ITU Journal
Future and evolving technologies

FREE I FAST I FOR ALL

Special issue

Internet of Everything

Volume 2 (2021), Issue 5

ITU Journal
Future and evolving technologies

Volume 2 (2021), Issue 5

Internet of Everything

ITU Journal
Future and evolving technologies

FREE I FAST I FOR ALL

The ITU Journal on Future and Evolving Technologies (ITU J-FET) is an international journal providing complete coverage of all communications and networking paradigms, free of charge for both readers and authors.

The ITU Journal considers yet-to-be-published papers addressing fundamental and applied research. It shares new techniques and concepts, analyses and tutorials, and learnings from experiments and physical and simulated test beds. It also discusses the implications of the latest research results for policy and regulation, legal frameworks, and the economy and society. This publication builds bridges between disciplines, connects theory with application, and stimulates international dialogue. Its interdisciplinary approach reflects ITU's comprehensive field of interest and explores the convergence of ICT with other disciplines.

The ITU Journal welcomes submissions at any time, on any topic within its scope.

Publication rights

SUGGESTED CITATION: ITU Journal on Future and Evolving Technologies, Volume 2 (2021), Issue 5.

ADDITIONAL INFORMATION

Please visit the ITU J-FET website at
https://www.itu.int/en/journal/j-fet/Pages/default.aspx

Inquiries should be addressed to
Alessia Magliarditi at: journal@itu.int

EDITORIAL BOARD

TABLE OF CONTENTS

LIST OF ABSTRACTS

Federated learning for IoE environments: A service provider revenue maximization framework

Pages 1-12

Benedetta Picano, Romano Fantacci, Tommaso Pecorella, Adnan Rashid

In accordance with the Internet of Everything (IoE) paradigm, millions of people and billions of devices are expected to be connected to each other, giving rise to an ever increasing demand for application services with a strict quality of service requirements. Therefore, service providers are dealing with the functional integration of the classical cloud computing architecture with edge computing networks. However, the intrinsic limited capacity of the edge computing nodes implies the need for proper virtual functions' allocations to improve user satisfaction and service fulfillment. In this sense, demand prediction is crucial in services management and exploitation. The main challenge here consists of the high variability of application requests that result in inaccurate forecasts. Federated learning has recently emerged as a solution to train mathematical learning models on the users' site. This paper investigates the application of federated learning to virtual functions demand prediction in IoE based edge cloud computing systems, to preserve the data security and maximise service provider revenue. Additionally, the paper proposes a virtual function placement based on the services demand prediction provided by the federated learning module. A matching based tasks allocation is proposed. Finally, numerical results validate the proposed approach, compared with a chaos theory prediction scheme.

View Article

IoE: Towards application-specific technology selection

Pages 13-27

Biswajit Paul, Gokul Chandra Biswas, Habib F. Rashvand

Determining the suitability of any technology for an Internet of Everything (IoE) application is essential in the presence of diverse technologies and application requirements. Some of the IoE applications include smart metering, wearables, healthcare, remote monitoring, inventory management and industrial automation. Energy efficiency, scalability, security, low-cost deployment and network coverage are some of the requirements that vary from one application to another. Wireless technologies such as WiFi, ZigBee, Bluetooth, LTE, NB-IoT, LoRa and SigFox will play crucial roles in enabling these applications. Some of the technological features are transmission range, bandwidth, data rate, security schemes and infrastructure requirements. As there is no one-size-fits-all network solution available, the key is to understand the diverse requirements of different IoE applications and specific features offered by different IoE enabling technologies. Application-specific technology selection will ensure the best possible utilization of any technology and the quality of service requirements. An overview of network performance expectations from various IoE applications and enabling technologies, their features and potential applications are presented in this paper.

View Article

From design to prototyping in the Internet of Things: A domotics case study

Pages 29-37

Sabrina Sicari, Alessandra Rizzardi, Alberto Coen-Porisini

Nowadays, the capability of rapidly designing and prototyping, simple, yet real domotics systems (e.g., smart homes and smart buildings applications) is even more compelling, due to the availability and increasing spread of Internet of Things (IoT) devices. Home automation services enable the remote monitoring of indoor environments and facilities. The main advantages include saving energy consumption and improving the overall management (and users' experience) in certain application domains. The pervasive adoption and diffusion of such remote monitoring solutions is hampered by the timing required for design, prototyping and further developing applications and underlying architecture, which must be often customized on the basis of specific domains' needs and involved entities. To cope with this issue, the paper proposes the analysis and prototyping of a domotics case study, in order to demonstrate the effectiveness of proper IoT-related tools in speeding up the testing phase.

View Article

RF-based low-SNR classification of UAVs using convolutional neural networks

Pages 39-52

Ender Ozturk, Fatih Erden, Ismail Güvenç

Unmanned Aerial Vehicles (UAVs), or drones, which can be considered as a coverage extender for Internet of Everything (IoE), have drawn high attention recently. The proliferation of drones will raise privacy and security concerns in public. This paper investigates the problem of classification of drones from Radio Frequency (RF) fingerprints at the low Signal-to-Noise Ratio (SNR) regime. We use Convolutional Neural Networks (CNNs) trained with both RF time-series images and the spectrograms of 15 different off-the-shelf drone controller RF signals. When using time-series signal images, the CNN extracts features from the signal transient and envelope. As the SNR decreases, this approach fails dramatically because the information in the transient is lost in the noise, and the envelope is distorted heavily. In contrast to time-series representation of the RF signals, with spectrograms, it is possible to focus only on the desired frequency interval, i.e., 2.4 GHz ISM band, and filter out any other signal component outside of this band. These advantages provide a notable performance improvement over the time-series signals-based methods. To further increase the classification accuracy of the spectrogram-based CNN, we denoise the spectrogram images by truncating them to a limited spectral density interval. Creating a single model using spectrogram images of noisy signals and tuning the CNN model parameters, we achieve a classification accuracy varying from 92% to 100% for an SNR range from -10 dB to 30 dB, which significantly outperforms the existing approaches to our best knowledge.

View Article

3-of-3 multisignature approach for enabling lightning network micro-payments on IoT devices

Pages 53-67

Ahmet Kurt, Suat Mercan, Enes Erdin, Kemal Akkaya

Bitcoin's success as a cryptocurrency enabled it to penetrate into many daily life transactions. Its problems regarding the transaction fees and long validation times are addressed through an innovative concept called the Lightning Network (LN) which works on top of Bitcoin by leveraging off-chain transactions. This made Bitcoin an attractive micropayment solution that can also be used within certain IoT applications (e.g., toll payments) since it eliminates the need for traditional centralized payment systems. Nevertheless, it is not possible to run LN and Bitcoin on resource-constrained IoT devices due to their storage, memory, and processing requirements. Therefore, in this paper, we propose an efficient and secure protocol that enables an IoT device to use LN's functions through a gateway LN node even if it is not trusted. The idea is to involve the IoT device only in signing operations, which is possible by replacing LN's original 2-of-2 multisignature channels with 3-of-3 multisignature channels. Once the gateway is delegated to open a channel for the IoT device in a secure manner, our protocol enforces the gateway to request the IoT device's cryptographic signature for all further operations on the channel such as sending payments or closing the channel. LN's Bitcoin transactions are revised to incorporate the 3-of-3 multisignature channels. In addition, we propose other changes to protect the IoT device's funds from getting stolen in possible revoked state broadcast attempts. We evaluated the proposed protocol using a Raspberry Pi considering a toll payment scenario. Our results show that timely payments can be sent and the computational and communication delays associated with the protocol are negligible.

View Article

SIoT for cognitive logistics: Leveraging the social graph of digital twins for effective operations on real-time events

Pages 69-79

Miha Cimperman, Angela Dimitriou, Kostas Kalaboukas, Aziz S. Mousas, Salvatore Quattropani

Over the years, with the migration of organizations towards the concepts of logistics 4.0, a paradigm shift was necessary to guarantee logistics efficiency. The challenge is to dynamically cope in real time with vast number of shipments and destinations, which need to be realigned both with a determined lead time and with a finite of available resources. Although a number of standards have already been adopted for the management of transport and logistics operations, taking advantage, for instance, of Decision Support Systems and Geographic Information Systems, new models are required for achieving effective handling of the dynamic logistics environment that is shaped today. In this paper, an integrated logistics framework addressing the previous challenges is presented, for the first time, as a result of the activities of the H2020 COG-LO project. This novel approach exploits Social Internet of Things (SIoT) and the digital twins technique to realize the concept of the Cognitive Logistics Object (CLO). A CLO is defined as an entity that is augmented with cognitive capabilities, it is autonomous, and bears social-like capabilities, which enable the formulation of ad hoc communities for negotiating optimal solutions in logistics operations.

View Article

Lysis chatbot: A virtual assistant for IoT platforms

Pages 81-91
Raimondo Cossu, Roberto Girau, Luigi Atzori

The configuration and management of devices and applications in Internet of Things (IoT) platforms may be very complicated for a user, which may limit the usage of relevant functionalities and which does not allow its full potential to be exploited. To address this issue, in this paper we present a new chatbot which is intended to assist the user in interacting with an IoT platform and allow them to use and exploit its full potential. The requirements for a user-centric design of the chatbot are first analyzed, then a proper solution is designed which exploits a serverless approach and makes extensive use of Artificial Intelligence (AI) tools. The developed chatbot is integrated with Telegram to message between the user and the Lysis IoT platform. The performance of the developed chatbot is analyzed to assess its effectiveness when accessing the platform, set the main devices' parameters and request data of interest.

View Article

Resource tokenization for crowdfunding of wireless networks

Pages 93-100
Volkan Sevindik

This paper presents a novel blockchain-based spectrum tokenization method used to crowdsource wireless network deployment projects. Crowdsourcing is a method of financing certain projects and ideas through the funds collected by individuals or businesses in an open marketplace. The method presented in this paper finances the wireless network deployment projects belonging to service providers or governments. The method tokenizes proposed novel wireless resource units, and sells these units to investors. A new Value Unit Per User (VUPU) resource unit is introduced with a new pricing scheme depending on a load of a base station. A novel Proof of Data Load (PoDLO) consensus algorithm is proposed which is used to verify data and traffic load of a base station. Device Diversity Factor (DDF) and Subscriber Unique Permanent Identifier (SUPI) Factor (SUF) are proposed new ways to determine the value of a base station and a network cluster.

View Article

FEDERATED LEARNING FOR IOE ENVIRONMENTS: A SERVICE PROVIDER REVENUE MAXIMIZATION FRAMEWORK

Benedetta Picano[1], Romano Fantacci[1], Tommaso Pecorella[1], Adnan Rashid[1]
[1]Dpt. Information Engineering, Università di Firenze, Italy

NOTE: Corresponding author: Benedetta Picano, benedetta.picano@unifi.it

Abstract *– In accordance with the Internet of Everything (IoE) paradigm, millions of people and billions of devices are expected to be connected to each other, giving rise to an ever increasing demand for application services with a strict quality of service requirements. Therefore, service providers are dealing with the functional integration of the classical cloud computing architecture with edge computing networks. However, the intrinsic limited capacity of the edge computing nodes implies the need for proper virtual functions' allocations to improve user satisfaction and service fulfillment. In this sense, demand prediction is crucial in services management and exploitation. The main challenge here consists of the high variability of application requests that result in inaccurate forecasts. Federated learning has recently emerged as a solution to train mathematical learning models on the users' site. This paper investigates the application of federated learning to virtual functions demand prediction in IoE based edge-cloud computing systems, to preserve the data security and maximise service provider revenue. Additionally, the paper proposes a virtual function placement based on the services demand prediction provided by the federated learning module. A matching-based tasks allocation is proposed. Finally, numerical results validate the proposed approach, compared with a chaos theory prediction scheme.*

Keywords – Edge computing, federated learning, Internet of Everything, matching theory, revenue maximization, virtual function placement

1. INTRODUCTION

The emergence of new network paradigms such as Edge Computing (EC) [1, 2, 3, 4], for which the limitations typical of the cloud architecture have been bypassed moving computation nodes to the network edges close to the end users, has given rise to a wide range of challenges in many research areas [5, 6]. Consequently, several new issues, such as user mobility, heterogeneity in Quality of Service (QoS) or service requirements, massive volume of data, user privacy, diversity on data types and so on, have led to numerous efforts from both academia and industry in providing highly effective and efficient solutions [7, 8, 9, 10, 11]. In particular, there exists a significant branch of literature regarding possible solutions to improve EC Network (ECN) performance in order to guarantee a high level of user satisfaction and to provide dynamic and flexible network resource allocation and decision-making strategies. Within this context, the Internet of Everything (IoE) paradigm, in which people, process, data, and things are connected and exchange data,has given rise to systems with increasing complexity and applications involving strict real-time requirements and sensitive data [12], heterogeneous traffic. Generally speaking, heterogeneity in data flow types implies different QoS or service requirements. Furthermore, from a Service Provider (SP) perspective, such diversity triggers new data flow management policies, service provision costs and selling prices. In this respect, the SP revenue maximization is strictly related to the adopted management and administration policy.

Indeed, a proper resource exploitation planning is essential to guarantee elevated levels of network efficiency, user satisfaction and consequent high SP revenues, as highlighted by literature such as [13], [14]. In particular, having an a priori knowledge about the data flow service demand can be properly exploited to perform suitable resource infrastructure planning with maximum income. In order to pursue this objective, Machine Learning (ML) [15, 16, 17, 18, 19] has emerged by providing many techniques to perform data behavior interpretation and analysis. The ability of ML techniques in catching data trends, patterns and hidden features, has ensured its applicability to many problems. However, although the knowledge and extrapolation of user data characteristics positively impacts many application areas, it may result in being non-compliant with some specific user privacy constraints [20]. In this respect, if on the one hand the users' data analysis may lead to remarkable advantages in reference to the network resources planning and exploitation, on the other the user data gathering may trigger user dissention, due to privacy concerns and violation. Within this context, a data-manipulation framework able to collect users' data without contravening users' privacy is a priority. In this respect, Federated Learning (FL) [21, 20, 8, 22, 23, 24] has recently emerged as a promising tool to perform, locally on the users' devices, statistical and mathematical training models based on ML methodologies without losing users privacy constraints. The FL framework consists of the devices level, generally indicated in literature as clients, and a central server unit

which aggregates and merges the data preliminary processed by the clients. Typically, FL has the following matters to face with [25]

- **Non-Independent Identically Distributed Data** The clients have different training datasets, therefore a single dataset cannot be considered representative of the other clients datasets;
- **Unbalanced Datasets** Different clients have different datasets, and each dataset may have a diverse number of elements in comparison to other clients datasets;
- **Large-Scale Distribution** The number of clients involved in the FL training procedure is generally higher than the amount of data processed at the client level;
- **Limited Communication** Mobile devices may or may not be available for data training and the computational capability or communication conditions could be poor.

In reference to the proposed contextualization, we have assumed here that sensitive user data may be derived from historical users functions utilization. In this perspective, sharing data about daily users habits may expose the users to undue risks. For this reason, the FL framework may represent a useful tool to counteract such a problem. However, a deep investigation of the privacy issues are out of the scope of this paper. The paper proposes the application of the FL framework, in order to forecast the service demands, without losing the user privacy constraints, in an IoE scenario. Moreover, on the basis of service demand forecasting, this paper proposes a suitable Virtual Functions (VFs) placement both on the ECN and cloud. Summarizing, the contributions of this paper are

- Application of the FL strategy to forecast the network VFs demand, in order to take into account the users privacy;
- Formulation of the SP maximum revenue problem, by considering Service Requests (SRs) with a different priority and hence, different cost and price. In particular, the SP can accept the data SRs with low priority if all the high priority flows have been satisfied;
- Proposal of a VFs placement strategy and a suitable matching-based SRs allocation algorithm based on the considered FL and the previously provided VFs forecasting scheme;
- Performance evaluation of the proposed approach and the comparisons with a centralized Chaos Theory (CT)-based prediction scheme, by resorting to extensive computer simulation runs.

The rest of paper is organized as follows. In Section 2 an in-depth review of the related literature is presented. Section 3, discusses the problem statement, while in Section 4 the FL framework and the placement strategy are presented. Then, in Section 5 the experimental results are analyzed and the alternative CT predictive approach explained. Finally, the conclusions are presented in Section 6.

2. RELATED WORKS

Recently, ML techniques have found extensive applications in big data analysis in fog/edge networks research area.

An overview of the ML techniques applied to fog is presented in paper [26]. Then, paper [26] investigates the ability of the ML strategies in detecting malicious attackers in fog networks, while paper [27] focuses on the ML solutions to evaluate the advantages deriving from an edge caching solution, taking into account user satisfaction perspective and energy efficiency. The improvement in sensing reliability and network latency is the aim of paper [28], in which the authors implement a multi-hidden multi-layer convolutional neural network solution to provide data authentication in a mobile crowd-sensing environment. The tree decisions strategy combined with the k-nearest neighbors method is applied in [29], in which authors deal with the position-based confidentiality problem in high real-time industrial application scenarios.

In a different way, SP maximization is the objective of paper [30], in which a deep supervised learning approach is applied to perform the minimization of the total network cost. A fog blockchain network is analyzed in paper [31], which formulates a solution based on the auction theory, where deep learning is applied to the maximization of the edge computing SP revenue.

Additionally, distributed ML is adopted in papers [32, 33, 34, 35]. In paper [32], a distributed version of the well-known support vector machine method is implemented to investigate its applicability. The reinforcement learning, and more in depth the Q-learning algorithm, is applied in paper [33], in order to minimize the users' outage in heterogeneous cellular networks scenarios. The control in crowd-sensing problem is the main objective of paper [34], exploiting the human in the loop methodology to propose a hierarchical crowd sensing framework with the aim of reducing cloud congestion and promoting the balancing of the data traffic. Then, the distributed stochastic variance reduced gradient is applied in paper [35], in which a target accuracy is fixed, and the optimization of the number of collection points to make data analysis provided. Furthermore, paper [35] proposes the minimization of the amount of network traffic sent towards the collection points. In a different way, the maximization of SP profit in a Mobile Edge Computing (MEC) blockchain network has been studied in paper [31], in which an auction strategy combined with deep learning is formulated to

perform edge resource allocation. Similarly, the auction theory is also applied to the profit maximization profit in [36], in which a novel combined optimal pricing and data allocation problem is solved with the Bayesian auction approach. The profit maximization in the cognitive virtual operator is addressed in paper [37], in which a dynamic network scenario is considered. Paper [37] develops a low complexity online control scheme to perform decisions about price and resource planning. A cloud allocation scheme for three classes of virtual machines is presented in [38], with the aim of maximising cloud provider profit.

Recently, FL has gained attention and papers [8, 39, 40, 41, 42, 20] provide its application to different contexts and situations. Paper [8] and paper [39] contextualize the FL in MEC networks, optimizing with the distributed gradient descent method the trade-off between local updates and global aggregations, formulating a loss function minimization problem, and introducing some resource constraints. Papers [8] maximize the number of clients involved in the aggregation process, aiming at minimizing the aggregation error. The MEC scenario is taken into account also in paper [39] which addresses the popularity content caching problem throughout the adoption of the hybrid filtering on stacked encoders to forecast content requests trend. Authors in [40] exploit the signal superposition property of wireless channels on the basis of which a novel aggregation data strategy for the over-the-air computation is presented. Furthermore, the model proposed in [20] is applied in [20] with the stochastic gradient descent algorithm as optimizer, aiming at training data in a distributed fashion by limiting the communication costs. The multi-task learning problem is solved with the FL and the novel Mocha context-aware optimization algorithm is presented in paper [22], while a blockchained FL architecture is proposed in [41]. Then, this architecture is designed to implement a distributed consensus strategy, by taking into account the blockchain end-to-end delay. Finally, a hybrid IoT-MEC network is considered for the application of FL in [42]. Paper [42] provides transmission and computational costs optimization, applying multiple deep reinforcement learning agents. Authors in [43] propose a QoE-driven delivery approach, in which there is cooperation between the Over-The-Top and Internet service providers, aiming at maximizing the revenue. Similarly, paper [44] addresses the economic aspects of a collaborative services management between Over-The-Top and Internet service providers. Consequently, authors propose an architecture to realize their collaboration, defining three different approaches on the basis of which the profit maximization of different customers is pursued. Then, the main objective of paper [45], is the investigation of the management procedures for multimedia services, proposing a collaborative zero-rated QoE approach to model the close cooperation between mobile network operators and the Over-The-Top service providers.

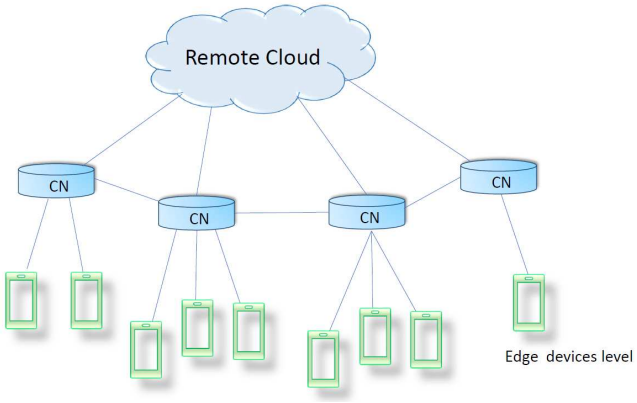


Fig. 1 – Hybrid cloud-fog network architecture

As summarized in Table 1, in contrast to papers [36, 37, 38], which provide profit maximization solutions without taking into account user privacy issues, we propose a revenue maximization framework based on data information elaborated locally on the users' devices, avoiding the typical privacy concerns of the other approaches. Hence, as in papers [40, 8, 39, 20, 41, 42], we propose an FL-based framework by using the gradient descent algorithm as optimizer. The motivation for this conservative choice resides in the fact that more complex methods may result in prohibitive consumption of the End Users' (EU) hardware resources, which is a crucial point in the distributed data training problems. Furthermore, in contrast to the previous up-to-date works, this paper contextualizes the application of the FL to the VFs deployment problem, by exploiting the FL framework to properly predict the application network demand, in order to maximize the SP revenue. Furthermore, a VFs placement and an SRs service allocation is provided to evaluate the actual validity of the proposed solution. In fact, the SRs service allocation algorithm, based on the matching theory, does not take into account the SP perspective, but only the users, i.e., the SRs, interests. Finally, to the best of our knowledge, this is the first paper which applies the FL to the SP revenue maximization problem, by considering even the users' perspective. The proposed approach performance has been evaluated by resorting to extensive numerical simulation and by providing comparison with the centralized CT-based predictive method.

3. PROBLEM STATEMENT

As an IoE reference scenario, we consider a single SP featuring an ECN constituted by $\mathcal{N}$ Computation Nodes (CNs) located at the network edges, and a more powerful cloud located far from the ECN. We suppose that all the CNs are equipped with a Central Processing Unit (CPU) with the same computational capability and number of available Storage Resource Blocks (SRBs) S. In a different way, the cloud is assumed to have a storage capacity of U SRBs, with $S < U$. In addition, we assume the availability of high speed wired links between CNs and from

Table 1 – Literature contributions

Standard Literature	Paper contribution
[36, 37, 38]	Proposal of a revenue maximization framework based on data information elaborated locally on the users' devices, avoiding the typical privacy concerns of the other approaches.
[40, 8, 39, 20, 41, 42]	Contextualization of the application of the FL to the VFs deployment problem, by exploiting the FL framework to properly predict the application network demand, in order to maximize the SP revenue.

Table 2 – Main symbols

Notation	Description
CN	Computation node
VF	Virtual function
FL	Federated learning
SRB	Storage resource block
SR	Service request
S	Number of SRBs per CN
U	Cloud SRBs
ECN	Edge computing network
$\mathcal{T}$	High priority requests
$\mathcal{M}$	Low priority requests
τ_i	Time deadline
x_i	Number of req. demanding for service i
y_j	Number of req. demanding for service j
$\mathcal{X}(x_i, q_i)$	SP revenue for the high priority req.
$\mathcal{Y}(y_j, z_j)$	SP revenue for the low priority req.
T_r	Service accomplishment time
$\omega_{z,h}$	Waiting time on the CN
$\omega_{z,C}$	Waiting time on the cloud

any CN to the cloud[1]. Furthermore, we guess that the ECN is able to support $\mathcal{T}$ different high priority service types, which are characterized by different provision costs and selling prices. Each service type $i \in \mathcal{T}$ has associated a QoS level expressed as a time deadline τ_i before which the type i service accomplishment has to be completed. In addition, we consider the presence of $\mathcal{M}$ service type requests with lower priority and without any time deadline constraint. The number of requests belonging to this class is indicated hereafter with y_j, with $j \in \mathcal{M}$.
Periodically, the SP updates the service demand and we assume that any new request does not arrive between two SP updates.
Let x_i be the number of SRs demanding for service i. We suppose that each SR is originated by an EU), and that an EU requires only one SR. Therefore, as a direct consequence, hereafter we assume interchangeable the SR and EU terms. Then, as regards the SP, the provision of a service has a cost mainly depending on x_i and following the model given by [46]

$$c(x_i) = \begin{cases} 0, & x_i = 0, \\ \beta_{c,i} + \beta_{l,i}\mu_i^{x_i}, & x_i > 0, \end{cases} \quad (1)$$

in which $\beta_{c,i}, \beta_{l,i}, \mu_i$ are real valued parameters whose value changes on the basis of the request type.
Similarly, the provision cost for providing y_j SRs of type j follows the rule [46]

$$b(y_j) = \begin{cases} 0, & y_j = 0, \\ \alpha_{c,j} + \alpha_{l,j}\nu_j^{y_j}, & y_j > 0, \end{cases} \quad (2)$$

where $\alpha_{c,j}, \alpha_{l,j}, \nu_j$ are, also in this case, real valued parameters.
Moreover, for each service type with high priority, the SP revenue results ruled by the following relation

$$U(x_i, q_i) = \frac{\log(1 + x_i)}{q_i}, \quad (3)$$

with $q_i = |x_i - k_i|$, where k_i is the number of SRs for which τ_i has been respected. Then, the SP revenue for the low priority SRs is given by

$$U(y_j, z_j) = \frac{\log(1 + y_j)}{z_i}, \quad (4)$$

where z_j is the number of SRs among y_j accepted by the network for their service. Hence, the SP revenue, corresponding to the provision of the i-th and the j-th service type, can be expressed as

$$\mathcal{X}(x_i, q_i) = U(x_i, q_i) - c(x_i), \quad (5)$$

and

$$\mathcal{Y}(y_j, z_j) = U(y_j, z_j) - b(y_j), \quad (6)$$

respectively.
Both the SRs with high and low priority, in order to be accomplished, require the presence of a VF in set $\mathcal{V}$ which has to be preliminary loaded on at least one CN of the network or on the far cloud. The loading process requires the CN or cloud availability in terms of SRBs, since each VF $v \in \mathcal{V}$ requires a number a_v of SRBs, different for each VF. Consequently, the time required for the service accomplishment (TSA) of a generic SR r, independently by its priority, is given by

[1] We have assumed that the connection towards the cloud is performed throughout the CN nearest to the SRs needing computation. Consequently, the communication latency cost between SRs and their nearest CN has no impact on the overall SR completion time and hence it has been neglected in defining *(7)*.

