an interrupt vector. Moreover, the last possible situation is the 'call waiting' function. We can switch to the incoming caller when a new call arrives by leaving the first one on hold until the highest priority caller completes their call. It is similar to what

we have explained about interrupt priorities on microcontrollers. Perceive how the complexity of handling incoming calls increases when we modify the way we take them. Therefore, we must use the most appropriate tool in each desired situation.

The last way to fetch the data is direct memory access. DMA is a peripheral that is not available on all microcontrollers. Usually, only high-end microcontrollers have it. It typically applies to applications where there is intensive data transfer. The processor does not participate in the transfers carried out by DMA, being free to carry out other activities. Let's remember Bob's problem in Sect. 1.2. Bob needed to capture the full audible spectrum of his music. Hence, he needed an AD converter that reads the signal generated by the microphone at a sampling rate of 40 kHz (see Nyquist–Shannon sampling theorem, in Sect. 1.2). It means that each 25 μs an ADC read needs to be performed. Note that this task would leave the processor completely busy if we used polling or interrupting techniques. In this case, we can program the DMA to transfer data from the ADC to the memory periodically. In Bob's example, we set this period to 25 μs. At the right time, the processor would access the memory to process the collected data in the best way possible.

3.6 Sensors in the Arduino Ecosystem

The growth of the Arduino ecosystem also had a significant impact on the sensor industry. In recent years, a vast ecosystem of sensors has become available at very affordable prices, enabling hobbyists, educators, researchers, and even designers who are not developing critical solutions, to use them [17]. The low-cost nature of Arduino's philosophy has created the demand for low-cost sensors. This sensor group often does not meet the stringent standards imposed on devices applied in industry or high-end consumer goods. Still, they can meet a range of other applications that do not require high accuracy or reliability.

To make the use of these sensors straightforward, many of them are available on the form factor of modules or shields. It allows to connect them to prototyping platforms in a very convenient way. There are also standardized mechanical interfaces, such as Seeed Studio's Grove System.[11] The Seeed Studio's approach provides standard connectors with an excellent appeal in educational applications and rapid prototyping of projects.

4 Hardware Interfaces

The microcontroller uses the hardware interfaces to communicate with other components in the system. These interfaces can use a Serial or Parallel communication model. Based on these models, protocols were developed to standardize how the

[11] Grove System— https://wiki.seeedstudio.com/Grove_System/.

low-level transmission is carried out to guarantee communication effectiveness. This section will present the Serial and Parallel communication concepts and some of the hardware interface protocols most employed by embedded systems.

A **Hardware Interface** specifies the physical and logical characteristics to enable communication between system components and the CPU. **Drivers** are used to control and trigger functions to execute some action based on the signals transmitted.

4.1 Serial and Parallel

Serial communication consists of sending and receiving data bit by bit, using basically two wires, called *Rx* and *Tx*, respectively. It can be implemented using formats such as **USART (universal synchronous/asynchronous receiver/transmitter)** or **UART (universal asynchronous receiver/transmitter)**, as seen in [25].

In **UART**, the clock is generated internally in the microcontroller, and it is synchronized with the data stream using the start bit. To receive data, as no clock signal is sent during transmission, the receiver needs to know the baud rate of the channel beforehand so that the data can be correctly received. The **baud rate** is defined in bits per second (bps), which tells how many times a signal on a communication channel can change its state in one second. The **Bitrate** is the number of bits transferred in one second, and it is calculated by multiplying the Baud rate by the number of bits per signal, resulting in a value measured in bps.

In **USART**, the synchronous mode can be set. In this mode, the clock signal is sent together with data, and an exclusive line can be used to send this clock. With this, the target peripheral does not need to know the baud rate. Furthermore, components must send data simultaneously, even when there is no need.

Communication can operate on one or two wires. When operating on two wires, it is called *full-duplex*, where each one is dedicated to sending or receiving data. With one wire, it is called *half-duplex*, where the wire alternates its function of sending and receiving data.

Parallel communication, instead of sending all the bits on a single wire, each bit of the packet has a dedicated wire to be transmitted. This approach increases the transmission speed as they are sent simultaneously. The amount of bits usually used is 8 bits or multiples. As a result of the need for more wires, its implementation cost is higher than Serial. Typically, parallel buses are half-duplex, with other pins controlling the data direction, clock, and chip connection [25]. It is widely used on buses for connecting peripherals to motherboards, such as video cards.

Due to the high implementation cost and complexity of parallel communication, especially with the increased efficiency of serial communication, it has its application only in specific projects, such as peripheral buses.

4.2 RS-232/RS-485

RS-232 (or EIA/TIA-232) is a standard binary serial communication protocol used to exchange data between DTE (Data Terminal Equipment) and DCE (Data Communication Equipment) devices created in the decade of 1950. The DTE is a data generator or consuming device, a PC, for example, and the DCE is a networking device, such as a modem. Its usage has diversified over time, and it is currently used in embedded systems, including at the industrial level, for communication between devices. It can even be used with Arduino or Raspberry Pi. It supports both synchronous and asynchronous modes, although the former is rarely used.

RS in RS-232 and RS-485 stands for **Recommended Standard**, a nomenclature used by the Electronic Industries Alliance (EIA) and Telecommunications Industry Association (TIA). The RS prefix has been replaced by **EIA/TIA** in recent years.

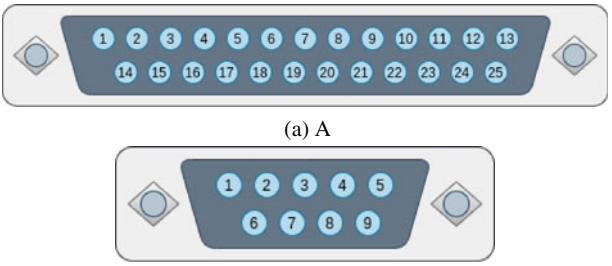
Info

RS-232 has more than nine defined signals, but almost no projects use this total. Many use a maximum of eight signals.

In Table 4 are RS-232 protocol signals present on DB-9 connectors and their equivalent on DB-25 connectors. Their positions on the connectors are indicated according to the standard. In Fig. 8 are the form factor of the DB-25 and DB-9 connectors, with their numbered pins. The DB-9 is the most common connector. The DB-25 is usually applied when we need to connect secondary equipment to a single port. Pay attention to the length of the cable, as very long cables, or with high capacitance, can cause transmission problems. Cables up to 15m (50 feet) in length, or capacitance up to 2500 pF, are recommended for the highest speed, but special cables can be used for lengths up to 20 times longer [25].

Table 4 Rs-232 defined signals

Signal	Name	Pin DB-9	Pin DB-25	Description
DCD	Data Carrier Detect	1	8	Indicates when the DCE is receiving a carrier from a remote DCE
RxD	Receive Data	2	3	Read data from the device on the other end of connection
TxD	Transmit Data	3	2	Send data to the device on the other end of connection
DTR	Data Terminal Ready	4	20	Indicates that the DTE is ready to initiate, receive or continue the call
GND	Ground signal	5	7	Common ground
DSR	Data Set Ready	6	6	DCE is ready to receive or send data
RTS	Ready To Send	7	4	DTE requests data from DCE
CTS	Clear To Send	8	5	DCE is ready to receive data from DTE
RI	Ring Indicator	9	22	Detects a ring signal on the phone line



(a) A

(b) B

Fig. 8 DB-25 and DB-9 interfaces examples: **a** DB-25; **b** DB-9

For connection between two DTE (two computers, for example), a special cable called **Null modem** is used. In this cable, the RxD and TxD pins are inverted at the ends. The RxD is on pin 2, while on the other side, it is on pin 3. The TxD follows the same logic.

In Fig. 9 are the voltage levels at which operations occur for data and control. In data, on the RxD and TxD pins, logic level 1, called Mark, is represented in the negative voltage range, between −15V and −3V, and logic level 0, called Space, is represented in the positive range, between +3V and +15V. The same negative range

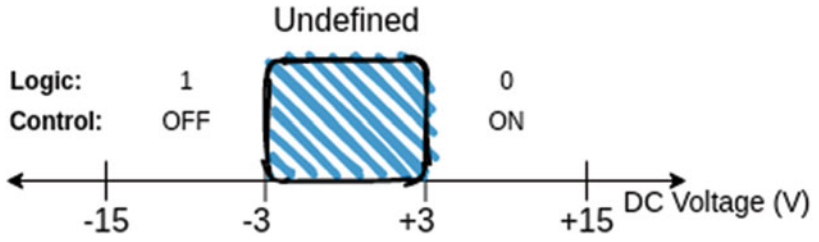


Fig. 9 RS-232 communication voltages

represents the status Off for the control pins, while the same positive range represents the On status. The range between −3V and +3V is undefined in the standard, and the data is considered invalid.

Tips

RS-232 communication can be done using just RxD, TxD and Ground.

RS-485 (or EIA/TIA-485) is another communication standard created by EIA and TIA that defines a serial binary communication channel that allows communication with multiple devices on a network, differently from RS-232 whose transmission occurs just between two devices. Up to 32 devices can be part of an RS-485 network whose recommended topology is the bus, connected through the same wiring. Topologies such as ring or star are not recommended, although they can be used with special repeaters.

For connection, it is recommended to use a twisted pair of wires in addition to the ground wire. The recommended length of this wire is up to 1,200 m (4,000 ft), showing that it can cover a larger area than using RS-232. The shorter the length of this wire, the higher speeds can be used, reaching a hypothetical maximum of 10 Mbps, but some drivers can reach higher bitrates. A Rule of Thumb is that the length of the wire, in meters, multiplied by the speed in bits/sec, must not exceed 10^8. For example, a 100m wire reaches a maximum of 1 Mb/s. Although there are more efficient drivers available, it helps to understand what to expect from the project circuit.

It can work in half-duplex and full-duplex (with 4 data wires), but the latter is not commonly used. Data transmission is done using two signals, named A (U+) and B (U-), and the status will be according to the level and potential difference between them. In the scenario where signal A is low, and signal B is high, it is the status of Mark or 1, and the reverse is the status of Space or 0. Signal levels must respect the acceptable continuous voltage range of −7V to +12V, where any value outside of that is considered invalid. The recommended differential should be at least 1.5 V at the output through a load of 54 Ω, and 200 mV at the input. In Fig. 10 the Mark and Space signals in the data transmission are shown.

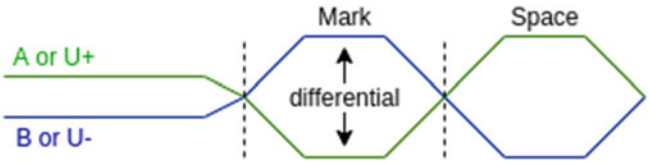


Fig. 10 RS-485 communication diagram

4.3 USB

In order to universalize the connections of peripherals in various devices, in 1996, the Universal Serial Bus (USB) was launched, whose adoption by manufacturers has grown over the years. Its concept was developed by a joint work of several organizations, including IBM, Microsoft and, Intel. It is currently maintained by the USB Implementors Forum (USB-IF),[12] a non-profit organization that regulates the standards and specifications of each version of the protocol.

In Table 5 are the versions of the USB protocol and their respective data transmission rates. When comparing version 1.0 with 3.2, there was an increase of approximately thirteen thousand times in the transmission rate, demonstrating how much the technology has evolved. Each version also carries a title, wherein version 1.0 is "low-speed," and version 3.x is "superspeed."

All communication using USB is initiated by a Master device, which is a computer or other controller. A Slave device, a mouse or keyboard, for example, sends essential data to the Master for its proper functioning. Each interaction is composed of a **Token packet**, optionally a **Data packet**, and ending with a **Handshake packet**.

The Token packet initiates the transaction, indicating whether the Master will receive or send data. From version 3.0 onwards, the Token packet is incorporated into the Data packet in data sending operations or replaced by a Handshake packet in data request operations. The Data packet transmits the desired data, and the receiver can return an error on not receiving or even request a pause when it is not available to receive. The Handshake packet informs the success of receiving data or possible errors.

In Fig. 11 are the essential pins in a USB communication. There are other connectors and cables with more pins, like USB type C, for example, but the scope of this section is to show the basic model. There are Ground and Vcc (+5V) pins in addition to the data pins. On the data pins, in order to avoid noise, they work as a differential pair. When '1' value (HIGH) is sent, a positive flow goes through the D+ (or Data+), and a negative flow goes through the D- (or Data-). So that when '0' value (or LOW) is sent, a positive flow through D- and a negative flow through D+.

[12] https://www.usb.org/.

Table 5 USB versions and speeds

Version	Speed
1.0 (Low-speed)	1.5Mbps
1.1 (Full-speed)	12Mbps
2.0 (High-speed)	480Mbps
3.0 (Superspeed)	5Gbps
3.1 (Superspeed)	10Gbps
3.2 (Superspeed)	20Gbps

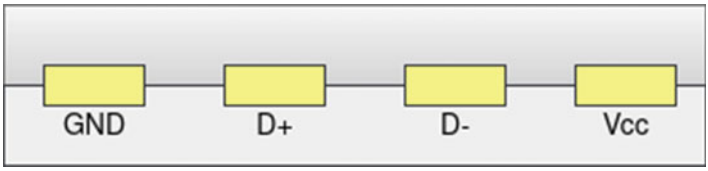


Fig. 11 USB connector pins

4.4 SPI

The Serial Peripheral Interface (SPI) protocol is a synchronous communication serial interface for short distances, suitable for components on the same circuit board [28]. Created by Motorola, it became a protocol standard, widely used for connecting LCD screens and SD memory cards, for example. It operates in full-duplex, using a Master-Slave architecture where only one Master component coordinates data transmissions for multiple Slaves.

The basic structure of the SPI requires a minimum of 4 lines. Data transmission is divided into two lines, **Master-Out-Slave-In (MOSI)** and **Master-In-Slave-Out (MISO)**, where the first is responsible for transmitting data from the Master for Slaves, and the latter is responsible for the flow of data from a Slave to the Master. We use the **Chip Select** line ($\overline{CS}$) to select the desired Slave, which can be more than one. The number can vary, as each one is dedicated to a single peripheral. The **Clock (SCLK or CLK)** transmits the clock signal generated by the Master to all Slaves.

Info

The **Chip Select** line can also be named as **Slave Select** ($\overline{SS}$).

The SPI is a synchronous serial protocol, so the Slaves remain in constant communication with the Master even when they are not active at CS. In contrast, the Clock remains active (the default in this situation is to send 1 byte with value 0xFF) [25]. Communication starts with the Master setting a low level on the pin $\overline{CS}$ of the

Slave that wants to communicate, keeping it that way until the end of transmission. Data is sent one bit at a time on the MOSI pin, being received on the Slave's MOSI pin. If requested, the Slave sends an acknowledgment by the MISO pin, one bit at a time, received on the Master's MISO pin.

We can send data continuously because the interface does not use any particular bit to start or end the transmission. It can send and receive data simultaneously, as there are dedicated pins for data output and input. However, it only allows a single Master in the circuit, and there is no way to check if the data arrived intact at the destination.

4.5 I^2C

The Inter-Integrated Circuit protocol (I^2C) is a serial interface that supports multiple Masters and Slaves. Its physical infrastructure is more straightforward than the SPI, requiring only a data line (SDA), a clock line (SCL), and the Ground. All devices are interconnected by these wires in common and have pull-up resistors on the data and clock wires, with voltages that can be 5V or 3.3V. To identify the devices, each has a unique address of size 7 or 10 bits, which can be checked in their respective documentation, but it is worth noting that addresses with 10 bits are rarely used. With 7 bits, it is possible to have 128 devices on the network.

The I^2C works with two bidirectional open-drain lines, in addition to the **Ground**. One of these lines is **Serial Data (SDA)**, where the data from the current Master to the Slaves is transmitted and vice versa. The other line is **Serial Clock (SCL)**, which is responsible for transmitting the clock signal generated by the Master.

Figure 12 demonstrates the master communication structure for a Slave using I^2C, and Table 6 shows the logic levels and transitions to perform some actions during the transmission. The first step is for the Master to send the Start condition and then send the address of the Slave it wants to communicate with to everyone on the network. The next bit is to indicate whether the Master wants to receive or send data to the Slave. If there is a Slave with the requested address, it will respond on the SDA line with a low-level signal. Sending data are packed into 8-bit data frames. After each data frame, an ACK, a low-level signal on SDA, is generated in the case of success. In the end, the Master sends a stop condition. Standard mode transmission speeds are 100 kbps, but some devices can achieve higher speeds. In Fig. 13, the timing diagram of this process is shown, highlighting the start and end moments of communication and the sending of packets during transmission.

In a circuit with multiple Masters, the Master who wants to initiate a communication needs to check the SDA signal. If it is at a low level, it is because another Master is in control. Then, it must wait for the SDA signal to return to a high level to transmit.

Fig. 12 I^2C communication structure

Table 6 Logic levels of actions on I^2C protocol

Action	Logic Level SDA	Logic Level SCL
Write	Low	High
Read	High	High
Ack	Low	High
Nack	High	High
Start condition	Transition High to Low	High
Stop condition	Transition Low to High	High

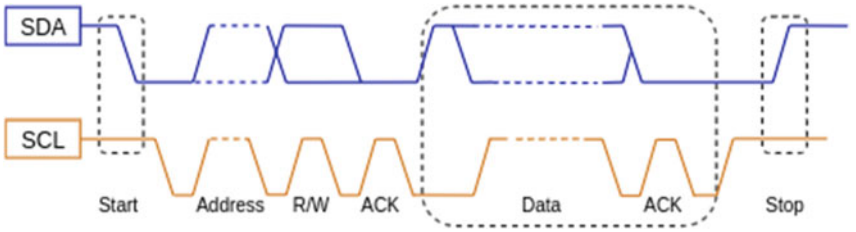


Fig. 13 I^2C timing diagram

The I^2C is widely used for communication between devices over short distances, and it is recommended that it be on the same circuit board, such as LCD and OLED displays. One of the main negative points of I^2C is that it uses addressing, running the risk of having two peripherals on the network with the same address, causing conflict.

5 Dataloggers

A Data Acquisition (DAQ) System comprehends the whole infrastructure employed to collect physical data of a varied set of sensors, which can be deployed on the ground, underwater, on satellites, etc. ([23, 27]). Sensors are attached to a datalogger, an essential piece of hardware in a DAQ system responsible for collecting data from sensors and storing the collected data in non-volatile memory. Stored data can be collected at the proper site where the datalogger is or, more common today, the datalogger can support telemetry that allows it to use communication interfaces to

send collected data to central databases for processing. Some versions also support remote control, which allows the datalogger to be remotely reconfigured.

Dataloggers are employed in many industries for monitoring manufacturing processes and pieces of equipment. A datalogger offers access to data in real-time, allowing precise control of the processes and timely decision making. In the first sector, dataloggers are deployed at farms to gather environmental data that will be used to support irrigation, thermal control of greenhouses, thermal comfort of animals and so on.

Dataloggers are also an auxiliary tool for academic research. Scientists from many diverse fields use dataloggers to collect experimental data. For example, in agriculture and earth sciences, dataloggers are deployed on remote sites to acquire field data. Air humidity, wind speed, soil humidity, and solar radiation are just a few instances of a wide range of physical variables that can be monitored. In the laboratory, dataloggers acquire and store data from accurate sensors in controlled experiments.

Next, we introduce the internal architecture of a datalogger (Sect. 5.1) and the main types of dataloggers (Sect. 5.2). Section 5.3 compares several commercial dataloggers according to their communication interfaces. Finally, Sect. 5.4 will provide some guidelines that one must consider when making its datalogger from scratch using development boards like the ones described in Sect. 2.

5.1 Datalogger Internal Components

The general architecture of a datalogger is shown in Fig. 14. The central component of a datalogger is the microcontroller that governs the whole operation of the equipment determining sensor readings, data storage, and remote communications. Two different pieces of software can be distinguished here: the Firmware and the Program. The former is the primary software of the datalogger, which gives the Program the programming primitives for interfacing with sensors, storing data, and transmitting and receiving data through the communication interfaces. The Firmware can be compared to the operating system in a personal computer. The Program is the software component that effectively orchestrates how a datalogger will behave, and the user adapts it according to their needs. The Firmware code changes slowly, only when the manufacturer corrects bugs and introduces new features. On the other hand, the Program is always updated when a new sensor is added, or new functions are required.

The sensors component are necessary components in the datalogger. Here we encompass various sensors with very different functioning principles and interfaces: temperature sensor, rain gauge, pyranometer, accelerometer, and others. For each sensor class, the datalogger uses specific circuitry to translate the read signals into the actual measurement of the physical variable. The type of interfaces supported and the features of the analog-to-digital converters embedded into the datalogger are essential aspects to be considered when deciding about different manufacturers and

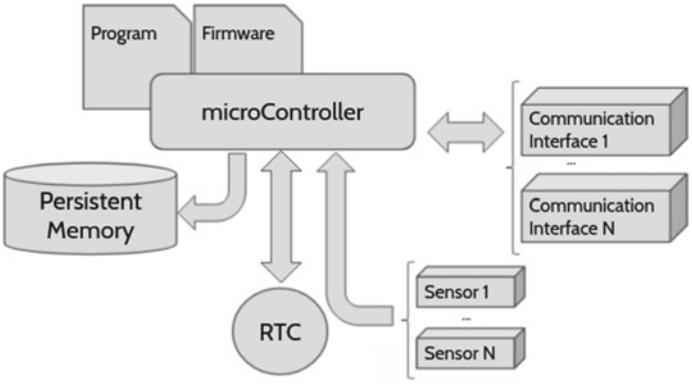


Fig. 14 Block diagram illustrating the main internal hardware and software components of a datalogger. Please note that this is a simplified view of a datalogger, since we do not show the internal buses, analog-to-digital converters and other important circuitry. Moreover, this can be seen as a minimum set of components, since dataloggers can have other components as actuators, displays etc.

models. There must be a correct match between the sensors intended to be used and the available interfaces at the datalogger.

The persistent memory component is a non-volatile memory for storing data. This component is essential, and its size is a characteristic to be considered when choosing a datalogger. Commonly, dataloggers make database rotation, i.e., they overwrite old data when the persistent memory is full. This way, the size of the persistent memory determines how long old data remain stored. If no telemetry is available, the memory size will determine how many times one must move to where the datalogger is installed to get the collected data. This component is essential even when the datalogger makes data telemetry. The persistent memory can act as a temporary buffer for storing data if the remote communication is down.

Another necessary component in a datalogger is the Real-Time Clock (RTC) circuitry. This hardware component measures the passage of time through a crystal oscillator. Thus an RTC is used to give precise timestamps for labeling sensor readings, which are needed for reconstructing the time series of collected data.

Finally, the communication interfaces are used to transmit and receive data to/from external devices or the Internet. Some interfaces are cabled serial interfaces (RS-232 and USB) for short-range communication. In contrast, other interfaces are cabled or wireless (as GPRS, 3G, 4G, or other) used for long-range Internet connection. It is also possible to connect modems through the datalogger's serial interfaces when it does not have built-in wireless communication or when the supported wireless communication technologies are not available at the site. More about the external communication aspect of dataloggers is discussed in Sect. 5.3.

5.2 Types of Dataloggers

There are many commercial dataloggers, and choosing the right one for a specific case depends on aspects such as size, robustness to environmental conditions, cost, support to solar panels, support to specific sensors, and its programming level. Figure 15 illustrates a commercial datalogger from the Campbell Scientific manufacturer.[13] At the top of the datalogger, one can find the sensors' input interfaces and the communication interfaces on the right side. The image intends to give a general idea of this equipment, whose external appearance differs from the disposition of the inputs, command buttons, and communication ports.

Dataloggers can be divided into classes. Despite previous proposals which look for the hardware perspective ([1]), with respect to the programming level, we can divide dataloggers into two broad categories: programmable dataloggers and configurable dataloggers.

Programmable dataloggers are general-purpose solutions that accept an array of sensors from their manufacturers. These dataloggers can be arranged with many different sensors making them flexible and indicated for different applications. They have just electrical interfaces which can be used to attach the sensors. The number and the types of the interfaces (Sect. 4), the memory size, the operational temperature, and humidity ranges are essential aspects to be observed when choosing a programmable datalogger.

Another important aspect, and the one that gives the name to this class, is that these dataloggers can be programmed in a high-level computer programming language like C or BASIC. However, manufacturer's specific programming languages are also common, including intuitive graphical programming languages. This code determines the frequency of sensor reading, data structures for storing data, and functions for telemetry. It allows adding to the datalogger complex behaviors as, for example, adjusting reading frequency according to specific events. This approach could save energy when instrumenting rare phenomena since sensing frequency could be maintained low most of the time and increased when the interest phenomena start.

The code of programmable dataloggers is written in a computer, cross-compiled, and then sent to the datalogger. It can be sent through a serial interface or, modernly, through an Internet-enabled communication interface. On the one hand, there is a steep learning curve for acquiring experience with programming these dataloggers. On the other hand, this comes with flexibility, allowing the data collection routines to the user's specific needs.

Differently, a configurable datalogger is a ready-to-use solution that comes with a limited set of sensors. These sensors are generally integrated with the datalogger in the same box. This way, the dataloggers are indicated for specific purposes. In a more detailed view, the configurable datalogger is distinguished from the programmable one since their Program and Firmware are coupled together, and the user cannot change the Program's code. Thus, only the frequency of sensor readings and other few aspects can be configured. These configurations can be made through buttons

[13] https://www.campbellsci.com/.

Fig. 15 Example of a commercial datalogger. The CR510 programmable datalogger is a basic solution from Campbell Scientific that has one digital input and four analog inputs, and it is programmed via the PC200W programming software

or other tactile interfaces or through the manufacturer's graphical software. Despite having many specific use cases, the usage of those dataloggers is a trivial task that does not require any programming skills.

5.3 Survey of Datalogger's Support for Communication Interfaces

Here we present a survey about the communication interfaces and support to protocols of 17 models of programmable dataloggers from 3 different manufacturers. This survey considered the main models of Campbell Scientific,[14] a world-leader manufacturer of instrumentation equipment, and also other popular manufacturers: Teracom[15] and Logic Beach. Please note that here we provide only general definitions of the protocols and communication interfaces. More information about the communication infrastructure of DAQ systems is available later in this book.

The selected datalogger models were analyzed in terms of the physical communication interfaces available, the application protocols supported, the supported formats for data transmission, and the SCADA (Supervisory Control and Data Acquisition) protocols implemented.

We focus on four standard physical interfaces: Ethernet, GSM/GPRS, RS-232, and USB. Those interfaces are used for external communication. The RS-232 and USB allow the datalogger to connect to computers and modems. In general, using a computer and software provided by the manufacturer, it is possible to acquire the collected data and send a new program to adjust its collection routines. In turn,

[14] https://logicbeach.com/.

[15] http://www.teracomsystems.com/.

the Ethernet (cabled) and GSM/GPRS (cellular) interfaces allow connection to the Internet, allowing easy integration of the datalogger to modern information systems.

Concerning the application protocols, we highlight the Network Time Protocol (NTP) that accesses Internet servers to retrieve the current hour and it is used for automatic definition of the internal RTC; the Simple Mail Transfer Protocol (SMTP) that offers support to sending (not receiving) collected data by e-mail; the File Transfer Protocol (FTP) that allows transferring the collected data for file servers; the Simple Network Management Protocol (SNMP) used for remote monitoring and management of dataloggers; and, at last, the Hypertext Transfer Protocol (HTTP) API (Application Programming Interface) that involves a wide range of Web-based application interfaces offered by dataloggers.

We highlight two data formats for data serialization: the JavaScript Object Notation (JSON) and the eXtensible Markup Language (XML). Both are popular methods used in Internet applications today.

Finally, we observed two SCADA protocols: Modbus, an open standard, and PakBus, a proprietary protocol from Campbell Scientific. Those standards are popular in the industry and are used for connecting devices to SCADA systems. Modbus and PakBus protocols mitigate heterogeneity problems for data acquisition and device control in industrial plants.

Table 7 shows the protocols and interfaces supported by six different series of Campbell Scientific's dataloggers. We surveyed devices currently supported by the manufacturer, and we highlight that we indicate only the technologies supported

Table 7 Protocol and interface support of Campbell Scientific's datalogger models

		CR300	CR200X	CR800	CR6	CR1000	CR3000
Interfaces	Ethernet	**Yes**	No	No*	**Yes**	No*	No*
	GSM/GPRS	No	No	No	No	No	No
	RS-232	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
	USB	**Yes**	No	No	**Yes**	No	No
Protocols	NTP	No	No	**Yes**	**Yes**	**Yes**	**Yes**
	SMTP	No	No	**Yes**	**Yes**	**Yes**	**Yes**
	FTP	**Yes**	No	**Yes**	**Yes**	**Yes**	**Yes**
	SNMP	No	No	**Yes**	**Yes**	**Yes**	**Yes**
	HTTP API	No	No	**Yes**	**Yes**	**Yes**	**Yes**
Data format	JSON	**Yes**	No	**Yes**	**Yes**	**Yes**	**Yes**
	XML	**Yes**	No	**Yes**	**Yes**	**Yes**	**Yes**
SCADA support	Modbus	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
	PakBus	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**

* Support through manufacturer's supplementary device

by the manufacturer without third-party upgrades or devices. We also indicated the case when the technology is supported using supplementary devices of the very own manufacturer.

In general, Campbell's dataloggers support to RS-232 interfaces is common to all dataloggers. It is common to find more than one RS-232 interface at a datalogger, allowing communication with multiple devices simultaneously. On the other hand, Internet-based interfaces are not commonly implemented in Campbell's dataloggers. Notably, none of these models implements GSM or GPRS natively, although third-party modems can be supported. The absence of a native communication channel is a drawback of these data loggers since they are used in the wilderness, obligating the acquirement of a third-party modem for remote communication.

Regarding the application protocols and data format supported, one can observe that the Ethernet-enabled dataloggers (or the ones that can support Ethernet through supplementary devices) also support a wide range of protocols and data formats. Finally, in terms of SCADA, we note complete support to both protocols, making these dataloggers the primary option for industrial applications that demand robust sensing and control.

Table 8 shows the protocols and interfaces supported by Teracom's datalogger models. When compared to Campbell Scientific, a distinguishing characteristic of these dataloggers is the native support to Internet connections. Almost all of these models have Ethernet interfaces, and the only one that does not have such support (the TCG120) uses a GSM/GPRS cellular interface. Moreover, it provides a USB interface for data collection when there is no cellular network available. Due to this design choice, these dataloggers do not use the popular RS-232 interfaces.

Table 8 Protocol and interface support of Teracom's datalogger models

		TCW112	TCW112B	TCW181B	TCG120	TCW241	TCW220
Interfaces	Ethernet	**Yes**	**Yes**	**Yes**	No	**Yes**	**Yes**
	GSM/GPRS	No	No	No	**Yes**	No	No
	RS-232	No	No	No	No	No	No
	USB	**Yes**	No	No	**Yes**	No	No
Protocols	NTP	No	No	No	**Yes**	**Yes**	**Yes**
	SMTP	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
	FTP	No	No	No	No	No	No
	SNMP	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
	HTTP API	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
Data format	JSON	No	No	No	No	**Yes**	**Yes**
	XML	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
SCADA support	Modbus	No	No	No	No	**Yes**	**Yes**
	PakBus	No	No	No	No	No	No

Table 9 Protocol and interface support of Logic Beach's datalogger models

		IL-10	IL-20	IL-80	IL-Mini
Interfaces	Ethernet	**Yes**	**Yes**	**Yes**	No
	GSM/GPRS	No	No	No	**Yes**
	RS-232	**Yes**	**Yes**	**Yes**	No
	USB	**Yes**	**Yes**	**Yes**	**Yes**
Protocols	NTP	No	No	No	No
	SMTP	**Yes**	**Yes**	**Yes**	**Yes**
	FTP	**Yes**	**Yes**	**Yes**	No
	SNMP	No	No	No	No
	HTTP API	No	No	No	No
Data format	JSON	No	No	No	No
	XML	No	No	No	No
SCADA support	Modbus	**Yes**	**Yes**	**Yes**	**Yes**
	PakBus	No	No	No	No

About the application protocols, three of the datalogger models support NTP, and all of them support SMTP and SNMP. Another distinguishing characteristic of these dataloggers is their HTTP APIs. These APIs enable data access and datalogger's control. About the SCADA protocols, the Modbus protocol is supported by two models. These dataloggers are indicated for industrial applications, and they are the only models that support both data formats: XML and JSON.

Table 9 shows the protocols and interfaces of the dataloggers models manufactured by Logic Beach. In this survey, these are the most connected dataloggers. The IL-10, IL-20, and IL-80 are variants of the same model. The IL-Mini is recommended for applications that demand low-power and low-cost dataloggers.

Whereas IL-10, IL-20, and IL-80 have Ethernet, RS-232, and USB communication interfaces, the IL-mini has only GSM/GPRS support, similar to Teracom's TCG120. All dataloggers support sending the collected data through SMTP, and just the IL-Mini does not support FTP. These Logic Beach's dataloggers do not support others communication protocols.

Concerning the data formats, no dataloggers support modern formats like JSON and XML. The telemetry of data is made through raw tabulated data files. Finally, the Modbus protocol is supported by all models, allowing easy integration of these dataloggers into industrial networks.

5.4 DIY Dataloggers

The commercial alternatives presented in Sect. 5.3 are indicated for instrumentation in a wide range of applications in different sectors. Due to its high cost, these solutions are interesting for setups with long-term requirements and whose return of investment can be made on years, as a factory or a specific-purpose laboratory. However, scientific research commonly demands versatile dataloggers that can be rearranged several times to adapt to new scientific projects, new research goals, and even experimental sensors. This way, low-cost alternatives to collect, store, and transmit data from many different sensors are needed.

The low-cost and open hardware boards presented at Sect. 2 are used today as viable alternatives for making scientific specific-purpose dataloggers [9]. Arduino, for example, is being employed in many different fields, as in agricultural science for livestock monitoring and soil humidity sensing and in neuro-engineering experiments for controlling micro electrical discharges [5, 15, 19, 19, 26]. Other boards like ESP8266 and ESP32 ([12, 14, 18]) are being used with similar purposes also. These studies show the usefulness of employing low-cost hardware for research.

However, choosing the right board or development kit for making a specific-purpose datalogger for scientific experimentation can be a very challenging task. Next, we provide some questions whose answers can guide the acquisition of the most suitable board.

Guidelines for designing your own datalogger

1. Which sensors do you need?
2. How much data will your datalogger store?
3. What are the embedded processing requirements of your project?
4. Does your datalogger need telemetry?
5. Does your datalogger need batteries?

The first question is the most important one. It says that before choosing a board or development kit, one needs to choose the sensors it needs. Please note that we should not have only a general idea of the sensors to be used. On the contrary, we should define the correct number and the specific model of each sensor to be used in our experiment. It is essential to read each chosen sensor's datasheet to know its hardware interfaces and power supply requirements. Both requirements will determine the hardware inputs that the board must have.

Table 10 shows a short comparison between Arduino, NodeMCU, and the CR1000, a commercial datalogger from Campbell Scientific. The main idea is to show the diversity in the hardware inputs (the I/O and the Output power lines in the table) of just a few options. As one can see, whereas NodeMCU has the lowest cost, it also provides a limited number of analog inputs (just one input), which may avoid using

Table 10 Comparison between two development boards and a commercial datalogger

	Arduino Uno	NodeMCU	CR1000
Power supply	7-12V	4.5-9V	9.6-16V
I/O	Ang: 6; Dg: 8	Dg: 9; Ang: 1	Ang: 16; Dg: 8; 4 RS232
Processor	16 MHz (ATMega328)	160 MHz (ESP8266)	7.3 MHz (Renesas H8S 2322)
Working memory	2KB	128KB	512KB
Persistent memory	32 KB	4 MB	4 MB
Output power	3.3–5V	3.3–5V	5V,12V
Price	Tens of USD	Less than ten USD	More than one thousand USD

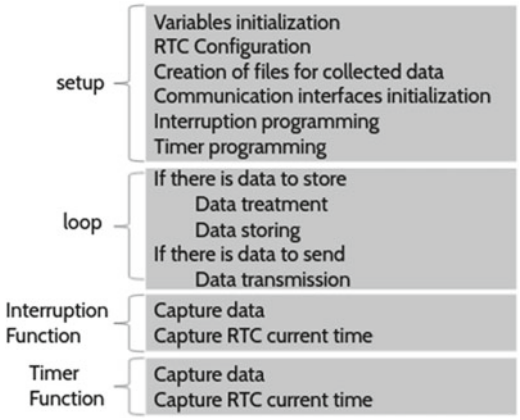


Fig. 16 Block diagram illustrating the structure of the code of a datalogger

it in some projects. Arduino, on the other hand, has more analog inputs, but its other characteristics must be taken into considerations also. These specifications are just the information that one must gather from sensors to choose the right board for a project.

The second and third questions are linked and must be answered together. They determine the computing characteristics of the underdevelopment datalogger. A general view of the code structure that a datalogger runs is shown in Fig. 16. This figure tries to give an idea of the code that runs on commercial or DIY dataloggers, but it does not show details of the code.

The program starts up by running the setup phase. In such a phase, variables, files, and communication interfaces are initialized, and the RTC is configured. Timers and interruptions are also programmed. Timers are used for periodical data collection by polling, and interruptions are used for collecting data of unexpected events.

Each timer and interruption has an associated function code, which is called when the respective event occurs. These functions have, commonly, just a few instructions

for capturing data from the respective sensor and the current time of collection. This way, the datalogger can treat more events without losing data since all tasks for saving or sending data are made in bulk in the loop section. Please note that some flags indicate to the loop function that there is some data to be sent or saved.

In the loop function also there is some code for treating the data. It is an essential advantage of using an open platform for designing a datalogger. Many different treatment methods can be applied to the data. Whereas commercial platforms can restrict support to simple functions like average, minimum, or maximum, the development boards provide access to a general-purpose programming language, allowing the implementation of sophisticated approaches using statistical methods and machine learning. However, the designer of the datalogger must look carefully if the processor and the working memory are appropriate for the intended computing task.

Questions 2 and 3 also are associated with persistent memory. Based on the frequency of data storing, the size of the data, and the frequency that data is sent through telemetry or gathered in loco, the designer can estimate the persistent memory needed and compare its estimate to the actual size of the persistent memory of each available board. Please note that Table 10 specifies only boards' internal memory. It is common for these boards to support extra external memory through modules, which must be considered.

The last two questions (4 and 5) are important to determine the communication capabilities and the power supply alternatives supported by the boards. Some boards have native communication interfaces. NodeMCU, for example, has native support to 802.11 communication, which can be interesting when the datalogger will be installed in a lab where this technology is widely available, i.e., it is indicated for short-range communication. Similar to persistent memory, capabilities for different communication interfaces can be added to open platforms through external modules. Please refer to Sect. 5.3 for more about communication interfaces. Similarly, the power supply of these boards will guide the choice by the specific batteries and power circuitry, if necessary.

Finally, other aspects can drive the design of a specific-purpose datalogger for scientific research, such as the board dimensions, the team's experience with one board or other, the need for control capabilities, etc. Please note that system designers must carefully consider each of these aspects if it is the case. To illustrate, let us imagine the last-mentioned aspect, control capabilities. It opens entirely new questions, such as the availability of digital/analog converters, power supply issues, circuitry to protect the board from overcurrent, and other questions out of this chapter's scope.

6 Further Advances

The Internet of Things brings opportunities for applying sensors in various economic sectors, allowing companies to better monitor and control their processes. The state-of-the-art on sensing technology is moving faster towards more innovative solutions.

Concepts as Edge Computing and Fog Computing have emerged in the last years to point how new cloud services must go into the ICT infrastructure. In a certain way, the cloud is being distributed to the edge of the Internet [11] and bringing a new range of cloud-based pay-as-you-go solutions as Sensing as a Service [22] and Control as a Service [10].

These concepts have a common view: new applications for the Industry 4.0 and 5.0 era have stringent performance, availability, and legal constraints, demanding the new services be hosted closer to end-users. The same occurs in the case of industry, in which the more and more refined processes control have similar requirements.

These demands are pushing the instrumentation industry to bring solutions with more computing capacity to support Machine Learning and other advanced computing techniques embedded directly into the sensing infrastructure. For example, deep learning architectures can autonomously analyze and fuse data from different sensors to detect anomalous behavior of industrial equipment. Embedding these architectures closer to the equipment enables fast anomalies detection, saving costs and lives [16].

On the other hand, orthogonal to the research on more powerful computing embedded systems, the industry of instrumentation have put significant effort to create ultra-low-power devices which are demanded by applications on unpowered or remote sites [6]. Deploying sensors on the field for agriculture or livestock management, for example, is a challenging task since they have to support long-term batteries, or they must have the capacity for energy harvesting for power renewal. Such requirements demand clever strategies and circuitry for reduced power consumption.

Acknowledgements This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior—Brasil (CAPES)—Finance Code 001.

References

1. Abdallah M, Elkeelany O (2009) A survey on data acquisition systems daq. In: 2009 international conference on computing, engineering and information. IEEE, pp 240–243
2. Almada-Lobo F (2015) The industry 4.0 revolution and the future of manufacturing execution systems (mes). J Innov Manag 3(4):16–21
3. Arduino (2021a) About us. https://www.arduino.cc/en/Main/AboutUs. Accessed 06 July 2021
4. Arduino (2021b) Arduino uno. https://docs.arduino.cc/hardware/uno-rev3. Accessed 06 July 2021
5. Batista PHD, de Almeida GLP, Sarmento RM, Pandorfi H, de Melo AAS, Rolim MM, de Medeiros VWC, Gonçalves GE (2019) Monitoring the bovine activity in grazing by an electronic sensing device based on gps. Revista de Ciências Agrárias 42(2):332–339
6. Bernal E, Spiryagin M, Cole C (2020) Ultra-low power sensor node for on-board railway wagon monitoring. IEEE Sens J 20(24):15185–15192
7. Bjornholt JE (2016) Microchip technology completes atmel acquisition and provides update on its fiscal fourth quarter, 4p
8. Brodkin J (2013) Arduino creator explains why open source matters in hardware, too. https://arstechnica.com/information-technology/2013/10/arduino-creator-explains-why-open-source-matters-in-hardware-too/. Accessed 26 June 2021
9. Daniel KF, Peter JG (2012) Open-source hardware is a low-cost alternative for scientific instrumentation and research. Mod Instrum

10. de Freitas Bezerra D, de Medeiros VWC, Gonçalves GE (2021) Towards a control-as-a-service architecture for smart environments. Simul Model Pract Theory 107:102194
11. Endo PT, de Almeida Palhares AV, Pereira NN, Goncalves GE, Sadok D, Kelner J, Melander B, Mangs J-E (2011) Resource allocation for distributed cloud: concepts and research challenges. IEEE Netw 25(4):42–46
12. Firmansyah R, Widodo A, Romadhon A, Hudha M, Saputra P, Lestari N (2019) The prototype of infant incubator monitoring system based on the internet of things using nodemcu esp8266. J Phys Conf Ser 1171:012015 (IOP Publishing)
13. Gaillard F (2013) Microprocessor (MPU) or Microcontroller (MCU)? What factors should you consider when selecting the right processing device for your next design, pp 1–4
14. Ghosh D, Agrawal A, Prakash N, Goyal P (2018) Smart saline level monitoring system using esp32 and mqtt-s. In: 2018 IEEE 20th international conference on e-health networking, applications and services (Healthcom). IEEE, pp 1–5
15. Koenka IJ, Sáiz J, Hauser PC (2014) Instrumentino: an open-source modular python framework for controlling arduino based experimental instruments. Comput Phys Commun 185(10):2724–2729
16. Li Z, Li J, Wang Y, Wang K (2019) A deep learning approach for anomaly detection based on sae and lstm in mechanical equipment. Int J Adv Manuf Technol 103(1):499–510
17. Mao F, Khamis K, Krause S, Clark J, Hannah D (2019) Low-cost environmental sensor networks: Recent advances and future directions. Front Earth Sci 7:221
18. Métrologie IM (2019) Water level detection system based on ultrasonic sensors hc-sr04 and esp8266-12 modules with telegram and buzzer communication media. J Homepage. http://iieta.org/journals/i2m 18(3):305–309
19. Sanders JI, Kepecs A (2014) A low-cost programmable pulse generator for physiology and behavior. Front Neuroeng 7:43
20. Saqib M, Ali Khan S, Mutee Ur Rehman HM, Yang Y, Kim S, Rehman MM, Young Kim W (2021) High-performance humidity sensor based on the graphene flower/zinc oxide composite. Nanomaterials 11(1):242
21. Tewari N, Deepak N, Joshi M, Bhatt JS (2021) Comparative study of IoT development boards in 2021: choosing right hardware for IoT projects. In: 2021 2nd international conference on intelligent engineering and management (ICIEM). IEEE, London, United Kingdom, pp 357–361
22. Verçosa NJ, Gonçalves GE, de Medeiros VWC (2018) Sensor selection model and algorithms based on user requirements for sensing as a service. Revista de Exatas e TECnológicas 7(1):20–29
23. Weber SK, Miotto GL, Almeida J, Blanc PH, Dias A, Malaguti G, Manninen HE, Pfeifer J, Ravat S, Onnela A et al (2020) Data acquisition system of the cloud experiment at cern. IEEE Trans Instrum Meas 70:1–13
24. Wehde M (2019) Healthcare 4.0. IEEE Eng Manag Rev 47(3):24–28
25. White E (2012) Making embedded systems: design patterns for great software, 1st edn. OCLC, O'Reilly, Beijing, 815881891p
26. Wickert AD, Sandell CT, Schulz B, Ng G-HC (2019) Open-source arduino-compatible data loggers designed for field research. Hydrol Earth Syst Sci 23(4):2065–2076
27. Wu H, Li Z, Tan H, Hua H, Li J, Hennig W, Warburton W, Luo D, Wang X, Li X et al (2020) A general-purpose digital data acquisition system (gddaq) at peking university. Nuclear Instrum Methods Phys Res Sect A Accelerators Spectrometers Detectors Associated Equipment 975:164200
28. Xiao P (2018) Designing embedded systems and the internet of things (IoT) with the ARM® Mbed™. Wiley
29. Zhai Z, Martínez JF, Beltran V, Martínez NL (2020) Decision support systems for agriculture 4.0: Survey and challenges. Comput Electr Agric 170:105256

Chapter 3
FPGA-Based Error Correction in MEMS Sensors: Case Study of Respiration Monitoring System

Idir Mellal, Youcef Fouzar, Laghrouche Mourad, and Jumana Boussey

Abstract Microelectromechanical System (MEMS) sensors are an essential branch of the microelectronic industry. They have been ubiquitously used in all live domains and applications, starting from general use with low precision and fewer constraints to high accuracy and harshest environments such as health and space applications. Thus, researchers have been investigating MEMS sensors in different aspects. One of the hottest aspects is related to the study and analysis of the measurement error in order to correct it. This work investigated and analyzed the sources and origins of different MEMS sensor errors and classified them into different categories. Moreover, various methods used in literature to correct the introduced errors and compensate the output drift have been explored. A practical example has been presented with a MEMS sensors applied to monitor human respiration activity. We designed a spirometry system to assess the respiration activity with a MEMS flow sensor and a 3D accelerometer placed on the chest. We modeled the output drift and proposed a compensation model based on Artificial Neural Network (ANN). Then we implemented the system on a Field Programmable Gate Array (FPGA) board. Different tests and measurements have been carried out. Therefore, different types of MEMS sensor errors have been presented and classified into random and deterministic errors. Furthermore, various parameters intervening at different stages and under different time/space constraints have been discussed. Moreover, the solutions used in the literature to correct the MEMS sensor outputs have been presented and compared. Finally, we presented an FPGA-based respiratory monitoring system using two MEMS sensors, a flow sensor, and an accelerometer. An ANN model has been used to compensate for the drift of the flow sensor. The performed tests on the nasal cannula showed a correlation between the analog and the digital outputs. This data is used to study the respiration rate. The accelerometer data collected on the chest

I. Mellal (✉) · L. Mourad
LAMPA Laboratory, Mouloud Mammeri University, Tizi Ouzou, Algeria
e-mail: idir.mellal@gmail.com

Y. Fouzar
University of Quebec in Outaouais, Gatineau, QC, Canada

J. Boussey
IMEP-LAHC Laboratory, Minatec, Grenoble, France

M. Chaturvedi et al. (eds.), *Recent Advancements in ICT Infrastructure and Applications*,
Studies in Infrastructure and Control, https://doi.org/10.1007/978-981-19-2374-6_3

has been used to detect chest movement and compute the tilt angle to understand the respiration activity.

1 Introduction

Recent studies and previsions showed that the Microelectromechanical System (MEMS) industry is in permanent growth. As a result, it has become more important and ubiquitous in our daily life. Consequently, MEMS reliability and accuracy have been more crucial and primordial for end-user health, safety, and security [1–4].

Different works have been done to study and investigate the reliability of MEMS systems, particularly MEMS sensors. Moreover, error analysis and modeling have emerged as an independent science using several software and hardware approaches to correct/calibrate the sensors' output. Researchers have used several statistical analysis tools, Artificial Neural Network (ANN) models, and different AI algorithms for modeling the MEMS sensor errors and different external impacts [5–8]. Medical devices are one of the most critical domains where the accuracy and safety of any Information and Communication Technologies (ICT) system should be ensured. For this reason, researchers and industrials continuously work hard to developing high accuracy and robust MEMS sensor-based systems for medical applications [8].

One of the person's most significant vital activities is respiration, a complex process of movement of oxygen from the outside environment to the cells and removing carbon dioxide outside the body. External respiration is known as inhalation and releasing air to the atmosphere is called exhalation. For this reason, the development of a continuous and automatic real-time monitoring system has been a high necessity for healthcare practitioners, particularly when it comes to monitoring patients with respiratory disorders. Furthermore, providing simultaneous and real-time information data of respiration and motion activities may be advantageous in the management of chronic illnesses such as chronic obstructive pulmonary disease (COPD) and improve the quality of life of millions around the world [9–11]. Hence, researchers and industrials developed and continue proposing various real-time monitoring systems with different approaches [10, 12–16]. Pernice et al. [17] designed an embedded portable and noninvasive electronic multisensor system capable of measuring different physiological signals, such as electrocardiographic (ECG), photoplethysmographic (PPG), and breathing signals. The experimental results showed a high-quality signal with high resolution, which gives a promising improvement in the accuracy of the extraction of critical cardiovascular parameters. They performed several analyses to extract essential features for each signal. The results showed the potential of the proposed system as a wearable system for various situations (apnea, respiratory monitoring). Pan et al. [18] designed a smart mask capable of monitoring different physiological parameters: heart rate, blood oxygen saturation, blood pressure, and body temperature. They proposed to use this integrated mask for motoring respiratory diseases such as COVID-19 patients. In addition, they used a wireless communication protocol and noncontact sensing to

minimize the risk of attracting the virus. Binu et al. [19] developed a wireless monitoring system capable of measuring the airflow using a thermal flow sensor; body postures using a triaxial accelerometer; and oxygen saturation using a photoelectric sensor to measure the airflow, body posture, and oxygen saturation. They developed algorithms for processing the data and making decisions. In 2014, Zhu et al. [20] designed a Body Sensor Networks (BSN) by integrating three different sensors to collect several vital signs for real-time monitoring of obstructive sleep apnea patients. A hot film flow sensor, accelerometer, and oximeter have been used to detect airflow, body postures, and blood oxygen saturation, respectively.

The continuous improvements of MEMS technology have a positive impact on the medical device industry. Indeed, MEMS devices are ubiquitous in any medical system or wearable device. For example, the hot wire anemometer is one of the widespread airflow MEMS sensors used for monitoring respiration flow. We can find several designed miniatured hot wire sensors using MEMS technologies [21–23]. The main disadvantage of this sensor is that its properties are highly dependent on the ambient temperature, requiring a compensation model to correct the output. Different types of temperature compensation methods for silicon sensors exist and can be divided into software, hardware, and hybrid. Moreover, different techniques and algorithms are used for error correction and temperature compensation in MEMS sensors, such as the Artificial Neural Network (ANN) [24, 25]. Furthermore, the hardware implementations of these algorithms are far more efficient and robust and might achieve a higher performance/power consumption ratio compared to the software methods [26–29]. Analog or digital signal conditioning circuits are required and should be designed within the circuit, which costs a significant amount of the available resources and area. Another robust and efficient approach is the use of the Application-Specific Integrated Circuit (ASIC) which is very efficient in terms of performance but very costly and requires a longer development period and more power consumption. Moreover, ASIC implementation lacks reconfigurability which limits its large deployment [26, 30]. In addition, DSP-based systems are not ideal for mimicking the simultaneous action of neurons due to their sequential character. Finally, to achieve compensation, the hybrid method employs both approaches.

In this chapter, we investigated MEMS sensor errors and explored few correction and compensation methods. We showed a MEMS flow sensor case study used to design a real-time system for monitoring human respiration activity. Finally, we presented a spirometry system using a MEMS flow sensor and a three-axis accelerometer to measure the nasal airflow and chest movement.

Temperature compensation of the MEMS flow sensor was performed using an ANN. The neurons in the hidden layer of the ANN model use the sigmoid activation function, which has been computed with the COordinate Rotation DIgital Computer (CORDIC) algorithm [31, 32]. The CORDIC algorithm proposed by Volder [33] is a simple and efficient approximation method to implement several mathematic functions, mainly trigonometric functions, through coordinate transformation [34–36]. Since its apparition in 1959, it has been widely introduced in different applications due to its performance and low cost compared to other methods. In 1971, Walther [34] proposed a unified and generalized CORDIC algorithm capable of computing

more functions, including logarithms, exponentials, and square roots [37]. On the other side, the three-axis accelerometer is intended to assess physical activity and measures x-, y-, and z-coordinates to use them for calculating the tilt angle. Thus, the tilt angle was computed using a 2D CORDIC-based algorithm employing the inverse tangent function to reduce the computational complexity and the used resources. Consequently, the implemented system showed high performance in less time while maintaining the high accuracy of the implementation [31]. More technical details and explanations are discussed in the case study section.

The rest of the chapter is divided as follows. Section 2 discusses the MEMS sensor technology and investigates different sources of errors and correction methods. The impact of MEMS sensors in ICT systems is also discussed in this section. FPGA technology is introduced in Sect. 3 and the proposed architecture has been presented. The case study of a monitoring respiration system has been shown in Sect. 4. Results were presented and discussed. Section 5 concludes this chapter and gives some potential use of the proposed system.

2 MEMS Sensors

MEMS technology has a huge impact on our life. MEMS products have been introduced in many fields and applications. This section discusses the MEMS sensor technology and explores reliability and performance of the MEMS sensors. Finally, the role of the MEMS sensors on ICT systems and how they have changed the ICT systems are also discussed.

2.1 Overview of MEMS Sensor Technology

Microelectromechanical system (MEMS) technology has a significant impact on human life. It has been introduced in every single life aspect. Sensor industry and manufacturing is an attractive area where research and industry investing astronomic amounts in improving the quality of sensors and developing more accurate, robust, and affordable sensors for various applications, especially for the healthcare industry. The MEMS industry has enormous advantages and benefits, from low cost due to mass production to high accuracy and small size, making MEMS sensors the best candidate for medical applications. However, MEMS sensors suffer from various failure mechanisms and limitations. Moreover, MEMS devices might be exposed to several defaults and errors because of several reasons related to their nature and production process, such as their size. We can mention some of the most pertinent failures involving the polysilicon's physical properties, such as stiction and other surface interaction phenomena, static and dynamic pull-in, and cracking. All these phenomena conduct to some temporary malfunctioning and failure of the MEMS

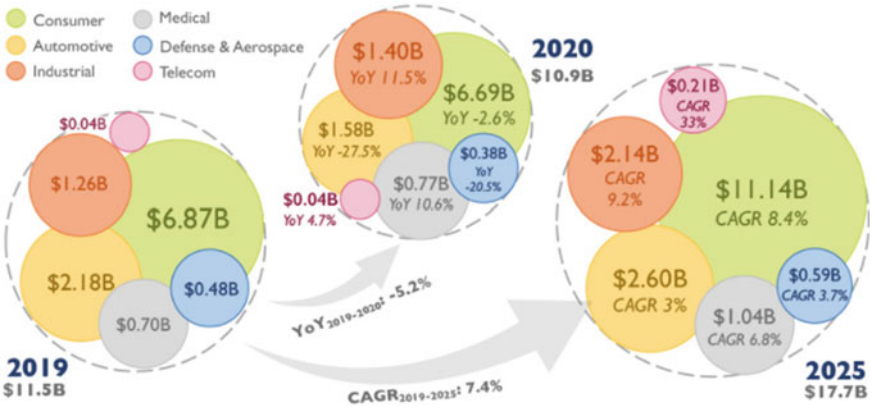


Fig. 1 2019–2025 MEMS market forecasts by end market [1]

sensors. Thus, they must be appropriately considered into account during reliability analysis [3, 38].

Furthermore, the impact of the external environment, such as temperature and moisture, plays an important role in defining the quality of the output signal and reliability of the device [39]. We designed a respiration monitoring system based on a MEMS flow sensor and a three-axis accelerometer in this work. The design of the flow sensor was performed and manufactured. The fabrication process of the MEMS flow sensor was described previously in different published manuscripts [3, 21–23, 30, 40, 41]. Many problems may be introduced and encountered during each step of the fabrication process. Thus, each of these problems can eventually provoke failure or defaults in the final device. These defects are mainly generated after the release of the hot wire. Also, many failure modes of MEMS sensors are introduced into the manufacturing process.

In recent 2020 reports [1], global MEMS sales are expected to increase from $11.5 billion in 2019 to $17.7 billion in 2025. Consumer electronics will continue to be the largest market for MEMS, accounting for roughly 60% of overall sales, with automobiles accounting for less than 20% of total sales. MEMS sensors will be an essential element for the wearable industry by providing engineers with highly sensitive and low-cost sensors for different devices based on the previsions and MEMS industry trends. Figure 1 shows the worldwide MEMS market.

2.2 The Used MEMS Sensors

a. Flow Sensor

The flow hot wire sensor has been designed using MEMS technology, and different papers have been published [3, 21–23, 40]. Figure 2 depicts the architecture of the

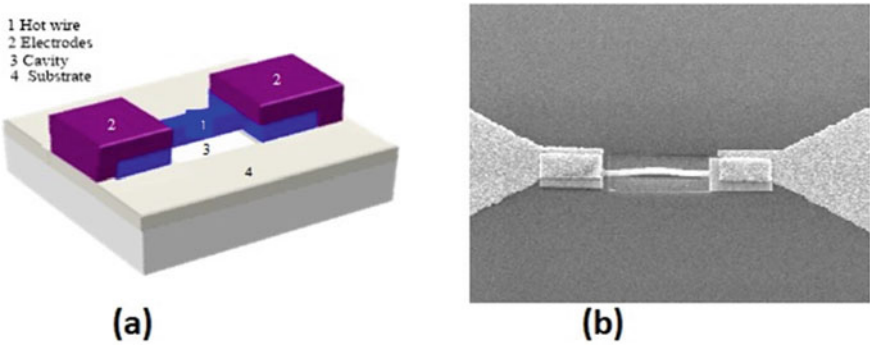


Fig. 2 **a** Architecture of the hot wire sensor, **b** Picture of the sensing element [3, 40]

developed sensor. The fabrication process might be summarized in different steps. To begin, silicon nitride was deposited to a thickness of 0.3 m using PECVD (plasma-enhanced chemical vapor deposition) on p-type 4″ wafers. Next, the silicon nitride sheet was patterned, and silicon was wet thermally oxidized through the nitride window. To get a smooth surface, the oxidation time was tweaked using technical step modeling. After that, polycrystalline silicon was deposited (LPCVD—thickness 0.5 m) and doped with boron ion implantation. After heat annealing, polycrystalline silicon was formed into variable section wire. Finally, electrical connections were made using chromium pads. After dicing the sensors, the wire was released by wet-etching the silica cavity.

Bensidhoum et al. [3] investigated the reliability of the used sensor. They reported a power consumption of less than 10 mW. They assumed that this sensor could be used as a temperature or airflow sensor (hot wire anemometer). The coefficient resistance temperature is about around 0.142%/°C.

Moreover, the authors performed several tests, including electrical overstress, thermal shock, and temperature cycling tests. Finally, an aging test was performed to explore the sensor behavior over time and prevent the eventual failure of the designed sensor.

Temperature is one of the most important factors impacting MEMS devices. Usually, we use the TCR (temperature coefficient of resistivity) to define the operating range of the MEMS device. The polycrystalline material comprises Ng grains of average size D and the grain boundaries with a width equal to δ. Equation (1) describes the polysilicon resistivity, which contains three elements: the result of the potential barrier ρ_B, the crystallite's bulk resistivity, and the grain boundary. The resistivity of polysilicon is [3, 21, 41]

$$\rho = \rho_B\left(\frac{\delta}{D}\right) + \rho_C\left(1 - \frac{\delta}{D}\right) \tag{1}$$

Equation (2) defines the TCR (temperature coefficient of resistivity):

$$\alpha = \frac{1}{\rho_0}\frac{\partial \rho}{\partial T} \tag{2}$$

The nominal working conditions and temperature are reached with a current of 2 mA and a voltage of 2 to 5 V where the temperature reaches 100 °C.

The temperature coefficient is shown to be independent of the geometry of the hot wire resistor after numerous experiments were conducted. Plotting the hot wire resistance versus temperature at several places of the two-point probe and measuring the slope of this curve is used to calculate the temperature coefficient. The same values are obtained at any region on the upper surface of the sensors under the same experimental conditions. Figure 3 depicts TCR experimental results.

b. 3-Axis Accelerometer

Nowadays, Inertial Measurement Unit (IMU) MEMS sensor is one of the most popular sensors. Usually, it measures angular rate, force, and sometimes magnetic field. A large variety of IMU sensors have been used widely in industry and research to develop various industries, including health, automobile, and robotic. For example, IMUs have many applications and uses in the healthcare industry and have been used as a motion sensor to assess physical activity, evaluate the respiratory rate, and follow the tremor for Parkinson's disease [42]. In this work, we used a commercial-grade accelerometer to follow the chest movement to find the tilt angle of the chest. First, we selected three-axis digital accelerometer sensor, BMA180, presented in Fig. 4. Then, we used the vectoring CORDIC module, implemented on a DE2 FPGA board, connected to the accelerometer outputs data to compute the tilt angle. The CORDIC module performs several rotations as an iterative step, micro-rotation noted θ_i, in a way to approximate the angle θ by a set of n iterations, where n is the number of iterations. It is important to mention that the larger the "n", the higher the precision and the higher the CORDIC module's hardware cost. Usually, the classical rectangular (x, y, z) to spherical (ρ, θ, φ) conversion is used [40, 43, 44].

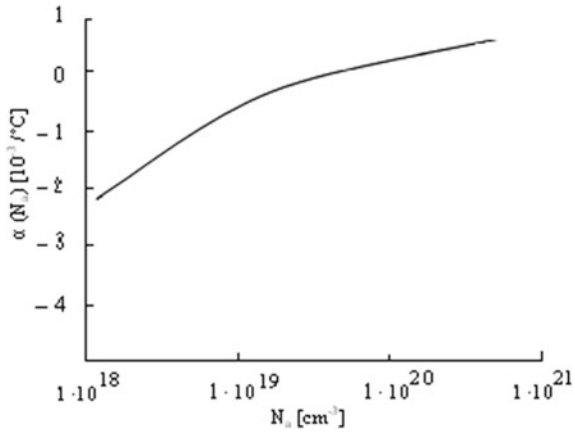


Fig. 3 TCR variation with doping Na

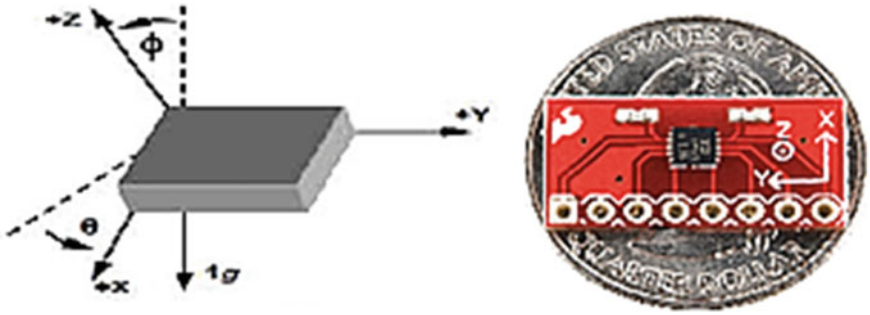


Fig. 4 Size and representation of the used IMU sensor, BMA180 [44]

2.3 Errors in MEMS sensors and Compensation Methods

Systems based on Microelectromechanical Systems (MEMS) technology are used in many fields such as aerospace, robotics, marine navigation, and biomedical. The large production and small size of these systems give them a significant place in the industry market. However, MEMS sensors are sensitive to the various interactions generated by the measurement environment and the forces that operate these sensors, which systematically cause measurement errors.

The sensor is the first electrical data source in the measurement chain that reflects the measurement's variation. Noise acts on the sensor's output and ends up modifying the electrical signal, like a quantity added to the useful signal [45]. It is evident that in order to give a measurement closer to reality or to stabilize a system using MEMS sensors, it would be necessary to first go through a thorough investigation of the origin of the errors [46]. Noise in sensors is the first element affecting the output signal, whose influence is sometimes misinterpreted. The two energies that interact in MEMS systems are electrical and mechanical energies, but other energies depend on the transduction mechanism. The interaction between electrical and mechanical energy gives rise to an electromechanical coupling force that generates noise [2, 47]. Therefore, it is interesting to consider noise in MEMS sensors depending on the energy.

The noise source is either extrinsic, which means that it is in the sensor's environment, or intrinsic, in which case the noise is related to the internal characteristics of the sensor. The changes in the internal characteristics of MEMS appear from the design, which differs from one manufacturer to another. In addition, other factors alter the characteristics of MEMS sensors; these are the failure mechanisms that occur during the manufacturing process, which is quite complex and requires rigorous monitoring. Otherwise, it will give rise to defects in the structure.

The influencing parameters of the sensors are all the physical parameters, other than the measurand, that disturb the output. There are two types of influence factors: interfering factors and modifying factors. The first intentionally disturbs the output by additive disturbance such as electromagnetic waves, vibrations, nuclear radiation,

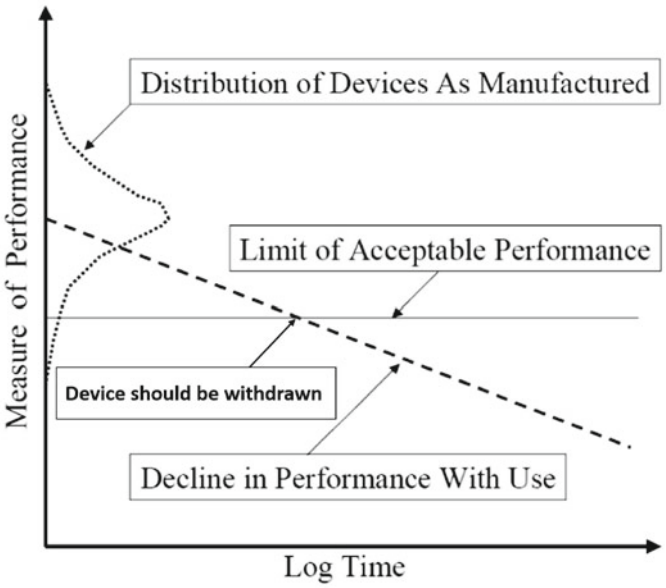


Fig. 5 Depiction of the relationship between reliability and quality of MEMS devices. We can see the line separating the acceptable performance and the rejected performance [2]

etc. The second one modifies the transfer function by a temporary or permanent alteration: temperature, humidity, and supply voltage. Thus, we can divide these errors into two categories as follows:

1. Systematic errors are clearly identified and predictable errors. They can be introduced by different sources, during the manufacturing and design process, during the packaging process, or in the electronic part of the sensor.
2. Random errors are all errors produced by a random phenomenon related to the measurement environment or the sensor characteristics, resulting in an error in the output value (see Fig. 5).

2.4 Sensing Layer in ICT

The sensing layer is the first layer in any ICT architecture. It consists of different sensor nodes collecting local data related to different activities and environment variations. In contrast with ordinary sensors, an ICT sensor is equipped with further modules with computations, processing, communications, and other intelligent functions (e.g., encryption). These modules ensure connectedness and secured communication to the ICT network.

In the proposed system, we proposed a smart wearable and wireless system for continuous monitoring of respiration activity. We used two MEMS sensors.

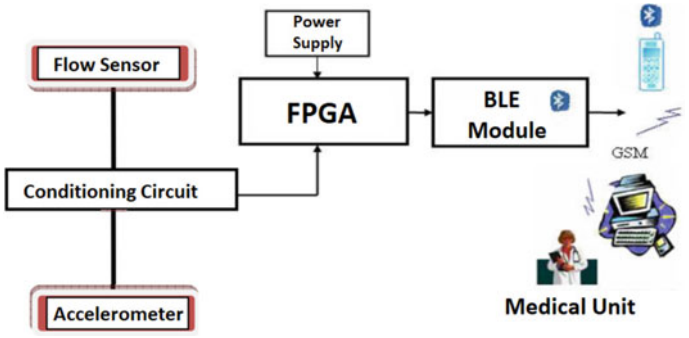


Fig. 6 Medical IoT for respiration monitoring system using a MEMS flow sensor and a three-axis accelerometer

One miniatured MEMS flow sensor and a commercial three-axis accelerometer (BMA180). We used an FPGA to implement the smart modules: correcting the errors, compute the tilt angle, and establishing the communication, as shown in Fig. 6. The flow sensor and the accelerometer measure the airflow and the acceleration data, respectively, then the data is sent to the conditioning circuit. The FPGA contains the cores implementing the computation and correction algorithms. The Bluetooth Low Energy (BLE) module ensures the transmission of the data to the medical unit.

In the next section, the FPGA technology will be discussed. We will explore its advantages and explain its differences compared to other technologies. The used FPGA will also be presented.

3 FPGA Review

An introduction to FPGA technology will be presented, and its opportunities will be discussed. Two different FPGA boards, from Xilinx and Intel (Altera), will be shown.

3.1 What is an FPGA?

a. FPGA definition

FPGA is a programmable device that can emulate any digital circuit and reproduce its behavior. FPGAs have the ability to reprogram them as much we want without any perturbation or loss of the device resources. They usually come within a board with several Input/Output (I/O) interfaces and extended modules such as wireless modules, memories, and specific processors [48].

Digital implementations offer valuable advantages in terms of precision of computing, parallelization, and available development tools. As described previously, we chose the second method, digital implementation, and selected an FPGA (field programmable gate array) device. Since his first apparition, this digital device has been used to implement the ANN and other mathematical tools like genetic algorithms, trigonometric functions, and modulator/demodulator. FPGA gives many advantages for hardware engineers. It is an array of logic cells that communicate internally through programmable wires and externally with I/O ports via routing channels. The wire resources are a metal segment with various lengths that run in horizontal and vertical columns with programmable switches. ASIC and Very Large-Scale of Integration (VLSI) offer the possibility to design parallel components, but it is expensive in time and cost. FPGA technology offers several advantages and opportunities such as the possibility of implementing digital systems with full parallelism, low cost, and high flexibility [25, 40]. Moreover, FPGA development is suitable for large and complex digital systems due to its

- Low cost compared to ASIC;
- Short development time;
- Density and capacity to increase more and more in a chip and use all available gates;
- Ease to use with high performance;
- Simplicity of the design tools to do all the work (place and route); and
- Big possibility to reprogram.

We can find several FPGA providers in the market. Xilinx and Intel Altera are the two big players. They offer a large variety of FPGA boards for different applications and with various interfaces. Each fabricant has its own development and simulation tools. For example, Xilinx proposes ISE/VIVADO for its FPGA boards, and Intel uses Quartus. The general internal architecture for any FPGA is composed of different Configurable Logic Block (CLB), which are the logic element of the circuit, In/Out Block (IOB), and the wiring network, which establishes the connections between different CLB, IOB, and other modules [25] as shown in Fig. 7.

The FPGA's ability and immense flexibility to build large and complex Systems on Chip (SoC) have attracted many engineers, designers, and developers to explore and use FPGAs. As a result, FPGAs have been used for many applications such as Digital Signal Processing, Control, Encryption, Image Processing, Artificial intelligence, networking, Big Data, Circuit Emulation, Smart Cities, Smart Vehicles, and much more [49, 50]. Figure 7 shows the Xilinx ZCU 111 FPGA board with different interfaces and available resources.

As a result of FPGA's high performances and abilities, FPGA technology has been introduced in different CIT and IoT systems in various fields, especially in health care [52–55].

b. Presentation of the Used FPGA: Intel (Altera) DE2 Development Board

The Intel DE2 board is an FPGA development board based on a Cyclone II FPGA chip. Figure 8 shows the DE2 board with different available interfaces connected to

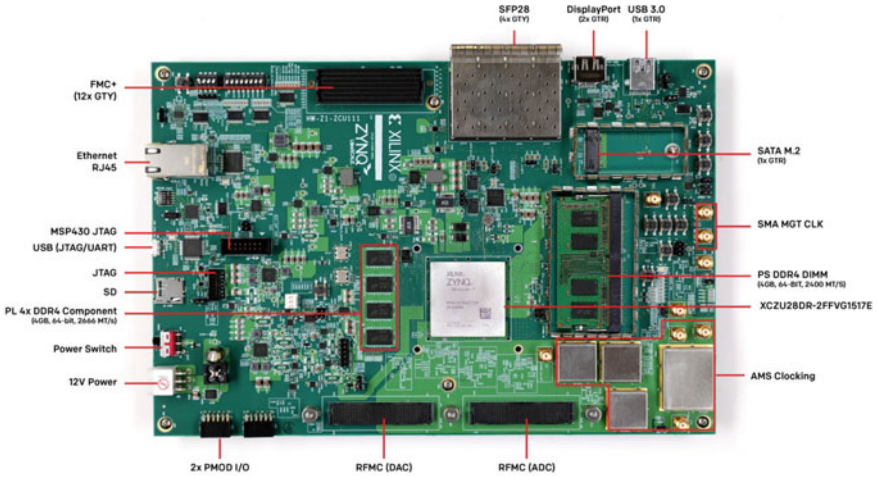


Fig. 7 Xilinx ZCU 111 FPGA board showing different I/O ports and several resources [51]

a motherboard and an accelerometer sensor. Intel offers different tools to program its wide range of FPGA products, such as Quartus, SOPC Builder, and Nios IDE [56].

The architecture of the proposed respiration monitoring system will be presented in the next section. The used modules and algorithms will also be discussed.

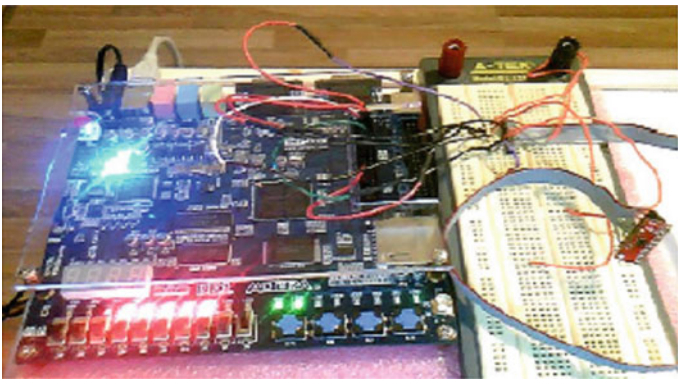
3.2 Proposed Architecture

Different modules of the proposed architecture for the respiration monitoring system will be discussed and presented. In addition, the control blocks and the data types will also be discussed.

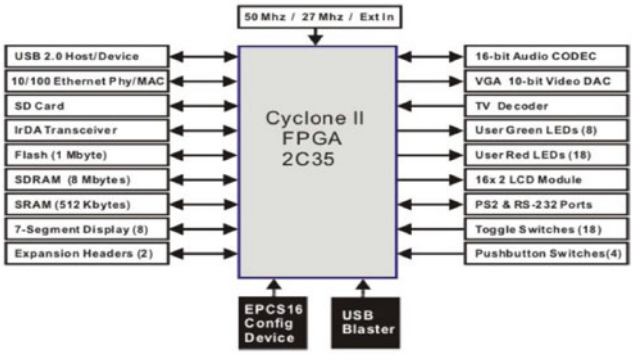
a. System Design

The flow sensor, with its multiple components ranging from signal conditioning to output correction, is included in our system, as is the accelerometer that uses the CORDIC core to compute the tilt angle and display the acceleration data. Implementing the proposed technique for temperature correction and computing the tilt angle on a target reconfigurable platform based on an FPGA device was used to test and evaluate the designed system. First, we used VHDL language to describe different design modules and then performed the implementation on the DE2 FPGA.

The proposed system is presented in Fig. 9. The spirometer, a medical device used to assess and evaluate respiration activity, uses a flow sensor to measure the airflow during inhalation and exhalation of a patient. It is used to diagnose few diseases like asthma, chronic obstructive pulmonary disease (COPD), and other breathing disorders. The ANN model is used to compensate for the temperature drift of the MEMS sensor. The accelerometer measures the acceleration coordinates (x, y z)



(a)



(b)

Fig. 8 DE2 Development Board: **a** picture of the board with an accelerometer, **b** Schematic of the board with the peripherals and FPGA chip [57]

used to compute the tilt angle θ. The output signal filtering and conditioning were performed before sending the data to the CORDIC module to compute θ.

b. Data Type

We used the 2's complement Fixed Point Format to represent the data and compute the outputs. First, we selected a data length of 12 bits with a Signed Fixed Point (SFP) s2.9 representation where one sign bit (Sn), 2 integer bits (I0 to I1), and 9 fractional bits (f0 to f7) are considered, as shown in Table 1.

Table 2 shows two different signed and fractional numbers, one positive and the other negative.

Fixed-point data is used to perform an efficient implementation, as described previously. On the other hand, the accelerometer output is 8 bits, and it is necessary to convert it to 12 bits. As a result, we employ a data converter. Additionally, a sign extension is carried out (the MSB bit). The following examples show how to convert 8-bit words to 12-bit words:

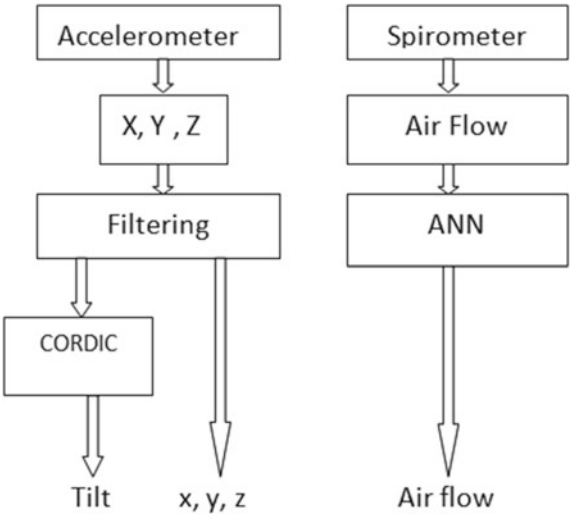


Fig. 9 Different blocks of the system

Table 1 Data structure

Sign	I1	I0	F0	……..	F8

Sign bit — Integer part — Radix point — Fractional part

Table 2 Number's representation

Format	Binary	Hex	Decimal
S2.9	000,011,101,101	0ED	0.4636
S2.9	111,100,000,110	F06	0.2449

$$X8 = 0xxx\ xxxx = = = = > X12 = 0000\ 0xxx\ xxxx.$$

$$X8 = 1xxx\ xxxx = = = = > X12 = 1111\ 1xxx\ xxxx.$$

The next section presents a case study of a respiration monitoring system. It will illustrate the implementation details and discuss the results.

4 Case Study: Respiration Monitoring System

This section will present a respiration monitoring system based on a MEMS flow sensor and a 3D accelerometer sensor. We will show the implemented architecture and

discuss the results. Moreover, temperature compensation and a tilt angle computation module will be studied and implemented on an Intel DE2 FPGA.

The proposed system comprises two MEMS sensors, a miniature flow sensor, and a three-axis digital accelerometer, BMA180. The flow sensor is used to measure the airflow through the nose and transmit the data for processing. The accelerometer is used to compute the tilt angle and assess the chest movement. It records the acceleration coordinates, which will be used for the computations. The reliability of the flow sensor has been tested, and various tests have been performed on the final structure [3]. The design's test and verification were carried out by implementing the proposed technique for temperature compensation and computing the tilt angle on an FPGA platform [40]. We can mention some potential applications of the proposed system in real time and continuous monitoring of sleep apnea syndrome, asthma, or Chronic Obstructive Pulmonary Disease (COPD). Further applications of the proposed system might be developed by embedding other modules using the collected data.

4.1 The Implemented System

The temperature drift compensation of the flow sensor's output was modeled with an ANN. An off-chip training has been done with MATLAB, and the architecture of the ANN has been defined. A Multi-Layer Perceptron (MLP) with two inputs (Ta ambient temperature and V flow velocity), one internal layer containing six neurons with a sigmoid transfer function, and one neuron output layer with a linear function. The weight vectors were defined with MATLAB and stored in a memory. The FPGA implementation of the pre-trained ANN was performed. Moreover, the tilt angle module was also implemented on the FPGA to compute the tilt angle using the CORDIC algorithm.

Figure 10 presents the different devices used and the signal conditioning blocks of the sensors. We used signal conditioning and an Analog-to-Digital Converter (ADC) to pre-process the signal and convert it digitally before sending it to the FPGA. In the FPGA chip, we implemented the compensation model and the CORDIC module.

The oscillator generates the clock, and different addresses are used to select the appropriate weight for each neuron's input. Amplifier, filter, sample, and hold circuits are all part of the conditioning circuit. The data in and out from the FPGA are converted using both ADC and DAC. Figure 11 depicts the FPGA architecture of the CORDIC tilt computation module.

a. Temperature compensation and test

The sensor was calibrated in a closed-circuit wind tunnel. The tests were carried out in a room with varying air temperatures of 15 °C, 20 °C, 25 °C, and 30 °C, respectively. Figure 12 shows the experimental setup of the calibration process [40].

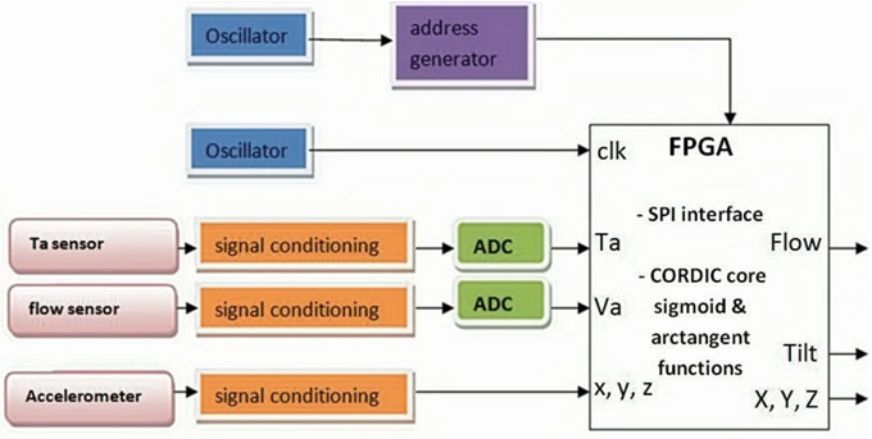


Fig. 10 Architecture of the respiration monitoring system where Va is the air velocity and Ta is the air temperature [40]

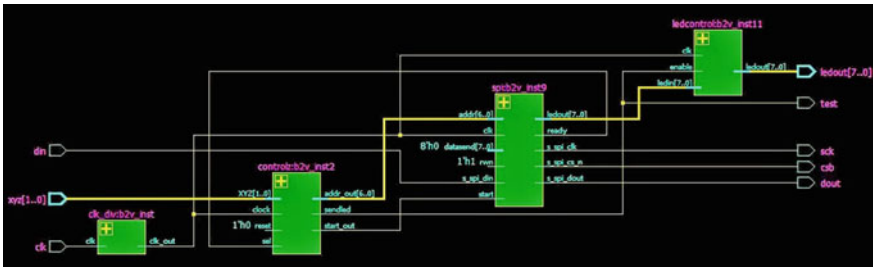


Fig. 11 FPGA architecture of the implemented tilt module: an SPI protocol was used to collect the data. The control block generates the clock and the address signals needed for computation [40]

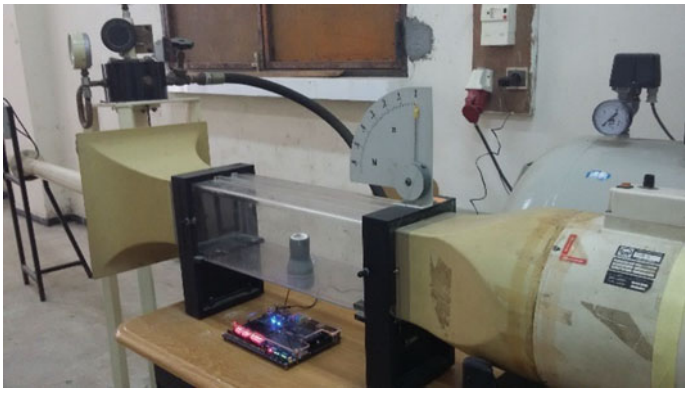


Fig. 12 Wind tunnel calibration of the hot wire sensor [40]

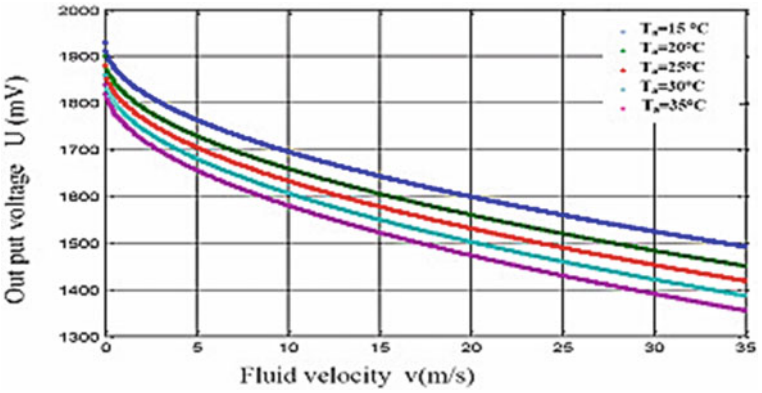


Fig. 13 Response of the MEMS flow sensor for different air temperatures [40]

We used different air temperatures to perform a calibration of the MEMS sensor. The results are depicted in Fig. 13. The flow sensor's features present two challenges. The first challenge is compensating for fluid temperatures, while the second is linearizing extremely nonlinear reactions. As a result, the circuit should be modeled to account for the temperature influence on the sensor's output. Inverse ANN modeling can be used to achieve this modeling [24, 25, 30].

The architecture of the ANN-based compensation model is presented in Fig. 14.

The backpropagation learning algorithm was used to train the three-layer Multi-Layer Perceptron (MLP). It is based on a feedforward network. We used two neurons in the input layer and six neurons in the hidden layer. Thus, we used 310 elements for learning and 90 elements for testing the proposed ANN. We used the Minimum Mean Square Error (MSE) to determine the number of neurons in the hidden layer and the weight factors. The defined architecture has been implemented in an FPGA board.

Figure 15 presents the hardware architecture of one single neuron with several internal modules. The multiplexer is used to select the appropriate neuron's input and weight from memory using a selection signal "Sel". The multiplier–accumulator

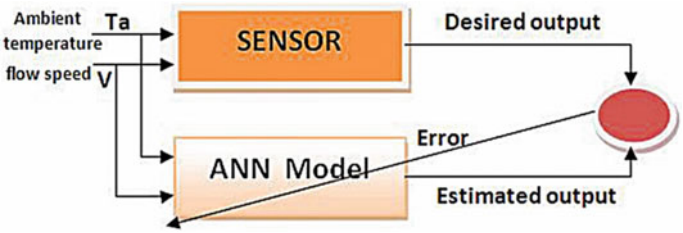


Fig. 14 Diagram of the developed ANN-based compensation model: the desired output is compared with the estimated output [40]

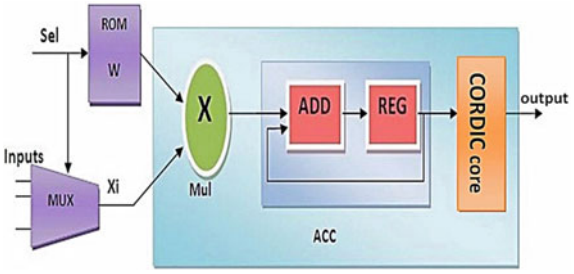


Fig. 15 Architecture of one single neuron: it contains a Sigmoid transfer function computed using CORDIC, a memory to store the neuron weights, and a multiplier–accumulator [40]

performs the addition–multiplication, and the CORDIC module is used to approximate the sigmoid activation function. Figure 16 presents the whole architecture of the FPGA-based implementation of the developed ANN. The control block generates different control signals, such as the selection signals used in the first and second layers, "Sel" and "SelLN", respectively. The hidden layer comprises six neurons with a sigmoid transfer function, and the output layer contains one linear neuron.

Results after compensation and linearization of the sensor's output are presented in Fig. 17. The maximum nonlinearity of the output signal is less than 0.35%, with a resolution of 0.07 V/m.s-1.

b. Tilt Angle Measurement

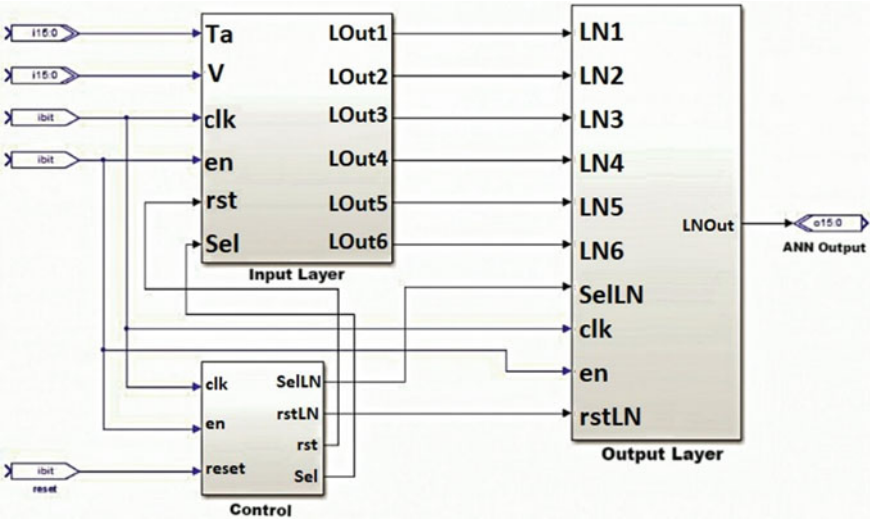


Fig. 16 Hardware architecture of the proposed ANN composed of three main modules: the input layer is composed of six neurons with a sigmoid transfer function, output layer containing one neuron with a linear transfer function, and a control block [40]

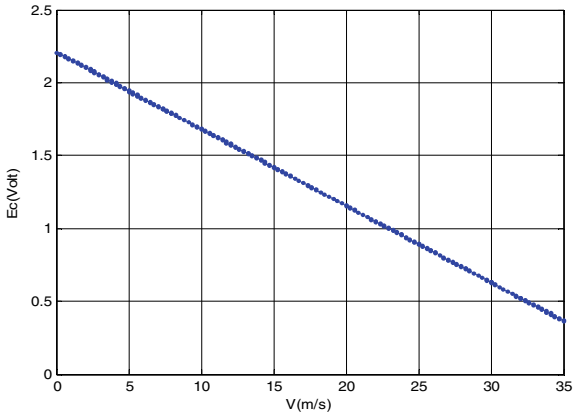


Fig. 17 The response of the flow sensor after compensation and linearization [40]

Using the CORDIC algorithm, we used a three-axis accelerometer, BMA 180, to assess the chest movement and compute the tilt angle [22, 23]. The CORDIC algorithm has two distinct modes: rotation mode, which determines the coordinates of any given vector after rotation through a given angle, and the vectoring mode, where the module computes the magnitude and phase of the vector [31]. In this work, we selected vectoring mode and the inverse tangent function to compute the tilt angle, $\theta = \tan^{-1}(Y/X)$.

$$Zi = Z0 + \tan^{-1}(Y0/X0) \tag{3}$$

By setting the function parameters to vectoring mode and $Z0 = 0$, we can compute the value of the $\tan^{-1}(Y0/X0)$ function using Eq. (3). Thus, the properties of the inverse tangent function can be used to simplify the implementation and the hardware architecture (Fig. 18).

Figure 19 presents the diagram of the state machine of the control block with its different states and transitions: the enable signal "en" and the rising edge clock trigger the transition from the idle state to state 1. Next, signal "sel" initiates the transition to state 2, and the tilt angle is calculated. Next, the transition to state 3 is triggered by the "selLN" signal. Finally, the "rst" signal sends the system to an idle state [40].

4.2 Results and Discussion

MEMS sensors have revolutionized human life by allowing the development of low-cost, high-performance, and low-power sensors. The emergence of the MEMS sensor industry in 2000 had paved the way for a considerable research and development community. Besides the emergence of the MEMS industry and its large deployment on ICT systems, many researchers started exploring MEMS sensor errors and

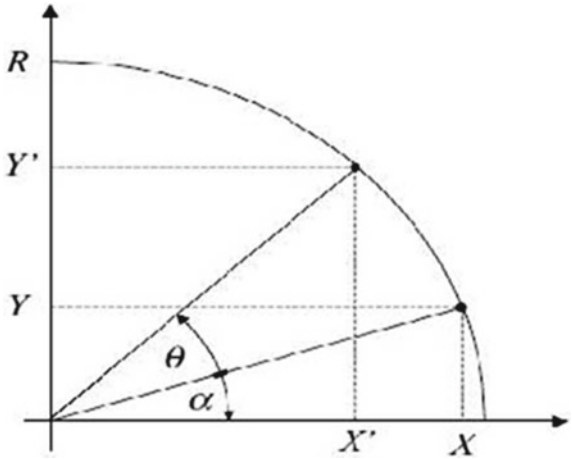


Fig. 18 CORDIC computation steps

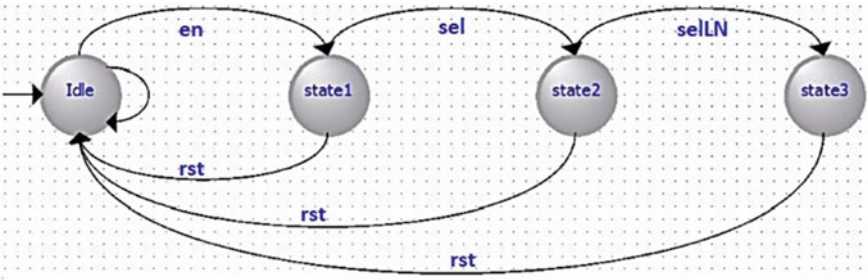


Fig. 19 State diagram description of the control block

defaults in the MEMS industry. Reliability has been a center of immense interest, especially with the introduction of MEMS devices in sensitive areas such as medical applications and the healthcare industry. Thus, many researchers started studying and investigating the sources of these errors and proposed several correction and compensation methods.

This chapter proposed a real-time respiration monitoring system using a flow sensor and a three-axis accelerometer to measure airflow and chest movement. An ANN-based compensation method has been used to compensate for the temperature drift of the flow sensors. In addition, we used the CORDIC algorithm to compute the tilt angle. The whole system has been implemented on an Intel FPGA with reasonable resource consumption, as reported in Table 3.

To verify the proposed system, we carried out a series test. First, the participant, a 35-year-old man, wore the wearable system for a short period. Then, thanks to the small size of the flow sensor, we could introduce it in the nasal cannula to measure the airflow. Figure 20 shows the response of the flow sensor recorded in the nasal

Table 3 The resource consumption

Registers	Pins	Logic elements	Memory
363	73/315	250/ 4,608	0

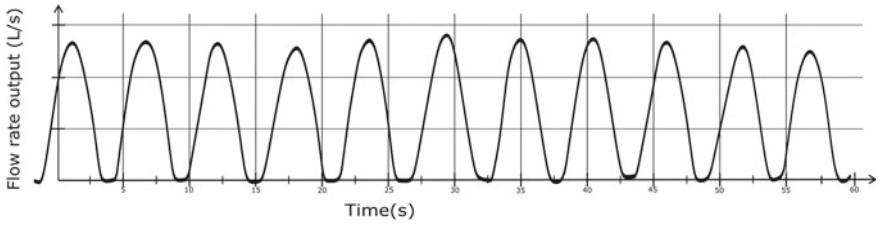


Fig. 20 Output of the flow sensor placed in the nasal cannula

cannula. The dynamic response of the airflow hot wire sensor was tested to verify its behavior. Experimental results showed that the flow range of the sensor is estimated to be 0–60 L/min. Furthermore, the area integral of the resultant flow-time curve demonstrated strong repeatability for 11 breath cycles using the flow sensor.

Figure 21 shows the breathing cycles measured with an accelerometer placed on the chest for three positions (sitting, sleeping, and standing). The raw data from the sensor X-axis showed a clearer signal when the subject was lying spine. Furthermore, the recorded respiratory frequencies correspond to 12–20 bpm which is the range of ordinary people.

5 Conclusion

In this chapter, we have discussed the MEMS sensors in ICT systems. We investigated the reliability of the MEMS sensors and explored different errors and correction methods. Two categories of errors which are systematic errors and random errors have been discussed. In addition, the impact of several factors on the MEMS sensors has been explored. Finally, we have proposed a hybrid system to monitor human respiration activity using two MEMS sensors: flow sensors and a 3D accelerometer. An ANN model has been used to correct the temperature drift of the flow MEMS sensors. Several reliability tests have been carried out to study the sensor reliability. A 3D accelerometer was used to measure the tilt angle and define the x-, y-, and z-coordinates. Furthermore, we used the CORDIC algorithm to compute the tilt angle and approximate the sigmoid function. Finally, we implemented the proposed system on a DE2 FPGA board using the VHDL language and performed some measurements. This system might be used for several healthcare applications to monitor respiration activity.

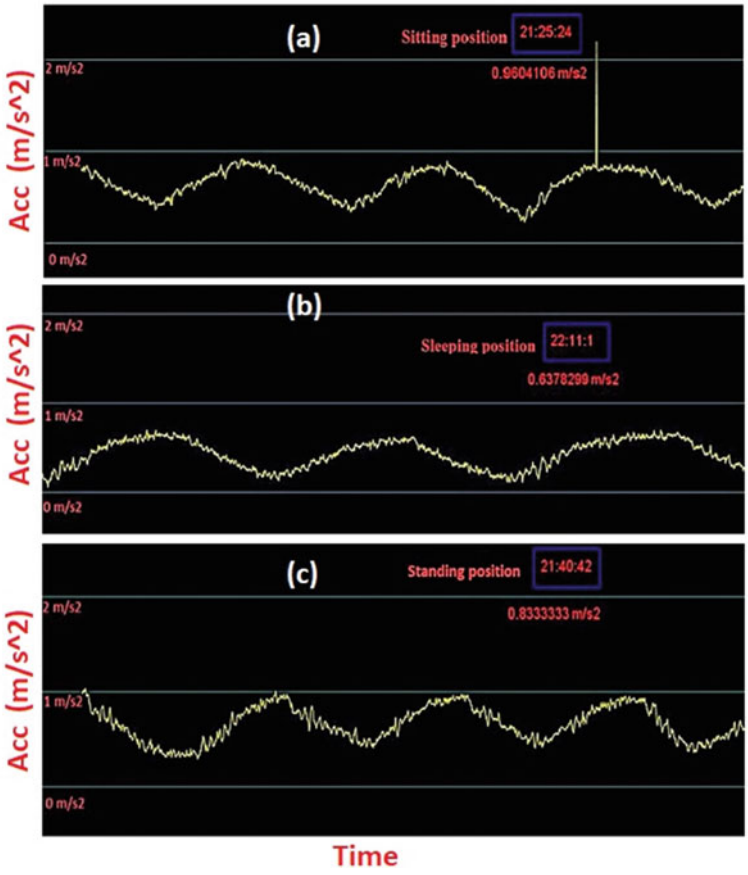


Fig. 21 Accelerometer output for three different positions along the x-axis for three different positions at different instants: **a** sitting, **b** sleeping, and **c** standing

References

1. Damianos D, Mounier E (2021) Status of the MEMS industry 2020. i-Micronews. https://www.i-micronews.com/products/status-of-the-mems-industry-2020/. Accessed 26 June 2021
2. Mohd-Yasin F, Nagel DJ (2021) Noise as diagnostic tool for quality and reliability of MEMS. Sensors 21(4). https://doi.org/10.3390/s21041510
3. Bensidhoum MT, Laghrouche M, Said AS, Montes L, Boussey J (2014) Fabrication flaws and reliability in MEMS thin film polycrystalline flow sensor. Microsyst Technol 7
4. Skogström L, Li J, Mattila TT, Vuorinen V (2020) Chapter 44-MEMS reliability. In: Tilli M, Paulasto-Krockel M, Petzold M, Theuss H, Motooka T, Lindroos V (eds) Handbook of silicon based MEMS materials and technologies, 3rd edn. Elsevier, pp 851–876. https://doi.org/10.1016/B978-0-12-817786-0.00044-X
5. Han S, Meng Z, Omisore O, Akinyemi T, Yan Y (2020) Random error reduction algorithms for MEMS inertial sensor accuracy improvement—a review. Micromachines 11(11). https://doi.org/10.3390/mi11111021

6. Naranjo CCM, Hgskolan KT (2008) Analysis and modeling of MEMS based inertial sensors. M.Sc. Thesis, school of electrical engineering Kungliga Tekniska Högskolan
7. Tamazin M, Noureldin A, Korenberg MJ (2021) Robust modeling of low-cost MEMS sensor errors in mobile devices using fast orthogonal search. J Sensors 9
8. Xu Y et al (2019) Silicon-based sensors for biomedical applications: a review. Sensors 19(13). https://doi.org/10.3390/s19132908
9. Chattopadhyay M, Chowdhury D (2016) A new scheme for reducing breathing trouble through MEMS based capacitive pressure sensor. Microsyst Technol 22(11):2731–2736. https://doi.org/10.1007/s00542-015-2707-0
10. Vandenbussche NL, Overeem S, van Dijk JP, Simons PJ, Pevernagie DA (2015) Assessment of respiratory effort during sleep: esophageal pressure versus noninvasive monitoring techniques. Sleep Med Rev 24:28–36. https://doi.org/10.1016/j.smrv.2014.12.006
11. Kulish V (2006) Human respiration: anatomy and physiology, mathematical modeling. WIT Press, Numerical simulation and applications
12. Nam Y, Park JW (2013) Child activity recognition based on cooperative fusion model of a triaxial accelerometer and a barometric pressure sensor. IEEE J Biomed Health Inform 17(2):420–426. https://doi.org/10.1109/JBHI.2012.2235075
13. Ghafar-Zadeh E et al (2017) Toward spirometry-on-chip: design, implementation and experimental results. Microsyst Technol 23(10):4591–4598. https://doi.org/10.1007/s00542-016-3200-0
14. Luo J et al (2015) rotating shaft tilt angle measurement using an inclinometer. Measure Sci Rev 15(5):236–243. https://doi.org/10.1515/msr-2015-0032
15. Vanegas E, Igual R, Plaza I (2020) Sensing systems for respiration monitoring: a technical systematic review. Sensors 20(18). https://doi.org/10.3390/s20185446
16. Becker DE, Casabianca AB (2009) Respiratory monitoring: physiological and technical considerations. Anesth Prog 56(1):14–22. https://doi.org/10.2344/0003-3006-56.1.14
17. Pernice R et al (2020) Low invasive multisensor acquisition system for real-time monitoring of cardiovascular and respiratory parameters. In: 2020 IEEE 20th mediterranean electrotechnical conference (MELECON), pp 306–310. https://doi.org/10.1109/MELECON48756.2020.9140716
18. Pan L et al (2020) Lab-on-mask for remote respiratory monitoring. ACS Materials Lett 2(9):1178–1181. https://doi.org/10.1021/acsmaterialslett.0c00299
19. Binu E, Varsha NS (2014) Real time monitoring of respiratory parameters using a wireless portable system. Int J Eng Devel Res 3(1):283–287
20. Zhu R, Cao Z, Que R (2014) Integration of micro sensors with mobile devices for monitoring vital signs of sleep apnea patients. In: The 9th IEEE international conference on nano/micro engineered and molecular systems (NEMS), pp 462–466. https://doi.org/10.1109/NEMS.2014.6908850
21. Laghrouche M, Montes L, Boussey J, Ameur S (2011) Low-cost embedded spirometer based on micro machined polycrystalline thin film. Flow Meas Instrum 22(2):126–130. https://doi.org/10.1016/j.flowmeasinst.2010.12.012
22. Laghrouche M, Montes L, Boussey J, Meunier D, Ameur S, Adane A (2011) In situ calibration of wall shear stress sensor for micro fluidic application. Proc Eng 25:1225–1228. https://doi.org/10.1016/j.proeng.2011.12.302
23. Makhlouf S, Laghrouche M, El Hamid Adane A (2016) Hot wire sensor-based data acquisition system for controlling the laminar boundary layer near plant leaves within a greenhouse. IEEE Sensors J 16(8):2650–2657. https://doi.org/10.1109/JSEN.2016.2518740
24. Kochan O (2014) Investigations of thermocouple drift irregularity impact on error of their inhomogeneity correction. Measur Sci Rev 14(1):29–34. https://doi.org/10.2478/msr-2014-0005
25. Mellal I, Laghrouche M, Idjeri B, Beguenane R, Ameur S (2012) Implementation of ANN in FPGA for improved temperature drift of the MEMS flow sensor. Sensors Trans 145(10):1–9
26. Brokalakis A, Papaefstathiou I (2012) Using hardware-based forward error correction to reduce the overall energy consumption of WSNs. In: 2012 IEEE wireless communications

and networking conference (WCNC), pp 2191–2196. https://doi.org/10.1109/WCNC.2012.6214156
27. Wang C, Burnham-Fay ED, Ellis JD (2017) Real-time FPGA-based Kalman filter for constant and non-constant velocity periodic error correction. Precis Eng 48:133–143. https://doi.org/10.1016/j.precisioneng.2016.11.013
28. Beechu NKR, Moodabettu Harishchandra V, Yernad Balachandra NK (2018) Hardware implementation of fault tolerance NoC core mapping. Telecommun Syst 68(4):621–630. https://doi.org/10.1007/s11235-017-0412-2
29. Correa-Caicedo PJ, Barranco-Gutiérrez AI, Guerra-Hernandez EI, Batres-Mendoza P, Padilla-Medina JA, Rostro-González H (2021) An FPGA-based architecture for a latitude and longitude correction in autonomous navigation tasks. Measurement 182:109757. https://doi.org/10.1016/j.measurement.2021.109757
30. Laghrouche M, Idjeri B, Hammouche K, Tahanout M, Boussey J, Ameur S (2012) Temperature compensation of micromachined silicon hot wire sensor using ANN technique. Microsyst Technol 18(3):237–246. https://doi.org/10.1007/s00542-012-1443-y
31. Tiwari V, Khare N (2015) Hardware implementation of neural network with Sigmoidal activation functions using CORDIC. Microprocess Microsyst 39(6):373–381. https://doi.org/10.1016/j.micpro.2015.05.012
32. Valls J, Kuhlmann M, Parhi KK (2002) Evaluation of CORDIC algorithms for FPGA design. J VLSI Signal Proc Syst Signal Image Video Technol 32(3):207–222. https://doi.org/10.1023/A:1020205217934
33. Volder JE (1959) The CORDIC trigonometric computing technique. IRE Trans Electr Comput EC-8(3):330–334. https://doi.org/10.1109/TEC.1959.5222693
34. Walther JS (1971) A unified algorithm for elementary functions. In: Proceedings of the May 18–20, 1971, spring joint computer conference. New York, NY, USA, pp 379–385. https://doi.org/10.1145/1478786.1478840
35. Kumar N (2011) Coordinate rotation digital computer algorithm: design and architectures
36. Liao W-T, Lin W-Y, Cheng W-C, Lei KF, Lee M-Y (2013) Precision enhancement and performance evaluation of a CORDIC-based tilting angle identification algorithm for three-axis accelerometers. In: 2013 International symposium on biometrics and security technologies, pp 187–192. https://doi.org/10.1109/ISBAST.2013.33
37. Duprat J, Muller J-M (1993) The CORDIC algorithm: new results for fast VLSI implementation. IEEE Trans Comput 42(2):168–178. https://doi.org/10.1109/12.204786
38. Mariani S, Ghisi A, Corigliano A, Martini R, Simoni B (2011) Two-scale simulation of drop-induced failure of polysilicon MEMS sensors. Sensors 11(5). https://doi.org/10.3390/s110504972
39. Zunino JL III, Skelton D (2005) Department of defense need for a micro-electromechanical systems (MEMS) reliability assessment program. Reliab Pack Testing Charact MEMS/MOEMS IV 5716:122–130
40. Mellal I, Laghrouche M, Bui HT (2017) Field programmable gate array (FPGA) respiratory monitoring system using a flow microsensor and an accelerometer. Measur Sci Rev 2:7
41. Laghrouche M, Boussey J, Adane A, Meunier D, Ameur S, Tardu S (2007) Polycrystalline silicon thin films for flow measurement 3
42. Varanis M, Silva A, Mereles A, Pederiva R (2018) MEMS accelerometers for mechanical vibrations analysis: a comprehensive review with applications. J Braz Soc Mech Sci Eng 40(11):527. https://doi.org/10.1007/s40430-018-1445-5
43. Vladimirova T, Tiggeler H (2006) FPGA implementation of sine and cosine generators using the CORDIC algorithm, p 12
44. Triple Axis Accelerometer Breakout-BMA180-SEN-09723-SparkFun Electronics. https://www.sparkfun.com/products/retired/9723. Accessed 26 June 2021
45. Asch G, Poussery B (2017) Les capteurs en instrumentation industrielle-8e éd. Dunod
46. Bhardwaj R, Kumar N, Kumar V (2018) Errors in micro-electro-mechanical systems inertial measurement and a review on present practices of error modelling. Trans Inst Meas Control 40(9):2843–2854. https://doi.org/10.1177/0142331217708237

47. Mohd-Yasin F, Nagel DJ, Korman CE (2009) Noise in MEMS. Meas Sci Technol 21(1):012001. https://doi.org/10.1088/0957-0233/21/1/012001
48. Iida M (2018) What Is an FPGA?. In: Amano H (ed) Principles and structures of FPGAs. Singapore, Springer, pp 23–45. https://doi.org/10.1007/978-981-13-0824-6_2
49. Ruiz-Rosero J, Ramirez-Gonzalez G, Khanna R (2019) Field programmable gate array applications—a scientometric review. Computation 7(4). https://doi.org/10.3390/computation7040063
50. Sadrozinski HFW, Wu J (2016) Applications of field-programmable gate arrays in scientific research. CRC Press
51. Zynq UltraScale+ RFSoC ZCU111 Evaluation Kit. Xilinx. https://www.xilinx.com/products/boards-and-kits/zcu111.html. Accessed 23 Sep 2021
52. Xu Y, Guo X (2020) Application of FPGA and complex embedded system in sports health data monitoring system. Microproc Microsyst 103445. https://doi.org/10.1016/j.micpro.2020.103445
53. Xu H, Sun Y (2021) Remote classroom system for Chinese linguistics teaching based on FPGA and embedded system. Microprocess Microsyst 81:103785. https://doi.org/10.1016/j.micpro.2020.103785
54. Guo X (2021) Application of agricultural IoT technology based on 5 G network and FPGA. Microprocess Microsyst 80:103597. https://doi.org/10.1016/j.micpro.2020.103597
55. Sklyarov V, Skliarova I, Sudnitson A (2011) FPGA-based systems in information and communication. In: 2011 5th international conference on application of information and communication technologies (AICT), pp 1–5. https://doi.org/10.1109/ICAICT.2011.6110989
56. Ali HK, Mohammed EZ (2010) Design artificial neural network using FPGA. IJCSNS 10(8):88
57. Technologies T (2021) Terasic-all FPGA boards-cyclone II-altera DE2 board. https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=53&No=30&PartNo=3. Accessed 26 June 2021

Chapter 4
Computation Infrastructure: Data Offloading, Processing, and Privacy

Yasasvitha Koganti, Ramnarayan Yadav, Sanjeet Kumar Nayak, and Manish Chaturvedi

Abstract This chapter elaborates the computation infrastructure, namely, cloud, fog, and edge computing. Various kinds of data offloading mechanisms and a general multi-layer computing framework addressing security aspects are presented. Presently, the smart Internet of Things (IoT) devices offload the large data aggregation and processing as well as storage to different computing platforms such as edge, fog, and cloud. In this chapter, various computing paradigms including their benefits and limitations are discussed. This chapter also discusses about the total cost in terms of latency and energy required to complete a task on user devices as well as remotely (on edge or cloud). Further, various necessary security and privacy issues are discussed that needs to be considered for large deployment of computing infrastructure for real-time ICT applications such as healthcare application. Finally, the chapter provides challenges and future directions for research in these computing paradigms, including security and privacy issues.

Y. Koganti (✉) · R. Yadav · M. Chaturvedi
Institute of Infrastructure Technology Research And Management, Ahmedabad, Gujarat 380026, India
e-mail: koganti.yasasvitha21pf@iitram.ac.in

R. Yadav
e-mail: ramnarayan@iitram.ac.in

M. Chaturvedi
e-mail: manishchaturvedi@iitram.ac.in

S. K. Nayak
Indian Institute of Information Technology Design and Manufacturing, Kancheepuram, Chennai, Tamil Nadu 600127, India
e-mail: sanjeetn@iiitdm.ac.in

M. Chaturvedi et al. (eds.), *Recent Advancements in ICT Infrastructure and Applications*, Studies in Infrastructure and Control, https://doi.org/10.1007/978-981-19-2374-6_4

1 Introduction

IoT devices which are an integral part of ICT infrastructure can play a crucial role in such prediction as well as in generating appropriate response. However, IoT devices are resource constrained and less powerful in terms of energy, computation, and storage. Thus, they cannot process and store huge amount of data generated locally. Currently, such limitations of these resource-constrained IoT devices are addressed by offloading processing tasks and storage to the distributed computing paradigms such as cloud. However, due to the huge amount of data generated by different smart sensors and IoT devices in the network, the cloud computing platform suffers from considerable mismatch such as (a) the large physical distance between IoT devices and the cloud requires Internet connectivity. This implies high latency and greater bandwidth demand, (b) transmitting large volumes of data to cloud and getting responses from it can quickly drain the energy of IoT devices, and (c) privacy and security are crucial factors in the success of cloud computing platform. However, the high concentration of data/information leaves the cloud highly susceptible to violent attacks and data offloaded to the cloud through wireless environments can be overheard by eavesdroppers, and cloud computing is not suitable for applications that require the widespread distribution of devices and users. Therefore, there is a need to advance computing and communication infrastructures which add more computing devices closer to sensors/IoT devices for time-sensitive applications. To overcome these limitations, the concept of edge computing has been proposed. Fog computing, a type of edge computing, has the capability of reducing the amount of data to be transferred to the central cloud. In this paradigm, computing resources are made available at the edge of the network. Close proximity of computational resources and IoT devices offers several advantages. For instance, if the load is beyond the capabilities of IoT devices they can offload it to the edge servers. Also, close proximity of edge servers and devices improves the latency. Further, the huge amount of data generated though different smart sensors needs to be stored for future use and processed (so as to find meaningful patterns) on cloud computing resources. Thus, cloud and edge computing complement each other. Now, we briefly discuss about the data aggregation and processing as follows.

1.1 Data Aggregation and Processing

Nowadays, automation plays a vital role in many applications. To achieve this, the previous data needs to be maintained to determine and execute the appropriate events/actions. Gathering and expressing information in a summary form often termed as data aggregation is useful for purposes such as statistical analysis. In general, aggregation consists of a set of datasets having finite number of variables which can affect the statistical analysis. In various IoT applications, data aggregation plays a vital role in improving efficiency and latency. Various software-based data aggrega-

tion tools (e.g., Microsoft Excel, MongoDB, DbVisualizer, IBM Cloud Pak for Data, etc.) are used to perform aggregation of large-scale data. Such data aggregation tools can be used to collect, process, and present the aggregated data.

- *Collection*: The data is collected from various sources such as sensors and IoT devices. In some cases, existing historical data is also considered.
- *Processing*: The collected data is processed to get a suitable statistical analysis. The data is processed by using various Artificial Intelligence (AI) and Machine Learning (ML) algorithms for predictive analysis.
- *Presenting*: The processed data is presented in such a format that it is considered as meaningful information which is of high quality.

2 Server-Based Computing Infrastructure

Server-based computing infrastructure refers to the scenarios where services that are running on servers can be accessed using web browser. Server infrastructure is nothing but to have the resources physically. The server infrastructure comprises both physical and virtual components. It consists of hardware, software, network resources, and services that it offers. Server contains all the data related to applications like programs and data to work on. So it is a bit difficult to maintain and have data backup. In general, servers are located in a secured place. In a server-based infrastructure, users have a thin client along with external devices like keyboard and mouse. Thin client is used to connect a user to the server. The thin client and the server are connected by standard computer networking. Power consumption and heat generation are very less in thin clients which improves their life.

We can also use thick client in which most of the data will be on client. Whereas in a thin client the majority data is stored in server rather than on client which improves the security. In a client–server architecture, there is a chance to use the software and hardware from different vendors. Due to heterogeneous software and hardware, middle-ware is used for smooth operation of client–server services. Middle-ware is a software that acts as a communication bridge between a client and a server's application software (Fig. 1).

In general, client–server infrastructure is of two types: centralized and decentralized. A centralized client–server infrastructure has a single data center that is shared by users. In this, server has all the administration access such as access rights and resource allocations. Whereas a decentralized client–server architecture has different data centers which are used and controlled by their respective users. Though it is easy to protect the data in a client–server infrastructure and to provide authorization and authentication, for having the server infrastructure, a lot of things such as managing data, files, configuration, security, storage, performance testing, disaster recovery, etc. are required to be taken care of. To avoid these overheads, a cloud infrastructure is proposed.

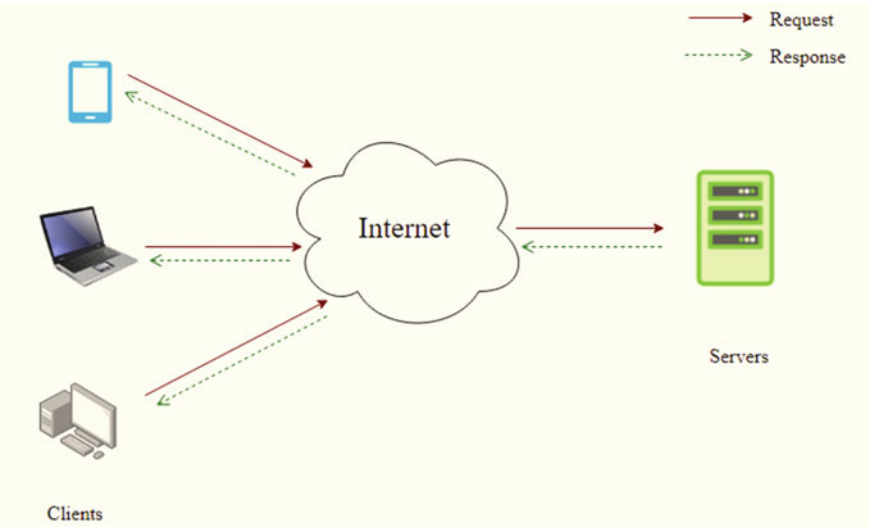


Fig. 1 Client–server architecture

3 Cloud Computing Infrastructure

Cloud computing is a technology that puts the entire computing infrastructure online. It uses the Internet to maintain data and applications at remote central servers. Several widely used applications such as Gmail, Yahoo mail, Facebook, etc. are examples of cloud-based services.

3.1 Evolution

Cloud plays a pivot role in IT industry as it reduces the server cost and maintenance. Companies mainly use cloud as it makes a large difference in storage and cost. One of the formal definitions of cloud computing as given by National Institute of Standards and Technology (NIST) is [1]—"Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment mode."

Cloud computing is the modification of previous concepts. In 1960s Mc Carthy proposed a concept called "Utility Computing" [1]. In utility computing, instead of any other product, utility is considered as major concept to meet its requirement. In

upcoming days, utility is modified into storage, computation capacity, etc. [2]. It is an on-demand type structure where the resources are allocated depending on utility.

Grid computing is the basic level of cloud computing. Grid computing mainly focuses on parallel and distributed computing. It shares the data based on geographical location, availability of resources, and users' quality of service. In grid computing, performance and cost incurred are also given importance. Cloud computing is similar to grid computing but is often more user-friendly. In cloud computing, we use large number of small applications to run application which introduces the concept of micro-services. Micro-services involve dividing the application into small parts. For example, consider a website which consists of Home, About, Contact, and Login page as minimum objects. In micro-services, each object is considered as one micro-service and deployed on each container which lessens the load and increases the availability. In such cases, large number of small applications can do better which reduces the large capacity application and reduces the cost. Containers play a key role in cloud computing.

Cloud computing consists of virtual servers where we can deploy the containers. Cloud computing layout mainly consists of three components: data centers, servers, and client as shown in Fig. 2. Data centers store the data of applications deployed on virtual servers. The physical server is used to maintain all the virtual servers which exists in cloud. Clients are the devices which are used to access the application in virtual servers. End users use clients to access the application. The next component is virtual servers which can be scaled up and down and also present at different locations of cloud so that the load is distributed and availability is increased. The applications can be accessed through thin and thick clients but the data of the application remains on virtual servers.

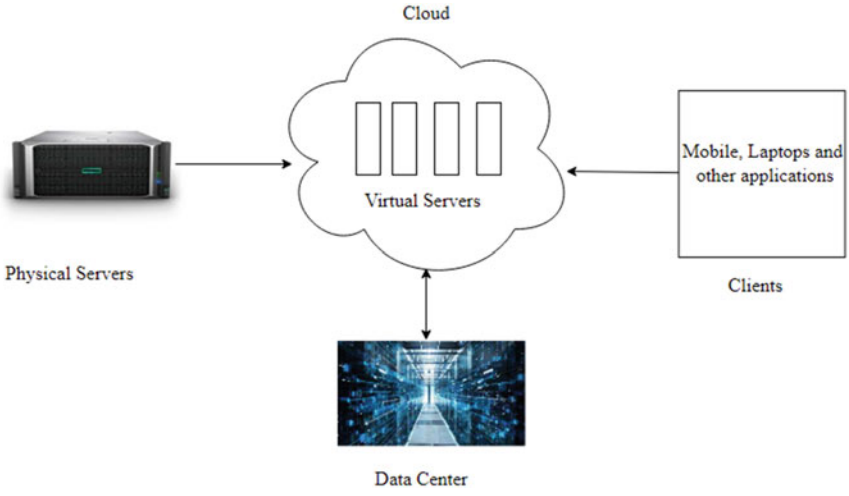


Fig. 2 Cloud computing layout

3.2 Essential Characteristics and Models

3.2.1 Essential Characteristics of Cloud Infrastructure

- On-demand self-service: The virtual servers required to deploy the applications are adjusted according to the traffic/load. There is no need of any human interaction to increase or decrease the number of servers.
- Resource pooling: The resources which are idle can be used by any customer as it is not specified to only one. Depending on traffic/load, the resources used by customers vary from time to time.
- Rapid elasticity: Computational capacity can be used depending on load. It has maximum elasticity as it can increase and decrease as required.
- Measured service: To attain the high resources performance, utility, and other parameters are monitored and based on that the usage of resources can be adjusted.

3.2.2 Cloud Computing Models

In general, cloud computing models are divided into two categories which are known as service models and deployment models.

- *Service models*: The services offered in cloud are categorized as service models.
  - Software as a Service (SAAS): Software is offered as a service on cloud platform where there is no need to take care of the service by customer except using it. In this, very limited manual work is required. The cost of software can be a bit high while compared with others. The application can be accessed on browsers through thin clients. Secure Socket Layer (SSL) is used so that application can be accessed securely.
  - Platform as a Service (PAAS): Cloud provides a platform where customer can install the required software and use it rather than using the installed one like in SAAS. The backdrop in this service is that there are compatibility issues between the different vendors which provide the software and the deployment platforms.
  - Infrastructure as a Service (IAAS): In this, cloud provides all the resources required for customer to deploy a software from scratch. Even the operating systems is also selected by the customer. Customer uses the resources such as security, network groups, data storage, and all to build the application. As the customer is setting up the whole environment, there will not be any compatibility issues and the cost is reduced while compared to SAAS. Customer has to take care of each and every aspect as the cloud will not take care of it.
- *Deployment models*: The cloud can be classified into four models based on which platform the application will be deployed.

- Public cloud: The cloud infrastructure is open to all users. The application deployed is visible to all users. There are no restrictions to users in using the application.
- Private cloud: The cloud infrastructure is assigned for an organization where the users of the organization will be able to access the cloud and its services.
- Community cloud: In this, there are users who belong to different organizations but whose interest will be same, i.e., users will be dealing with same topic. So, users from different organizations but dealing with same topic can be formed as community cloud.
- Hybrid cloud: It is a combination of one or more distinct clouds.

Since many applications require predictions and quick responses, it is essential to accurately predict different events such as abnormal health conditions, disasters so as to save the lives of thousands of people. For the same, Internet-of-Thing (IoT) devices play a crucial role. The use of computation for decision-making during various events is also supported by IoT devices. There is a serious mismatch between the growing traffic volume and the availability of resources to support the traffic for computation and communication. Therefore, latency-aware computation infrastructure required developed and deployed. Though IoT devices seem to be a promising tool, they are resource constrained and have several issues. It is difficult for IoT devices to store as well as process the huge amount of data generated locally. Currently, limitations of IoT devices are addressed by offloading processing and storage to the cloud. These aforementioned technology named cloud computing has so far proved to be effective in providing the infrastructure and connectivity required to collect and analyze data from users as well as physical environment. This in turn helps in the required computing and cognitive decision-making. However, the cloud computing platform suffers from several disadvantages such as (i) the large physical distance between the IoT devices and the central cloud requires Internet connectivity for data transmission. This implies high latency and greater bandwidth demand (ii) transmitting large volumes of data to cloud and getting responses from it can quickly drain the energy of IoT devices. To overcome these limitations, the concept of edge computing has been proposed.

4 Edge Computing Infrastructure

The main thing which favors edge computing is virtualization. The number of resources required in edge computing is limited. Virtualization allows the use of a limited number of resources with maximum efficiency. Complex libraries can be used in virtualization that can help by having separate resources for each service. This reduces disruption on other services in case a particular service is down. Further, there is an increase in the response speed which is crucial in an edge computing paradigm. This will be useful while Deep Learning (DL) and Federated Learning (FL) are used in edge computing.

Table 1 Integrated commodities for edge computing

Owner	Production	Feature
Microsoft	Data box edge	Competitive in data processing and data transmission
Intel	Movidius neural compute stick	Prototype on any platform with plug and play
NVIDIA	Jetson	Easy-to-use platforms that run in as small as 5 W
Huawei	Atlas series	An all-scenario AI infrastructure solution that bridges "device, edge, and cloud"

Advantages of edge computing over cloud computing are as follows:

- *Backbone network alleviation*: In edge computing, the edge devices can execute the tasks without sending them to cloud. This reduces the traffic load.
- *Agile service response*: As the services can be executed on edge devices, this will reduce the time taken for data transmission and improves response time.
- *Cloud backup*: Service which requires more computational capability will be sent to cloud from edge devices.

The services offered by edge computing are numerous but for some factors the services may be limited [3]. So, based on application requirements, we can distribute proportion of the task across different computing infrastructures. Some of the major factors affecting the capabilities of computing architectures are discussed as follows:

- *Cost*: Applications using various Machine Learning (ML) algorithms process huge data for prediction and learning. In general, ML algorithms are trained in cloud. In such cases, huge amount of data need to be transferred to cloud from the user device which increases its cost (since it requires huge network bandwidth).
- *Latency*: The time to access cloud cannot be specified and it cannot be matched with the requirements of various real-time applications.
- *Reliability*: The total data transfer will be depending on the Internet connection through which the communication occurs. Some services require that the data need to be transferred even when the Internet connectivity is lost.
- *Privacy*: The data required for ML carries a lot of sensitive information which we can't afford to risk in cases like smart homes, cities, etc.

For applications where ML is applied, we can choose edge over cloud as we can have the low latency, higher reliability, and higher privacy. A comparison among on-device intelligence, edge, and cloud is shown in Fig. 3.

Hardware for edge computing

Now, we discuss the hardware components for edge computing. A detailed list of AI-enabled hardware is given in Tables 1, 2, and 3.