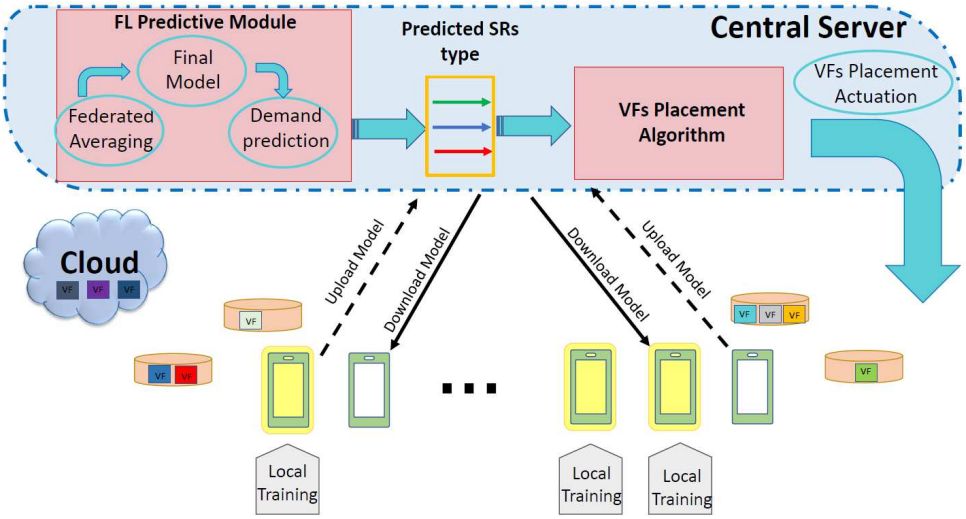


Fig. 2 – FL framework for the VFs placement

$$T_r = \sum_{v\in\mathcal{V}}\sum_{h\in\mathcal{N}}(\gamma_z+\omega_{z,h})\rho_{r,h}\theta_{v,h}+(1-\rho_{r,h})\zeta_{v,C}(\gamma_C+\omega_{z,C}), \tag{7}$$

where γ_z and γ_C are the execution time spent by the SR z on the CPU of a CN and of the cloud, respectively. It is important to note that both the execution times γ_z and γ_C mainly depend on the size of the SR z, the CPU frequency of the node hosting its elaboration, and the time spent by the SR on that node waiting for the actual computation. Therefore, $\omega_{z,h}$ and $\omega_{z,C}$ represent the queuing time experienced by the the SR z waiting for its execution on the CN h and cloud, respectively[2]. Furthermore, $\rho_{r,h}$ is a binary value equal to 1 if the SR j is executed on the CN h, 0 otherwise. Similarly, $\theta_{v,h}$ is equal to 1 when the VF v is present on CN h, 0 otherwise. Finally, $\zeta_{v,C}$ is equal to 1 if the VF v is loaded on cloud, 0 otherwise. It is important to make evident that the TSA in *(7)* strongly depends on the queuing time experienced by the SR on the service provision site. In fact, a proper deployment of VFs on the ECN may drastically reduce the TSA time.

In formal terms, the aim of this paper is the maximization of the SP revenue by providing decision making on the VFs placement, in order to satisfy the SRs. Therefore, the main goal of the paper is given by

$$\min_{\mathbf{q},\mathbf{z}} \sum_{i=1,\dots,\mathcal{T}} \mathcal{X}(x_i,q_i) + \sum_{j=1,\dots,\mathcal{M}} \mathcal{Y}(y_j,z_j), \tag{8}$$

s.t.

$$T_i \le \tau_i, \forall i = 1,\dots,\mathcal{T}, \tag{9}$$

$$\sum_{v\in\mathcal{V}} \theta_{v,h}a_v \le S, \forall h \in \mathcal{N}, \tag{10}$$

$$\sum_{v\in\mathcal{V}} \zeta_{v,C}a_v \le U. \tag{11}$$

[2]The CPU queue has been modeled with the first-in-first-out service policy.

In problems *(4)-(10)*, constraint *(9)* expresses the fact that each SR with a high priority has to be served, while constraints *(10)* and *(11)* represent that the VFs allocation has to respect the storage limit of CNs and cloud, respectively.

4. FEDERATED LEARNING FRAMEWORK

4.1 The learning problem

The aim of ML is the exploitation of some data used for training, to learn models. In order to do that, typically, ML involves the definition of a loss function representing the error implicitly resulting from the model training [8]. The loss function depends on the data sample z and a parameter vector $\mathbf{w}$, and it is named hereafter as $f_z(\mathbf{w})$. As previously introduced, this paper supposes the presence of L SRs, with $L = \mathcal{T} + \mathcal{M}$, deriving from an underlying level of EUs, each of which disposes of a local dataset Θ_l, $l = 1,\dots,L$. Therefore, as assumed in [8, 20], we suppose the collective loss function equals to

$$F_l(\mathbf{w}) = \frac{1}{|\Theta_l|}\sum_{z\in\Gamma_l} f_z(\mathbf{w}), \tag{12}$$

where $|\Gamma_l|$ is the number of elements belonging to Γ_l, referred as the cardinality of the Γ_l set. Respectively, the global function evaluated at the central server site, the global loss function, based on the distributed local dataset Θ_l and defined as [8, 20], is expressed by the following relation

$$F(\mathbf{w}) = \frac{\sum_{l=1,\dots,L}|\Theta_l|F_l(\mathbf{w})}{\sum_{l=1,\dots,L}|\Theta_l|}. \tag{13}$$

Therefore, the objective here is to find $\mathbf{w}^\star$ such that [8]

$$\mathbf{w}^\star = \text{argmin}\, F(\mathbf{w}). \tag{14}$$

Accordingly, with numerous contemporary papers [20, 8] recently proposed in literature, the optimization of *(14)* limiting the computational complexity, is pursued by applying the gradient descent method.

4.2 Federated learning framework

Algorithm 1 Client Side

for each NE involved in learning process **do**
    update $\mathbf{w}_\chi(u) = \hat{\mathbf{w}}_\chi(u-1) - \xi \nabla F_\chi(\hat{\mathbf{w}}_\chi(u-1))$;
    return $\mathbf{w}_\chi(u)$ to the central server;
end for

Algorithm 2 Server Side

initialize $\mathbf{w}_0$;
for each NE involved in learning process **in parallel do**
    Receive and update $\mathbf{w}_\chi(u)$
end for
update global model $\mathbf{w}(u) = \frac{\sum_{\chi \in \mathcal{K}} |\Theta_\chi| \mathbf{w}_\chi}{\sum_{\chi \in \mathcal{K}} |\Theta_\chi|}$.

Algorithm 3 VFs Placement Planning

Input: predicted application popularity vector **p**;
for each VF $v \in$ **p do**
    for each CN $h \in \mathcal{N}$ **do**
        if h has enough SRBs **then** load v on h;
        else
            if cloud has enough SRBs **then**
                load v on cloud;
            end if
        end if
    end for
end for

As represented in Fig. 2, the proposed FL framework consists of the client level, responsible for the distributed local data training, and of a server side. The server side is typically represented by a base station or a more general central unit, set up for improving the global learning model, and to merge the locally trained EU models. The client and server sides interact with each other, throughout a series of iteration rounds u. It is important to highlight that the number of EUs involved in the training procedure are a subset of the totality of the EUs.
The FL procedure consists of the following steps

- Let $\mathcal{K}$ be the set of the EUs involved in the training process. In parallel, each EU belonging to $\mathcal{K}$, i.e. EU χ, updates its local parameter vector $\mathbf{w}_\chi(u)$, which depends on its local dataset Θ_χ, accordingly with the following rule [8]

$$\mathbf{w}_\chi(u) = \hat{\mathbf{w}}_\chi(u-1) - \xi \nabla F_\chi(\hat{\mathbf{w}}_\chi(u-1)), \quad (15)$$

  where ξ is the learning rate and $\hat{\mathbf{w}}_\chi(u-1)$ represents the term $\mathbf{w}_\chi(u-1)$ after global aggregation.

- As detailed in [20], the server side computes the weighted average expressed by

$$\mathbf{w}(u) = \frac{\sum_{\chi \in \mathcal{K}} |\Theta_\chi| \mathbf{w}_\chi}{\sum_{\chi \in \mathcal{K}} |\Theta_\chi|}. \quad (16)$$

It is important to make evident that EUs, in performing distributed data training accordingly with the FL framework, achieve numerous advantages in terms of client privacy, and limited exploitation of their computational resources. This is directly connected to the fact that training data locally on the client's site, helps users to keep their sensitive and personal information reserved, since the uploading of the EU χ parameter vector $\mathbf{w}_\chi$ does not expose the client to any sort of privacy matter. More specifically, from $\mathbf{w}_\chi$, it is not elementary to retrieve Θ_χ. Finally, each algorithm iteration round involves just a part of the whole EUs' set, reducing the message passing between client and central server entities. Strongly connected with this aspect, the usage of the gradient descent algorithm is able to afford the learning problem without implying an excessive resource consumption, meeting the limited computational capabilities intrinsic of each mobile device.
Algorithms 1 and 2 exhibit the pseudocode corresponding to the client and server sides, respectively.

4.3 VFs placement planning

Once the FL framework is applied to obtain SRs prediction on the basis of the historical EUs' information, properly aggregated by the central server, the VFs' placement planning strategy starts. The placement acts on the basis of the VFs popularity, expressed with the popularity vector **p**. The popularity vector **p** has length equal to $\mathcal{V}$ and contains the type of the VFs sorted by descending order on the basis of the occurrence frequency of each VF type in the pool of the whole network requests.
In order to validate the benefits of the proposed framework to the VFs placement problem, we propose a straightforward placement strategy strictly dependent on **p**. Supposing that the predicted network SRs are given in terms of the VFs' popularity and expressed with the popularity vector **p**, the VFs' placement is realized through the following steps

1. Process the popularity vector **p** starting from the most popular VF in **p**, i.e., $r^\star$, hence from the most requested VF;

2. Deploy $r^\star$ on the first CN with enough available SRBs to host $r^\star$;

3. Deploy $r^\star$ on the cloud if it has enough available SRBs to host $r^\star$;

4. If $r^\star$ cannot be loaded neither on the CNs nor on the cloud

(a) if the VF $\hat{r}$ which can be hosted by a CN or cloud does not exist in **p**, then terminate placement;

(b) Otherwise repeat steps $1) - 4)$.

The pseudocode of the VFs planning strategy is detailed in Algorithm 3.

Algorithm 4 SRs Allocation Planning

```
until all the SRs are not allocated repeat
for each unallocated SR r do
    builds its preferences on C_r and proposes to its favorite element in C_r;
end for
for each computation site do
    accepts the SR requiring the VF type with the more stringent deadline;
    updates the corresponding queuing time;
end for
end repeat
```

4.4 SRs allocation planning

The designed SRs allocation policy is based on the matching theory principles [47, 48], and consider the EUs' perspective. In order to better explain this point, it is important to highlight that the SRs allocation strategy is based on metrics which do not consider the SP revenue, but only the EUs' interests. In this regard, the two parts involved in the matching are the SRs and the computational sites, referred hereafter, for each SR r, as $\mathcal{C}_r$. The set of the computational sites may be different for diverse SRs since, given the SR r, $\mathcal{C}_r$ consists of the CNs which contain the VF requested by r and of the cloud, if this contains the desired VF. Each SR r expresses the preference in being matched, i.e., in being computed, with each element of $\mathcal{C}_r$ and vice versa. The SRs aim at minimizing their own TSA defined as in *(7)*, hence they prefer to be executed on computational sites which lower *(7)*. By contrast, the computational sites prefer SRs requiring VFs with stringent deadline requirements.
Therefore, the matching algorithm consisting of a modified version of the Gale-Shapley [47] algorithm can be summarized through the following steps

1. Each SR builds its preference on the elements belonging to $\mathcal{C}_r$;
2. Each SR r, proposes to be computed on its most preferred computational site;
3. Each computational site, among the received computational proposals, accepts the SR requiring the VF type with the closest deadline, and discards the other proposals;
4. Update queuing time on each CN;
5. Update preferences of the unallocated SRs;

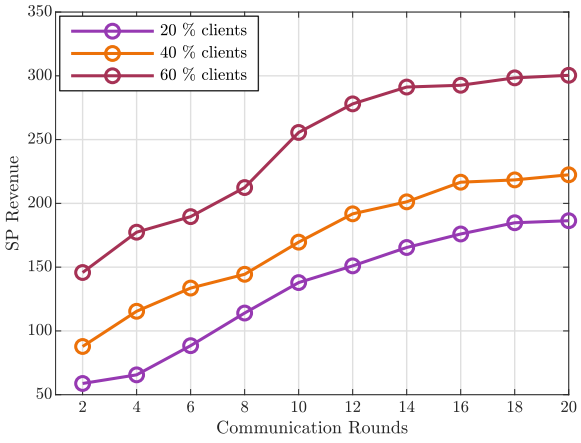


Fig. 3 – SP revenue by varying communication rounds, considering 100 SRs and 20 VFs

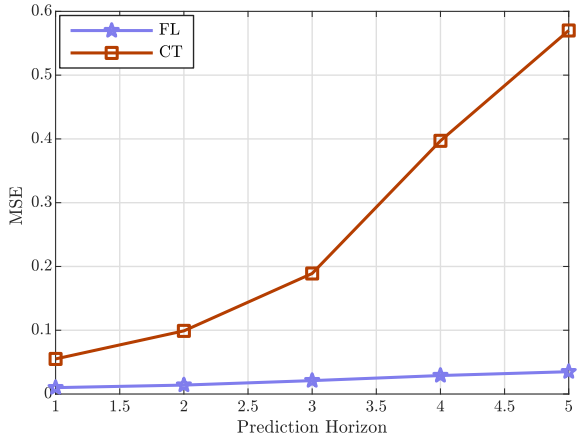


Fig. 4 – MSE by varying the time prediction horizon for type 1 SRs

6. repeat steps $2) - 6)$ until all the SRs are allocated.

Algorithm 4 explains in more detail the SRs allocation planning procedure.

5. NUMERICAL RESULTS

The proposed FL-based framework has been tested by resorting to numerical simulations in the Tensorflow environment. We supposed an IoE scenario consisting of $\mathcal{N} = 3$ CNs, equipped with a CPU frequency equals to 2.4 GHz, while the cloud has been equipped with a CPU frequency equals to 4.6 GHz. Furthermore, we set $S = 70$ and $U = 120$.
The VFs required by SRs have been modeled in a similar way as in [39, 49, 50], and we considered the presence of two priorities, corresponding to the set MovieLens 1M dataset [51] and MovieLens 100K dataset [51], respectively. We modeled 10 VFs, each of which needs a number of SRBs uniformly distributed in $[50, 80]$. All the FL network hyperparameters and the neural architecture have been assumed to be the same as those in [39]. Each SR has been modeled as a number of 64 bits format instructions uniformly distributed in $[250, 800]$, needing 8 CPU

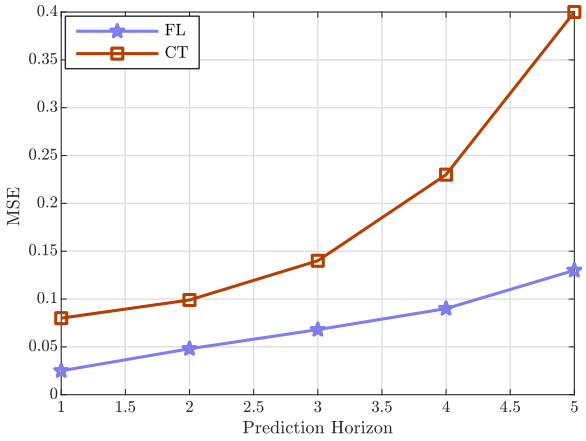


Fig. 5 – MSE by varying the time prediction horizon for type 2 SRs

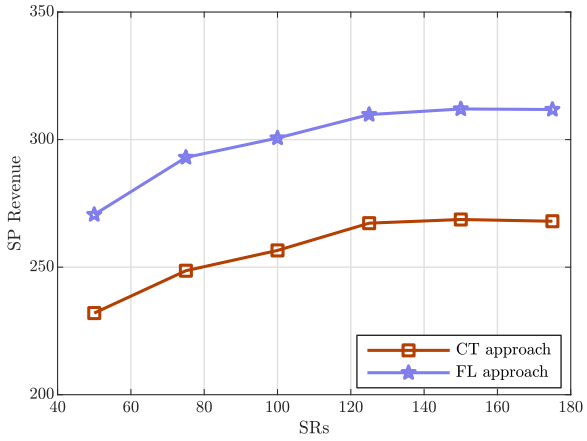


Fig. 6 – SP revenue by varying the number of SRs, considering 10 VFs

cycles per instruction. Furthermore, as loss function, we adopted the Mean Squared Error (MSE) which, for each data ι_ϕ in Θ_χ is defined as

$$MSE = \frac{1}{\Phi}\sum_{\phi=1}^{\Phi}(\hat{\iota}_\phi - \iota_\phi)^2, \qquad (17)$$

where Φ is the number of samples in the test data, and $\hat{\iota}_\phi$ is the predicted value. Then, to test the effectiveness of the proposed approach, we made comparison in terms of accuracy of our strategy, with the prediction scheme based on the application of the CT principles by performing the phase space reconstruction method as explained in [52, 53], and by using the predictive model of the k-neighbors discussed in [54]. It is important to note that the CT approach is performed on the central server site, on which all the user data is gathered without considering the preservation of their privacy.

Fig. 4 and Fig. 5, which exhibit the MSE behavior by varying the prediction horizon, confirm the greater accuracy of the proposed model in comparison to CT. As it is evident in Fig. 4 and Fig. 5, the MSE grows as the prediction horizon increases. This is a direct consequence of the natural difficulty in predicting the long-term behavior of the series. Nevertheless, both the figures show the superiority of the proposed approach in comparison with the alternative here considered.

Then, Fig. 3 makes clear the significant improvement obtained by increasing the number of communication rounds, i.e., information updates, between the server and the clients, for different numbers of EUs involved in the FL process. The direct implication is that higher is the number of the EUs taking part in the learning process, the greater the levels of accuracy on the acquired information on which the VFs placement strategy is based. Moreover, the SP revenue improves its trend. It is important to highlight here that the FL requires a converge time of 12.42 seconds to converge, against the 6.17 seconds required by the CT approach. Fig. 6 shows the SP revenue behavior by increasing the number of SRs. As it is straightforward to note, the SP revenue tends to grow by increasing the number of SRs, until the network infrastructure is not saturated and consequently it cannot accept new SRs. Such a situation is clearly a consequence of the physical resources limitation of the network. Finally, Fig. 7 depicts the behavior of the percentage of the SRs discarded, i.e., the percentage of the SRs which have not been served by the network infrastructure since their computation is not finished before the expiration of their deadline. In conclusion, the resulting system performance makes clear the validity of the FL application for our problem, highlighting the importance of considering the data expressing the users' preferences and daily habits.

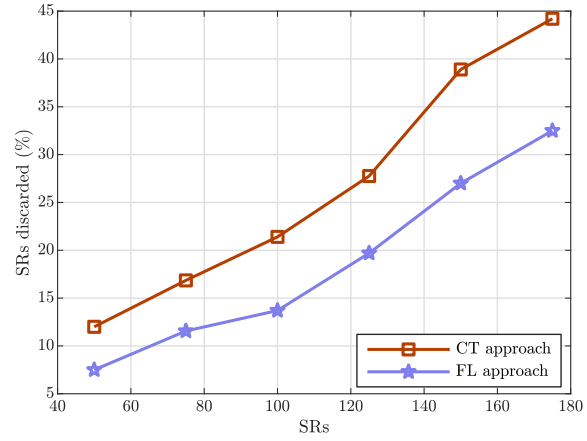


Fig. 7 – Percentage of SRs discarded, by increasing the SR number

6. CONCLUSION

This paper has dealt with a framework based on the federated learning paradigm to maximise SP revenue, in a hybrid cloud-edge system, arranged to support IoE applications. The proposed framework resorts to the use of the FL approach to predict the SRs demand, in compliance with the users' privacy. Furthermore, a VFs placement on the basis of the obtained SRs demand prediction has been performed and, the related SRs allocation, modeled as a matching game problem, has been hence

accomplished. The effectiveness of the proposed framework has been finally validated by providing performance comparisons with an alternative predictive approach based on the chaos theory. In reference to the future research directions, a very interesting topic needing further exploration may be represented by the definition of novel solutions and methodologies to allow the design of privacy-based learning and inference of deep learning and advanced signal processing in heterogeneous hardware architectures. Such a privacy-preserving approach will rely on Homomorphic Encryption that enables processing directly on encrypted data.

REFERENCES

[1] H. Tianfield. "Towards Edge-Cloud Computing". In: *2018 IEEE International Conference on Big Data (Big Data)*. Dec. 2018, pp. 4883–4885. DOI: 10.1109/BigData.2018.8622052.

[2] H. Liu, F. Eldarrat, H. Alqahtani, A. Reznik, X. de Foy, and Y. Zhang. "Mobile Edge Cloud System: Architectures, Challenges, and Approaches". In: vol. 12. 3. Sept. 2018, pp. 2495–2508. DOI:10.1109/JSYST.2017.2654119.

[3] X. Shan, H. Zhi, P. Li, and Z. Han. "A Survey on Computation Offloading for Mobile Edge Computing Information". In: *2018 IEEE 4th International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing, (HPSC) and IEEE International Conference on Intelligent Data and Security (IDS)*. May 2018, pp. 248–251. DOI:10.1109/BDS/HPSC/IDS18.2018.00060.

[4] P. Mach and Z. Becvar. "Mobile Edge Computing: A Survey on Architecture and Computation Offloading". In: *IEEE Communications Surveys Tutorials* 19.3 (2017), pp. 1628–1656. ISSN:1553-877X. DOI:10.1109/COMST.2017.2682318.

[5] S. Singh. "Optimize cloud computations using edge computing". In: *2017 International Conference on Big Data, IoT and Data Science (BID)*. Dec. 2017, pp. 49–53. DOI:10.1109/BID.2017.8336572.

[6] P. Corcoran and S. K. Datta. "Mobile-Edge Computing and the Internet of Things for Consumers: Extending cloud computing and services to the edge of the network". In: *IEEE Consumer Electronics Magazine* 5.4 (Oct. 2016), pp. 73–74. ISSN:2162-2248. DOI:10.1109/MCE.2016.2590099.

[7] M. Chiang and T. Zhang. "Fog and IoT: An Overview of Research Opportunities". In: vol. 3. 6. Dec. 2016, pp. 854–864. DOI:10.1109/JIOT.2016.2584538.

[8] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. S. Chan. "When Edge Meets Learning: Adaptive Control for Resource-Constrained Distributed Machine Learning". In: 2018, pp. 63–71.

[9] R. Fantacci and B. Picano. "A Matching Game With Discard Policy for Virtual Machines Placement in Hybrid Cloud-Edge Architecture for Industrial IoT Systems". In: *IEEE Transactions on Industrial Informatics* 16.11 (2020), pp. 7046–7055. DOI: 10.1109/TII.2020.2999880.

[10] F. Chiti, R. Fantacci, and B. Picano. "A matching game for tasks offloading in integrated edge-fog computing systems". In: *Transactions on Emerging Telecommunications Technologies* 31.2 (2020). e3718 ett.3718, e3718. DOI: https://doi.org/10.1002/ett.3718. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ett.3718. URL:https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3718.

[11] R. Fantacci and B. Picano. "When Network Slicing Meets Prospect Theory: A Service Provider Revenue Maximization Framework". In: *IEEE Transactions on Vehicular Technology* 69.3 (2020), pp. 3179–3189. DOI: 10.1109/TVT.2019.2963462.

[12] W. Saad, M. Bennis, and M. Chen. "A Vision of 6G Wireless Systems: Applications, Trends, Technologies, and Open Research Problems". In: *IEEE Network* 34.3 (2020), pp. 134–142. DOI: 10.1109/MNET.001.1900287.

[13] C. Tselios, I. Politis, M. Tsagkaropoulos, and T. Dagiuklas. "Valuing Quality of Experience: A Brave New Era of User Satisfaction and Revenue Possibilities". In: *2011 50th FITCE Congress - "ICT: Bridging an Ever Shifting Digital Divide"*. 2011. DOI:10.1109/FITCE.2011.6133422.

[14] F. Conti, S. Colonnese, F. Cuomo, L. Chiaraviglio, and I. Rubin. "Quality Of Experience Meets Operators Revenue: Dash Aware Management for Mobile Streaming". In: *2019 8th European Workshop on Visual Information Processing (EUVIP)*. 2019, pp. 64–69. DOI:10.1109/EUVIP47703.2019.8946152.

[15] C. Zhang, P. Patras, and H. Haddadi. "Deep Learning in Mobile and Wireless Networking: A Survey". In: vol. abs/1803.04311. 2018. arXiv: 1803.04311. URL:http://arxiv.org/abs/1803.04311.

[16] S. Athmaja, M. Hanumanthappa, and V. Kavitha. "A survey of machine learning algorithms for big data analytics". In: *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*. Mar. 2017, pp. 1–4. DOI: 10.1109/ICIIECS.2017.8276028.

[17] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani. "Deep Learning for IoT Big Data and Streaming Analytics: A Survey". In: *IEEE Communications Surveys Tutorials* 20.4 (2018), pp. 2923–2960. ISSN: 1553-877X. DOI: 10.1109/COMST.2018.2844341.

[18] M. A. Alsheikh, S. Lin, D. Niyato, and H. Tan. "Machine Learning in Wireless Sensor Networks: Algorithms, Strategies, and Applications". In: *IEEE Communications Surveys Tutorials* 16.4 (2014), pp. 1996–2018. ISSN: 1553-877X. DOI: 10.1109/COMST.2014.2320099.

[19] P. V. Klaine, M. A. Imran, O. Onireti, and R. D. Souza. "A Survey of Machine Learning Techniques Applied to Self-Organizing Cellular Networks". In: *IEEE Communications Surveys Tutorials* 19.4 (2017), pp. 2392–2431. ISSN: 1553-877X. DOI: 10.1109/COMST.2017.2727878.

[20] H. B. McMahan, E. Moore, D. Ramage, and B. Agüera y Arcas. "Federated Learning of Deep Networks using Model Averaging". In: vol. abs/1602.05629. 2016. arXiv: 1602.05629. URL: http://arxiv.org/abs/1602.05629.

[21] Q. Yang, Y. Liu, T. Chen, and Y. Tong. "Federated Machine Learning: Concept and Applications". In: vol. abs/1902.04885. 2019. arXiv: 1902.04885. URL: http://arxiv.org/abs/1902.04885.

[22] V. Smith, C.-K. Chiang, M. Sanjabi, and A. Talwalkar. "Federated Multi-Task Learning". In: vol. abs/1705.10467. 2017. arXiv: 1705.10467. URL: http://arxiv.org/abs/1705.10467.

[23] N. H. Tran, W. Bao, A. Zomaya, N. Minh N.H., and C. S. Hong. "Federated Learning over Wireless Networks: Optimization Model Design and Analysis". In: *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*. Apr. 2019, pp. 1387–1395. DOI: 10.1109/INFOCOM.2019.8737464.