- *AI hardware for edge computing*: It is divided into three categories based on their technical aspects. First is Graphics Processing Unit (GPU)-based hardware. The

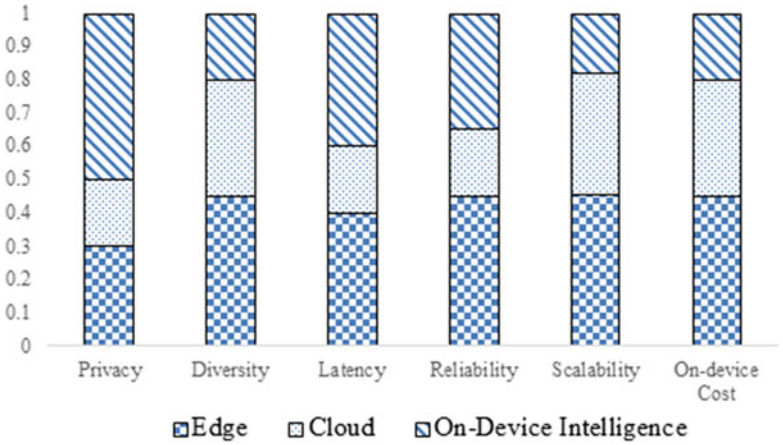


Fig. 3 Capabilities comparison of computing infrastructure [3]

Table 2 AI hardware for edge computing

Owner	Production	Feature
Qualcomm	Snapdragon 8 Series	Powerful adaptability to major DL frameworks
HiSilicon	Kirin 600/900 series	Independent NPU for DL computation
HiSilicon	Ascend series	Full coverage—from the ultimate low energy consumption scenario to high computing power scenario
Media Tek	Helios P60	Simultaneous use of GPU and NPU to accelerate neural network computing
NVIDIA	Turing GPUs	Power capabilities and compatibility but with high energy consumption
Google	TPU	Stable in terms of performance and power consumption
Intel	Xeon D-2100	Optimized for power and space-constrained cloud-edge solutions
Samsung	Exynos 9820	Mobile NPU for accelerating AI tasks

performance and compatibility of this is high but the power consumption is also high (e.g., NVIDIA's GPU based on Turing architecture). Second is Field Programmable Gate Array (FPGA)-based hardware in which the energy consumption is less and limited number of computational resources are available. Compatibility is also an issue for FPGA-based hardware. Third is Application-Specific Integrated Circuit (ASIC)-based hardware, it is a customized design to improve the performance and power consumption (Google's TPU and HiSilicon's Ascend series are examples of this type of hardware).

- *Edge nodes having computing and caching capabilities*: Edge nodes have computing ability to process tasks. Data can be sent to edge nodes from end devices for

Table 3 Edge computing framework

Owner	Production	Feature
Huawei	KubeEdge	Native support for edge-cloud collaboration
Baidu	OpenEdge	Computing framework shielding and application production simplification
Microsoft	Azure IoT edge	Remotely edge management with zero-touch device provisioning
Linux foundation	EdgeX	IoT edge across the industrial and enterprise use cases
Linux foundation	Akraino edge stack	Integrated distributed cloud-edge platform
NVIDIA	NVIDIA EGX	Real-time perception, understanding, and processing at the edge
Amazon	AWS IOT Greengrass	Tolerance to edge devices even with intermittent connectivity
Google	Google Cloud IoT	Compatible with Google AI products, such as TensorFlow Lite and Edge TPU

processing as it can reduce time. Edge devices can also provide caching capacity and high-quality network so that it can easily communicate with end devices and caching popular contents can be helpful to reduce the response time. For example, Azure Data Box Edge equipped with AI-enabled edge computing capabilities.

- *Edge computing frameworks*: Edge computing system is making its way in all aspects. The same is for ML, as it requires complex configuration and intensive resources. Micro-service is the advanced concept in edge computing. Kubernetes is the famous micro-service platform which deploys the application on containers and automatic load balancing is also applied for the application based on the traffic. For example, KubeEdge by Huawei is developed for implementation of edge computing. It is mainly used to improve networking, deployment, and data synchronization between cloud and edge. Open edge also falls under same category mainly focusing on computer framework and application production.

Mobile edge computing Mobile edge computing (MEC) is an architecture where cloud services are extended to the edge of networks using mobile base stations. Nowadays, mobiles are largely used. Every person is carrying mobile phones. The computation power, memory used, and other configurations of mobile are not sufficient as cloud computing so Mobile Edge Computing (MEC) is introduced from Mobile Cloud Computing (MCC) so that mobiles can offload their tasks to the nearby edge devices like Small-cell Base Stations or Macro-cell Base Stations so that the execution time, processing time, and others will be reduced. Edge devices are used to connect the mobiles with clouds. MEC is evolved from MCC. It was first proposed by European Telecommunications Standards Institute in 2014. MEC consists of mobiles/vehicles, Base Stations, and Central Centers. MEC has the characteristics of close range, ultra-low latency, ultra-high energy efficiency, and ultra-high relia-

bility. MEC is a key technology to work in 5G. There are different ways to realize edge computing services. They are as follows:

1. *Cloudlet and Micro-data Centers*: Cloudlets are also considered as mini-clouds. It is considered as the middle element in architecture, i.e., mobile devices, micro-clouds (cloudlets), and cloud. It is mainly used to perform low-latency computations.
2. *Fog Computing*: Fog computing is similar to cloud computing. However, fog is confined only to some geographic areas which is not the same for cloud. Fog mainly focuses on IoT as it is mainly designed for low-latency applications.

4.1 Resource Allocation Problem in Edge Computing

Resource allocation is a prominent topic in edge computing. Resource allocation is the process of assigning the tasks with required resources according to their preferences and priority to suitable edge servers [4]. The key terms in resource allocation are discussed below:

- *Resources*: Resources are the provided services through which a task can be completed. Communication resources, storage resources, and computational resources are the types of resources that exist in edge computing.
- *Tasks*: This refers to the work to be done. It varies from application to application. In general, an arbitrary task T can be described as follows: $T = \{D, c, p, d\}$, where D is the size of the data of task T, the processing density (in CPU cycles/bit) of T is denoted by c, p represents the parallelizable fraction of the task T, $(0 \leq p \leq 1)$, and d is the delay constraint of the task T [5].
- *Participants*: In edge computing to perform a task mobile devices/vehicles, edge devices and cloud are used. The devices which are used to perform task are considered as participants.
- *Objectives*: Each participant has their own preferences depending on the task they are performing. Some wants to reduce energy consumption whereas some prefers latency. These can be referred to as performance indicators.
- *Actions*: Actions are referred to as the way to meet the goals set by the participants. It can be achieved in three steps. First is computational offloading which represents the task offloading to edge or cloud. Second is resource allocation which refers to the allocation of communication, storage, and computing resources to the users/devices. Third is resource provisioning in which a user-resource pair is made as per the requirements of task mostly from the user's perspective.
- *Methodology*: The methods used to complete the actions comprise the methodology. It constitutes the techniques and algorithms to be used. It is categorized into two types: centralized and distributed. A centralized methodology needs a central unit to collect the information of the whole system whereas a distributed methodology doesn't need any such unit.

The resource allocation strategies are designed from three perspectives as follows:

- *User*: Users constitute the end devices through which we can get the data. There are a lot of heterogeneous end devices which constitute mobiles, vehicles, smart watches, etc. Each have their own preferences in completing tasks.
- *Service Provider*: Edge computing consists of service providers which are used to provide the services for edge computing, application, and mobile operators. The service providers will charge the amount based on the task they will be executing. So the economical benefit is checked, i.e., at which rate it can obtain maximum gain with minimum cost. Such a strategy is often preferred here.
- *Edge Network*: Edge network consists of distributed edge resources. These resources can be used efficiently through resource scheduling.

5 General Edge Computing Model and Research Issues

In this section, we discuss the basic model for resource scheduling in an edge computing framework. Based on the current communication and computing resource requirements and their QoE requirements, users decide whether to offload their tasks. Then, the state-of-the art research on resource scheduling in edge computing is discussed from three aspects: computation offloading, resource allocation, and resource provisioning.

5.1 Basic Model

In an edge computing paradigm, several tasks are generated from resource-constrained IoT/user devices. In general, any arbitrary task T can be described as follows: $T = \{D, c, p, d\}$, where the size of the data of task T is represented using the processing density (in CPU cycles/bit) of task T is represented using c, p represents the parallelizable fraction of task T, $(0 \leq p \leq 1)$, and d is the delay constraint of task T [5].

Connection of resource-constrained IoT devices to the edge servers can take place through various communication channels such as 4G/5G, Wi-Fi, etc. The processing of any generated task T can take place locally at the IoT device or offloaded to the edge. The offloading decision is taken based on different requirements such as consumption of energy, latency requirement, cost, and computing acceleration.

Let an offloading decision variable be denoted by x such that $(0 \leq x \leq 1)$. Such an offloading decision variable represents the ratio of the offloaded data size to the total data size of task T. The value of the decision variable represents whether the task will be processed locally or offloaded. For example, if $x = 0$, task T will be processed at the IoT device locally; if $x = 1$, offloading of the task T will take place; in other cases, offloading of task with data having size $x \cdot D$ will occur and the

remaining data having size $(1-x)\cdot D$ will be processed at the IoT device locally. Further, we discuss the processing of task locally and at edge servers.

Local computation analysis for task T: Now, we discuss different costs associated with local computation.

1. Local computation time: The computing time for $(1-x)\cdot D$ bits data of the task locally at the IoT device comprises the computing time of the serialized part and the computing time of the paralleled part. Let the number of cores present in the user's device is denoted as n_l, and the processing capability (i.e., the amount of CPU frequency in cycles/s) of each core be represented as f_l. So local computation time is

$$T^l = T_s^l + T_p^l \tag{1}$$

$$T^l = \frac{c(1-p)(1-x)D}{f_l} + \frac{cp(1-x)D}{f_l n_l} \tag{2}$$

2. Local energy consumption: The consumption of power at each core for a user's device for local processing of the data can be expressed as $p_l \propto c_1 \cdot f_l$, where c_1 is a constant [4]. So, the local energy consumption is defined as

$$E^l = p_l \cdot t_s^l + n_l \cdot p_l \cdot t_p^l \tag{3}$$

Computation analysis of partial task T offloaded to the edge: The data (partial or complete) of a task T can be offloaded to the edge using wireless communication links. According to Shannon's formula, when data is offloaded from the user device j to the edge i over an assigned wireless communication link having bandwidth B_{ji}, the rate of transmission r_{ji} is represented as follows.

The data rate of user j task offloaded to edge i is formulated as

$$r_{ji} = B_{ji}\log_2(1+\frac{P_j h_{ji}}{N}) \tag{4}$$

where P_j denotes the transmission power of the user device j, h_{ji} is the channel gain between user device j and the edge node i. N is the background noise power. Thus, when the task is offloaded to the edge, the completion time and the consumption of the energy can be computed as follows:

- Transmission delay for offloading to edge: The transmission delay for offloading $x \cdot D$ bits of data to the edge can be computed as follows:

$$T^{up} = \frac{xD}{r_{ji}} \tag{5}$$

- Energy consumption in task offloading: The consumption of energy of the user device for transmitting the offloaded $x \cdot D$ bits of data is represented as follows:

$$E^{up} = P_j \cdot T^{up} \tag{6}$$

- Task computing time at edge: After data offloading, the processing of the data at the edge takes place. Let n_e denote the number of cores of the edge assigned for task processing , f_e denote processing capability (i.e., the amount of CPU frequency in cycles/s) of each core at edge. The consumption of power at each core is expressed as $p_e \propto c_2 \cdot f_e$, where c_2 is a constant. So, the time for computation at edge is given as

$$T^e = T_s^e + T_p^e \tag{7}$$

$$T^e = \frac{c(1-p)(x)D}{f_e} + \frac{cp(x)D}{f_e n_e} \tag{8}$$

- Energy consumption in task processing at the edge: The energy consumption of the edge for computing the $x \cdot D$ bits of data is as follows:

$$E^e = p_l \cdot t_s^l + n_e \cdot p_l \cdot t_p^l \tag{9}$$

The result of the computation is sent back to the user device once the processing of the task T is completed. In general, the time taken to send back this result is neglected due to its small size [4, 6–8].

Overall cost of computing task T: The overall cost of processing the task T includes three aspects: consumption of energy, computation resource requirement, and network bandwidth requirements. Let a_e denote the cost of one unit energy consumption so total energy consumption cost is calculated as follows:

$$C_{energy} = a_e(E^l + E^{up} + E^e) \tag{10}$$

For network bandwidth, let a_2 be the cost of using per unit of bandwidth per unit of time, then the cost of the bandwidth cost can be expressed as follows:

$$C_{bandwidth} = a_2 \cdot B \cdot T^{up} \tag{11}$$

For calculating the computation cost of task T (at edge and local), let a_3 be the cost of using per unit of processing capability/unit of time, then the computation cost of T is as follows:

$$C_{computation} = a_3.n_l.f_l.T^l + a_3.n_e.f_e.T^e \tag{12}$$

Thus, the overall cost for completing the task T can be represented as follows:

$$C_T = C_{energy} + C_{bandwidth} + C_{computation} \tag{13}$$

In general, the aim is to minimize this overall cost.

6 General Edge-Enabled Cloud Computing Architecture

A generalized framework consisting of edge nodes is shown in Fig. 4. The three-layer architecture of edge/cloud framework for smart applications such as healthcare and transportation is shown in Fig. 4. The sensing layer contains biomedical sensors embedded around the patient, and/or IoT devices related to transportation. The data sensed by these devices is sent to edge devices in edge computing layer securely with preserving privacy. The edge devices perform data processing, data analysis, and data exchange with other edge devices. At the cloud layer, data center nodes are responsible of time-consuming task and data storage for future use.

1. *Sensing layer*: This layer is responsible for data collection and transmission to the edge computing layer. This layer in the architecture is also known as the perception layer. Data could be collected through Wireless Identification and Sensing Platform (WISP) [9], smart IoT sensors, and nodes within radio range of each other may also collaborate for ubiquitous data transmission among the IoT devices within the networks. To incentive the participation of users/people, various truthful models based on contract theory and game theory are designed. Moreover, different data contributors and service providers also aimed to maximize their revenue through participation in the system [10]. In order to maintain the fairness criteria, graph theory and other fairness models will be exploited in the system [6].
2. *Fog Intelligence layer*: The use of edge computing minimizes the computational load on cloud servers located far away from the IoT devices. The inclusion of edge computing increases the real-time response and reduces the latency in the network. The use of edge computing enhances the distributed nature of the networking framework and supports the mobility within the networks [18]. The potential

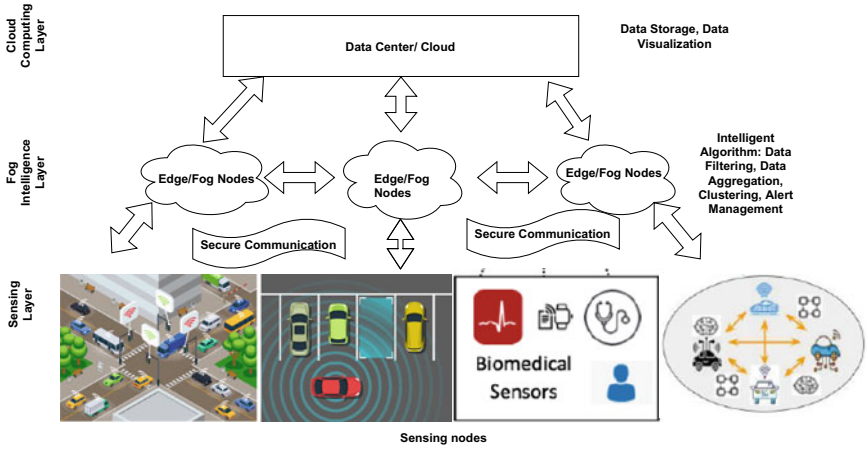


Fig. 4 General edge-enabled cloud computing architecture

implementations of edge computing are Mobile Edge Computing (MEC) and Cloudlet Computing. The potential machine learning algorithms that could be applied for data estimations are Federated Learning, Long Short-Term Memory (LSTM), Convolution Neural Network (CNN), Deep Learning, etc. to compute the traffic-related data and further forward the data to the next layer in the architecture. To overcome the problem of minimizing network congestion, minimizing the end-to-end latency, tackling connectivity bottlenecks, and enabling security for data, the architecture can be enabled with the fog computing platform. This layer will analyze and perverse the data for better performance and predictions. The blockchain model will be implemented at the fog layer to provide the secure communication framework in the network. The secured information will be further forwarded to the cloud for global update and alert in the system.

3. *Cloud computing layer*: The cloud computing platform will work as the decision-maker and potential directions for urban city management in this project. Based on the data collected from the local fog computing layer, the cloud server will make the decisions and accordingly inform people to follow a particular direction to avoid the congestion in the traffic. Different policy-makers such as Government personnel, Police, and NGO can take input from the cloud in order to manage the anomaly in urban areas. Different advanced machine learning, deep learning algorithms will be employed to find out the accurate results. Losses due to congestion such as fuel and time because of uncertain anomalies will be minimized with the help of the proposed prototype model. II. Communication framework: The proposal estimates coming up with a four-tier model IoT, edge, fog, and cloud-based architecture. The data will be collected with help of different IoT sensors and computed at different edge and fog layers. The computed data further forwarded to the cloud server for global update. The communication framework will be developed over 4G/5G networking model in order to transfer data and compute results across IoT to cloud. Theoretical improvements in communication model will be utilized by designing new algorithms at MAC layer for Orthogonal Frequency-Division Multiple Access (OFDMA) and Non-Orthogonal Multiple Access (NOMA) technology for reliable communication framework.

We can divide the overall tasks into several modules as provided below.

6.1 Module 1: Computation Offloading Module

For near-real-time monitoring of health of individuals, several wearable several biomedical sensors are required. Since these wearable devices are resource constrained, the sensed data needs to be communicated to the nearby fog nodes for processing the data so as to extract meaningful information in a timely manner. For this, an efficient communication between the biomedical sensors and the fog nodes needs to be established. Since IoT devices are resource constrained, the tasks that need to be carried out are offloaded between the fog and cloud. This ensures

the scalability of the system which is critical in time-sensitive applications such as healthcare applications. To overcome the computational limitation of biomedical sensors, computation-intensive tasks are sent to fog nodes in proximity for processing and analysis. One of the objectives of the proposed project is to develop a hybrid approach (combination of partial and binary) for combining local computation and offloading which analyze the fundamental trade-off between local computation and task offloading on the computational efficiency. To achieve optimal system efficiency, an optimization problem combining communication and computation latency will be formulated with the objective of minimizing the sum of task latency of IoT devices. Further, the gradient descent algorithm will be used to find the solution of the formulated problem. The battery power, processing power, and storage capacity of IoT devices are limited. With the huge number of IoT devices and large number of services as well as with their heterogeneous QoS requirements, it is crucial to optimally offload the tasks. This requires the servers to allocate the limited resources to the applications and meet their QoS requirements. As part of this project, we aim to develop an optimal selective task offloading strategy by designing an efficient edge resource allocation algorithm which satisfies the capacity constraint of edge servers and maximizes the number of services and minimizes the average service response time (meet the deadline). Such an optimal selective task offloading strategy will consist of the following key operations:

1. The IoT devices will generate offloading requests.
2. Execute the offloading decision algorithm.
3. Based on the offloading decision, the required resources will be allocated to the IoT devices using a resource allocation algorithm.
4. The IoT device will send the task-related data to the allocated server.
5. Task execution will take place at the server.
6. The server will communicate the result of the task execution to the IoT device.

6.2 Module 2: Spectrum Management Module

Due to increasing number of IoT devices, simultaneous access to limited spectrum resource is very challenging. One of the ways to deal this spectrum scarcity is that all the IoT devices are capable of spectrum-sensing functionality which requires dedicated spectrum-sensing hardware and computing capabilities, which seems difficult in the IoT environment. On the other hand, one can utilize the edge computing infrastructure for effective spectrum management, where the edge nodes will assist resource-constraint IoT devices in spectrum access as shown in Fig. 5. The geographical area of interest can be divided into sectors and every sector will have at least one edge node. An edge node maintains spectrum usage database of its sector. The network-wise global distributed edge spectrum database can be controlled by remote cloud.

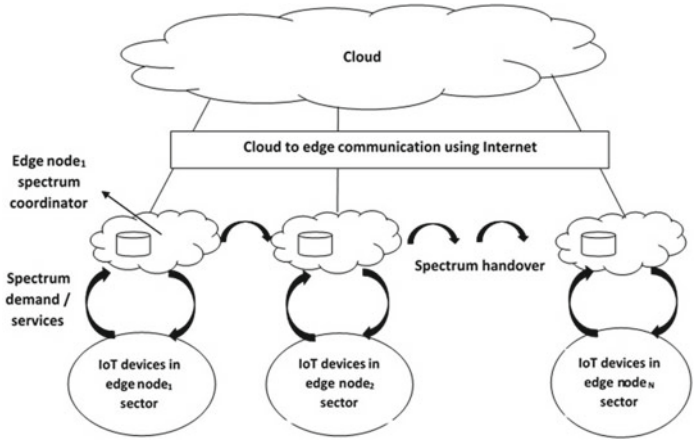


Fig. 5 Edge-assisted spectrum management module

Table 4 Standardized QCI

QCI	Service type	Priority	PDB(ms)	PER	Example service
1	GBR	2	100	10^{-2}	Conversational voice (VoIP)
2	GBR	4	150	10^{-3}	Conversational video (live streaming)
3	GBR	5	300	10^{-6}	Non-conversational video (buffered streaming)
4	GBR	3	50	10^{-3}	Real-time gaming
5	Non-GBR	1	100	10^{-6}	IMS signaling
6	Non-GBR	7	100	10^{-3}	Voice, video (live streaming), interactive gaming
7	Non-GBR	6	300	10^{-6}	Video streaming (buffered streaming)
8	Non-GBR	8	300	10^{-6}	TCP based (e.g., www, email), chat, FTP, p2p file sharing
9	Non-GBR	9	300	10^{-6}	TCP based (e.g., www, email), chat, FTP, p2p file sharing

Further, based on different types of IoT services, different priorities can be assigned to them using service characteristics such as requested bandwidth, maximum tolerable duration, service arrival rate, etc. For an example, standardized Quality-of-Service Class Identifier (QCI) [11] is shown in Table 4.

The quality of service (QoS) is an important factor in spectrum resource management. AS per the QoS requirements, connections are classified into various QoS classes. In 3GPP LTE specifications, nine different services have been defined.

Factors such as packet delay, L2 packet loss rate, and guarantee bit rate (GBR) are used to specify class of QCI services. The priority to the connections is given as per QCI services.

The spectrum management module considers different attributes of idle spectrum band such as error rate, available time, bandwidth, and coverage area, etc. Then, a multi-criteria decision-making (MCDM) approach can be used to select the optimal spectrum band for a IoT service.

6.3 Module 3: Security and Privacy Module

In applications like smart health care, security and privacy module is very important. This module should contain security and privacy solutions corresponding to different threats/issues, which are listed below [12]:

1. *Threat to Confidentiality*: Confidentiality of the data is utmost important for the edge-assisted smart cloud healthcare system. Many cyberattackers use different network sniffer tools like wireshark, network miner, etc. to capture and analyze the data in transit. The health-related data of an individual is very much sensitive and should not be leaked. There are two possible communications where confidentiality of the data is in threat in the smart healthcare application (shown in Fig. 4). Firstly, when the biomedical sensors interact with the edge node to offload their tasks. Secondly, when the edge nodes communicate among themselves for efficient management of resources. Generally, the biomedical sensors have no idea about the edge nodes because whenever it is required they can start communication with any of the edge node. The mapping of biomedical sensor to edge node is not static. Therefore, symmetric key encryption mechanism is not a feasible solution to provide confidentiality to data in transit between the biomedical sensor and the edge node. So, one of the challenges of this module is to provide an efficient public key-based encryption mechanism to withstand against the eavesdropping attack. Simultaneously, another aim should be to minimize the overhead of the message due to the encryption mechanism. So, the encryption mechanism be lightweighted. The solution should not have any computationally intensive cryptographic operations like bilinear pairing and modular exponentiation [13]. Cryptographic operations like secure hash, exclusive, elliptic curve multiplications, etc. can be explored for developing the solutions. On the other hand, communication among the edge nodes need to be secured. This process can start with establishment of secure key exchange and secure data connection. This can provide additional security to the data communicated using Constraint Application Protocol (CoAP). As a result, the communications are resistant to man-in-the-middle attack (MITM). The encrypted data should be divided into several fragments and can be shared with the intended nodes through different routes.

Another important solution for handling such threat is by using attribute-based encryption (ABE)-based access control mechanisms. In this kind of encryption mechanism, ciphertext as well as the secret key of a user depends on the attribute of the user. One user can decipher a particular cipher text only when the attributes of the ciphertext matches with the attributes present in the secret key of the individual user [14]. Using ABE, data owners can attach access policy (who can decrypt the ciphertext and who cannot) to the ciphertexts. So, using this mechanism, threat to confidentiality can be avoided in the smart healthcare system. The encrypted data which is stored in either edge node/cloud can only be decrypted by the entity who has the corresponding attribute. So another task is to develop a latency-aware fine grain ABE mechanism having following requirements:

a. The task of creation of the access policy should not be provided to the resource-constrained devices as the process of creation of access policy involves computationally intensive operation. This task can be provided to the nearest edge node.
b. Backward secrecy and forward secrecy have to be maintained. Due to this, the entity whose attribute is revoked cannot decrypt the newly encrypted data and whose attribute is added cannot decrypt the previously encrypted data, respectively.
c. The mechanism should be scalable. As the number of biomedical sensors and IoT devices will increase day by day, more number of edge nodes will behave as attribute authority and need proper synchronization and collaborate among them to create access policy.

2. *Verifiable Computation*: In some application, computation over the data is required. For example, one NGO may need the information regarding the total number of patients who are having normal cardiac behavior within the range of certain age group. But, it becomes challenging when the cardiac behavior data of the patients are stored in encrypted format (to preserve the confidentiality of the health data) [15]. In such cases, at the same time, we have to preserve the privacy of the individual patient and also to perform computation over the data [16]. A diagrammatic representation of the same is shown in Fig. 6. The next task should be to develop a lightweight computation scheme when the data are encrypted with individual data owner's key [17]. This will have the following requirements:

a. Each patient's data has to be encrypted using their own individual key. Then, performing computation over the encrypted data, we need to convert the data encrypted using their own individual key to data encrypted using computation server's key. Here, computation server will act as a proxy whose job is to convert the ciphertext encrypted with one key to ciphertext encrypted with another key. The process of conversion of ciphertext is secure so that the proxy server can't know the data it is converting.
b. Once all the required data needed for computation are encrypted under the same computation server's key, crypto-systems like Paillier and Elgamal can be used to perform required computation over the encrypted data.

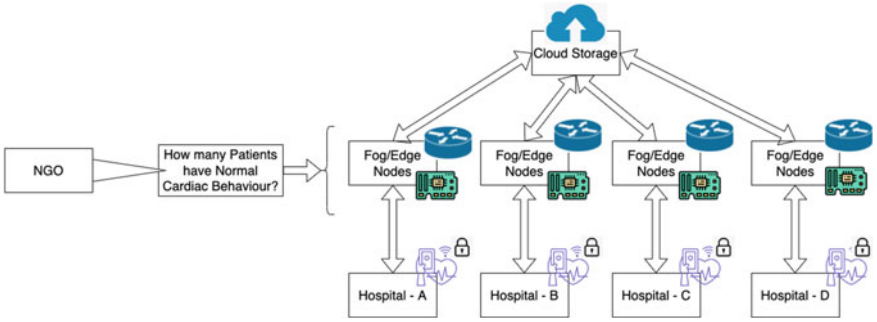


Fig. 6 Verifiable computation over encrypted data

c. Once the computation over the encrypted data is done, the result would be in the encrypted format. Next job is to convert the result encrypted under the computation server's key to the result encrypted under the NGO's key. Then only, NGO can decrypt the result and see the required computation result. This task can be done using the same proxy re-encryption mechanism discussed in Step (a).
d. The NGO can get the decrypted computation result by decrypting the encrypted result with its private key.

If a task like computing one function over the data of several patients is offloaded to the edge node, then how can the doctor or the NGO be sure about the correctness of the result returned by the edge node. The edge nodes are untrusted servers, so they have to provide the result of the computation as well as a verification proof of computation result. The verification is needed to be done in an efficient way. Simply, computing the result again and then matching both the results is not practically possible in case of edge node. The reason for which edge node is outsourcing is that they cannot perform the computation on their own. So, another objective of this module should be to develop a lightweight verifiable computation scheme where

a. The data on which the computation is to be performed is kept secret from the edge node and therefore the data need to be encrypted before communication.
b. The verification algorithm should support batch verification. It means that different verification requests from different users need to be batched together. One verification equation can be used instead of many for the verification of the data. The functions on different data can be batched together and using one equation/one step we should be able to verify the correctness.

Medical history of several patients is helpful in providing accurate diagnosis for new patients. But, utmost priority is to preserve the privacy of the patient is very important. So another issue of this module is to provide privacy-preserving machine learning algorithms (to learn from the data of several patients to come

to a conclusion which is a collaborative decision) when the data are encrypted. The inherent machine learning algorithms should not disclose the identity of the individual patients, at the same time the privacy of the features should be preserved. So, neither the features is revealed to the participating patients nor the secret medical data of each patient is provided to the analyst. So, to address such problems, intersection of two research fields machine learning and security is needed.

3. *Threat to Integrity and Authentication*: During the transmission of health-related data from the biomedical sensors to the edge node, if some portion of the data is modified or deleted, erroneous conclusion or wrong diagnosis result may come. In turn it may be life threatening for patients. Therefore, integrity of the data must be preserved, during the communication of the data [18, 19]. So, one of the important issues is to propose a privacy-preserving lightweight and efficient integrity verification technique. This technique should have following steps:

   a. The edge nodes should digitally sign the analysis of the health-related data with their private key before sending to the doctors. Here, some secure hashing techniques should be explored to preserve the integrity of the health data.
   b. The doctors use a verification algorithm which will take public key of the edge node and the signed data sent in step a to check the integrity of the data.
   c. The verification algorithm should support batch verification. It means that different requests from different edge nodes can be grouped together and the verification can be simplified. Here, using one equation/one step, one should be able to verify the correctness.

   The identity of the IoT devices must remain intact throughout its working period. The identity must not be a false identity or stolen identity. If the identity of the IoT device is compromised then the received messages will be forged messages and the computation or analysis over those data will lead to faulty or erroneous conclusions. Due to the vast heterogeneity of the vendor landscape of IoT devices, challenges arise regarding the ability to identify devices and their core functionalities in a uniform and trustworthy way, as many manufacturers implement vendor-specific solutions for device identities in their products with varying levels of trustworthiness [20]. So, the next issue is to propose a lightweight mutual authentication scheme that should resist identity theft and have following features:

   a. Elliptic curve cryptography (ECC) and secure hash algorithms should be used so that higher security can be achieved with smaller key sizes [21].
   b. The scheme should be resistant against attacks like reply attack, meet-in-the-middle attack, user impersonation attack, insider attack, dictionary attack, and fog node impersonation attack.

   Nowadays, Blockchain has become a nice alternative for the verification of the integrity of the data. The data that is stored in the blockchain can easily be verified and tampering with the data stored in the blockchain is very difficult. This happens as the blockchain is a distributed platform and the same copy of the data maintained at several peers. In this context, smart contracts are also helpful. It is used to

automatically run a certain program if some condition is met. This can act as an important direction for resolving threats to integrity and authentication.

4. *Threat to Privacy*: To provide privacy-preserving solutions for the smart healthcare systems, we need to address the following privacy issues while developing/proposing any security framework for solutions with respect to the issues mentioned in Modules (1) and (2):

   a. Generally, the location information of the patient is very sensitive. This information should not be collected until and unless it is definitely required for the medical diagnosis. Unnecessary collection of location information may lead to few privacy threats like determination of personal behavior pattern (can be used by marketers, government), determination of specific appliances used (can be used by insurance companies), can perform real-time surveillance (can be used by law enforcement and press), and target home invasions (can be used by criminals).
   b. The identity of the patient has to be preserved. Generally, the identity of the person involves name, address, email ID, mobile number, etc. If these information are reached to an attacker, then they can launch social engineering attacks. These information will only be collected when it is required.

   A summary of different research contributions is provided in Table 5 covering task offloading, spectrum assignment, security, and privacy issues.

7 Challenges and Future Research Directions

Despite the fact that a lot of research work have been done so, there are still many important issues that need exploration. Below, we discuss open research challenges and future research challenges in computing domain.

- Research directions focusing on computation offloading and service migration: since user devices are resource constrained, task processing involves cooperation among multiple edge nodes and user devices. This is called computation and service migration. There are mainly following steps: computation migration decision, task division for migration, task uploading, task computing/executing, and result is sent back to user device and result aggregation. Among all, task division and migration decision are the two most critical steps. Future research can focus on the minimizing total cost for implementing computation migration. Further, the concern whether the integrated results are the same as those of none-partitioning processing may arise during the process of offloading? This concern leads to a future study on how to partition/integrate so that result is nearly same as if they are processed locally. Studies related to computation offloading and relevant parameters are discussed in Table 5.

Table 5 Summary of research contributions

Paper	Objective	Research content
[22]	Delay, energy consumption	(a) Binary offloading decision, (b) transmission power allocation
[23, 24]	Energy, consumption	(a) Binary offloading decision, (b) task destination association
[25]	Utility	(a) Binary offloading decision, (b) task destination association
[26]	Revenue	(a) workload decision for partial task offloading, (b) task destination association, (c) energy harvesting
[27]	Delay	(a) workload decision for partial task offloading, (b) task destination association
[28]	Execution time	(a) sub-task placement, (b) schedule of the IoT tasks
[29]	Energy consumption	(a) auction-based approach, (b) consider communication cost only, (c) does not consider computing and storage cost
[30]	Delay and energy consumption	(a) consider computing cost only, (b) does not consider communication and storage cost, (c) based on non-dominated sorting genetic algorithm
[31]	Delay	(a) consider computing cost only, (b) does not consider communication and storage cost, (c) based on game theory
[32]	Cost	(a) consider computing and communication costs, (b) based on many-to-one matching algorithm
[33]	Revenue	(a) consider computing and communication costs, (b) based on alternating direction method of multipliers
[34]	Utility	(a) consider computing, communication, and storage costs, (b) based on alternating direction method of multipliers
[14]	Encryption and access control	Ciphertext policy-based ABE with fine-grained access mechanism
[13]	Encryption	Modular encryption Standard (MES) based on the layered modeling
[15]	Secure computation	Efficient fully homomorphic encryption with the help of twin key encryption and magic number fragmentation
[16]	Secure searchable encryption	Autonomous path delegation
[20]	Authentication	(a) Advanced signature-based encryption (b) Blockchain technology

- Heterogeneity: In general, the architecture of edge computing generally includes device layer, edge layer, and cloud layer. Many heterogeneous nodes such as end devices (with different specifications, make, and model), edge servers increase the complexity of resource management.
- Security and privacy: The requirements of security services in edge computing infrastructure are discussed in Subsect. 6.3 (Module 3: Security and Privacy). However, more dedicated research in the direction of trust mechanisms and privacy preservation policies need to be carried out.
- User's mobility: Due to frequent movement of users, task offloading decision and cache provisioning may not be optimal after few seconds when a user has moved out of range of current assigned server. So, considering mobility pattern of users is an important parameter while taking offloading decisions. This will provide seamless services and improves experience of users to access edge services.

Tools for evaluation of algorithms: There are simulators such as iFogSim [35], EdgeCloudSim [36], MyiFogSim [37], and MATLAB that are being used for evaluation of task offloading algorithms in edge computing. Few studies evaluate their algorithms in real edge systems.

Tools for evaluation of security mechanisms: There are various tools/simulators which can be used to test the solutions that address the issues stated above. Few such tools or simulators are PBC Library [38], MIRACL crypto SDK,[1] Thinkercad,[2] ThinkSpeak,[3] Amazon Cloud Service,[4] AVISPA,[5] etc.

Acknowledgements This work was supported by the SERB-DST, Government of India under Grant SRG/2020/000575.

References

1. Mell P, Grance T (2011) The nist definition of cloud computing (draft), nist spec. Publ 800:145
2. Shaw SB, Singh AK (2014) A survey on cloud computing. In: 2014 international conference on green computing communication and electrical engineering (ICGCCEE), pp 1–6
3. Wang X, Han Y, Leung VCM, Niyato D, Yan X, Chen X (2020) Convergence of edge computing and deep learning: a comprehensive survey. IEEE Commun Surv Tutor 22(2):869–904
4. Luo Q, Hu S, Li C, Li G, Shi W (2021) Resource scheduling in edge computing: a survey. IEEE Commun Surv Tutor
5. Lin L, Liao X, Jin H, Li P (2019) Computation offloading toward edge computing. Proc IEEE 107(8):1584–1607
6. Du J, Zhao L, Feng J, Chu X (2017) Computation offloading and resource allocation in mixed fog/cloud computing systems with min-max fairness guarantee. IEEE Trans Commun 66(4):1594–1608

[1] https://miracl.com/.

[2] https://www.tinkercad.com/dashboard.

[3] https://thingspeak.com/.

[4] https://aws.amazon.com/.

[5] https://www.ercim.eu/publication/Ercim_News/enw64/armando.html.

7. Yang Z, Pan C, Wang K, Shikh-Bahaei M (2019) Energy efficient resource allocation in UAV-enabled mobile edge computing networks. IEEE Trans Wirel Commun 18(9):4576–4589
8. Peng G, Wu H, Wu H, Wolter K (2021) Constrained multiobjective optimization for IoT-enabled computation offloading in collaborative edge and cloud computing. IEEE Internet of Things J 8(17):13723–13736
9. Sample AP, Smith JR (2013) The wireless identification and sensing platform. In: Wirelessly powered sensor networks and computational RFID. Springer, pp 33–56
10. Moura J, Hutchison D (2018) Game theory for multi-access edge computing: survey, use cases, and future trends. IEEE Commun Surv Tutor 21(1):260–288
11. Mamman M, Hanapi ZM, Abdullah A, Muhammed A (2019) Quality of service class identifier (QCI) radio resource allocation algorithm for LTE downlink. PloS One 14(1):e0210310
12. Lin W, Xu M, He J, Zhang W (2021) Privacy, security and resilience in mobile healthcare applications. Enterprise Inf Syst 1–15
13. Shabbir M, Shabbir A, Iwendi C, Javed AR, Rizwan M, Herencsar N, Lin JC-W (2021) Enhancing security of health information using modular encryption standard in mobile cloud computing. IEEE Access 9:8820–8834
14. Satar SDM, Hussin M, Hanapi ZM, Mohamed MA (2021) Cloud-based secure healthcare framework by using enhanced ciphertext policy attribute-based encryption scheme. Int J Adv Comput Sci Appl 12:393–399
15. Kara M, Laouid A, Yagoub MA, Euler R, Medileh S, Hammoudeh M, Eleyan A, Bounceur A (2021) A fully homomorphic encryption based on magic number fragmentation and el-gamal encryption: smart healthcare use case. Expert Syst e12767
16. Wang Q, Lai C, Lu R, Zheng D (2021) Searchable encryption with autonomous path delegation function and its application in healthcare cloud. IEEE Trans Cloud Comput
17. Nayak SK, Tripathy S (2021) SEMKC: secure and efficient computation over outsourced data encrypted under multiple keys. IEEE Trans Emerg Topics Comput 9(1):414–428
18. Yang X, Yi X, Nepal S, Khalil I, Huang X, Shen J (2021) Efficient and anonymous authentication for healthcare service with cloud based WBANs. IEEE Trans Serv Comput
19. Nayak SK, Tripathy S (2016) Privacy preserving provable data possession for cloud based electronic health record system. In: 2016 IEEE Trustcom/BigDataSE/ISPA, IEEE, pp 860–867
20. Shukla S, Thakur S, Hussain S, Breslin JG, Jameel SM (2021) Identification and authentication in healthcare internet-of-things using integrated fog computing based blockchain model. Internet of Things 15:100422
21. Nayak SK, Mohanty S, Majhi B (2017) CLB-ECC: certificateless blind signature using ECC. J Inf Process Syst 13(4):970–986
22. Fang J, Shi J, Lu S, Zhang M, Ye Z (2021) An efficient computation offloading strategy with mobile edge computing for IoT. Micromachines 12(2):204
23. Gu B, Zhou Z, Mumtaz S, Frascolla V, Bashir AK (2018) Context-aware task offloading for multi-access edge computing: matching with externalities. In: 2018 IEEE Global Communications Conference (GLOBECOM), IEEE, pp 1–6
24. Yang L, Zhang H, Li M, Guo J, Ji H (2018) Mobile edge computing empowered energy efficient task offloading in 5G. IEEE Trans Vehic Technol 67(7):6398–6409
25. Liu B, Cao Y, Zhang Y, Jiang T (2020) A distributed framework for task offloading in edge computing networks of arbitrary topology. IEEE Trans Wirel Commun 19(4):2855–2867
26. Chen W, Wang D, Li K (2018) Multi-user multi-task computation offloading in green mobile edge cloud computing. IEEE Trans Serv Comput 12(5):726–738
27. Ning Z, Dong P, Kong X, Xia F (2018) A cooperative partial computation offloading scheme for mobile edge computing enabled internet of things. IEEE Internet of Things J 6(3):4804–4814
28. Shu C, Zhao Z, Han Y, Min G, Duan H (2019) Multi-user offloading for edge computing networks: a dependency-aware and latency-optimal approach. IEEE Internet of Things J 7(3):1678–1689
29. Li L, Zhang X, Liu K, Jiang F, Peng J (2018) An energy-aware task offloading mechanism in multiuser mobile-edge cloud computing. Mob Inf Syst

30. Xu X, Liu Q, Luo Y, Peng K, Zhang X, Meng S, Qi L (2019) A computation offloading method over big data for IoT-enabled cloud-edge computing. Future Gener Comput Syst 95:522–533
31. Yu S, Langar R, Fu X, Wang L, Han Z (2018) Computation offloading with data caching enhancement for mobile edge computing. IEEE Trans Vehic Technol 67(11):11098–11112
32. Qian LP, Shi B, Wu Y, Sun B, Tsang DH (2019) Noma-enabled mobile edge computing for internet of things via joint communication and computation resource allocations. IEEE Internet of Things J 7(1):718–733
33. Wang Y, Tao X, Hou YT, Zhang P (2019) Effective capacity-based resource allocation in mobile edge computing with two-stage tandem queues. IEEE Trans Commun 67(9):6221–6233
34. Zhou Y, Yu FR, Chen J, Kuo Y (2017) Resource allocation for information-centric virtualized heterogeneous networks with in-network caching and mobile edge computing. IEEE Trans Vehic Technol 66(12):11339–11351
35. Gupta H, Vahid Dastjerdi A, Ghosh SK, Buyya R (2017) iFogSim: a toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments. In: Software: practice and experience, vol 47, no 9, pp 1275–1296
36. Sonmez C, Ozgovde A, Ersoy C (2018) Edgecloudsim: an environment for performance evaluation of edge computing systems. Trans Emer Telecommun Technol 29(11):e3493
37. Lopes MM, Higashino WA, Capretz MA, Bittencourt LF (2017) Myifogsim: a simulator for virtual machine migration in fog computing. In: Companion proceedings of the10th international conference on utility and cloud computing, pp 47–52
38. Lynn B (2006) PBC library manual 0.5. 11

Chapter 5
Fog Computing Infrastructure for Smart City Applications

Manju Lata and Vikas Kumar

Abstract Fog computing offers an integrated key in support of communications, data gathering, device management, services capabilities, storage, and analysis at the edge of the network. This allows the deployment of centrally managed infrastructure in an extremely distributed environment. The present work discusses the most significant applications of fog computing for smart city infrastructure. In the smart city environment running a lot of IoT-based services, computing infrastructure becomes the most important concern. Thousands of smart objects, vehicles, mobiles, and people interact with each other to provide innovative services; here the fog computing infrastructure can be very useful from the perspective of data and communication. The chapter focuses on three main aspects: (a) deployment of data and software in fog nodes (b) fog-based data management and analytics, and (c) 5G communication using the fog infrastructure. Working models in all the perspectives have been presented to illustrate these fog computing applications. Use-cases have been added from the successful implementations of smart city infrastructure. Further, the challenges and opportunities have been presented from the perspective of growing interest in smart cities.

Keywords Fog computing · Smart city · IoT · Smart lighting · Cloud computing

1 Introduction

Smart cities are the new fashion of the world. The idea behind smart cities is to provide services and improve the quality of life of the citizens, with the help of advanced technological reaches. Across the globe, large numbers of countries have arranged big projects to renovate most cities into smart cities. A lot of smart city

M. Lata
Chaudhary BansiLal University, Bhiwani 127021, Haryana, India
e-mail: manjulata94@gmail.com

V. Kumar (✉)
Central University of Haryana, Mahendergarh 123031, Haryana, India
e-mail: prof.vikaskumar@gmail.com

M. Chaturvedi et al. (eds.), *Recent Advancements in ICT Infrastructure and Applications*,
Studies in Infrastructure and Control, https://doi.org/10.1007/978-981-19-2374-6_5

development projects can be followed in Europe [14], Asia, Australia, China [19], and North America [8]. A lot of technologies are supporting the development of smart cities, and the Internet of Things (IoT) is a promising technology among them. IoT is constantly evolving and going to offer a lot of potential applications in near future [57]. IoT offers a cluster of elements, which are inherent or embedded by the sensors, stimulating power, software, actuators, and attached appliances. All of these are utilizing the Internet to draw together the data and restore data throughout the refine and additional devices [55]. This kind of structure offers a direct application in a smart city, which can be organized through several equipments and structures implemented by IoT applications. For example, cameras in the system of monitoring, sensors in the system of transportation, mobile phones in the system of communication, etc. The IoT makes use of the Internet to merge several heterogeneous devices or things [52]. Due to this, a smart city may consist of sensor networks and intelligent appliances to monitor them from remote places. For example, power usage monitoring, air conditioner management, light management, etc. All of these processes require faster access to data as well as faster processing. The latency in the communication network can lead to poor performance of the network, hence needs to be taken up in a coordinated manner. In order to support the low-latency access for IoT applications, fog computing can play a very vital role. With the integration of fog computing, IoT devices and sensors can be comprehensively deployed at several locations, and data processing and analysis can be carried out at a faster pace [7]. Integration of the fog nodes can also support the implementation of IoT applications with big data analytics. The key components of such a platform can be data management operations in fog nodes that consist of data pre-processing, prototype mining, classification, forecast, and visualization [56]. Additionally, IoT-based applications can do a lot of monitoring work in the smart city through the implementation of data analytics at the fog level [40], where edge gateways act upon pre-processing and data aggregation [46].

The smart city provides various community services, involving smart signals, smart houses, smart transportation, and smart medical services. With the integration of sensors, data, and algorithms, a smart city deploys a lot of related functions with the purpose of developing the life of residents [29]. Most countries are fairly in the early stages of implementing innovative technologies for design and implementation of smart cities. Apart from the indistinct procession, in 2018 Deloitte estimated that more than 1000 smart city developments have been done throughout the world, whereas 50% of this has been contributed by China [9]. The smart city market size is estimated at USD 83.9 billion in 2019 and this is expected to grow with a compound annual growth rate (CAGR) of 24.7% during 2020–2027 [18]. The AI, IoT, and 5G (AIoT5G) are the most significant technologies, which are expected to contribute to smart city development [12]. The expected smart city worldwide market revenue during 2020–2025 has been shown in Fig. 1, where smart infrastructure is the highest contributing segment to the worldwide smart city market revenues with estimated revenue of more than 70 billion USD in 2021.

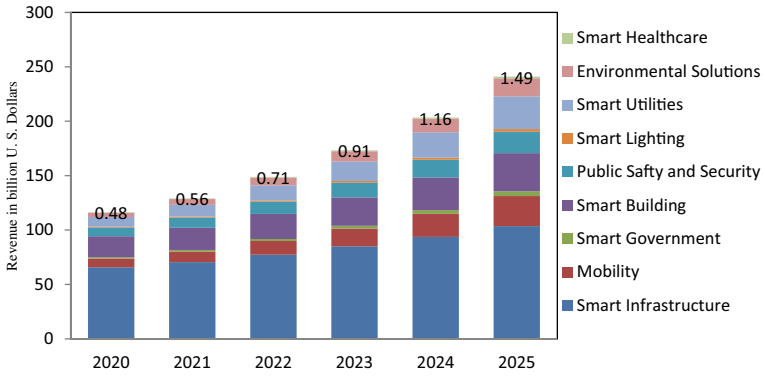


Fig. 1 Smart city worldwide market revenue growth (2020–2025)

2 Development of Smart City with Deployment of Data and Software

Many cities across the globe are considering the deployment of a smart city model to improve the quality of services being offered to the residents. A variety of software, tools, and technologies are responsible for sustaining the smart city [30], whereas the focus is on providing smart services to improve transportation, education, energy, healthcare, and other related areas [23]. Instances of these software and technologies are fog computing, cloud computing, Internet of Things (IoT), cyber-physical systems (CPS), wireless sensor networks (WSNs), robotics, big data analytics, and unmanned aerial vehicles (UAVs) [23]. The integration of these advanced tools and technologies is done to offer different levels of services in smart cities. For example, WSNs are used for real-time monitoring of the actions and conditions of smart city infrastructure [54]. The IoT makes easy integration of entities within the city network [59], where the cyber-physical systems are utilized to provide valuable communications involving the physical world and cyber world in smart cities [20]. UAVs and robotics are utilized to provide automation and functional services in support of smart cities [13]. Cloud computing offers the data storage platform, cost-effective, and scalable computation to sustain the smart city applications [10, 50], whereas big data analytics offers brilliant and optimized (short and long term) data-driven decisions toward the smart services [27]. Fog computing is utilized to offer location awareness, streaming and real-time support, low-latency support, and better mobility support in favor of smart city applications [16]. With the latest development in fog computing, service like DaaS have become possible to support the storage of data and deployment of software on the fog nodes.

Therefore, the dependence on high-speed data analysis becomes essential through the application, where data is able to be deployed on the fog node. However, time-sensitive analysis has the need for low latency and makes use of data stored in the fog node. At the same time, an increasing number of developments in favor

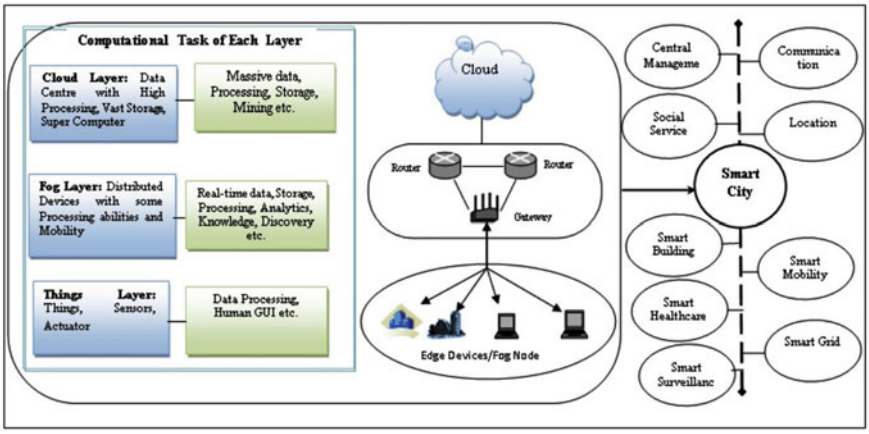


Fig. 2 Deployment of fog nodes in smart city

of transport and healthcare deployed through IoT-based resolutions also produce a huge amount of data. Thus, the proficient technique in favor of transferring a huge amount of data to fog servers is utilizing modern transfer technologies. For instance, IBM Aspera's immediate file transfer technology (Aspera FASP) surpasses the conventional file transfer technologies in terms of speed. A number of large corporations rely on this technology in support of the quick transfer of a huge amount of data [32, 35]. Therefore, great deal of software tools utilized by dissimilar city branches is considered in support of workstations. Thus, to store, share, and analyze big data by the way of low price and enhanced efficiency is very much important in smart cities. A large amount of smart city data can be shared with the public with a wide range of software applications through fog-based services.

High-level deployment of IoT-related data and software in fog nodes has been illustrated in Fig. 2 for smart city applications. Smart cities also exploit the design of new smart services utilizing the (a) data captured through the central management and power management to reduce human activity, (b) innovative distributed smart control algorithms, (c) detaining pattern identification of device connection, (d) communication and location, (e) controlling renewable power calculation, (f) utilizing power data into the security of services, etc. [23]. Deployment of data and software in fog nodes develops the smart city with energy-saving systems, continuance automation, security, etc. [52]. To establish a correlation between the existing cloud-based systems and the large-scale Internet of Things (IoT) deployment, fog and edge computing is emerging as a promising paradigm for smart city development. The edge and fog computing contains scalability, locality, low latency, security, and privacy [1].

Thus, the notion of smart city deploys and implements the embedded devices, hardware to sustain the advanced algorithms, edge and fog node capabilities with dissimilar kinds of devices, and functionalities. Smart city interconnects to the things, fog node, and IoT technologies-based application that analyzes and obtains the data

to develop the new insights able to drive real-added value and better performance [22, 49]. The value-added approaches offer the capability of designing the subsequent system as well as intelligent services. These aspects mainly rely on the managing of heterogeneity of dissimilar types of related things and the interoperability resting on the potential of processing of data to offer intelligence [21, 51]. The key concept of edge and fog computing emerges as a paradigm to carry out the computations for IoT appliances and applications. This paradigm also supports the latest requirements for security and quality of services (QoS). In view of the smart city development, a lot of new features can be used to establish the new aspects [3, 15]. The following phases are used in the whole process:

Analysis: Experts from different backgrounds work in this phase. This includes both the technicians and managers from security, power, energy, and information and communications technology (ICT) domains. Experts discuss and identify the main processes that require control. ICT experts contribute as a link of integration in this procedure. This analysis becomes a very user-centered technique to outline the subsystem requirements.

Design: Three-level architecture design is used, which is based on the edge, fog, and cloud layers. Layered planning integrates the edge and fog level with techniques to offer interoperability among the subsystems along with an increase in smart services. The system utilizes the edge and fog paradigms with the integration of IoT devices, applications, and protocols in the Intranet, where the cloud layer communicates with the services to complete this layered architecture.

Execution with Data Analysis: The subsystems are implemented after installing and integrating all through the related phase. Services are derived from rules within every subsystem. The data produced through the things or objects are analyzed to propose automated and advanced services.

Establish: Expert rules are extended with direction for every subsystem. Subsequently, rules are installed with opinion or view processes. The automatic and modified rules are inferred using smart tools and techniques. A technique is derived from the user-centered method to design, improve, and validate innovative services in interoperability necessities.

An additional level of design can be used to suggest the node specifications and node requirements utilized for the management of things. Design of services and procedures is implemented in edge nodes or fog nodes. Every node should be specific to determine its services, internal functions, and communication. Processing and intelligence abilities are extended, where the data is achieved. Both edge and fog layers contain nodes that become close to data sensor, controllers, and actuators. This technique of edge and fog node is planned with two functionalities, where each layer is able to deploy a network of interrelated nodes that make possible the interoperability [15].

Functionality in Edge Layer: This layer deals with the functionality control and manages the software developed on embedded devices to connect the sensors or actuators. A number of advanced algorithms are installed on an edge node. Communication devices and interfaces are installed to enable integration into the local network.

Functionality in Fog Layer: The functionality in the fog layer is based on the storage, communication, configuration of the files, and the action of the observation on the local area network. Fog node processes the data into an IoT gateway, server, or other devices with capabilities of communication, processing, and storage. Integration of all different services is done at this level only. The fog layer device is also able to perform the edge node functionalities during services.

Additionally, three actions are also required to implement the edge and fog nodes [15, 23].

Communication and Connection Services: All devices should be in a similar network with interoperability features. Every part of sensors and actuators should be available to participate in the services. For example, reading the power parameters from remote places, environment-based conditions, and open weather predicted data on the Internet. Some other functionalities may include the security, interoperability, and reliability of connections. Thus the purpose is to offer seamless communication.

Controlling and Data Processes in Embedded Devices: In this process, data analysis services and essential control rules are implemented and the devices are able to develop certain novel capabilities. A lot of control and data processing happens in this process. This may include the calculations on the climatic data, analyzing power consumption, event recognition, pattern recognition, etc.

Developed Services on Gateways Nodes: This action deals with the advanced computing paradigms and IoT-based applications and devices with the communication protocols. Fog nodes carry out an intelligent analysis of data: store-filter-communicate the data. It also approves the control actions at lower levels or generates the information related to concerns toward the cloud services. Examples of the actions include: analyzing innovative patterns, smart detection, calculating power and water consumption, and other analytical services.

3 Fog-Based Data Management and Analytics

Smart city produces a huge amount of data continuously at growing rates. Data is produced from digital platforms and social media, devices and sensors, and from services and applications. Efficiently managing data is essential for getting a better life for smart city and for momentum and dynamics protection [2, 25, 38]. Smart city data produced in the city's operation perspective [48] is predictable as considerable features on behalf of the deployment of smart city. It is presently marked as a new area that appears in data economy. During smart city data economy, a novel production model makes use of the connected data to reveal the analytics that makes the city's future [39]. Particularly, the smart city data collected from the IoT-based infrastructure and analyzed by dissimilar techniques are able to mainly develop the number of monitoring and rejoin services and tasks (i.e., [11, 44]). Smart city data effect on various services in a variety of interdisciplinary fields are resource efficiency, smart transportation, and mob resource-based services [11, 60]. For instance, the transport management system (TMS) procedure is derived from the utilization

of actual data (like social media data on behalf of the detection of road accidents and traffic congestions) and innovative technologies (like smartphones and smart cars) that intend to put aside time and inhabitants' road-based protection [11].

The consequence of mob-sensing and big data go over the data resources, analytical advances, and application-related methods with the preface to the deployment and development of intelligent transportation system (ITS) services [53, 60]. Cisco declared that cities leveraging the data can achieve nearby 30% growing energy efficiency. Additionally, they also examined and exposed to about 4.9 billion interconnected objects that are accepted to attain over 50 billion by 2020 and in excess of 1.4 billion smartphones [4]. At the same time, as the radio frequency identification (RFID) market attaches a label worth $11.1 billion, 500 million motor vehicles are likely to be attached to the internet by 2020. Explicitly, in accordance with Statista, it was reported that 1.8 billion attached objects were contained by Smart City by 2015, where the value is usually estimated to attain 3.33 billion by 2018 [4].

However, the management and development of open datasets have become very serious in support of smart city when extending citizen engagement, data economy, and decision-making. Smart city data production sets are also a serious issue when it comes to informative patterns and detecting events in the city context. In consequence, smart cities utilize a variety of software tools and techniques in support of data analysis and operations. Cloud-based software as a service (SaaS) delivery model is suitable for smart cities [3]. This model provides the online software services with different software tools that are deployed on fog computing to make easy remote access on the internet or network. This makes it easy to access the newest version of the services without installing the fog-based services locally. Therefore, by the way of cloud-based data management proliferation and service-oriented architecture (SOA), solutions in the form of data as a service (DaaS) can be established [3]. DaaS has appeared as a data delivery model that relies on similar conceptions as other cloud delivery models like infrastructure as a service (IaaS), platform as a service (PaaS), and SaaS. DaaS platform is also a delivery model that delivers the data to end-users on demand-based despite the location [36, 41]. DaaS-based system's components consist of SOA as a core platform, data intake and process, applications, visualization dashboards, and microservices. The procedure of this delivery model in fog computing is increasing. DaaS-based resolution sustains the data delivery in a fog environment and allows the efficient transfer of data between the providers and end-users [45]. This data is very much significant for decision-making and downstream analysis. National Institute of Standards & Technology (NIST) has recommended the use of the following structure of fog nodes to support the data analysis and management [31, 42].

Public Fog Node: This node is considered to be designated for open use by the general public. For example, industry, government organization, institution of higher education, or an amalgamation of three entities. Public entities should be allowed to possess, control, and manage the node or cluster.

Private Fog Node: The private fog node is designed to be used by a single association or an entity with several customers. The association can possess the cluster,

control and manage it, or the management can be outsourced to the third party in full or partial mode.

Community Fog Node: Commonly, a fog node is designed for specific use by a particular community of customers belonging to several associations. One or more community associations are able to possess, control, and manage it. The management can also be assigned to a third party.

Hybrid Fog Node: This includes a combination of at least two special fog nodes (public, private, or community) that control and manage independently. The portability of data and applications among nodes (like fog satisfied in support of load balancing among fog nodes) made use of standardized or proprietary technology.

There are many successful applications to support the planning and conceptualizing of smart city development. Some innovative techniques and models have been applied in support of improving a city's operational areas; for instance, education, transportation, health care, etc. [34]. The following are some of the examples where useful information can be derived in order that visitors, citizens, companies, and local government are able to perform better [3, 5, 33].

(i) **Driving Protection Development Service**: This can be a service derived from analyses of driving data on commercial automobile drivers such as, taxi, bus, and truck drivers in support of citizens. Additionally, the data can be collected on the motor vehicle procedures or functional data commencing transportations by applying the digital tachograph (DTG) devices. This can be very much useful for the transportation authorities to enhance the transportation services. The intended service may monitor the protection of city local transport as well as monitor the involvement of the drivers and transport companies. However, the monitoring is mostly based upon the data collected through sensors across the cit.

(ii) **Sustainable Eco-Driving Services**: The case of sustaining eco-driving services can be derived from the analysis of data on driving as well as fuel burning. For example the fuel efficiency of public transport vehicles (like buses) is regularly poorer than that of other motor vehicle for the reason that buses become heavier with the weight of passengers and vehicle. Hence, the transport authorities are always concerned to enhance this efficiency. The efficiency may have strong relationship with the driving behavior of the driver as well. The specifically designed service examines the efficiency of fuel in buses and make them available to the knowledge based learning contents for the drivers towards the eco-driving. For example, when drivers make high or slow down quickly, an alarm of quick deceleration can be sent by in-vehicle device.

(iii) **Health Related Data Support**: Health related data based services are very much of interest to the whole world and in particular to the insurance services [28]. A variety of health related data can be collected like the data of medical examination, diagnosis, treatment, and insurance [6]. A number of notable projects have been reported from different places. One of the designed services is known as 'cloud family doctor' that sustains mutually the local family

doctors and citizens in assessing the individual health records of citizens and considerate the health utilizing the resources of local healthcare. One more designed service is also known as 'local health-nostics' offer prognostic and diagnostic health information, like syndrome maps as well as local health related data toward the local governments. A further designed service is also known as 'hospital QA' analysis the hospitals' service quality and makes available the information to patients as well as the family also to support them deciding sufficient local hospital.

(iv) **Hypertension Patient Management Service**: Hypertension patients are growing across the world. The governments are always concerned about the cost required to care for hypertension and require data based hypertension management toward the citizens. This data is also very much useful for the public health insurance system. So, a database collects the diagnosis concerning data, treatment concerning data, medical examination data, insurance related data, and hypertension inception data of citizens. This database can be used to develop a hypertension prediction model. Such a data driven model can help to recognize the high risk hypertension patients and also facilitate the automation of the services for users to visit public health centers in cities and obtain health checkup.

4 5G Communication Using the Fog Infrastructure

Fog computing facilitates autonomous management functionalities with the integration of 5G-enabled smart cities [37]. This becomes essential to adapt the available network technologies toward the upcoming requirements and design new concepts to sustain the severe requirements. A large amount of data produced from various sensors and actuators in the IoT has been a bottleneck for the traditional networks [26, 47]. This requires improvement in computing servers to manage the continuous increasing demands of smart devices. Fog computing follows a distributed structure to support efficient data sharing services. For instance, the concept of network function virtualization (NFV) in wide-ranging networks makes use of fog computing as an enabler toward 5G. Fog computing infrastructure is considered to enable autonomous management and orchestration (MANO) functionalities in support of smart city applications within the 5G networks [47]. MANO and NFV paradigms can be completely integrated and an autonomous fog node management system can be set up. Along with this, some new fog protocol can be established that allows replacing the application service information among fog nodes and the cloud level, getting the better performance of the application to resource-provisioning consequences. The concept of fog computing is able to compute the work close to the mobile device when computational sources are accessible. Allowing for the devices in a smart city includes high-speed Wi-Fi connections accessible through 5G networks. These devices are able to select whether to work together to compute the task nearby or transmit the task to several central access points where other resources are obtainable to compute

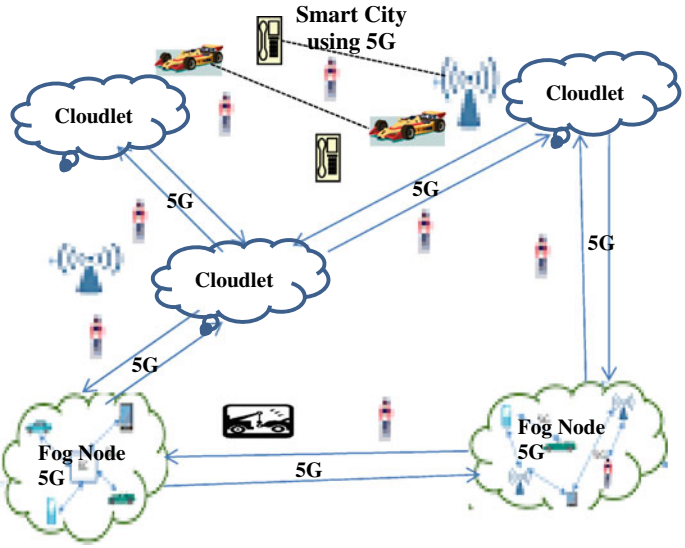


Fig. 3 Smart city fog-based model using fog nodes with 5G technology

the task. Figure 3 describes the smart city fog-based model using fog nodes with 5G technology.

The 5G technology with various networking devices can be used to support a range of sensors, automobiles, mobile users, cellular base stations, small fog node with particular servers, and other communication devices. The large number of applications associated with the devices need to be in communication with each other. All applications in this smart city development include dissimilar service requirements and 5G-based wireless networks that can sustain related requirements. Consequently, 5G can make the core networking requirements available in support of anywhere, anytime, and any network-related scenarios [3, 24]. 5G technology secures and evolutes the cellular network to interconnect people and control the machines, things, along with devices. Using the 5G networks in a smart city can provide zero latency, high-speed data transfer, and ubiquitous connectivity that can support a wide variety of high-performance applications with services [43]. Additionally, 5G provides the multi-Gbps data rates, huge capacity, ultra-low latency, and other constant user practices. For example, fog computing to sustain the 5G allowed Internet of Vehicles (IoV) that makes available the exact location address and service but the problem occurs related to the failure of the global navigation satellite system (GNSS) [58].

5 Discussion

Smart City wants to continue on the revolutionary track and require to jump into the reflective end of advanced IoT networks that systematically develop with fog computing. At the same time, it also facilitates managing data processing, makes available several benefits in support of cyber security, and also makes available actual financial savings. Furthermore, all the way through 5G connectivity still a bit away from the future, fog computing increases as a stimulating aspect in support of smart cities nowadays. Smart cities are quickly developing by technological modernization and proliferation of the IoT advances close to each aspect of city existence. Some years ago, cities simply had the smallest amount of embedded sensors, even though currently sensors are all over the place such as thermostats, roadways, trash bins, and others. This capability of connecting, communicating, and remotely managing a wide range of devices provided an increase to an innovative trend as IoT with fog computing in smart cities. In support of smart cities management, fog-based platform can be very much useful in implementing IoT with big data analytics. Data fusion can be created commencing the big heterogeneous datasets that have data combination, data aggregation, and data integration [40]. Consequently, the utilization of big data in smart cities is differentiated by various sets of dimensions together with the data, data collection technique, and value produced by data.

Integration of 5G in smart city development further enhances the sharing of data, exploits it in real time, aggregates it, and connects autonomously. Citizens of smart cities expect immediate or real-time information involvement practices. Hence, fog computing, being a distributed network model, communicates with the essential computing paradigm that plays a vital role to reduce the latencies of information processing and sharing in real time. Fog nodes using 5G technologies derived from ultra-low-latency abilities and high bandwidth offer great support to smart communication [17, 24, 37]. The transformative impact of 5G based on IoT, ultra-low-latency capabilities, and high bandwidth stimulates the appliances of financial growth and modernization, with cost optimization and produces the service quality developments. On the other hand, the smart city fog-based model with 5G technology makes certain the events, record measurements, warnings, and notifications that facilitate to identify policy violations, network failures, and security threats in a timely mode. Actually, this system maintains and modernizes the status of the overall fog computing infrastructure together with the fog node connected to the IoT devices. Fog node-based structural design predicts the fog decision component that houses the intelligence in a smart city mostly responsible for in support of

- Life-cycle control and decision-making
- Functionalities of self-configuration
- Concerns regarding the network behavior required by network managers
- Offering autonomous responses to indefinite conditions during data analysis monitoring.

Fog nodes can be completely integrated and autonomous management systems with the functionalities of decision-making and data analysis. Basically, the fog node manages the insignificant set of computational resources within the small cloud environments. This approach of fog node is able to consider the small cloud entity.

6 Conclusion

In smart cities, many smart objects work together to implement the complex emerging applications. Fog computing can play a vital role in supporting the emerging applications to run at the periphery of the Internet, offering flexibility and low latency. The deployment of fog computing in the smart city model can find a lot of interest, including the deployment of data and software in fog nodes, fog-based data management, analytics, and 5G communication using the fog infrastructure. These applications will certainly mature and lead to better developments in near future. At the same time, a more robust smart city infrastructure will be developed with better control of user security and privacy.

References

1. Ahmad S, Mazhar Afzal M (2020) Deployment of fog and edge computing in IoT for cyber-physical infrastructures in the 5G era, sustainable communication networks and application 39:351–359. ISBN: 978–3–030–34514–3
2. Anthopoulos L, Attour A (2018) Smart transportation applications' business models: a comparison. In: Companion proceedings of the the web conference, pp 927–928
3. Badidi E, Mahrez Z, Sabir E (2020) Fog computing for smart cities' big data management and analytics: a review. Future Internet 12(11)190:1–28
4. Bernard M (2017) Internet of things, facts everyone should read. Forbes. Retrieved from. https://www.forbes.com/sites/bernardmarr/2015/10/27/17-mind-blowing-internet-of-things-facts-everyone-should-read/#7694cbee3505
5. Bettencourt L (2010) Service innovation: how to go from customer needs to breakthrough services. McGraw Hill Professional
6. Bhardwaj A, Kumar V (2021) Electronic healthcare records: indian versus international perspective on standards and privacy. Int J Service Sci Manag Eng Technol (IJSSMET) 12(2):44–58
7. Botta A, De Donato W, Persico V, Pescapé A (2016) Integration of cloud computing and internet of things: a survey. Futur Gener Comput Syst 56:684–700
8. Chen C, Wang Z, Guo B (2016) The road to the Chinese smart city: progress, challenges, and future directions. IT Profess 18(1):14–17
9. Chou W, Ma C, Chung R (2018) Supercharging the smart city-smarter people and better governance. Deloitte Perspective
10. Clohessy T, Acton T, Morgan L (2014) Smart city as a service (ScaaS): a future roadmap for e-government smart city cloud computing initiatives. In: Proceedings of the 2014 IEEE/ACM 7th international conference on utility and cloud computing. IEEE Computer Society, pp 836–841
11. Djahel S, Doolan R, Muntean GM, Murphy J (2014) A communications-oriented perspective on traffic management systems for smart cities: challenges and innovative approaches. IEEE Commun Surv Tutor 17(1):125–151

12. Dublin (2020) Smart cities market by strategy, technology, and outlook for solutions, applications and services 2020–2025
13. Ermacora G, Rosa S, Bona B (2015) Sliding autonomy in cloud robotics services for smart city applications. In: Proceedings of the tenth annual ACM/IEEE international conference on human-robot interaction extended abstracts. ACM, pp 155–156
14. European (2019) European smart cities project. http://www.smart-cities.eu/. Accessed 05 Aug 2019
15. Ferrández-Pastor FJ, Mora H, Jimeno-Morenilla A, Volckaert B (2018) Deployment of IoT edge and Fog computing technologies to develop smart building services. Sustainability 10(11):3832
16. Giordano A, Spezzano G, Vinci A (2016) Smart agents and Fog computing for smart city applications. In: International conference on smart cities. Springer, pp 137–146
17. Gohar A, Nencioni G (2021) The role of 5G technologies in a smart city: the case for intelligent transportation system. Sustainability 13(9):5188
18. Grand View Research (2020) Smart cities market size, share & trends analysis report by application (smart governance, smart building, smart utilities, smart transportation, smart healthcare), by governance model, by region, and segment forecasts, 2021–2028. Published Date: May, 2021 Base Year for Estimate: 2020 Report ID: 978–1–68038–270–9, p 180. https://www.grandviewresearch.com/industry-analysis/smart-cities-market
19. Guo M, Liu Y, Yu H, Hu B, Sang Z (2016) An overview of smart city in China. China Commun 13(5):203–211
20. Gurgen L, Gunalp O, Benazzouz Y, Gallissot M (2013) Self-aware cyber-physical systems and applications in smart buildings and cities. In: Proceedings of the conference on design, automation and test in Europe. EDA Consortium, pp 1149–1154
21. Hui TK, Sherratt RS, Sánchez DD (2017) Major requirements for building smart homes in smart cities based on internet of things technologies. Futur Gener Comput Syst 76:358–369
22. Intel Building Management Platform (2018) Smart building with internet of things technologies. https://www.intel.com/content/www/us/en/smart-buildings/overview.html. Accessed 16 June 2018
23. Jawhar I, Mohamed N, Al-Jaroodi J (2018) Networking architectures and protocols for smart city systems. J Internet Services Appl 9(1):1–16
24. Khan MA (2019) Fog computing in 5G enabled smart cities: conceptual framework, overview and challenges. In: 2019 IEEE international smart cities conference (ISC2). IEEE, pp 438–443
25. Kitchin R (2014) The real-time city? Big data and smart urbanism. Geo J 79(1):1–14. https://doi.org/10.1007/s10708-013-9516-8
26. Klonoff DC (2017) Fog computing and edge computing architectures for processing data from diabetes devices connected to the medical internet of things. J Diabetes Sci Technol 11(4):647–652
27. Kumar V, Ayodeji OG (2020) Web analytics for knowledge creation: a systematic review of tools, techniques and practices, international journal of cyber behavior. Psychol Learning (IJCBPL) 10(1):1–14
28. Kumar V, Bhardwaj A (2020) Deploying cloud based healthcare solutions: a holistic approach. Int J Service Sci Manag Eng Technol (IJSSMET) 11(4):87–100
29. Lata M, Kumar V (2021a) IoE applications for smart cities. In: Internet of energy for smart cities: machine learning models and technique. CRC Press, pp 127–139
30. Lata M, Kumar V (2021b) Smart energy management in green cities. In: Handbook of green engineering technologies for sustainable smart cities. CRC Press, pp 105–120
31. Lata M, Kumar V (2021c) Standards and regulatory compliances for IoT. Int J Service Sci Manag Eng Technol (IJSSMET) 12(5):133–147
32. Lee J, Bagheri B, Kao HA (2015) A cyber-physical systems architecture for industry 4.0-based manufacturing systems. Manufacturing Letters 3:18–23
33. Lim C, Kim KJ, Maglio PP (2018) Smart cities with big data: Reference models, challenges, and considerations. Cities 82:86–99
34. Lim C, Kim K, Hong Y, Park K (2012) PSS board: a structured tool for product service system process visualization. J Clean Prod 37:42–53

35. López P, Fernández D, Jara AJ, Skarmeta AF (2013) Survey of internet of things technologies for clinical environments. In: 2013 27th international conference on advanced information networking and applications workshops. IEEE, pp 1349–1354
36. Mell P, Grance T (2019) The NIST definition of cloud computing. http://faculty.winthrop.edu/domanm/csci411/Handouts/NIST.pdf. Accessed 16 May 2019
37. Meng Y, Naeem MA, Almagrabi AO, Ali R, Kim HS (2020) Advancing the state of the Fog computing to enable 5G network technologies. Sensors 20(6)1754:1–26
38. Moustaka V, Vakali A, Anthopoulos LG (2017) City DNA: smart city dimensions' correlations for identifying urban profile. In: Proceedings of the 26th international conference on world wide web companion. ACM, pp 1167–1172
39. Moustaka V, Vakali A, Anthopoulos LG (2018) A systematic review for smart city data analytics. ACM Comput Surveys (CSUR) 51(5):1–41
40. Nguyen S, Salcic Z, Zhang X (2018) Big data processing in fog-smart parking case study. In: 2018 IEEE international conferences on parallel & distributed processing with applications, ubiquitous computing & communications, big data & cloud computing, social computing & networking, sustainable computing & communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom). IEEE, pp 127–134
41. Olson JA (2009) Data as a service: are we in the clouds? J Map Geogr Libraries 6(1):76–78
42. Oughannou Z, Kandrouch I, Chaoui NEH, Chaoui H (2021) Proposal architecture based on Fog Computing allowing real-time analysis in Smart Cities. In: 2021 7th international conference on optimization and applications (ICOA). IEEE, pp 1–7
43. Panwar N, Sharma S, Singh AK (2016) A survey on 5G: the next generation of mobile communication. Phys Commun 18:64–84
44. Patil CS, Pawar KN (2016) A review on: protocols and standards in different application areas of IoT. Int J Adv Res Comput Commun Eng 5(2):163–168
45. Plebani P, Salnitri M, Vitali M (2018) Fog computing and data as a service: a goal-based modeling approach to enable effective data movements. International conference on advanced information systems engineering. Springer, Cham, pp 203–219
46. Qin B, Tang H, Chen H, Cui L, Liu J, Yu X (2019) Review on big data application of medical system based on Fog computing and IoT technology. J Phys Conf Series 1423(1):012030. IOP Publishing
47. Santos J, Wauters T, Volckaert B, De Turck F (2018) Fog computing: Enabling the management and orchestration of smart city applications in 5G networks. Entropy 20(1):4
48. Schieferdecker I, Tcholtchev N, Lämmel P (2016) Urban data platforms: an overview. In: Proceedings of the 12th international symposium on open collaboration companion, pp 1–4
49. Siemens (2018) Digitalize your building. https://www.siemens.com/global/en/home/company/topic-areas/intelligent-infrastructure/buildings/digital-buildings.html. Accessed 16 June 2018
50. Singh J, Kumar V (2014) Multi-disciplinary research issues in cloud computing. J Inf Technol Res (JITR) 7(3):32–53
51. Stojkoska BLR, Trivodaliev KV (2017) A review of internet of things for smart home: challenges and solutions. J Clean Prod 140:1454–1464
52. Talari S, Shafie-Khah M, Siano P, Loia V, Tommasetti A, Catalão JP (2017) A review of smart cities based on the internet of things concept. Energies 10(4)421:1–23
53. Wang X, Zheng X, Zhang Q, Wang T, Shen D (2016) Crowdsourcing in ITS: The state of the work and the networking. IEEE Trans Intell Transp Syst 17(6):1596–1605
54. Watteyne T, Pister KS (2011) Smarter cities through standards-based wireless sensor networks. IBM J Res Develop 55(1.2):7–1
55. Yang Y, Wu L, Yin G, Li L, Zhao H (2017) A survey on security and privacy issues in internet-of things 4662:1–10
56. Yassine A, Singh S, Hossain MS, Muhammad G (2019) IoT big data analytics for smart homes with Fog and cloud computing. Futur Gener Comput Syst 91:563–573
57. Yousuf O, Mir RN (2019) A survey on the internet of things security. Inf Comput Security 27(2):292–323.© Emerald Publishing Limited, 2056–4961

58. Yu S, Li J, Wu J (2019) Emergent LBS: If GNSS Fails, how can 5G-enabled vehicles get locations using fogs? In: 2019 15th international wireless communications & mobile computing conference (IWCMC). IEEE, pp 597–602
59. Zanella A, Bui N, Castellani A, Vangelista L, Zorzi M (2014) Internet of things for smart cities. IEEE Internet Things J 1(1):22–32
60. Zheng X, Chen W, Wang P, Shen D, Chen S, Wang X, Yang L (2015) Big data for social transportation. IEEE Trans Intell Trans Syst 17(3):620–630

Chapter 6
Distributed Storage Infrastructure: Foundations, Analytics, Tools, and Applications

Yashwant Singh Patel, Pushkar Kumar, Ramnarayan Yadav, and Rajiv Misra

Abstract The rapidly shifting technology landscape has allowed organizations to acquire the benefits of streamlined processes and cost-efficient operations. However, the availability of data from sources such as social data, machine data, transactional data, and many more has become a game changer for businesses of all sizes. These large stores of data known as Big data is challenging for an organization to handle or process. To improve the IT operations and optimize the faster processing, enterprises have adopted cloud computing. The integrated model of Cloud and Big data is a powerful combination that can transform the IT operations of the organization. This chapter provides a systematic overview of distributed storage infrastructures and discusses the current state-of-the-art solutions for storage technologies using Big data and cloud models. This chapter investigates the foundations, tools, and open research challenges of storing data using distributed storage infrastructures.

Keywords Datacenter networks · Data analysis · Fast data retrieval · Distributed databases · Cloud service model · Big data applications

Y. S. Patel (✉)
Thapar Institute of Engineering and Technology, Patiala, India
e-mail: yashwant.patel@thapar.edu

P. Kumar
Indian Institute of Information Technology Ranchi, Ranchi, India
e-mail: pushkar.btech.ec17@iiitranchi.ac.in

R. Yadav
Institute of Infrastructure Technology Research and Management (IITRAM) Ahmedabad, Ahmedabad, India
e-mail: ramnarayan@iitram.ac.in

R. Misra
Indian Institute of Technology Patna, Patna, India
e-mail: rajivm@iitp.ac.in

M. Chaturvedi et al. (eds.), *Recent Advancements in ICT Infrastructure and Applications*,
Studies in Infrastructure and Control, https://doi.org/10.1007/978-981-19-2374-6_6

1 Introduction

Due to the speedy growth of smart and connected devices, i.e., sensors and smartphones, vast volume of information are generated every second. The term "Big data" was first described in a 1997 article of NASA researchers to define the ability to process massive amounts of information that may exceed the local disk, main memory, and even remote disk capacities [1]. In accordance with the National Institute of Standards and Technology's definition, "The Big data depicts the vast volume of data generated in the digitized, networked, and sensor-laden, information-driven world" [2]. The given interpretation points out to the enormous data volume from variety of data sources.

Big data provides data management technologies to support consortium to store, exploit, and handle huge volume of data [3]. On the other hand, the Big data growth [4, 5] encompasses the outbreak of photos, videos, unstructured, and structured text, social media, in addition to the data received via sensing devices. Some other problems such as searching, capturing, sharing, storage, data visualization, and analysis are among the many complex problems that go along with Big data.

In recent times, data paradigm has shifted from Megabytes (MBs) to Gigabytes (GBs) and Terabytes (TBs) but in modern days, data has been moved from Terabytes to Petabytes (PBs) and even Yotabytes (YBs) commonly named Big data. For example, at the level of ERP (Enterprise Resource Planning), data is represented in MBs but at the level of CRM (Customer Relationship Management) data is represented in the form of GBs. Moving further, web-level data is in the form of TBs that may include sites, servers, web logs, searching, networks, etc. Atlast due to abundant volume of data collected from live videos, sensors, GPS coordinates, images, and social networking, the data ordering has been shifted up to Petabytes and Yotabytes. The comprehension of Big data evolution is depicted in Fig. 1 [6].

Current researches in big data characterized it by using 5Vs namely volume, variety, velocity, veracity, and value. These 5V characteristics are presented in Fig. 2 [7].

Some other researchers in [8–11] extended these V's into heterogeneous Big data features comprising volatility, vulnerability, visualization, validity, and variability as listed in Table 1.

Due to the rapid increase in volume of data, there are several critical challenges in the data management's life cycle, preferably infrastructures of storage, query models, methods of data analysis, data sharing schemes, and numerous other sides. Management of Big data demands an inter-operable, and scalable architecture to efficiently perform storage and analysis of data. Cloud computing is undoubtedly a fine approach to support the Big data life cycle. It offers features of on-demand access, pooled resources, elasticity, and pay-per-use policy [12]. Technologies of Big data such as storage and analytics have further evolved and offer significant advantages with the integration of cloud computing. It offers the properties of pooled

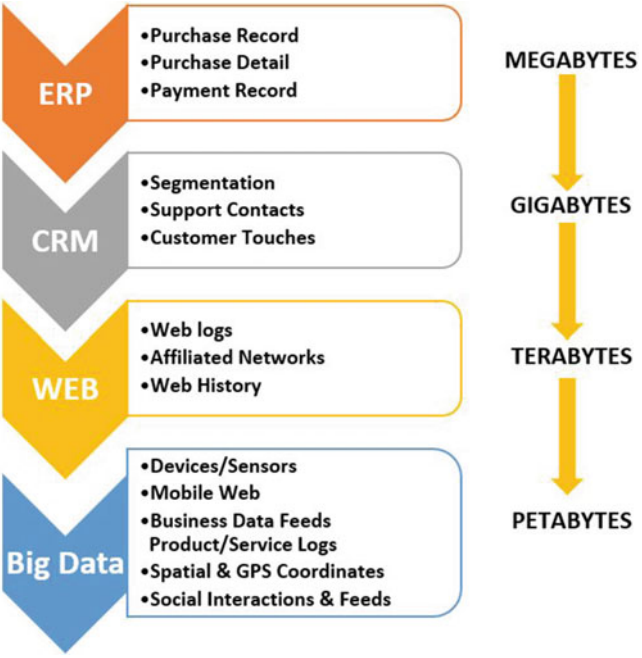


Fig. 1 Evolution of big data

resources, on-demand access, elasticity, and pay-per-use paradigms [12]. Integration of Big data storage and analytics technologies with cloud computing model provide various significant advantages [13, 14].

The importance of big data technologies integration with cloud computing models is attracting researchers' attention. In particular, Awaysheh et al. [14] provided a systematic review of big data architecture and environments for resource management. In the context of digital earth, Li et al. [13] investigated data analysis methods and architecture using cloud and big data processing. From security perspectives, Khan et al. [15] presented a detailed analysis and systematic approach to support software vendors' organizations for securing big data on the cloud platforms. To provide a systematic security analysis for Big data frameworks, Awaysheh et al. [16] designed novel security-by-design framework. Liu et al. [17] presented a critical review and taxonomy of stream processing systems to solve the resource management problem. Rana et al. [18] introduced an in-depth study on Industry 4.0 manufacturing by using the assimilation of cloud, IoT, and Big Data. However, still the practitioners and domain experts are not fully aware about cloud integrated big data frameworks and storage infrastructures from a network-based perspective. Therefore, to alleviate these research gaps, this chapter provides an extensive study of how cloud models enhance Big data technologies in reference to distributed storage infrastructure. This study also introduces Big data framework, Datacenter networks preliminar-

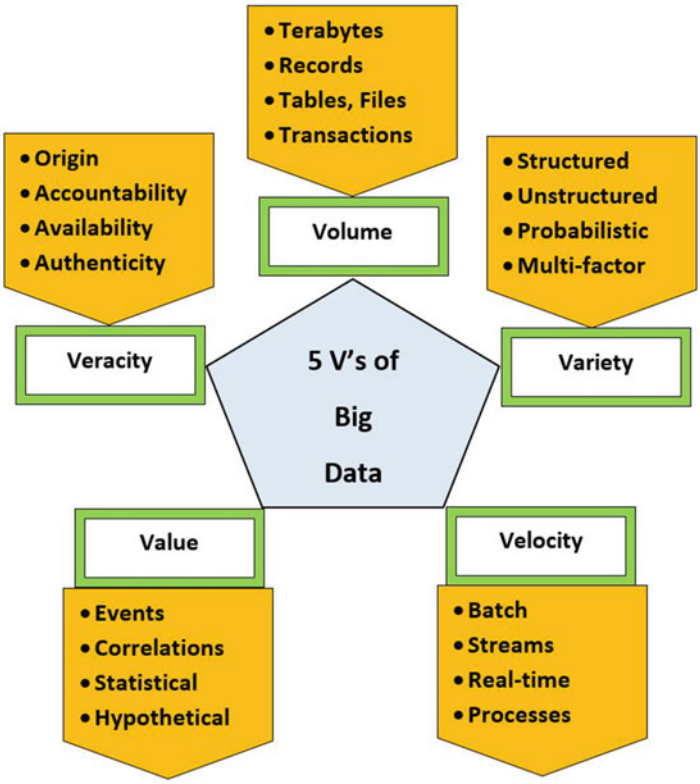


Fig. 2 The 5V's s of big data

ies and research issues, Protocols for faster data retrieval, proxy/cache mechanisms and investigates distributed databases to exemplify cloud-driven Big data computing models with reference to distributed storage infrastructure.

The remaining of this chapter is organized as follows: Sect. 2 provides a brief overview of Big data framework and storage technologies. To understand the data center networks, we have discussed its architecture along with issues and research challenges, types of flows, and topologies in Sect. 3. Moreover, we have provided a brief overview of traditional network and Software Defined Network (SDN) based technologies in Sect. 4. Distributed databases and different protocols for distributed storage infrastructures are presented in Sect. 5. Atlast, Sect. 6 concludes this chapter with future directions.

Table 1 Big data "9Vs"

"V"	Definition
Volume	Huge amounts of data that traditional data management technologies cannot easily store and process
Velocity	Speed of data creation, generation, storage, processing, analysis, and visualization
Variety	Different sources of data and their nature. The data is present in different formats including audio files, videos text files, images, PDFs, sounds etc.
Veracity	It defines the degree of trustworthiness, quality or accuracy of data sources
Variability	Refers to the data inconsistencies and also the inconsistent speed of loading the data into the database
Validity	Refers to the how accuracy of data
Value	Data value is often quantified as the potential scientific and social or economic value that the data might create
Vulnerability	Security concerns such as Big data breaches due to dark web
Visualization	How challenging the Big data is to visualize due to limited memory, functionality, and scalability concerns
Volatility	Represents the freshness and timeliness of data

2 Big Data Framework

Big data comprises new generation of storage and analytics that deals with structured and unstructured forms of data. Big data extracts the high-level information while enabling the high-velocity, high-volume, and high-variety features. The Big data framework is represented in Fig. 3 [19].

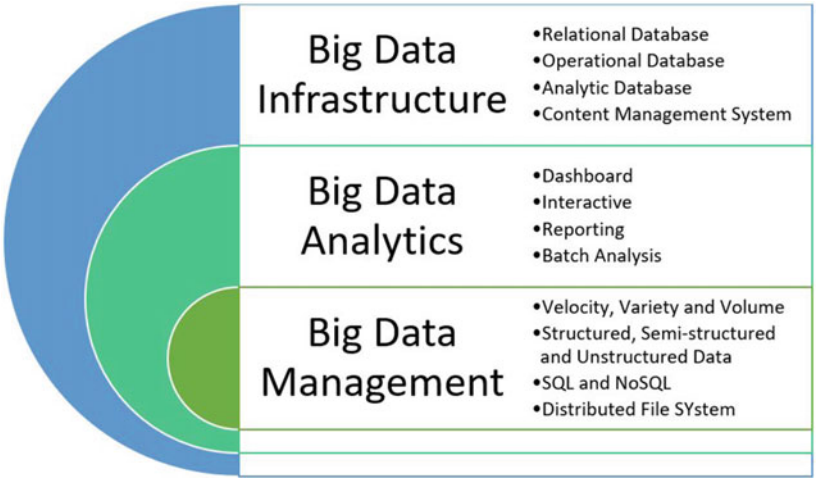


Fig. 3 Big data framework [19]

2.1 Types of Big Data

Understanding the source of raw data and how it has to be treated before entering into the analysis phase is very critical task to understand the different types of data. At different levels of analytics, Big data could be categorized into three different types named Structured Data, Semi-Structured Data, and Unstructured Data as represented in Fig. 4 with examples [6].

– Structured Data-
Structured data is highly organized with dimensions defined by set parameters. It indicates to the data that can be accessed, processed, and stored in fixed-format having a defined type, length, format, and design. Structured data is queried by using SQL: Structured Query Language. Such type of data can either be human or machine generated.

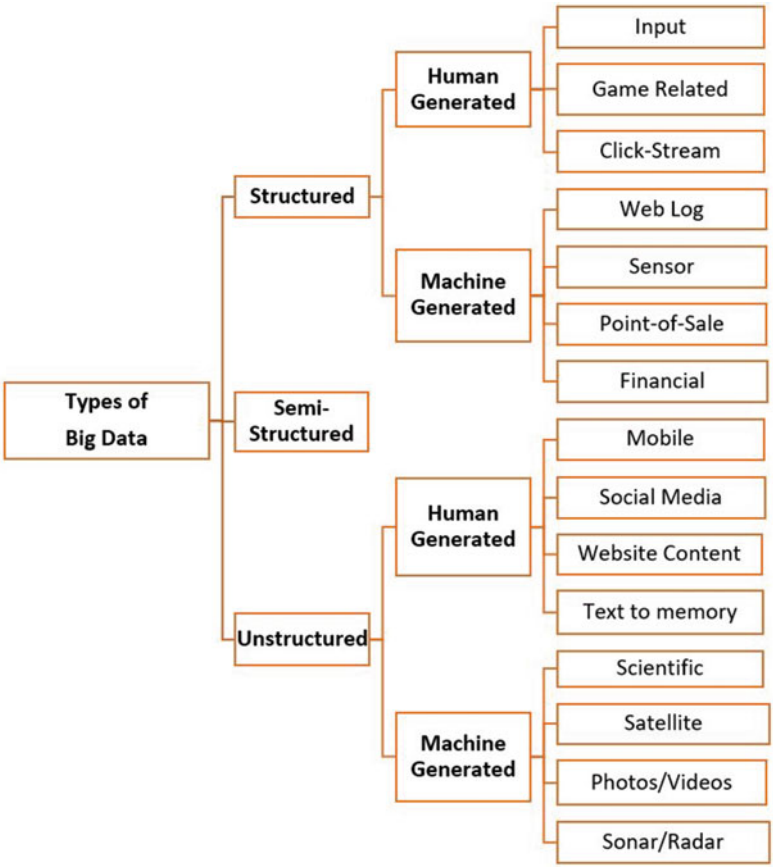


Fig. 4 Types of big data

- Human Generated Data
  Input Data—User given input such as income, age, debit/credit card numbers, name, address, Contact, etc.
  Gaming Related Data—Data related to game sessions which helps to obtain the broad understanding of players and their activities.
  Click-Stream Data—Clickstream data are a detailed log of how customers navigate through the Web site. The log typically helps to determine the user's behavior and their buying patterns.
- Machine Generated Data
  Web Log Data—Data of servers, application, networks, security threat prediction and handles the data related to service level agreements (SLAs).
  Sensor Data—Data comprising RFID Tag (Radio Frequency ID), Smart meters, smartphones, Medical Devices, and GPS (Global Positioning System), etc.
  Financial Data—Financial health and performance of company i.e., Stock Trading data.
  Point-of-Sale (POS) Data—Data collected by a business when a transaction happens, i.e., swiping the product's bar code.

– Unstructured Data—It depicts the data format of the relative multitude of unstructured files having different type, variable, length, and format. An illustration of this is an intricate data source comprising mix of images, text files, and videos. Unstructured data is really most of the data in this globe. The data can either be human or machine generated.

- Human Generated Mobile Data—Messages sent or received, GPS Information
  Social Media Data—Data collected from different social media platforms such as Facebook, WhatsApp, YouTube, Instagram, etc.
- Machine Generated Scientific Data-Data with geophysical parameters, such as atmospheric pressure, terrestrial surface measurements.
  Satellite Image-Satellite images(i.e., Google Earth) and weather data
  Videos and Photographs—Surveillance, security, and city traffic-related videos

– Semi-Structured Data—It toes the line between structured and unstructured data. Most of the time, this refers to unstructured data with some metadata attached to it. Such type of data can be collected from, location, device ID stamp, time, or email address, etc.

2.2 Concerns for Big Data Management

Processing and storing of massive data volumes need additional high-performance rental storage space with efficient utilization of resources. The integration of Big data and Cloud can manage such large-scale computations. The cloud offers high

performance and better scalability with unlimited rental storage. While the programming models of Big data such as Map-Reduce and Spark perform better with large data sets. However, Big data researchers and practitioners are still unaware of the potential applicability Big data computing models. This uncertainty is due to the lack of standard classification of frameworks from networking aspect.

2.3 Tools and Databases

There are three types of data primarily described in Fig. 4, and subsequently there are various storage methods too. Structured data are highly organized data and can be managed by programming language known as structured query language (SQL) [20]. Since it is an organized data, it requires less space for storage and thus can be stored and managed in tabular format like Relational Database Management System (RDBMS). While Unstructured data requires more storage space and since its shape is not defined it can't be managed by conventional databases. For unstructured data non-relational databases are a good-fit also known as NoSQL databases. Now on the basis of storage type NoSQL databases are differentiated into various types such as Cassandra (Column Storage), MongoDB (Document Storage), Redis (Key-Value storage), and Neo4j (Graph based storage). All these databases have their own advantages over each other depending upon the requirements like Cassandra [21] is for 100% scalability and availability, MongoDB having an additional advantage over Cassandra with high performance, Redis having in-memory storage and automated scaling, and Neo4j establishing graph-based relationship between data. Subsequently comes into picture the bridge between structured and unstructured data, i.e., semi-structured data which are difficult to be managed and stored in a relational database but has some organizational property that helps in easy management. It mostly arranges data into any kind of hierarchy based on metadata such as tags or some identifiers to separate semantic data. XML is widely used to store semi-structured data. Object Exchange Model (OEM) and database specially designed to store semi-structured data can also be used. In Table 2, we present comparative study of Big data tools.

3 Datacenter Networks

The advances in virtualization have made it possible to recognize the growth of Internet clouds as a novel computing paradigm. On-demand access is one of the essential characteristics of cloud problem. It is different from upfront cost, where you have to pay in advance and then use as per your requirement. In the industry terms, on-demand access can be classified as AAS classification as shown in Fig. 5. For example, HaaS: hardware-as-a-service, SaaS: software-as-a-service, PaaS: platform-as-a-service, and IaaS: infrastructure-as-a-service.

Table 2 Comparison of Big data tools

Tools	Database	Platform	Advantages	Limitations
MapReduce	Non-relational database	Cloud-based and open source	Performs well with unstructured and semi-structured data such as audio and visual data	Absence of indexing ability of modern database systems
Hadoop	Non-relational database	Cloud-based and open source	Ability to store data with any structure	Lacks security and technical support
Microsoft Windows Azure	Relational database	Public cloud-based platform	Permits users to build relational queries against unstructured, structured, and semi-structured files	Limited database size
Google Big Query	Columnar database	Cloud-based and open source and	Grants data to be replicated across distributed data centers	Indexes are not supported
Jaql	Query language	Proprietary query language	Supports semi-structured, and structured data	No user defined types
Cassandra	Column oriented database	Open Source NoSQL database	100 percent availability and scalability with flexible data storage	Partition size and value is limited
MongoDB	Document-based storage database	Source available cross platform	Easy to scale and uses internal memory for faster access data	Uses high memory and no transaction support
Redis	Key-Value based storage database	Fast, open source and data structure based platform	Flexible data structures, ease of use and in-memory fast computuation	Does not support relational algebra, query language or ad-hoc queries.
Neo4j	Graph-based storage database	Open source NoSQL database	lightning-fast read and write performance and easier to load data	limitation supporting number of Nodes, Relationships and Properties and doesn't support shrading

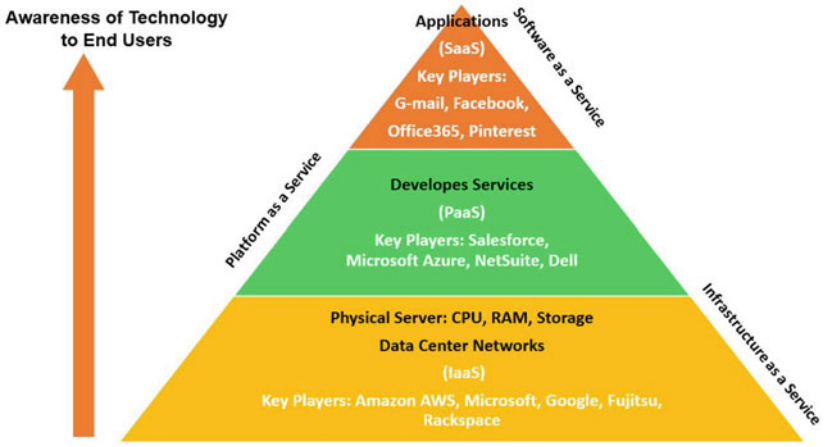


Fig. 5 AAS classification

3.1 Cloud Datacenter Architecture

This cloud lies within one premise known as the data center. At one site, the data center will house a lot of connected server racks. The rack comprises the compute nodes called servers, which are interconnected with each other. Switches connect these racks, and every rack has a top-of-rack (ToR) switch, which connects all the racks as shown in (Fig. 6). The typical network topology is hierarchical. Within the specific rack, the storage or back-end nodes are connected to the network. So, there are nodes even within the rack, which are primarily intended for storage purposes with solid-state drives (SSDs) within it, which are mostly used for storage. The front-end is designed for submitting the jobs and receiving client requests. This architecture can also be treated as a "three-tier architecture". In the three-tier architecture, one core switch is connected to all ToR switches. In such architecture, the software applications are used to run different applications.

3.2 Issues and Research Challenges in Cloud Data Center

Cloud offers cost-effective and flexible services for the on-demand provisioning of computational resources based on the subscription-based business model. However, to maintain geographically distributed cloud data centers (DCs) with several sites and heterogeneous inter-cloud infrastructures-as shown in Fig. 7, is challenging from networking, heterogeneity, multitenancy, pricing, workload management, availability, elasticity and mobility, and cloud security alliance perspectives [22–26].

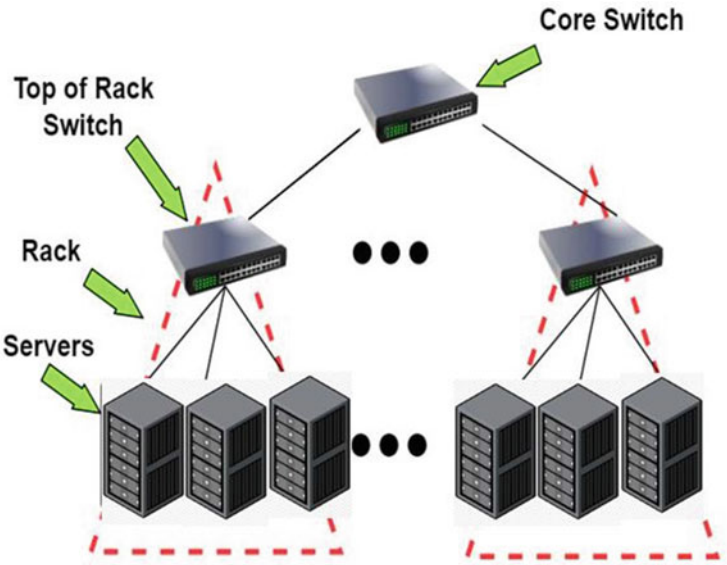


Fig. 6 Cloud data center architecture

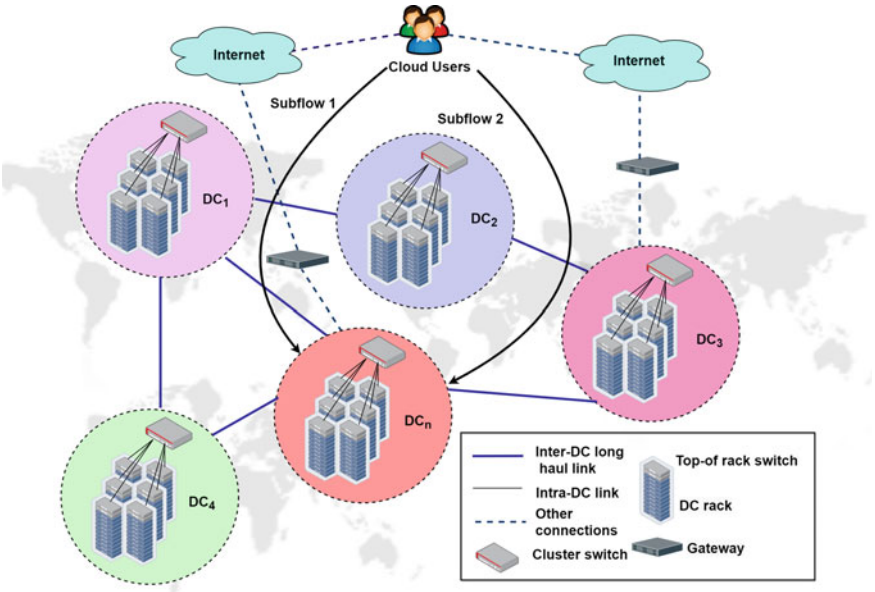


Fig. 7 Geo-distributed cloud data centers (DCs) [22]

Some of the open research issues in geo-distributed data centers are:

1. Heterogeneity and Right-Sizing: There is no "one-size-fits-all" kind of solution for all cloud applications, because several cloud applications share the infrastructure for running different workloads. To allow this, fine-grained access control of latency and availability are required. Applications with heterogeneous SLAs and workloads have different latency demands, or even multiple latencies and network goals for different client population [24].
2. Autonomous Management: Autonomous management within a single-owner infrastructure is crucial. It further increases the complexities in multi-owned data centers, where co-located multi-tenant facilities host one or several sites [23].
3. Workload Consolidation: It is a powerful optimization solution to reform the economics of virtualized data centers. It increases performance and decreases power consumption costs within a cloud data center. The inter-datacenter (inter-DC) based migration with networking constraints is a more novel research topic. It includes some extremely complex decision making during long-distance migrations such as, which site to select for migration, what portion of the workload is feasible for migration, how to reallocate the migrating workload in the destination site, and also to check whether the migration is beneficial [23].
4. Elasticity and Mobility: Software containers are widely used to handle and execute distributed applications. Containers are able to quickly scale the resources for computation with the help of vertical and horizontal elasticity. However, considering the presence of fog and edge computing resources, modern solutions are required that can deploy containers along with their allocation on decentralized resources [25].
5. Pricing: In federated cloud environment, multiple service providers collaborate together to share their computing resources to achieve the high profit and QoS. However, there can be a scenario, when the federations adopt different pricing models. In such cases, it could be difficult for cloud users to pick a satisfactory federation who can deliver the services at a reasonable price. Therefore, a fair resource assignment policy is needed for federated cloud environment [24, 27].
6. Latency and Availability: To eliminate bandwidth cost and to reduce latency substantially, more spare capacity is needed to be placed closer to each client. It will result in lower multiplexing of demand across client locations and suitable to perform latency-sensitive computation [24].
7. Cloud Security Alliance: In federated cloud environment, multiple cloud service providers (CSPs) build trust agreements to share their resources and services to fulfill the demand spikes. With the help of Federated Identity Management (FIM), CSPs can share their subscriber's information with each other on demand basis. However, it also raises several privacy and security issues related to the sharing and management of sensitive information such as maintenance of cloud user information beyond multiple CSPs, access control to user information, standardized privacy and security procedures across multiple CSPs, etc. To handle these issues, several security-based FIM systems have been proposed. However to

provide holistic solution for covering privacy, self-service, interoperability, real-time synchronized user provisioning, and deprovisioning have become critical research challenges in the federated cloud environment [26].

3.3 Types of Flows in Datacenter Networks

Existing works [28, 29] have observed that in the networks of cloud datacenter the majority of network flows tend to be short-term, while the most of packets are some long-lived large network flows. The large (elephant) and small (mice) flows phenomenon are critical problems for network performance. Long-lived flows often named elephant flows which transport massive data volumes in data center networks. These elephant flows represent large transfers and consume a lot of bandwidth. The short-lived flows are termed as mice flows. These flows are often bursty, and delay-sensitive [28]. Different types of applications have different constraints and needs of network resources. Elephant flows are inclined to fill network buffers end-to-end and generate major delays for the latency-sensitive mice network flows, which may lead to significant performance degradation in the network. In literature, hash-based multi-path routing approaches such as ECMP [30] is applied in cloud datacenters, which hashes several elephant flows onto one same link while pushing away other links to be free and creates suboptimal usage for network [31].

3.4 Datacenter Network Topologies

Datacenter network (DCN) topologies can be categorized into two parts: server-centric and switch-centric [32]. In server-centric based DCNs, the routing intelligence is deployed on cloud servers which are connected to the network via multiple ports. In this case, the switches serve merely as cross-bars. On the other hand, in switch-centric-based DCNs, the routing intelligence is deployed on switches, where each server is connected to the network via single port. Some of the well-known switch-centric DCN architectures are three-tier, fat-tree, and VL2 architectures, for the most realistic and practical DCN simulation. The legacy three-tier DCNs are widely deployed and Fat-tree is popular in terms of robustness, scalability, and cost. The VL2 is a practical network architecture that is scalable to support massive cloud data centers with a uniform high capacity between the cloud servers. For server-centric DCNs, the examples are BCube, which is a robust and high-performance and network architecture builds on DCell for a modular data center (MDC) [32].

4 Virtualization Versus Containerization

In a traditional virtualized server, each virtual machine (VM) guest uses a complete OS along with drivers, binaries or libraries, and the required application. But container runs a discrete process, that is, it does not take more memory than any other executable, which makes it lightweight. Figure 8 shows the architectural differences between containerization and virtualization. Docker is an open source platform which couples products of PaaS and Saas. It utilizes OS-level virtualization to deliver software in the form of different packages named containers. To host the heterogeneous containers, Docker engine software is used. It was developed by Docker, Inc. in 2013 [33].

4.1 Traditional Network Versus Software Defined Network

In traditional networks architecture, each device is coupled with three different planes namely control plane, data plane, and management plane. Whenever any new hardware needs to be placed in an existing network, it may notably increase the cost of network management. In conventional network, if the routers are configured for external routing, then it is called BGP routing, which is done in the configuration files as shown in Fig. 9. So, the router-configured files are pushed out to various devices. Once it reaches these devices, the router software and its protocol will run the distributed algorithm. Based on the configuration, it will arrive at a routing solution to produce the ultimate behavior that is installed in the network hardware.

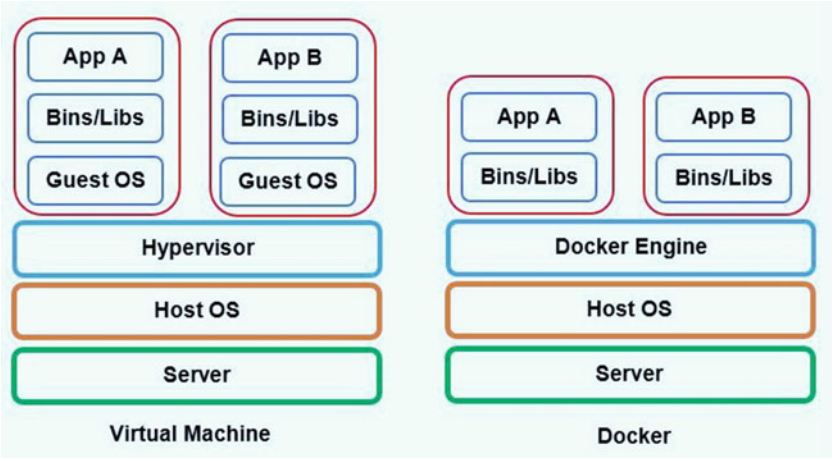


Fig. 8 Virtualization versus containerization

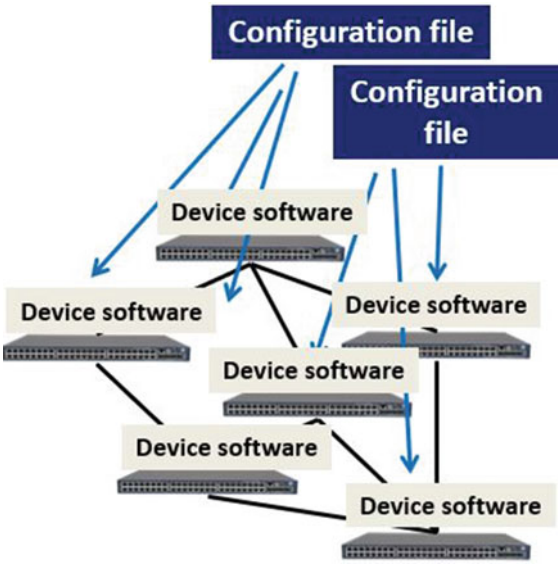


Fig. 9 Traditional network

Therefore, in such traditional networks, the policies and mechanisms are both embedded into the devices, and the distributed algorithm runs to deal with the traffic flow. To accomplish high throughput and low latency and in this scenario, then the controls are not given in the form of programming level, but they are embedded into the distributed protocol. These protocols are standard implementation, and modified based on the demand of applications. SDNs deal with such complexities, so that it can be useful in different cloud environments [34].

The traditional software and OSs are constructed systematically in the form of layers and APIs. The SDN provides the separation of policies and mechanisms. The hardware of network devices is built in the form of low-level interface so that it can be accessed directly. As far as the policies are concerned, they are separated from this particular logic. In SDN, there exists a logically centralized controller as shown in Fig. 10. It communicates with the distributed switches and other devices, and all the smart logic is embedded in the centralized controller. User can express all their policies centrally through the logically centralized controller. All the decisions are policy decisions, and its programming is now taken out from them. Hence, the switching gear is made as simple as possible. Thus, any vendor can provide its hardware, and the logically centralized controller will embed the software within it. These two separations are present in SDN. All intelligence is deployed in a centralized location. Moreover, on top of that we can build the software abstractions that will support for building different applications [34]. In Table 3, we present a comparison between traditional networking and SDN.

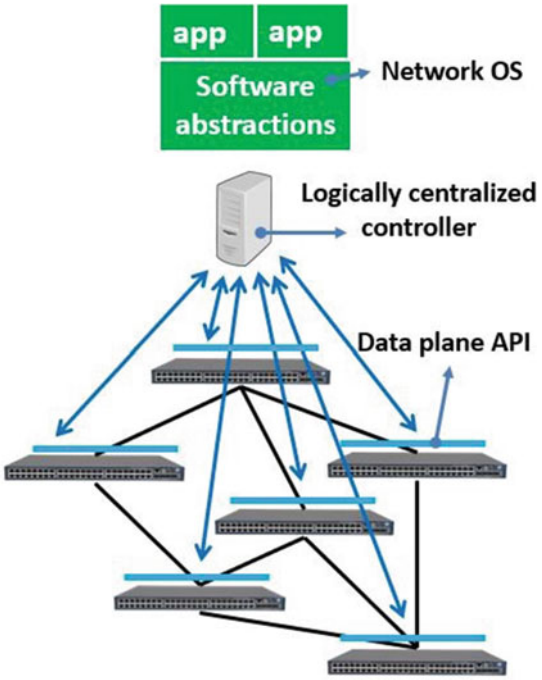


Fig. 10 Software defined network

Table 3 Comparison of traditional networking versus SDN

Characteristics	Traditional networking	Software-defined networking
Type of network	Inflexible and static network	Programmable network
Architecture	Distributed control plane	Programmable centralized control plane
Hardware and software-based support	Comprises hardware network devices	Configured via open software
Scalability	Due to high complexity, network scaling is unsustainable	Highly scalable due to agile centralized control
Dynamics	Complex dynamic and multi-device environment	Adaptable in changing requirements
Cost	High management cost	Less management cost

5 Distributed Databases

Why for Big data traditional databases is not a solution for distributed storing? The answer is traditional databases can't be horizontally scaled the reason being we cannot add resources and computational nodes to enhance data processing power and speed. Since Big data can consist of variety of unstructured data and several other characteristics being seen in Table 1, a traditional database designed for structured data is not a good choice. To counter these disadvantages of traditional databases the Google File System (GFS) was implemented and developed as Google started it all in the first place publishing a paper in 2003 named Google File System later MapReduce as a programming language which was all about scalable distributed file system and processing large distributed datasets [35]. The Google File System was implemented to meet the demands of the ever-growing data and its processing needs inside the organization. It is designed to run over clusters containing thousands of commodity hardwares solving the resulting challenges of not only managing distribution but coping up with the increased danger of hardware faults accompanied by using an existing traditional file system [34].

A GFS cluster consists of several nodes where the primary node called the Master node and other nodes called Chunk servers. The files being appended in the GFS are divided into chunks of fixed size 64 megabytes followed by identification of each chunks using chunk label (64-bit label) and consequently mapping of files to chunks is achieved. Each chunk is replicated multiple times for reliability purpose. Each chunk is replicated multiple times over chunk servers depending upon user need by default the value is 3. Maintenance of file system metadata is done by Master including namespace, access control, and mapping information in addition to location of chunks. To collect the state of chunk servers and give information master sends periodic signals in the form of heartbeat. A client specific to GFS is attached to each application which implements the file system API and interact with Chunk servers and the master node to read and write data of the application. It also interacts with Master node for elaborating all file system metadata for running the applications.

Inspired by GFS an open source framework was invented working across large datasets and across cluster of computers acting as nodes was developed using simple programming model for distributed computing. By using Hadoop even the large enterprise can also efficiently manage their data. Hadoop uses its own file system having distributed characteristic known as HDFS (Hadoop File System). Hadoop is well known for its characteristics mainly like high data availability and accessibility despite hardware failures by keeping the copies of data being stored, providing horizontal scaling, fault tolerant, and economic viable. It can be seen how Hadoop is solving the drawbacks of using traditional databases. Input data in Hadoop are stored as files. The Hadoop architecture primarily comprises 4 components namely HDFS, MapReduce, Yarn, Common Utilities or Hadoop Common [34] as shown in Fig. 11.

HDFS is the primary storage for Hadoop. It's a distributed file system that runs over commodity hardware. The architecture of HDFS consists of two components or

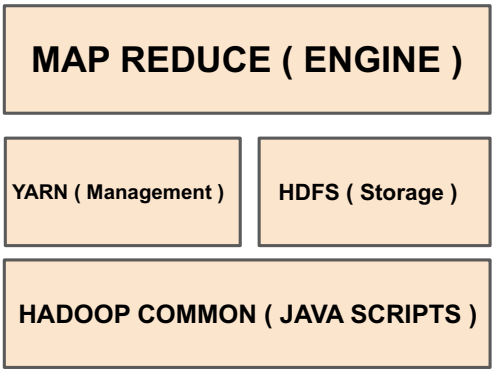


Fig. 11 Hadoop architecture

working architecture based on Master and Slave design where master node and data node are called Namenode and Datanode respectively. The namenode is responsible for containing information or metadata associated with the data being pushed to HDFS, basically a tree structure withholding files in a certain hierarchical manner. In addition to that there are number of datanodes responsible for managing storage attached to the nodes they run over. Datanode also performs block creation after receiving instructions from namenode, datanode also performs block creation, deletion, and replication. Earlier namenode was the single point of failure for HDFS and any shutdown associating namenode subsequently resulted in the fall of whole file system, the next viable option present is secondary namenode that could be hosted as a separate machine acting originally just as a helper for namenode responsible for collecting edit logs from it just as a type of bill and applying it to fsimage. Namenode needs a point from where to restart and so once the new fsimage is being created, it copies back to namenode and it will use this fsimage to restart from same point where failure once had occurred reducing the startup time. The whole purpose of secondary namenode is to collect checkpoints in HDFS and it can't be replaced as namenode or namnode's failure. To solve this very problem and due to high availability Hadoop 2.0 was introduced with two namenodes out of which one would be in standby mode and become active after the failure occurs in the active one [34] as depicted in Fig. 12.

The second component MapReduce acts as an executing engine of Hadoop [36]. It is more like an algorithm based on Yarn framework [37]. The main task associated with MapReduce is to provide parallel processing across the cluster providing Hadoop fast computation as shown in Fig. 13. When Big data is being dealt serial processing is of no use. MapReduce mainly consists of two tasks Mapper and Reducer. The data is moved to Hadoop gets divided into HDFS blocks, these blocks are stored in slave nodes actually residing as HDFS blocks. Can be seen in Figure]. The processing is actually divided into small chunks of data executed parallelly in multiple locations. It saves time as well as network width. Yarn is a module responsible for resource management also called as the operating system of Hadoop helpful in monitoring and managing the tasks scheduled for processing. Whereas Hadoop Common can be referred to as the collection of common utilities which would support other

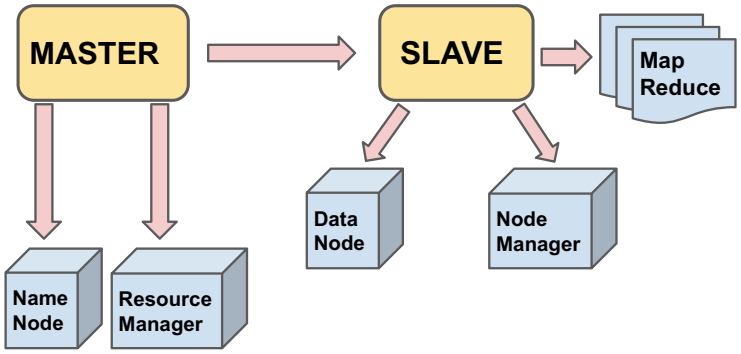


Fig. 12 HDFS architecture

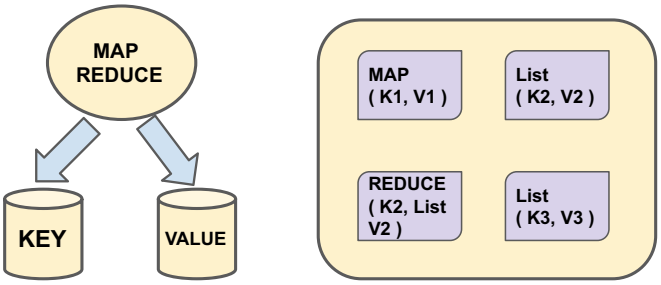


Fig. 13 Map reduce working

Hadoop modules. It is considered as the integral or foundational part of the Hadoop framework consisting of scripts and associated jar files for configuring and starting Hadoop. It also layouts abstraction to underlying operating system and associated file system along with source code, documentation, and contributions being provided by the Hadoop community.

Next comes spark developed under apache which was an open source cluster computing platform that works with fault tolerance and inferred data parallelism providing fault tolerant parallelism [38, 39]. Spark is better than Hadoop in terms of computational speed as it performs in-memory computation reducing the computational time required for processing complex applications. Key features of spark are in terms of performance, rich api's—libraries and scalability, and fault tolerance. In terms of performance as early discussed spark is faster than Hadoop up to 10x (onDisk) and 100x (in Memory). In spark jobs are threads whereas in Hadoop jobs generate a separate JVM. Spark is not limited by any particular programming language as it offers profound level of languages like python, R, Scala, and Java under a single API. It can also be said that spark uses very less lines of code in comparison to Hadoop mapreduce programming as it uses functional programming constructs. In terms of scalability spark is scalable up to 8000 nodes in production and uses RDDs as a fundamental block being an immutable collection of distributed objects

partitioned over nodes present in a system [40]. Adding to its advantages spark could be easily integrated with HDFS or Hadoop ecosystem along with having a separate architecture block support and machine learning libraries [34]. The complete working process of spark is presented in Fig. 14.

In today's world everything focuses around data analytics being the central problem. It's a serious problem but we can't just directly jump to that step because we need a properly managed storage place to contain such amount of data for long term providing a stable foundation. No-SQL emerged to counter the problems of RDBMS in terms of scalability and cost. No SQL stands for not Only SQL. It is simply a supportive addition to Relational databases and SQL. It offers facilities to store and retrieve the data which is stored in tabular format as is being presented in relational databases. These are database management systems being tangible in its nature and working that provides a way to store, manage and process both structured and semi-structured data. Advantages of No SQL over RDBMS are it doesn't require schema, it doesn't require up front design thus providing flexibility to the user, retrieving data doesn't need working conditional bounds. NoSQL follows a Shared Nothing architecture. The key design solution of NoSQL databases is to distribute data across multiple machines for a single database as shown in Fig. 15.

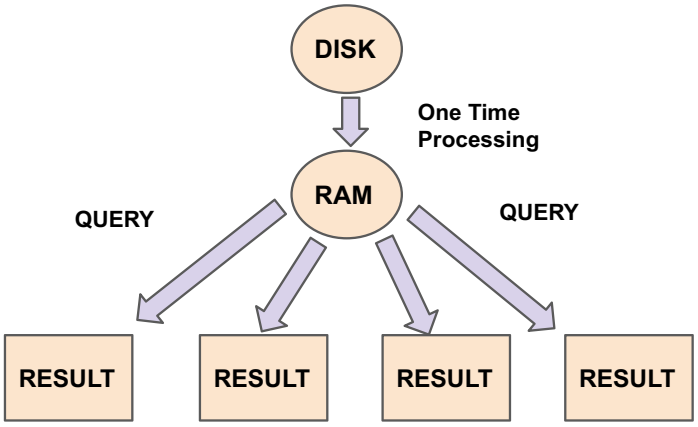


Fig. 14 Spark process working

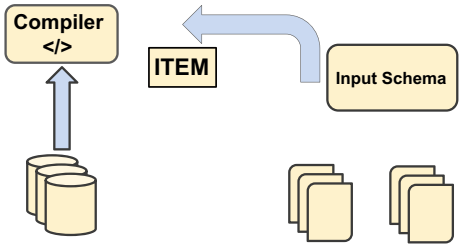


Fig. 15 No SQL process working

There are four types of NoSQL databases present such as wide-column, key-value, graph, and document type. HBase is a kind of open-source implementation of Big Table introduced by Google after analyzing the drawbacks of GFS and MapReduce. Big Table not only uses GFS for data storage, but it is also used for processing a small number of data files in an effective way. Big Table was defined as a distributed storage system to handle data which is present in structured form. It is designed to scale to a very large size, data present across hundreds of commodity hardware in the range of petabyte. HBase is a NoSQL-based database which is working on top of Hadoop. It is inheriting the storage design from column-oriented databases. Apache HBase system consists of set of tables and each table being like any traditional database containing rows and columns. It is one of the part of Hadoop ecosystem which enables real-time read and write access to files in HDFS. HBase is capable of storing semi-structured as well as structured data along with feature of horizontal scalabililty. The three main components which take part in HBase operation are Hmaster, Zookeeper, and RegionServer [34].

ZooKeeper [41] is an open source centralized service for maintaining configuration, naming, information, providing group services, and providing distributed configuration. Services involved in working of ZooKeeper are used in any kind of distributed processing. Every time services are implemented plenty amount of work goes on to race conditions or fixing bugs which are inevitable. Even after doing everything in a correct manner, different implementations of these services lead to complexity management when the applications are deployed.

To make systems more reliable, for propagation of changes and improving management while making all these processes easier to be implemented is our ultimate goal. ZooKeeeper follows a simple model involving client and server where client uses services that are being offered by respective servers [34] as shown in Fig. 16. Figure 17 presents the working of ZooKeeper process.

Cassandra is another NoSQL distributed database, data is being distributed among all the nodes present in associating cluster as well as distributed system across all the nodes present. The role being played every nodes is the same. But why do we need another NoSQL database when other options such as HBase are available. So the answer is quite straightforward both the databases are having their own advantages and disadvantages resulting out of their particular features. Cassandra can be considered a self-sufficient and manageable technology for data storage and management in comparison with HBase because HBase is used as a tool for data input and output purposes from HDFS which is being used as a data storage location and for containing data in a hierarchical manner on particular basis or form along with working as a service status manager where zookeeper service is being used. Other advantages include writing operation is way good compared to HBase relegating the reading operation respectively. One more reason of differentiablity witholding both drawbacks would be the feeble spot of cassandra being data consistency whereas Hbase's suffering data availability [34]. The replication profess of Cassandra across the datacenters is shown in Fig. 18.