[24] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi. "Beyond Inferring Class Representatives: User-Level Privacy Leakage From Federated Learning". In: *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*. Apr. 2019, pp. 2512–2520. DOI: 10.1109/INFOCOM.2019.8737416.

[25] H. B. McMahan, E. Moore, D. Ramage, and B. Agüera y Arcas. "Communication-Efficient Learning of Deep Networks from Decentralized Data". In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Vol. 54. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA: PMLR, Apr. 2017, pp. 1273–1282. URL: http://proceedings.mlr.press/v54/mcmahan17a.html.

[26] P. Subramaniam and M. J. Kaur. "Review of Security in Mobile Edge Computing with Deep Learning". In: (Mar. 2019), pp. 1–5. DOI: 10.1109/ICASET.2019.8714349.

[27] Z. Chang, L. Lei, Z. Zhou, S. Mao, and T. Ristaniemi. *Learn to Cache: Machine Learning for Network Edge Caching in the Big Data Era*. Vol. 25. 3. June 2018, pp. 28–35. DOI: 10.1109/MWC.2018.1700317.

[28] Z. Zhou, H. Liao, B. Gu, K. M. S. Huq, S. Mumtaz, and J. Rodriguez. "Robust Mobile Crowd Sensing: When Deep Learning Meets Edge Computing". In: vol. 32. 4. July 2018, pp. 54–60. DOI: 10.1109/MNET.2018.1700442.

[29] A. K. Sangaiah, D. V. Medhane, T. Han, M. S. Hossain, and G. Muhammad. "Enforcing Position-Based Confidentiality with Machine Learning Paradigm through Mobile Edge Computing in Real-Time Industrial Informatics". In: *IEEE Transactions on Industrial Informatics* (2019), pp. 1–1. ISSN: 1551-3203. DOI: 10.1109/TII.2019.2898174.

[30] S. Yu, X. Wang, and R. Langar. "Computation offloading for mobile edge computing: A deep learning approach". In: *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. Oct. 2017, pp. 1–6. DOI: 10.1109/PIMRC.2017.8292514.

[31] N. C. Luong, Z. Xiong, P. Wang, and D. Niyato. "Optimal Auction for Edge Computing Resource Management in Mobile Blockchain Networks: A Deep Learning Approach". In: (May 2018), pp. 1–6. ISSN: 1938-1883. DOI: 10.1109/ICC.2018.8422743.

[32] T. Tuor, S. Wang, T. Salonidis, B. J. Ko, and K. K. Leung. "Demo abstract: Distributed machine learning at resource-limited edge nodes". In: *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. Apr. 2018, pp. 1–2. DOI: 10.1109/INFCOMW.2018.8406837.

[33] T. Kudo and T. Ohtsuki. "Cell selection using distributed Q-learning in heterogeneous networks". In: *2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*. Oct. 2013, pp. 1–6. DOI: 10.1109/APSIPA.2013.6694368.

[34] L. Li, K. Ota, and M. Dong. "Human in the Loop: Distributed Deep Model for Mobile Crowdsensing". In: *IEEE Internet of Things Journal* 5.6 (Dec. 2018), pp. 4957–4964. ISSN: 2327-4662. DOI: 10.1109/JIOT.2018.2883318.

[35] L. Valerio, A. Passarella, and M. Conti. "Optimal trade-off between accuracy and network cost of distributed learning in Mobile Edge Computing: An analytical approach". In: (June 2017), pp. 1–9. DOI: 10.1109/WoWMoM.2017.7974310.

[36] Y. Jiao, P. Wang, D. Niyato, M. Abu Alsheikh, and S. Feng. "Profit Maximization Auction and Data Management in Big Data Markets". In: *2017 IEEE Wireless Communications and Networking Conference (WCNC)*. Mar. 2017, pp. 1–6. DOI: 10.1109/WCNC.2017.7925760.

[37] S. Li, J. Huang, and S. R. Li. "Dynamic Profit Maximization of Cognitive Mobile Virtual Network Operator". In: *IEEE Transactions on Mobile Computing* 13.3 (Mar. 2014), pp. 526–540. ISSN: 1536-1233. DOI:10.1109/TMC.2013.10.

[38] M. Li, Y. Sun, H. Huang, J. Yuan, Y. Du, Y. Bao, and Y. Luo. "Profit maximization resource allocation in cloud computing with performance guarantee". In: *2017 IEEE 36th International Performance Computing and Communications Conference (IPCCC)*. Dec. 2017, pp. 1–2. DOI:10.1109/PCCC.2017.8280482.

[39] Z. Yu, J. Hu, G. Min, H. Lu, Z. Zhao, H. Wang, and N. Georgalas. "Federated Learning Based Proactive Content Caching in Edge Computing". In: *2018 IEEE Global Communications Conference (GLOBECOM)*. 2018, pp. 1–6. DOI:10.1109/GLOCOM.2018.8647616.

[40] K. Yang, T. Jiang, Y. Shi, and Z. Ding. "Federated Learning via Over-the-Air Computation". In: Dec. 2018.

[41] H. Kim, J. Park, M. Bennis, and S. Kim. "Blockchained On-Device Federated Learning". In: 2019, pp. 1–1. DOI:10.1109/LCOMM.2019.2921755.

[42] J. Ren, H. Wang, T. Hou, S. Zheng, and C. Tang. "Federated Learning-Based Computation Offloading Optimization in Edge Computing-Supported Internet of Things". In: vol. 7. 2019, pp. 69194–69201. DOI:10.1109/ACCESS.2019.2919736.

[43] A. Ahmad, A. Floris, and L. Atzori. "QoE-centric service delivery: A collaborative approach among OTTs and ISPs". In: *Computer Networks* 110 (2016), pp. 168–179. ISSN: 1389-1286. DOI:10.1016/j.comnet.2016.09.022.

[44] A. Floris, A. Ahmad, and L. Atzori. "QoE-Aware OTT-ISP Collaboration in Service Management: Architecture and Approaches". In: *ACM Trans. Multimedia Comput. Commun. Appl.* 14.2s (Apr. 2018). ISSN: 1551-6857. DOI:10.1145/3183517.

[45] A. Ahmad and L. Atzori. "MNO-OTT Collaborative Video Streaming in 5G: The Zero-Rated QoE Approach for Quality and Resource Management". In: *IEEE Transactions on Network and Service Management* 17.1 (2020), pp. 361–374. DOI: 10.1109/TNSM.2019.2942716.

[46] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. S. Chan, and K. K. Leung. "Dynamic Service Migration in Mobile Edge-Clouds". In: CoRR abs/1506.05261 (2015). arXiv: 1506.05261. URL:http://arxiv. org/abs/1506.05261.

[47] S. Bayat, Y. Li, L. Song, and Z. Han. "Matching Theory: Applications in wireless communications". In: IEEE Signal Processing Magazine 33.6 (Nov. 2016), pp. 103–122. ISSN:1053-5888. DOI:10.1109/MSP.2016.2598848.

[48] A. E. Roth and M. Sotomayor. *Two-Sided Matching: A Study in Game-Theoretic Modeling and Analysis*. Cambridge University Press, UK, 1990.

[49] S. Mü ller, O. Atan, M- van der Schaar, and A. Klein. "Context-Aware Proactive Content Caching with Service Differentiation in Wireless Networks". In: *IEEE Transactions on Wireless Communications* PP (June 2016). DOI:10.1109/TWC.2016.2636139.

[50] S. Li, J. Xu, M. van der Schaar, and W. Li. "Popularity-driven content caching". In: *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*. Apr. 2016, pp. 1–9. DOI: 10.1109/INFOCOM.2016.7524381.

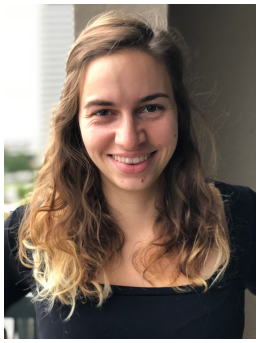
[51] F. M. Harper and J. A. Konstan. "The MovieLens Datasets: History and Context". In: *ACM Trans. Interact. Intell. Syst.* 5.4 (Dec. 2015), 19:1–19:19. ISSN: 2160-6455. DOI: 10.1145/2827872. URL: http://doi.acm.org/10.1145/2827872.

[52] Z. Liu. "Chaotic Time Series Analysis". In: *Mathematical Problems in Engineering* (Feb. 2010).

[53] F. Takens. "Detecting strange attractors in turbulence". In: *Dynamical Systems and Turbulence, Warwick 1980*. Ed. by David Rand and Lai-Sang Young. Berlin, Heidelberg: Springer Berlin Heidelberg, 1981, pp. 366–381. ISBN:978-3-540-38945-3.

[54] H. Kantz and T. Schreiber. *Nonlinear Time Series Analysis*. 2nd. Cambridge: Cambridge University Press, 2003. DOI:10.1017/CBO9780511755798.

AUTHORS



Benedetta Picano (Member, IEEE) has been a postdoctoral research fellow in the Department of Information Engineering at the University of Florence (Italy), since 2019. She received a Ph.D. degree in Information Engineering from the Department of Information Engineering at the University of Florence , in February 2020. She received the 1st and 2nd level Laurea Degree in Computer Networks and Computer Engineering at the University of Florence, in July 2013, and October 2016, respectively. She was a visiting researcher at the University of Houston. Her research fields include matching theory, nonlinear time series analysis, resource allocation in edge and fog computing infrastructures, and machine learning.

Romano Fantacci is a Full Professor of Computer Networks at the University of Florence, Florence, Italy, where he heads the Wireless Networks Research Lab. He is the founding director of the Information Communication Technology Inc. (TiCOM), with Leonardo spa and of the Wireless Communications Research Centre (LiRS) with TIM -Telecom Italia. He received an M.S. degree in Electrical Engineering from the University of Florence, Italy and a Ph.D. degree in Computer Networks from the University of Florence, Italy. Dr. Fantacci was elected Fellow of the IEEE in 2005 for contributions to wireless communication networks. He received several awards for his research, including the IEE Benefactor Premium, the 2002 IEEE Distinguished Contributions to Satellite Communications Award, the 2015 IEEE WTC Recognition Award, the IEEE sister society AEIT Young Research Award and the IARIA Best Paper Award, the IEEE IWCMC'16 Best Paper Award and the IEEE Globecom'16 Best Paper Award. He served as Area Editor for IEEE Trans. Wireless Commun., Associate Editor for IEEE Trans. on Comm., IEEE Trans. Wireless Comm., Area Editor for IEEE IoT Journal, Regional Editor for IET Communications and Associate Editor for several non-IEEE Technical Journals. He guest edited special issues for IEEE Journals and Magazines and served as symposium chair of several IEEE conferences, including VTC, WCNC, PIRMC, ICC and Globecom. Dr. Fantacci currently serves as a member of the Board of Governors of the IEEE sister society AEIT, and of the Steering Committee of IEEE Wireless Comm. Letters.

Tommaso Pecorella (Senior member, IEEE) received Ph.D. and M.Sc. degrees in Electronic Engineering (Telecommunications track) from the Department of Information Engineering at University of Florence (Italy) in 2000 and 1996 respectively. From 2001 to 2007 he was a researcher at Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT). Since November 2007 he is a tenure-track Assistant Professor in the Department of Information Engineering at University of Florence (Italy). In 2018 and 2019 he was also visiting professor at University of Saint Louis, Missouri (USA). He received the Best paper award at the IEEE GLOBECOM 2016, and in 2021 got the Italian Habilitation (Abilitazione Scientifica Nazionale) for Associate Professorship in Telecommunication Engineering. He is the author of more than 90 publications between conference papers and journals. His research interests focus on IoT communication systems, network security, and application of machine learning to networking systems.

Adnan Rashid (IEEE and Internet Society member) is currently working as a Ph.D. Research Scholar in the Department of Information Engineering (DINFO), University of Florence, Italy, since November 2018. He received a Master's degree in Computer Engineering from the Department of Electrical Engineering, Capital University of Science & Technology (CUST), Islamabad, Pakistan, in December 2015. He received a Bachelor of Science in Computer Science (BSCS) from the Federal Urdu University of Arts, Science, and Technology (FUUAST), Islamabad, Pakistan, 2009. He was a visiting faculty member from 2017-to-2018 in the Department of Computer Science at the PMAS-Arid Agriculture University, Rawalpindi, Pakistan. His research activity focused on the security and resilience of IoT systems, Software Defined Networking (SDN), and Fog Networks. He is involved in multiple IETF working groups and doing research to standardize the IPv6 and 6LoWPAN-ND features. He is collaborating with the open-source ns-3 simulator maintainers for the development of the 6LoWPAN-ND protocol.

IOE: TOWARDS APPLICATION-SPECIFIC TECHNOLOGY SELECTION

Biswajit Paul[1], Gokul Chandra Biswas[2], Habib F. Rashvand[3]
[1]Electrical and Electronic Engineering, Shahjalal University of Science and Technology, Sylhet, Bangladesh, [2]Genetic Engineering and Biotechnology, Shahjalal University of Science and Technology, Sylhet, Bangladesh, [3]Advanced Communication Systems, University of Warwick, Coventry, UK

NOTE: Corresponding author: Biswajit Paul, biswajit-eee@sust.edu

Abstract – *Determining the suitability of any technology for an Internet of Everything (IoE) application is essential in the presence of diverse technologies and application requirements. Some of the IoE applications include smart metering, wearables, healthcare, remote monitoring, inventory management and industrial automation. Energy efficiency, scalability, security, low-cost deployment and network coverage are some of the requirements that vary from one application to another. Wireless technologies such as WiFi, ZigBee, Bluetooth, LTE, NB-IoT, LoRa and SigFox will play crucial roles in enabling these applications. Some of the technological features are transmission range, bandwidth, data rate, security schemes and infrastructure requirements. As there is no one-size-fits-all network solution available, the key is to understand the diverse requirements of different IoE applications and specific features offered by different IoE enabling technologies. Application-specific technology selection will ensure the best possible utilization of any technology and the quality of service requirements. An overview of network performance expectations from various IoE applications and enabling technologies, their features and potential applications are presented in this paper.*

Keywords – IoE, LPWANs, M2M, MTC, network design, suitable technology selection, wireless technologies

ABBREVIATIONS

- 3GPP - 3rd Generation Partnership Project
- 8PSK - Eight Phase Shift Keying
- AMI - Advanced Metering Infrastructure
- BPSK - Binary Phase Shift Keying
- CAPEX - Capital Expenditure
- COPD - Chronic Obstructive Pulmonary Disease
- CSMA/CA - Carrier Sense Multiple Access with Collision Avoidance
- D2D - Device to Device
- DBPSK - Differential Binary Phase Shift Keying
- DMM - Disributed IP Mobility Management
- DSO - Distribution System Operators
- DSSS - Direct Sequence Spread Spectrum
- EC-GSM-IoT - Extended Coverage Global System for Mobile Communications for the Internet of Things
- eDRX - extended Discontinuous Reception
- eGPRS - enhanced General Packet Radio Service
- eMTC - enhanced Machine Type Communication
- EVs - Electric Vehicles
- FHSS - Frequency Hopping Spread Spectrum
- GFSK - Gaussian Frequency Shift Keying
- GMSK - Gaussian Minimum Shift Keying
- GW - Gateway
- HANs - Home Area Networks
- HCO - Healthcare Organization
- IC-IoE- Information-Centric IoE
- IIoT - Industrial Internet of Everything
- IoE - Internet of Everything
- IoT - Internet of Things
- IP - Internet Protocol
- ISM - Industrial, Scientific and Medical
- LoS - Line of Sight
- LPWANs - Low Power Wide Area Networks
- LR-WPAN - Low Rate Wireless Personal Area Network
- M2M - Machine to Machine
- MLANs - Meter Local Area Networks
- MMC - Massive Machine Communications
- mMTC - Massive Machine Type Communications

- MN - Moving Networks
- MTC - Machine Type Communication
- MTDs - Machine Type Devices
- NANs - Neighborhood Area Networks
- NB-IoT - Narrowband Internet of Things
- NOMA - Non Orthogonal Multiple Access
- OPEX - Operational Expenditure
- OS - Operatig System
- PER - Packet Error Ratio
- PLC - Power Line Communication
- PMIPV6 - Proxy Mobile IPv6
- PSM - Power Saving Management
- PWPN - Power Wireless Private Network
- QoS - Quality of Service
- RF - Radio Frequency
- RPMA - Random Phase Multiple Access
- RSUs - Roadside Units
- SGs - Smart Grids
- SMs - Smart Meters
- SPHERE - Sensor Platform for Residential Environment
- UAV - Unmanned Aerial Vehicle
- UDN - Ultra-Dense Network
- UNB - Ultra-Narrowband
- URLLC - Ultra-Reliable Low Latency Communications
- V2I - Vehicle to Infrastructure
- V2N - Vehicle to Network
- V2P - Vehicle to Pedestrian
- V2V - Vehicle to Vehicle
- V2X - Vehicle to Everything
- WANs - Wide Area Networks
- WBANs - Wireless Body Area Networks
- WIA-PA - Wireless Networks for Industrial Automation for Process Automation
- WirelessHART - Wireless Highway Addressable Remote Transducer
- WISA - Wireless Interface for Sensors and Actuators
- WLAN - Wireless Local Area Network
- WPCN - Wireless Powered Communication Network
- WSNs - Wireless Sensor Networks

1. INTRODUCTION

The term 'Internet of Things' (IoT) refers to the network of physical objects or things embedded with electronics, software, sensors and network connectivity where information exchange takes place automatically [1]. The term IoE is preferred over IoT by many as IoE comprehensively addresses the connectivity of various technologies, processes and people while IoT addresses interconnectivity of physical objects, data inputs and outputs. Humans, monitoring sensors, healthcare equipment, sensor-equipped automobiles etc. are considered in 'Everything' [2]. A significant increase in the number of deployed IoE devices can be observed in recent years as the IoE concept receives broader industry momentum. Some predictions on the IoE deployment scale [3], technology's market penetration [4] and estimated revenue generation [5] can be found in the literature. IoE promises ease of flow of information efficiently in a fast-paced world with various envisioned application types such as IoE devices from mobiles, smart home energy management systems, supporting disabled people, tracking human behaviour, underwater sensor networks, military affairs and autonomous cars. Agriculture, healthcare, environment, transport, industrial automation etc. are some of the potential IoE application domains. IoE will incorporate both humans and machines as suggested by some IoE applications where interaction with humans [6], places of residence [7], human nature [8], and environment [9] are observed.

Since IoE application requirements are diverse, network designs are often facilitated by differentiating Machine to Machine (M2M) networks from Machine Type Communication (MTC) networks. M2M communication includes the remote control of machines, monitoring, and collecting data from machines, whereas in MTC, typically, devices are small, inexpensive and can operate for an extended period without human intervention. M2M communication networks differentiate themselves from networks that relay traffic generated or consumed by humans in IoE. Examples of MTC are smart community, smart building, smart grid, smart water system etc. Network connectivity, communication protocols, middleware frameworks, etc. need careful consideration to support the massive number of devices. The heterogeneous nature of traffic such as static, intermittent, delay-sensitive, delay-tolerant, small or large packets and application-specific performance objectives can make the wireless network design more complicated and challenging. For example, the tolerable delay and an update frequency for the waste management application are 30 minutes and 1 hour, respectively. On the other hand, in-

dustrial monitoring and supervision applications can tolerate delays in a range of milliseconds and have update frequencies in the range of seconds [5].

Cellular networks will play a major role in the IoE domain in supporting M2M communication networks. However, future cellular standards will require optimizing the access network for both broadband and M2M communications to meet varying design challenges. In contrast to broadband networks, large-scale deployment of inexpensive low-complexity devices, smaller payload sizes with non-uniform traffic density, energy efficiency, extended network coverage are required for M2M networks [4]. Some enhancements have been proposed in the 3rd Generation Partnership Project (3GPP) to efficiently support M2M applications in 2G, 3G, LTE Cat-1 and higher networks. Extended Coverage Global System for Mobile Communications for the Internet of Things (EC-GSM-IoT) and Narrowband Internet of Things (NB-IoT) are cellular-based IoE enabling technologies. Besides the cellular-based technologies, short-range technologies such as Bluetooth, ZigBee and Wi-Fi, and non-cellular-based technologies such as LoRa and Sigfox will play vital roles to meet the huge connectivity demand placed by MTC networks.

Application-specific technology selection requires careful preparation such as analyzing energy efficiency, latency, reliability, reliability, scalability and security requirements. Video surveillance, a smart city application, cannot tolerate large delays compared to other smart city applications such as structural health monitoring and waste management. Video surveillance is an example of a high data-rate application while structural health monitoring and waste management are low data-rate applications. Some industrial applications such as closed-loop control/ interlocking and control require low data rates while delays in milliseconds are tolerated with a high update frequency. Average message sizes and average message transaction rates also vary from one application to another. For example, average message sizes and average message transaction rates are 20 bytes and 1.67×10^{-3}/s respectively for a typical home security application, and 1 bytes and 3.33×10^{-2}/s respectively for roadway signs. Some of the smart city applications such as road safety in urban/highways and most of the industrial applications such as factory automation/packaging machines are latency-critical IoE applications with high-reliability requirements.

Operating frequency, bandwidth, transmission range and data rate are some of the technological features of any technology. LoRa and Sigfox operate in the unlicensed Industrial, Scientific and Medical (ISM) spectrum band while EC-GSM-IoT and NB-IoT operate in licensed spectrum bands. Bluetooth and WiFi are two short-range technologies having transmission ranges of 50 m and 100 m respectively. Although highly dependent on communication environments, some researchers reported that LoRa and SigFox can achieve approximately 15 km and 20 km transmission ranges respectively. Cellular-based technologies such as NB-IoT, EC-GSM-IoT, eMTC can also achieve a long transmission range. Bluetooth, WiFi, NB-IoT, EC-GSM-IoT, eMTC have higher channel bandwidths compared to LoRa and SigFox. Channel bandwidths for Bluetooth and WiFi are 2 MHz and 22 MHz respectively while the channel bandwidth of SigFox is 100 Hz only. Bluetooth, WiFi, NB-IoT, EC-GSM-IoT, eMTC can support high data-rate applications while LoRa, SigFox support low data-rate applications.

The knowledge of application requirements and technological features of any technology can help us determine the suitability of that technology for a particular application. For example, WiFi, Bluetooth, NB-IoT will fit well in high throughput applications while LoRa and SigFox will not fit such applications. However, WiFi and Bluetooth are suitable for applications that require a small network coverage. On the other hand, LoRa and SigFox can provide larger network coverage. These observations can be collectively applied towards application-specific technology selection. Our concept is illustrated in Fig.1. Fig.2 shows the speed at which the IoE market is growing [10]. This paper is organized as follows: various IoE application requirements and network design constraints are discussed in Section 2, some IoE enabling technologies and features are discussed in Section 3, various IoE applications and enabling technologies are discussed in Section 4 and conclusions are drawn in Section 5.

2. IOE APPLICATION REQUIREMENTS AND NETWORK DESIGN CONSTRAINTS

The diverse nature of IoE application scenarios may have a diverse set of requirements. Some of the requirements could be application-specific while others fall into general expectations. Some requirements arise from the typical IoE devices and business objectives while others are relevant to networks. As an example, average message transaction rates and average message sizes are shown in Table 1. Long battery life, support for the massive number of devices, extended coverage, low device cost, low deployment cost, security and privacy etc. are some of the key requirements for some applications. Network scalability, throughput, cell capacity, interference and delay are important considerations for other IoE applications.

2.1 Energy efficiency

The most important issue in IoE networks is probably energy efficiency [5]. Since the end devices are operated by irreplaceable batteries and the network is expected to be functional for a long time without human intervention in applications such as fire warning and pipeline inspection [11], battery energy should be utilized most efficiently. A battery life span expectation of 10 years for network operation is reported in [5]. A significant amount of energy is consumed in packet transmission and reception processes compared to other processes. The author in [12] discussed the requirement of delicate balancing between

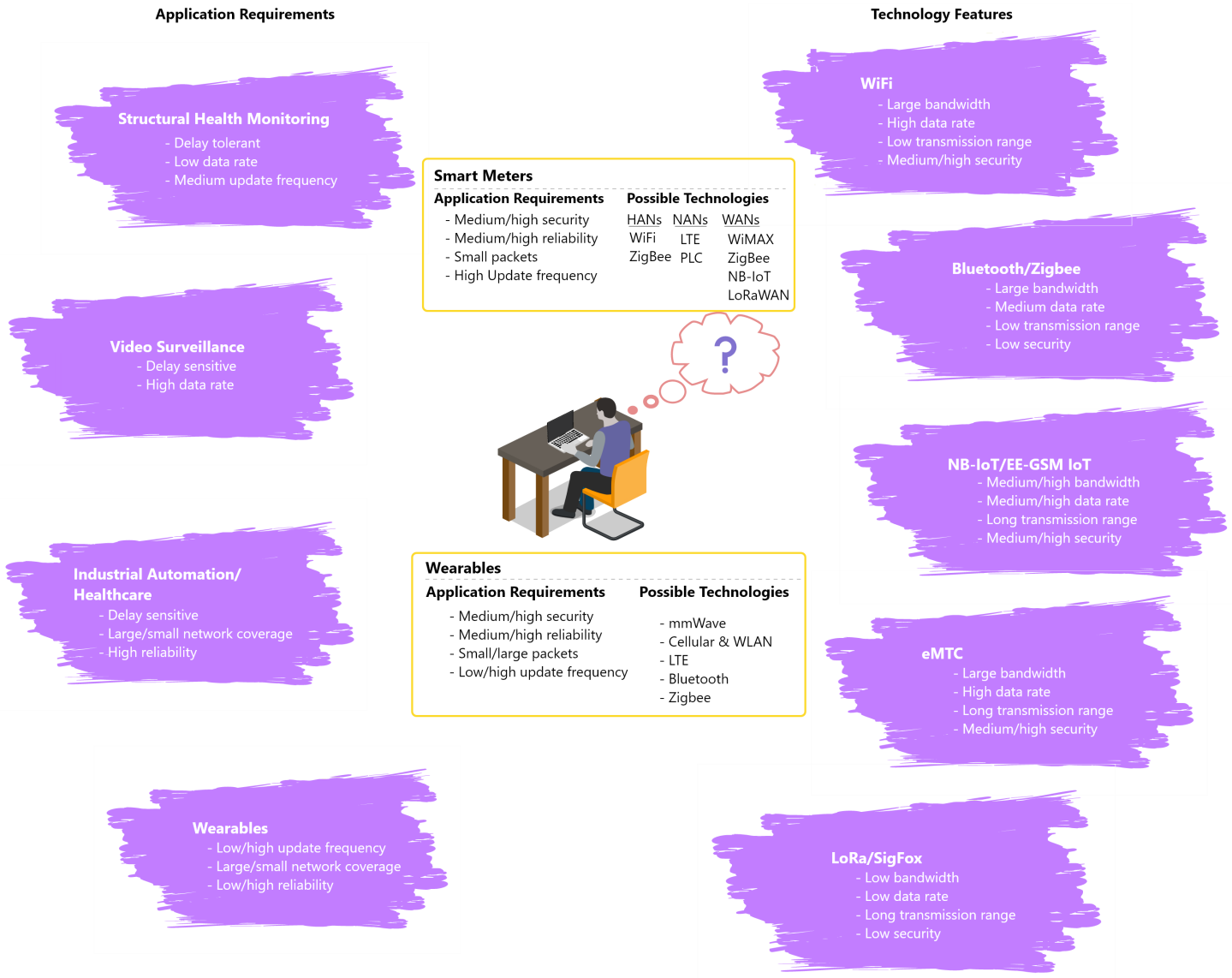


Fig. 1 – Application-specific technology selection

the number of packet transmissions and network lifetime. However, in some applications such as wearables devices where a significant volume of data may need to be processed, the circuit power consumption is often comparable to the transmit power [13].