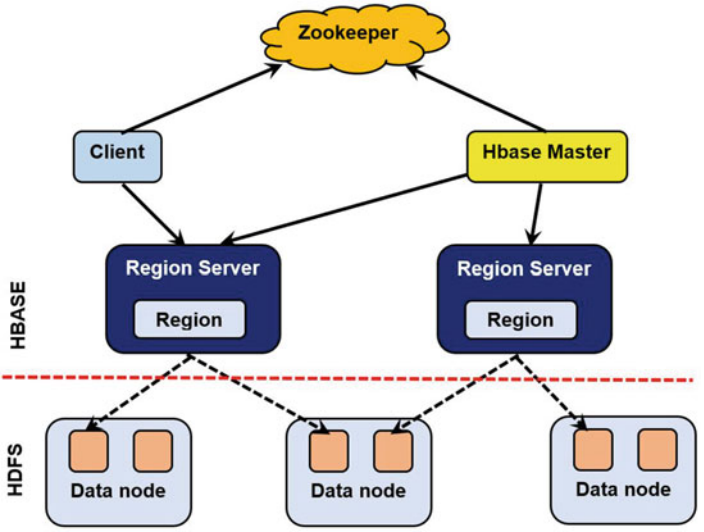


Fig. 16 Architecture of ZooKeeper

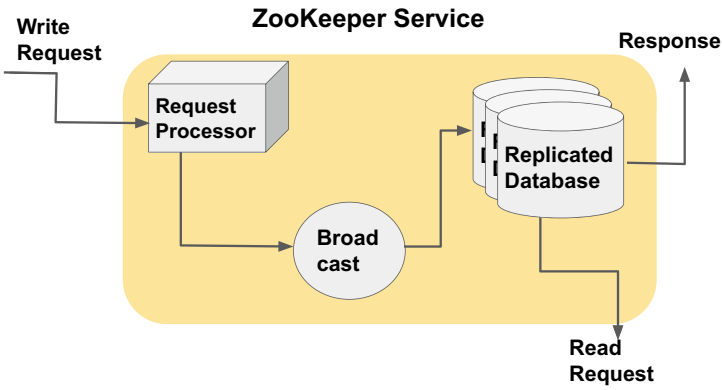


Fig. 17 ZooKeeper process working

Apache Storm had appeared as the platform of preference for industry leading to create real-time and data processing platforms. It gives a set of primitives that can be practiced to produce applications having the capacity to manage huge amount of data in highly scalable manner. It is an open source distributed real-time computation and management system. What Hadoop did with batch processing, storm makes it simpler and easier to process limitless streams of data, able to perform in real time. Storm and Hadoop could be considered similar in some ways. Like in Hadoop we run map-reduce jobs, on storm topologies is rolled but jobs and topologies themselves are not similar in any way. One key differentiation between both the processes is that

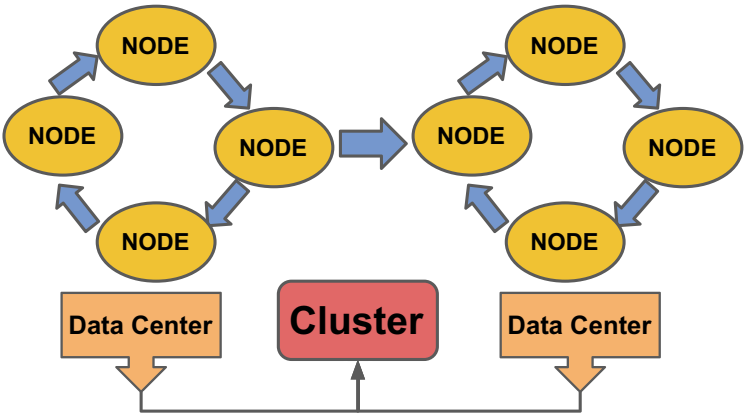


Fig. 18 Cassandra replication process

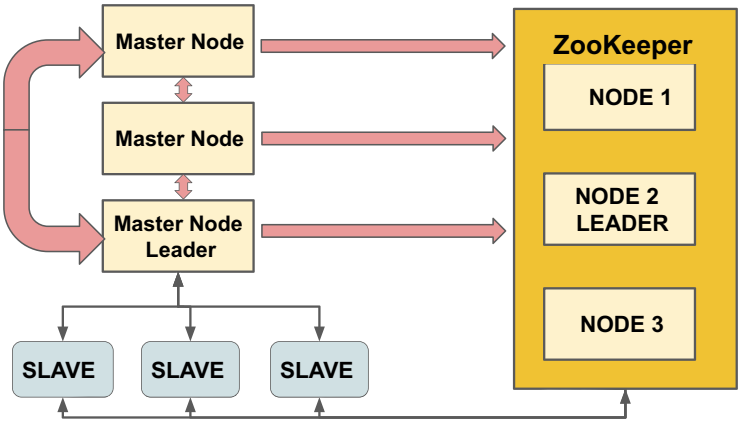


Fig. 19 Apache storm cluster process working

a map-reduce job would surely finish at some point, whereas a topology performs processing of infinite messages until it is killed. In Fig. 19, we have shown the working process of Apache Storm [39].

5.1 Protocols for Faster Data Retrieval, Proxy / Cache Mechanisms

We discussed earlier about data storage and its importance but saying this there are certain problems associated with it. Two of the major problems associated with data storage is making multiple type of data available to multiple users simultaneously and

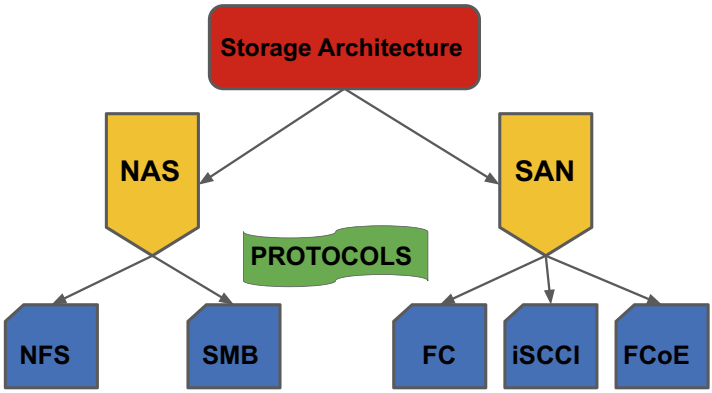


Fig. 20 Network protocols

segregating storage from computation tasks to enhance availability and scalability. Network storage is the best resource to provide solutions to these problems. Today organizations are using mainly two storage architectures mainly Network Access Storage (NAS) and SAN (Storage Area Network) but these storage architectures also must comply with any one of the storage protocols (rules being created for data transmission between applications and devices). NAS and SAN follow common protocols based on their own architectural practice such as NAS uses file level storage for deployment so that data being stored follows a hierarchy. On the other hand, SAN uses block level storage for deployment so that the data is stored in form of block each having their own identifiers associated with itself. Figure 20 represents the classification of network protocols.

Now on the basis of file level storage some common protocols used are Network File System (NFS) and Server Message Block (SMB). NFS is a client-server protocol used to share and access data present on the same network. It uses remote procedure calls (RPCs) to execute request between client and storage servers and is subjected to initiate insecurities, thus it could be vulnerable to internet threats and so should be used on privately secured network. Whereas SMB is a communication protocol used to access network storage and other resources on a remote server. Before the connection is established several messages are exchanged between client and server allowing shared access to multiple resources unlike NFS which allows file sharing only [42].

Then comes the block level storage following few common protocols namely Fiber Internet Small Computer Systems Interface (iSCSI), Fiber channel (FC)—-low latency, Fiber Channel over Internet (FCoE). FCs are most widely used by SAN backed up by low latency, lossless, inorder data transfer bolstered by host bus adapters (HBA), one downside having being expensive compared to other network storage protocols. In comparison iSCSI are a lot easier and less expensive to deploy and has a long range through wide area networks, some latency and extra server load being downside of it. Now FCoE is relatively new and combines both the properties

of FC and iSCSI consequently removing their few drawbacks too like FCoE doesn't require dedicated HBA, but it doesn't offer remote storage access [42].

Cache System also plays an important role in building faster data retrieval and management system. It is used to scale on high demand and being highly available, acts like a short-term memory and is faster than the source of data. Use cases include database caching where cache could be placed between application server and database. Where accessing the data from cache instead of main data store reduce load on main data storage and reduces latency too. The data being written into the cache and the corresponding database at the same time maintains the complete data consistency between the cache and the main storage. Caches also uses different eviction policies too after they are full like LIFO, FIFO or LRU—discards the least recently used items first [43].

6 Conclusion

The integrated model of Cloud and Big data is a powerful combination that can transform the enterprise-level IT operations. To investigate this phenomenon, this study provides a systematic overview of distributed storage infrastructures and discusses the current state-of-the-art solutions for storage technologies using Big data and cloud models. To further explore on distributed storage infrastructure, this chapter firstly provides a brief overview on foundations, analytics, tools, and applications. Moreover, this chapter explores the cutting-edge cloud and Big data technologies and how this integration will produce system-level challenges for different applications. To understand the importance of storage infrastructures, we have also discussed different types of storage, networks, infrastructures, tools, and databases.

6.1 Future Works

The distributed storage technologies and architectures of Big data and cloud support the new generation relying on data-intensive applications. However, there are some of the research directions are further needed to improve the state-of-the-art and current investigations on its landscape and deployment architectures. Some of them are as follows:

- Enabling the dynamic sharing of networks to support heterogeneous architectures and frameworks.
- Investigation of novel data collection methods and new data sources for deep understanding of real-world social activities.
- Study of serverless computing platforms for robust dynamic resource provisioning.
- Additional measurements and enhancements for efficient handling of failures and security in decentralized environments.

– Development of better autoscaling policies for sizing, scaling, and managing the workload during the application's run-time.
– Development of federated identity management systems for building trust agreements and sharing of resources and services during peak demand periods.
– Designing of fair resource management policies to support services and deliver computing resource in federated cloud environments.
– Designing of efficient methods to reduce latency substantially and offering more spare capacity across client locations for latency-sensitive computation.

References

1. Friedman U (2021) Big data: a short history. https://foreignpolicy.com/2012/10/08/big-data-a-shorthistory. Accessed 18 Aug 2021
2. Chang WL, Grady N (2015) NIST big data interoperability framework. big data definitions(1) (No. special publication (NIST SP)-1500-1)
3. Big Data-SAS. https://www.sas.com/en_in/insights/big-data/what-is-big-data. Accessed 02 Aug 2021
4. Hu H et al (2014) Toward scalable systems for Big data analytics: a technology tutorial. IEEE Access 2:652–687
5. Purcell B (2013) The emergence of big data technology and analytics. J Technol Res 1-6
6. Pattnaik K, Prasad Mishra BS (2016) Introduction to big data analysis. In: Mishra B, Dehuri S, Kim E, Wang GN (eds) Techniques and environments for big data analysis. Studies in Big Data, vol 17. Springer, Cham
7. Mishra MK, Patel YS (2016) The role of grid technologies: a next level combat with big data. In: Mishra B, Dehuri S, Kim E, Wang GN (eds) Techniques and environments for big data analysis. Studies in big data 17. Springer, Cham
8. Gantz J, Reinsel D (2011) Extracting value from chaos. IDC review 1142(2011):1–12
9. Zikopoulos B, Barbas H (2012) Pathways for emotions and attention converge on the thalamic reticular nucleus in primates. J Neurosci 32(15):5338–5350
10. Marr B (2015) Big data: using SMART big data, analytics and metrics to make better decisions and improve performance. John Wiley & Sons, Hoboken, NJ
11. Firican G (2017) The 10 versus of big data. Upside where data means business. https://tdwi.org/articles/2017/02/08/10-vs-of-big-data.aspx
12. Mell PM, Grance TS (2011) The nist definition of cloud computing, NIST, Gaithersburg, MD, pp 145–800
13. Li Y, Yu M, Xu M, Yang J, Sha D, Liu Q, Yang C (2020) Big data and cloud computing. Manual of Digital Earth. Springer, Singapore, pp 325–355
14. Awaysheh FM, Alazab M, Garg S, Niyato D, Verikoukis C (2021) Big data resource management and networks: taxonomy, survey, and future directions. In: IEEE communications surveys and tutorials
15. Khan AW et al (2021) Analyzing and evaluating critical challenges and practices for software vendor organizations to secure big data on cloud computing: an ahp-based systematic approach. IEEE Access 9:107309–107332
16. Awaysheh FM, Aladwan MN, Alazab M, Alawadi S, Cabaleiro JC, Pena TF (2021) Security by design for big data frameworks over cloud computing. IEEE Trans Eng Manag
17. Xunyun L, Rajkumar B (2020) Resource management and scheduling in distributed stream processing systems: a taxonomy, review, and future directions. ACM Comput Surv 53(3):1–41

18. Rana AK, Sharma S (2021) Industry 4.0 manufacturing based on IoT, cloud computing, and big data: manufacturing purpose scenario. In: Hura G, Singh A, Siong Hoe L (eds) Advances in communication and computational technology. Lecture notes in electrical engineering, vol 668. Springer, Singapore
19. Mishra S (2014) Survey of big data architecture and framework from the industry. In: NIST big data public working group
20. Michael A, Reynold SX, Cheng L, Yin H, Davies L, Joseph KB, Xiangrui M, Tomer K, Michael JF, Ali G (2015) Spark SQL: relational data processing in spark. In: ACM SIGMOD international conference on management of data. ACM, New York, pp 1383–1394
21. Lakshman A, Malik P (2010) Cassandra: a decentralized structured storage system. ACM SIGOPS Oper Syst Rev 35-40
22. Secci S, Murugesan S (2014) Cloud networks: enhancing performance and resiliency. Computer 47(10):82–85
23. Forestiero A, Mastroianni C, Meo M, Papuzzo G, Sheikhalishahi M (2017) Hierarchical approach for efficient workload management in geo-distributed data centers. IEEE Trans Green Commun Netw 1(1):97–111
24. Iyswarya N, Aman K, Anand S, Bhuvan U, Sriram G (2014) Towards a leaner geo-distributed cloud infrastructure. In: HotCloud'14. USENIX association
25. Fabiana R, Valeria C, Francesco LP (2019) Elastic deployment of software containers in geo-distributed computing environments. In: 2019 IEEE symposium on computers and communications (ISCC), pp 1–7
26. Habiba U, Masood R, Shibli MA (2015) Secure identity management system for federated cloud environment. Software engineering. Networking and parallel/distributed computing. Springer International Publishing, Artificial Intelligence, pp 17–33
27. Asif IM, Benay R, Sarbani R (2019) Auction based resource allocation mechanism in federated cloud environment: Tara. IEEE Trans Serv Comput
28. Afaq M, Rehman S, Song W (2015) A framework for classification and visualization of elephant flows in SDN-based networks. Proc Comput Sci 65:672–681
29. Qian T, Huan Z, Jun D, Lianming Z (2020) Elephant flow detection mechanism in SDN-based data center networks. Scient Progr
30. Hopps CE (2000) Analysis of an equal-cost multi-path algorithm. RFC Editor, USA
31. Fonseca R (2014) Datacenter network large flow detection and scheduling from the Edge Rui (Ray)
32. Zafar S, Bashir A, Chaudhry SA (2016) On implementation of DCTCP on three-tier and fat-tree data center network topologies. Springer Plus 5:766
33. Docker. https://www.docker.com/. Accessed 18 Aug 2021
34. Misra R, Patel YS (2020) Cloud and distributed computing: algorithms and systems. Wiley pp 1-456
35. Jeffrey Dean and Sanjay Ghemawat (2008) MapReduce: simplified data processing on large clusters. Commun ACM 51(1):107–113
36. Konstantin S, Hairong K, Sanjay R, Robert C (2010) The hadoop distributed LE system. In: Proceedings of the 2010 IEEE 26th symposium on mass storage systems and technologies (MSST), pp 1–10
37. Vinod KV, Arun CM, Chris D, Sharad A, Mahadev K, Robert E, Thomas G, Jason L, Hitesh S, Siddharth S (2013) Apache hadoop YARN: yet another resource negotiator. In: Proceedings of the 4th annual symposium on cloud computing. ACM, New York
38. Zaharia M, Chowdhury M, Franklin MJ, Shenker S, Stoica I (2010) Spark: cluster computing with working sets. Hot Cloud 10:1–10
39. https://spark.apache.org
40. Matei Z, Mosharaf C, Tathagata D, Ankur D, Justin M, Murphy M, Michael JF, Scott S, Ion S (2012) Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. In: Proceedings of the 9th USENIX conference on networked systems design and implementation, pp 2–12
41. https://cwiki.apache.org/confluence/display/ZOOKEEPER

42. https://searchstorage.techtarget.com/tip/Choosing-your-storage-networking-protocol . Accessed 15 Aug 2021
43. https://www.sciencedirect.com/topics/computer-science/cache-storage

Chapter 7
Stream Data Processing Systems with Progressive Quality Improvement

Chaxiong Yukonhiatou, Tomoki Yoshihisa, Tomoya Kawakami, Yuuichi Teranishi, and Shinji Shimojo

Abstract One of the main applications of stream data processing systems is detecting objects in video data streams. In such stream data processing systems, each camera device sends its recorded video data to the remote processing computer for the detections in cases that the computational resources of the camera devices are insufficient for the detections. To improve the performances of the stream processing systems for object detections, most methods reduce the communication traffic between the cameras and the processing computers. Although object detection processes do not always require the predetermined original data quality, the processing computers of conventional systems always receive original quality data. By considering the necessity of original quality data, we can reduce redundant communication traffic in some situations and can improve the performance indexes of the stream data processing systems. Hence, in this book chapter, we explain a Progressive Quality Improvement (PQI) approach to further improve the performances. In the PQI approach, for only the case that the data for higher qualities are needed, the processing computer progressively collects them from the data sources. Moreover, we implement a video surveillance system with the approach.

1 Introduction

Recently, stream data processing systems such as Apache Flink, Spark Streaming, etc. have attracted great attention. One of their applications is detecting objects in video

C. Yukonhiatou
National University of Laos, Vientiane, Laos
e-mail: chaxiong@fe-nuol.edu.la

T. Yoshihisa (✉) · S. Shimojo
Osaka University, Osaka, Japan
e-mail: yoshihisa@cmc.osaka-u.ac.jp

T. Kawakami
University of Fukui, Fukui, Japan

Y. Teranishi
National Institute of Information and Communications Technology, Tokyo, Japan

M. Chaturvedi et al. (eds.), *Recent Advancements in ICT Infrastructure and Applications*, Studies in Infrastructure and Control, https://doi.org/10.1007/978-981-19-2374-6_7

data streams. For instance, a surveillance system in an airport detects human faces in videos recorded by the surveillance cameras to find thieves. In these stream data processing systems, each camera device sends its recorded video data to the remote processing computer for the detections in cases that the computational resources of the camera devices, such as computational powers or memory capacities, are insufficient for the detections. The processing computer processes each frame included in the received video data continuously for finding objects.

There are two main performance indexes for the stream data processing systems for object detections. One is the transaction time (transaction time is from the time to get a video frame on cameras to the time to finish detecting objects in the processing computer) and the other one is the transaction rate (the number of finished transactions per second). A shorter transaction time and a faster transaction rate can improve application performances for real-time object detections. For instance, the probability to catch a thief increases as the processing computers analyze the videos with a shorter transaction time and a faster transaction rate. To improve the performances, most methods reduce the communication traffic between the cameras and the processing computers.

Although object detection processes do not always require the predetermined original data quality, the processing computers of conventional systems always receive original quality data. For example, in face detection systems, the processing computer can sometimes recognize faces even when the image size is smaller than the original size (The quality of data in this case is the resolution). In human detections, the processing computer can sometimes detect target humans in a part of image areas (The quality in this case is the image region). By considering the necessity of original quality data, we can reduce redundant communication traffic in some situations and can improve the performance indexes of the stream data processing systems.

Hence, in this book chapter, we explain a Progressive Quality Improvement (PQI) approach to further improve these two performance indexes. In the original PQI approach, for only the case that the data for higher qualities are needed, the processing computer progressively collects them from the data sources. Moreover, we explain a method (PQI-CDI, PQI with Cycle-based Dynamic Interval) to improve the transaction rate by changing the transaction interval dynamically under the PQI approach. To investigate whether the PQI-CDI method can improve the performances in real situations, we implement a video surveillance system with the PQI-CDI method. In the system, three camera devices are used to serve as cameras, and a laptop computer functions as the processing computer. We measure the transaction time and transaction rate of real situations using our implemented system.

The remaining chapter is organized as follows. In Sect. 2, we introduce the related work. In Sect. 3, we explain our proposed PQI approach and PQI-CDI method. In Sect. 4, we explain the system design for the implementation of the video surveillance system with the PQI-CDI method. The implementation details are presented in Sect. 5, and experimental results are presented in Sect. 6. Finally, the chapter concludes in Sect. 7.

2 Related Work

In this section, we introduce several researches for improving the transaction time and the transaction rate for the stream data processing systems for object detections. After that, we introduce some surveillance systems.

2.1 Transaction Time Reduction

Reducing the time required for processing data leads the transaction time reduction. An efficient computational resources allocation scheme for stream data processing was developed in [1]. In the scheme, the processing computer allocates computational resources to process each stream data. Thus, the processing times for each stream data are reduced because the occurrence probability of the computational overheads such as swapping, page faults, and thrashing is reduced. A method to reduce the delay for starting data processing on the processing computer was proposed in [2]. The processing computer prepares separate data queues for each process and selects the data to be processed to reduce the processing delay. For object detection systems, which is a typical application in this study, [3] developed fast object detection frameworks. The programs and the communication parameters of these frameworks were optimized for the target systems and thus established faster object detection. However, these methods did not consider the necessity of the original data quality because they did not focus on data quality.

Some video codecs adopt multi-quality video, which include several video quality data in a video data stream. Therefore, methods to improve video data quality progressively were proposed. A quality selection considering the players' buffers to reduce the number of video pauses was proposed in [4]. A faster motion vector calculation method was proposed in [5]. Motion vectors indicate the direction of moving objects and are used for video data compression. The method calculates the motion vector in a short time by using a low quality (size) video data. Several video data compression techniques such as Wyner-Ziv coding use some video quality data as side information. Methods to get efficient side information to establish fine and effective video data compression were proposed in [6–8]. However, these methods did not consider the necessity of the original quality data. Stream data processing systems sometimes do not need the original quality data for the processes.

2.2 Transaction Rate Improvement

To improve transaction rates, several methods have been proposed in the literature. A method to improve the transaction rate for multidimensional stream data was proposed in [9]. This method aimed to increase the transaction rate by reducing the

processing loads per transaction. In their method, the streaming data sources sent only the necessary dimensional axes of data for processing. As the processing computer did not receive unnecessary data, there was no need for data refinement activities to reduce processing loads. However, this method assumed a static transaction interval, thus the applications need to set an appropriate transaction interval for the multidimensional stream data.

Dias et al. [10] and Agrawal and Jani [11] used object detection frameworks to detect objects as fast as possible. They proposed fast object detection algorithms to improve the transaction rates. However, their frameworks did not care the necessary quality of image data which our PQI approach can adjust.

To improve transaction rates for object detection, various approaches have been proposed, such as searching-area reduction [12], detection accuracy improvement [13, 14], and dynamic background updating [15–18] focused on hardware (e.g., field-programmable gate arrays) for improvements. Their methods improved transaction rates by reducing the processing time. However, they also assumed a static transaction interval. The performance and quality of object detection applications were not improved even when there is a room in the processing computer to process to achieve higher transaction rates.

GPGPUs, digital signal processors, and field-programmable gate arrays designated for stream data processing are recently developed to improve the transaction rate. In [19], the authors proposed a simple data processing model for these processing units to enable low complexity performance prediction for stream data processing. Their model was expressed by straight directed graphs with queues at each vertex. Vertices represent processing kernels. They compared the amount of the stream data (called flow) that the kernels receive per second in their proposed model with that in real situations. Their results showed that the modeled flow matched with the actual flow when the queuing scheme is simple such as batching, i.e., picking up data from the queue when the data amount stored in the queue exceeds the threshold. However, the proposed model did not consider the data quality.

2.3 Surveillance Systems

A well-known video encoding scheme that could ensure a high video quality with a limited communication bandwidth is H.264. A surveillance system using H.264 was implemented in [20]. The authors evaluated the video quality and the bandwidth consumption. Although the authors confirmed the reduction of the data amount transmitted from the camera devices to the viewer machines, the time required to detect objects in the video surveillance systems was not reduced.

A scheme was established to reduce the bandwidth consumption of video surveillance systems [21]. In [21], the camera devices did not transmit data to the processing computer unless intrusions were detected. The processing computer verified whether

the intrusion actually occurred and sent a notification to the system manager only when the intrusion was occurred. This scheme could reduce the bandwidth consumption between the cameras and processing computer.

A simple approach to reduce the time required to detect objects was to limit the target area in the image to detect objects. In [22], the system identified the background by using a background subtraction technique and realized the reduction by omitting the background area from the target area. Another approach in [23] sets a threshold to clarify the background area. The method regarded the pixels for which the difference from the previous frame image is less than the threshold as the background area. A larger threshold yielded a larger background area because the pixels that are even largely different from that of the previous frame image are regarded as the background area. This method was similar to the proposed system in that the background was updated (progressive collection of the remaining data) when an object was detected. However, in the PQI-CDI method, the transaction interval was dynamically changed to enhance the transaction rate.

3 Progressive Quality Improvement Approach

In this section, we explain our proposed approach and method. First, we explain the progressive quality improvement approach [24].

3.1 Basic Idea

Generally, data have certain qualities, and the original data gives the highest quality. Processes can be executed even if the data quality is lower than the original quality, and data with the highest original quality often give the best performance for applications. For example, one of the quality indexes of image data is resolution. Image data with 640 x 480 pixels have a higher quality than image data with 320 x 240 pixels. Image processing to detect faces can be executed for various pixel sizes; whereas higher resolution image data generally gives higher accuracy because they have more information. Therefore, by obtaining data with the highest original quality, applications can achieve the same performance.

When the processing computer processes data sequentially in the ascending order of quality in a transaction, they can skip to the next transaction when the subsequent processes for higher quality data are meaningless [25]. Assuming, similar to the example in the introduction section, that a processing computer executes the processes for detecting perpetrators in video data streams. The processing computer first receives the lowest quality image data of a frame and executes the processes to detect faces in the image. In cases where the faces are not detected, the processing

computer skips the processing of higher quality image data because perpetrators will not appear in the frame. The data amount of lower quality data is smaller than that of the original quality data. Therefore, if the probability to proceed to the processes of higher quality data is small, the total data amount required for each transaction can be reduced compared with the cases in which the stream data sources always transmit the original quality data. Thus, the transaction times are reduced by maintaining the application performance. This approach is called as *progressive quality improvement* approach in this study.

We will explain the detail of the process flow in Sect. 4.

3.2 Example Flow

This section demonstrates example transaction flows under the conventional approach and the PQI approach.

Figure 1 shows the situation. In this example, the number of cameras is two.

First, an example transaction flow is explained under the conventional approach. Camera 1 sends its recorded image data frame-by-frame to the processing computer. In Fig. 1, the tth frame is shown as $D_{1,a}(t)$ $(t = 1, \cdots, T)$. $D_{n,a}(t)$ denote the original data of the stream data source n at the tth transaction. a represents 'all' and is not a variable. For example, when the frame rate is 10 [Hz], Camera 1 sends an image data every 0.1 [s]. Camera 2 sends its recorded image data to the processing

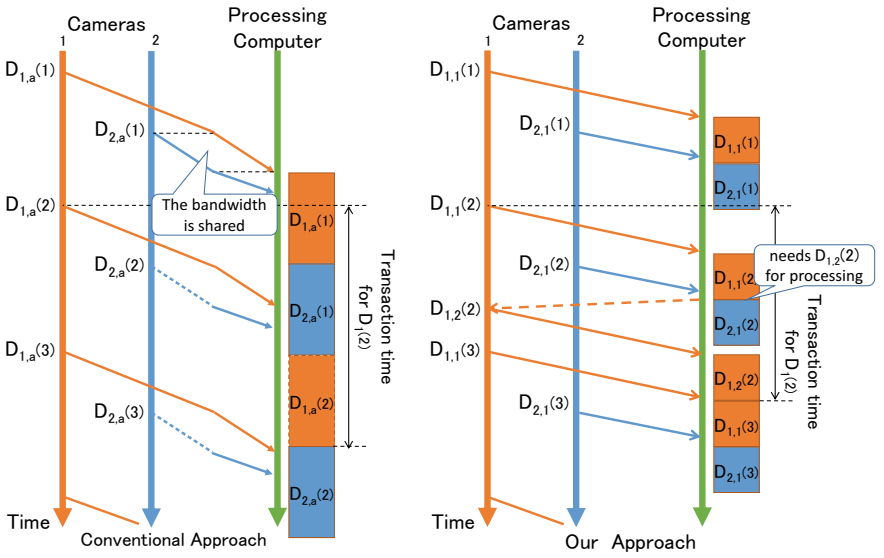


Fig. 1 Example flow of a conventional approach versus ours

computer. In this example, the frame rate for Camera 2 is the same as that of Camera 1, but the time to begin sending the data differs. After Camera 1 sends $D_{1,a}(t)$, Camera 2 sends $D_{2,a}(t)$. When the data transmissions from Camera 1 and Camera 2 overlap, the input communication bandwidth for the processing computer is equally divided. This is the reason why the communication speed of Camera 1 decreases while communicating with the processing computer , as shown in the figure. After Camera 1 finishes sending $D_{1,a}(t)$, the input communication bandwidth is dedicated to Camera 2 whose communication speed increases as shown in the figure. When the processing computer finishes receiving $D_{1,a}(t)$, it begins processing $D_{1,a}(t)$. While processing $D_{1,a}(t)$, the processing computer finishes receiving $D_{2,a}(t)$. As it processes $D_{1,a}(t)$ at this time, it stores the received $D_{2,a}(t)$ in its buffer and begins processing it after finishing with $D_{1,a}(t)$. Similarly, while processing $D_{2,a}(t)$, the processing computer finishes receiving $D_{1,a}(t+1)$. It then begins processing after finishing with $D_{2,a}(t)$. The transaction time for $D_{1,a}(t+1)$ in this case is shown in the figure. This is the time consumed from the start of sending $D_{1,a}(t+1)$ to the end of $D_{1,a}(t+1)$ process.

Next, an example transaction flow is explained under our proposed PQI approach. The stream data sources can construct Q data items, $D_{n,q}(t)$ ($q = 1, \cdots, Q$), which provide the data needed to get the qth quality data of $D_{n,a}(t)$. Similar to the example for the conventional method, Cameras 1 and 2 send their recorded image data to the processing computer periodically. Unlike that in the conventional method, the image data are divided into two quality levels ($Q = 2$). The first is the lowest, and the second is the original quality. We assume that the data for the second quality only includes the difference in data from the first and the amount of the data for each quality is the same. For simplicity, we assume that the data amount of each quality is half of that of $D_{1,a}(t)$ ($t = 1, \cdots, T$). Therefore, the time required to send $D_{n,q}(t)$ ($n = 1, 2, q = 1, 2$) is half of that needed to send $D_{n,a}(t)$ over a fixed communication bandwidth. Therefore, the communication of $D_{1,1}(t)$ does not overlap $D_{2,1}(t)$, although the communication of $D_{1,a}(t)$ overlaps that of $D_{2,a}(t)$ under the conventional approach. The processing computer does not request the second quality data item in the first cycle because it cannot detect human faces in the image data. In the $t+1$th transaction, the processing computer begins processing $D_{1,1}(t+1)$ after receiving it. Then, the processing computer requests the second quality data item in the transaction because it detects human faces in the image data. When Camera 1 receives the request for $D_{1,2}(t+1)$, it begins transmission for the difference data between $D_{1,2}(t+1)$ and $D_{1,1}(t+1)$. In this example, the processing computer does not request the second quality data item of $D_{2,1}(t+1)$. After receiving $D_{1,2}(t+1)$, the processing computer executes the necessary processes and completes the transaction. The transaction time needed for the tth transaction and for Camera 1 is visualized in the figure as the time from the start of sending $D_{1,1}(t+1)$ to the completed processing of $D_{1,2}(t+1)$.

In summary, the transaction time of our approach is shorter than that of the conventional approach because several transmissions of higher quality data are skipped.

3.3 Proposed Method

We propose a method called the PQI-CDI (Progressive Quality Improvement approach with Cycle-based Dynamic Interval) method ([26–28]). In this section, we first discuss the adjustment of the transaction interval to improve the transaction rate. After that, we explain the design and the algorithms of the PQI-CDI method. Further, we show an example on how to improve the transaction rate under the PQI-CDI method. In the PQI-CDI method, the system changes transaction intervals dynamically and adopts the PQI approach to reduce transaction time. The details of the PQI approach were described in Sect. 3.2, where we demonstrated that the PQI approach reduces transaction time. However, the transaction interval under the original PQI approach is static and cannot improve the transaction rate. Therefore, the PQI-CDI method dynamically changes the transaction intervals. For this, the PQI-CDI method determines the timings to change the intervals and new transaction intervals. These are explained below.

3.3.1 Timings to Change Intervals

A frequent adjustment of intervals increases the transaction rate. Less-frequent changes decrease the transaction rate because it also takes time to adjust the transaction intervals. The transaction interval is not changed until the next adjustment time even when the transactions are overlapped. The appropriate timing for interval changes depends on the communication time and the processing time as these times change dynamically. Therefore, we determine the period for changing transaction intervals by introducing a notion which we call *cycles*.

3.3.2 Determining New Intervals

PQI-CDI method defines a fixed cycle length to change intervals, C_n. When the number of the complete transactions reaches C_n, the processing computer changes the transaction intervals of the data source n. Once the transaction interval changes, the processing computer starts counting the number of transactions. We adopt the average transaction time from the previous cycle as the new interval. The average transaction time $AveTT_n(t)$ is given by the following equation:

$$AveTT_n(t) = \frac{\sum_{\tau=t-C_n+1}^{C_n} TT_n(\tau)}{C_n}. \quad (1)$$

$TT_n(\tau)$ is the transaction time of the data source n at the τth transaction.

3.4 Algorithms

In this section, we explain the algorithms for the data sources and the processing computer, respectively.

Figure 2 shows the flowchart of data sources in the case where the processing computer requests the change of the transaction intervals. When the data source n receives the request to change the transaction interval to i, the data source changes the transaction interval and resets the counting value of the number of transactions for the next cycle.

Figure 3 shows the flowchart of the processing computer. When it receives $D_{n,q}(t)$, it processes the data. When $q = Q$, the tth transaction finishes. Otherwise, the processing computer judges the necessity of $D_{n,q+1}(t)$. In case that $D_{n,q+1}(t)$ is needed for process execution, the processing computer requests $D_{n,q}(t)$ to the stream data source n. Otherwise, the transaction finishes. In the PQI-CDI method, the processing computer then checks whether c_n reaches C_n when a transaction finishes. Here, C_n is the interval needed to change the transaction interval of the data source n. If c_n reaches C_n, the processing computer calculates $i = AveTT_n(t)$ and sends a request to n in order to change the transaction interval to i. Then, c_n is initialized as 0 and the new cycle starts.

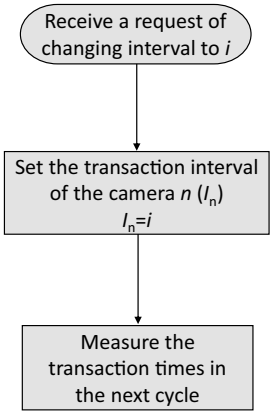


Fig. 2 The flowchart for the data sources under the PQI-CDI method

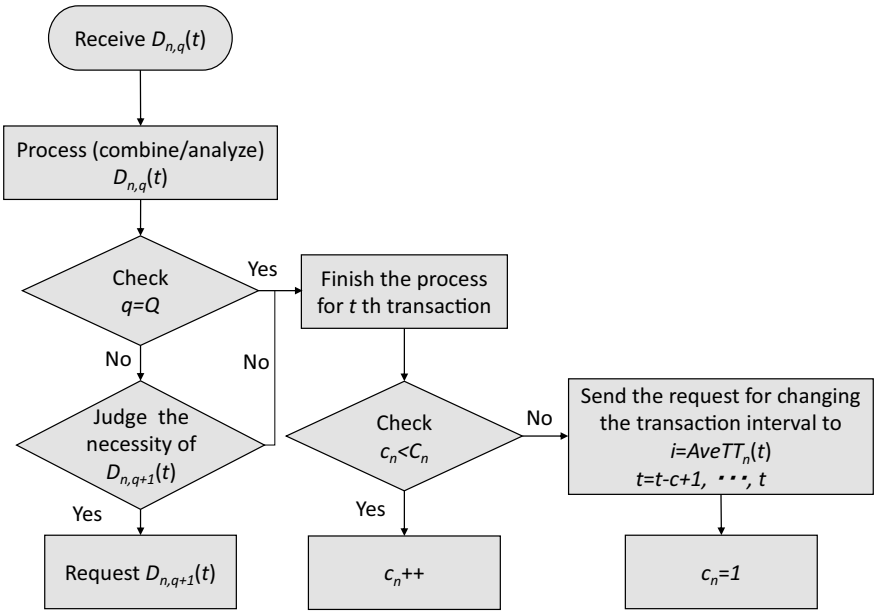


Fig. 3 The flowchart for the processing computer under the PQI-CDI method

4 Design of a Video Surveillance System

In this section, we explain the design for the implemented video surveillance system.

4.1 System Architecture

Figure 4 shows the system architecture. Some camera devices and a processing computer are connected to a computer network. The camera devices and processing computer can communicate with one another. The computational resources of the camera devices are lower than that of the processing computer because we assume that compact devices perform as the camera devices such as Raspberry Pi devices equipped with camera modules. The processing computer can be a laptop or a desktop computer, among other alternatives.

The camera devices obtain images and produce image data that have different qualities. For example, the Raspberry Pi devices obtain the JPEG image data from the equipped cameras. Notably, in the progressive JPEG format, the image data has several qualities (known as scans). Figure 5 shows an example of the JPEG image data having different qualities. $q_1, \cdots, q_{10}$ represents the quality numbers. The lowest quality data corresponds to the smallest amount of data. These produced data are temporarily stored in the memory of the camera devices. The processing

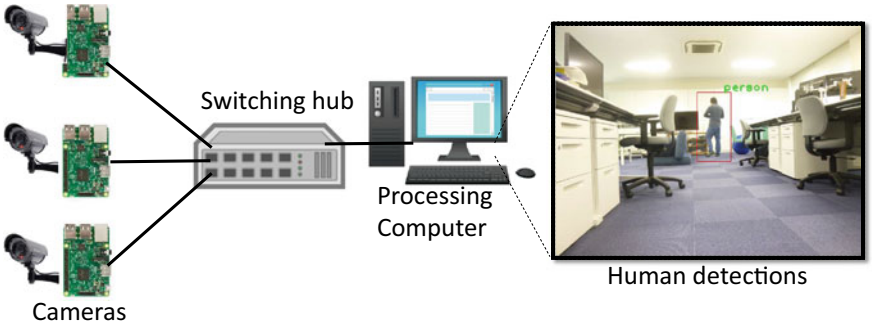


Fig. 4 System architecture design

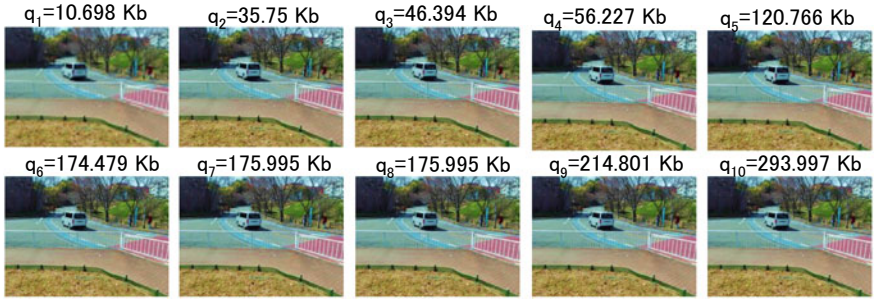


Fig. 5 Example images with different qualities

computer collects the stored image data from camera devices and executes image processing tasks such as object detection.

The camera devices and the processing computer adopt the PQI-CDI method to enhance the transaction time and rate. The details of the PQI-CDI method was explained in Chap. 3. The camera devices cyclically transmit the lowest quality data and transmit high quality data only if required by the processing computer. The processing computer changes the transaction intervals of each stream data when a predetermined number of transactions is finished.

4.2 Process Flows for Camera Devices

In this chapter, we briefly explain the process flow for camera devices designed for implementation. Figure 6 shows an image of the generation of each quality data in the cameras. The transactions at the camera device n ($n = 1, \cdots N$) arise a predetermined transaction interval elapses. Each camera device obtains $D_{n,a}(t)$ which is the original video frame image data of camera device n at the tth transaction. The subscript a refers to the original data. After obtaining $D_{n,a}(t)$, each camera generates

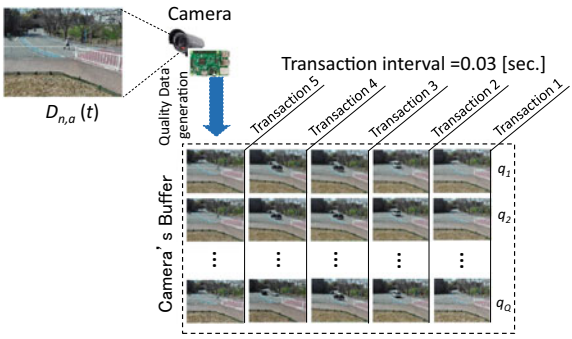


Fig. 6 Generation image of each quality data in the cameras

$D_{n,q}(t)$ $(q = 1, \cdots, Q)$ from $D_{n,a}(t)$ and temporarily stores it to its storage. $D_{n,q}(t)$ represents the data needed to generate the qth quality image data in combination with $D_{n,p}(t)$ $(p = 1, \cdots, q-1)$.

After generating $D_{n,q}(t)$, each camera transmits $D_{n,1}(t)$ to the processing computer and deletes $D_{n,1}(t)$ from its storage. After the transmission, the camera device waits for the request for higher quality data until the next transaction arises. If the camera device receives the request for the qth quality data, it transmits $D_{n,q}(t)$ to the processing computer and deletes it.

4.3 Process Flows for Processing Computers

Details of the process flow for the processing computers were presented in Sect. 3 (Fig. 3). In this chapter, we briefly explain the process flow for the processing computer designed for the implementation. Figure 7 illustrates the processing image of the processing computer. When the processing computer receives $D_{n,r}(t)$ $(r = 2, \cdots, Q)$, it generates the image data having the rth quality by combining it with

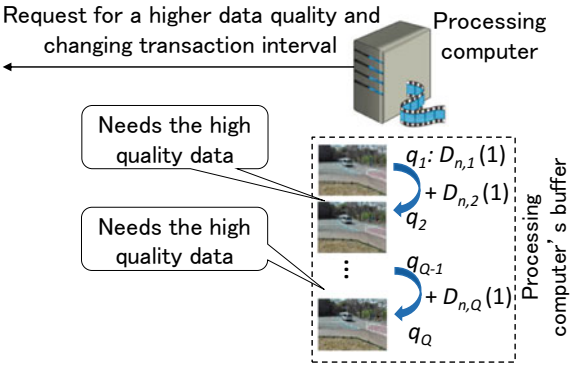


Fig. 7 Processing image of the processing computer

$D_{n,s}(t)$ $(s = 1, \cdots, r-1)$. The first quality data is $D_{n,1}(t)$ itself and the processing computer does not need to combine it with other data. After the generation, the processing computer attempts to detect human bodies in the image data by executing the object detection processes. If objects are detected, the computer requests the camera device n for $D_{n,r+1}(t)$. Otherwise, the tth transaction completes and the processing computer waits for the next data to be transmitted from the camera devices.

In the PQI-CDI method, the processing computer updates the transaction interval after a cycle finishes. The cycle for the camera device n includes C_n transactions. If C_n transactions are completed for the camera device n, the processing computer calculates the new transaction interval and sends the message to change the transaction interval with the new value to the camera device n.

4.4 Software Modules

Figure 8 shows the software modules. The system incorporates two types of installed software: one on the processing computer, and the other on the camera devices. We explain each module as follows.

4.4.1 Camera Devices

The software module for the camera devices consists of three parts.

- **Communication Module**: The communication module of the camera devices establishes the communication session with the processing computer. Once the communication session is established, stream data are transmitted to the processing computer using the communication session. Also, the processing computer sends the request for the data using the same session. In cases where the communication session breaks down, the communication module starts to establish the communication session again.
  The address for the processing computer is designated to the camera devices by the user or the stream processing system manager. Not to concentrate the loads that are caused by establishing the communication session on the processing computer , the camera devices start the communication session establishment process (client-side communication session establishment).
  When the communication module receives data from the first data quality transmission module or the higher data quality transmission module, it transmits the received data to the processing computer. When the communication module receives the request for a changing transaction interval, it transmits the request to the first data quality transaction module, and when the request for higher quality data arrives, to the higher data quality module.
- **First Quality Data Transmission Module:** The first quality data transmission module obtains the original image data from the camera and temporarily stores

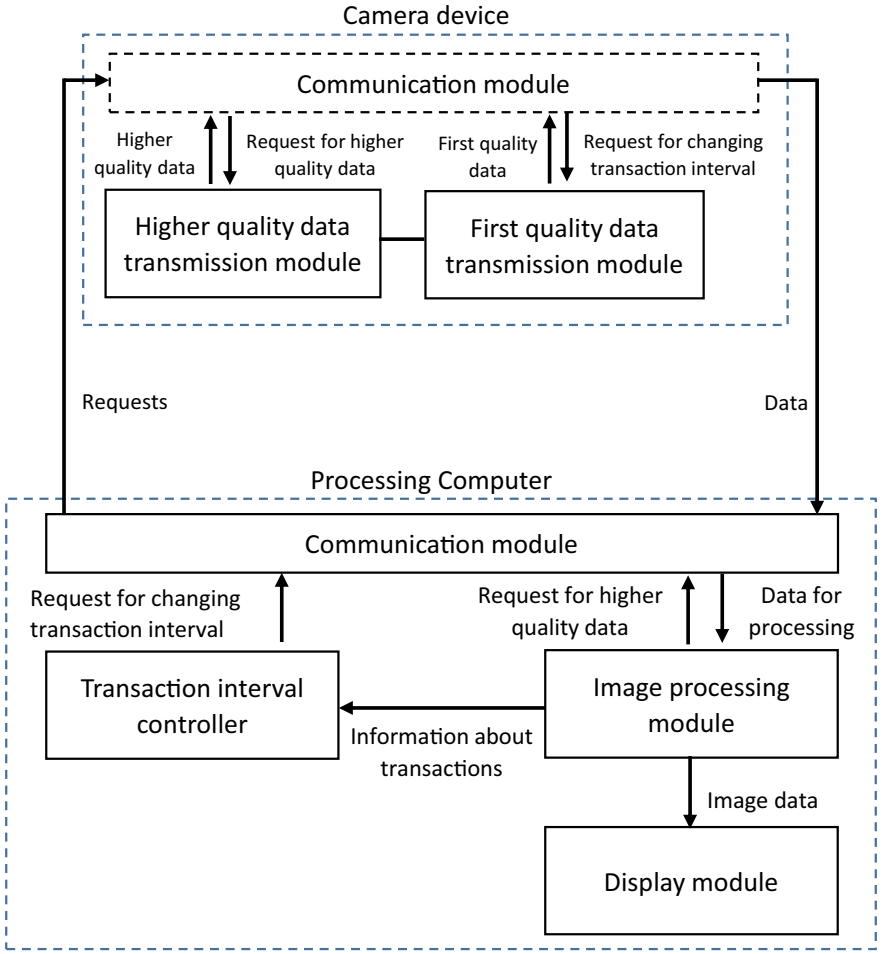


Fig. 8 Software architecture of the implemented system

it in the buffer when each transaction starts. The transactions start every time the instance at which the transaction interval elapses. Moreover, the module produces the data needed for qth quality data, $D_{n,q}(t)$ $(q = 1, \cdots Q)$, and stores it in the buffer.

When the first quality data transmission module has produced the first quality data, $D_{n,1}(t)$, the module transmits the data to the communication module and then the communication module transmits it to the processing computer.

When the first quality data transmission module receives the request for changing the transaction interval, the module changes the transaction interval to the designated value.

- **Higher Quality Data Transmission Module:** When the higher quality data transmission module receives the request for higher data from the communication modules, it obtains the requested data from the buffer and transmits it to the communication module.

4.4.2 Processing Computer

The software module for the processing computer consists of four parts.

- **Communication Module**: The flow of the communication module of the processing computer is like that of the camera devices.
  The communication module of the processing computer establishes the communication session with the camera devices. When the module receives the request for establishing the communication session with the communication module of the camera devices, the module responds to the request and the communication session is established.
  When the communication module receives data from the camera devices, the module transmits the data to the image processing module. When the communication module receives the requests for changing transaction interval or higher quality data, the module transmits the requests to the camera device.
- **Transaction Interval Controller**: The transaction interval controller manages the transaction intervals of each camera device based on the PQI-CDI method.
  When the module receives the information about transactions from the image processing module, the module counts up the number of the transactions from the start of the cycle (c_n in Sect. 3.3.2) to check whether the number reaches the cycle length, C_n or not. If the number reaches C_n, the module calculates the new transaction interval and sends the request for changing the transaction interval to the communication module. To calculate the new transaction interval, information about the transaction times is required. The information is sent from the image processing module.
- **Image Processing Module:** The image processing module processes the received image data according to the user designated processes.
  When the module receives the data for processing, the module executes the processes to the data and judges the necessity of higher quality data. If the higher quality data is needed, the image processing module sends the request for the higher quality data to the communication module. If the higher quality data is not needed or the received data is the highest quality data, the transaction finishes. Therefore, it sends the information about transactions to the transaction interval controller.
  Moreover, to check the behavior of the system, the image processing module transmits the image data to the display module.
- **Display Module**: The display module displays the image data received from the image processing module to the user's display.

5 Implementation

In this section, we explain the implementation of the video surveillance system.

5.1 Equipment

This section describes the equipment used for the implementation. Figure 9 shows the devices used in the implementation. Three Raspberry Pi devices with camera modules and a laptop computer are connected to a network switch. An NTP (network time protocol) server is connected to the network for the synchronization of the times among the processing computers and the stream data sources. Time synchronization is required to measure the transaction time. The devices can communicate with one another via the computer network. The laptop computer functions as a processing computer. Table 1 lists the specifications of the implemented system.

The camera devices convert the stored image data into the progressive JPEG format containing two scans. That is, the number of the qualities is two.

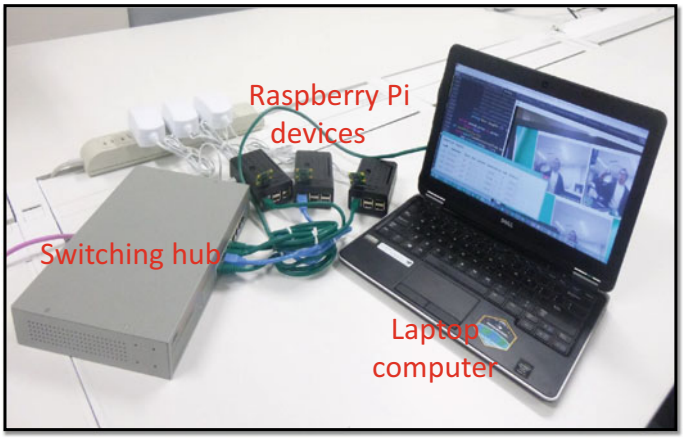


Fig. 9 Devices used in the implementation

Table 1 Specifications of the implemented system

Items	Details
Recording computer	Intel Core 2 Duo T6570 (2.1 [GHz] dual-core), 4 [GB] Memory
Camera device	Stored images as camera
Processing computer	AMD Ryzen 5 2500U (2 [GHz] quad-core), 8 [GB] Memory, AMD Radeon Vega 8 Graphics
Network	100BASE-TX/1000BASE-T

5.2 Programs

We created two Python scripts. One is the script for the camera devices, which runs on each Raspberry Pi device in the system. The script consists of three parts as described in Sect. 4.4.1. The Raspberry Pi devices does not originally contain a function to record different quality data. To produce image data that have different qualities, a function embedded in OpenCV (version 3.2) is adopted to convert the original image data to the progressive JPEG format. The converted progressive JPEG images have two scans as different qualities.

Another script is for the processing computers, which runs on the laptop computer in the system. The script consists of four parts as described in Sect. 4.4.2. The image processing is performed by OpenCV. For the detection of human bodies, we used the function 'Haar Feature-based Cascade Classifier', which uses Haar-like features for human body detection. Human bodies can be detected also in the lowest quality image data.

5.3 Sample Images

This section describes the procedures to generate the data that have different qualities and request higher quality data for a single image frame.

Figure 10 shows an example of images displayed under the PQI-CDI method. The left-sided image shows the case where the processing computer does not detect humans. In this case, the processing computer does not request higher quality data. On the other hand, the right-sided image shows the case where the processing computer detects humans. In this case, the processing computer requests the data with the highest quality to obtain high-resolution image data. Therefore, the resolution of the left-sided image is lower than that of the right-sided image.

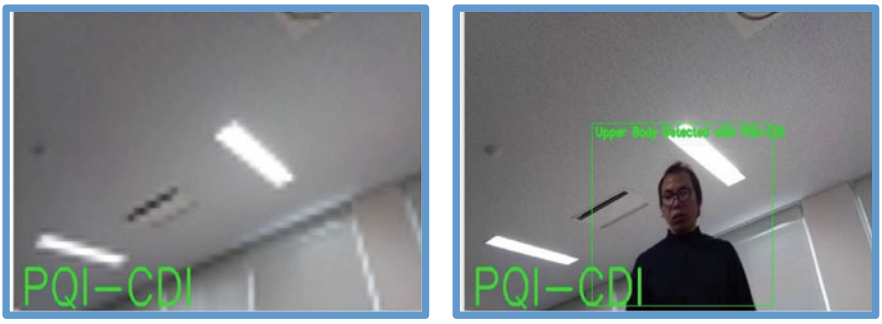


Fig. 10 Example images under the PQI-CDI method

No human detected (104064 Bytes) Human detected (104064 Bytes)

Fig. 11 Example images when the PQI-CDI method is not used

Figure 11 shows example images displayed under the conventional approach. Each camera device transmits its original image data to the processing computer without producing other quality data. The processing computer executes the human body detection processes against the clear image, which has the highest quality. The figure shows that each image shows a clear image even when no human is detected. In this case, the data sizes for each image are large compared with the case under the PQI-CDI method.

6 Evaluation of the Implemented System

To highlight the differences between the simulated performances such as the transaction time or transaction rate and the real performance, we compare the simulation results and actual results obtained using the implemented system in this section.

6.1 Evaluation Environments

We cannot strictly control the values in real situations that are needed to run the simulator such as the input bandwidth, the output bandwidth, the final probability, and the processing ratio, because the values are given as the result of the experiments. Therefore, we store the video recorded by the camera devices to the storage of each camera and use the video data for simulating the same situation. We experiment with some situations to get the result under some parameter values.

6.2 Performance Deterioration in Real Situations

We focus on two aspects that cause the performance deterioration in real situations.

First, the parameters are given by considering the average values in real situations actually fluctuate in reality. For example, we set 10 [Mbps] as the input bandwidth of the processing computer considering the realistic value. In real situations, the input bandwidth is not always 10 [Mbps] and fluctuates. Parameter fluctuations, such as the input bandwidth and the processing time ratio, cause the performance deteriorations because the communication time or the processing time lengthens when some parameter values are worse than the average values. Once a longer communication time or a longer processing time causes the overlapping of transactions, the transaction time is extended. This extended transaction time causes another overlap. Thus, the performances decrease even if the simulator uses the average values based on real situations. Averaging the results measured in the simulator using the parameters given by average values in real situation do not reflect the results in real situations.

Moreover, the simulator did not strictly simulate the communication and the processing procedures to get general results. For example, the simulator omitted the overheads that are caused by starting communications and processing such as handshaking or memory allocations. This is because it is difficult to simulate them since they depend on the implementations. Differences between the procedures in the simulator and those in real situations result in the different performances.

6.3 Transaction Time

The transaction time is one of the main performance indexes for stream data processing. Therefore, we measure the transaction time under our implemented system and compare that with the simulated values.

Figure 12 shows the average transaction time when the number of the camera devices is one. The horizontal axis represents the type of methods, no PQI (conventional method), PQI-CDI method, and simulated data. The vertical axis is the average transaction time. In this implementation, we run the system for 60 [s] for the upper human body detection and calculate the final probability as the number of image frames in which the upper human body is detected divided by the total number of received image frames.

In the actual system, we cannot fix the final probability as it depends on the objects detected by the system. To evaluate the PQI-CDI method, we ensure that human bodies appear in front of a camera to enable their detection and measure the average transaction time. If more instances of human body appearances are ensured, the system performs more detections. Thereby, increasing the final probability. In this case, two instances are considered to measure the final probability. In the first instance, the total number of received image frames is 479, among which human bodies are detected in 82 frames. In the second instance, the total number of received

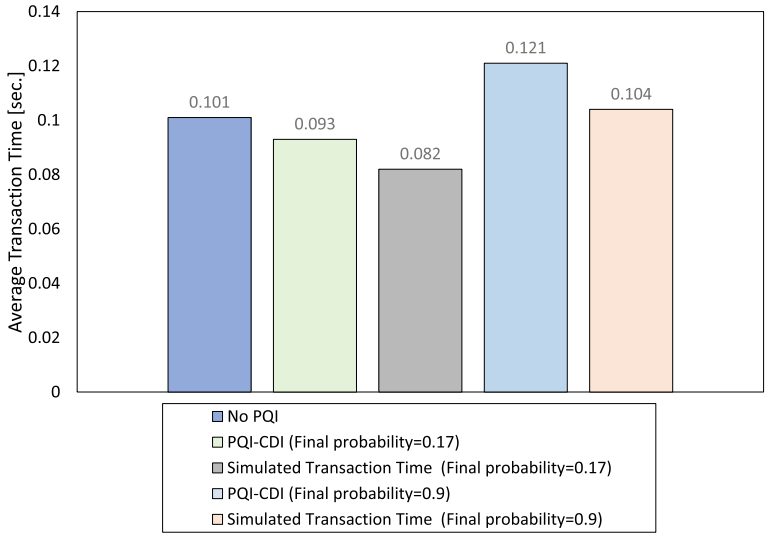


Fig. 12 Average transaction time when one camera is used

image frames is 452, among which human bodies are detected in 434 frames. Therefore, the final probabilities for human body detection are 0.17 and 0.9.

A higher final probability value corresponds to a larger number of detections made in the system. The average transaction time under the conventional approach is higher than that of the PQI-CDI method under a small final probability. This phenomenon occurs because the transmitted data amount under the conventional approach is larger regardless of the human detection. The transaction time under the PQI-CDI method increases as the final probability increases because the transmitted data amount increases. When the final probability is 0.9, the average transaction time for the PQI-CDI method is longer than that for the conventional approach because the detected objects and the number of image frames are different (a human is detected in nearly every frame, which increases the processing time in the processing computer). For the simulated data, we set two final probabilities as in the PQI-CDI method. The input and output bandwidths are 3 Mbps. The number of qualities is 10. When the final probability is 0.17, the average transaction time is shorter than those for the other approaches owing to the small amount of transmitted data. When the final probability is 0.9, the average transaction time is longer than the case where the final probability is 0.17. This phenomenon likely occurs because the input and output bandwidths in the simulator are small, and the transmitted data amount is long. Therefore, the average transaction time is high. By increasing the bandwidths in the simulator, the average transaction time can be reduced.

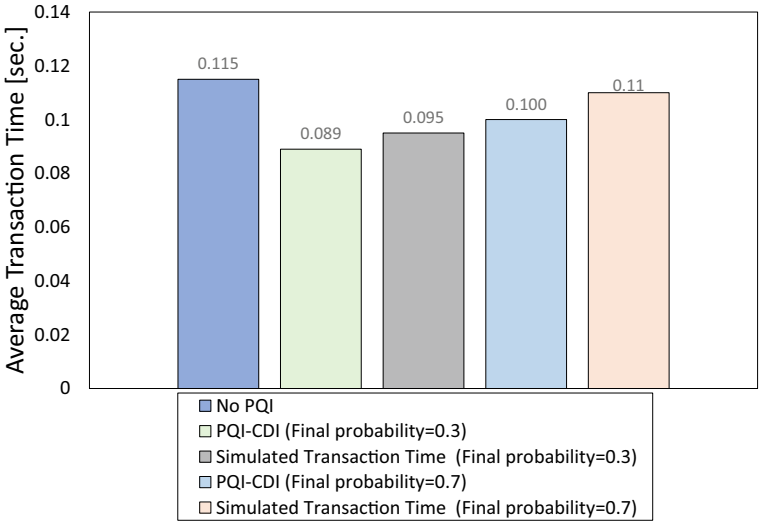


Fig. 13 Average transaction time when three cameras are used

Figure 13 shows the average transaction time when the number of the camera devices is three. The horizontal axis represents the type of methods with the specific final probabilities. The vertical axis represents the average transaction time. The transaction time under the conventional approach is the highest as image frames with the highest data quality (original data) are transmitted even when no objects are detected. Higher quality data correspond to a larger amount of data. Transmitting a larger amount of data causes the communication time to increase. Thereby, increasing the processing time. When the final probability is 0.3, the PQI-CDI method achieves a shorter average transaction time than the simulated transaction time because the processing computer does not detect human bodies in the frames in certain cases. Consequently, the processing time at the processing computer is reduced. When the final probability is 0.7, the average simulated transaction time is higher than the average transaction time under the PQI-CDI method because of the same reason as that for the case shown in Fig. 12 (small input and output bandwidths for the simulator).

6.4 Transaction Rate

Figure 14 shows the average transaction rate when the number of the camera devices is one. The horizontal axis represents the type of methods with the specific final probabilities. The vertical axis represents the average transaction rate. The other settings are the same as those of the case shown in Fig. 12. The average transaction rate under the conventional approach is the smallest because the transmitted

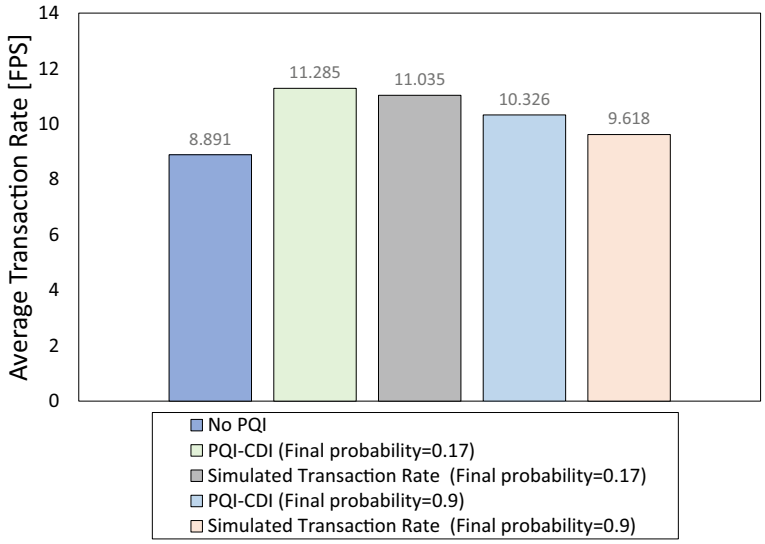


Fig. 14 Average transaction rate when one camera is used

data amount in this approach is the largest. When large data amounts are transmitted, the communication time increases, and the processing computer receives fewer transactions. In the simulation data, when the final probability is 0.17 (minimum value) and 0.9, the transaction rates are 11.035 and 9.618, respectively. In the PQI-CDI method, when the final probability is 0.17 and 0.9, the average transaction rates are 11.285 and 10.326, respectively. Notably, in the case of the simulated data, for the lowest final probability, a higher average transaction rate is achieved owing to the reduced amount of transmitted data in the simulator. Consequently, the communication time (congestions) between the cameras and processing computer is reduced, and the processing computer can perform more transactions. In the case of the PQI-CDI method, a higher transaction rate compared to that of the conventional approach is achieved when the final probability is smaller than 1. This phenomenon occurs because the transaction interval in the PQI-CDI method dynamically changes depending on the transaction time, which depends on the amount of transmitted data. Moreover, the proposed method yields a higher transaction rate than that of the conventional approach because the transaction intervals under the PQI-CDI are dynamically changed.

Figure 15 shows the average transaction rate when the number of the camera devices is three. The horizontal axis represents the type of methods with the specific final probabilities. The vertical axis represents the average transaction rate. The other settings are the same as those of the case shown in Fig. 12. The proposed method yields a higher transaction rate than that of the conventional approach owing to the same reason as that pertaining to the case shown in Fig. 14.

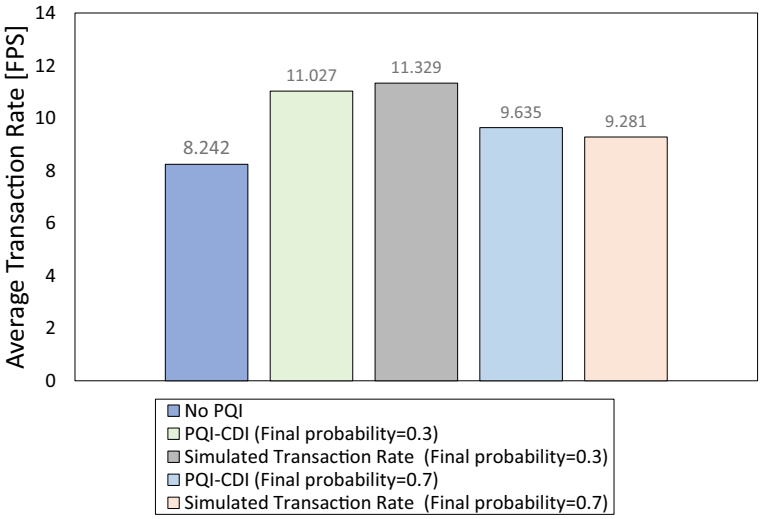


Fig. 15 Average transaction rate when three cameras are used

6.5 Communication and Processing Time

As mentioned in Sect. 6.2, the simulator does not strictly simulate the communication and the processing procedures. This results in the differences of the simulated values and the actual values in the communication time and the processing time. Therefore, we measure them. The number of cameras is three in this experiment.