Energy efficiency issues will have to be addressed through the design of hardware, software or MAC protocols, suitable routing scheme, efficient energy management system and energy harvesting. The multi-hop routing in [14] was found to be more energy-efficient than single-hop routing in LoRa networks while ensuring high network connectivity, low computational complexity for end nodes and addressing dynamic node distribution scenarios. On the other hand, the routing algorithm in [15] combined different energy harvesting techniques to improve the network lifetime and Quality of Service (QoS) under variable traffic load and energy availability conditions. Wireless power transfer enables the IoE nodes to collect energy from the Radio Frequency (RF) of the surrounding transmitters [16]. For Unmanned Aerial Vehicle (UAV) applications, UAV swarms can have a relatively good channel state to complete wireless power transfer as the probability of finding a Line of Sight (LoS) link is high [11]. Cognitive radio and Non-Orthogonal Multiple Access (NOMA) are candidate technologies for 5G networks for improving network spectral efficiency and scalability and the authors in [17] introduced a resource management framework for cognitive IoE networks with RF energy harvesting. In a Wireless Powered Communication Network (WPCN), multiple energy-limited devices first harvest energy in the downlink and then transmit information in the uplink. Although NOMA has been proposed to improve the system spectral efficiency in 5G networks, the authors in [13] found that NOMA-based WPCN not only consumes more energy but also is less spectrally efficient than TDMA-based WPCN.

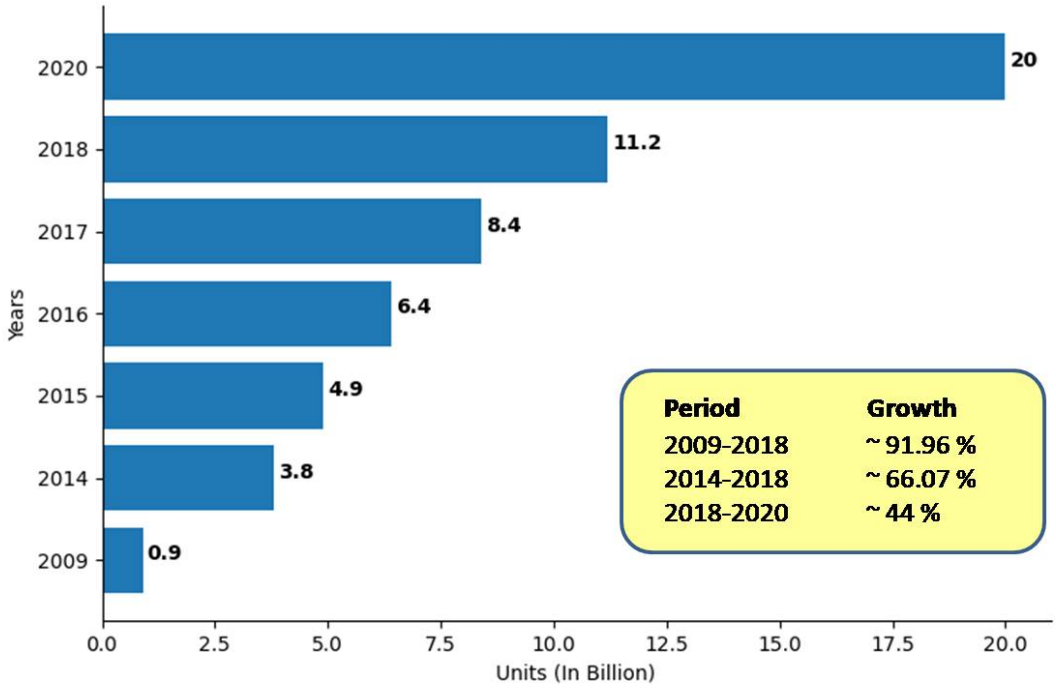


Fig. 2 – Growth of IoE devices

2.2 Network coverage

The requirement for extended network coverage is another key driving force for the introduction of LPWAN technologies. Extended network coverage will be required for some of the IoE applications; for example, smart meters located in the basement of buildings, behind a concrete wall or inside elevators will require an enhanced link budget. Also, the wireless coverage of UAVs for IoE should be extended rapidly and effectively in disaster-affected areas [18]. UAV-aided networks can establish wireless interconnections quickly, which is necessary for achieving larger wireless coverage. Multi-hop Device to Device (D2D) communications can be utilized to achieve larger coverage for UAVs [18]. Link budget and design parameters can be exploited to increase network coverage. The authors in [19] found that NB-IoT 882 MHz and LoRaWAN can increase coverage by up to 398% and 142% respectively with a 10% improvement in receiver sensitivity. They also found that RPMA, NB-IoT and LTE-M incurs at least 9 dB additional path loss relative to Sigfox and LoRaWAN.

2.3 Security and privacy

Security incidents weaken the confidence in the IoE paradigm, hindering its widespread implementation. The disclosure of private and confidential information causes various privacy violations and business disruptions. However, the most significant danger remains the

Table 1 – Average message transaction rate and average message size for different IoE applications

Application	Average Message Transaction Rate (s^{-1})	Average Message Size (bytes)
Roadway Signs	3.33×10^{-2}	1
Traffic Lights or Traffic Sensors	1.67×10^{-2}	1
House Appliances	1.16×10^{-5}	8
Credit Machine in a Shop	5.56×10^{-4}	24
Home Security	1.67×10^{-3}	20
Process Automation	0.2 to 10	40 to 100
Smart Grids	10 to 100	80 to 1000
Road Safety Highway	10	≤ 500
Traffic Efficiency	1	1 K
Urban Intersection	1	1 M/car

threat to people's lives and wellbeing from IoE devices' exposure. Security risks in a healthcare setting, management of traffic lights/connected vehicles may cause accidents leading to fatalities besides causing havoc and increasing pollution [20]. The substantial difference between standard and IoE networks is the resourcefulness of the end devices. In contrast to traditional networks with overflowing resources, IoE devices mostly operate on low power, limited memory, limited computing ability and storage facility. Thus, a balance is required between security and resources as limited resources may restrict enabling technologies to lightweight security algorithms and protocols [21]. Besides, the IoE ecosystem faces diverse data formats and contents due to different application functionalities and the lack of a standard Operating System (OS). They are prone to generic threats such as hardware vulnerabilities, vulnerabilities of social engineering, DoS/ DDoS attacks [21]. Architecture layerwise threats may include eavesdropping, node cloning in the physical layer; unauthorized access, replication of nodes and injection of fake devices in the network layer etc. [21]. Research efforts are made to improve security in IoE networks. Security threats at different layers such as the sensing layer, network layer, middleware layer, gateways and application layer are presented in [22]. The authors in [22] also discussed existing and upcoming solutions to IoE security threats including blockchain, fog computing, edge computing and machine learning. Adoption of Distributed IP Mobility Management (DMM) for 5G networks and affiliated applications is highly predicted [23]. The flat architecture of DMM harmonizes well with 5G networks while overcoming the critical shortcomings of the centralized mobility management technologies such as Mobile IPv6 and Proxy Mobile IPv6 (PMIPv6) [24]. Protecting transmitted data traffic between user mobile devices and their in-home IoT appliances is of paramount importance as the data traffic may include users' sensitive and critical private information. The authors in [24] focused on secure route optimization to enable direct communication between end devices securely while minimizing the possibility of information leakage during data transmission.

2.4 Network scalability

Network scalability will be a key consideration as soon as the market gets bigger. IoE networks will have to support the inclusion of many new heterogeneous devices or exclusion of old devices to sustain market demand in the long run. Applications and functions for the interest of end users without compromising the quality and provision of existing services will have to be addressed too which in turn will put a constraint on network capacity. Network scalability, throughput and/or cell capacity issues have been studied in [25-29]. Transceivers are assumed to undergo high levels of cross and self-technology interference from heterogeneous environments of various wireless technologies and the massive number of IoE devices as LPWAN technologies remain their operations in unlicensed spectrums. Severe interference can potentially degrade network performance and service quality. A high level of interference will increase the Packet Error Ratio (PER) resulting in a loss of reliability. A high number of packet retransmissions might be required under these circumstances.

2.5 Reliability

Reliability is imperative for the safety-critical or mission-critical nature of the IoE applications. The diverse nature of technical requirements for different IoE networks poses a lot of challenges and for some applications, IoE networks are required to simultaneously support high reliability, low latency and massive connectivity [30]. Stringent transmission reliability is required for some applications such as industrial automation, Vehicle to Everything (V2X) networks, and smart grids [30]. Malfunctions of IoE devices, failure to capture critical data, network outage and data loss may result in catastrophic effects, such as mission failure, financial loss, and harm to people and environments [31]. The heterogeneous nature of IoE devices and networks requires diverse reliability protocols. Reliable architecture, operation and application development must address errors in the hardware, the software, interactions with the physical environment, and interactions with the human users [32]. The authors in [33] explored the reliability of the NB-IoT network in intelligent systems.

2.6 Delay

In the context of new 5G use cases, IoE applications have been categorized into two classes: massive Machine Type Communications (mMTC) and Ultra-Reliable Low Latency Communications (URLLC) [34]. mMTC applications will have demands for high network capacity, low-cost end devices and longer battery lifetime. On the other hand, mission-critical applications will rely on URLLC and will demand uninterrupted service with the huge volume of data exchange. M2M communication is widely utilized in a vast number of Industrial Internet of Everything (IIoE) applications. M2M communications in IIoE can be categorized as delay-sensitive and delay-tolerant. The control system in smart manufacturing lines monitors the condition of the manufacturing lines and makes real-time decisions. However, Machine Type Devices (MTDs) such as temperature and humidity sensors in manufacturing factories can tolerate a large delay. The coexistence of delay-sensitive MTDs and delay-tolerate MTDs requires clustering for efficient provisioning of heterogeneous delay requirements [35].

The contention over the limited network radio resources will increase, leading to network congestion with the increasing number of devices. Providing delay-aware channel access in cellular networks is essential for many IoE applications. Node clustering and data aggregation can

play important roles in meeting the various service quality requirements of diverse applications [36]. In this context, a two-hop NOMA-enabled data aggregation architecture was proposed in [36] for massive cellular IoE applications. Moreover, a delay of no more than a few milliseconds is expected in biomedical applications. The authors in [37] discussed task offloading in wireless networks to save energy for devices and reduce the delay of processing tasks in IoE networks. A significant amount of medical data traffic will be produced with extensive use of IoE-based Wireless Body Area Networks (WBANs), leading to an imperative requirement for radio resource management with high utilization efficiency. It will be necessary to offer a priority-based transmission order to guarantee varying medical-grade QoS requirements [38].

2.7 Network deployment cost

Facilitating profitable business cases for IoE requires low device and network deployment costs. A modulo cost of less than $5 is the current industrial target. Capital Expenditure (CAPEX) and Operational Expenditure (OPEX) should be kept at a minimum cost in the pursuit of achieving massive IoE applications and ensuring network connectivity [5]. With the non-uniform distributions of both the applications and humans with sensor devices in Information-Centric IoE (IC-IOE) networks, the information in the urban regions will be redundant and timely information collection in some regions will be challenging. Arranging plenty of static sensor devices will incur unrealistically huge costs for the IC-IoEs [39]. The authors in [40] focused on the design for jointly optimizing downlink and uplink operations to reduce costs in cellular-based IoE networks which provide connections to a massive number of IoE equipment following random access. Cost reduction in LoRa, Sigfox, and NB-IoT networks is also a vital issue, as they too are expected to connect a massive number of IoE equipment [40].

3. IOE ENABLING TECHNOLOGIES

D2D communications, Massive Machine Communications (MMC), Moving Networks (MN), Ultra-Dense Networks (UDN) and ultra-reliable networks are expected to be supported by 5G networks, while MMC forms the the basis of IoE [41, 42]. Low Power Wide Area Networks (LPWANs) are suitable for massive IoE applications and typical applications include logistics, utilities, smart cities, consumer electronics, smart buildings, environment, agriculture and industry. LoRa, Sigfox, Ingenu, Random Phase Multiple Access (RPMA), DASH-7 and Weightless are some potential LPWAN technologies. Some of the traditional solutions like Bluetooth, Wi-Fi, ZigBee, WLAN, Z wave, GSM, LTE can provide wireless connections of the IoE devices in the network. However, these solutions demand high cost, high energy consumption and high complexity. While some of these technologies can support high bandwidth applications, they are unable to provide a larger communication range. IEEE working group 802.11ah enhanced communication development resulting in Bluetooth Low Energy 4.0, ZigBee and Wi-Fi/IEEE802.11 to support short-range communication for MTC [5]. On the other hand, EC-GSM-IoT, NB-IoT, LTE Cat-M1 are cellular-based LPWAN technologies that are intended to address the different IoE application requirements such as long-range, low power consumption, high bandwidth etc. Brief descriptions of some technologies are provided in the following subsections [43].

3.1 Non-cellular-based LPWAN technologies

LoRa: LoRa performs signal modulation in sub-GHz ISM bands using a spread spectrum technique which spreads a narrowband input signal over a wider channel bandwidth [44]. LoRa networks can utilize different data rates ranging from 300 bps to a maximum of 50 kbps and various transmission ranges with different spreading factors. The topology of LoRa networks is star-to-star where end devices communicate with a LoRa Gateway (GW) directly in single-hop using an ALOHA medium access scheme and to combat interference it relies on Frequency Hopping Spread Spectrum (FHSS) [5]. The technology utilizes different channel bandwidths such as 7.8 kHz, 10.4 kHz, 15.6 kHz, 31.2 kHz, 41.7 kHz, 62.5 kHz, 125 kHz, 250 kHz and 500 kHz. LoRaWAN adds a network layer to address network congestion between end devices and central nodes. 868 MHz ISM bands in Europe and 915 MHz bands in North America are used for network operation.

Sigfox: Sigfox utilizes Ultra-Narrowband (UNB) to offer complete end-to-end connectivity. Base stations in Sigfox are configured with cognitive software-defined radios while IP-based network infrastructure is utilized to connect them with backend servers [44]. End devices utilize a Binary Phase Shift Keying (BPSK) modulation scheme in an ultra-narrowband of 100 Hz sub-GHz ISM band carrier to connect themselves to the BS. SigFox operates in different frequency bands such as 868 MHz and 915 MHz. Gaussian Frequency Shift Keying (GFSK) for downlink and Differential Binary Phase Shift Keying (DBPSK) for uplink transmission are used. The maximum packet size of 12 bytes and the maximum throughput of 100 bps limit the number of use cases [44].

3.2 Cellular-based LPWAN technologies

Enhanced Machine Type Communication (eMTC): eMTC also known as LTE Cat-M1 or Cat-M is an enhancement for LTE networks to support MTC applications. This technology was introduced to reduce modem complexity, cost and power consumption while extending coverage [5]. The use of 20 dBm power classes in Cat-M1 enables integration of power amplifiers and through avoiding a dedicated power amplifier achieves a lower device cost. A maximum coupling loss of 155.7 dB can be achieved with eMTC which marks an improvement of 15 dB over LTE base-line of 140.7 dB. Utilizing Power Saving Man-

agement (PSM) and extended Discontinuous Reception (eDRX) like power-saving mechanisms, a long battery life of approximately 10 years for Cat-M1 devices is achieved while using a 5 Watt-Hour battery system.

Narrowband-Internet of Things (NB-IoT): 3GPP Release-13 specification introduced the cellular LPWAN technology NB-IoT, also known as LTE Cat-NB1. Allowing a small fraction of network resources, NB-IoT can coexist with legacy GSM, GPRS and LTE technologies. Cat-NB1 supports a minimum system bandwidth of 180 kHz which allows a GSM operator to replace one GSM carrier of 200 kHz. The maximum data rates are 66 kbps and 16.9 kbps for multi-tone and single-tone uplink transmission respectively. In the case of a downlink transmission, the maximum data rates are 32 kbps and 34 kbps for in-band scenarios and standalone deployment respectively. NB-IoT is seen as a promising technology to meet the huge traffic arising from various IoE applications making it an essential block for the 5G radio network.

Extended Coverage GSM for the Internet of Things (EC-GSM-IoT): EC-GSM-IoT is based on enhanced General Packet Radio Services (eGPRS), introduced by 3GPP standardization in its Release-13 specification. Extended coverage and long employment duration are achieved through the upgradation of GSM networks. Utilizing eDRX, an efficient battery lifetime of 10 years can be achieved. 20 dB coverage extension is achieved with EC-GSM-IoT compared to legacy GPRS networks. EC-GSM-IoT can utilize two different modulation techniques: Eight Phase Shift Keying (8PSK) and Gaussian Minimum Shift Keying (GMSK). EC-GSM-IoT would enable the existing GSM networks to support massive IoE application deployment.

3.3 Short range technologies

Bluetooth: Bluetooth was designed for short-range ad-hoc communication between devices operating in the 2.4 GHz ISM bands and can support data rates in low Mbps. Bluetooth 4.0 improves power consumption and the recent amendment to the standard uses 40 channels with a width of 2 MHz channel spacing. The technology uses GFSK for modulation, and FHSS to combat interference and multipath fading. Increased interest in developing the architecture for mesh networking can overcome the major drawback of Bluetooth which is a one-to-one communication between only two devices at a time.

IEEE 802.15.4 and ZigBee: IEEE 802.15.4 is the de facto standard for Low Rate Wireless Personal Area Networks (LR-WPAN). Network operation is performed in either 868 MHZ or 914 MHz or 2.4 GHz band. Direct Sequence Spread Spectrum (DSSS) is used as the modulation scheme in IEEE 802.15.4. The maximum supported data rate is 250 kbps. A network layer on top of IEEE 802.15.4 physical and data link layer by ZigBee. ZigBee uses Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) for channel access and can support star, mesh, cluster tree topologies.

Wi-Fi: Wi-Fi is Wireless Local Area Network (WLAN) technology that belongs to the IEEE 802.11 standard series. It operates within 5 GHz and 2.4 GHz ISM spectrum bands. This technology provides high throughput connectivity between devices located nearby. Low-power Wi-Fi, which is also called IEEE 802.11ah is intended to serve a massive number of nodes distributed in a larger coverage area while consuming less power. The new standard targets approximately 100's of milliwatts of energy consumption for end devices and a data rate up to 347 Mbps which would enable it to be used in different IoE applications such as parking metering, autonomous lightning, smart security etc.

Some of the important features of IoE enabling technologies are summarized in Table 2 [5, 45].

4. IOE APPLICATIONS AND ENABLING TECHNOLOGIES

The communication range of Wi-Fi/Bluetooth is much smaller than other IoE technologies and therefore limiting the possible IoE use cases. Some potential IoE applications of Bluetooth and Wi-Fi can be found in [46, 47]. Personal activity, local object tracking, hospital asset tracking and point of sale could be some of the possible application scenarios of Wi-Fi/Bluetooth. Some of the conceivable applications for ZigBee are waste management systems, warehouse logistics, home automation [4, 48]. Wi-Fi/Bluetooth/ZigBee are suitable candidates for short-range high throughput applications while Wi-Fi/Bluetooth can be also used for applications that require low latency and high reliability.

The well-established global ecosystem is a distinct advantage for cellular-based IoE enabling technologies. NB-IoT, eMTC, EC-GSM-IoT are more likely to lead the high throughput/low latency applications market. They would also be able to scale up/scale down the network capacity according to market demands. However, spectrum sharing for IoE applications in the cellular domain is a challenging issue as it can hamper the existing applications. Resource optimization will be challenging too since the IoE application requirements might vary from the requirements of existing cellular channels. Smart surveillance/smart automatic driving/smart transportation [5], connected car/fleet management/remote health monitoring/smart metering [4] etc. are some of the potential IoE applications of cellular-based LPWAN technologies.

Non-cellular-based LPWAN technologies are more appropriate for IoE applications requiring low data rates with a long communication range, where reliability and mobility are not among the core priorities. Sigfox outdoor localization system [49] and LoRa sailing monitoring system are studied in [50]. While SigFox provides a larger range, LoRa provides more flexibility in terms of data rate as reported in different papers. Also, LoRa has 500 ms one-hop latency while Sigfox has 2s latency [51]. A DASH7 power metering system is analyzed in [52]. Smart cities, smart buildings, smart grids, and oil and gas pipelines are some

Table 2 – IoE enabling technologies and features

Technology	Frequency Band	Range	Maximum Date Rate	Channel Bandwidth	Security	Reliability	Latency
LoRa	868 MHz, 915 MHz	15 km	50 kbps	125, 250, 500 kHz	Low	Low	High
SigFox	915 to 928 MHz	20 km+	100 bps	100 Hz	Low	Low	High
eMTC	$700-900$ MHz	<15 km	1 Mbps	1.08 MHz (1.4 MHz carrier bandwidth)	Medium/ High	Medium/ High	Low
NB-IoT	$700-900$ MHz	<35 km	DL: 170 kbps UL: 250 kbps	180 kHz (200 kHz carrier bandwidth)	Medium/ High	Medium/ High	Low
EC-GSM-IoT	$800-900$ MHz	<15 km	74 kbps (GMSK), 240 kbps (8 PSK)	0.2 MHz	Medium/ High	Medium/ High	Low
Bluetooth	2.4 GHz	50 m	2 Mbps	2 MHz	Low	Medium/ High	Low
ZigBee	868 MHz, 915 MHz, and 2.4 GHz	Typically less than 1 km	250 kbps	2 MHz	Low	Low	High
Wi-Fi	2.4 GHz, 5 GHz	100 m	54 Mbps	22 MHz	Medium/ High	Medium/ High	Low

potential application domains for non-cellular-based LPWAN technologies.

IoE will also play a major role in industrial automation in the near future [53, 54]. Most industrial automation applications require high reliability and low latency. Small scale networks such as Wireless Highway Addressable Remote Transducer (WirelessHART), Wireless Interface for Sensors and Actuators (WISA), and Wireless Networks for Industrial Automation for Process Automation (WIA-PA), which are based on the IEEE 802.15.4 standard, and the WIAFA [4], which is based on the IEEE 802.11 standards are typically used in industrial automation [30]. However, they do not meet the high scalability and reliability requirements required by many applications. In some industrial applications, the wireless transmission should potentially guarantee the PER around 10^{-9} within the transmission delay constraint as low as 10 µs [30] which may be difficult for many LPWAN technologies. The current state of the art of different technologies and research studies suggest cellular-based LPWAN technologies are the most suitable candidates for industrial automation applications. URLLC is one of the most important features of the 5G mobile network. Thus, cellular-based technologies may be able to meet some of the industrial automation application criteria. The typical data size of a packet in an industrial setting is only a few bytes with different update frequency, latency and reliability requirements while the typical communication range is very low. Some valuable insights can be obtained from [4, 34, 55]. ZigBee and Wi-Fi could also be suitable for some industrial applications as well [4].

IoE networks are expected to play a crucial role in improving transportation capability and efficiency. Some communication scenarios for V2X networks are 1) Vehicle to Vehicle (V2V) communications, in which information is exchanged between vehicles; 2) Vehicle to Infrastructure (V2I) communications, which occur between vehicles and Roadside Units (RSUs), traffic lights, and base stations; 3) Vehicle to Pedestrian (V2P) communications, in which vehicles communicate with people who are along the side of the road; and 4) Vehicle to Network (V2N), where the vehicles connect to an entity in the networks e.g., a backend server or a traffic information system [30]. However, the requirements on latency and reliability are very high for V2X networks. Some basic requirements for V2X communication networks are low latency, high reliability, high throughput, interference-robust, communication range and mobility support. It is expected that 5G cellular networks will play an important role in this application domain.

Currently, most of the LPWAN technologies use a star topology and rely on wired infrastructure (e.g., cellular LPWANs) or Internet (e.g., LoRaWAN) to integrate multiple networks to cover large areas. The adoption of LPWAN technologies in rural and remote area applications such as agricultural IoE and industrial IoE (e.g., for oil/gas fields) that may cover large areas is challenging. Some technologies for achieving last-mile connectivity have been discussed in [56]. Cellular networks can be an efficient last-mile solution for rural areas due to significant cellular penetration in many rural areas across the world. Although WiFi is a mature technology, the IEEE 802.11 MAC protocol gives poor end-to-end performance for long-range communication. Femtocell, which uses a small low-power cellular base station, can be used to provide cost-effective cellular connectivity within its

Table 3 – Possible technologies for some IoE application scenarios

Application	Application Requirements	Possible Technologies
Structural Health (Smart City)	Tolerable delay: 30 min, update frequency: 10 min, data rate: low	LoRaWAN, SigFox, LTE, NB-IoT, ZigBee
Waste Management (Smart City)	Tolerable delay: 30 min, update frequency: 1 hour, data rate: low	LoRaWAN, SigFox, LTE, NB-IoT
Video Surveillance (Smart City)	Tolerable delay: seconds, update frequency: real-time, data rate: high, netwaork coverage: large/small	LTE, NB-IoT, WiFi
Air Quality Monitoring (Smart Home)	Tolerable delay: 5 min, update frequency: 30 min, data rate: low	Wi-Fi, Bluetooth, NB-IoT
Patients Healthcare Delivery and Monitoring (Healthcare)	Tolerable delay: seconds, update frequency: 1 report per hour/day, data rate: high security: high, reliability: high	Bluetooth, LTE, NB-IoT
Real-time Emergency Response and Remote Diagnostics (Healthcare)	Tolerable delay: seconds, update frequency: ad-hoc emergency communication, data rate: high security: high, reliability: high	Bluetooth, LTE, NB-IoT
Smart Grids (Industrial)	Tolerable delay: 3 to 20 ms, update frequency: 10 to 100 ms, reliability: 10^{-6} PLR, network coverage: a few meter to kilometers	WiFi, ZigBee, LTE, WiMAX, NB-IoT
Road Safety Highway (Smart City)	Tolerable delay: 10 to 100 ms, update frequency: 100 ms, reliability: 10^{-3} to 10^{-5} PLR, network coverage: 2000 m	LTE, NB-IoT
Factory Automation (Industrial)	Tolerable delay: 0.25 to 10 ms, update frequency: 0.5 to 50 ms, reliability: 10^{-9} PLR, network coverage: 50 to 100 m	LTE, NB-IoT, WiFi
Manufacturing Cell (Industrial)	Tolerable delay: 5 ms, update frequency: 50 ms, reliability: 10^{-9} PLR, network coverage: 50 to 100 m	LTE, NB-IoT, WiFi
Process Automation (Industrial)	Tolerable delay: 50 to 100 ms, update frequency: 100 to 5000 ms, reliability: 10^{-3} to 10^{-4} PLR, network coverage: 100 to 500 m	LTE, NB-IoT

coverage range. High user mobility and extended battery life of mobile terminals can be achieved using LTE. WiMAX supports broadband applications as well as providing large coverage, and deployment of a WiMAX network is much cheaper than the deployment of an LTE network for last-mile connectivity. Cognitive radio technologies can achieve large coverage with non-LoS links in last-mile connectivity in rural areas utilizing unused licensed spectrum.