In the actual simulation, when data have the first quality and are transmitted, the data amount is large because the data contain the complete image data. This case corresponds to a "large data amount". When data have other qualities and are transmitted, the data amount is small because the data only contain the different data from the first quality. This case corresponds to a "small data amount". Because the communication time depends on the data amount for the communication, different cases are considered and the specific and average results are presented.

Figure 16 shows the communication and processing times under the real situation for the cases in which the data amount is small and large, average values, and simulation results. Three cameras are adopted in this case. The vertical axis represents the average time. From this result, we can see that the average communication time under the actual situation is longer than the simulated value. This finding occurs because we set 3.77 [Mbps] as the communication bandwidth for the simulation, considering the average communication bandwidth in the experiment. However, in the real situation, the actual communication bandwidth fluctuates and is sometimes less than the average value. A smaller bandwidth causes a longer communication time. Lengthening communication time can cause the overlapping of transactions. Once a transaction overlaps, the transaction time of the next transaction lengthens.

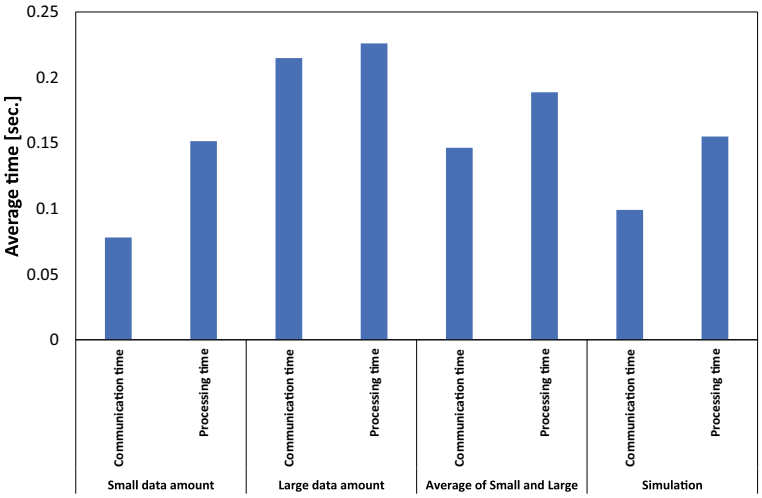


Fig. 16 Average communication time and average processing time

Thus, an overlapping transaction sometimes causes another overlap. Therefore, the transaction time increases even if the simulator uses the average values based on real situations. Moreover, in the actual situation, a longer processing time is required compared to the simulated processing time. The reason is the same as that for the communication time.

Although the simulation results are different from the results of the actual situation in terms of the parameter values, the variation tendency for the communication time and processing time agreed.

7 Conclusion

In this book chapter, we explained the Progressive Quality Improvement (PQI) approach and the extended PQI-CDI method. In the PQI approach, the processing computers progressively collect higher quality data only in cases where they are needed for processing (e.g., identifying perpetrators). By reducing the average data amount for data collection and processing, the approach reduces average transaction time. To improve the transaction rate, the PQI-CDI method changes the transaction interval dynamically under the PQI approach. Moreover, we implemented a video surveillance system and investigated the differences in the results obtained using our developed simulator and the implemented system. The experimental results indicated that the performances under the real situation were different from the simulation results although we set the parameters by averaging the values seen in the real situation. This was because the parameter values actually fluctuated in real situations.

In the future, we will focus on the stream processing system in the cases that the remote stream data sources transmit their data to some processing computers. Moreover, we will consider the power consumption both of the processing computer and the stream data sources.

Acknowledgements This work was partially supported by JSPS KAKENHI Grant Numbers JP21H03429, JP20H00584, and JP18K11316.

References

1. Buddhika T, Pallickara, T (2016) NEPTUNE: real time stream processing for internet of things and sensing environments. In: Proceedings of IEEE international parallel and distributed processing symposium (IPDPS), May 2016, pp 1143–1152
2. Beard JC, Chamberlain RD (2013) Use of simple analytic performance models for streaming data applications deployed on diverse architectures. In: Proceedings of IEEE international symposium on performance analysis of systems and software (ISPASS), April 2013, pp 138–139
3. Hu Q, Paisitkriangkrai S, Shen C, Van Den Hengel A, Porikli F (2016) Fast detection of multiple objects in traffic scenes with a common detection framework. IEEE Trans Intell Transp Syst 17(4):1002–1014
4. Navarro-Ortiz J, Ameigeiras P, Lopez-Soler JM, Lorca-Hernando J, Perez-Tarrero Q, Garcia-Perez R (2013) A QoE-aware scheduler for HTTP progressive video in OFDMA systems. IEEE Commun Lett 17(4):677–680
5. Liu W, Dong L, Zeng W (2010) Motion refinement based progressive side-information estimation for Wyner-Ziv video coding. IEEE Trans Circ Syst Video Technol 20(12):1863–1875
6. Taieb MH, Chouinard J, Loukhaoukha K, Wang D, Huchet G (2012) Progressive distributed video coding with multiple passes for side information update. In: Proceedings of international conference on sciences of electronics, technologies of information and telecommunications (SETIT), March 2012, pp 453–459
7. Shen Y, Cheng H, Luo J, Lin Y, Wu J (2017) Efficient real-time distributed video coding by parallel progressive side information regeneration. IEEE Sens J 17(6):1872–1883
8. Nazir S, Stankovic V, Attar H, Stankovic L, Cheng S (2013) Relay-assisted rateless layered multiple description video delivery. IEEE J Selected Areas Commun 31(8):1629–1637
9. Colmenares JA, Dorrigiv R, Waddington DG (2017) A single-node datastore for high-velocity multidimensional sensor data. In: Proceedings of IEEE international conference on big data (big data), December 2017, pp 445–452
10. Dias GM, Nurchis M, Bellalta B (2016) adapting sampling interval of sensor networks using on-line reinforcement learning. In: Proceedings of IEEE world forum on internet of things (WF-IoT), December 2016, pp 460–465
11. Agrawal UA, Jani PV (2017) A real-time object detecting system using compressive sensing. In: Proceedings of international conference on signal processing and communication (ICSPC), July 2017, pp 68–74
12. Zeng M, Wei Z, He H, Yang X (2018) Application of improved bhattacharyya coefficient based multi-object detection and tracking integrated strategy. In: Proceedings of international conference on network infrastructure and digital content (IC-NIDC), August 2018, pp 60–64
13. Raghunandan A, Raghav MP, Aradhya HVR (2018) Object detection algorithms for video surveillance applications. In: Proceedings of international conference on communication and signal processing (ICCSP), April 2018, pp 0563–0568
14. Ali ST, Goyal K, Singhai J (2017) Moving object detection using self adaptive gaussian mixture model for real time applications. In: Proceedings of international conference on recent innovations in signal processing and embedded systems (RISE), October 2017, pp 153–156

15. Zhang X (2010) Research on vehicle object detection in traffic video stream. In: Proceedings of IEEE conference on industrial electronics and applications, June 2010, pp 1874–1878
16. Sharma RD, Agrwal SL, Gupta SK, Prajapati A (2017) Optimized dynamic background subtraction technique for moving object detection and tracking. In: Proceedings of international conference on telecommunication and networks (Tel-Net), August 2017, pp 1–3
17. Hryvachevskyi A, Fabirovskyy S, Prudyus I, Lazko L, Matuszewski J (2019) Method of increasing the object detection probability by the multispectral monitoring system. In: Proceedings of IEEE international conference on the experience of designing and application of cad systems (CADSM), March 2019, pp 77–80
18. Jin S, Kim D, Nguyen TT, Kim D, Kim M, Jeon JW (2012) Design and implementation of a pipelined datapath for high-speed face detection using FPGA. IEEE Trans Indus Inf 8(1):158–167
19. Beard JC, Chamberlain RD (2013) Analysis of a simple approach to modeling performance for streaming data applications. In: Proceedings of IEEE international symposium on modelling, analysis and simulation of computer and telecommunication systems, pp 345–349
20. Chu D, Jiang CH, Hao ZB, Jiang W (2013) The design and implementation of video surveillance system based on H.264, Sip, Rtp/Rtcp and Rtsp. In: Proceedings of international symposium on computational intelligence and design, October 2013, pp 39–43
21. Khan IR, Hassan A, Ahsan S, Alshomrani S, Iqbal G (2017) Remote surveillance with reduced transmission overhead. In: Proceedings of international conference on frontiers of sensors technologies (ICFST), April 2017, pp 435–439
22. Lai KC, Chang YP, Cheong KH, Khor SW (2010) Detection and classification of object movement - an application for video surveillance system. In: Proceedings of international conference on computer engineering and technology, April 2010, pp 17–21
23. Wu L, Liu Z, Li Y (2012) moving objects detection based on embedded video surveillance. In: Proceedings of international conference on systems and informatics (ICSAI), May 2012, pp 2042–2045
24. Yukonhiatou C, Yoshihisa T, Kawakami T, Teranishi Y, Shimojo S (2019) a method to reduce transaction time for real-time IoT applications. IPSJ J Inf Process (JIP) 27:701–710
25. Yukonhiatou C, Yoshihisa T, Kawakami T, Teranishi Y, Shimojo S (2019) A performance evaluation of object detections by progressive quality improvement approach. In: Proceedings of IEEE global conference on consumer electronics (GCCE), October 2019, pp 321–325
26. Yukonhiatou C, Yoshihisa T, Kawakami T, Teranishi Y, Shimojo S (2019) A scheme to improve stream transaction rates for real-time IoT applications. In: Proceedings of international conference on advanced information networking and applications (AINA), vol 926, March 2019, pp 787–798
27. Yukonhiatou C, Yoshihisa T, Kawakami T, Teranishi Y, Shimojo S (2019) A dynamic intervals determination method based on transaction rates for real-time IoT applications. In: Proceedings of IEEE computer software and applications conference (COMPSAC), July 2019, pp 7–12
28. Yukonhiatou C, Yoshihisa T, Kawakami T, Teranishi Y, Shimojo S (2020) A fast stream transaction system for real-time IoT applications. Elsevier Internet of Things J 11:701–710

Chapter 8
Explainable AI for ICT: System and Software Architecture

Devam Dave, Het Naik, Smiti Singhal, Rudresh Dwivedi, and Pankesh Patel

Abstract Artificial Intelligence (AI) has become a revolution in the ICT domain due to the swift progress of analytical techniques and the availability of structured/unstructured data. With the indispensable role of AI in different applications, there are growing concerns over the lack of transparency and explainability. In addition, potential bias may affect the predictions of a model. This is where Explainable Artificial Intelligence (XAI) comes into the picture. XAI increases the trust placed in an AI system by researchers, medical practitioners, and others. Thus, it leads to widespread deployment of AI in healthcare, agriculture, online mart, and many more. The aim is to enlighten practitioners on the understandability and interpretability of EAI systems using a variety of techniques available which can be very advantageous in the ICT domain. In this chapter, we present two different techniques leveraging EAI where a user has to make the right choices based on his requirements. The software architecture of the first techniques is based on a medical diagnosis model where we need to be confident enough to treat a patient as instructed by a black-box model. Another approach presents an online Mart where a reliable pricing method can be developed by ML models that can read through historical sales data. The objective here is to match buyers and sellers, to weigh animals, and to oversee their sale. However, when AI models suggest or recommend a decision, that in itself does

D. Dave · H. Naik · S. Singhal
School of Technology, Pandit Deendayal Energy University (PDEU), Gandhinagar, India
e-mail: devam.dce18@sot.pdpu.ac.in

H. Naik
e-mail: het.nce18@sot.pdpu.ac.in

S. Singhal
e-mail: smiti.sce18@sot.pdpu.ac.in

R. Dwivedi (✉)
Department of Computer Science and Engineering, Netaji Subhas University of Technology (NSUT), Delhi 110078, India
e-mail: rudresh.dwivedi@nsut.ac.in

P. Patel
AI Institute, University of South Carolina, Columbia, South Carolina, USA
e-mail: ppankesh@mailbox.sc.edu

M. Chaturvedi et al. (eds.), *Recent Advancements in ICT Infrastructure and Applications*, Studies in Infrastructure and Control, https://doi.org/10.1007/978-981-19-2374-6_8

not reveal too much (i.e., it acts as a black box). Hence, a model capable of explaining the different factors that impact the price point is essential for the needs of a user.

Keywords Explainable AI · Healthcare · Programming frameworks · Video Analytics · Internet of Things · vision-based feature extraction · ML-based price prediction

1 Introduction

With the growth of AI and ML and the deployment of advanced models in novel application domains, hitherto unasked questions and challenges come to the fore. One such question arises where it is not easy to understand their inner workings, their algorithms and prediction reasoning seemingly stand unexplained. This is also fuelled by the need for increased efforts at removing bias where discriminatory features are found to be used by the model in the process of prediction. These two perceptions have led to increasing levels of distrust on ML/AI systems that are in use. Interpretability has also played an important role in applied ML over the years, but as black-box techniques such as DL take wings, it has assumed the form of concerns requiring urgent attention. As such, the field that attempts to fix these problems goes by the name of Explainable AI (XAI) [4]. To address these issues, XAI proposes to make a model interpretation that ensures the predictive model is fair (without bias or discrimination), accountable (reliability of results), and transparent (able to be queried and validated).

In this chapter, we present a case study that aids explainability on predictions related to livestock mart and predictions related to heart disease by medical practitioners. First of the two, determines the price of a cattle in a livestock mart based on certain essential features such as weight, age, and other similar qualities. In a black-box model, due to the absence of these features, a fair price cannot be estimated. This lack of transparency eventually leads to price exploitation and excessive commissions by the agents responsible for buying and selling the livestock. The presence of XAI in these black box price prediction models would not only lead to fair pricing but would also make buyers and sellers trust the model in terms of the prices offered. Another case study is based on healthcare applications where explanations are important for people who make decisions. If AI cannot explain itself in the domain of health care, then its risk of making a wrong decision may override its advantages of accuracy, speed, and decision-making efficacy. This would, in turn, severely limit its scope and utility. Therefore, it is very important to look at these issues closely.

The rest of this chapter is structured as follows: Section 2 provides a discussion on XAI frameworks and usecases. Section 3 presents a discussion on requirements, procedures, and outcomes for two case studies. Section 4 includes a discussion on explainability supported towards prediction made towards livestock mark and healthcare case studies. Section 5 concludes the chapter.

2 XAI Frameworks and Usecases

2.1 Smart Analytics

Smart learning has emerged as a new term to portray technological and social developments (e.g., Internet of things, Big Data, RFID, and NFC) which enables efficient, engaging, effective, and customized learning. "Smart" notion refers to narrate interconnection, synchronization, and rationale use of different technologies that constitute smart behavior [5]. These technologies and state-of-the-art approaches are centred on data collection, data-sharing, and decision-making. At present, the majority of learning schemas are affined to video affordances. Hence, livestream data or learning from image frames come to the fore in most of the contemporary learning systems. While the schemes on smart learning rely on sensors, large data, interconnections, and the exchange of information, there is still a scarcity of empirical analytics-based investigation. Therefore, it is required to adopt empirically driven research in the area of Smart Learning Analytics where interpretability and explainability play a major role to ensure the predictive model is fair (without bias or discrimination), accountable (reliability of results), and transparent (able to be queried and validated). To review systematically and summarize the different aspects, Liñán et al. [1] recognized the underlying components of learning analytics: modeling user knowledge, behavior, and experience; creating user profiles; modeling knowledge domains; trend analysis; and personalization and adaptation.These components exhibit the importance of learner knowledge, domain knowledge, and pedagogical knowledge in learning analytics. Siemens et al. [9] defined the analytics in terms of utilization of intelligent data, learner-produced data, and analysis models to outcrop information and social interconnections for prediction. In the following, they presented a smart learning model analytics (Fig. 1) where decision-making (predictions) encompasses connected knowledge, semantic, and linked data. It is also considered to include social media, physical, learning environments for user profiling. Next, predictions are made based on user profile and knowledge base followed by personalization and adaptation.

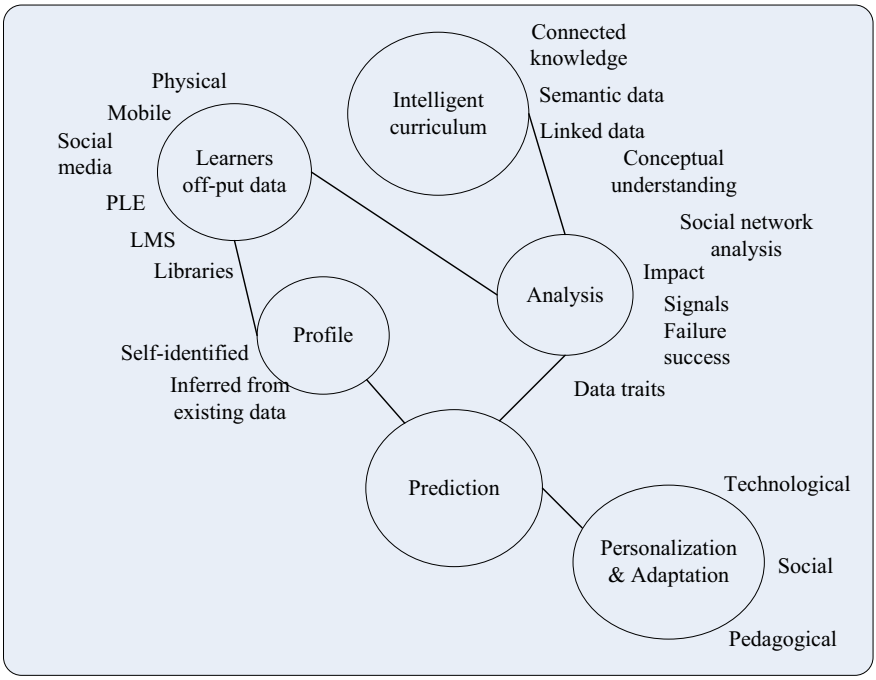


Fig. 1 Integrated knowledge and learning analytics model with intelligent curriculum

2.2 Machine Learning Operations

Machine Learning Operations (MLOps) [10] aim to create a mechanism that involves tools, pipelines, and different related tasks to develop applications/projects. The project lifecycle includes a set of roles, software frameworks, and resources. For a machine learning lifecycle, the four different phases (see Fig. 2) are defined:

(1) model programming, (2) model training, (3) model assessment, and (4) model releasing.

The first phase involves a reusable code template, an online IDE for model coding, utilities to access online data repositories, and GPU/CPU computing choices. The model training phase encompasses cluster resources and scheduling managers. Further, models are trained, updated, and fine-tuned. After training and validation, it is fed to the next phase i.e. model assessment. Model evaluation/assessment may utilize multiple databases to evaluate the model performance and explainability. It is also validated that how much accurate and imperative a model is. Therefore, model assessment must require akin workflow involving data access, computing resources, and model execution. Finally, the model release is approved either to a repository or to a hosting application as a service.

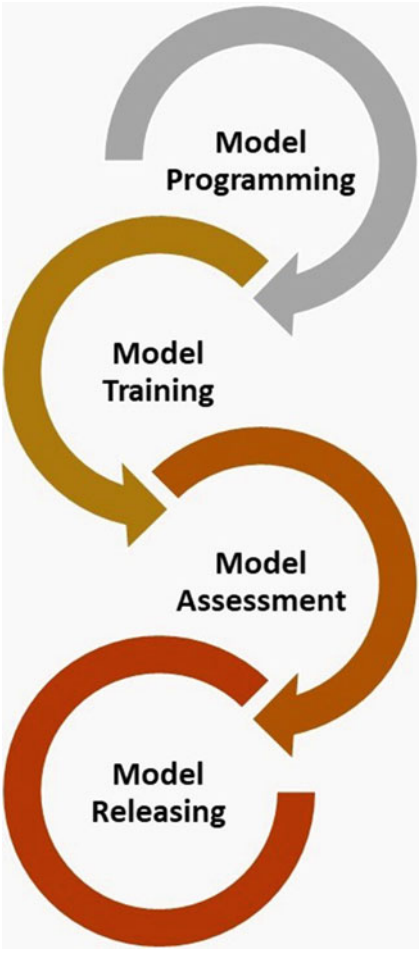


Fig. 2 Phases in ML lifecycle

2.2.1 Roles and Cooperation Context

In MLOps, each phase is handled by roles and these roles integrate across phases. Figure 3 illustrates the roles and their cooperative activities. The activities in different context are discussed in the following:

Data scientist context: This context requires model creation and model training, i.e., the first two phases of the model lifecycle. There are three different working spaces that span over private, public, and staging spaces. The data may be shared by a data scientist to engineers, project managers, or developers under public space. In addition, it may be kept isolated to work in a private space.

Manager context: In this context, the manager evaluates model performance before it is fed to release. First, a model is retrieved after a private test performed by a

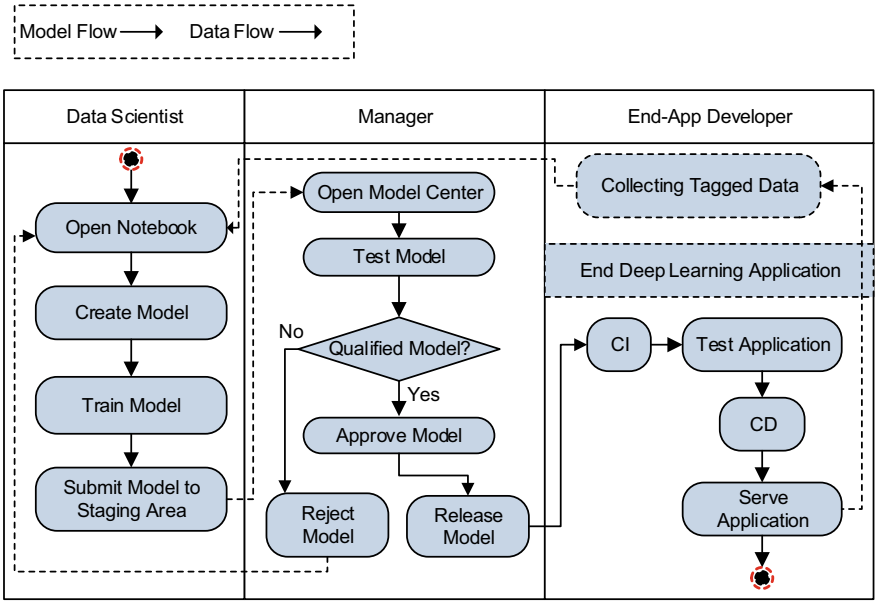


Fig. 3 Cooperative operations in different phases

data scientist in a personal working space. The manager's operations are supported by staging space. In staging space, a manager downloads a model and tests it over multiple datasets. Finally, it is decided whether it should be approved for release or not. If it is not approved, it may be rolled back to the scientist's private space for correction.

End application developer context: The context is associated with model deployment. In essence, the last phase of MLOps approves and releases a model from the staging phase to the end-user.

2.3 Deep Learning for IoT Application

The integration of deep learning in IoT are rapidly emerging area at present [2, 8]. Executing neural networks on resource-constrained IoT devices is challenging. One of the most prominent reasons behind this is the limited availability of memory and processing capabilities. It results in a fewer number of operations per second. On the other hand, neural network models include millions of parameters requiring large storage. Further, it has no support for floating-point operations making it limited to parallel processing. Finally, such models activate a considerable number of arithmetic operations and memory accesses for a single inference which leads to more power consumption and heat dissipation. Therefore, there is a dire need for IoT devices

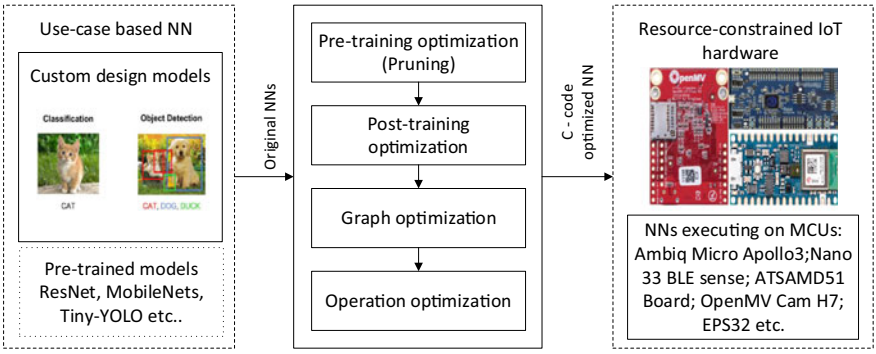


Fig. 4 The architecture of our multi-component NN optimizer

in multiple aspects to produce small size, low latency, and low-power consumption hardware. A lot of efforts in the past few years have made it convenient to utilize IoT devices to execute deep learning models with a focus on optimizing ML models to enable their execution on a small Microcontroller Unit (MCU) [8, 11]. In Fig. 4, we present a multicomponent NN optimizer. It includes three stages as discussed in the following:

Usecase-based NN: The optimizer takes a neural network as input and produces a highly optimized version of the input neural network that can run on low-cost resources and power MCU-based IoT hardware [2, 6]. The input network can be a custom or a pre-trained model such as Inception, Xception, Mobilenet, Tiny-YOLO, Resnet, etc., deployed to fulfill a specific task (e.g., animal detection, object classification).

Pruning and Quantization Aware Training (QAT): The custom-designed, yet to be trained networks should pass through the pre-training optimization component, i.e., pruning. In model pruning, compression involves the elimination of a few model weights and inference requirements. The post-training quantization is utilized to compress model size for deployment over edge devices, allowing inference speed up and reducing power dissipation without significant accuracy losses. The quantization also performs computation and stores tensors at lower bit widths (e.g., INT8 or FLOAT16). Quantization-aware training applies the quantization during training to achieve speed up inference and reduce memory requirements at the deployment stage. The third phase, graph optimization, is used for inference over graphical models. The interior of trained models is a graph with defined data flow patterns with an arrangement of nodes and edges. Nodes represent the operations of a model and edges represent the flow of data between the nodes. Next, we perform graph optimization in sequential steps, first arithmetic simplification is performed followed by graph structure optimization. Further, operation optimization includes kernel and pooling optimizations which aid kernels to design a parallel processing architecture for multiple kernel execution. Finally, the deployment of optimized models is carried out

through model translation and application integration. Model translation converts the quantized version of the trained model into a C-array and compiles it along with the program for the IoT application which is to be executed on the MCU-based hardware devices.

Resource-constrained IoT hardware: Microcontroller Units (MCU) are essentially small-scale processing units such as Arduino Uno which are used to carry out specific operations in an embedded system. Specific neural networks are executed on these MCUs. The resource-constrained IoT hardware consists of Ambiq Micro Apollo3, Nano 33 BLE sense, ATSAMD51 Board, etc.

3 Case Studies

3.1 Explainability with Heart Disease Use Case

Explainability techniques are capable of a plethora of intuitive visualizations for the causes leading to the predictions revealing how the model decided a particular outcome. The trust in AI models can thus be escalated with the help of human practitioners who have prior knowledge related to the features of the dataset. For instance, in the case of a person suffering from heart disease, a medical practitioner can verify if the prediction of a model aligns with the conventions by examining the explanations returned by XAI techniques [3, 6].

This case study presents the explainability techniques that help to describe the predictive modeling in AI applications, especially in the healthcare domain. These smart healthcare systems integrated with XAI can then be implemented for diagnosing diseases and determination of the most suitable treatment plan.

3.1.1 Methodology and Dataset Descripton

Our case study focuses on the research of building EAI-based approaches for healthcare applications. It elucidates how the historical healthcare data collected for training is consumed to learn latent patterns and relationships within the data. The Heart Disease Dataset used is taken from the UCI ML Repository, which consists of 76 features. The main aim is the detection of patients prone to heart disease. The models generated from training data are deployed wherein the XGBoost (eXtreme Gradient Boosting) algorithm is implemented. It is an algorithm based on gradient boosted decision trees best suited for not only performance in classification problems but also speed. An integral part of XAI is feature importance which is delivered by one of the only algorithms, XGBoost. After the deployment, the goal is to understand how the outcomes are returned. These explanations can be useful for medical practitioners in validating the model.

Our dataset, Heart Disease Cleveland UC Irvine dataset is formed on 13 attributes and is the re-processed version of the original set of data having more than 70 features. The features are as follows[1]:

1. *age*: It denotes the age of an individual and is of numeric datatype.
2. *sex*: Binary value for male (1) and female (0).
3. *cp*: cpshort for Chest Pain Type consists of values ranging from 0 to 3 which include 0—Typical Angina, 1—Atypical Angina, 2—Non-Anginal Pain, 3 Asymptomatic Pain.
4. *trestbps*: *trestbps* signifies the resting blood pressure computed in units of millimeters in mercury (mmHg).
5. *chol / cholis*: The cholesterol levels of an individual.
6. *fbs*: *fbs* means Fasting Blood Sugar levels of a patient, measured using gauging the amount of glucose present in the blood. Here, 1 signifies that the patient has a blood sugar level above 120 mmol/L and the opposite for 0.
7. *restecg*: *restecg* represents the electrocardiograph results of a patient, ranging between 0 and 2.
8. *thalach*: *thalach* depicts the maximum heart rate of an individual using a Thallium Test.
9. *exang*: *exang* is a feature that reveals whether a patient has exercise-induced angina (signified by 1) or not (signified by 0).
10. *oldpeak*: *oldpeak* is the amount of depression of the ST Wave in the case of a patient having a value of 1 in *restecg*.
11. slope: slope is also concerned with the ST Wave in ECG Results. (a) 0 signifies an upward slope in the ST Wave, (b) 1 signifies that the ST Wave is flat, (c) 2 signifies a downward slope in the ST Wave.
12. *ca*: Angiography is performed which shows up the blocked blood vessels on an X-Ray. In ideal cases, the three vessels should be visible on the X-Ray. Angioplasty should be carried out for the treatment of blocked vessels if *ca* is high.
13. *thal*: *thal* is the individualistic results of the Thallium test in which 0 denotes that the results were normal, 1 denotes a fixed defect, and 2 denotes a reversible defect.

Figure 5 shows a snapshot of the heart disease dataset. We present the case study further using this dataset.

3.1.2 Feature-Based Techniques

Local Interpretable Model Agnostic Explanations (LIME).
LIME is a local and model agnostic EAI technique. A local XAI technique is a technique that does not take the whole dataset and its predictions into account, but instead, interprets and provides an explanation of a specific instance of the dataset

[1] Heart disease dataset: https://www.kaggle.com/cherngs/heart-disease-cleveland-uci.

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	condition
0	69	1	0	160	234	1	2	131	0	0.1	1	1	0	0
1	69	0	0	140	239	0	0	151	0	1.8	0	2	0	0
2	66	0	0	150	226	0	0	114	0	2.6	2	0	0	0
3	65	1	0	138	282	1	2	174	0	1.4	1	1	0	1
4	64	1	0	110	211	0	2	144	1	1.8	1	0	0	0

Fig. 5 Example instances of dataset

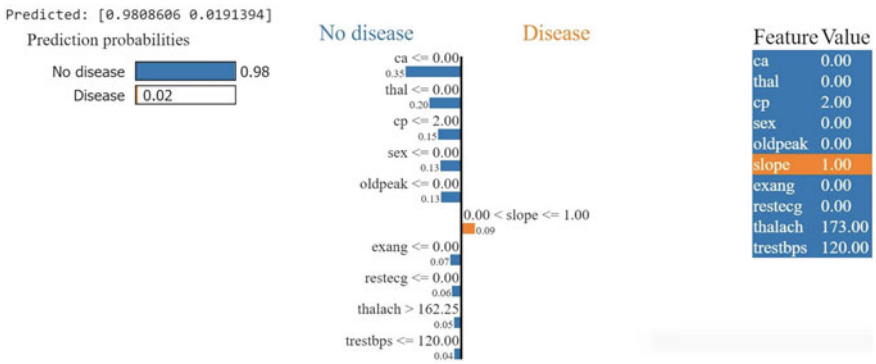


Fig. 6 Local explanation generated by LIME

and its respective prediction by the model. As LIME is a model agnostic technique, which is to say that LIME can be applied and used to interpret any black-box model. LIME works by altering values of various features in a local instance and then comprehending the changes in the prediction because of these changes. An example has been provided in order to further demonstrate LIME and its capabilities [7]:

The right side of the figure (Feature Value column) shows the values of each feature for this instance. The left side (Prediction Probabilities column) shows the predictions made by the model, which in this case is an XGBoost Model. The central part of the Fig. 6 illustrates that the most influential feature value contributing to the prediction made by the XGBoost model that the patient does not have a disease is that *ca* is equal to 0. Similarly, thal being equal to 0 and *cp* being equal to 2 also play influential roles in the prediction.

SHapley additive exPlanations (SHAP).

SHAP is an XAI technique that is used to measure the influence of a specific value of a certain feature in respect to the prediction. Shapley comes from a game theory concept where the profits of the winning team's players are fairly divided. In our case, features of the dataset can be termed as players and prediction can be correlated with the winning amount.

An output of the SHAP plot of an instance is shown below. The features colored red push the prediction to the higher (right) side and the blue features colored blue push the prediction to the lower (left) side.

From the plot, we can see that ca (number of blocked vessels) is the most important feature contributing the prediction to be 0 (person not having a heart disease). Also, slope, with feature value 1 is the only feature pushing the prediction towards the person having a heart disease.

3.1.3 Example-Based Techniques

Anchors.
Anchors technique is capable of explaining the predictions of text, image, and tabular data with the use of decision rules which anchor or guarantee the prediction sufficiently. Furthermore, it tries to maximize the region where these rules hold true. This means that regardless of the changes in other feature values absent in the area, the outcome predicted should continue to be the same. Anchors are easily interpreted and do not require expert knowledge. They are model agnostic and can even work for nonlinear and complex predictions. However, they require a highly configurable design and involve a lot of hyperparameter tuning with many calls to the machine learning model (Fig. 4).

We take an instance of a person whose value of *thalach* is 131 and *ca* is 3. We now apply the AnchorTabular method to see which features contribute significantly for such type of prediction.

The person's *thalach* (maximum heart rate) is 131 (which is less than 138). The value of *ca* (blood vessels colored by fluoroscopy) is 3 (greater than 1). The above features act as anchors for the patient and deduce that the person has a heart disease.

Counterfactuals.
Counterfactuals are an XAI technique for classification dataset (best suited for binary datasets), is responsible for flipping the outcome of prediction by changing as few features as possible in addition to the fact the counterfactual values acquired should be likely and feasible. There can be multiple possible counterfactual explanations for reaching a particular output that might contradict each other but be equally successful in providing the resultant target value. An extended method of Counterfactuals, Counterfactual with prototype uses a sample representative of instances to provide explanations. It leads to a much faster and interpretable approach to the convergence of algorithms without a lot of hyperparameter tuning. A use case of counterfactuals and counterfactual prototypes in healthcare can be to identify the minor changes in the features patients can make to increase their chances of being not diagnosed with heart disease (Figs. 8 and 9).

```
Person has heart disease
Anchor generated feature(/s) ['thalach <= 138.00', 'ca > 1.00']
```

Fig. 7 Example: an output of anchor

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	condition
0	67.0	1.0	3.0	120.0	229.0	0.0	2.0	129.0	1.0	2.6	1.0	2.0	2.0	0.99965

Fig. 8 Input provided to counterfactuals

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	condition
0	67.0	1.0	3.0	94.0	229.0	1.0	2.0	129.0	0.0	2.6	1.0	0.0	0.0	0.283
1	67.0	1.0	3.0	120.0	186.0	1.0	1.0	124.0	0.0	2.6	2.0	0.0	0.0	0.159
2	67.0	1.0	3.0	120.0	229.0	1.0	1.0	172.0	0.0	3.1	1.0	0.0	2.0	0.315
3	67.0	1.0	3.0	120.0	229.0	0.0	1.0	129.0	0.0	0.6	1.0	0.0	0.0	0.274

Fig. 9 Output generated by counterfactuals

```
Feature names: ['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal'
Original instance: [[-1.3859065  -1.44454157 -0.16401266 -0.65831955 -0.73753753 -0.41075703
  -1.00172798  1.02001221 -0.695246   -0.90518389  0.64269638 -0.72075958
  -0.87281841]]
Predicted class: [0]
Pertinent negative: [[-1.4474772  -1.4445416   0.24188966 -0.65831953 -0.7375375  -0.41075704
  -1.0017279   1.0200123  -0.5739004  -0.9051839   0.6426964   1.7650422
   0.5487773 ]]
Predicted class: [1]
```

Fig. 10 Pertinent negative explanations generated by CEM on heart disease dataset

```
Original instance: [[-1.3859065  -1.44454157 -0.16401266 -0.65831955 -0.73753753 -0.41075703
  -1.00172798  1.02001221 -0.695246   -0.90518389  0.64269638 -0.72075958
  -0.87281841]]
Predicted class: [0]
Pertinent positive: [[-4.63393008e-08 -1.09832214e-02 -4.27903346e-09 -8.10524396e-02
  -2.21655790e-08  9.42199779e-09 -9.06142362e-02  1.00736806e-01
  -2.08339260e-08 -2.68793621e-01 -3.37093253e-09 -8.50149595e-09
  -3.48234908e-09]]
Predicted class: [0]
```

Fig. 11 Pertinent positive explanations generated by CEM on heart disease dataset

Contrastive Explanation Method.
The Contrastive Explanation Method is used to get local explanations of a black-box model. It is very useful in the scenarios where we need to know for an instance about what should be the minimum value present and also what values should be absent to keep the prediction class unchanged. We can illustrate this by providing sufficient pixels of an image through which the model predicts the same class as the original.

The two kinds of explanations are illustrated using the example (Fig. 10).

Pertinent Negative:
Pertinent Negative explanations can be understood taking the case of counterfactual explanations as both of them generate similar types of outputs. In this example, we can see that the prediction class is changed from 0 to 1 on applying CEM with pertinent negative. From the values, we can say that features such as *cp*, *ca*, and *thal* should necessarily not change to retain the original prediction as 0 as they are responsible for generating a counterfactual (Fig. 11).

Pertinent Positive:
As Pertinent Negatives are similar to counterfactual explanations, pertinent positives are similar to anchors. From the example, we can see the pertinent positives generates the feature values that should necessarily be present to retain the original class as predicted class.

As the CEM values are close to 0, we can say that they are the most minimal and compulsory to retain original class 0 as the predicted class.

3.1.4 Discussion

The healthcare industry is a high-stakes industry where a medical diagnosis is a sensitive subject and one seemingly insignificant error made by a machine learning model can have significant, major repercussions. In this case study, the importance of EAI, especially in the healthcare and medical industry has been showcased by utilizing a variety of XAI techniques on a heart disease dataset.

Each technique has its specialities, advantages, and disadvantages. There are special use cases for each technique and cases that require a niche that one of the techniques might provide. LIME and SHAP are arguably the two most accepted and widespread XAI techniques. They are both similar in some aspects while also being extremely different in other aspects. LIME is a local XAI technique, while SHAP can give explanations that are both local as well as global. There are two techniques similar to SHAP, namely, Tree Shapley and Kernel Shapley. Kernel Shapley has been discussed in the preceding section. Many diverse techniques such as Anchors, Contrastive Explanation Methods, Counterfactuals, and Counterfactuals with Prototypes and Integrated Gradients were also discussed in the previous section. From all of these techniques, we observe that each feature of a dataset or pixel of an image plays a role in the predictions made by the model. These techniques can be used to explain black-box models and why and how it makes the predictions that it does.

3.2 Explainability with Livestock Mart Industry Usecase

In this case study, a team has worked with MartEye, an AI firm based in Ireland specializing in live auctions of livestock as well as machinery for farmers to include an aspect of XAI in their software. This case study, opposed to the previous case study, shows XAI being used on video-graphic data on a smart video analytic platform built alongside it.

3.2.1 Methodology and Early Results

Figure 12 gives an overview/outline of the approach and methodology that was followed.

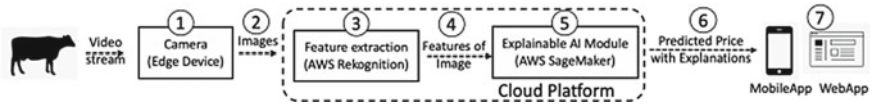


Fig. 12 An outline of our approach

1. **Edge Device.** A CCTV camera is present at the site of the livestock, which produces real-time video footage. This video footage is then uploaded to an edge device connected to the Amazon Web Services (AWS) cloud platform. Using various Computer Vision techniques, video pre-processing is carried out and the redundant and unnecessary frames of the video are removed by the edge device before the video is uploaded to the cloud.
2. **Feature Extraction.** The processed video has been uploaded to AWS Kinesis. Now, AWS Rekognition, a cloud-based software specializing in computer vision is used to analyze the video. Following analysis of the video, the major salient features of livestock such as age, weight, and height are extracted by using classical computer vision techniques as well as computer vision techniques based on a CNN.
3. **Price Prediction.** As the major features have been extracted by AWS Rekognition, the price of an animal can be predicted. A multivariate linear regression machine learning model has been trained by the team using data supplied by MartEye. The data consists of a variety of datasets from different livestock markets all over Ireland. The price of the livestock is the target variable and more than 20 different features are utilized as independent variables. AWS Sagemaker is used for the building, training, and deployment of the aforementioned machine learning model.
4. **Explainable AI.** This is where EAI comes into the picture. The proposed method is different in the fact that it not only provides a prediction of the price of an animal but also provides an explanation as to why the model arrived at the prediction that it did. A melange of XAI techniques has been used to assist the explainability and interpretability of the model.
   LIME, which has been talked about in the previous case study, was used extensively as an XAI technique in our approach. In Fig. 13, we see an instance of explanation provided by LIME.
   In Fig. 13, the left part signifies the prediction of price made by the machine learning model deployed by us. As we can see, the model predicted the price to be EUR 697.95. The central part of the figure is paramount to explaining the predictions. Features can either influence the prediction positively (increasing the price), or negatively (decreasing the price). WT influences the price of livestock negatively, whereas PPK influences the price positively. The influence of WT is greater than PPK, as can be seen in the figure. The right part depicts the values of two features, WT and PPK for this particular instance.

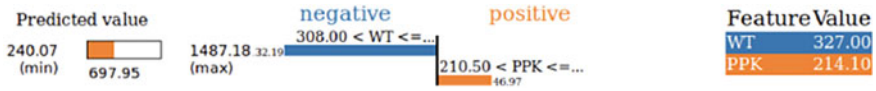


Fig. 13 LIME-generated explanation

3.3 Video Analytics for Weight Estimation

1. **Teachable Machine.** A teachable machine is a tool created by Google that allows for the creation, training, and deployment of machine learning models in a hassle-free manner. A Teatchable machine can train models for video-graphic and phonetic data. A teachable machine-based model was used by the researchers in three different phases.

   - Phase I: Single Point of View Images:
     In this phase, the images used to train and test the model were from a single POV, which was a side view of the cow. The target classes were divided in such a way that each class had a range of 200 kgs. Approximately 25–35 images were taken for each class. The accuracy of this method turned out to be 92%. Figure 14 shows an example of Phase I at work. As we can see on the left column of the output, cows were divided into ranges of weights of 200 kgs. Both, a correct prediction and an incorrect prediction have been shown in Fig. 14 on the left and right sides, respectively.
   - Phase II: Single Point of View Images with more classes:
     In this phase, similar to Phase I, the images used to train and test the model were all a side view of the cow. The target classes were divided in such a way that each class had a range of 100 kgs, thereby increasing the total number of classes. Around 25 images were taken for each class to train the model. In Phase II, the accuracy improved to 95%. In similar fashion to Figs. 14, 15 shows a correct prediction and an incorrect prediction on the left and right sides, respectively. Moreover, we can see that the range of output classes has decreased from 200 kgs in Phase I to 100 kgs in Phase II.
   - Phase III: Multiple Point of View Images:
     In this phase, instead of images captured from a single point of view (side view), the model was trained with images spanning from multiple points of view such as front view, back view, diagonal view, etc. The target classes were divided into 100 kg ranges likewise Phase II. The accuracy for Phase III was 65%. In Fig. 16, the predictions made by Teachable Machine for Phase III are shown.

   It can be observed from the three phases that the accuracy of the model is directly correlated to the POV of the training dataset. The researchers observed that side view provided a better accuracy as compared to other points of view. Moreover, they also observed that color did not play a significant role for the evaluation of accuracy, which makes the model unbiased.

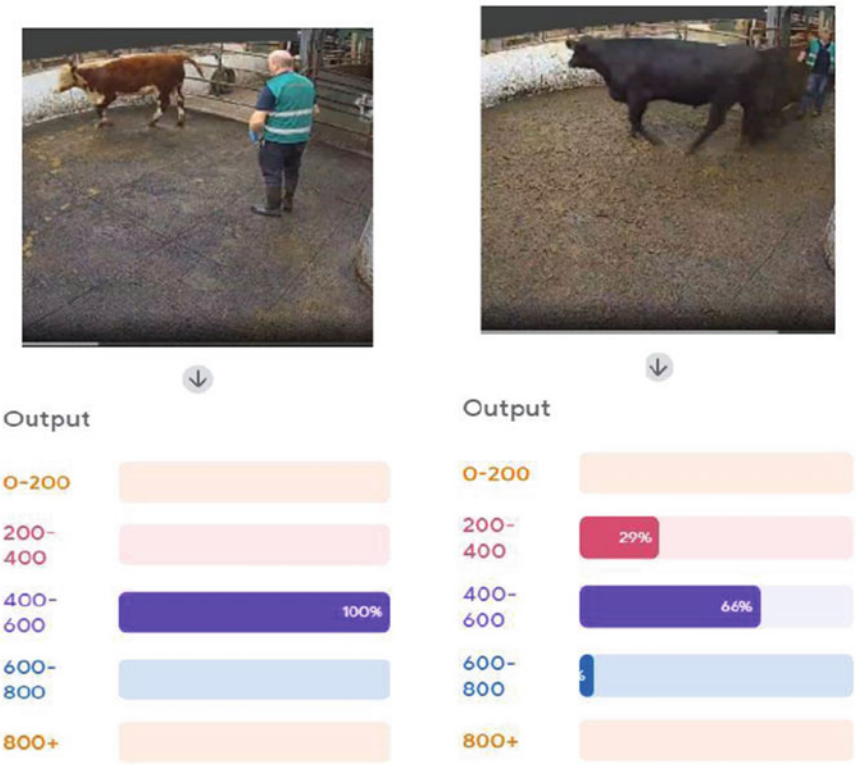


Fig. 14 Predictions made by teachable machine in phase I

2. **Body Mass Index (BMI) method** This method uses a ResNet not only because of the quality of performance it provides in aspects related to computer vision and object detection/classification but also for the tremendous efficiency that it provides in terms of speed. Initially, this method was used on humans and produced excellent results in figuring out the BMI and other traits of a human being using nothing but an image of their face. The researchers tried to simulate the same results in cows but were unsuccessful. This was attributed to three main factors, which have been mentioned below:

    - In humans, a variety of predictions can be made based on an image of the face of an individual. For example, a model can potentially successfully predict the approximate weight, age, sex, and the BMI of an individual. However, it is quite difficult to emulate these results in cows because an image of the face of a cow is not enough to determine the weight, age, and even the sex of a cow. Additional images of the structure of the body are required to make these predictions.

Fig. 15 Predictions made by teachable machine in phase II

- Upon training the model using images of the faces of different cows, the model gets biased. The model starts to predict the weight based on the color of the cow instead of the facial features and fat storage in the face of a cow. This leads to a decrease in the accuracy and performance of the model.
- The video-graphic data is quite low resolution, which leads to blurry images of the face of the cow. These images are not suitable to be used for training the model.

4 Discussion

We have discussed the most popular feature-based techniques LIME and SHAP which are very similar to each other in purpose but have very different approaches. Unlike LIME, which is only capable of local explanations, SHAP can explain both globally and locally. Multiple types of plots that explain the dataset globally are drawn using the dataset which provides information about relationships between the features

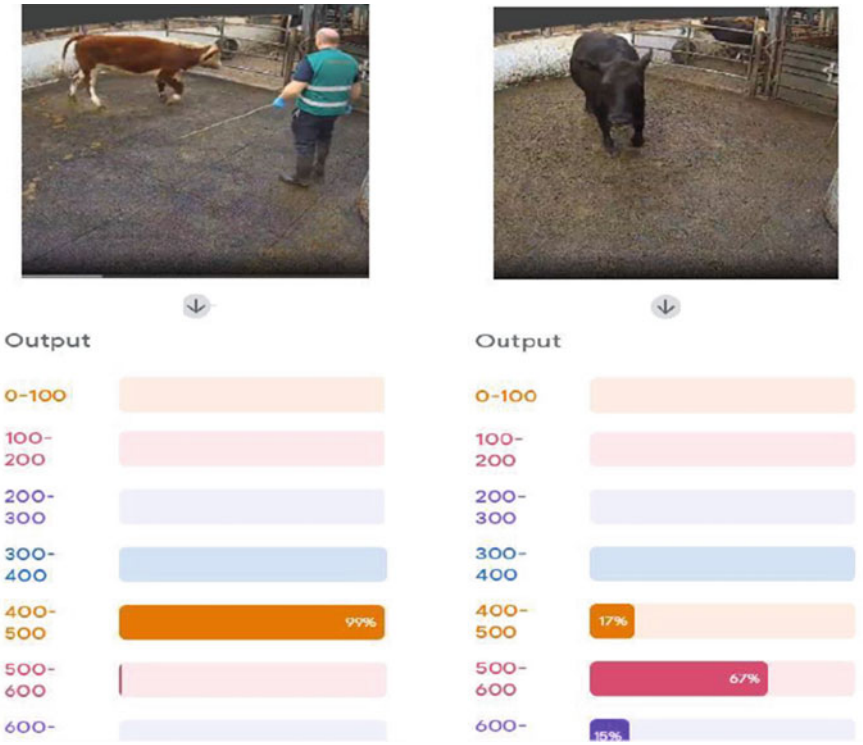


Fig. 16 Predictions made by teachable machine in phase III

and their importance. In a later section, a similar technique Kernel Shapley is discussed which is mathematically better and gives results in fewer evaluations. We then propose and discuss the different example-based techniques namely Anchors, Counterfactuals, Integrated gradients, Contrastive Explanation Method, Kernel Shapley that play an integral role in revealing the black-box nature for model transparency and accountability. Anchors, based on If-then rules, are better than LIME and SHAP in terms of generalisability and computation cost, respectively. Counterfactual mainly presents the what-if explanations and its improved faster version Counterfactual with Prototype uses the prototypes of classes of target variable for these explanations. While counterfactuals say how the features should change to change the target variable, CEM does the opposite and talks about what should necessarily be present to prevent the prediction to flip. CEM is a unique technique that provides Pertinent Positive—minimally present and Pertinent Negative—compulsorily absent features for the original prediction class. The techniques so far only explain how the features present give rise to the prediction and say nothing about the features that influenced to decide against the predicted class. This is what Integrated Gradients implement. Along with the positive attributions that help to make a prediction, it is the only feature that mentions the features that lead the model to believe a different prediction,

known as negative attributions. Thus, the most important benefit learned from the study of these techniques is that the approaches all speak about how various features are responsible for the model's outcomes. They are intuitive and thus assist in the process of learning what the black-box model thinks and being able to explain the behavior of the model.

5 Conclusion

In this chapter, we have talked about the case studies which encapsulate the healthcare and livestock mart domain. The aim is to enlighten practitioners on the understandability and interpretability of EAI systems using a variety of techniques available which can benefit a user for certain decisions. The medical diagnosis model is responsible for human life and we need to be confident enough to treat a patient as instructed by a black-box model. Similarly, it is essential to understand data that determine animal's true value and to oversee their sale. To ensure fair and just pricing of the animals, careful modeling and appropriate transparency consideration for the valuation and pricing of animals should aid in stopping kickbacks, exorbitant commissions, and price manipulations. Our case studies based on heart disease identification and livestock mart price prediction elucidate how the explainability techniques should be preferred to create trustworthiness while using deep learning systems.

References

1. Calvet Liñán L, Juan Pérez ÁA (2015) Educational data mining and learning analytics: differences, similarities, and time evolution. Int J Educ Technol Higher Educ 12(3):98–112
2. Chauhan S, Patel P, Sureka A, Delicato FC, Chaudhary S (2016) Demonstration abstract: Iotsuite—a framework to design, implement, and deploy iot applications. In: 2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), pp 1–2. 10.1109/IPSN.2016.7460669
3. Dave D, Naik H, Singhal S, Patel P (2020) Explainable ai meets healthcare: a study on heart disease dataset
4. Doshi-Velez F, Kim B (2017) Towards a rigorous science of interpretable machine learning
5. Höjer M, Wangel J (2015) Smart sustainable cities: definition and challenges. In: Hilty LM, Aebischer B (eds) ICT innovations for sustainability. Springer International Publishing, Cham, pp 333–349
6. Intizar M, Patel P, Kanti Datta S, Gyrard A (2017) Multi-layer cross domain reasoning over distributed autonomous IoT Applications. Open J Int Things 3. https://hal-emse.ccsd.cnrs.fr/emse-01644333
7. Mehta P, Dwivedi R, Feeney C, Patel P, Ali MI, Breslin J (2021) Towards designing an explainable-ai based solution for livestock mart industry. In: 8th ACM IKDD CODS and 26th COMAD. p. 421. CODS COMAD 2021, Association for Computing Machinery, New York, NY, USA. 10.1145/3430984.3431071

8. Pathak T, Patel V, Kanani S, Arya S, Patel P, Ali MI (2020) A distributed framework to orchestrate video analytics across edge and cloud: a use case of smart doorbell. In: Proceedings of the 10th International Conference on the Internet of Things. IoT '20, Association for Computing Machinery, New York, NY, USA. 10.1145/3410992.3411013
9. Siemens G, Baker RSJD (2012) Learning analytics and educational data mining: towards communication and collaboration. In: Proceedings of the 2nd International Conference on Learning Analytics and Knowledge, pp 252–254. ACM, New York, NY, USA. 10.1145/2330601.2330661
10. Vartak M (2021) From ml models to intelligent applications: the rise of mlops. Proceedings of the VLDB Endowment 14(13):3419. 10.14778/3484224.3484240
11. Zhou Y, Yu Y, Ding B (2020) Towards mlops: a case study of ml pipeline platform. In: 2020 International Conference on Artificial Intelligence and Computer Engineering (ICAICE), pp 494–500 . 10.1109/ICAICE51518.2020.00102

Chapter 9
Security Infrastructure for Cyber Attack Targeted Networks and Services

Anmol Kumar and Gaurav Somani

Abstract Cyber security covers various aspects of confidentiality, integrity, availability, and other related concepts for organizations and individuals. With the increase in web services, Internet enabled devices, and the amount of data generated, we see a huge growth in the number of cyber security incidents across the globe. Denial of service attacks, covert channel attacks, side-channel attacks, malware-driven attacks, and ransomware attacks are some of the most common cyber attack classes with a number of variants. These attacks may show various direct and indirect effects such as data loss, data leakage, data alteration, reputation loss, maintenance cost, malware spread, and service unavailability among others. There is number of levels in the overall target network or service hierarchy which need to have security deployments in the form of devices, software, filtering, and admission control mechanisms. In this chapter, we provide comprehensive detail on various modern security solutions across the overall spectrum of services. We categorize the modern security infrastructure into four categories which include network security, server security, cloud security, and device security infrastructures. We detail various primitives available in each of these categories in the state of the art and discuss their related security aspects.

Keywords Security and protection · Network security · Server security · Cloud security · Device security · Security services · Infrastructure security

A. Kumar · G. Somani (✉)
Department of Computer Science and Engineering, Central University of Rajasthan, Ajmer, India
e-mail: gaurav@curaj.ac.in

A. Kumar
e-mail: anmol_jrf@curaj.ac.in

M. Chaturvedi et al. (eds.), *Recent Advancements in ICT Infrastructure and Applications*, Studies in Infrastructure and Control, https://doi.org/10.1007/978-981-19-2374-6_9

1 Introduction

We are seeing the number of Internet users touching to a whopping 4.66 billion devices in the year 2021 [28]. The amount of data belonging to all Internet users is of the order of a million terabytes. Managing and securing this scale and amount of data is a very challenging task for the current networks. Data leakage and data altercation are some of the very common security threats of the current information edge. We require an advanced security infrastructure and devices to protect and manage the data. There are numerous reports [5, 13, 32] that indicate various types of attacks that were launched to affect the target organizations. Cyber attack reports in [5, 13] discuss a number of DDoS (distributed denial of service) attacks including a 2.3 Tbps peak attack bandwidth in 2020 and a 800 Gbps attack in 2021. In 2021, web-based applications faced a tripled DDoS attacks rate as compared to 2020 and most of these attacks used common API vulnerabilities [32]. SQL injection, local file inclusion, and cross-site scripting are among the most common techniques to target the gaming industries in 2021 [32].

Various attacks such as ransomware attack, spoofing, phishing, identity thefts, denial of service (DoS) attacks, malware-driven attacks, zero-day exploits, and DNS tunneling are used to target the victim organization [12, 21]. These attacks directly or indirectly affect the service availability, data loss, backdoor creation, reputation loss, and high maintenance costs of the victim organization. These attacks are becoming a hurdle for cyber security teams of the target organization to handle them. We see that a lot of innovation is happening in the security technologies in the form of new devices or techniques to combat, mitigate and minimize attack effects at the victim organization.

Various emerging technologies such as cloud services, big data, Internet of things (IoT) devices bring a number of new attack vectors, variants with different scales, and sophistication. Investment-related to cyber security reached to around $2 trillion in year 2021 with a projected growth of 15% on annual basis [8]. We also see a continuous effort from academia, security research industry, communities, and governments to evolve and strengthen the overall cyber security and privacy framework. This is also augmented by the development of next-generation firewalls, cryptographic algorithms, security protocols, software-defined networks (SDN) devices, and network security controls to monitor, face and subvert the cyber security threats and attacks.

In this chapter, we focus on providing an abstract yet comprehensive tutorial to all major developmental directions of modern security infrastructure that help in achieving cyber security aims of an organization. We classify the cyber security infrastructure into four categories such as network security, server security, cloud security, and device security. We organize this chapter as follows: We provide a taxonomy-based classification of modern security infrastructure in Sect. 2. We discuss network security solutions in Sect. 3. We provide details of server security aspects in Sect. 4. We elaborate on cloud security in Sect. 5 and device security in Sect. 6. In Sect. 7, we conclude this chapter.

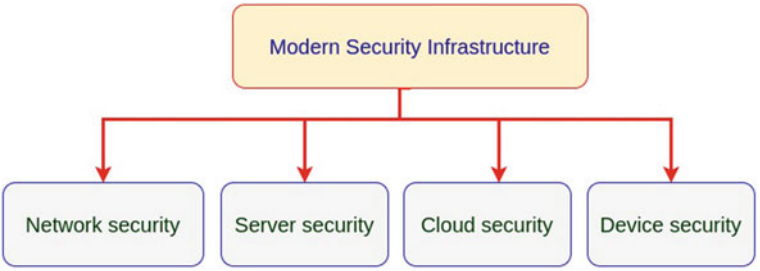


Fig. 1 Four categories of modern security infrastructures

2 Modern Security Infrastructures: A Classification

We see a number of methods, devices, software, and hardware solutions for achieving the cyber security goals of an organization. Many of the techniques or tool just focus on providing a solution to a plethora of security objectives such as network security which involves security at multiple layers. On the other hand, there are solutions which focus on a specific purpose such as storage encryption on embedded devices. For comprehension, we classify modern security infrastructure into four major categories. The four categories include network security, server security, cloud security, and device security infrastructures. These four categories are based on the technique or perimeter used at various levels of a organization. We show the classification of modern security infrastructure in Fig. 1.

3 Network Security Infrastructure

Network security refers to the policies and prevention mechanisms to defend the network services and their assets from unauthorized access to prevent confidentiality, integrity and availability. Network security focuses on providing security at various levels in the network and combining them to prevent the overall infrastructure. Network security helps in providing flawless services to authorized users and prevents access to unauthorized users. Common threats to networks are [35]: distributed denial of service (DDoS), malware, spyware, adware, trojan horse, and computer worms. Access control mechanisms, firewalls, intrusion prevention systems, and event management are basic building blocks of network security [35]. Network security management consists of three different tasks such as detection, response, and protection. A simple architecture of network security is shown in Fig. 2. In Fig. 2, unauthorized accesses get blocked by firewalls. Unauthorized accesses are identified using traffic filtering and network reconfiguration which may be supported by modern devices such as software-defined networking.

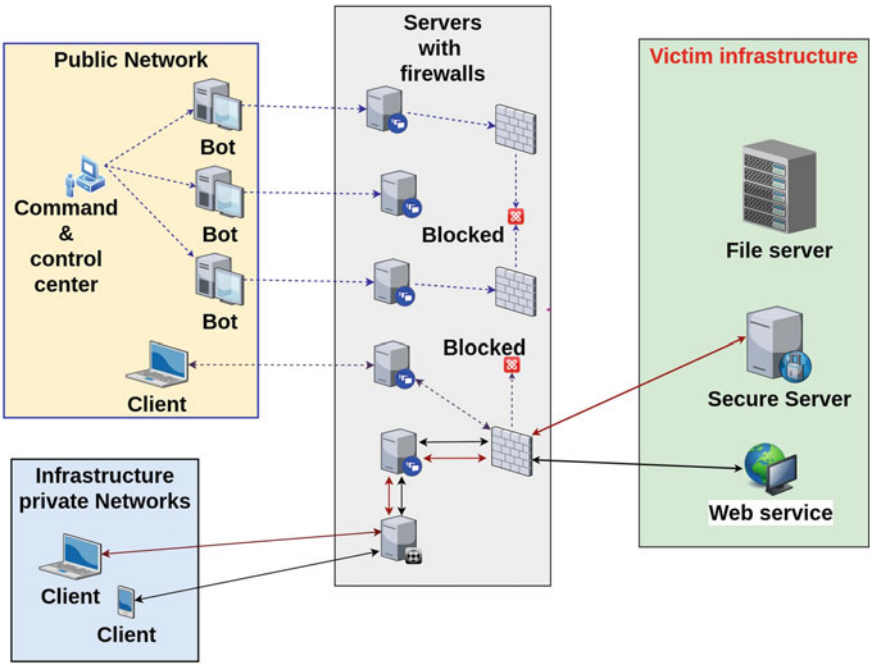


Fig. 2 A scenario showcasing the network security primitives in an organization

There are various solutions available for implementing network security. Common solutions for network security are shown in Fig. 3. Few solutions [10, 11, 35] are: firewall, access control, software-defined network (SDN), intrusion prevention system (IPS), network segmentation, and virtual private network (VPN).

1. **Firewall**
   Firewall is a device that monitors and manages incoming traffic of infrastructure. Firewall prevents unauthorized access of infrastructure. Firewall is considered the first line of defense for any infrastructure. There are different types of firewall[1] such as proxy firewall, stateful inspection firewall, unified threat management (UTM) firewall, and next-generation firewall (NGFW).

   - **Proxy Firewall**
     Proxy firewall is a single-point entry-exit system in the infrastructure. In a proxy firewall, one device (say it as "master node") is directly connected with the Internet and all other devices' requests pass through the master node. Proxy firewall is also called as application firewall/gateway firewall.[2] Proxy firewall manages traffic at the application layer of open systems interconnection (OSI)

[1] https://www.cisco.com/c/en_in/products/security/firewalls/what-is-a-firewall.html.
[2] https://www.fortinet.com/resources/cyberglossary/proxy-firewall.

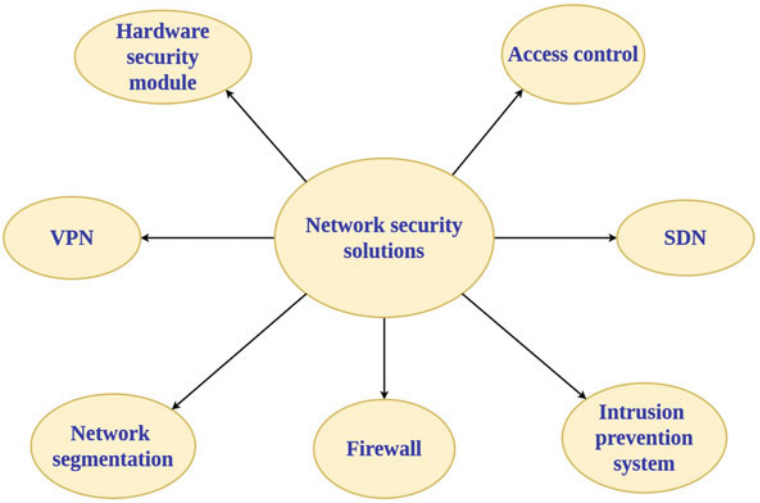


Fig. 3 Solutions for network security

layer by performing various strategies like deep packet inspection (DPI). Proxy firewall suffers from a bottleneck situation in the network since it creates a new connection for every incoming and outgoing packet.[2] Proxy firewalls also face a problem of single-point failure.

- **Stateful inspection firewall**
  inspection firewall[3] is a type of firewall that keeps track of incoming and outgoing requests of a network. Stateful firewall works on the network layer and transport layer of OSI model. In stateful firewall, a profile of "safe" connections is made based upon previous connections. Whenever any new connections are requested, then it is matched with a safe profile, if it is matched then it is allowed otherwise it is discarded.
- **Unified threat management firewall**
  threat management (UTM) firewall consists of various security services such as antivirus, anti-spyware, anti-spamming, and intrusion detection. UTM firewall performs two types of inspection[4]: flow-based inspection and proxy-based inspection. Flow-based inspection inspects for malicious activity and proxy-based inspection inspects the context of the incoming packet. UTM suffers from the classical problem of a single-point failure.
- **Next-generation firewall (NGFW)**
  is a type of firewall that consists of standard firewall services, intrusion prevention/detection mechanisms, and other facilities like application control and

[3] https://www.fortinet.com/resources/cyberglossary/stateful-firewall.

[4] https://www.techtarget.com/searchsecurity/definition/unified-threat-management-UTM.

threat intelligence [2]. NGFW may be operated at the application layer of OSI model.[5]

2. **Access control**
   Access control is a mechanism to protect your data or devices from unauthorized access in the network. Access control specifies about who is allowed to access which data/devices and how they can use it. Access control mechanism consists of various components such as authentication, authorization, access, manage, and audit [1]. Authentication and authorization help in identifying users and assist them to access a resource or a service. Different access control features can be managed by adding or removing certain rules and users are accountable for it during the audit. There are different types of access control mechanisms [1] such as mandatory access control (MAC), attribute-based access control (ABAC), discretionary access control (DAC), role-based access control (RBAC), rule-based access control, and break glass access control. MAC places policies for separate users and resources, data, and devices, different users want to access. DAC allows the resource owner to define access rights for different users [1]. ABAC draws policies for resource attributes such as user attributes (user name, user role, user id, and organization type), environment attributes (access time, organizational threat level, and data location), and resource attributes (creation date, resource owner, resource sensitivity type, and resource name). [6] RBAC draws policies for individual users based upon user's role in the organization. Break-glass based access control creates a privileged account that bypasses all the policies in case of emergency [1].
   Network access control (NAC) helps to restrict unauthorized users to use private network.[7] NAC provides an additional layer of security even after the access of the network. NAC also obstructs those devices from accessing the network which do not satisfy the network security policy of the organization. NAC is of two types : pre-admission and post-admission [37]. Pre-admission NAC helps in assessment of network access attempts and also monitors that only authorized devices should access the network. Post-admission NAC restricts lateral movement and re-authenticates those users who try to access incompatible chunks of the network.
3. **Software defined network (SDN)**
   Software defined network (SDN) is an emerging technology for network reconfiguration, traffic rerouting, network policy implementation, and also to provide an aid to the network security deployments. SDN separates the control and forwarding features of networking devices by using the programming paradigm [24]. SDN helps to collect the information of the network and helps in detection of attacks by analyzing traffic information. SDN controller directs SDN devices to transit the network services wherever needed. Based upon the past historical behavior of

[5] https://www.vmware.com/topics/glossary/content/next-generation-firewall.html.
[6] https://www.dnsstuff.com/rbac-vs-abac-access-control.
[7] https://www.vmware.com/topics/glossary/content/network-access-control.html.

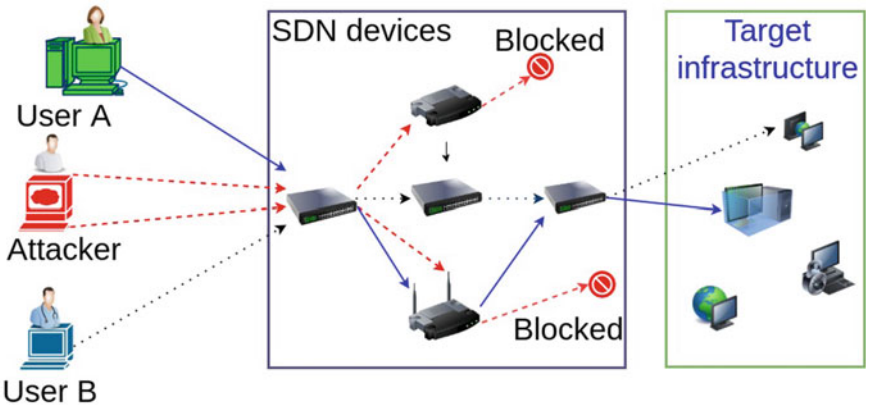


Fig. 4 Role of Software Defined Network (SDN) enabled devices in network security

requests or distinct parameters such as number of requests, arrival time, time to live (TTL) value, hop count, source address, destination address, and a combination of multiple features, some rules may be installed to SDN devices. In case the incoming or outgoing flows meet any of the installed rule, SDN devices may take a specified action against the rule. A simple architecture of SDN enabled network is shown in Fig. 4. In Fig. 4, whenever requests come from users such as attackers or benign users, then SDN devices perform routing, blocking, or forwarding of requests based upon the rules defined inside devices.

SDN helps in network automation, low operational cost, and infrastructure abstraction [3]. SDN also provides flexibility, scalability, and efficiency as compared to traditional networking devices.[8] SDN also faces problems like man-in-the-middle attack, forwarding-device attack, denial of service (DoS) attack, and fake traffic flow [24].

4. **Intrusion prevention system (IPS)**

   Intrusion prevention system (IPS) is a security software that monitors and detects the anomalous behavior of traffic activities in the network. IPS inspects each of the incoming packets with the help of anomaly behavior/rules, malicious traffic signatures, or security policies in the network. In case a match is found then the packet is dropped which prevents the packet from reaching the destination. A simple architecture of an intrusion prevention system is shown in Fig. 5. In Fig. 5, whenever requests arrive at the IPS then it checks the packet with the anomaly database and based upon the result, packet is either discarded, forwarded, or routed to its destination. Most of the modern network-based IPS are implemented in the firewall.

   There are various techniques to perform intrusion prevention such as signature based, anomaly based, and policy based.[9] In signature-based IPS, incoming packet

[8] https://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-SDN.html.

[9] https://www.vmware.com/topics/glossary/content/intrusion-prevention-system.html.

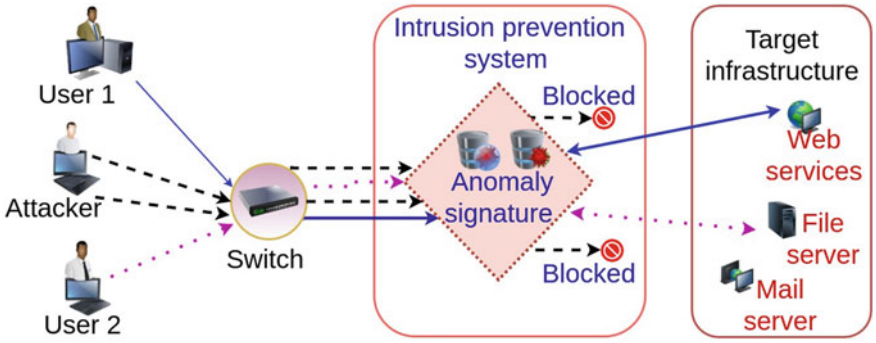


Fig. 5 Intrusion prevention systems for network security

is matched with a list of signatures of known attacks. Signature-based IPS does not work for zero-day attacks or for those attacks whose signature is not present in the list. Anomaly-based IPS checks random samples of traffic in the network and compares them with baseline standard. Anomaly-based IPS may generate high false positive alarms. Policy-based IPS checks incoming packets of the network with security policies and takes preventive measures against the violation of rules. There are different types of intrusion prevention system such as network intrusion prevention system (NIPS), host intrusion prevention system (HIPS), network behavior analysis (NBA), and wireless intrusion prevention system (WIPS).[9] NIPS scans the network traffic and scans for dubious traffic by inspecting protocol activity. WIPS scans wireless networks for dubious traffic/unauthorized access/unauthorized devices connected to the network. HIPS is similar to NIPS, there is a small difference in that HIPS is installed on the host machine and monitors the traffic of the host machine. NBA monitors network traffic and inspects for threats such as DDoS attacks that generate unusual traffic flow.

5. **Network segmentation**
   One of the emerging ideas to protect highly confidential data from unauthorized users is network segmentation. Network segmentation divides the entire network into sub-parts to manage the network smoothly.[10] Network segmentation can be achieved by physical segmentation or virtual segmentation [42]. In physical network segmentation, each sub-net is connected with dedicated cables. In virtual network segmentation, routing devices manage the entire virtual network. One of the practical implementations of network segmentation is a demilitarized zone (DMZ). DMZ is a network segmentation technique which protects prime and indispensable information of the victim organization from unauthorized access.[11] A simple architecture of DMZ in the target organizational network is shown in Fig. 6.

[10] https://www.cisco.com/c/en_in/products/security/what-is-network-segmentation.html.

[11] https://www.fortinet.com/resources/cyberglossary/what-is-dmz.

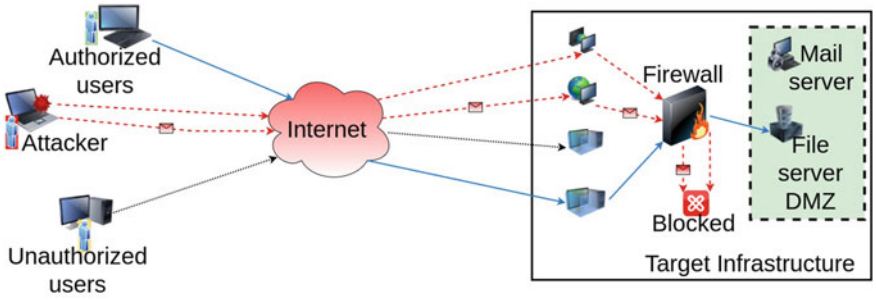


Fig. 6 Demilitarized Zone for network security

In Fig. 6, target organizations kept their essential data and devices such as file server and mail server in the DMZ zone where only authorized requests can access those data or devices. DMZ is protected by either single firewall or two firewalls to prevent unauthorized access. DMZ controls and reduces the access control on the sensitive data of the organization.[11] Network segmentation reduces the damage caused by attack, protects vulnerable devices, reduces compliance scopes, and improves network performance.[10]

6. **Virtual private network (VPN)**
   Virtual private network (VPN) helps to set up a secure network connection while using public network [14]. VPN uses encryption technique to set up a secure connection. VPN supports various features like secure connectivity, distributed network, network scalability, and access control [9, 22]. A good VPN should ensure security features like confidentiality, integrity, and authentication. A simple architectural view of VPN is shown in Fig. 7. In Fig. 7, when an authorized user tries to communicate in the network then receivers may use a private network to transmit the data. Transmitted data may be encrypted using asymmetric encryption techniques.
   There are various types of VPN such as site-to-site VPN, remote access VPN, and VPN as service (VPNaaS) [9]. Site-to-site VPN is also known as router-to-router VPN. Site-to-site VPN ensures secure communication between different parts of the network. Remote access VPN allows access to authorized users to connect from outside the network. VPNaaS is also known as cloud VPN. VPNaaS is an emerging technology where authorized users connect with network appliances or data without having a VPN infrastructure at the user end. VPN suffers from different problems such as poor scalability, endpoint vulnerabilities, fragment visibility, and inefficient routing [9].
7. **Hardware security module**
   Hardware security module (HSM) is a trained and highly reliable device to provide high security to sensitive data of websites, banks, payments, smart meters, crypto-currencies, and digital documents [18]. HSM uses various security techniques such as encryption, key management, and key exchange to secure the network. HSM devices have vigorous operating system and restricted network access which

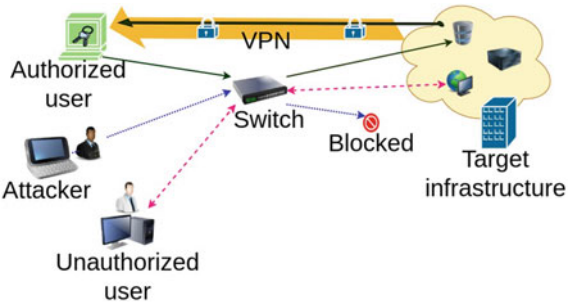


Fig. 7 Virtual Private Network (VPN) connections

helps in preventing unauthorized access [17, 18]. HSM helps in preserving the life-cycle of cryptography keys. HSM follows six steps to manage keys such as provisioning, storage, deployment, management, archiving, and destruction [17]. Provisioning helps in creating random keys in HSM. A copy of generated keys need to be securely stored by HSM and these keys need to be deployed on the cryptographic devices. Generated keys need to be monitored and it should follow the organizational policies and standards. Generated keys need to be archived for future usages. Once there is no need for a generated key, it needs to be securely disposed of. HSM helps in securing design, APIs, and operating system, tamper resistance, isolation, and access controls [17].

4 Server Security Infrastructures

In this section, we detail the security issues, threats, and solutions for securing a server infrastructure. A server provides services to different programs or requests given by different clients (users). There are different types of servers such as file server, web server, email server, online game server, database server, proxy server, and application server [4, 41]. A server has access to a huge amount of data and this data may be sensitive and confidential, therefore, we need to protect their confidentiality, integrity, and availability. A simple attack scenario for servers is shown in Fig. 8. Attackers try to target various servers by sending huge requests to them to make the service unavailable, perform data leakage, or affect the data integrity. There are numerous modern security devices and techniques such as load balancers, antivirus software, encryption, backups, antivirus, client side checking, Input/output validation and sanitization, and fixing of common vulnerabilities and exposures (CVE)/ common weakness enumeration (CWE). In addition, other equally important needs include privilege escalation prevention and use of secure coding, secure APIs and crypto to prevent the servers from various attacks.

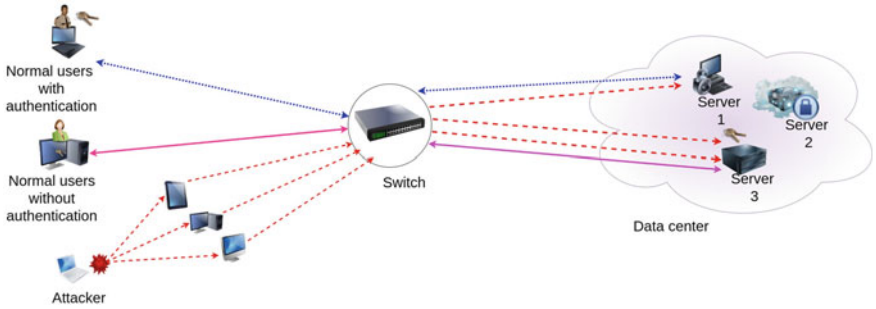


Fig. 8 Attack scenario in data center having multiple servers

1. **Load balancer**
   A load balancer also plays a significant role in minimizing the effects of attacks on service security. A load balancer acts as an obstruction between users requests and the target server. Load balancer distributes incoming requests to different servers to achieve high availability, cost effectiveness, flexibility, and scalability.[12] There are two types of load balancers such as hardware-based load balancer and software-based load balancer.[13] Hardware-based load balancer is a physical device with specialized software to forward requests. Hardware-based load balancers have flexible multi-tenant architecture, strong isolation, high performance, and are capable of processing huge number of requests. Software-based load balancer works similar to hardware-based load balancer and there is a slight difference that software-based load balancer is directly installed on a server machine. Various software-based load balancers are HAProxy, NGINX, mod_athena, Varnish, and LVS.
2. **Antivirus software**
   Antivirus is the most common tool to protect from malicious programs in servers. Antivirus scans the servers based upon certain parameters such as signature, heuristics, file length, and checksum [27]. In signature-based antivirus, antivirus compares files with a database of malware signatures and if it matched, then the file is quarantined. In file length-based antivirus, antivirus compares suspicious file length with actual file length of the file. If the file length differs then it is an infected file. In checksum-based antivirus, checksum value is calculated for a suspicious file and matched with the checksum of the file when it was not affected. If the checksum value differs then it confirms that the file is infected.
3. **Encryption**
   Encryption helps in protecting data from various attacks such as data leakage, unauthorized access, and data alteration. Encryption and obfuscation allows transforming plain data into encrypted data which makes the encrypted data unusable

[12] https://www.citrix.com/en-in/solutions/app-delivery-and-security/load-balancing/what-is-load-balancing.html.

[13] https://www.vmware.com/topics/glossary/content/software-load-balancing.html.

for the attacker even if a security breach occurs. Various software tools that help in performing encryption to servers are FujiFilm Object Archive, Data Masque, Sophos Firewall, and Barracuda Backup [38].

4. **Input/Output sanitization**
   A very effective technique to save from most of the security vulnerabilities and possible attacks is having a user input validation and sanitization mechanism in the server side code and client side code running in the browser.[14] Attackers try to pass administrator permissible codes in input fields to gain access to the system default configuration and try to take the control over the server. Buffer overrun, OS command injection, and SQL injection are most popular attacks mostly launched with the help of the user input.[14] Validation is performed to achieve the conformity of the input with some ranges and allowed inputs with the help of a whitelist.[14] On the other hand, sanitization helps in removing any unwanted input/characters which may assist an attacker to introduce the vulnerabilities.
5. **Use of secure coding**
   Many of the server vulnerabilities occur due to the poor code, code-reuse, and usages of unsafe functions. Secure coding guidelines advise programmers to focus on security as an important need of the developed application instead of treating it as a optional feature. There is number of secure coding guidelines available which are also motivated by the common vulnerabilities. Secure coding standards are a set of rules or guidelines to design or develop the application. Secure coding helps in preventing servers from attacks as it removes vulnerabilities from applications on servers. Secure coding[15] involves input validation, using safe cryptographic algorithms, safe functions, error handling and logging, data protection, and communication security.

A architecture of server security is shown in Fig. 9. Private network is used for secure communication between server (server 1) and authorized users. Secure and encrypted communication is used between server (server 2) and normal users without authentication. By this, data integrity can be maintained and logging and tracing can be done.

There are many threats to server security such as open and unused ports of server, services without timely patches, insecure software/application, insecure services, and poor administration.[16] Common attacks that are targeting the server are DDoS attacks, DNS tunneling, side-channel attacks, directory traversal, phishing attacks, and SQL injection attacks. Many of the server vulnerabilities are listed as common vulnerabilities and weaknesses. Some of the important vulnerabilities for client-server model include CWE-200 (exposure of sensitive information to an unauthorized actor), CWE-388 (use of cryptographically weak Pseudo-Random Number

[14] https://www.esecurityplanet.com/endpoint/prevent-web-attacks-using-input-sanitization/.

[15] https://www.whitehatsec.com/glossary/content/secure-coding.

[16] https://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-sg-en-4/s1-risk-serv.html.

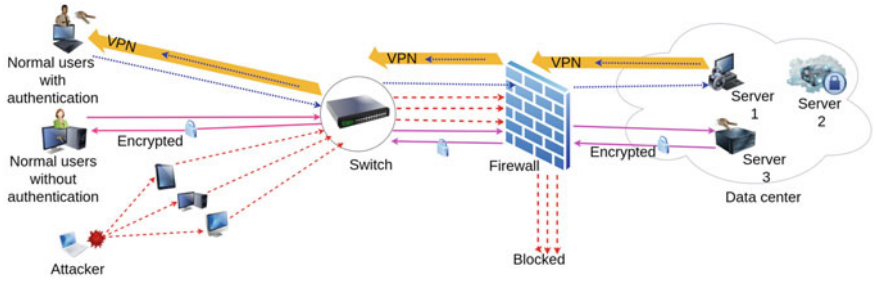


Fig. 9 Server security using different mechanisms

Generator (PRNG)), CWE-500 (exposure of application data that may be modified), and CWE-1065 (runtime resource management control element to run on application servers).[17]

5 Cloud Security Infrastructures

Cloud computing is a paradigm of on-demand services and resources based on the pay-as-you-go model. These services may include web services, database services, storage services, and security services. Resources range from servers to storage devices. Cloud computing offers various features such as cost-effectiveness, scalability, resource pooling, security, and reliability [6, 7]. Based on the offered services, cloud computing is categorized into three categories, infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS).[18] IaaS is a cloud-computing technique in which a cloud service provider (CSP) manages computational and storage servers using virtualization. In PaaS, CSP manages all the platform services of IaaS like OS, runtime environment, and middleware. In SaaS, CSP offers complete web-service instances with complete business logic and data management.

As cloud computing offers various features, it attracts numerous customers to switch from traditional hosting-based infrastructure or on-site infrastructure to a cloud-based infrastructure. The growth in the cloud-based hosting also leads to an increase in the attackers shifting their target to clouds. Cloud security is a collection of security measures and techniques to address cloud-based security threats and combat possible attacks. Attacks such as DDoS attacks, side-channel attacks, covert channel attacks, and hyperjacking are common on cloud targets. In addition, most of the traditional attacks are also applicable to the cloud environments. We show a typical scenario showing different attacks on target cloud infrastructure in Fig. 10.

[17] https://cwe.mitre.org/data/published/cwe_latest.pdf.

[18] https://www.redhat.com/en/topics/cloud-computing/public-cloud-vs-private-cloud-and-hybrid-cloud.

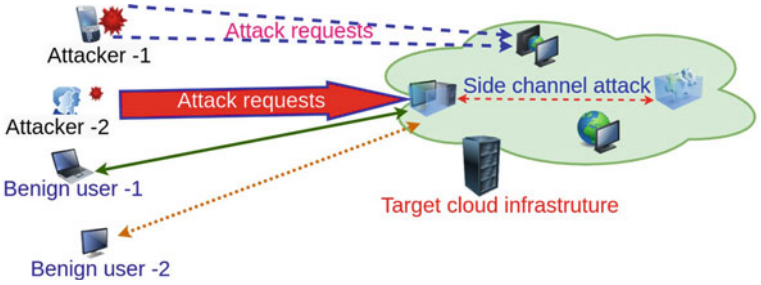


Fig. 10 Cyber attacks on cloud-based service

In Fig. 10, different attackers launch distinct attacks on target virtual machines or web services in the cloud to make service unavailable for genuine users in the cloud infrastructure. Sometimes, an attacker places its malicious VMs with the other VMs in the cloud infrastructure to launch attacks such as covert channel attacks, side-channel attacks, and resource contention for the victim VMs. There are numerous solutions available in the state of the art to defend or minimize attack effects on target organization.

1. **Security solutions in IaaS**[19]
   There are numerous solutions available for securing the IaaS in cloud environment such as monitoring and logging, virtual network security platforms, cloud access security broker, cloud workload protection platforms, and cloud security poster management.

   - **Use of cloud access security broker (CASB)**
     CASB is a cloud-based software that acts as mediator between cloud service provider (CSP) and clients. CASB is based upon four security factors such as data protection, threat protection, visibility, and identity [31]. Tools for implementing CASB are Bitglass and Cisco Systems Cloudlock [20, 31].
   - **Use of cloud workload protection platforms (CWPP)**
     CWPP helps in securing assets such as containers and VMs of cloud infrastructure. CWPP bridges a gap between CSP and legacy components of the cloud. CWPP is suitable for those organizational units situated at different geographic locations. Illumio Core and Orca Security are useful tools for the implementation of CWPP [31].
   - **Use of virtual network security platforms (VNSP)**
     VNSP scans incoming and outgoing traffic of virtual instances in the cloud environment. VNSP performs intrusion detection and prevention system (IDPS) to protect resources of virtual instances in cloud environments.[19]
   - **Cloud security posture management (CSPM)**
     CSPM helps in auditing security issues of IaaS in cloud environments.[19] CSPM also helps in automating security management. CSPM tools operate

[19] https://www.mcafee.com/enterprise/en-in/security-awareness/cloud/what-is-iaas.html.

by continuously looking for misconfigurations and making changes automatically, if required. Tools for performing CSPM are Fugue and XM Cyber.

2. **Security solutions in PaaS**.[20]
Various security issues of PaaS in cloud environment can be solved by using following techniques such as:

   - **Implement role-based access controls**
   Role-based access management helps to ensure which users can access what resources and tools.[20] Role-based access control helps in protecting sensitive data leakage and helps in achieving security features like confidentiality, integrity, and availability.
   - **Check for inherited software vulnerabilities**
   Third-party applications/software have multiple vulnerabilities and uses of these applications in PaaS cloud environment helps attackers to create a backdoor to access the target machines. Inherited software vulnerabilities increase the risk factor for PaaS environment owing to the lack of security awareness in developers.
   - **Use threat modeling**
   Many of the security issues may be identified during the early stage of application/software development. These issues may be identified and fixed by using the threat modeling tools. A threat model includes a description of the subject, potential threats of the subject, mitigation techniques for listed threats, and ways to validate threats [16].

3. **Security solutions in SaaS**.[21]
Various security solutions for SaaS in cloud infrastructure also includes various tools such as CASB which also assists security solutions for IaaS in cloud. Some more solutions are as follows:

   - **Advanced malware prevention**
   Using the real-time threat intelligence tools can help to protect against zero-day attacks and spreading of malicious files in the cloud.[21]
   - **Data loss prevention (DLP)**
   DLP helps to safeguard the sensitive data in the cloud. DLP helps in detection and prevention of data breaches and unwanted demolition of sensitive data in the cloud. Symantec DLP, Teramind DLP, SecureTrust Data Loss Prevention, and Code42 are few tools to implement the DLP [15].
   - **Fear of insider attacks**
   Insider attacks are the most dangerous attack for any organization. Monitoring and logging activities, role-based access management techniques, secure backups and regular auditing may help in preventing insider threat.
   - **Security configuration audits**
   Regular auditing of security policies/mechanisms may help in identifying

[20] https://www.mcafee.com/enterprise/en-in/security-awareness/cloud/what-is-paas.html.

[21] https://www.mcafee.com/enterprise/en-in/security-awareness/cloud/what-is-saas.html.

holes in the cloud as well as in the hosts. Security auditing also advises guidelines/preventive measures to protect the host against zero-days attacks.

There are numerous threats to cloud infrastructure [25, 33, 39]. In IaaS cloud environment, threats for infrastructure are larceny of data hosted, insider attacks, application vulnerabilities, and inconsistent security controls above multi-cloud environments. For PaaS cloud environment, security threats are permission to tenants, weak authentication mechanisms, account hijacking, and default application configurations. Security threats in SaaS cloud environment are visibility issues, data theft, data misuse, and less control over sensitive data and its management. Most such attacks took advantage of common vulnerabilities to target the victim machines in the cloud. Common vulnerabilities in the cloud environment are CVE-2012-3446 (incorrect regular expression for matching hostname), CVE-2019-4521 (allows arbitrary command execution via CSV injection), CVE-2012-5819 (host invalidation for cloud storage management application), and CVE-2018-9057 (uses a non-cryptographically secure PRNG).[17]

6 Device Security Infrastructures

We have a variety of computing, storage, and network devices such as firewalls, servers, IPS, load balancers, switches, and routers. In addition, we have a number of personal devices such as mobile phones, IoT devices, home modems, embedded devices for smart homes, and personal digital assistants (PDAs). In this section, we discuss the security primitives of such devices as shown in Fig. 11. These primitives include physical security, malware detection, privacy, and communication security.

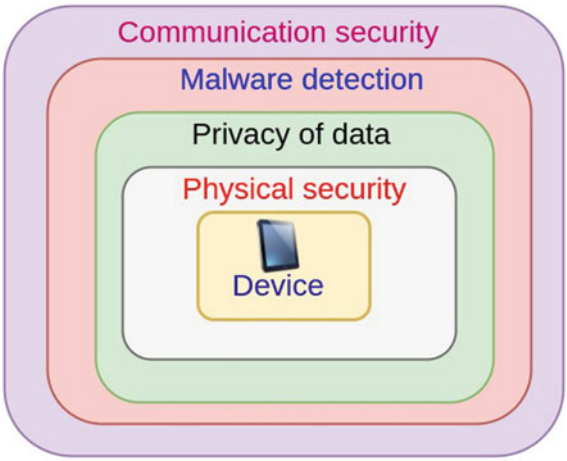


Fig. 11 Device security infrastructure

1. **Physical Security**
   Protecting sensitive data is an important concern for modern cyber infrastructure. We use various cryptographic modules such as single chip, multiple chip, and multiple chip standalone cryptographic modules to protect these sensitive data [34]. However, protecting these cryptographic modules is yet another challenging task. To detect the suspicious activities with cryptographic modules, we may use various physical security mechanisms such as tamper-evidences (like coating, sealing, or locks), role-based authentication, strong enclosures, environmental failure protection (EFP), environmental failure testing (EFT), and tamper detection/response circuitry [34]. Cryptographic modules should provide evidence collection for tampering which may indicate suspicious activities performed on the modules.
   During the physical maintenance phase, all sensitive data should be zeroized [34]. The tamper response and zeroization circuitry should remain operational during physical maintenance. Cryptographic modules should also contain continuous monitoring of tamper response and zeroization circuitry to prevent the sensitive data [34].
2. **Privacy**
   Various devices such as smart home devices or mobile devices may use third-party applications for performing certain tasks that require transfer of sensitive data which becomes an important concern to maintain the privacy, integrity, and confidentiality of sensitive data. Preserving privacy required secure connections for communication, use of private networks, auditing the logging activities, avoiding use of unnecessary software, and use of IDS [36]. Modern infrastructure like blockchain helps in preserving privacy as blockchain uses asymmetric cryptographic techniques to perform the transactions for users. There are numerous techniques available in blockchain to preserve the privacy such as shuffling technology, zero-knowledge proof, group signatures, and ring signature [40].
3. **Communication Security**
   As the mobile users are increasing day by day, securing their communication is becoming a major concern for cyber security team of the organization. As mobile communication has evolved from one generation to another, security mechanisms also became advanced. 4G cellular network offers various security techniques such as network access security, network domain security, user domain security, application domain security, and visibility and configuration of security to mobile users for smooth and secure communication. 4G suffers from bidirectional authentications, IP address spoofing, user ID theft, Theft of Service, Denial of Service, and intrusion attacks.
   These challenges play a key role for the evolution of 5G cellular network. 5G is still in its evolution but it supports various security features like mutual authentication capabilities, enhanced subscriber identity protection, Security Edge Protection Proxy (SEPP), Authentication Key Agreement (AKA), and privacy and integrity cipher for secure communication [23]. 5G also supports various features such as distributed clouds and edge computing, NFVi (network functions virtualization infrastructure), and appliance-based functions [19].

4. **Malware detection**
   Most of the mobile platforms and IoT devices are vulnerable to malware-driven attacks. As the number of mobile users increases, mobile-based services become popular and risk for these services become high. There are numerous malware attacks launched on android devices that are categorized into various categories such as banking malware, mobile ransomware, mobile spyware, MMS malware, and mobile adware [30]. To protect from android malware, we have numerous antivirus software such as avast mobile security, Malwarebytes, VIPRE android security, nox security, firefox focus, and safe security [26].

7 Discussion

As the number of users, devices, and services on the Internet increase, security risks, and their sophistication are also on the continued rise. Data confidentiality and integrity becomes a major concern for the users as well as for various organizations with the rise. Data breaches, loss of data, and data alteration are threats for network security, server security, cloud security, and mobile security. Service availability is another major concern for cloud security and server security. These confidentiality, integrity, and availability characteristics help the security community to see security at different perimeter levels. As the modern technologies like cloud, IoT, and android devices have widespread usage across the globe, cyber security risks are also on the rise to the users as well as cyber infrastructure.

There are different mechanisms to prevent the cyber infrastructure for different attacks such as security policies, secure coding, antivirus, modern security devices, and load balancer but these all need costs. Cost is an important factor while choosing the correct defense mechanisms. Various policies can be written but its implementation in the right way is a major concern. We need to look closely whether security policies are implemented in prescribed manner. Implementation of security policies suffers from insider attacks as well as implementation cost. To deal with insider attacks, we need to perform regular auditing of logging activities, role-based access control management, and use of multi-factor authentication mechanisms to access sensitive data. We also need to regularly update the software and devices so that the identified vulnerabilities should not be taken as a privilege to the attackers.

In a cloud environment, possibility of insider threats and data leakage is a major concern for both customers as well as for CSP. To prevent insider attacks, CSP should ensure that various policies such as password management policy, account management policy, and third-party management policy should be followed in proper manner and its auditing is done on a regular basis. CSP should also ensure that modern security devices such as intrusion detection and prevention system, antivirus, load balancer, and firewalls should be used in their data center. CSP should also ensure that physical activities in data centers should be recorded in cameras, registers, and databases, and deployment of persons to guard the data center. Attacks such as zero-day attacks may bypass the existing security infrastructure, therefore, we need to

identify their pattern at an early stage and need to build more sophisticated alert mechanisms, security devices, techniques, and policies to prevent zero-day attacks. In the cloud environment, the role of CSP (cloud service providers) becomes very important to preserve the security features like confidentiality, availability, and integrity of the data in a multi-tenant cloud environment. CSPs need to decide which security policies, guidelines, techniques and devices are important to protect their data center from attacks. CSP has to make a trade-off between the cost and security for their data centers with a focus on shared responsibility with tenants. CSP should also take proper steps to prevent insider attacks and cloud-originated attacks.

8 Conclusion

Cyber security has a number of directions and levels which needs to be addressed by the victim organizations from time to time. In this chapter, we discuss four important security categories which address the security aspects at different levels. Network security, server security, cloud security, and device security are different classes of security infrastructures. We discuss each of these four categories in detail and provide different security primitives, devices, and techniques available in the modern state-of-the-art from the practical perspective. We also discuss a number of important related issues such as role of the CSP, insider attacks, zero-day vulnerabilities, and the role of security policies at the end of this chapter.

Acknowledgements This research work is partially supported by the Science and Engineering Research Board (SERB), Department of Science and Technology (DST), Government of India with the help of a Core Research Grant number CRG/2020/005759.

References

1. Fortinet (2022) Access control. https://www.fortinet.com/resources/cyberglossary/access-control. Accessed 19 Jan 2022
2. Gartner, Inc.'s (2022) About next generation firewall. https://www.gartner.com/en/information-technology/glossary/next-generation-firewalls-ngfws. Accessed 18 Jan 2022
3. Amit Puri, Manjari Sharma, Ramesh Padmanabhan (2022) Benefits of SDN. https://www.wipro.com/infrastructure/sdn-adoption-in-enterprises/. Accessed 22 Jan 2022
4. ApacheBooster (2019) Server security issues. https://apachebooster.com/blog/what-is-server-security-what-are-the-measures-to-protect-servers/. Accessed 15 Jan 2022
5. Diego Asturias (2021) DDoS attack statistics. https://www.cloudbric.com/blog/2021/04/most-notorious-ddos-attacks-in-history-2021-update/. Accessed 24 Jan 2022
6. Azure (2022) Cloud computing. https://azure.microsoft.com/en-in/overview/what-is-cloud-computing/. Accessed 30 Jan 2022
7. Box (2021) Cloud security. https://www.box.com/en-in/resources/what-is-cloud-security. Accessed 16 Jan 2022

8. David Braue (2021) Investment on cyber security. https://cybersecurityventures.com/cybersecurity-spending-2021-2025/. Accessed 24 Jan 2022
9. Checkpoint (2022) Type of virtual private nework. https://www.checkpoint.com/cyber-hub/network-security/what-is-vpn/. Accessed 19 Jan 2022
10. Archana Choudary (2021) Type of network security. https://www.edureka.co/blog/what-is-network-security/. Accessed 15 Jan 2022
11. Cisco (2021) Type of network security. https://www.cisco.com/c/en_in/products/security/what-is-network-security.html. Accessed 15 Jan 2022
12. Cisco (2022) Type of cyber attacks. https://www.cisco.com/c/en_in/products/security/common-cyberattacks.html. Accessed 24 Jan 2022
13. Cloudfare (2021) DDoS attack statistics. https://www.cloudfare.com/en-in/learning/ddos/famous-ddos-attacks/. Accessed 24 Jan 2022
14. Cloudflare (2022) Virtual private nework. https://www.cloudflare.com/en-in/learning/access-management/vpn-security/. Accessed 19 Jan 2022
15. STEPHEN COOPER (2022) DLP tools. https://www.comparitech.com/data-privacy-management/data-loss-prevention-tools-software/. Accessed 9 Feb 2022
16. Victoria Drake (2022) Threat modeling. https://owasp.org/www-community/Threat_Modeling. Accessed 8 Feb 2022
17. Elizabeth Davies (2022) Cameron McKenzie. Hardware security module. https://www.techtarget.com/searchsecurity/definition/hardware-security-module-HSM. Accessed 29 Jan 2022
18. Encryption Consulting (2022) About hardware security module. https://www.encryptionconsulting.com/education-center/what-is-an-hsm/. Accessed 29 Jan 2022
19. Ericsson (2022) 5G security features. https://www.ericsson.com/en/security/a-guide-to-5g-network-security. Accessed 9 Feb 2022
20. Eyal Katz (2022) Cloud security tools. https://spectralops.io/blog/top-12-cloud-security-tools/. Accessed 31 Jan 2022
21. FBI (2022) Type of threats in cyber world. https://www.fbi.gov/investigate/cyber. Accessed 24 Jan 2022
22. Fortinet (2022) Benefits of virtual private nework. https://www.fortinet.com/resources/cyberglossary/benefits-of-vpn. Accessed 19 Jan 2022
23. GSMA (2022) 5G security. https://www.gsma.com/security/securing-the-5g-era/. Accessed 9 Feb 2022
24. Iqbal Maham, Iqbal Farwa, Mohsin Fatima, Rizwan Muhammad, Ahmad Fahad (2019) Security issues in software defined networking (sdn): risks, challenges and potential solutions. Int J Adv Comput Sci Appl 10(10):298–303
25. Walter Isharufe, Fehmi Jaafar, Sergey Butakov (2020) Study of security issues in platform-as-a-service (paas) cloud model. In *2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, pp 1–6
26. Kanishk Jain (2022) Android antivirus. https://geekflare.com/android-security-apps/. Accessed 9 Feb 2022
27. Jeff Melnick (2022) Device security. https://blog.netwrix.com/2019/01/22/network-security-devices-you-need-to-know-about/. Accessed 1 Feb 2022
28. Joseph Johnson (2022) digital population worldwide. https://www.statista.com/statistics/617136/digital-population-worldwide/. Accessed 9 Feb 2022
29. Justin Ellingwood, Lisa Tagliaferri, Jamon Camisso, Dbrian (2020) Security measures to protect servers. https://www.digitalocean.com/community/tutorials/recommended-security-measures-to-protect-your-servers. Accessed 15 Jan 2022
30. Kaspersky (2022) Android malware. https://www.kaspersky.co.in/resource-center/threats/mobile. Accessed 9 Feb 2022
31. LIKU ZELLEKE (2022) Cloud security tools-2. https://www.comparitech.com/net-admin/cloud-security-tools/. Accessed 31 Jan 2022
32. Martin McKeay, Steve Ragan, Amanda Goedde, Chelsea Tuttle (2022) Gaming in a Pandemic. https://www.akamai.com/content/dam/site/en/documents/state-of-the-internet/akamai-state-of-the-internet-gaming-in-a-pandemic.pdf. Accessed 24 Jan 2022

33. Mcafee (2022) Cloud security issues. https://www.mcafee.com/enterprise/en-in/security-awareness/cloud/security-issues-in-cloud-computing.html. Accessed 16 Jan 2022
34. National Institute of Standards and Technology (2001) Threat modeling. https://csrc.nist.gov/publications/detail/fips/140/2/final. Accessed 9 Feb 2022
35. Paloalto Networks (2021) Threats prevention. https://www.paloaltonetworks.com/cyberpedia/what-is-network-security. Accessed 15 Jan 2022
36. Phoenixnap (2019) Privacy preserving techniques. https://phoenixnap.com/kb/server-security-tips. Accessed 9 Feb 2022
37. Rahul Awati (2022) Tpe of network access control. https://www.techtarget.com/searchnetworking/definition/network-access-control. Accessed 19 Jan 2022
38. Drew Robb (2021) Server security tools. https://www.serverwatch.com/servers/server-security-tools/. Accessed 15 Jan 2022
39. Secureworks (2021) Cloud security threats. https://www.secureworks.com/blog/cloud-security-guide-to-platforms-threats-solutions. Accessed 16 Jan 2022
40. Fei Tang, Zhuo Feng, Qianhong Gong, Yonghong Huang, Dong Huang (2021) Privacy-preserving scheme in the blockchain based on group signature with multiple managers. *Security and Communication Networks*
41. Techopedia (2022) Type of server. https://www.techopedia.com/definition/23735/server-software. Accessed 25 Jan 2022
42. Andreja Velimirovic (2021) Type of network segmentation. https://phoenixnap.com/blog/network-segmentation-security. Accessed 25 Jan 2022

Chapter 10
Case Studies: Cancelable Biometric-Based Secure Systems

Rudresh Dwivedi and Somnath Dey

Abstract Security theft and privacy invasion are two passive issues that still persist in the effective deployment of biometric-based authentication systems. Compromise of biometric data can potentially lead to serious security violation as the user's biometric trait cannot be changed. In order to prevent the invasion of biometric templates, it is desired to morph the original biometric template through non-invertible or irreversible transformation function. This transformed template is referred to as cancelable template and can be replaced or reissued in case of compromise. The problem still persists if a protected multi-biometric template gets compromised. Objective of this book chapter is to address the mentioned concerns associated with template protection and investigate the template protection schemes for unimodal and multimodal biometric traits with large-scale biometric data so that the matching can be accomplished in transformed domain without compromising the verification performance. In this chapter, we present a discussion on efficient template protection scheme for unimodal and multimodal biometric authentication systems. Further, we also present the security models for few case studies on biometric systems, which attain performance improvement and provide adequate security to protect original biometric data.

Keywords Biometrics · Cancelable biometrics · Secure authentication · Access control

This publication has emanated from research supported by grants from the SERB (ECR/2017/000027), Department of science & Technology, Govt. of India. The authors are also thankful to Indian Institute of Technology Indore for providing the laboratory support and research facilities to carry out this research.

R. Dwivedi (✉)
Department of Computer Science and Engineering, Netaji Subhas University of Technology (NSUT), Delhi, India
e-mail: rudresh.dwivedi@nsut.ac.in

S. Dey
Discipline of Computer Science and Engineering, Indian Institute of Technology Indore, Indore, India
e-mail: somnathd@iiti.ac.in

M. Chaturvedi et al. (eds.), *Recent Advancements in ICT Infrastructure and Applications*, Studies in Infrastructure and Control, https://doi.org/10.1007/978-981-19-2374-6_10

1 Introduction

Today, majority of the firms are cognizant about the importance of emulous schemes and leading ICT solutions that assist in authenticating a user in a secured way. Biometric template protection is a notion coined for automated recognition of a user based on his/her physiological traits such as iris, fingerprints, vein patterns, signature, and voice [13]. For ICT infrastructure, it could play a notable role in sophisticated applications such as land border control [1, 2], military cantonment, and protecting banking assets thereby providing a secure environment [9]. However, there are few unresolved concerns in postulating biometric authentication for these subtle areas. Different investigations on security processes and strategic concepts have been carried out to refine the security model employed to cater firm needs [3]. In turn, it aids to introduce security models for a viable and secure biometric authentication-based military and banking environment.

1.1 Background

At present, majority of banks are targeting virtual/online banking applications to meet the market's competitive standards. However, such system provides ample facilities; they are exposed to common cyber risks and threats. The said risks and threats include worms, viruses, denial-of-service, information phishing, malware [13], etc. Recent conclusive facts and reviews indicate that most of the banks are affected by security threats and privacy invasions caused due to illegitimate access by internal employees and few external customers. A glimpse of security and privacy invasion concerns are illustrated in Fig. 1.

For these underlying concerns, there is a necessity to design a biometric system with substantial template protection scheme [13]. Generally, the following axioms are followed to ensure adequate secrecy and privacy [12, 13]:

1. Non-invertibility/irreversibility: It should be sufficiently infeasible to retrieve original template from the protected template or the helper data.
2. Revocability: In the situation of compromise, a new template should be reissued to replace the compromised one.
3. Diversity: The transformation should be able to derive numerous secure template, and those secure templates should be uncorrelated/unlinkable to each other.
4. Performance: The performance with respect to unprotected biometric system should be preserved, i.e. there should not be larger performance degradation compared to unprotected biometric system.

To avoid these threats, biometric-based mechanisms have become an effective way for user identification. However, considerable security concerns are associated with the implementation of such schemes. Moreover, frequent changes in template

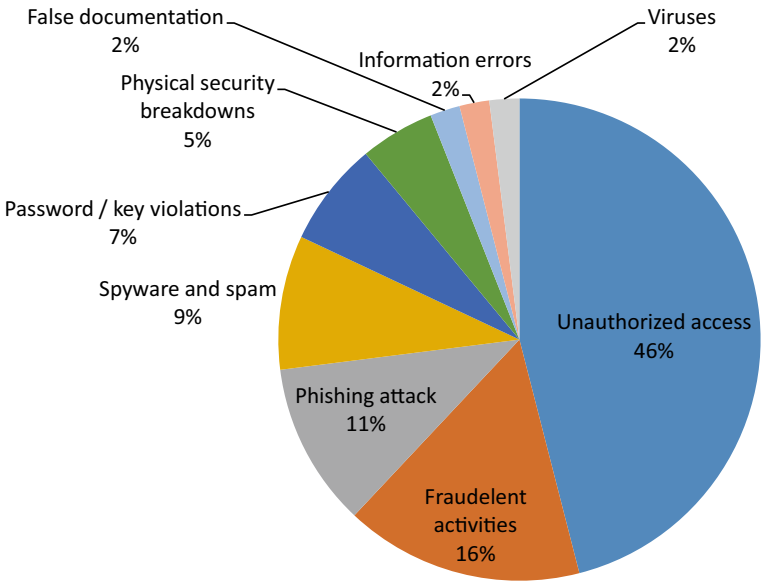


Fig. 1 Security and privacy invasion incidents occurred in past decades

protection standards (ISO-IEC) [4, 12] and privacy policies [17] over the last few years made it challenging. It results in numerous techno-economic factors for the banking firms to adopt any new edge mechanism with operative global standards.

1.2 Objectives

This case study facilitates a comprehensive analysis of possible risks and security issues involved in a biometric-enabled banking infrastructure. We investigate the existing unimodal biometric security model and potential enhancement required to alleviate the aforementioned security concerns. In this study, we also recognize the secure access control for biometric systems. Further, we also evaluate the security aspects from users perspective as well as constraints required to design a viable authentication paradigm for banking organization.

This chapter is structured as follows. The next section, i.e. Sect. 2 presents a summary of existing works and emerging infrastructures proposed in literature that signifies the thrust for the deployment of security models for ICT infrastructure. In this context, we study the usability of unimodal and multi-biometric security models and access control mechanism for such environments. Section 3 presents a general security model for unimodal and multi-biometric authentication in protected and unprotected domain. Following, the case studies for military organizations and

banking infrastructures have been discussed in Sects. 4 and 5, respectively. Section 6 presents the countermeasures defined for disaster management such as system failure or privacy invasion. Finally, we conclude with future research prospects in Sect. 7.

2 State-of-the-art

At present, facilitating a secure and trustworthy environment is a concern for majority of the sophisticated applications where security and privacy are paramount [7, 18]. A security strategy must fulfill the requirements of confidentiality, availability, integrity, accountability, and non-repudiation. The underlying requirements need to be accomplished with a capability of controlled information access and audit mechanism. At present, these access control schemes are setup by physical devices such as smart cards/keypads monitored with security personnel. The access may be granted or denied based on active users and passive entity (e.g. file, token, or cards). Here, we investigate eight different security models deployed and widely used in literature [5, 6]. These security models differ in their goals, characteristics, weaknesses, and strengths when incorporated in banking, military, or any other ICT infrastructure. Bell-LaPadula Model (BLM) provides the access based on subjects, objects, and operations for static infrastructure. Since most of the application are on distributed network and expect resilience against attacks, BLM becomes obsolete in handling dynamic systems. BIBA model (BM) provides access based on integrity level. It also tries to prevent impersonation and theft attempts. However, access to sensitive information will not be granted to users. Face, fingerprint, and hand geometry are observed to be the most suited biometric traits for user authentication under BLM and BM models. Third, Clark-Wilson model (CWM) provides access to certain objects and user access based on roles. Due to association of subjects and information, it alleviates unauthorized users to perform any alteration to records. However, it is not suited for distributed systems running over network. Brewer-Nash Access Model (BNAM) [8] allows access control in dynamic systems that are susceptible to change. The most suited biometric trait is observed to be fingerprint. Further, BNAM tries to mitigate the conflict of interest present in different firms. However, it fails to provide adequate privacy protection. Nash Graham Denning Model (NGDM) [11] facilitates with rigorous protection rights since it includes rights to create and delete subjects, roles, and objects. It is treated as an alternative to DAC and suited over fingerprint biometric. Theoretically, any security model may be utilized for an application. However, there is a significance of certain features that are desired and required for certain applications. For example, iris-based authentication is desired for application where high accuracy is an utmost need. Also, it does not deteriorate over time. The limitation lies in authentication time for making a call to private ID driver and imposing a user to open eye long enough.

In Role-Based Access Control (RBAC) model [10], different roles and rights are defined for designated users based on job functions. Role-based control restricts them to query (read or write access) confidential resources within their rights. This

makes RBAC ideal for profile generation and network-based applications. Also, it allows the authentication tool to identify user within a limited amount of time. Hence, iris recognition is suited for RBAC model due to the restricted matching time characteristics. In a similar way, NGMD model [11] is suited for fingerprint-based authentication due to the presence of eight protection provisions with 'owner' and 'controller'. Due to these multiple checks in authentication, fingerprint is proved to be the most suited trait under this model.

In contrast to above two models, Mandatory Access Control (MAC) model [15] does not allow a user to have complete access to any classified resource even it is created by staff. MAC focused on data sharing for different sensitivity levels. Hence, it is recommended to deploy it over sophisticated or sensitive resources that need to be kept confidential. It is also advised to utilize thermal imaging due to the fact that a user may be in a very close contact and information access may happen in darkness also. Finally, Discretionary Access Control (DAC) [16, 19] model restricts access based on user authorization. It is appropriate to network applications since it relies too much on objects and provides protection to data from users.

Conclusively, we observe that each model has its own pros and cons with suitable biometric traits and associated objectives. The suitability is also determined on the basis of reliability, accountability, integrity, confidentiality, and non-repudiation. In these case studies, we have performed an assessment of a military and a banking organization with reference to their access control mechanism. The observation depicts that most of the organization deploy DAC model with a foundation of RBAC model for secure authentication. The lower level admin and staffs are handled by MAC model as their entry/exit can be managed by a set of definite rules and permissions. In few placed, a hybrid model involving DAC, MAC and RBAC are utilized where RBAC controls the permissions based on functional units within organization. This mechanism generally streamlines the complex authentication process by verifying user with biometric and assigning role/privileges for access.

3 Conventional Security Model for Biometric-Based Authentication

In this section, we present a general security model with all the components to perform exhaustive security analysis for our method and discuss the three major requirements needed for template protection as described in Sect. 1. For reader's clarity, we also describe each assumption taken into account for the entities associated with the verification procedure providing a more general perspective of how the multimodal fusion framework deals with different threats or privacy invasion attempts. An explanatory diagram (see Fig. 2 (left)) illustrates the verification procedure adopted for an unprotected scenario for two entities:

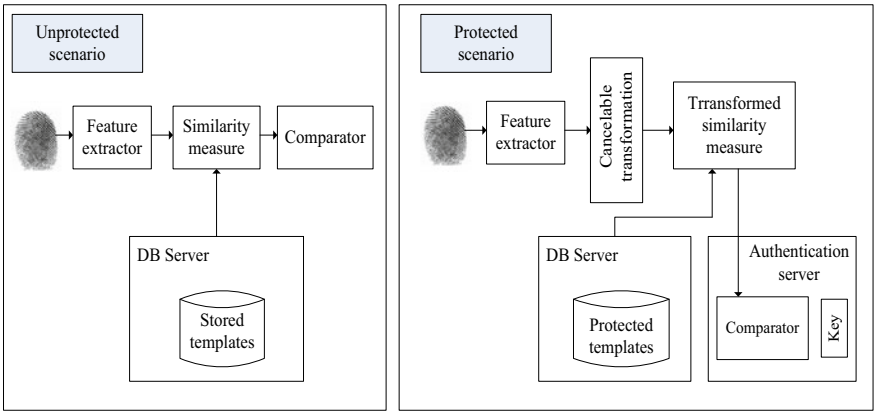


Fig. 2 Unimodal biometric verification in unprotected versus protected domain

Client: The client performs data acquisition, feature extraction, and represents the features in the form of verifiable templates. Next, it computes the similarity score between the query and stored template. Finally, user's identity is verified based on a predefined threshold.

Server: The server maintains the true biometric template for each user present in the database and shares these templates with the client for verification. To strengthen the privacy of a user, the server must send client's biometric data without pulling any other information and protect the biometric information stored in the database simultaneously.

In contrast, a different security model is utilized for verifying protected biometric template as shown in Fig. 2 (right). In the protected scenario, all the biometric information that is either stored or communicated between client and server are transformed (i.e., protected). Hence, the mentioned entities play the following roles:

Client: The client first acquires the data and extracts the features. Next, it applies a cancelable transformation to derive protected biometric templates and stores it onto DB server.

DB server: It contains the database consisting of only protected templates and shares these templates with the client for verification.

Authentication server: It comprises the user-specific key and comparator. Also, it computes the final verification decision by comparing stored and query template.

The following assumptions are taken into account to perform secure authentication in a multi-biometric framework:

– An imposter may get access to any one of the server but the DB server and authentication server would not intrigue.

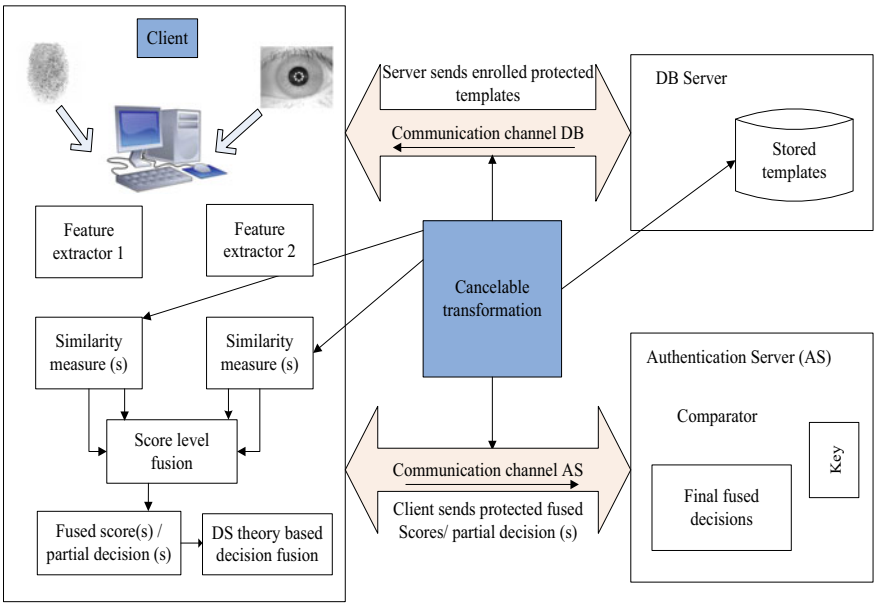


Fig. 3 Security model: Hybrid fusion

– The client does not know the user-specific key hence it can neither extract the original template from the protected one nor the similarity score obtained through protected modalities assuming that the client serves honestly. As a result, there is no invasion possible of biometric information in the communication link.
– Similarly, the authentication server would not be allowed to access either the original template or stored protected template avoiding any trace for instigating biometric data. Also, it is assumed that all involved entities adopt the protocol and thus the score evaluated by the clients is correct.

Based on the security model illustrated in Fig. 3, the secure and privacy-preserving authentication in a multi-biometric fusion framework should exhibit the following requirements:

1. The client alone should have access to the original biometric template,
2. Only the protected template should be stored in the DB server. Hence, it can never be visible to any other entity,
3. The match score/decision output cannot be transmitted as it may be utilized to launch inversion/ hill-climbing attacks.

To ensure the privacy protection, the authentication system should fulfill the four requirements, i.e. non-invertibility, diversity, revocability, and performance as described in Sect. 1.

4 Case Study: Automated Integrated Fingerprint Biometric System for Military Organization

The current military authentication systems have a lot of security and privacy vulnerabilities in terms of mechanism and applications. These vulnerabilities are mostly discovered at the time of identifying visitors and dependents living in military cantonment. In general, the visitors were being identified using user identities, National Register of Citizens (NRCs), or traveling documents. This procedure introduces security concerns such as impersonation and masquerading if anyone possesses someone else's identity. This case study deals with the situation if no proper record and identification parameters are found with dependents and children living in the military cantonment or barracks. Few other security concerns in this case study are traced from the heart of Zambia's Army Headquarters offices, which host civil functions such as wedding and other social meetings.

4.1 Requirement of the New System

Development of a new system is divided into iris/fingerprint biometrics and military database requirements. These two requirements are integrated to form an automated integrated fingerprint biometric system for military organization (AFBSMO). The integration of biometric application into military database provides the defense architecture with the stipulated security resilience. In addition, AFBSMO would aid military personnel an efficient service and document delivery system. However, the product must be narrated and modeled in the best possible manner to be fit into software development processes. It would contain documentation of the proposed system as it transverses from one development stage to another.

4.2 Product Perspective

The AFBSMO is the latest security system that was intended to replace the present manual visitor/staff authentication in the Zambia Army [14]. Figure 4 shows the context diagram with external entities and system interfaces. The system is expected to mature over several version/releases and ultimately installed at a number of barracks and formations.

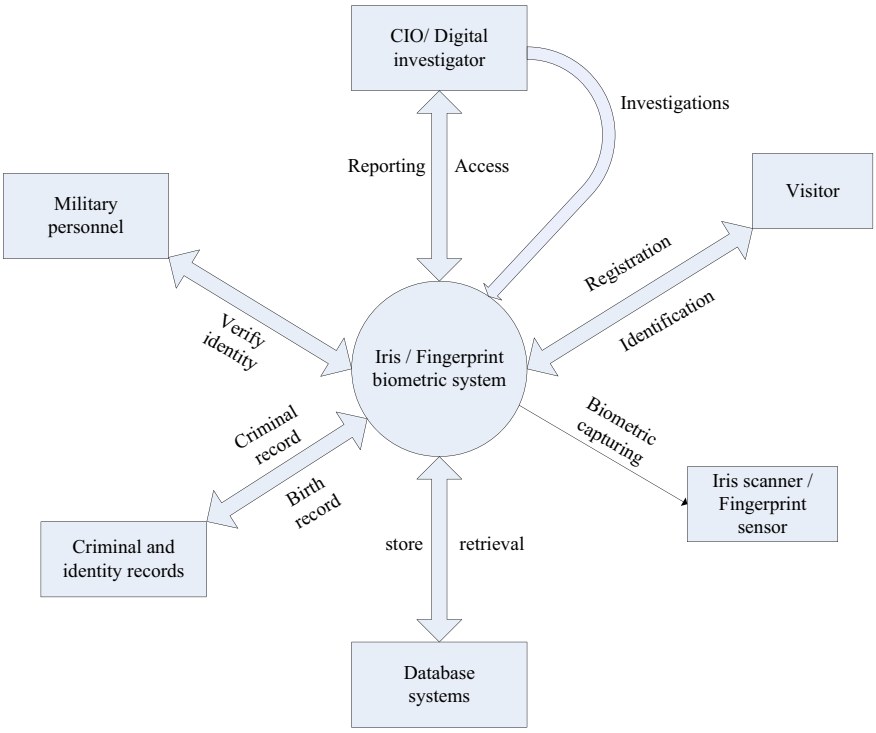


Fig. 4 AFBSMO context diagram

4.3 Objectives and Process Management

The objective behind the development of this new system is to amend the security concerns observed in the present manual system. These concerns should be mitigated in order to upgrade military area security clause and services. The security clauses and services can only be improved through automation at the application level. The improvements are sought through fetching information such as applicant details, fingerprint template, officer information recording, criminal data investigations, and human-descriptive information. The first module is the iris/fingerprint biometric authentication system, which should be installed at entry/exit points or checkposts for secure access control. The second module is to ask visitors/staff to produce automated service identification card. The last module includes collection of security information from the guests. Figure 5 represents different processes, actors, and data flow for this new automated system.

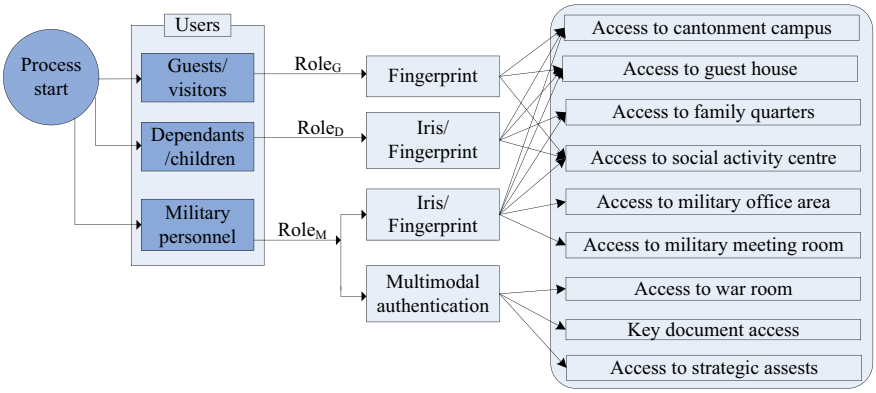


Fig. 5 Users and their roles for different applications

4.4 Military Personnel Database

The military database is utilized to handle operations at peace time and war times. It consists of record/log tables, queries, and procedures committed for personnel duties.

4.5 Roles of Military Personnel

The Military personnel are devoted to supply tactical defense support to the Zambia Army in military operations. If deployed, few roles the regimental military personnel includes:

- General investigation in peace times
- War/crime scene inspection in war time
- Collection of criminal evidence
- Reconnaissance patrols
- Prisoner handling
- Search operations and road blocks
- General policing duties within operational bases
- Foreign personnel and military training
- Provide close protection operatives for senior military personnel on operations

4.6 Military Personnel Functioning

The military personnel unit is responsible to maintain law and order within the military region. The unit is deployed at entry/exit points for traffic control, and to provide security to offices, offices personnel, installations, ammunition units, and the barracks.

As illustrated in Fig. 5, civilian/Guests have to disclose their fingerprint to get verified. The guests/visitors can only access the cantonment campus area, guest houses and social activity center. Foreigners are compelled for security clearance from Ministry of Defense each time they need to visit a barrack. For dependents or children living inside the campus have to use their biometric information such as iris or fingerprint to access the cantonment campus, family quarters, and social activity center. For this purpose, military personnel capture 10 fingerprints from each and manually record their voice. Their duty also includes keeping database of disciplinary record of service personnel, offences committed and identity information of officers in active service and reserve forces.

The different sectors require permissions to different subjects based on their roles. The model explained resembles with DACs role-based access with RBACs implementation for permissions. Military personnel also use their biometric information to access their barracks, quarters, and social activity center. For more sophisticated or privileged access, multi-biometric verification may be performed (e.g. iris and fingerprint) utilizing more than a single modality. Sophisticated access includes access to war room, key documents, strategic assets, missiles, radar rooms, etc.

5 Case Study: Automated Integrated Fingerprint Biometric System for Banking Infrastructures

In this case study, a summary for current banking infrastructure has been conducted in New Zealand. The summary depicts two analyses as follows: (1) Assessment of security model adopted for the suited biometric trait; (2) Project Bio-Sec implementation in New Zealand bank. The assessments have been carried out through a number of primary sources (e.g. interviews) and secondary sources (e.g. journal articles, textbooks, technical reports etc.). Next, a questionnaire has been created comprising qualitative and quantitative aspects to assure the mitigation of dominant privacy concerns in banking firms. These aspects include security processes, policies, and feedbacks from employees and management.

5.1 Analysis of Biometric Technologies

Majority of biometric solutions include sensor and pattern matching software used for data acquisition and user verification, respectively. These two components must be precise and unerring for effective implementation. Moreover, the large-scale deployment of such system would rely on sensing surface, computation time, and performance of pattern matching software. Then, feedbacks and consumer satisfaction reports have been collected from different participant biometric traits. Overall, the observations from iris, fingerprint, face, and palmprint trait signify the privacy and reliability concerns due to visible sensing surface. Such concerns arise due to the information forging or privacy invasions. However, vein pattern is considered to alleviate these limitations as they are not visible to human eye and difficult to manipulate.