The application of IoE promises smart, innovative and comfortable medical services to the patients and/or individuals needing healthcare, and furnishes their class of life through easing emergency medical support, security and continuous care [57]. The prospective applications of IoE in medical sectors include health monitoring using wearable devices that measure the physical activities/behaviour [58], supporting health-related information for regular patient care, and networking through devices for clinical care with issues of an unvarying electrocardiogram, blood oxygen and blood pressure [59]. IoE can lead to constructing big data on a particular health issue and can play a pivotal role in the further progress of IoE through the analysis and application of big data. Also, IoE has the prospect of on-time medical assistance by connecting the network to traffic and hospital administration in case of an accident. Moreover, IoE also supports the electronic reporting of patients' mobility (i.e. contact tracing) to ensure homecare. IoE-coupled smart wearable devices/systems have been reported for monitoring cardiovascular disease, Chronic Obstructive Pulmonary Disease (COPD), Parkinson's disease, pregnancy and cognitive disorder. Usually, the acquired data (biomarkers such as ECG, respiratory rate, body temperature, EMG muscle activity, gait and others) using sensory devices are transmitted to the Healthcare Organization (HCO) using the intermediate concentrators and platforms connected with short-range radio such as Zigbee or low-power Bluetooth under the governance of a smartphone's WiFi or cellular data connection [60].

Some IoE applications, typical requirements and possible technologies are shown in Table 3. Wearables and smart metering are two potential IoE application areas. The networking technologies used in these technologies are discussed in the following subsections.

4.1 Wearables in healthcare

In today's digital world the term "wearable" refers to accessories such as a smartwatch on a business executive's wrist, a head-mounted display worn by an immersive gamer, a tiny sensor on a cyclist's helmet, or a smart garment a runner uses to track and monitor his steps [61]. The ability of sensing comes from the embedded sensors in wearables. The functional attributes such as multi-functionality, configurability, responsiveness and bandwidth depend on the nature of an application. Currently, two industry giants, Apple and Google dominate the wearable technology market by-products released [62]. The seamless integration of wearables in healthcare settings will have to ensure compatibility with existing wireless technologies and established operational protocols in these settings. Sensor Platform for Healthcare in Residential Environment (SPHERE) is a multi-modal platform of non-medical sensors for behaviour monitoring in residential environments that utilize inherently cost-efficient and scalable IoE technologies [63, 64]. The original health evidence is collected from the physiological signals of a human body using diverse biosensors. These biosensors can be deployed in an implantable (in-body), wearable (on-body), portable (off-body) or environmental modality. The home environment and the resident interaction with the environment are monitored in a Home (SH) by a system of pervasive information and communication technologies consisting of sensor systems.

Enabling the sensing platform for remote monitoring requires networking technologies to provide ubiquitous network connectivity between residents and clinicians. LTE and Bluetooth are possible networking solutions for medical sensors as the application requires low latency, high reliability and low capacity [65]. Energy-efficient, IP-enabled sensing networks can allow access to existing Internet infrastructures removing the need for translation gateways or proxies in hardware and software. It will improve the user experience and require less maintenance effort. Although WiFi has the significant advantage of being Internet Protocol (IP) enabled, the hardware used in WiFi connectivity consumes relatively more power and therefore, less suitable for long-term deployments of an application that utilizes battery-powered sensor nodes. 6LoWPAN has better support for multi-hop mesh and thus, it was selected for the environmental sensor network and data forwarding in SPHERE [63]. On the other hand, BLE was chosen for collecting the data from the wearable nodes for being more convenient. SPHERE uses IPv6 on top of the IEEE 802.15.4 TSCH protocol to provide time synchronization to the network and ensure time-stamping all of sensor data with high accuracy. ZigBee was used in the first version of the SPHERE. However, ZigBee uses a single channel at a time and does not have time slots. WiGig products based on IEEE 802.11ad may replace Bluetooth and WiFi at some point in future for applications with high throughput requirements as Bluetooth and WiFi have very limited scaling capability.

4.2 Smart metering

Advanced Metering Infrastructure (AMI) is an integral part of Smart Grids (SGs) and smart metering is one of the most promising applications of IoE. AMI, besides enabling accurate consumer billing in the presence of dynamic pricing and improving efficiency and reliability of electricity distribution in the presence of distributed generation, will be used in water and gas utility distribution networks in smart cities as an application of IoE. Renewable energy producers and mobile energy storage can be linked and utilized by SGs' infrastructure. AMI communication networks can be divided into Home Area Networks (HANs), Neighborhood Area Networks (NANs) or Meter Local Area Networks (MLAN) and Wide Area Networks (WANs) [66, 67]. Connections among distributed energy resources, GWs, Electric Vehicles (EVs), Smart Meters (SMs), etc. are provided by the HANs. SMs that need to send their data to the corresponding data concentrator are facilitated by the NANs or MLAN. Appliances such as entertainment systems, lighting systems, energy storage and EVs constitute HANs and SMs act as home GWs that link the HANs with the NANs [68]. Connections between some data concentrators and the central system are provided by WANs.

The choice of a suitable technology in AMI depends on application requirements such as security, privacy, bandwidth, latency, reliability, energy efficiency etc. Power Line Communications (PLC) and wireless communications are widely used in SGs as the overall system reliability can be enhanced by exploiting the diversity achieved from the simultaneous transmission of the same signal over power lines and wireless links. Wireless Sensor Networks (WSNs) are attractive solutions for AMI because of their low-cost deployment and multiple functionalities. However, one of the challenging tasks for WSNs is to ensure QoS requirements for AMI applications. Typically, SMs are connected to the Distribution System Operators' (DSO) backend system in two ways: 1) a concentrator gathers the data from the SMs in its neighbourhood using Wi-Fi or PLC connections and then relays it using cellular or a wired connection to the DSO backend, or 2) Each SM sends data to the DSO backend using a cellular network [69]. IEEE 802.15.4 (e.g., ZigBee and Zwave), IEEE 802.11 (WiFi) are some of the technologies used in HANs [66]. Although PLC has been the primary choice for communication between the SMs and data concentrators, wireless mesh networks in AMI have been proposed and deployed widely. The use of LTE as a NAN technology was discussed in [68]. Some of the potential WAN technologies are IEEE 802.16 (i.e., WiMAX), IEEE 802.20 (MobileFi), PLC, IEEE 802.11 (WiFi) and IEEE 802.15.4 (ZigBee) [66]. LoRaWAN can be used in applications with relaxed QoS requirements such as latency tolerant services of a Power Wireless Private Network (PWPN) [70].

5. CONCLUSIONS

The paper discussed IoE application requirements such as latency, energy efficiency, data rate, reliability, security and communication range. Features, advantages and disadvantages of short-range, cellular and non-cellular-based IoE enabling technologies are presented as well. It is evident from the discussion that choosing a particular IoE enabling technology depends on the specific application. It is also possible that specific application requirements are met through more than one existing technology. However, it is very likely that among the potential technologies, one technology performs better than others with a priority list of key network performance indicators. Understanding the application requirements and technological features will play a key role in determining the most suitable IoE enabling technology for a particular application.

ACKNOWLEDGEMENT

This project was supported in part by the University Research Center, Shahjalal University of Science and Technology under the research grant: AS/2020/1/31.

REFERENCES

[1] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015), Internet of things: A survey on enabling technologies, protocols, and applications, IEEE communications surveys & tutorials, 17(4), 2347-2376.

[2] Saxena, N., Roy, A., Sahu, B. J., & Kim, H. (2017), Efficient IoT gateway over 5G wireless: A new design with prototype and implementation results, IEEE Communications Magazine, 55(2), 97-105.

[3] Dhillon, H. S., Huang, H., & Viswanathan, H. (2017), Wide-area wireless communication challenges for the Internet of Things, IEEE Communications Magazine, 55(2), 168-174.

[4] Stankovic, J. A. (2014), Research directions for the internet of things, IEEE Internet of Things Journal, 1(1), 3-9.

[5] Akpakwu, G. A., Silva, B. J., Hancke, G. P., & Abu-Mahfouz, A. M. (2017), A survey on 5G networks for the Internet of Things: Communication technologies and challenges, IEEE access, 6, 3619-3647.

[6] Stefanizzi, M. L., Mottola, L., Mainetti, L., & Patrono, L. (2017), COIN: Opening the internet of things to people's mobile devices, IEEE Communications Magazine, 55(2), 20-26.

[7] Chen, S., Liu, T., Gao, F., Ji, J., Xu, Z., Qian, B., & Guan, X. (2017), Butler, not servant: A human-centric smart home energy management system, IEEE Communications Magazine, 55(2), 27-33.

[8] Martella, C., Cattani, M., & Van Steen, M. (2017), Exploiting density to track human behavior in crowded environments, IEEE Communications Magazine, 55(2), 48-54.

[9] Jiang, J., Han, G., Zhu, C., Chan, S., & Rodrigues, J. J. (2017), A trust cloud model for underwater wireless sensor networks, IEEE Communications Magazine, 55(3), 110-116.

[10] https://www.statista.com/statistics/764026/number-of-iot-devices-in-use-worldwide/ (Accessed: 27 Feb, 2021)

[11] Yao, Y., Zhu, Z., Huang, S., Yue, X., Pan, C., & Li, X. (2019), Energy Efficiency Characterization in Heterogeneous IoT System With UAV Swarms Based on Wireless Power Transfer, IEEE Access, 8, 967-979.

[12] Paul, B. (2020), A Novel Mathematical Model to Evaluate the Impact of Packet Retransmissions in LoRaWAN, IEEE Sensors Letters, 4(5), 1-4.

[13] Wu, Q., Chen, W., Ng, D. W. K., & Schober, R. (2018), Spectral and energy-efficient wireless powered IoT networks: NOMA or TDMA?, IEEE Transactions on Vehicular Technology, 67(7), 6663-6667.

[14] Paul, B. (2020), A Novel Energy-Efficient Routing Scheme for LoRa Networks, IEEE Sensors Journal, 20(15), 8858-8866.

[15] Nguyen, T. D., Khan, J. Y., & Ngo, D. T. (2018), A distributed energy-harvesting-aware routing algorithm for heterogeneous IoT networks, IEEE Transactions on Green Communications and Networking, 2(4), 1115-1127.

[16] Sen, S., Koo, J., & Bagchi, S. (2018), TRIFECTA: security, energy efficiency, and communication capacity comparison for wireless IoT devices, IEEE Internet Computing, 22(1), 74-81.

[17] Alzahrani, B., & Ejaz, W. (2018), Resource management for cognitive IoT systems with RF energy harvesting in smart cities, IEEE Access, 6, 62717-62727.

[18] Liu, X., Li, Z., Zhao, N., Meng, W., Gui, G., Chen, Y., & Adachi, F. (2018), Transceiver design and multihop D2D for UAV IoT coverage in disasters, IEEE Internet of Things Journal, 6(2), 1803-1815.

[19] Ikpehai, A., Adebisi, B., Rabie, K. M., Anoh, K., Ande, R. E., Hammoudeh, M., ... & Mbanaso, U. M. (2018), Low-power wide area network technologies for internet-of-things: A comparative review, IEEE Internet of Things Journal, 6(2), 2225-2240.

[20] Neshenko, N., Bou-Harb, E., Crichigno, J., Kaddoum, G., & Ghani, N. (2019), Demystifying IoT security: an exhaustive survey on IoT vulnerabilities and a first empirical look on internet-scale IoT exploitations, IEEE Communications Surveys & Tutorials, 21(3), 2702-2733.

[21] Iqbal, W., Abbas, H., Daneshmand, M., Rauf, B., & Bangash, Y. A. (2020), An In-Depth Analysis of IoT Security Requirements, Challenges, and Their Countermeasures via Software-Defined Security, IEEE Internet of Things Journal, 7(10), 10250-10276.

[22] Hassija, V., Chamola, V., Saxena, V., Jain, D., Goyal, P., & Sikdar, B. (2019), A survey on IoT security: application areas, security threats, and solution architectures, IEEE Access, 7, 82721-82743.

[23] Giust, F., Cominardi, L., & Bernardos, C. J. (2015), Distributed mobility management for future 5G networks: overview and analysis of existing approaches, IEEE Communications Magazine, 53(1), 142-149.

[24] Shin, D., Yun, K., Kim, J., Astillo, P. V., Kim, J. N., & You, I. (2019), A security protocol for route optimization in DMM-based smart home IoT networks, IEEE Access, 7, 142531-142550.

[25] Mikhaylov, K., Petaejaejaervi, J., & Haenninen, T. (2016, May), Analysis of capacity and scalability of the LoRa low power wide area network technology, In European Wireless 2016; 22th European Wireless Conference; Proceedings of (pp. 1-6). VDE.

[26] Alsohaily, A., Sousa, E., Tenenbaum, A. J., & Maljevic, I. (2017, October), LoRaWAN radio interface analysis for North American frequency band operation, In Personal, Indoor, and Mobile Radio Communications (PIMRC), 2017 IEEE 28th Annual International Symposium on (pp. 1-6). IEEE.

[27] Reynders, B., Wang, Q., Tuset-Peiro, P., Vilajosana, X., & Pollin, S. (2018), Improving Reliability and Scalability of LoRaWANs Through Lightweight Scheduling, IEEE Internet of Things Journal.

[28] Rahman, M., & Saifullah, A. (2020), Integrating low-power wide-area networks for enhanced scalability and extended coverage, IEEE/ACM Transactions on Networking, 28(1), 413-426.

[29] Adame, T., Bel, A., & Bellalta, B. (2019), Increasing LPWAN scalability by means of concurrent multiband IoT technologies: an industry 4.0 use case, IEEE Access, 7, 46990-47010.

[30] Ma, Z., Xiao, M., Xiao, Y., Pang, Z., Poor, H. V., & Vucetic, B. (2019), High-reliability and low-latency wireless communication for internet of things: challenges, fundamentals, and enabling technologies, IEEE Internet of Things Journal, 6(5), 7946-7970.

[31] Xing, L. (2020), Reliability in Internet of Things: Current status and future perspectives, IEEE Internet of Things Journal, 7(8), 6704-6721.

[32] Bagchi, S., Abdelzaher, T. F., Govindan, R., Shenoy, P., Atrey, A., Ghosh, P., & Xu, R. (2020), New Frontiers in IoT: Networking, Systems, Reliability, and Security Challenges, IEEE Internet of Things Journal, 7(12), 11330-11346.

[33] Jia, G., Zhu, Y., Li, Y., Zhu, Z., & Zhou, L. (2019), Analysis of the Effect of the Reliability of the NB-Iot Network on the Intelligent System, IEEE Access, 7, 112809-112820.

[34] Schulz, P., Matthe, M., Klessig, H., Simsek, M., Fettweis, G., Ansari, J., & Puschmann, A. (2017), Latency critical IoT applications in 5G: Perspective on the design of radio interface and network architecture, IEEE Communications Magazine, 55(2), 70-78.

[35] Zhang, C., Sun, X., Zhang, J., Wang, X., Jin, S., & Zhu, H. (2019), Throughput optimization with delay guarantee for massive random access of M2M communications in industrial IoT, IEEE Internet of Things Journal, 6(6), 10077-10092.

[36] Moussa, H. G., & Zhuang, W. (2019), Energy-and delay-aware two-hop NOMA-enabled massive cellular IoT communications, IEEE Internet of Things Journal, 7(1), 558-569.

[37] Deng, Y., Chen, Z., Yao, X., Hassan, S., & Ibrahim, A. M. (2019), Parallel offloading in green and sustainable mobile edge computing for delay-constrained IoT system, IEEE Transactions on Vehicular Technology, 68(12), 12202-12214.

[38] Yi, C., & Cai, J. (2018), A truthful mechanism for scheduling delay-constrained wireless transmissions in IoT-based healthcare networks, IEEE Transactions on Wireless Communications, 18(2), 912-925.

[39] Li, T., Ota, K., Wang, T., Li, X., Cai, Z., & Liu, A. (2019), Optimizing the coverage via the UAVs with lower costs for information-centric Internet of Things, IEEE Access, 7, 15292-15309.

[40] Kwon, T., Choi, S. W., & Shin, Y. H. (2019), A comprehensive design framework for network-wide cost reduction in random access-based wireless IoT networks, IEEE Communications Letters, 23(9), 1576-1580.

[41] Agiwal, M., Roy, A., & Saxena, N. (2016), Next generation 5G wireless networks: A comprehensive survey, IEEE Communications Surveys & Tutorials, 18(3), 1617-1655.

[42] Buzzi, S., Chih-Lin, I., Klein, T. E., Poor, H. V., Yang, C., & Zappone, A. (2016), A survey of energy-efficient techniques for 5G networks and challenges ahead, IEEE Journal on Selected Areas in Communications, 34(4), 697-709.

[43] Paul, B. (2019). Impacts of Multi-Hop Routing and Channel/Transmission Configuration Planning on LoRa Networks (MSc dissertation, University of Saskatchewan).

[44] Raza, U., Kulkarni, P., & Sooriyabandara, M. (2017), Low power wide area networks: An overview, IEEE Communications Surveys & Tutorials, 19(2), 855-873.

[45] Poursafar, N., Alahi, M. E. E., & Mukhopadhyay, S. (2017, December), Long-range wireless technologies for IoT applications: A review, In Sensing Technology (ICST), 2017 Eleventh International Conference on (pp. 1-6). IEEE.

[46] Alapetite, A., & Hansen, J. P. (2016, December), Dynamic Bluetooth beacons for people with disabilities, In Internet of Things (WF-IoT), 2016 IEEE 3rd World Forum on (pp. 36-41). IEEE.

[47] Vikram, N., Harish, K. S., Nihaal, M. S., Umesh, R., Shetty, A., & Kumar, A. (2017, January), A low cost home automation system using wi-fi based wireless sensor network incorporating Internet of Things (IoT), In Advance Computing Conference (IACC), 2017 IEEE 7th International (pp. 174-178). IEEE.

[48] Karthikeyan, S., Rani, G. S., Sridevi, M., & Bhuvaneswari, P. T. V. (2017, May), IoT enabled waste management system using ZigBee network, In Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2017 2nd IEEE International Conference on (pp. 2182-2187). IEEE.

[49] Ribeiro, G. G., de Lima, L. F., Oliveira, L., Rodrigues, J. J., Marins, C. N., & Marcondes, G. A. (2018, June), An Outdoor Localization System Based on SigFox, In 2018 IEEE 87th Vehicular Technology Conference (VTC Spring) (pp. 1-5). IEEE.

[50] Li, L., Ren, J., & Zhu, Q. (2017, February), On the application of LoRa LPWAN technology in Sailing Monitoring System, In Wireless On-demand Network Systems and Services (WONS), 2017 13th Annual Conference on (pp. 77-80). IEEE.

[51] Morin, E., Maman, M., Guizzetti, R., & Duda, A. (2017), Comparison of the device lifetime in wireless networks for the internet of things, IEEE Access, 5, 7097-7114.

[52] Cetinkaya, O., & Akan, O. B. (2015, January), A DASH7-based power metering system, In Consumer Communications and Networking Conference (CCNC), 2015 12th Annual IEEE(pp. 406-411). IEEE.

[53] Wollschlaeger, M., Sauter, T., & Jasperneite, J. (2017), The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0., IEEE Industrial Electronics Magazine, 11(1), 17-27.

[54] Delsing, J. (2017), Local cloud internet of things automation: Technology and business model features of distributed internet of things automation solutions, IEEE Industrial Electronics Magazine, 11(4), 8-21.

[55] Lennvall, T., Gidlund, M., & Åkerberg, J. (2017, September), Challenges when bringing IoT into industrial automation, In AFRICON, 2017 IEEE (pp. 905-910). IEEE.

[56] Nandi, S., Thota, S., Nag, A., Divyasukhananda, S., Goswami, P., Aravindakshan, A., ... & Mukherjee, B. (2016), Computing for rural empowerment: enabled by last-mile telecommunications, IEEE Communications Magazine, 54(6), 102-109.

[57] Gatouillat, A., Badr, Y., Massot, B., & Sejdić, E. (2018), Internet of medical things: A review of recent contributions dealing with cyber-physical systems in medicine, IEEE Internet of Things Journal, 5(5), 3810-3822.

[58] Son, D., Lee, J., Qiao, S., Ghaffari, R., Kim, J., Lee, J. E., ... & Kim, D. H. (2014), Multifunctional wearable devices for diagnosis and therapy of movement disorders, Nature nanotechnology, 9(5), 397.

[59] Paradiso, R., Loriga, G., & Taccini, N. (2005), A wearable health care system based on knitted integrated sensors, IEEE transactions on Information Technology in biomedicine, 9(3), 337-344.

[60] Hu, F., Xie, D., & Shen, S. (2013, August). Hu, F., Xie, D., & Shen, S. (2013, August), On the application of the internet of things in the field of medical and health care, In 2013 IEEE international conference on green computing and communications and IEEE Internet of Things and IEEE cyber, physical and social computing (pp. 2053-2058). IEEE.

[61] Park, S., & Jayaraman, S. (2021), Wearables: Fundamentals, advancements, and a roadmap for the future, In Wearable sensors (pp. 3-27). Academic Press.

[62] Pyattaev, A., Johnsson, K., Andreev, S., & Koucheryavy, Y. (2015), Communication challenges in high-density deployments of wearable wireless devices, IEEE Wireless Communications, 22(1), 12-18.

[63] Elsts, A., Fafoutis, X., Woznowski, P., Tonkin, E., Oikonomou, G., Piechocki, R., & Craddock, I. (2018), Enabling healthcare in smart homes: the SPHERE IoT network infrastructure, IEEE Communications Magazine, 56(12), 164-170.

[64] Zhu, N., Diethe, T., Camplani, M., Tao, L., Burrows, A., Twomey, N., ... & Craddock, I. (2015), Bridging e-health and the internet of things: The sphere project, IEEE Intelligent Systems, 30(4), 39-46.

[65] Sun, H., Zhang, Z., Hu, R. Q., & Qian, Y. (2018), Wearable communications in 5G: challenges and enabling technologies, IEEE vehicular technology magazine, 13(3), 100-109.

[66] Kumar, P., Lin, Y., Bai, G., Paverd, A., Dong, J. S., & Martin, A. (2019), Smart grid metering networks: A survey on security, privacy and open research issues, IEEE Communications Surveys & Tutorials, 21(3), 2886-2927.

[67] Ye, F., Qian, Y., Hu, R. Q., & Das, S. K. (2015), Reliable energy-efficient uplink transmission for neighborhood area networks in smart grid, IEEE Transactions on Smart Grid, 6(5), 2179-2188.

[68] Anjana, K. R., & Shaji, R. S. (2018), A review on the features and technologies for energy efficiency of smart grid, International Journal of Energy Research, 42(3), 936-952.

[69] Bou-Harb, E., Fachkha, C., Pourzandi, M., Debbabi, M., & Assi, C. (2013), Communication security for smart grid distribution networks, IEEE Communications Magazine, 51(1), 42-49.

[70] Bao, L., Wei, L., Jiang, C., Miao, W., Guo, B., Li, W., ... & Zou, J. (2018), Coverage analysis on NB-IoT and LoRa in power wireless private network, Procedia computer science, 131, 1032-1038.

AUTHORS

Biswajit Paul completed his BSc in Electronics and Telecommunication Engineering from North South University, Bangladesh with the distinction of Summa Cum Laude and MSc from University of Saskatchewan, Canada. He started his career as a Lecturer at Leading University and later joined Shahjalal University of Science and Technology (SUST). Currently he is an Associate Professor in the Department of Electrical and Electronic Engineering at SUST. He was the Founder Chairman of IEEE Student Branch, NSU. So far, he has published a few referred international journal and conference papers. He also serves as a reviewer for some prestigious international journals.

Gokul Chandra Biswas achieved his B.Sc. in agriculture and MS in biotechnology from the Bangladesh Agricultural University, Bangladesh in 2007 and 2009 respectively. He received his Ph.D. in Nano-Science and Nano-Technology in 2017 from the University of Tsukuba, Japan. In 2010, He joined Bangladesh Agricultural Research Institute as scientist. In 2011, he moved as lecturer to Shahjalal University of Science and Technology (SUST), Bangladesh. Since 2017, he is an associate professor of the Department of Genetic Engineering and Biotechnology, SUST. His research interest is on the point-of-care diagnostics, micro-total-analysis-system (μTAS), biomedical engineering and evidence-based disease surveillance. He has published many peer-reviewed articles in reputed international journals. He became one of the founding members of Bangladesh Nano Society in 2020. He has also membership for several professional scientific communities of life science research arenas.

Habib F. Rashvand, CEng, LIEEE received his distinguished engineering BE and Postgrad Diploma qualifications from the University of Tehran in 1970 and 1971. Then, selected for training to head a major research and development operation visited Japan for representing the University and Iranian PTT in a two-years association with NTT, KTT and other Japanese industries; and, headed an international project for building the Telecom Research Centre (ITRC) as a distinct national resources of the country. His Doctorate of Philosophy from the University of Kent in 1980 shows his appetite for his contributions to the new world of data communications and the Internet with his industrial presence in high-speed modems in the 1980s and in the 1990s in mobile and wireless technologies until his professorship on 'Networks, Systems & Protocols' granted in 2001 by the German Ministry of Education. His rich blend of 30 continuous years of industrial and academic research and development involving international industries including Racal, Vodafone, Nokia and Cable & Wireless at various senior positions worked and collaborated with a wide range of academies including University of Tehran, University of Zambia, Portsmouth University, Southampton University, Warwick & Coventry Universities, Open University and Magdeburg German Universities. He has direct consultation experience as Editor-in-Chief(s) and Guest Editor(s) for IEE, IET and IEEE for editorial issues and research journals for over 15 years. He has presented many prestigious keynotes and and has been invited as a guest speaker, and has managed well over 100 cooperative projects, five books and over 100 research papers and book chapters. His technical books are titled: Distributed Sensor Systems (Wiley 2012), Using Cross-Layer Techniques for Communication Systems (IGI Global 2012ed), Dynamic Ad Hoc Networks (IET 2013ed), Wireless Sensor Systems (Wiley 2017ed), and Design Solutions for Wireless Sensor Networks in Extreme Environments (Artech House 2019). Since 2004, seeking for a technology oriented innovative ICT solutions paradigm for a better future of humanity in association with University of Warwick, heading a special operation as the Director of Advanced Communication Systems involving global academics, innovative industries and professional institutions, he has dedicated his time to help build a sustainable future global village.