5.2 Analysis of Information Security Models Used in Banking Systems

The development of a privacy-preserving infrastructure is a major concern for banking firms. It should fulfill the objectives, namely, confidentiality, availability, accountability, integrity, and non-repudiation with a role-based access of information. The access control mechanism generally includes the usage of smart cards, tokens, or keypads where entry is monitored by security managers. However, the above-stated mechanisms are easy to forge/impersonate due to the fact that a user has to remember a key. With biometric-based authentication, this inherent drawback associated with conventional access control can be mitigated. Though any security model can be deployed for access control, few features match certain requirement for an application. For example, iris authentication is utmost accurate and exhibits optimal performance yet it requires a user to open eyes for a while for verification.

5.3 Project Bio-Sec Development

In this section, we present a case study on project Bio-Sec that aims to adopt biometric-based access control. The assessment from literature review and questionnaire acts as input to Bio-Sec frameworks. Here, an effort to handle unauthorized access (external partners/internal staff) and privacy concern is prioritized. The project initiates its operations by the authentication of internal bank staff through a standard access control mechanism.

Figure 6 illustrates the architectural units of the proposed biometric-based authentication system, i.e. Bio-Sec. The overall objective lies in providing role-controlled identification of three categories of people, i.e. internal staff, business personnel, and support staff. The project defines the roles and access rights to each individual. The

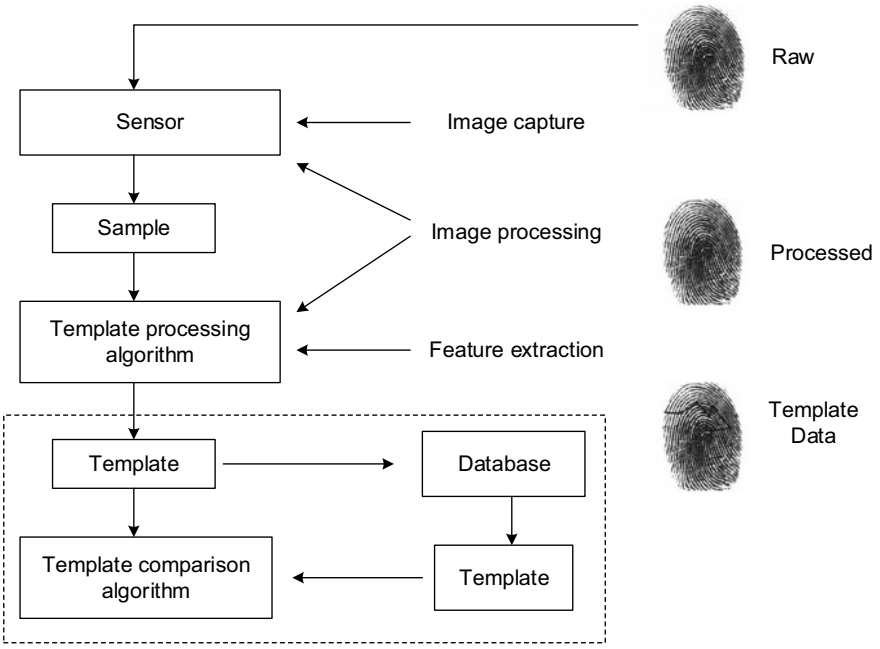


Fig. 6 Proposed biometrics security components (Bio-Sec project)

analysis also derives the framework and unveils the outcomes of the case study for project. Figure 7 depicts the layout of secure framework that aims to aid policy management and standard operating procedures. Bio-Sec introduces two primary access control paradigm to provide resource access using fingerprint-based smartcard. This smartcard or Bio-secure integrated ID card is utilized for service access and SSL.

A 15-member Bio-Sec team was appointed to provide directions, testing feedback, and recommendations for their own transactions. Following the suggestions of 15-member squad, the amendments are made to existing security provisions. Next, an action plan is scheduled for training to all other employees for this integration to the existing infrastructure. Any secure infrastructure should have managerial controls, data recovery in event of breaches, and user acceptance towards a successful and trustworthy system. The case study affirms that biometric-based authentication would ensure a secure banking environment if performance, permanence, ease of use with technological mobility are offered.

Hence, Bio-Sec project was deployed to create a secure banking platform to all users based on their roles in cognizance with human rights protection. This also supported to frame data management policies and legal standards.

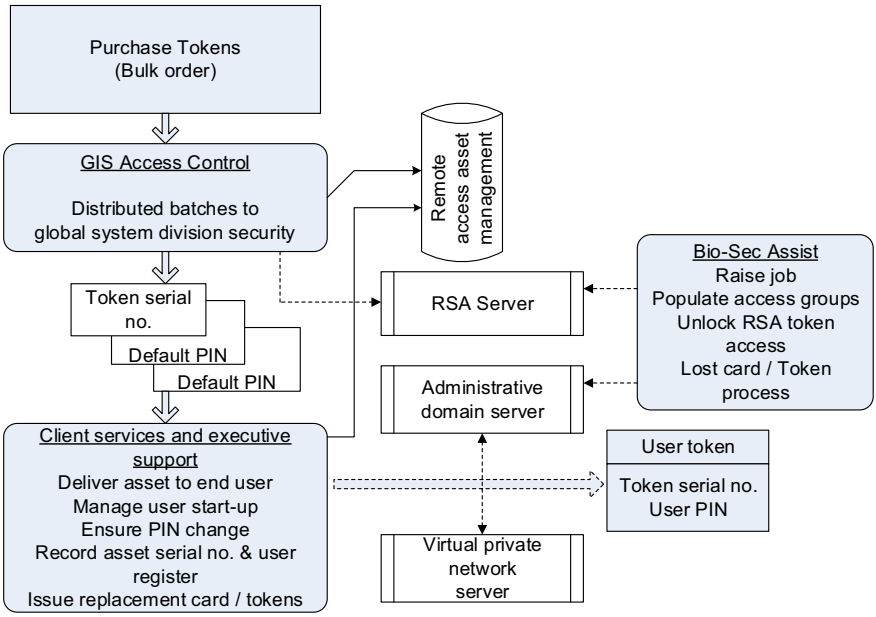


Fig. 7 RSA token unlocking: Bio-Sec project

5.4 Discussion on Case Study Findings

In the preliminary study, the feedbacks are collected through interviews and questionnaires based on privacy concerns and security plans were analyzed. It would in turn aid to cope up with the technological issues and feasible strategies that should suffice prime challenges such as privacy fear, human tolerance, legal issues, and firm change. These factors can be grouped into four broad categories viz. technology, monetary, management, and legal factors. Technology factors include accuracy, fault-tolerance, and aging effect on performance. Privacy factors are associated with confidentiality, and level of security (low, moderate, and high) based on roles. Monetary factors include cost of implementation whereas management factors deal with willingness, adaptability, and coping up with the new technologies. Finally, legal and ethical factors involve the government rules, social, and physiological issues.

6 Countermeasures

The primary issues and safety concerns faced by banking organizations are identified after Bio-Sec implementation. Four categories of different factors are analyzed and success factors are reported here:

6.1 Technology Factors

Technology factors relate to accuracy and error rate measure are considered to be utmost importance. These factors are accountable and provide robust user authentication.

The technology countermeasures include following:

- The biometric trait promoting flexibility to viable tolerance levels and possibility of evolvement in future should be chosen to enhance the ICT infrastructure.
- There must always be a scope to adopt a cooperative approach in preserving existing privacy but improving security measures also.
- It is required to assess the risks involved in different applications.
- There can be a provision for multi-biometric authentication for sophisticated resources and access control.

6.2 Monetary Factors

Monetary factors include producing recommendations and requirement collection by comprehensive analysis of cost/benefits. Another aspect lies in evaluating interoperability and process efficiency.

6.3 Management Factors

Following management factors are observed with Bio-Sec review:

- A top-down lead approach for resolution to firm challenges.
- Discourage information sharing even within organization to promote new security solutions.
- Promote plans and strategies to clinch new techniques for security measures.
- Maintain a transparent and environment for users trust building.
- Make incremental modifications to security provisions to meet the security standards.
- Availability of motivated, trained, and skilled employee. For this, suitable staff training programs must be conducted.

6.4 Legal/Ethical Factors

The ethical and legal factors are associated with ensure government laws and human rights inclusion in the firm. The following legal/ethical factors may exist:

– Setup controls to ensure compliance to security laws and regulations.
– Ensure safety of information from different attacks.
– Promote novel biometric authentication schemes to navigate from conventional password-based traditional techniques.

7 Summary

These case studies were focused at identifying the concerns and issues that are needed to be mitigated before usage. The studies conducted at military cantonment and bank unveiled that role-based authentication at several checkpoints is a mandatory concern. The studies also established that there should be a disaster management strategy to alleviate the fear caused due to technology shift. The different countermeasure discussed in this chapter would aid military, banking and other sophisticated infrastructures to plan their operating procedures and protocols with required flexibility.

This chapter describes two such biometric projects AFBSMO and Bio-Sec, which were deployed based on the countermeasures discussed. Both of the AFBSMO and Bio-Sec have also associated a supplementary mechanism to manage reliance and failure of biometric system. Hence, it can be concluded that an adaptable, viable, and rightful biometric solution is required that must address different social, technological, and ethical concerns across applications to meet future needs.

References

1. Canadian passenger accelerated service system (CANPASS) Private aircraft program,canada border services agency. https://www.cbsa-asfc.gc.ca/prog/canpass/privateair-eng.html. Accessed 14 Sept 2018
2. CLEAR Program, CLEAR airport security and automate security. https://www.clearme.com/home. Accessed 14 Sept 2015
3. The eyes have it. http://www.accessexcellence.org/WN/SU/irisscan.html. Accessed 25 Apr 2017
4. ISO/IEC 2382-37:2012 (2012) Information technology—vocabulary—part 37: biometrics 2012. https://www.iso.org/standard/55194.html. Accessed 19 Nov 2015
5. Bell D, LaPadula L (1973) Secure computer systems: mathematical foundations and model. Mitre Corp Report, pp 74–244. https://ci.nii.ac.jp/naid/10013110878/en/
6. Biba KJ (1977) Integrity considerations for secure computer systems. Mitre Corp Report, pp 1–64. https://apps.dtic.mil/sti/citations/ADA039324
7. Boukhonine S, Krotov V, Rupert B (2005) Future security approaches and biometrics. Commun Assoc Inf Syst 16(1):937–966
8. Coull SE, Green M, Hohenberger S (2011) Access controls for oblivious and anonymous systems. ACM Trans Inf Syst Secur 14(1):1–28. https://doi.org/10.1145/1952982.1952992
9. Daugman J (2009) Iris recognition at airports and border-crossings. In: Li SZ, Jain A (eds) Encyclopedia of biometrics. Springer, Boston, USA, pp 819–825

10. Ferraiolo DF, Barkley JF, Kuhn DR (1999) A role-based access control model and reference implementation within a corporate intranet. ACM Trans Inf Syst Secur 2(1):34–64. https://doi.org/10.1145/300830.300834
11. Gopinath K (2006) Access control in communication systems. In: 2006 1st international conference on communication systems software middleware, pp 1–8. https://doi.org/10.1109/COMSWA.2006.1665185
12. ISO/IEC 24745:2011 (2011) Information technology—security techniques—biometric information protection. https://www.iso.org/standard/52946.html
13. Jain AK, Nandakumar K, Nagar A (2008) Biometric template security. EURASIP J Adv Signal Process 2008(113):1–17
14. Kalunga J (2015) Integrating fingerprint biometrics system into the military police database: the case of zambia army. University of Zambia, Tech. rep
15. McCollum C, Messing J, Notargiacomo L (1990) Beyond the pale of mac and dac-defining new forms of access control. In: Proceedings of the IEEE computer society symposium on research in security and privacy, pp 190–200. https://doi.org/10.1109/RISP.1990.63850
16. Poniszewska-Maranda A (2008) Access control models in heterogeneous information systems: from conception to exploitation. In: 2008 international multiconference on computer science and information technology, pp 821–826. https://doi.org/10.1109/IMCSIT.2008.4747337
17. Rathgeb C, Busch C (2017) Biometric template protection: state-of-the-art, issues and challenges. Instit Eng Technol. https://doi.org/10.1049/PBSE004E_ch8
18. Simoens K, Bringer J, Chabanne H, Seys S (2012) A framework for analyzing template security and privacy in biometric authentication systems. IEEE Trans Inf Forensics Secur 7(2):833–841
19. Venkatraman S, Delpachitra I (2008) Biometrics in banking security: a case study. Inf Manag Comput Secur 16(4):415–430. https://doi.org/10.1108/09685220810908813

Chapter 11
Cloud-based Remote Healthcare Delivery and Its Impact on Society: A Case Study

Himadri Sekhar Ray, Sunanda Bose, and Nandini Mukherjee

Abstract Delivery of primary healthcare services to the doorstep of rural people in developing countries is a challenging task. It requires huge skilled workforce along with proper medical equipment for medical diagnosis and monitoring. In this chapter, we first present a survey of the cloud-based remote healthcare applications based on sensor, mobile and cloud technologies. Next, we describe a highly modular, easily re-configurable, touch-screen-based application with a user-friendly graphical interface for use in rural areas with no or minimal healthcare facilities. We have set up kiosks in the villages of West Bengal, India, where health assistants, with the help of our application, measure the patient's vitals, collect symptoms by using a knowledge base and perform clinical examinations. The collected data are stored in a back-end cloud database server. Sitting at urban locations, doctors can check and examine patients' clinical data, and prescribe medicines and tests. After 2 years of operation, a survey was conducted to understand the impact of using this mode of healthcare delivery. The chapter summarizes people's perceptions and views in the second part of the chapter for further improvement and to strengthen the arena of the public health system. We also claim that such a system can become useful to handle a pandemic situation.

1 Introduction

Primary health care centers (PHCs) are set up in rural and remote areas to provide healthcare services to a majority of the rural people. But these centers suffer from a lack of resources, including doctors [1], health workers and all types of clinical facilities. This scarcity of resources can turn out to be disastrous for rural people.

H. Sekhar Ray (✉) · S. Bose · N. Mukherjee
Jadavpur University, Jadavpur, India
e-mail: himadri.1111@gmail.com

S. Bose
e-mail: sunanda.bose@msn.com

N. Mukherjee
e-mail: nmukherjee@cse.jdvu.ac.in

M. Chaturvedi et al. (eds.), *Recent Advancements in ICT Infrastructure and Applications*,
Studies in Infrastructure and Control, https://doi.org/10.1007/978-981-19-2374-6_11

For example, during the COVID-19 pandemic also, it was observed that no steps could be taken in the vast rural area of the country where hundreds of thousands of migrant workers from the cities have returned. It has become difficult for the administration to track these people and deliver basic health care to the villagers living in these areas. The village health workers, such as Accredited Social Health Activist (ASHA) and Auxiliary Nurse Midwife (ANM) are being engaged to handle this situation. However, in the absence of proper guidance [2], even preliminary care and advice have become almost impossible. Moreover, after the wide spread of the Corona pandemic, telemedicine is considered to be an effective solution as "hospitals remain at a risk of spreading COVID-19 among co-patients and visitors and their family members". In the countries which have limited medical resources, doctors should not be infected during this pandemic, and therefore telemedicine or mobile apps can be effectively used as pointed out by some telemedicine workers.

Medical web portals [3] and telemedicine apps [4] have been developed to help health assistants working in the rural centers to provide information about treatment and follow-up actions. However, such portals are focused on only the dissemination of relevant information. They are unable to provide assistance for vital measurements and capture the required information for proper treatment. When doctors are not available at the PHCs, rural health assistants provide assistance for the treatment and in case of emergency they contact the urban doctors using phone calls. This is certainly not an effective way of healthcare delivery.

Thus, there is a need for developing alternative techniques for remote healthcare delivery by medical professionals. The proliferation in the use of mobile devices, such as mobile phones and tablets, and the innovation of newer medical sensor devices with standard communication interfaces facilitate access to health data in real-time scenarios. Our aim is to use these technologies to develop proper medical services at the primary level in rural areas. This paper first describes a real-time application KiORH (**Ki**osk **O**perated **R**ural **H**ealthcare) [5] based on sensor, mobile and cloud technologies that provide an integrated environment for the patients and remotely located medical professionals to interact in a meaningful manner and to create a treatment episode for delivering basic healthcare services to the rural patients [6]. The health assistants collect the patient's complaint-related data and transmit the data to the doctor over the Internet. The doctor on the other side of the system (probably in urban areas) studies those data in real-time and generates prescriptions or puts forward advice related to the treatment of the patient.

The application was used for 3 years by setting up health kiosks in rural environments. One of the kiosks was set up in the village 'Barrah' of Khoyrashole block in Birbhum district in West Bengal, India, and a group of health assistants were engaged to provide services to the patients visiting the kiosks. The kiosk not only served the Barhrah village, but also extended its support toward adjacent villages, like kankartala, Heriektale, etc. A survey has been conducted among the rural people in these villages. In this paper, we also present the result of the survey based on the experiences of the rural people.

The paper is organized as follows. Section 2 gives an overview of the present state of the art. Section 2 describes an overview of our application. Section 3 and

Sect. 4 discuss the major features of the kiosk side and doctor side interfaces of the application. The methodologies of conducting a survey to understand the social impact of running a health-kiosk have been discussed in Sect. 5, and the results of the survey are presented in Sect. 6. Section 7 summarizes the observations from the survey results. Section 8 concludes the paper.

2 Present State of the Art

In the current scenario, kiosk-based healthcare startup is transforming health care in rural areas. In this section, a few well-known applications for rural health and kiosk-based health care are introduced [7].

- Gramin Healthcare was founded to provide affordable primary health care and specialist care in the rural and poorest regions of India. This Gramin Healthcare kiosk provides diagnosis, subsidized doctor consultation by live online video/audio chat and medicines through their Healthcare platform [8].
- iKure Health Monitoring kiosks were set up in rural areas of West Bengal, India, where healthcare infrastructure is negligible. These kiosks have been set up by iKure Techsoft, in partnership with Aegle Angels Foundation. The kiosks are equipped with WHIMS—the wireless health incident monitoring system which can capture data from a maximum of 16 medical instruments or 16 patients at the same time. The WHIMS reads measurements from these instruments, and wirelessly transfers and stores in a data server as the patient's medical history [9]. Based on the test results, the Registered Medical Practitioner (RMP) can prescribe medicines.
- e-Mitra is a kiosk-based healthcare delivery solution for the rural areas, launched by the government of Rajasthan, India. Through this application, an expert doctor across the globe attends the patient in rural areas. The e-Mitra service allows a patient to write a health query, along with pictures, a lab report, etc. and post to doctors across the world and also get answers online or over the phone from experts [10].

There are many other mobile-based healthcare applications available in the market. The most well-known and popular applications like Dr Lal PathLabs [11], Practo [12], Credihealth [13], Tweet2Health [14], MedicExpress, Curofy [15], Healthonphone [16], Netmeds [17], Symptomate Symptom Checker [18], MDCalc [19] and Prognosis : Your Diagnosis [20] work fine in the places where enough network bandwidth is available. But these applications are unable to offer a coherent procedure to enable the healthcare professionals or the doctors to treat the patients remotely by gathering precise information about the symptoms using a previously gathered knowledge base. Most of these applications use the English language interface at the front-end which is not fully usable by the moderately trained health assistants or rural people in non-English-speaking countries.

Our application works with a different principle compared with the applications mentioned above. First of all, this application can help the health assistants with the processes, like checkups and examinations. It gathers knowledge about the symptoms of a patient through a well-organized question-answer-based interactive session. This organized question-answer-based symptom collection section of the application is carried out using a knowledge base which has been developed by various medical practitioners. With the help of our symptom collection procedure, doctors can analyze the health condition of the patient even if there is no physical interaction between a doctor and a patient. Users can also interact with this app using their native language. Patients can check the details of prescribed medication, medical advice and proposed medical tests in their language.

Another important feature of this application is the interaction facility among the users, i.e. doctors, patients and health assistants, in different modes. Even if sufficient Internet bandwidth is not available, our application checks the connectivity and sends the vitals to the cloud server via an SMS server using SMS and proceeds to the necessary next step.

Overview

The KiORH application has been developed for the rural kiosk center where resources like Internet connectivity and electricity are big constraints. The application has two main sections. One is for the kiosk side, from where the kiosk operations are carried out [21]. There are five modules in this section for performing five different tasks.

- Demographic information gathering of patient.
- Vitals collection of patient.
- Symptom collection using knowledge base.
- Collection of family history, patient's habits and other medical documents.
- Clinical examination, depending upon the collected symptoms, family histories and patient's habits.

Another section is at the doctor's side, from where the doctor can log in, check patient vitals and make prescriptions. The Kiosk side application is developed using Complex HTML5, CSS3 and JQuery3.x wrapping with Cordova [22] (formerly PhoneGap) into a native container which can access the device functions on several platforms. For running, developing and testing the KiORH app using Cordova, we have mainly focused on Android devices such as Android Tabs/smartphones that are relatively cheap and easily available in the market. For cloud storage, we have used VPS hosting with 2GB RAM, 2vCPU and 1000GB primary bandwidth [23].

To operate the kiosks, health assistants (rural girls) have been trained to use the information and communication technologies (ICT) and to collect necessary data from the patients. They have been provided with Android-based mobile phones and tablets. Our Android app is installed on a mobile device [24]. On the other side of the application, doctors can view the complaint of the patient, along with other details

Fig. 1 KiORH: Kiosk Operated Rural Healthcare delivery

like vitals, history and even old prescriptions and relevant reports. Based on these data and through remote interactions over the Internet, a doctor can put forward his/her advice for the patient and prescribe medicines. A high-level overview of the kiosk-based operation is shown in Fig. 1.

3 Kiosk-Based Patient Data Collection

When a patient visits a kiosk, the health assistant measures clinical parameters, like blood pressure, SpO2 and pulse rate along with some other vitals of the patient using e-Health sensors and collects demographic data of the patient [25]. The data are sent to the application running on the Android tablet/mobile phone through Bluetooth. In the next step, the health assistant asks various questions depending on the symptoms of the patient. The questions are displayed on the app screen from a knowledge base and the health assistant selects appropriate answers in the provided space on the application screen [26]. After gathering present symptoms of the patient, the health assistant queries about family history, patient's habits, gets images of old prescriptions and other medical documents and inputs these data/documents to the app. Next, the application guides the health assistant to make a clinical examination of the patient. The application makes an analysis with the details given by the patient and based on this analysis, it displays a list of clinical examinations which are required to be performed for the particular patient. The health assistant carries out the exami-

nation and puts the results in the provided space in the app and selects a doctor from a pool of currently available doctors. The data is then uploaded to the cloud server for further use by the doctor.

3.1 Notable Features of KiORH

The following are the notable features of KiORH that make it different from other similar applications and also suitable for the developing countries where scarcity of resources is a challenge.

- **Role-based login:** Role-based login is the first module of the application. This module authenticates valid users to use the application. The module connects with the cloud server via REST API [27]. It sends the username and password of the user (health assistant/patient/doctor) to the server and fetches the user details as a response. If the application gets a positive valid response from the server, it stores the fetched user (e.g. health assistant) information in the local memory for further use. The role-based login allows the users to access only a part of the system, and prohibits them from accessing other parts that they are not authorized to access.
- **Sensor data acquisition:** Sensor data gathering and real-time flow from kiosk to doctor end is another important activity of the kiosk application. A module acquires sensor data of the patient who is currently under checkup using e-Health sensor devices through Bluetooth connection and sends the data continuously or the last seen record to the cloud server for further use by the doctor [28].
  Presently, the Android application installed on the mobile device communicates with the sensors via USB serial communication. Sensors are attached to an Arduino board which sends the sensed data through the USB port. The Arduino board is equipped with the e-Health [29] shield. The set of sensors attached with the shield includes a body temperature sensor, SPO2 sensor, airflow sensor and blood pressure sensor that come bundled with the e-Health sensor kit as shown in Fig. 2. Figure 3 displays the sensor data collected from a patient.
- **Symptom collection:** The kiosk-side Android application uses a knowledge base for collecting symptoms and other details of the patient. In the first step, multiple main symptoms are listed. The health assistant can select one or more main symptoms by tapping the app screen based on the patient's complaint. After the selection of symptoms, sub-symptoms are shown on the app screen. Next, after the selection of sub-symptoms, the currently selected symptoms and sub-symptoms-related questions are displayed along with multiple possible answers [30].
  An example of symptom collection for a patient with a headache using the knowledge base is given below. At first, from the list of symptoms, like *Pain, Swelling, Difficulty in breathing*, etc. the health assistant selects *Pain*. Then the list of sub-symptoms of *Pain* will show up. The list includes the symptoms *Headache, Ear Pain, Teeth Pain, Throat Pain, Back Pain*, etc. The health assistant selects 'Headache' as a sub-symptom. After this selection, the symptom-related ques-

Fig. 2 Health sensors

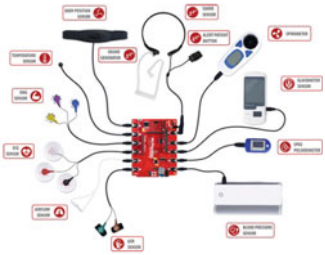
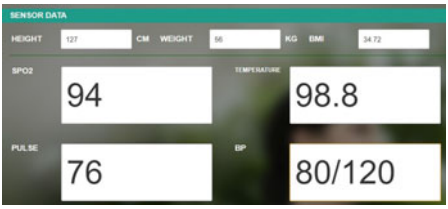


Fig. 3 Sensor data screen



tions like *duration, starting location, current location, intensity, nature*, etc. will be displayed along with multiple answers to each question. Answers to all the related questions are to be entered by the health assistant. Figures 4 and 5 show the screens displayed by this module.

A multidimensional hash table is used for storing the knowledge base. The hash table maintains key-value pairs for storing the details of all symptoms, sub-symptoms and related questions and answers. The idea of hashing is to distribute entries (key-value pairs) uniformly across an array. Each symptom is assigned a key. There are also multiple keys for maintaining the sub-symptoms of the symptoms and similarly, there may be multiple keys for questions and a set of possible answers under sub-symptoms. For fetching from the hash table, hash functions are used, with which we can pass the keys (symptom keys) as arguments and return the symptom details as returned hash values. The structure follows the JSON format for storing the symptoms and their details. Grammar of the knowledge-base JSON document is shown below.

$$
\begin{aligned}
&Knowlegdebase =< Symptom, ClinicalExam > \\
&Symptom =< Category^{+} > \\
&Category =< id, title, Subcategory^{+} > \\
&Subcategory =< id, title, SymptomQuestion^{+} > \\
&SymptomQuestion =< id, title, type, answer^{+} > \\
&ClinicalExam =< ExamCategory^{+} > \\
&ExamCategory =< id, title, ClinicalQuestion^{+} > \\
&ClinicalQuestion =< id, title, answer^{+} > \\
&type = string \,|\, numeric \,|\, select \,|\, check \,|\, radio
\end{aligned}
\tag{1}
$$

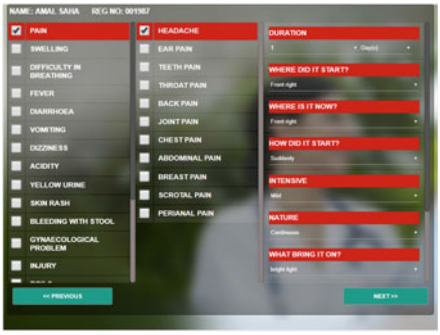


Fig. 4 Symptom gathering screen (English)

Fig. 5 Symptom gathering screen (Bengali)

- **Multilingual interface:** Since the application is intended for use by rural people for providing primary healthcare facilities in rural areas, local language interfaces are required [31]. Most of the rural people are not efficient in the English language, and are more free to use their native regional language. So, we provide the facility to use the kiosk application using their regional languages. There is a module with which the application provides multilingual support to the app and anytime the health assistant can switch to any regional language. Currently, only 'Bengali' support is set up and installed as shown in Fig. 5.
  For the entire application, we are maintaining a JSON table to represent multilingual texts. Inside the table, we have a collection of different languages that the application supports. Each such language has a dictionary that maps a fixed set of keys to values corresponding to that language. As the set of keys is fixed for all languages, we can query these dictionaries to get the phrases corresponding to a language by providing the key and language title [32]. A method takes the 'language' and the 'key' as arguments and returns the text phrase from the JSON table. Grammar of the language as a JSON document is shown below.

$$\begin{aligned} table &=< language^{+} > \\ language &=< id, title, kv^{+} > \\ kv &=< key, value > \end{aligned} \tag{2}$$

- **SMS-based transmission:** Since technological advancements happen slowly in the rural and remote areas, ICT-based healthcare services face many challenges. In our infrastructure, we intend to solve some of the challenges. One of the major constraints in rural areas is the non-availability of fast and reliable Internet connection. Hence, we use the idea of transmission of the patient's vitals to cloud when the Internet connection is poor or not available. A separate module is used for this purpose. After the collection of all patient details, the application tries to send the data over the Internet. If there is no Internet connection, the application accumulates patient's important vital data (symptoms) in a form of JSON string, compresses the string and sends it via SMS to an SMS server. The SMS server sends the compressed data to the cloud server. Another application running on the cloud decompresses the data and changes it to the original JSON format [5].
- **Handling pandemic situation:** KiORH is a cloud-based modular application. A new knowledge base can be added easily any time, and the application can be extended to handle the pandemic situation. We have enriched our knowledge base by adding Coronavirus (COVID-19) assessment scan-related questions and options, based on the guidelines from WHO [33] and MHFW, Govt. of India [34]. This assessment can help the health assistants to identify the COVID-affected patients and ask them for quarantine. Currently, only English language support is set up as shown in Figs. 6 and 7.

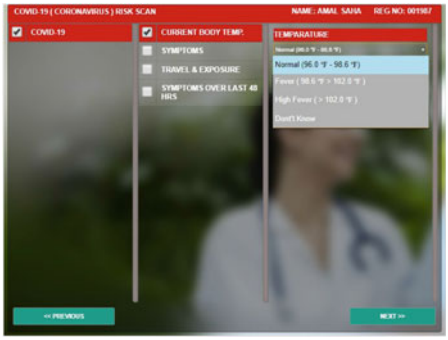


Fig. 6 COVID-19 assessment screen 1

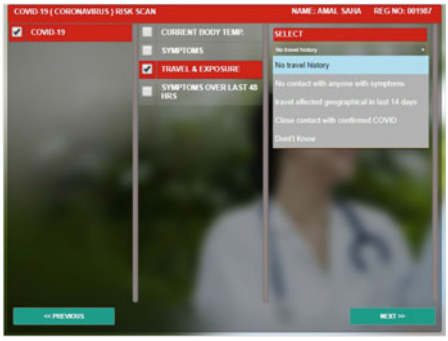


Fig. 7 COVID-19 assessment screen 2

4 Connecting Doctors

On the other side of the health kiosk, there is a pool of doctors having access to the other part of the application. When a health assistant connects a doctor from the pool, and a complaint is created, the selected doctor (selected at the kiosk side) is notified. The doctor can view the patient's complaints and all related information. Figure 8 shows a patient's complaint and its corresponding prescription as it appears on the web interface. A doctor can check the complaints along with the past records of that patient which are uploaded to the cloud by the health assistant. Medications, investigations and other advice can be prescribed. The web interface also provides WebRTC-based audio video chat facility, which can be used for communication between doctors, patients and health assistants. For the convenience of patients and the rural health assistants, questions associated are asked in local languages. However, on the doctors' screen, these questions are represented in the English language.

Key-value pairs in complaints questionnaires could be represented in a tabular form to the doctor. However, for convenience a string is generated out of the key-value pairs as shown in Fig. 9. Similarly, medicines and advice are entered in a generalized structure which is represented as a string in the prescription. The home page of

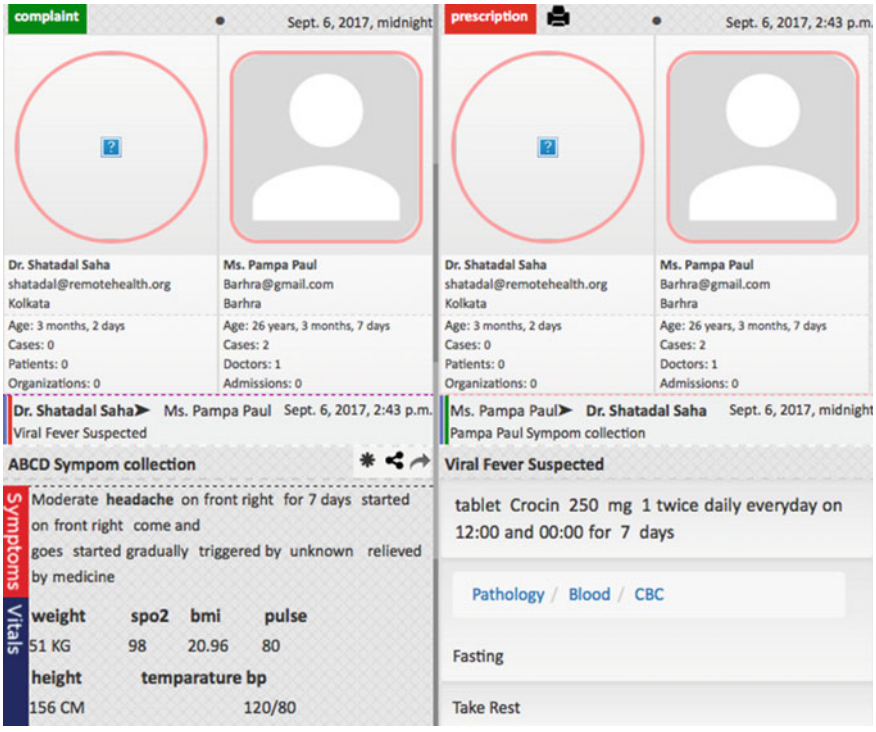


Fig. 8 Complaint screen

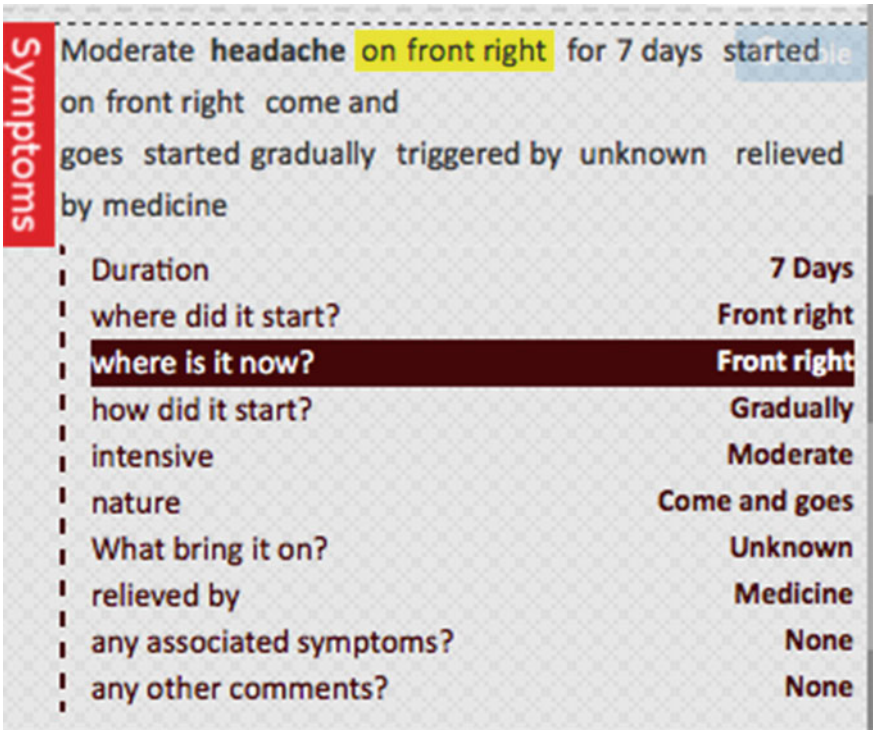


Fig. 9 Details of complaint

the doctor-side interface shows an overview of the recent admissions, complaints, prescriptions, patients and appointments. After selecting a patient, the complaints and prescriptions of that patient can be viewed by a registered doctor at any time anywhere.

4.1 Prescription Generation

Prescriptions at the doctor's end are composed in English. However, the medications and instructions for medications are shown using a bilingual interface to the patient. Doctors can also prescribe medical tests through the web interfaces as shown in Figs. 10 and 11. Medicines are filled up by the doctor using a web interface. Some of these fields (such as type, name, dose, unit, termination and count) are mandatory. For the remaining fields, a value may not be provided if that field is not applicable. If the mode of medication is clock-based, then a set of times are entered in the *clocks* field. If the mode of medication is event-based, a *when* field and a *context* field are specified (e.g. 'after' (when) and 'dinner' (context)).

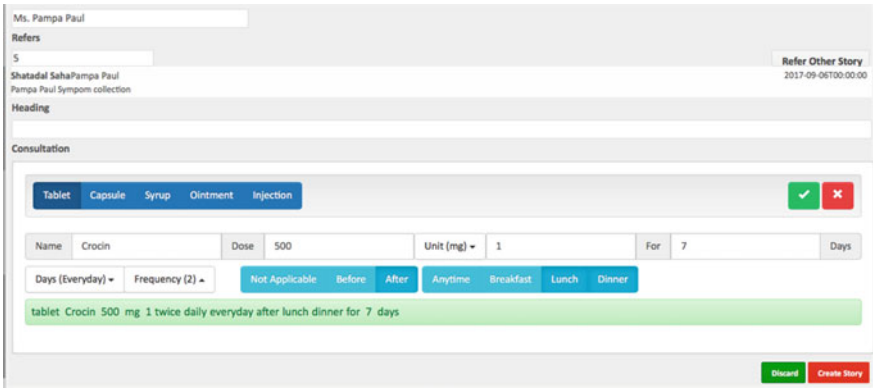


Fig. 10 Prescription composer: prescribing medicine

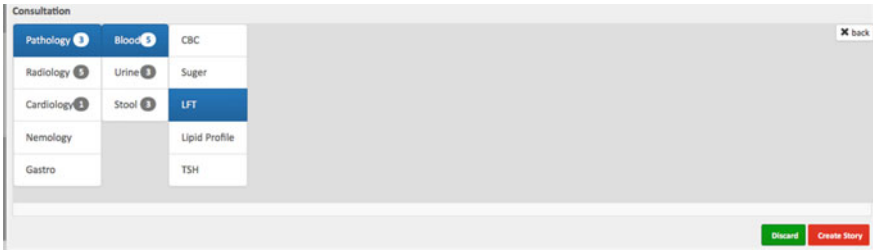


Fig. 11 Prescription composer: adding investigation

However, the prescription needs to be composed in languages which are understood by the patients as shown in Figs. 13 and 14. As there may be large variations in the grammar of two different languages (e.g. in case of English and Bengali), the composition of a string from the data entered in the prescriptions for every language should be done differently. A meaningful translation can be achieved by substituting the placeholders with values of the keys as mentioned in Fig. 12.

At the kiosk side, the advice and the prescribed medications can be printed in user-accessible language. Every prescription has a unique QR code printed on its top right corner which can be used for reference in future.

4.2 Interface Between Doctor, Health Assistant and Patient

Once a prescription is created by the doctor, health assistants can view and print the prescription. Before printing, health assistants may also discuss the prescription or request for a change of medicines, which may lead to the regeneration of the prescription. For additional discussions, doctors, patients and health assistants can

Fig. 12 Medicine definition

```
type: '', // tablet | capsule | ointment
name: '', // name of the medicine
dose: 0,  // dose
unit: '', // unit of dose
termination: 0, // for how many days
count: 1, // how many
interval: {
    frequency: '', // numeric
    days: 0, // days interval
    mode: '',    // clock or event
    clocks: [],
    event: {
        context: [], // Lunch | Dinner
        when: '' // before | after | N/A
    },
    custom: ''
},
note: '' // some extra notes if required
```

Fig. 13 Translation of prescription in English

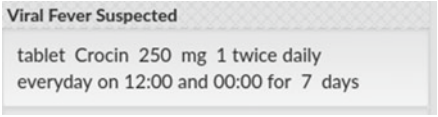
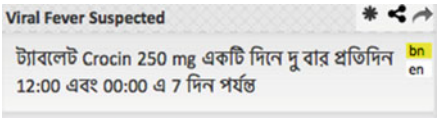


Fig. 14 Translation of prescription in Bengali



interact over live chat provided by the application. We have implemented peer-to-peer audio and video conferencing via WebRTC.

5 Assessment of Usefulness of KiORH

After 2 years of operation of the kiosk-based healthcare delivery system, a survey has been conducted among the rural population to access the acceptance of the KiORH application, in particular the approach taken by it and the sustainability of the kiosks of similar types. The survey has been conducted by the Department of Computer Science and Engineering and School of Mobile Computing and Communication, Jadavpur University, to understand the impact of the project. An independent organization was involved to conduct the survey. This survey purely stands on the basis of the perception of the villagers, i.e. the beneficiaries.

5.1 Survey Methodology

For the survey procedure of KiORH, two methods have been used. At first, villagers have been stratified in terms of 'Goers' and 'Non-goers' of the kiosk, i.e. the persons who have visited the kiosk at least once and the persons who have never visited. To evaluate the 'Goers' perception, 100 random samples were collected from the registered patient list, and face-to-face interviews were conducted using pre-tested, structured and close-ended questionnaires.

The perception of 'Non-goers' about KiORH was assessed by a random systematic method using an electoral roll of Barrah Village. Thus, as usual, 100 more samples were picked from the list by this method. The same face-to-face interactive interview process was conducted using a pre-tested, structured and close-ended questionnaire. In both cases, vernacular language for the questionnaire was used. The survey work was completed within seven consecutive days in December 2020 without compromising the pre-determined sample frame. After the completion of data collection, computation and analysis were performed by multi-stairs observation using the Statistical Package for Social Sciences (SPSS-15 version).

5.2 Village Description

A total of 2,223 families reside in the village Barrah and in its surrounding area. The total population of the village is 10,798 of which 5,578 are male and 5,220 are female. The total area of the village is nearly about 614.7 hectares [35]. The child population in the village aged 0–6 is 1,490. According to the survey, 99% of the respondents have their own house and they are living in the village since their ancestral time. Most of the houses are in poor condition indicating the economic condition of the people living in the village. The only primary healthcare center in the village had not been operational for a long time. Thus, the patients need to travel to the nearby towns. The nearest town (Dubrajpur) is 33 km away from this village and the district headquarter, Suri, is almost 50 km far from this area. The villagers are mostly engaged as daily laborers.

5.3 Drinking Water and Food Habit

At the time of investigation, it was noticed that 54% of the respondents depend on tube well as their source of water. In the summer season, groundwater level falls drastically. Then the villagers suffer most to get water from the tube wells. Only 23% households use tap water. To date (Dec'20), 17% of the respondents are dependent on open wells which creates an impact on the villagers' health. For this reason, water-borne diseases are significantly high in this area. 2% of the respondents depend on the

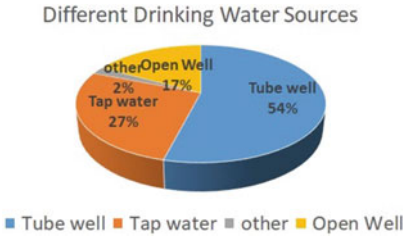


Fig. 15 Drinking water sources in the village

submersible motor pump for drinking water. The positive side is that the villagers do not need to go far for drinking water, i.e. at least 70% of the villagers get a source of drinking water within a 100-meter distance and among these 70%, 31% respondents have a source of drinking water within their own housing premises. Drinking water sources are represented in Fig. 15.

6 Survey Results

As the survey has been conducted by dividing the respondents into two groups—*health kiosk goers* and *non-goers*—at first, the analysis was made by age group. The age distribution of these two groups is shown in Fig. 16. According to the survey, 41% of kiosk goers are in the 35–50-year age group, 24% are in the 24–30-year age group and age of 27% of the population is more than 50 years. Only 8% falls below 20-year age group. On the other hand, in the case of non-goers the numbers are 32%, 42%, 21% and 5%, respectively. The results show that older persons (age > 35), particularly persons in the age group 35–50, have utilized the facility more than the younger persons.

6.1 Profile of Kiosk Goers

The first analysis was done based on the responses received from kiosk goers, the respondents who have already experienced the kiosk system at least once. The socio-

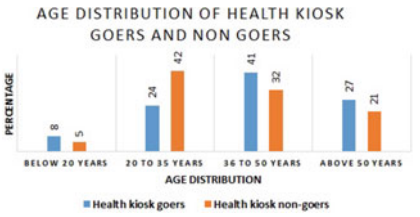


Fig. 16 Age group distribution: goers and non-goers to health kiosk

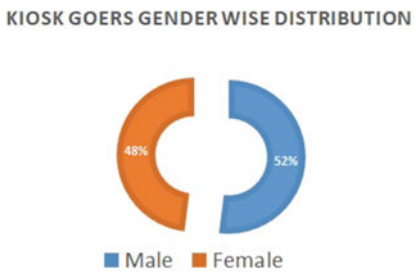


Fig. 17 Gender-wise distribution among the patients

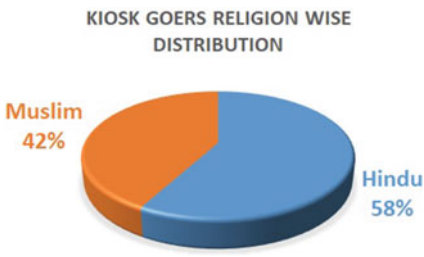


Fig. 18 Religion-wise distribution among the patients

economic profile of the *health kiosk goers* was analyzed first. The analysis was based on gender, religion, caste, education profile and income of the patient.

- **Gender Distribution:** Gender-wise distribution, shown in Fig. 17, displays the absolute representation in terms of census ratio of the village. The percentage of male patients is slightly high. No gender discrimination was noticed in the analysis.
- **Religion Distribution:** From a religion point of view, the health kiosk goers are distributed into two groups. Hindus represent 58%, followed by Muslims who are 42%. No patient belonging to other religions resides in the village. No religious constraints are noticed in the survey data. Figure 18 describes the religion distribution.
- **Education Profile:** While analyzing the education profile, it is noticed that among the kiosk goers, 65% are graduates. This may be the result of their awareness and hesitation-free attitude toward the tab-based online healthcare concept. On the other hand, 29% of the kiosk goers are non-literate. One reason for relying on a kiosk-based healthcare system in the case of these patients may be the geographical location of the kiosk and affordability. They depend on the online cloud-based kiosk system, because they have no provision to go to Dubrajpur or nearby town for treatment. A graphical representation is shown in Fig. 19.
- **Income Group of Patients:** When the patients are divided into income groups, it is observed that 72% of the patients come from low-income group. The rich are usually dependent on either homeopathic medicine or allopathic clinics in the nearby towns. A graphical representation of the distribution of the income groups is shown in Fig. 20.
- **Disease Patterns:** According to the survey findings, it is observed that most of the patients going to the kiosk suffer from chronic diseases. Almost 59% of the sample

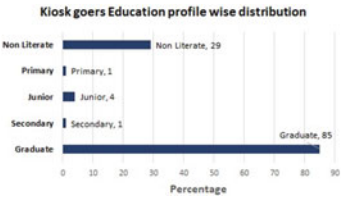


Fig. 19 Education profile of the sample patients

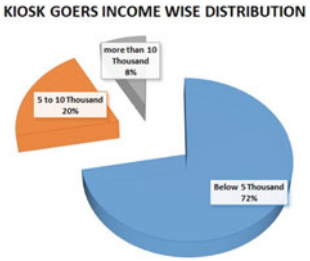


Fig. 20 Income-wise distribution among the patients

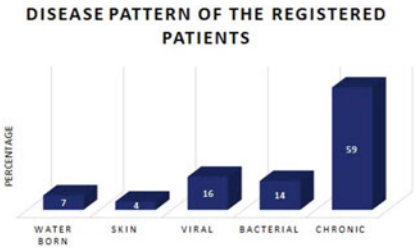


Fig. 21 Disease patterns of the patients

stated that they go to the kiosks for diseases like diabetes, colitis, etc. long-term diseases. Among the kiosk goers, only 16% and 14% stated that they visited the kiosk for viral and bacterial diseases. Figure 21 shows the distribution based on disease patterns.

On the other hand, from another survey record, it is observed that nearly 55% of the patients also depend on other clinics (Allopathic, Homeopathic) for viral and water-borne diseases. They do not have enough confidence to go to the health kiosk of this kind for day-to-day emergency medical needs, because basic emergency necessities like oxygen or saline water are not available in the kiosk. So most of the kiosk patients visit the kiosk for chronic diseases where there are no emergency needs.

6.2 Improvement in Patients' Health Condition

During the survey, patients were also asked about their experiences regarding improvement in their health conditions. From Fig. 22, it is observed that 53% of

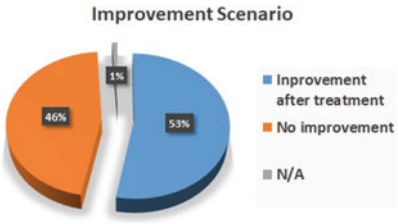


Fig. 22 Health condition improvement scenario

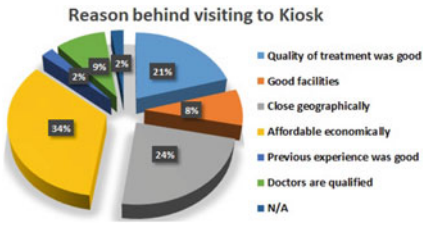


Fig. 23 Reasons behind visiting kiosk

the patients reported that their health condition improved after their treatment under the health kiosk system. This result is inspiring. However, the remaining 46% of the patients replied negatively when they were asked about the improvement in their health condition after visiting and following the procedure advised by the online doctors in the health kiosk.

The patients mostly agreed that the process followed in the health kiosk system is lengthy, but they did not have complaints about the treatment process. 78% realized that the additional time was necessary for this kind of treatment. At the time of consultation using KiORH, there is no provision to meet the doctors physically, however, the patient can speak with the doctor through video chat. Almost 76% of the patients had been able to speak with the doctors, and among them, around 85% are satisfied after the video chat. When asked about the reasons for visiting the health kiosk for treatment, the patients have clearly spoken out and mentioned the primary reasons which are shown in Fig. 23.

The health kiosk is geographically close to the villagers, thus, they (24% of patients) think it is beneficiary for them. 34% opine that it is economically affordable and therefore, they depend on it. These two major points came out as positive vibes about health kiosks and can significantly boost up kiosk-based primary health care in villages where communication and mainstream healthcare facility is poor. 8% of the patients visit the kiosk because the doctors appointed for the health kiosk are really good.

The above-mentioned satisfaction level is confirmed when it is noticed in Fig. 24. Among the patients, 15% are dissatisfied and said that the service and treatment are not helpful. When they were asked about the reason behind their dissatisfaction, it was found that these patients were mostly from the low-income group and they faced difficulties in buying medicines from the local medicine shop, because either the

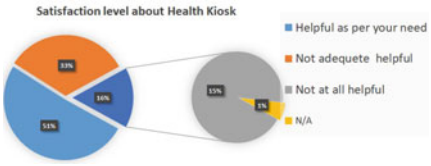


Fig. 24 Satisfaction level among kiosk goers

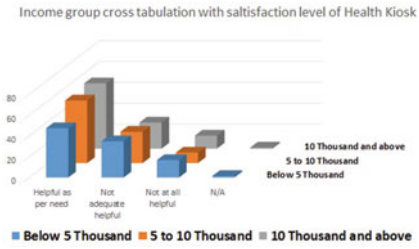


Fig. 25 Income group chart with satisfaction level of the kiosk goers

medicines were not available or expensive. The higher income group does not have much complaints about the medicine cost. Figure 25 shows the levels of satisfaction among different income groups.

6.3 Kiosk Non-goers

Next, the survey has been carried out by collecting samples from the villagers who never visited the kiosk. This part of the survey helped to identify the lacuna and drawbacks of the kiosk scheme. The sample may be considered as representative of the village population, but the nature of this set is uncontrolled.

The age group and gender distribution of the population were analyzed. Results are shown in Fig. 16 and in Fig. 26, respectively. Regarding gender distribution, it was noticed that the pattern is similar to Census 2011 data.

Measuring the general awareness level about the health kiosk among this section of the population has been another focus of this survey. It was found that 53% of the

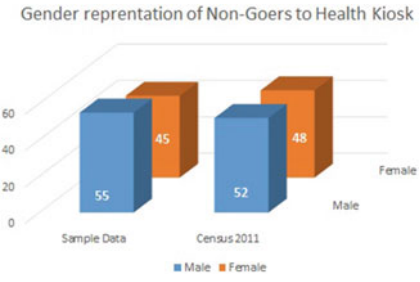


Fig. 26 Gender representation among kiosk non-goers

Fig. 27 Reasons for not going to the kiosk

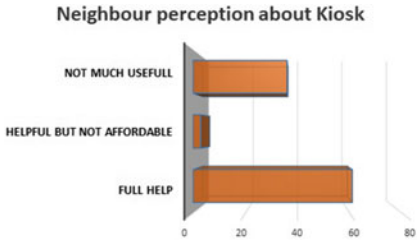


Fig. 28 Neighbors' perception about the health kiosk

villagers recognized the health kiosk. This set of the sample population was further asked why they did not go to the Health kiosk for treatment. Most of the people (almost 44%) said that they did not go there because doctors and nursing staff are not available. This kind of treatment, i.e. treatment without the physical presence of a doctor, could not earn the trust of these villagers. A sizable amount of respondents also complained about the lack of facilities which are found in any usual health center. These two major points helped to understand the views of the non-goer villagers. Other reasons are also shown in Fig. 27.

Apart from this, 52% of the respondents have the knowledge that their neighbors are going to the health kiosk for treatment and 61% stated that their neighbors' experiences were good and gave a positive feedback. Only 3% said that the kiosk is helpful, but not affordable for them. This result is shown in Fig. 28.

6.4 Reasons for Not Revisiting the Kiosk

Further study was made to understand the satisfaction level of the patients who visited the kiosk at least once for treatment. In this regard, the respondents' data was analyzed and it was noticed that 36% did not visit the kiosk for the second time for a checkup. When they were asked about the reasons, they gave different answers, which are shown in Fig. 29.

It has been found that the unavailability of the prescribed medicine was one of the strong reasons for not revisiting the kiosk. The patients who stated that the medicines are not available were cross-checked, and it was found that they consistently main-

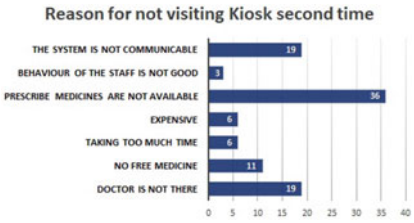


Fig. 29 Reasons behind not visiting the kiosk for the second time

tained their view in a revised questionnaire. Thus, it was concluded that scarcity or unavailability of medicine was one of the primary issues related to the performance of the kiosk-based treatment process.

7 Observations

A health kiosk in the villages in interior parts of India can open up a new window for public health, as well as rural employment. The online cloud-based health kiosk system shows a ray of hope to overcome the challenges due to the scarcity of doctors and related problems in the rural healthcare system. In particular, one such system becomes really helpful in a pandemic situation, like the ongoing COVID-19 pandemic. The observations gathered by setting up the health kiosk system in one village in West Bengal, India, and from the related survey are listed below.

- The prescribed medicines should be available in the local market or the kiosk needs to maintain its own pharmacy to stock the necessary medicines.
- Duration of doctors' online availability and its frequency should be increased.
- The sustainability of the fees should be calculated properly and moderately.
- Kiosk needs to be upgraded by an arrangement of saline, oxygen, urgent injections and medicines. In minor cases, an emergency may be mitigated at the village level, so that the patient does not need to go to the nearby towns.
- The physical presence of doctors should be arranged on a regular basis, at least once a month.
- Each patient needs to be contacted regularly by the health assistants either personally or using an automated reminder system.
- Multiple replicas of such a system should be set up and more canvassing is necessary for every nearby village.

Though the application has been deployed in a real-life scenario before the start of the recent COVID-19 pandemic, the application can be effectively used in this situation, for example, for remote screening, monitoring the pregnant women, providing care for the elderly people, tracking the travel history of the patients and providing appropriate advise to the villagers in case of exigencies. We are presently working toward this development.

8 Conclusion and Future Work

Our solution to remote healthcare delivery has been running in two health kiosks in the state of West Bengal, India. One kiosk has been set up in a village in Birbhum district which is 325 km away from the city Kolkata, and the other has been set up in an island of Sundarban. The fact that these two kiosks have successfully operated for 3 years provides sufficient evidence that such a system can be effective in the given situation. Although the application has been tested in the kiosk scenario, it can also be used when the patient is on the move which is similar to the proposal made in [36].

Many different adaptations, tests and experiments have been left for the future due to lack of time, as the experiments with real data are usually very time-consuming, requiring even days to finish a single run.

However, several challenges are yet to be considered for the efficient functioning of these kiosks. Presently, the existing technologies are being used in the KiORH application. There is a serious need to address the issues, such as heterogeneity of sensor devices and communication protocols, sharing the data streams among various applications on demand basis, and efficient storage and management of the data generated through this kiosk application, including streaming data. We need to ensure the accessibility of doctors and health assistants by employing efficient algorithms and business models. We are also looking into the issues related to efficient storage and retrieval techniques for medical documents.

Security and privacy are also important issues for this type of application. There is already a proposed algorithm to secure the gathered data and their storage integrating symmetric keys [37]. We are working on the security issue by adding a challenge-response mechanism or biometric option for the user.

One major problem is training health assistants. The knowledge base needs to be improved with the help of the medical practitioners and image-based assistance should be provided to the health assistants so that they can capture the symptoms with better accuracy. Learning techniques may also be integrated with the application. We are presently working toward the extension of the knowledge base to assist the health workers in the current scenario of the Corona pandemic.

Currently, KiORH only runs using two languages. It would be good to add more regional languages to make the application more popular and useful among the rural people of India.

We have integrated SMS-based transmission using Lempel-Ziv (LZ) lossless data compression algorithm. There is a scope to use a more complex algorithm to send more data through SMS or other media.

Finally, more involvement of the doctors is necessary. Often, doctors, even in urban locations, are not available due to their busy schedules. The doctors' pool should be further developed to get doctors always online.

Acknowledgements This work was financially supported by Information Technology Research Academy (ITRA), Ministry of Communications and Information Technology, Govt. of India.

References

1. Mondal S, Mukherjee N (2016) Mobile-assisted remote healthcare delivery. In: 2016 Fourth international conference on parallel, distributed and grid computing (PDGC). IEEE, pp 630–635
2. Yvonne Chan Y-F, Nagurka R, Bentley S, Ordonez E, Sproule W (2014) Medical utilization of kiosks in the delivery of patient education: a systematic review. Health Promot Perspect 4(1):1–8. https://doi.org/10.5681/hpp.2014.001
3. Verma A, Dhand H, Shaha A (2008) Healthcare kiosk next generation accessible healthcare solution. In: 10th International conference on e-health networking, applications and services (HealthCom 2008)
4. Radvan D, Wiggers J, Hazell T (2004) HEALTH C.H.I.P.s: opportunistic community use of computerized health information programs. Health Educ Res 19:581–590
5. Mukhopadhyay P, Ray HS, Mukherjee N (2019) E-healthcare delivery solution. In: 2019 11th international conference on communication systems & networks (COMSNETS), Bengaluru, India, pp 595–600. https://doi.org/10.1109/COMSNETS.2019.8711429
6. McGregor C, Kneale B, Tracy M (2007) On-demand Virtual Neonatal Intensive Care units supporting rural, remote and urban healthcare with Bush Babies Broadband. J Netw Comput Appl 30(4):1309–1323
7. Staff G (2016) 360 Roundup: 10 Indian healthcare startups you should know about. Retrieved from https://gadgets.ndtv.com/apps/features/roundup-10-indian-healthcare-startups-you-should-know-about-792075
8. Kashyaap S (2018) Kiosk by kiosk, this healthcare startup is transforming healthcare for rural India. https://yourstory.com/2018/06/kiosk-by-kiosk-this-healthcare-startups-is-transforming-healthcare-for-rural-india
9. Team EFY (2017) iKure Health Monitoring Kiosks. https://electronicsforu.com/india-corner/innovations-innovators/ikure-health-monitoring-kiosks
10. Network, eH (2018) eHealth Kiosks to Connect Rural Rajasthan. https://ehealth.eletsonline.com/2016/05/ehealth-kiosks-to-connect-rural-rajasthan/
11. Dr Lal PathLabs (n.d.) Download mobile app for Android|Dr Lal PathLabs. https://www.lalpathlabs.com/
12. Your Home for Health, Your Home for Health, Practo. www.practo.com/
13. credihealth: Aapka Health Partner (n.d.) https://www.credihealth.com/
14. Tweet2Health—Apps on Google Play (n.d.) https://play.google.com/store/apps/details?id=com.trainedge.doctor
15. Curofy—Discuss medical cases—Apps on Google Play (n.d.) https://play.google.com/store/apps/details?id=com.curofy
16. Singh B (2015) HealthOnPhone: a startup creating a secure cloud database of health records for anywhere access. http://www.iamwire.com/2015/05/healthonphone/116671
17. Indian Online Pharmacy|Buy Medicines Online, Fast Delivery. Netmeds.com, India Ki Online Pharmacy, www.netmeds.com/
18. Infermedica. Symptomate—Online Health Checkup, https://symptomate.com/diagnosis/
19. Medical calculators, equations, algorithms, and scores. MDCalc, www.mdcalc.com/
20. Prognosis: your diagnosis. Prognosis: your diagnosis, www.prognosisapp.com/
21. Dobson R (2003) Study reports on use of "touch screen" health kiosks. BMJ 326:184
22. Bosnic S, Papp I, Novak S (2016) The development of hybrid mobile applications with Apache Cordova. In: 24th telecommunications forum (TELFOR). Belgrade, pp 1–4. https://doi.org/10.1109/TELFOR.2016.7818919
23. Bose S, Mukherjee N (2016) SensIaas: a sensor-cloud infrastructure with sensor virtualization. In: 2016 IEEE 3rd international conference on cyber security and cloud computing (CSCloud). IEEE
24. Nicholas D, Huntington P, Williams P, Vickery P (2001) Health information: an evaluation of the use of touch screen kiosks in two hospitals. Health Info Libr J 18:213–219

25. Allenby A, Matthews J, Beresford J, McLachlan SA (2002) The application of computer touchscreen technology in screening for psychosocial distress in an ambulatory oncology setting. Eur J Cancer Care (Engl) 11:245–253
26. Peters J, Jackson M (2005) Accessibility and use of touchscreens by black and ethnic minority groups in the three cities project. Ethn Health 10:199–211
27. Sachdeva S, Mchome S, Bhalla S (2010) Web services security issues in healthcare applications. In: 2010 IEEE/ACIS 9th international conference on computer and information science
28. Ramirez M, Wu S, Jin H, Ell K, Gross-Schulman S, Sklaroff LM, Guterman J (2016) Automated remote monitoring of depression: acceptance among low-income patients in diabetes disease management. JMIR Mental Health 3(1). https://doi.org/10.2196/mental.4823
29. Cooking Hacks, e-Health Sensor Platform V2.0 for Arduino and Raspberry Pi [Biometric/Medical Applications]. https://www.cooking-hacks.com/documentation/tutorials/ehealth-biometric-sensor-platform-arduino-raspberry-pi-medical
30. Westman J, Hampel H, Bradley T (2000) Efficacy of a touchscreen computer based family cancer history questionnaire and subsequent cancer risk assessment. J Med Genet 37:354–360
31. Lindholm LH, Isacsson A, Slaug B, Möller TR (1998) Acceptance by Swedish users of a multimedia program for primary and secondary prevention of malignant melanoma. J Cancer Educ 13:207–212
32. Akbar F, Fernandez-Luque L (2016) What's in the store? a review of arabic medical and health apps in the app store. In: 2016 IEEE international conference on healthcare informatics (ICHI)
33. Technical guidance (n.d.) https://www.who.int/emergencies/diseases/novel-coronavirus-2019/technical-guidance
34. Ministry of Health and Family Welfare (n.d.) https://www.mohfw.gov.in/
35. District Census Handbook, Birbhum,Village And Town Directory. (2011) (XII-A). https://tinyurl.com/ww6zu3t
36. Batistatos MC, Tsoulos GV, Athanasiadou GE (2012) Mobile telemedicine for moving vehicle scenarios: wireless technology options and challenges. J Netw Comput Appl 35(3):1140–1150
37. Neogy S, Saha S (2015) Developing a secure remote patient monitoring system. In: International conference on cyber situational awareness, data analytics and assessment (CyberSA). London, pp 1–4

Author Index

M. Chaturvedi et al. (eds.), *Recent Advancements in ICT Infrastructure and Applications*, Studies in Infrastructure and Control, https://doi.org/10.1007/978-981-19-2374-6