FROM DESIGN TO PROTOTYPING IN THE INTERNET OF THINGS: A DOMOTICS CASE STUDY

Sabrina Sicari[1], Alessandra Rizzardi[1], Alberto Coen-Porisini[1]
[1]Dipartimento di Scienze Teoriche e Applicate, Università degli Studi dell'Insubria, via O. Rossi 9 - 21100 Varese (Italy)

NOTE: Corresponding author: Sabrina Sicari, sabrina.sicari@uninsubria.it

Abstract *– Nowadays, the capability of rapidly designing and prototyping, simple, yet real domotics systems (e.g., smart homes and smart buildings applications) is even more compelling, due to the availability and increasing spread of Internet of Things (IoT) devices. Home automation services enable the remote monitoring of indoor environments and facilities. The main advantages include saving energy consumption and improving the overall management (and users' experience) in certain application domains. The pervasive adoption and diffusion of such remote monitoring solutions is hampered by the timing required for design, prototyping and further developing applications and underlying architecture, which must be often customized on the basis of specific domains' needs and involved entities. To cope with this issue, the paper proposes the analysis and prototyping of a domotics case study, in order to demonstrate the effectiveness of proper IoT-related tools in speeding up the testing phase.*

Keywords – Domotics, Internet of Things, monitoring application, prototyping

1. INTRODUCTION

Applications for indoor and remote monitoring are nowadays adopted in different domains, ranging from smart homes to smart offices, and tailored to many scopes, such as minimizing possible local mismanagement and wastage of resources. Moreover, in order to reduce the negative influences of buildings on the environment, *green* building, which is also known as *sustainable* building, has begun to spread, aimed at at creating a better indoor environmental quality for occupants, while reducing natural resources consumption [1].

Different technologies concur to the realization of remote monitoring applications, which are strictly related to the Internet of Things (IoT) paradigm. They include Wireless Sensor Networks (WSN), Wireless Multimedia Sensor Networks (WMSN), Near Field Communication (NFC), Radio-Frequency Identification (RFID), actuators, and communication protocols such as Message Queue Telemetry Transport (MQTT), ZigBee, Constrained Application Protocol (CoAP), 6LowPAN (IPv6 over Low-Power Wireless Personal Area Networks), and so on [2]. The basic idea behind the IoT paradigm is the possibility of acquiring heterogeneous kinds of information from the environment where IoT devices are placed. Such devices embed both sensing and actuating capabilities, which make them "smart" and enable them to interact with the surrounding environment. Such features allow the IoT system to share a huge amount of information throughout the network and the Internet. Such data can be used to provide customized services to the interested users [3].

To achieve such a goal, the numerous technologies and communication protocols, just mentioned above, should often cooperate, in order to realize an efficient IoT infrastructure and to regulate the information exchange process. Tools, simulators or testing-platforms, for supporting the realization of such IoT infrastructures, from the design towards the development phase, are needed. Their scope is representing all the components acting within the envisioned environment, so as to give an overview of the whole system before real deployment, in a limited scale. Hence, this paper proposes the use of different supporting tools, targeted at the IoT, providing a representative case study related to domotics.

The remainder of this paper is organized as follows. Section 2 investigates the actual state-of-the-art tools and methods used by the researchers for validating remote monitoring systems, thus revealing our motivations. Section 3 presents the technologies and tools adopted for investigating the case study, which is detailed in Section 4. Finally, Section 5 ends the paper, drawing some hints for the direction of future research.

2. RELATED WORKS

A well-investigated field in remote controlling is that of e-health [4] [5], ranging from the monitoring of chronic diseases, to vital signs current status monitoring, and, finally, to the triage prioritization of patients.

Other solutions available concern smart buildings, which mainly include smart homes and smart offices. The work, presented in [6], uses real data-sets, collected from existing smart homes (i.e., reporting information such as energy consumption, lighting, heating, and so on), for testing a middleware conceived to evaluate the security of the information, which is transmitted within the underlying IoT infrastructure. The middleware runs on Raspberry Pi, and it is implemented in Node.js [1], while JSON formal language and the MongoDB [2] database are used for data exchange and storage, respectively.

[1]Node.JS. http://nodejs.org/
[2]MongoDB. http://www.mongodb.org/

Another test bed, consisting of a Raspberry Pi, is detailed in [7]. Also here, the final goal is to evaluate a security protocol for enforcing the usage of control policies. In both cases (i.e., [6] and [7]) the test bed consists of a limited number of devices, thus preventing the conducting of relevant considerations about the scalability of the proposed approaches. Note that such an aspect is not so relevant with respect to the approach presented in this paper.

Instead, the authors of [8] present three different use cases to demonstrate the feasibility and efficiency of their architecture, by measuring home conditions, monitoring home appliances, and controlling home access. Such a solution integrates the IoT paradigm with web services and cloud computing. The following technologies have been adopted for the test bench: Arduino platform for sensing and actuating functionalities; Zigbee for networking; cloud services; JSON data format for information exchange.

A prototype service for a smart office is provided in [9] to evaluate, from a functionalities' viewpoint, the proposed Integrated Semantics Service Platform (ISSP). The solution is based on an ontology and a model for semantic interpretations of user inputs through a proper web app. The whole architecture is based on Mobius [3], which is a *oneM2M-compatible* IoT service platform.

In [10], the work describes a practical realization of an IoT architecture, targeted to the University of Padova (Italy), which allows the interaction of WSN and actuators to standard networks, such as web services. It is an example of smart building, since the IoT network spans the floors and different areas within the Department of Information Engineering. Basic services, such as environmental monitoring and localization, regulated by roles and authorizations, are provided by means of the proposed approach.

A similar work has been carried out at the University of Bari (Italy), where existing hardware and software IoT solutions have been glued together to provide a reliable monitoring system, able to handle both scalar and media data belonging to either Internet Protocol version 4 (IPv4) and IPv6 realms [11]. In more detail, an IoT middleware, named NOS (Networked Smart object) [12], is able to manage IoT heterogeneous data, and has been integrated with: (i) TLSensing platform, which is able to efficiently acquire environmental information; and (ii) an IP camera, in charge of acquiring images from the surrounding environment. An experimental test bed has been been deployed in a university's laboratory, in order to continuously monitor environmental conditions and access control, also against malicious behaviours (e.g., to perform intrusion detection tasks).

Based on a coordinator-based ZigBee network, the smart home control system, presented in [13], has been written as a C# program in charge of simulating the users' behaviour. A similar approach is that of [14], where ZigBee nodes are simulated by means of a well-known WSN simulator, named *NS2*. The main drawback, emerged from such solutions, is that an IoT system is too complex for being simulated by a WSN simulator or by a "simple" software.

Note that, in general, the growth and diffusion of remote monitoring systems was favoured by the availability of sensor devices, able to acquire, in real time, information from the surrounding environment and transmit it throughout the network towards a sink point, which is usually in charge of collecting and processing all the gathered data from a specific application [15]. What emerges from literature is the need for a tool or a set of interoperating tools, able to represent the whole remote monitoring architecture closer, as much as possible, to the future working system, in order to provide provide designers and developers with a complete view of the final architecture and underlying logic, before its real deployment. Such a role has been played by WSN's simulators/emulators for many years [16], but, with the advent of IoT, new systems must be adopted, due to the heterogeneity of the involved devices and to the different services provided. Hence, the main contribution of the work, presented herein, can be summarized as follows:

- The adoption and integration of proper tools and technologies are proposed, in order to represent a domotics IoT scenario.
- A general overview of the envisioned system is given by means of a complete test-bed, to be validated before real deployment on a large scale.

3. TECHNOLOGIES AND TOOLS

Before detailing the case study of interest, the involved technologies and tools are introduced herein. An overview of the envisioned domotics system is provided in Fig. 1, which resembles all the components described for the use case. A demo video is available at *https://youtu.be/-5Gg5I0B3Ak*.

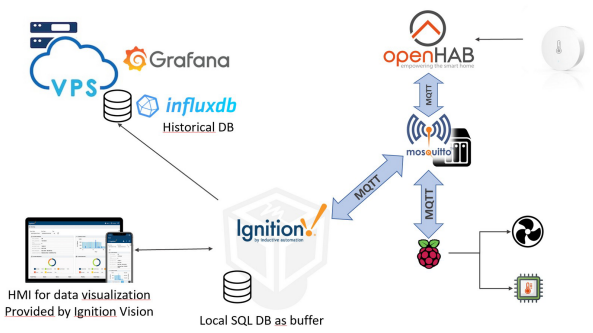


Fig. 1 – Domotics - system architecture

[3] Mobius oneM2M, "oneM2M-compatible IoT service platform". http://wiki.onem2m.org/index.php?title=Open_Source

3.1 MQTT

MQTT[4] is a publish&subscribe network broker-based messaging protocol, which is used to transport messages between devices in IoT networks or, in general, in constrained scenarios. Note that it is largely used in the IoT domain thanks to its robustness and power-saving communication. MQTT is based on two types of entities: message broker and clients. In more detail, a one-to-many message distribution is performed as well as another feature involved in the decoupling of information between sources and consumers. In general, MQTT is agnostic about the content of the payload. The broker can be implemented by using Mosquitto [5], but other solutions are also available, such as HiveMQ [6].

The concept of *topic* is fundamental in MQTT; it consists of strings used by the broker to filter messages gathered by the connected client; a topic has one or more levels, separated by a forward slash, so as to obtain a logical tree structure. Topics are used by clients for publishing messages and for subscribing to the updates from other clients, thus avoiding a continuous polling among producers and consumers. There is the possibility to subscribe to an exact topic or to multiple topics at once by using the wildcards, represented by the following symbols: (i) + for a single-level wildcard (i.e., exactly one topic level); (ii) # for a multi-level wildcard (i.e., an arbitrary number of topic levels). When a message is published under a certain topic, it is delivered to each matching subscription registered at that time.

In the case study presented in Section 4, MQTT protocol plays a central role in message passing, due to its efficiency for the investigated scenario.

3.2 InluxDBd

InluxDBd[7] is a database belonging to the NoSQL family. It has been specifically designed and developed for managing time-series data, thus making it an ideal choice for periodically logging sensor information. Data is stored into "measurements" by using timestamps, fields and tags. Fields are used to store data information, which can be strings, floats, integer or boolean and are always associated with a timestamp. Tags are similar to fields, but are also indexed; this allows the storage of important metadata in tags to optimize querying performance. InluxDBd is adopted in the case study, presented in Section 4, since its data structure perfectly fits the need of heterogeneity, which is dictated by IoT environments.

[4]MQTT v3.1/v3.1.1, https://mqtt.org/mqtt-specification/
[5]Mosquitto, open source MQTT v3.1/v3.1.1 broker. http://mosquitto.org
[6]HiveMQ MQTT broker. https://www.hivemq.com
[7]InfluxDB, time series platform. https://www.influxdata.com/

3.3 OpenHAB

Open Home Automation Bus (OpenHAB) [8] is an open-source project home application platform used to run smart homes. It naively supports many devices and allows the user to further extend its capabilities by installing modules and plugins. Moreover, OpenHAB allows the writing of custom logical rules, which can be triggered using deployed sensors and perform user-defined actions (e.g., turn on lights on a given time or when a motion sensor is activated). OpenHAB makes use of various different concepts to model the smart home environment, two of which are relevant for the case study presented in this paper:

- *Things* can be seen as entities that can be physically added to the system, like a vendor sensor gateway, or virtual, like a web service which can provide information to the system
- *Items*, instead, represent functionality used by the application. For example, the temperature values read by a sensor, or the current state of a switch, are considered as *items*; while the sensor itself is the *things* which provides such *items*.

For logical operations, OpenHAB uses *rules*, which are composed in the following way:

- Rule's *name*, which defines a unique name to reference the rule
- *When* statement, which provides the trigger to activate the rule
- *Then* statement, which defines the tasks to be performed when the rule is triggered.

3.4 Ignition

Ignition [9] is a commercial, server-based cross-platform software, which is managed through web technology. Built with customization in mind, it supports a modular structure so that its deployment can be tailored for every specific use case. Ignition includes a Human Machine Interface / Supervisory Control And Data Acquisition (HMI/SCADA), which can be built, in a customized way, depending on the intended purpose. Ignition makes use of *tags* as points of data; these can be both static or dynamic on the basis of the final scope. While it provides a set of predefined types, it also allows the user to extend them through the use of *User Data Types (UDT tags)*. The basic *tags* are the following:

- *OPC Tags* are particular kinds of tag, which use the Open Process Connectivity (OPC) standard to communicate and read/write values directly to the Programmable Logic Controller (PLC).

[8]Openhab, open source automation software. https://www.openhab.org
[9]Ignition software. https://inductiveautomation.com

- *Memory Tags* are tags which hold and store information; they can be seen as variables in a programming language.
- *Expression Tags* are tags which are driven by a user-defined expression, such as a mathematical operation, a logical operation, and so on.
- *Query Tags* are tags which pool their value from an SQL statement. They can also refer to other tags to build dynamic queries.
- *Reference Tags* are tags which simply refer to other tags to fetch their value.

Ignition, and the next Grafana tool, will be coupled with OpenHAB to realize a simple yet real domotics system with the support of real devices, as presented in Section 4.

3.5 Grafana

Grafana [10] is an open-source web-based tool for data visualization and analysis. It allows the modeling of custom dashboards, based on the required use cases and supports different data sources, like: InfluxDB, Microsoft SQL Server, PostgresSQL, AWS CloudWatch, etc. Also, it allows the development and installation of custom modules/plugins which can expand its capabilities.

4. CASE STUDY AND PROTOTYPE

In the case study presented herein, real devices are directly connected to software applications and tools, which are able to change their status. The idea is to build a system using the Ignition software SCADA solution to control real devices connected to it. The list of used devices includes: (i) a RaspberryPi Zero W[11] with an attached computer fan; (ii) three Xiaomi room temperature and humidity sensors; and (iii) two Xiaomi smart plugs. Also, different host systems are involved: (i) Virtualbox to host the SCADA system; (ii) home NAS (Network Attached Storage) to host the MQTT broker (deployed using Mosquitto); and (iii) cloud VPS to host historical database and data visualization UI.
The RaspberryPi Zero W behaves like an IoT enabled PLC. PLC devices are equipped with sensors (to gather information) and actuators (to perform actions). In this case, CPU temperature information is used to feed the sensor's data. For the actuator, instead, an external fan is used and controlled through the GPIO pins using Pulse Width Modulation (PWM). PWM is a method used to control devices that require power or electricity. It essentially makes use of a digital signal, which is periodically turned on or off to modulate the connected device. In particular, PWM is used to control the fans' motor; the larger the time frame between pulses is, the slower the motor turns. The envisioned scheme is shown in Fig. 2.

[10] Grafana, interactive visualization tool. https://grafana.com
[11] https://www.raspberrypi.org/products/raspberry-pi-zero-w/

The RaspberryPi communication is managed through the usage of MQTT, where:

- Sensors' information is published to the topic *rpi/cpu/temperature*
- Actuators' information is fetched by subscribing to the topic *rpi/fan/speed*

All the required logic is written using Python and executed at boot time by means of a *crontab task*.

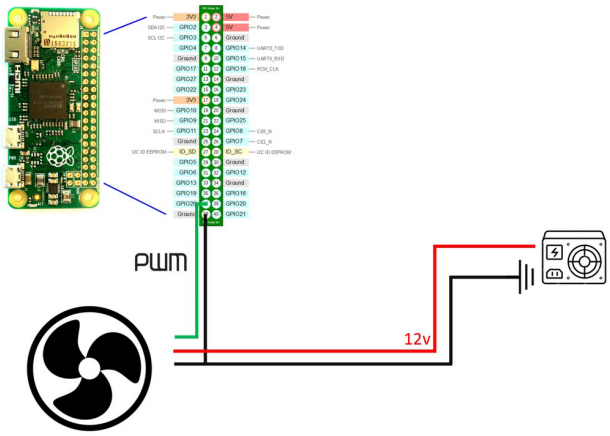


Fig. 2 – Domotics - fan's connection schema

OpenHAB, presented in Section 3 and hosted on the home NAS, is used to bridge the connection among the proprietary Xiaomi smart home sensors. The adopted modules are: (i) Xiaomi Smart Home Binding[12], which is used to communicate with the sensor gateway; and (ii) MQTT Binding[13], which is used to connect to an MQTT broker. Configuration for the Xiaomi binding is done through the OpenHAB web GUI, by providing the gateway IP and private API key, as shown in Fig. 3. By means of a web GUI, it is possible to automatically search and add all detected/connected sensors creating for each a dedicated *item*. Such an *item* is used in *rule*'s definition to identify the sensor. The MQTT binding is configured by adding an appropriate *thing* file in the OpenHAB configuration folder, thus allowing it to be referred in MQTT communication. The next step is defining the logical rules to follow for publishing sensor data over MQTT. These are defined by adding *rules*' file into the OpenHAB configuration folder. The rule triggers on every detected state change on the observed sensor by publishing its new value to the appropriate topic, as defined before.
The core of the system is the Ignition SCADA, which is the software in charge of reading and managing all attached devices, connecting via the MQTT protocol. Ignition leans on a Microsoft SQL server database, which is used as a local buffer to periodically store and change the status of sensors' information in case the connection towards the historical server is lost, but also to manage data migration

[12] https://www.openhab.org/addons/bindings/mihome/
[13] https://www.openhab.org/addons/bindings/mqtt/

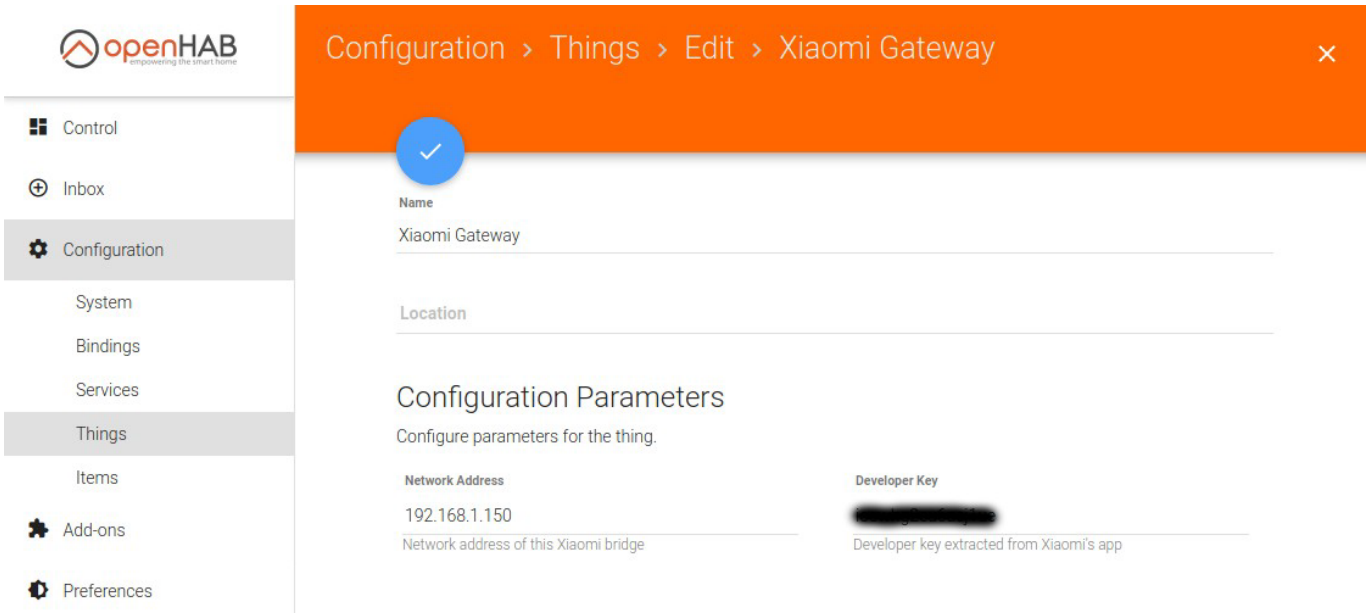


Fig. 3 – Domotics - OpenHAB Xiaomi binding configuration

towards the historical database, which is hosted on an external VPS server. The database structure is very simple, since it includes the two tables shown in Fig. 4. Instead, the topics' structure is the following:

- RaspberryPi CPU temperature *rpi/cpu/temperature*
- Current fan speed *rpi/fan/rpm*
- Room's temperature and humidity sensors:
  - *home/room_1/temperature* & *home/room_1/humidity*
  - *home/room_2/temperature* & *home/room_2/humidity*
  - *home/kitchen/temperature* & *home/kitchen/humidity*
- Current light state:
  - *home/light_1/state*
  - *home/light_2/state*

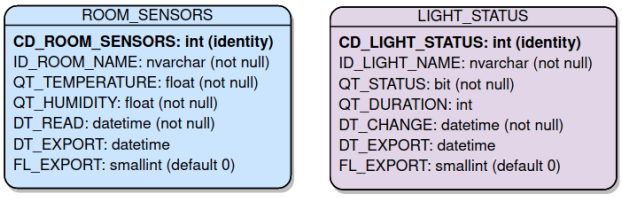


Fig. 4 – Domotics - table structure

A memory tag is added to hold the current desired fan speed, ranging from 0-100. Such a tag implements a Python script, which triggers on a value update, checks if the new value is valid and publishes it to the dedicated topic *rpi/fan/speed*. To automatically manage the fan, two support memory tags are needed: (i) a boolean, to toggle on or off the automatic fan profile; and (ii) a float, to set the desired target CPU temperature. Also, a script is applied to the CPU temperature tag, which triggers on each new read value, comparing it to the desired temperature and deciding if its necessary to turn the fan on or off. To avoid continuously switching the fan state, due to the temperature's fluctuations around the threshold value, the fan is turned off once a temperature, which is 2°C lower than the target one, is reached. A scheme of the just described behaviour is sketched in Fig. 5.

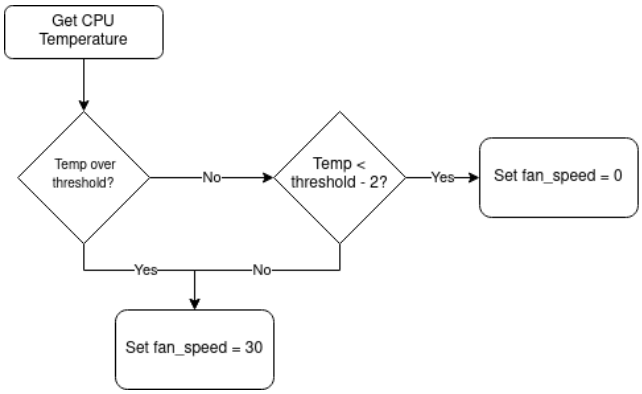


Fig. 5 – Domotics - fan's profile flowchart

Furthermore, a web GUI is needed to visualize and control the connected devices; it is realized by means of Vision, which is an UI building tool, provided by Ignition, and it is able to build a local HMI, as shown in Fig. 6. Vision is composed of four parts:

- MQTT broker status, where the right LED indicates if the broker is currently available (green) or not (red)

- Room temperature and humidity, which indicate, for each room, the last detected information, while also displaying a chart with the trend of the last hours. The chart toggles between temperature and humidity by pressing the appropriate current value
- Light status, which shows the current light state and the last date-time when it toggled
- RaspberryPi fan control, which displays a gauge with the current real-time CPU temperature. This is divided into three colour-coded areas: (i) green as desired temperature; (ii) yellow above target; and (iii) red as over temperature. The user can interact with the dial being able to change the desired temperature. Also, it is possible to override the automatic fan controller with the appropriate switch and, then, use the slider to manually set the fan speed.

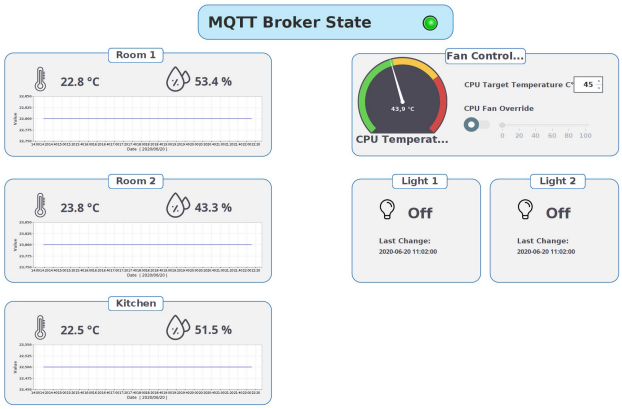


Fig. 6 – Domotics - Vision HMI

As just anticipated, the historical database is hosted on an external VPS server, and it is paired with a data visualization tool, named Grafana (see Section 3), that allows the visualization of the gathered (and stored) data in various user's defined charts. The database used for this task is InluxDBd(see Section 3) mainly for two reasons: (i) it is designed specifically to manage time-based information; and (ii) it is officially supported by the Grafana suite. For security reasons, two separate accounts have been defined on the historical database: (i) read/write, for the migration task; and (ii) read only, for the Grafana connection. As stated in Section 3, InluxDBddefines two types of data: tags and fields. For this specific use case, the $roomID$ and $lightID$ are added as tags, since they are mainly used in the $WHERE$ statement to filter data; while all the other information is stored into fields. Two measurements are used: one for rooms and one for lights. Hence, as shown in Fig. 7, two types of panels are visible: (i) a chart to plot the temperature/humidity trend; and (ii) a panel to visualize how long the light states lasted.

5. CONCLUSION

Early design and prototyping are fundamental to speed up the development process of monitoring systems. To achieve such a goal, proper tools and technologies must be adopted. In this respect, a domotics case study has been pointed out in this paper, where different technologies, protocols and languages are grouped together without worrying about interoperability issues, thanks to the capabilities of the adopted tools to interact between themselves. In that sense, the presented approach represents a viable solution for performing preliminary tests on a domotics IoT-based scenario. Note that other technologies could be adopted for the same purpose, such as MongoDB as data store, instead of InfluxDB, and CoAP (Constrained Application Protocol) as transmission protocol, instead of MQTT. Both such solutions are targeted to IoT and constrained scenarios; however MongoDB is a document-oriented (and not time based, as InfluxDB) database, while CoAP does not follow a publish and subscribe philosophy, as MQTT. For such reasons, in this work InluxDBdand MQTT have been preferred, since they better fit the requirements of a domotics context, where handling data following a topics' hierarchy, instead of unstructured information, represents the most viable solution.

Two important aspects still deserve attention, as an open research activity: scalability and securit and privacy. Such a kind of analysis could be carried out by defining re-usable modules and components to be integrated (and replicated) in a more complex system. Instead, security&privacy requirements can be achieved at various levels: ranging from securing the MQTT communication exchange [17] [18], to protecting the data when they are stored into the database [19] [20], or to providing security and privacy policy enforcement mechanisms at the IoT core platform's level [21].

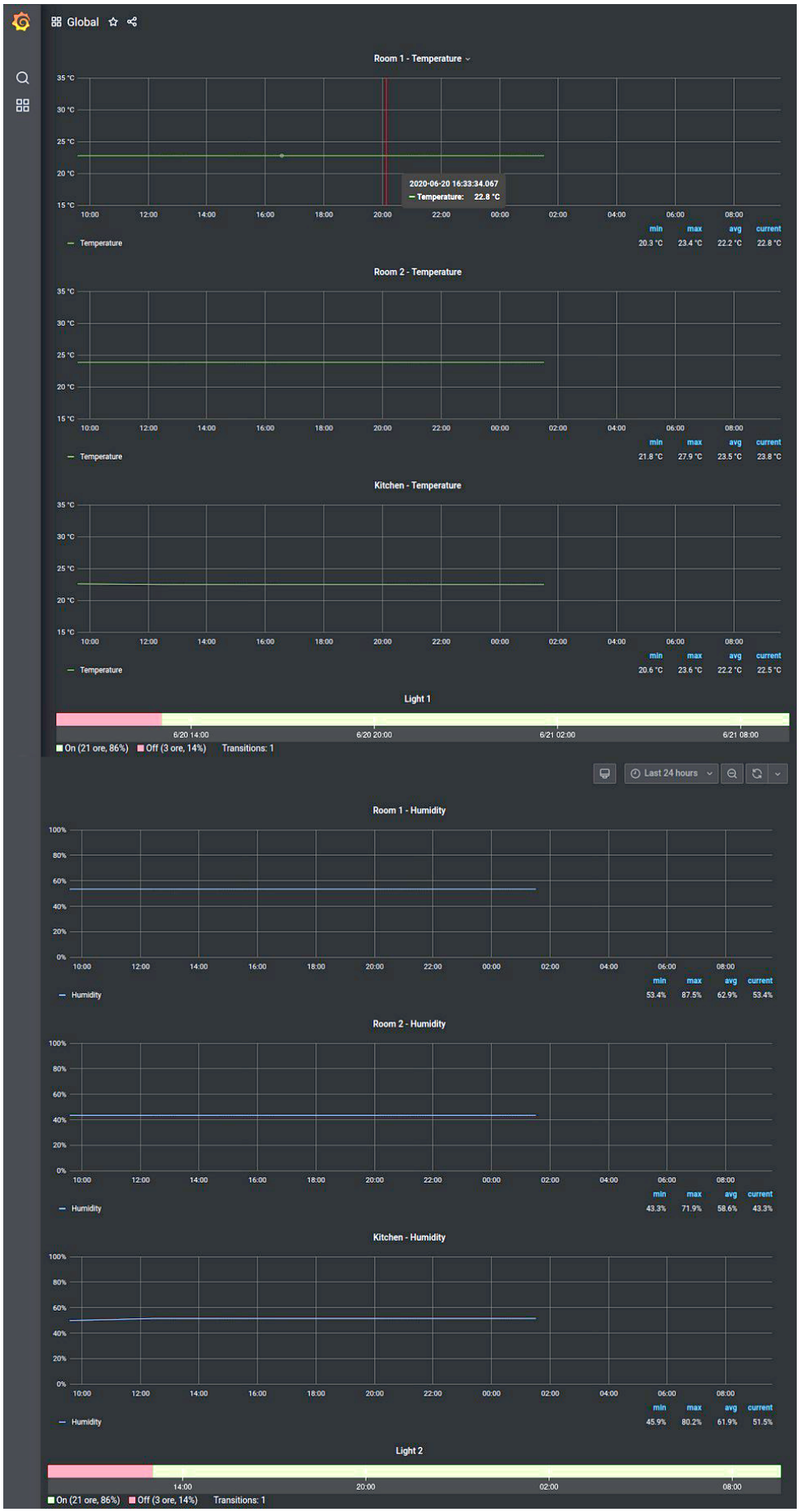


Fig. 7 – Domotics - Grafana user's defined dashboard

REFERENCES

[1] Yang Geng, Wenjie Ji, Zhe Wang, Borong Lin, and Yingxin Zhu. "A review of operating performance in green buildings: Energy use, indoor environmental quality and occupant satisfaction". In: *Energy and Buildings* 183 (2019), pp. 500–514.

[2] Shadi Al-Sarawi, Mohammed Anbar, Kamal Alieyan, and Mahmood Alzubaidi. "Internet of Things (IoT) communication protocols". In: *2017 8th International conference on information technology (ICIT)*. IEEE. 2017, pp. 685–690.

[3] Maria Stoyanova, Yannis Nikoloudakis, Spyridon Panagiotakis, Evangelos Pallis, and Evangelos K Markakis. "A survey on the internet of things (IoT) forensics: challenges, approaches, and open issues". In: *IEEE Communications Surveys & Tutorials* 22.2 (2020), pp. 1191–1221.

[4] OS Albahri, AS Albahri, KI Mohammed, AA Zaidan, BB Zaidan, M Hashim, and Omar H Salman. "Systematic review of real-time remote health monitoring system in triage and priority-based sensor technology: Taxonomy, open challenges, motivation and recommendations". In: *Journal of medical systems* 42.5 (2018), p. 80.

[5] Rachael C Walker, Allison Tong, Kirsten Howard, and Suetonia C Palmer. "Patient expectations and experiences of remote monitoring for chronic diseases: systematic review and thematic synthesis of qualitative studies". In: *International journal of medical informatics* 124 (2019), pp. 78–85.

[6] Sabrina Sicari, Alessandra Rizzardi, Daniele Miorandi, and Alberto Coen-Porisini. "Securing the smart home: A real case study". In: *Internet Technology Letters* 1.3 (2018), e22.

[7] Antonio La Marra, Fabio Martinelli, Paolo Mori, and Andrea Saracino. "Implementing usage control in internet of things: a smart home use case". In: *Trustcom/BigDataSE/ICESS, 2017 IEEE*. IEEE. 2017, pp. 1056–1063.

[8] Moataz Soliman, Tobi Abiodun, Tarek Hamouda, Jiehan Zhou, and Chung-Horng Lung. "Smart home: Integrating internet of things with web services and cloud computing". In: *2013 IEEE 5th International Conference on Cloud Computing Technology and Science (CloudCom)*. IEEE. 2013, pp. 317–320.

[9] Minwoo Ryu, Jaeho Kim, and Jaeseok Yun. "Integrated semantics service platform for the Internet of Things: A case study of a smart office". In: *Sensors* 15.1 (2015), pp. 2137–2160.

[10] Angelo P Castellani, Nicola Bui, Paolo Casari, Michele Rossi, Zach Shelby, and Michele Zorzi. "Architecture and protocols for the internet of things: A case study". In: *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on.* IEEE. 2010, pp. 678–683.

[11] D. Costantino, G. Malagnini, F. Carrera, A. Rizzardi, P. Boccadoro, S. Sicari, and L. A. Grieco. "Solving Interoperability within the Smart Building: A Real Test-Bed". In: *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*. May 2018, pp. 1–6.

[12] S.Sicari, A. Rizzardi, D. Miorandi, C. Cappiello, and A. Coen-Porisini. "A secure and quality-aware prototypical architecture for the Internet of Things". In: *Information Systems* 58 (2016), pp. 43–55.

[13] Murad Khan, Bhagya Nathali Silva, and Kijun Han. "Internet of things based energy aware smart home control system". In: *IEEE Access* 4 (2016), pp. 7556–7566.

[14] P Xiang. "Design of smart home system based on the technology of internet of things". In: *Research Journal of Applied Sciences, Engineering and Technology* 4.14 (2012), pp. 2236–2240.

[15] SR Jino Ramson and D Jackuline Moni. "Applications of wireless sensor networks - A survey". In: *2017 international conference on innovations in electrical, electronics, instrumentation and media technology (ICEEIMT)*. IEEE. 2017, pp. 325–329.

[16] Ivan Minakov, Roberto Passerone, Alessandra Rizzardi, and Sabrina Sicari. "A comparative study of recent wireless sensor network simulators". In: *ACM Transactions on Sensor Networks (TOSN)* 12.3 (2016), p. 20.

[17] A. Rizzardi, S. Sicari, D. Miorandi, and A. Coen-Porisini. "AUPS: An Open Source AUthenticated Publish/Subscribe system for the Internet of Things". In: *Information Systems* 62 (2016), pp. 29–41.

[18] Chang-Seop Park and Hye-Min Nam. "Security Architecture and Protocols for Secure MQTT-SN". In: *IEEE Access* 8 (2020), pp. 226422–226436.

[19] Guo Yubin, Zhang Liankuan, Lin Fengren, and Li Ximing. "A solution for privacy-preserving data manipulation and query on NoSQL database". In: *Journal of Computers* 8.6 (2013), pp. 1427–1432.

[20] Kanika Goel and Arthur HM Ter Hofstede. "Privacy-Breaching Patterns in NoSQL Databases". In: *IEEE Access* 9 (2021), pp. 35229–35239.

[21] S. Sicari, A. Rizzardi, D. Miorandi, C.Cappiello, and A. Coen-Porisini. "Security Policy Enforcement for Networked Smart Objects". In: *Computer Networks* 108 (2016), pp. 133–147.

AUTHORS

Sabrina Sicari is Associate Professor at University of Insubria (Varese). She received a degree in Electronical Engineering, 110/110 cum laude, from University of Catania, in 2002, where in 2006 she got a Ph.D. in Computer and Telecommunications Engineering. She is a member of COMNET, IEEE IoT, ETT, ITL editorial board. Her research activity focuses on security, privacy and trust in WSN, WMSN, IoT, and distributed systems.

Alessandra Rizzardi received a BS/MS degree in Computer Science 110/110 cum laude at University of Insubria (Varese), in 2011/2013. In 2016 she got a Ph.D. in Computer Science and Computational Mathematics at the same university, under the guidance of Prof. Sabrina Sicari. She is Assistant Professor in Software Engineering at the University of Insubria. Her research activity is on WSN and IoT security issues.

Alberto Coen-Porisini received his Dr. Eng. degree and Ph.D. in Computer Engineering from Politecnico di Milano in 1987 and 1992, respectively. He has been Professor of Software Engineering at Università degli Studi dell'Insubria since 2001, Dean of the School of Science from 2006 and Dean since 2012. His research regards specification/design of real-time systems, privacy models and WSN.

RF-BASED LOW-SNR CLASSIFICATION OF UAVS USING CONVOLUTIONAL NEURAL NETWORKS

Ender Ozturk[1], Fatih Erden[1], Ismail Guvenc[1]
[1]Electrical and Computer Engineering, NC State University, Raleigh, NC 27606, United States

NOTE: Ender Ozturk, eozturk2@ncsu.edu

Abstract *– Unmanned Aerial Vehicles (UAVs), or drones, which can be considered as a coverage extender for Internet of Everything (IoE), have drawn high attention recently. The proliferation of drones will raise privacy and security concerns in public. This paper investigates the problem of classification of drones from Radio Frequency (RF) fingerprints at the low Signal-to-Noise Ratio (SNR) regime. We use Convolutional Neural Networks (CNNs) trained with both RF time-series images and the spectrograms of 15 different off-the-shelf drone controller RF signals. When using time-series signal images, the CNN extracts features from the signal transient and envelope. As the SNR decreases, this approach fails dramatically because the information in the transient is lost in the noise, and the envelope is distorted heavily. In contrast to time-series representation of the RF signals, with spectrograms, it is possible to focus only on the desired frequency interval, i.e., 2.4 GHz ISM band, and filter out any other signal component outside of this band. These advantages provide a notable performance improvement over the time-series signals-based methods. To further increase the classification accuracy of the spectrogram-based CNN, we denoise the spectrogram images by truncating them to a limited spectral density interval. Creating a single model using spectrogram images of noisy signals and tuning the CNN model parameters, we achieve a classification accuracy varying from 92% to 100% for an SNR range from –10 dB to 30 dB, which significantly outperforms the existing approaches to our best knowledge.*

Keywords – Convolutional neural networks (CNN), low SNR regime, RF fingerprinting, spectrogram, UAV classification

1. INTRODUCTION

Unmanned aerial vehicles (UAVs) or drones have recently gained a great deal of interest among researchers due to unrivaled commercial opportunities in various fields, such as wireless communications, logistics, delivery, search and rescue, smart agriculture, surveillance, among others [1]. In addition, the recent COVID-19 outbreak revealed the importance of remote operations in every aspect of life, which may accelerate social acceptance of drone use cases such as delivery of goods and medication [2, 3, 4]. With the new advances in airspace regulations and drone-related technologies, it is expected that there will be more and more UAVs in the skies for various use cases, sharing the airspace with other aerial vehicles [5]. The increase in daily drone usages can be considered in the context of The Internet of Everything (IoE), a broader term than Internet of Things (IoT), aiming to include the entire realm of information sources and destinations in one paradigm.

Innate advantages of UAVs that make them popular, such as ease of operation and low cost, could also be considered as major disadvantages from security and privacy perspectives. This motivates detection, classification, and tracking of different types of UAVs, and interdicts unauthorized or malicious UAVs to maintain privacy and security. Classification of UAVs can also be critical for forensics use cases, e.g. for identifying a UAV after a malicious activity (e.g. eavesdropping, espionage) based on the captured signals of the UAV. There have been many criminal activities recently with drones involved, and their small sizes make it difficult to detect, classify, and interdict them [6, 7]. Latest surveys also demonstrate that 75% of the subjects exhibit privacy and security concerns about all unmanned aerial use cases [8]. In this regard, Federal Aviation Agency (FAA) of the United States recently announced a Proposed Rule that elaborates the future action that would require remote identification of unmanned aircraft systems to address safety and security concerns [9]. UAVs can be identified through a set of features that uniquely represent them. These features can be extracted from various data sources, such as visual data, acoustic, RF, or radar signals. Each of these source types has its own pros and cons which we will review in the next section. Our contributions with this work are summarized below.

- In this study, we develop a Convolutional Neural Network (CNN)-based classifier using both time-series signal images and spectrogram images of 15 different drone controller RF signals to classify drones of different makes and models. These signals are transmitted by proprietary circuit designs and contain distinct fingerprints of commercially available drones; they can be exploited by a machine learning model to classify the make and model of the drone. We use controller signals as the data set [27] was already in possession; however, the proposed approach can also be directly applied to the signals transmitted from drones to their controllers. A flowchart of the overall procedure is given in Fig. 1.

Table 1 – Related work on detection and classification of drones using ML techniques.

Literature	Source type	Features	Data process method	Classification	# of UAVs	Accuracy	Noise consideration
[10]	Drone RF signals	Slope, kurtosis skewness	Several ML algorithms	X	N/A	96.36%	X
[11]	Drone RF signal	CSI data	Channel state information	X	N/A	86.6%	X
[12]	Acoustic waves	MFCC and LPCC	SVM	X	N/A	96.7%	X
[13]	Acoustic waves	STFT features	CNN	X	N/A	99.87%	X
[14]	Camera images	RGB arrays	CNN for moving body detection and kNN for detection	X	N/A	93%	X
[15]	Camera images	RGB arrays	CNN on ZF and VGG16 and Fast R-CNN	X	N/A	0.66 mAP	X
[16]	Radar signals	Spectrogram	2-D complex-log-Fourier transform	X	N/A	3.27% EER	X
[17]	Radar signals	Range Doppler Matrix	SVM	X	N/A	98%	X
[18]	Radar signals	Micro-Doppler signature	PCA feature extraction on spectrograms	✓	3	94.7%	X
[19]	Radar signals	Micro-Doppler spectrogram	CNN and LSTM-RNN	✓	5	97.7%	X
[20]	Radar signals	Micro-Doppler signature	CNN	✓	6	94.7%	X
[21]	Radar signals	Micro-Doppler signatures through EMD	SVM	✓	11	>95%	X
[22]	Radar signals	Micro-Doppler signatures	SVM	✓	11	95.4%	X
[23]	Radar signals	Range Doppler spectrum	CNN	X	N/A	99.5% and 54.2% for 0 dB	✓
[24]	Drone RF signals	Statistical features e.g., mean, median, RMS	Logistic regression	✓	8	88-94% in 0.35 s	X
[25]	Radar signals	Micro-Doppler signature	ANN on MLP	✓	4	Various	✓
[26]	Controller RF signals	Shape factor, kurtosis, variance	Several ML algorithms	✓	17	98.13% and 40% for 0 dB SNR	✓
This work	**Controller RF signals**	**Time-series signal and spectrogram RGB arrays**	**CNN**	✓	**15**	**99.7% and 99.5% for 0 dB SNR**	✓

- For the classification tasks that involve RF fingerprinting, variations in the Signal-to-Noise Ratio (SNR) of the received RF signals is a challenging problem. In this work, we also address this practical problem by considering a range of SNR levels from −10 dB to 30 dB while training the CNN models. Noisy training data is generated by adding artificial white noise to the original data. When using spectrogram images to train the CNN models, we only focus on the frequency range of interest, which improves classification accuracy significantly in comparison with time-series images.
- In this work, we apply denoising on the spectrogram images to further improve the performance at low SNRs. We tune the spectral density level that will appear on the spectrogram image and filter out spectral densities lower than the tuned level. Our proposed classifier highly outperforms previously published work, especially at low SNRs.

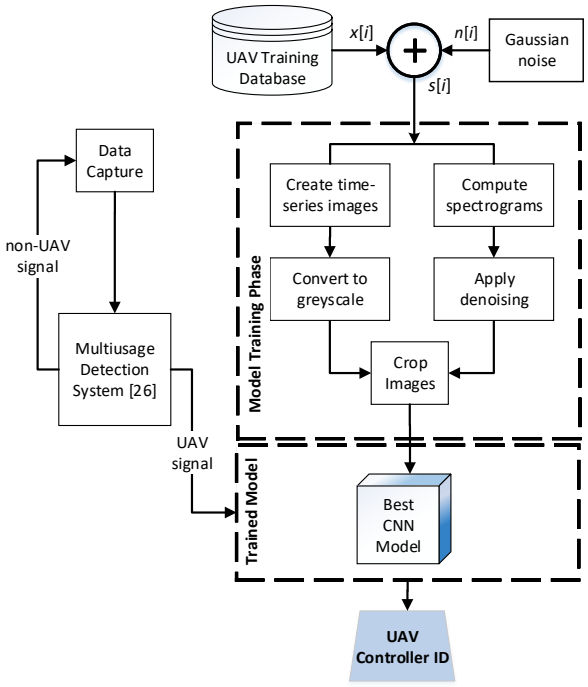


Fig. 1 – Overview of the proposed system. Multistage detector classifies the captured data as of type *UAV* or *non-UAV*. In the case of a *UAV* signal, captured data is artificially noised, and time-series and spectrogram images are created afterwards for training the corresponding CNN models. Time-series images are converted to grayscale to increase computational efficiency. Spectrograms are denoised to increase the model accuracy. Separate CNN models are trained and predictions are made using these CNN models. The best CNN model is deployed at the end.

A possible use case of the proposed system would be about the upcoming FAA regulation on Remote ID [9]. Remote ID is defined as the ability of a UAV to provide the relevant identity information to other parties. Even the drones will be obliged to reveal their IDs to comply with this regulation, it will still be possible for the malicious drones to fake their IDs. The system proposed in this work can be a part of a framework that verifies the drone IDs and make sure that the flying drone has the same ID as in the FAA's logs. This way countermeasures can be taken in the presence of a threat.

With regard to the type of images used, even though CNN models trained on spectrogram images perform better than models trained on time-series images for every scenario, we kept the results for the latter to provide a better basis for comparison of our contribution. This is because our present work is an extension of the work in [26], where statistical features extracted from time-series data have been used previously.

The rest of the paper is organized as follows. In Section 2, a comprehensive literature review including the information of noise consideration is given. In Section 3, the data set and the procedure for obtaining noisy samples are introduced. Section 4 discusses an image data preprocessing step and the CNN-based classifier used in this work. Experimental results and relevant discussions are presented in Section 5. Finally, the paper is concluded in Section 6.

2. LITERATURE REVIEW AND CONTRIBUTIONS

Various approaches have been proposed in the literature for the detection and classification of drones. In Table 1, we summarize the related literature on drone detection and classification with some representative work and emphasis on the number of UAVs considered, classification accuracy, and noise considerations. Here we use the term *detection* as a special case of classification that has only two classes (i.e., UAV/non-UAV). Techniques used to achieve these tasks can be categorized based on the type of data being captured (e.g., radar signals, drone or controller Radio Frequency (RF) signals, acoustic data, or camera images), features extracted from the data (e.g., RF fingerprints, spectrogram images), and the Machine Learning (ML) algorithms deployed for classification. Acoustic sensors do not require line-of-sight (LOS); however, they suffer from short range, as drones could operate very quietly [12, 28], and data gathered using microphone systems are prone to wind and environmental clutter. On the other hand, a LOS vision under daylight is essential for techniques that utilize camera images [14, 29]. Using thermal or laser-based cameras to overcome this issue increases the cost significantly.

Radar signals are immune to environmental factors, such as acoustic noise and fog. However, drones are small devices with tiny propellers which make it hard to perceive and distinguish them from each other by most radars. A high-frequency wideband radar could be used to deal with these difficulties [20, 30, 31, 18]. Such radars are considerably expensive and suffer from high path loss. RF signals of either drones themselves or controllers are mostly at sub-6 GHz band and share unlicensed Wi-Fi bands. As a result of this, equipment to capture RF signals are affordable, but on the downside, RF-based techniques require special attention for handling interference from other co-channel signal sources. Besides, no LOS is required, and these techniques are immune to many problems that acoustic and visual techniques suffer from.

RF signals can be used for classification of the UAVs, either directly or indirectly after some processing. In [26, 10, 24], time-domain statistical properties of the RF signal, such as slope, kurtosis, skewness, shape factor and variance, are used as features along with different ML algorithms to detect and classify drones. However, since unlicensed bands are heavily employed, time-domain information suffers from low SNR. Frequency-domain representation of RF signals can also be used to distinguish

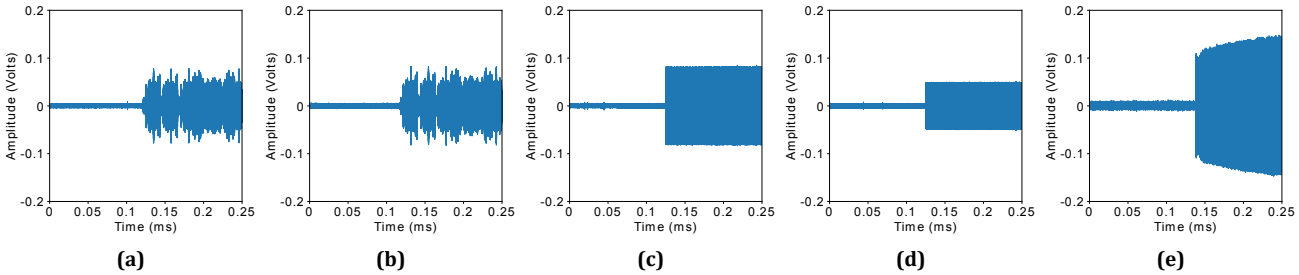


Fig. 2 – Sample controller time-series RF signals: (a) DJI Matrice 100, (b) DJI Matrice 600, (c) Spektrum DX5e, (d) FlySky FS-T6, and (e) Spektrum JR X9303. RF signals from different controllers may look alike, making it difficult to identify the drones based on only the envelopes of the captured signals.

between different types of drones. Transforming RF signals into the frequency domain filters out the out-of-band noise and helps improve classification accuracy up to a certain extent.

In the literature, there are studies using radar signals and spectrograms to detect and classify drones [16, 18, 19, 20, 21, 22, 23]. However, there is no study that utilize spectrograms of RF signals in the context of UAV detection/classification to the best of our knowledge.

Even though the mass majority of classification efforts in this field aim to identify drone make and model to support a decision of friend/foe, there are some other work that use ML techniques to identify drone pilots. For example, in [32], drone controller RF signals are recorded to characterize pilot activity, and different types of maneuvers that a pilot could do are used as features.

Classification accuracy should be considered together with the number of UAVs as it gets harder to classify UAVs with high accuracy as the number of classes increases. For studies which have X marks in the *Classification* column, the proposed models performed only *detection* which means there are only two classes. We also provide the information about whether the work considers noise or not, to better emphasize our contribution.

3. DATA SET AND NOISING PROCEDURE

In this work, the data set in [26] is used. This data set consists of RF signals from 15 different off-the-shelf UAV controllers listed in Table 2. RF signals were captured using an Ultra-Wideband (UWB) antenna and an oscilloscope with a sampling rate of 20 Gsa/s. Total number of samples in each signal is 5×10^6, which corresponds to a time duration of 250 µs. Time-series and spectrogram images are created from the training RF signals, and CNN models are generated for each image database.

3.1 Image creation process

The time-series RF signal of a controller is kept in a 1-D array. Time-series images are simply acquired by plotting these 1-D arrays. RF signals captured from different UAV controllers are illustrated in Fig. 2. As it can be observed from the figure, RF signals exhibit different waveforms. Digital image processing literature bestows useful techniques to distinguish such signals using an envelope detector and template matching-based approaches [33]. However, some controller signals may exhibit similar envelopes (e.g., RF signals in Fig. 2(a) and Fig. 2(b), or the signals in Fig. 2(c) and Fig. 2(d)), making it challenging to identify the controllers with these approaches. Besides, taking into account that signal envelopes get distorted at high noise levels, more advanced approaches are needed to achieve high classification accuracy.

Table 2 – UAV controllers used in this work.

UAV ID (#)	Brand & Model
1	Jeti Duplex DC-16
2	DJI Matrice 100
3	DJI Matrice 600
4	DJI Phantom 3
5	DJI Inspire 1 Pro
6	Spektrum DX5e
7	Spektrum DX6e
8	FlySky FS-T6
9	Futuba T8FG
10	Graupner MC-32
11	Hobby King HK-T6A
12	Spektrum JR X9303
13	DJI Phantom 4 Pro
14	Spektrum DX6i
15	Turnigy 9X

Spectrogram images are created calculating power spectral densities of the signals using Welch's average periodogram method, which is also called Weighted Overlapped Segment Averaging (WOSA) method [34]. In this method, time-domain signal $x[i]$ captured from a UAV is divided into successive blocks and averaged to esti- mate the power spectral density after forming the periodograms for each block, i.e.,

$$x_m[i] = w[i]x[i + mR] \,, \qquad (1)$$

where $i = 0, 1, ..., M - 1$ is the sample index, M is the window size, $m = 0, 1, ..., K - 1$ denotes the window index, K is the total number of blocks, R is the window's

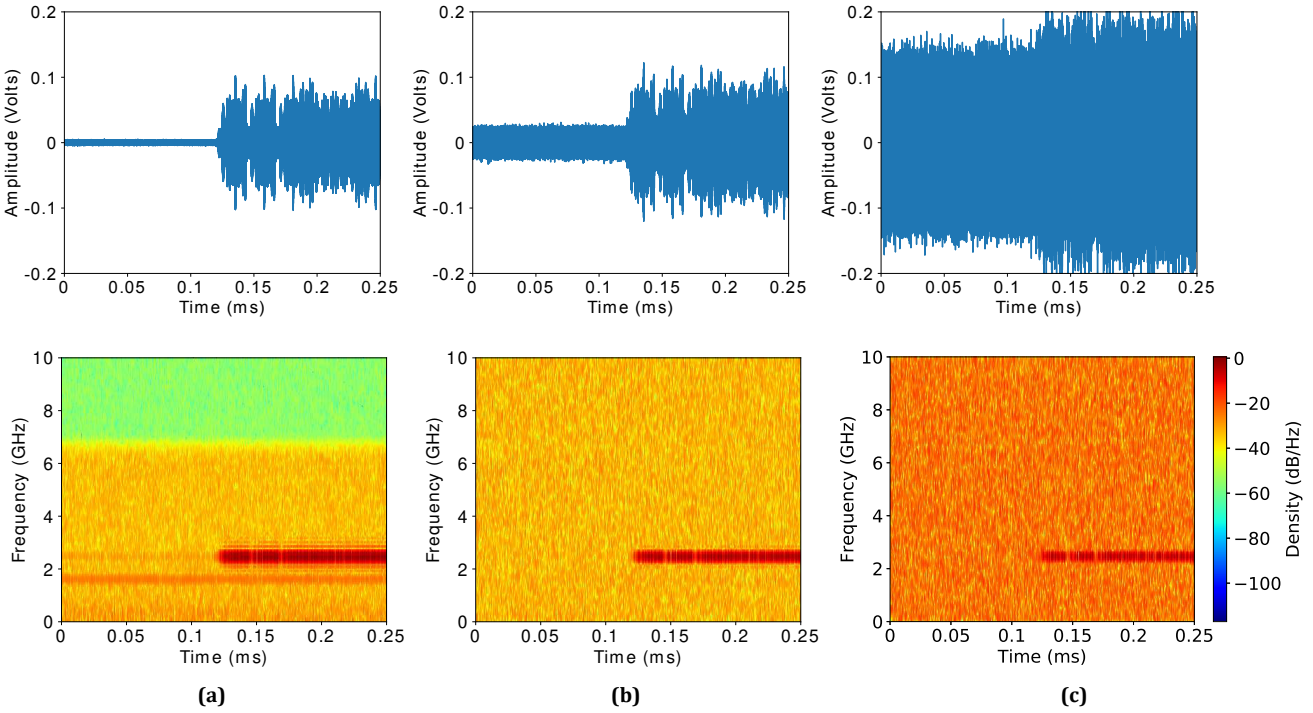


Fig. 3 – Noisy controller signal of a DJI Inspire 1 Pro for different SNRs: a) 30 dB, b) 15 dB, and c) 0 dB. Base noise in lab environment is around 30 dB. As SNR decreases, distortion occurs for both image types. All spectrogram images have the same density color scale.

hop size that tunes the amount of overlap between consecutive windows, and $w[i]$ is the window function. Then the periodogram of a block is calculated as

$$P_{x_m,M}(w_k) = \frac{1}{M}|FFT_{N,k}(x_m)|^2 = \frac{1}{M}\left|\sum_{i=0}^{N-1} x_m[i]e^{-2\mathrm{j}\pi ik/N}\right|. \quad (2)$$

Consequently, Welch estimate of power spectral density is calculated as follows

$$\hat{S}_x^W = \frac{1}{K}\sum_{m=0}^{K-1} P_{x_m,M}(w_k). \quad (3)$$

In this paper the Hanning window is used while calculating preiodograms. Then the calculated densities are mapped to a color scale to create spectrograms. We use a color map that spans the whole color space evenly, i.e., passes through all the colors in the visible range, which increases the accuracy of the proposed model significantly.

3.2 Noising procedure

Assuming fixed environmental noise, the SNR level of an RF signal decreases as the source gets farther away from the receiving antenna. Since the drone or the controller position and hence their distance to the receiver antenna may vary in different scenarios, systems that can work under low SNR regimes are required. In this study, we propose a method that can identify drones even at very low SNRs. Since the data [26] is collected in a lab environment, the noise is stable and the same for all measurements. To train and test our models for noisy signals, we add white Gaussian noise to the raw data, and then generate the corresponding time-series and spectrogram images. While generating the noisy signals, we first use Higuchi's fractal dimension method [35] to find the approximate position of the transient signal segment. We use this information to distinguish between the noise and the RF signal and calculate their actual power separately as

$$P_{\text{noise}}\ [\text{dB}] = 10 \times \log_{10}\left(\frac{\sum_{i=0}^{T_b}|x[i]|^2}{T_b}\right), \quad (4)$$

and

$$P_{\text{signal}}\ [\text{dB}] = 10 \times \log_{10}\left(\frac{\sum_{i=T_e}^{N}|x[i]|^2}{N-T_e}\right), \quad (5)$$

where T_b and T_e are the indexes where the transient begins and ends, respectively.

Next, we calculate the SNR (Γ) of the received signal and the difference between the current SNR level and the desired SNR level as follows:

$$\Gamma_{\text{signal}}\ [\text{dB}] = P_{\text{signal}} - P_{\text{noise}}, \quad (6)$$

and

$$\Delta\Gamma\ [\text{dB}] = \Gamma_{\text{signal}} - \Gamma_{\text{desired}}. \quad (7)$$

Finally, an appropriate amount of random noise $n[i]$ is added to the whole signal to set the signal to the desired

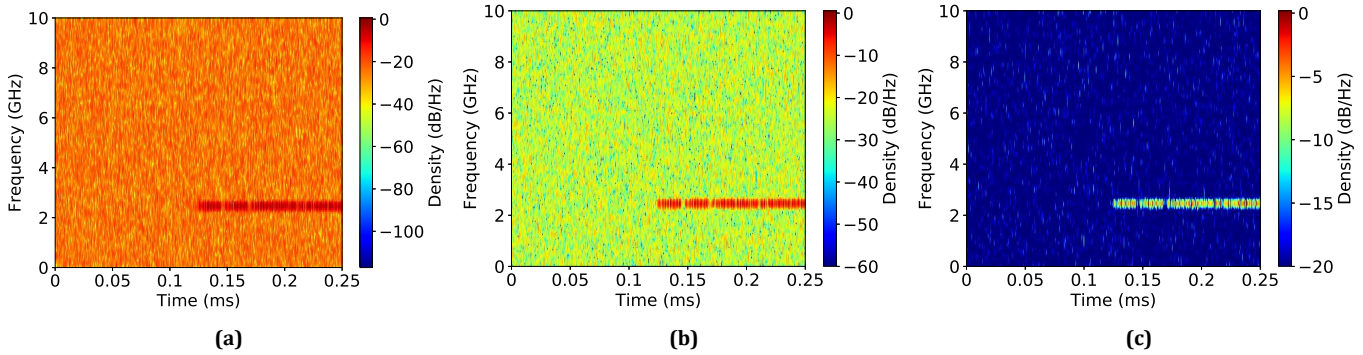


Fig. 4 – Denoising of a 0 dB SNR signal at different cut-off values: a) no truncation, b) -60 dB/Hz, and c) -20 dB/Hz. Truncation process sets the lower limit of the density color scales. This increases the level of representation of the high density signal components on the spectrogram image, consequently model accuracy is improved.

SNR level as

$$s[i] = x[i] + n[i], \tag{8}$$

where $n[i] = \Delta\Gamma \times \mathcal{N}(0,1), i = 0, 1, ..., N-1$, and N is the total number of samples in $x[i]$. Note that $\Delta\Gamma$ that is used to generate the noise sequence is not in dB scale.
A set of artificially noised time-series images and the corresponding spectrograms are given in Fig. 3. Subject to the type of controller, the original data has an SNR of about 30 dB. Increased noise causes distortion visible in both image types. However, time-series images are affected more. Spectrograms preserve signal characteristics better than time-series images as signal components can be better resolved in the frequency domain.

4. IMAGE PREPROCESSING AND CNN-BASED UAV CLASSIFICATION

CNN is a deep learning algorithm which has been proven to perform well in image recognition and classification tasks [36]. CNNs have layers just as any other neural networks; however, different from other deep learning algorithms, convolution layers are used to apply various filters to an image to extract features no matter at which part of the image they reside. This nature of the algorithm makes CNN a perfect fit for 2D data (e.g., images), and also reduces the number of required weights in a neuron, thus yields lower computational complexity in comparison with conventional deep neural network architectures. In this work, spectrograms and time-series images of RF signals have been used as inputs to the CNN models.

Even though CNN is a very powerful approach for extracting features from images, the preprocessing phase of the source data is crucial to increase the overall success of the classification and decrease the computational cost.

4.1 Conversion to grayscale and image cropping

As reviewed in Section 3.1, spectrograms reflect the power spectral densities of the signals. Since color depth preserves distinctive information, these images should be kept in red/green/blue (RGB) format. However, time-series images are not represented in such a format, and therefore, to decrease the complexity, time-series signal images should be converted to grayscale if these images do not come in grayscale by default.

Time-series and spectrogram images typically have axes, ticks and labels regardless of the software tool that is used to create them. We remove all those parts before beginning post-processing the images. Besides, captured images would include both the noise-only signal (when there is no transmission) and the transmitted signal (see the time-series signals in Fig. 3). By using Higuchi's fractal dimension method as suggested in Section 3.2, it is possible to remove out the noise-only part in both image types. In addition, one of the axes of the spectrograms will includes frequency domain information. In case the frequency range of interest is known, it is appropriate to crop the spectrograms further to lower the computational cost focusing on the desired frequency band only.

4.2 Denoising the spectrograms

Denoising is an important step towards improving the accuracy of the spectrogram image-based classification. Denoising by truncation is only applied to spectrogram images. Power spectral densities should be calculated up to a certain frequency that is defined by the sampling rate for several instants in the time domain that covers the whole signal. These spectral density values are mapped to the RGB color scale while creating spectrograms: the minimum and the maximum spectral densities are mapped to the coolest and warmest colors of a chosen color map, whereas the colors for the intermediate values are adjusted accordingly. In order to denoise the spectrogram, a cut-off density is picked as a threshold, and the spectrogram is truncated by setting the elements of the spectral density array that are smaller than this cut-off to the cut-off value itself. This process assures signal components with smaller densities to be cleared. Since most of the noise components have lower power densities than the drone controller signal itself for a wide range of SNRs,

the truncation process is essentially a denoising procedure. The rest of the signal is mapped to the same color range set, which increases the level of representation of the details. As a result, non-noise (i.e., RF) signal components come forward that help the CNN models learn better. The procedure described above could be explained mathematically as follows:

$$S'[f,i] \xrightarrow{f_t} \begin{cases} \gamma_c, & \text{if } S[f,i] \leq \gamma_c \\ S[f,i], & \text{else} \end{cases}, \qquad (9)$$
$$S'[f,i] \xrightarrow{f_c} r_{f,i}, g_{f,i}, b_{f,i},$$

where f_t is the truncation function, $S'[f,i]$ is the truncated signal subject to the cut-off value γ_c, f_c is the color mapping function, and $r_{f,i}$, $g_{f,i}$, and $b_{f,i}$ are the color intensities in the corresponding channels.

There exists a critical trade-off that depends on the chosen cut-off threshold. For a given SNR level of the signal in hand, spectrograms should be truncated at an optimum level for that SNR. More specifically, in the case of under-denoising, excess noise causes overfitting, while in the case of over-denoising, useful information is wiped out together with the noise, which yields to underfitting. To illustrate this trade-off, we will consider Fig. 4 which shows the spectrograms of a DJI Inspire 1 Pro controller signal, that is artificially noised to 0 dB SNR, at different truncation levels. In this figure, it is observed that as the threshold increases (i.e., from no truncation to -20 dB/Hz), the lower limit of the density on the spectrograms changes. This lowest density is the lowest value in the domain set. As a result of truncation, high density components of the signals are represented better on the images.

Another aspect of creating CNN models for different data sets at various SNR levels is the necessity of defining the SNR of a signal beforehand to invoke the appropriate model. To manage this, we propose to follow two approaches while working with the spectrograms. In the first approach, we create and optimize different CNN models for different SNR levels. The idea is that, assuming that the captured signal's SNR can be measured, the model having the closest SNR is called to perform classification. Even though, calculating the SNR of a received signal in real time is a tricky task, we believe, with the help of featured state-of-the-art measurement devices and newly developed algorithms [37], this would not be a problem. In the second approach, we define an optimum cut-off value (i.e., minimum average validation loss among all different cut-offs), merge all the images of different SNR levels truncated at this level to create a new comprehensive data set, and then train a single model. The major advantage of this second approach is that it is no longer required to determine the SNR of the signal in advance.

4.3 Training and testing CNN models

In this work, CNNs are trained using Keras with Tensorflow at the backend. In the models created, we have three convolution layers (Conv2D) followed by pooling layers (MaxPool2D) and then a fully connected layer followed by the output layer. Convolution layers get deeper (i.e., the number of filters increase), and size of the images get smaller as the data travels deep into the model, in accordance with the general convention. The CNN models have been trained and tested with $3:1$ ratio for each UAV class. Optimum hyperparameters are determined after running a vast amount of simulations. Results are presented in the next section.

An illustration of the CNN architecture is shown in Fig. 5. While training the models, the categorical cross-entropy function is used as the loss function

$$\mathcal{L}(W) = -\frac{1}{N}\sum_{i=1}^{N} [y^{(i)} \log(\hat{y}^{(i)}) + (1-y^{(i)}) \log(1-\hat{y}^{(i)})], \qquad (10)$$

where W represents the model parameters, and $y^{(i)}$ and $\hat{y}^{(i)}$ represent the true labels and predicted labels for the i-th image, respectively. This function gets smaller as the true and the predicted results get closer to each other. The aim of the model is to find the optimum set of model parameters to minimize this function, i.e.,

$$\hat{W} = \underset{W}{\operatorname{argmin}}\, \mathcal{L}(W). \qquad (11)$$

Probability of the i-th test image, expressed as $\mathbf{x}^{(i)}$ in vector form, being a member of the k-th class is calculated using normalized exponential function as:

$$p_k(\mathbf{x}^{(i)}) = \frac{e^{\hat{v}_k^{(i)}}}{\sum_j e^{\hat{v}_j^{(i)}}}, \qquad (12)$$

where $\hat{\mathbf{v}}^{(\mathbf{i})}$ is the $K \times 1$ vector output of the final model that uses optimized weights given in (*(11)*), and K is the number of classes. The class that has the maximum probability is chosen to be the prediction of the model for the i-th test image, $\hat{y}^{(i)}$, for the given image

$$\hat{y}^{(i)} = \underset{k}{\operatorname{argmax}}\, p_k(\mathbf{x}^{(i)}). \qquad (13)$$

The next section presents the experimental results that are acquired with the CNN models created for both the time-series and spectrogram images. Note that data sets used to train CNN models include either time-series images or spectrograms.

5. EXPERIMENTAL RESULTS

During training the CNN models, the original data set in [26], where the SNR is about 30 dB for the whole set, was used. We extended this data set by considering four

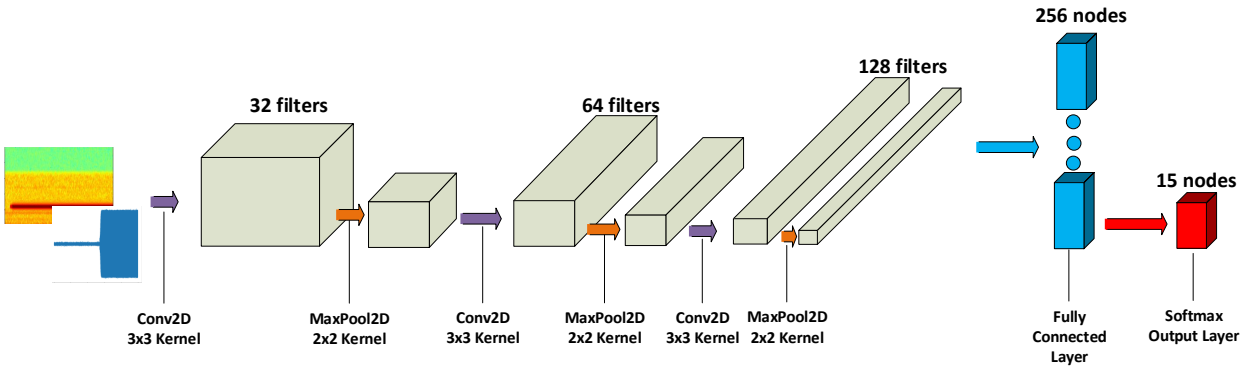


Fig. 5 – CNN architecture formed of three convolution and pooling layer pairs followed by a fully connected and a softmax output layer. The number of filters increase as data travels deep into the model to capture a widening variety of features better. Softmax output layer gives a set of predictions resolving the maximum likelihood of the signals class reference. The one with the highest probability is predicted by the model to be the class of the signal.

additional SNR levels ranging from 0 dB to 20 dB for time-series signal-based classification, and seven different additional SNR levels ranging from -10 dB to 20 dB for spectrogram-based classification, with SNR increments of 5 dB in both cases. To train the models, we created 100 images for each class and for each unique SNR truncation threshold pairs following the noising procedure described in Section 3.2 and the denoising procedure (for spectrogram images only) described in Section 4.2. Throughout the study, we created more than 100 data sets, each having 1500 images (15 classes with 100 images each). A Hanning window function of size 128 with 16 overlap samples is used while creating the spectrograms.

Before feeding the CNN, we crop the images appropriately to get rid of the unnecessary parts of the images and reduce the file sizes, which helps speed up the converging of the CNN models. Resulting spectrogram and time-series signal images have the sizes of $(90 \times 385 \times 3)$ and $(779 \times 769 \times 1)$, respectively. In this work, we used brute force searching to optimize CNN model parameters. We utilized the NC State University HPC (High Performance Computing) Facility to run parallel simulations for different sets of hyperparameters to find the optimum parameter set.

Note that, after rigorous simulations, the optimum activation function came up to be ReLu for all hyperparameter combinations. We also used single stride in both directions on images with no dilation, and valid padding for all models created regardless of SNR level and type of data. Remaining details of the models are given in Fig. 5, and Table 3 and Table 4.

In the rest of this section, we will first give considerations about the environmental interference issues and then present the classification results for the time-series images and spectrogram images. Subsequently, we will discuss the relation between classification accuracy and training set size and, finally, share the results for out-of-library classification performance of the proposed model.

Table 3 – Optimum set of hyperparameters for models trained on time-series images.

SNR (dB)	Optimizer	Batch size	Validation accuracy (%)
30	SGD	4	99.7
20	Adagrad	4	96.5
10	Nadam	16	81.6
5	Nadam	1	65.3
0	Adagrad	1	50.1

5.1 Comments on environmental interference

All the signals used in this study are recorded for a wide range of frequencies, i.e., $0-1$ 0 GHz, as illustrated by the spectrogram in Fig. 3(a). The first observation that can be made in there is that the frequency utilization significantly decreases above roughly 7 GHz, which is because there is no wireless transmission for that frequency range near the locations where we conducted the measurements. One can also notice the high color intensity at the GSM band around 1800 MHz. Since all of the drone controllers considered in this study transmit in the 2.4 GHz ISM band, notable densities in other bands on spectrograms have no effect on the model accuracy. However, the 2.4 GHz band is also used heavily by Wi-Fi and Bluetooth transmitters. In case Wi-Fi and/or Bluetooth signals are received, our proposed model applies a multistage detection system described in [26] to detect those type of signals and filter them out.

Raw data used in this work have been gathered in an indoor environment where Wi-Fi and Bluetooth signals could exist. A 24 dBi gain directional antenna has been used to capture the signals. It is known that IEEE 802.11 standards family routers implement Carrier-Sense Multiple

Table 4 – Optimum set of hyperparameters for models trained on spectrogram images.

SNR (dB)	Cut-off level (dB/Hz)	Optimizer	Batch size	Validation accuracy (%)
30	−100	Adamax	8	99.7
20	−90	Nadam	2	99.7
15	−10	Adam	32	100.0
10	−10	Nadam	2	100.0
5	−10	Adam	4	99.7
0	−10	Nadam	8	99.5
−5	−20	RMSProp	8	99.5
−10	−15	Nadam	16	92.0
Merged	−10	SGD	1	98.8
Merged*	−10	SGD	1	96.9

*Refers to the set of images created by assuming SNR levels different than the ones used to train the merged model.

Access (CSMA) techniques, which may help reducing the probability of interference with Wi-Fi transmitters when the drone controllers are close to the receiving antenna. Besides, low-power Bluetooth transmitters will not possess a high risk of severe distortion on the received signal. Moreover, our classifier makes a decision each time after processing a signal frame of 250 μs . A short duration of signals allow our system to catch drone controller signals even in the existence of other packet-based communication technologies as they do not transmit packets continuously. While capturing a drone data-only signal frame may introduce time delays in identifying the drone, this delay will be on the order of milliseconds. Therefore, we can safely conclude that labeling the training set as if there are no WiFi and Bluetooth signals complies with real-world scenarios.

5.2 UAV classification using time-series images

We optimize five different CNNs for time-series images by a brute force searching approach. We ran simulations for each data set using all combinations of seven different optimizers, seven different batch sizes, and five different activation functions, which add up to 245 distinct simulations. The parameter set that gives the highest accuracy is chosen. Optimized parameters for these models are given in Table 3 for reproducibility. We observe that CNNs gather distinctive features from both the transient (i.e., the signal segment where the noise-only region ends and the RF signal begins) and the envelope of the RF signal. As the signal swamps into noise as SNR decreases, first the transient information disappears whereas the information carried in the signal envelope survives a little longer. When the SNR is further decreased, envelope information also disappears. Thus, the validation accuracy

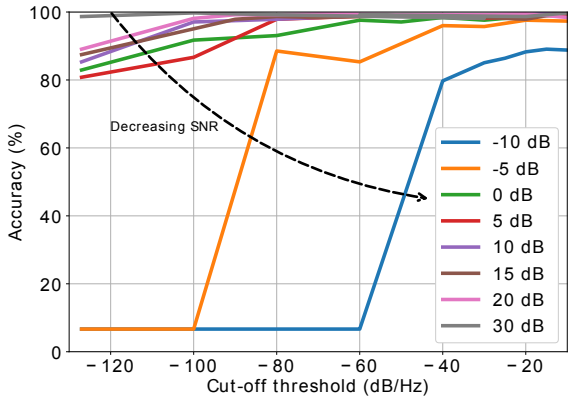


Fig. 6 – Spectrogram model classification accuracy versus the cut-off threshold for different SNR levels. Denoising the spectrograms by truncating the spectral densities, subject to a threshold, increases the model accuracy in general. Models trained with high-SNR data give reasonable accuracy even without denoising. Low-SNR models need to be denoised a priori.

drops from 99.7% to 50.1% as the SNR goes down from around 30 dB to 0 dB. Even though different optimizers could give the maximum accuracy for different SNR levels, all optimum models use Rectified Linear Unit (ReLu) as the activation function.

Both in-band and out-of-band noise cause distortion in time-series images of RF signals, and so the models trained on time-series images suffer from noise more than the models that use the spectrogram images. Besides, while using the time-series images, trained models extract features from the amplitude of the signals itself. However, the amplitude of a received signal depends on the distance between the receiver and transmitter antenna. This is an obvious problem when the only distinctive difference between the time-series signal images of any two controllers is the difference in their amplitude (e.g., see the RF signals in in Fig. 2(c) and Fig. 2(d)). It is worth noting that best results reported for the same number of classes in the previous work [26] that uses kNN (k-nearest neighbors), random forest and discriminant analysis techniques are slightly better for high SNR levels ($\approx 98\%$ vs 96.5% at 20 dB). However our results for models trained on time-series images are significantly better for low SNR levels ($\approx 40\%$ vs 50.1%) . Moreover, models trained on spectrograms show even better performances. The next subsection is dedicated to results of models that employ spectrogram images.

5.3 UAV classification using spectrogram images

Two approaches are adopted while creating models on spectrogram images. In the first approach, we assumed that the SNR level of a received signal can be measured prior to classification and created different models for different SNR levels. In the second approach, we used a merged data set that includes spectrogram images of different SNRs to create a model that can be used to classify any received signal without any prior information about its SNR. Details of these approaches are given in the following subsections.

5.3.1 *Models with single SNR training sets*

We have created eight models for eight different SNR levels that are truncated at their own optimum levels. To use this approach, the SNR of the received signal should be calculated first, and then the model that has the closest SNR should be called to perform the classification. We observed that all of the models give their highest accuracy with the ReLu activation function. The sensitivity of the validation accuracy to a single output was found to be 0.27% sample^{-1}. Optimized parameters of the models using spectrogram images are given in Table 4. It is seen that the lowest accuracy belongs to the SNR level −10 dB among the individual sets. Performances of all the other models can be considered almost perfect. It is also observed from Table 4 that the optimum cut-off levels are different for different SNR levels.

Classification accuracy at different truncation thresholds for different SNR levels are given in Fig. 6. By considering this figure and Table 4 together, one can conclude that, in general, the classification accuracy tends to increase with the increasing level of truncation. For high SNRs (i.e., 20 dB and 30 dB), spectral densities of the signals are much higher than the noise; therefore, truncating the images at different levels does not wipe out much information. As a result, the accuracy curve navigates flatter, and the necessary cut-off threshold is low (−100 dB/Hz and −90 dB/Hz). At medium SNRs (i.e., 0−15 dB), a high level of truncation is required to preserve as much information as possible (all −10 dB/Hz). On the other hand, at the lowest end of SNRs (i.e., −5 dB and −10 dB), without truncating the images, no learning occurs at all. For these lowest two SNRs, distinctive information in the spectrograms is swamped into noise so with no truncation, the accuracy is found to be only 6.66%. As the cut-off threshold increases, first a reasonable accuracy is acquired for a −5 dB SNR data set at the −80 dB/Hz threshold level. This amount of filtering is still not sufficient for −10 dB SNR, which only begins to learn at a comparably higher threshold of −40 dB/Hz. Moreover, the −10 dB/Hz threshold level gives lower accuracy than the models trained at medium SNRs (i.e., 0−15 dB) using the same threshold. This is because over-denoising chops the meaningful information together with the noise, and consequently, the optimum cut-off level is slightly lower than −10 dB/Hz (i.e., −20 dB/Hz for −5 dB SNR and −15 dB/Hz for −10 dB SNR). If the cut-off threshold is too high, this wipes out all the information, making all spectrograms look alike and consequently, there will be no learning.

The advantage of using spectral domain information could be seen from the results of a 0 dB SNR model where the classification accuracy for time-series images is only 50.1% (Table 3), whereas it is 82.9% (Fig. 6) for the spectrogram model at the same SNR level without denoising. Also note that, CNN models optimized using proposed denoising technique perform substantially better than both time-series image models and the models in [26], where conventional ML techniques are used at the latter, for every SNR level. For example, classification accuracies reported in [26] range from 40% to 98%, whereas CNN models trained on spectrograms with denoising range between 99.5% and 100% for SNR levels from 0 dB to 30 dB.

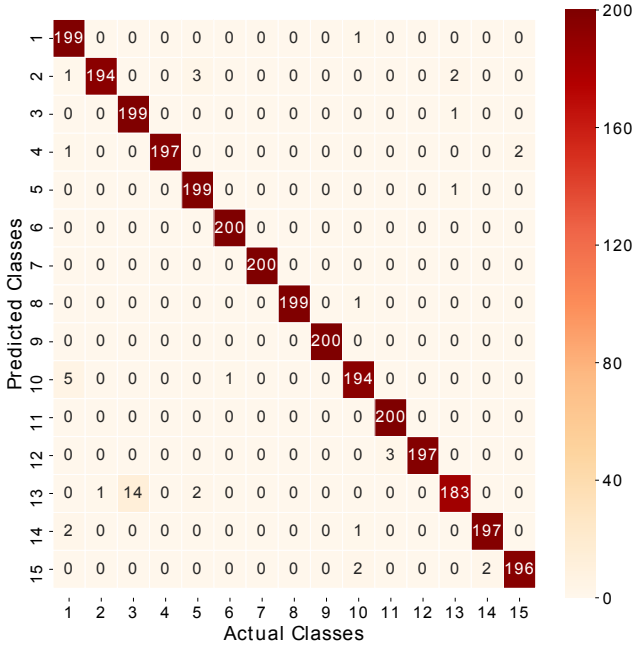


Fig. 7 – Confusion matrix for the merged model. Diagonal elements represent true positives, and off-diagonal elements represent the confusion between the classes.

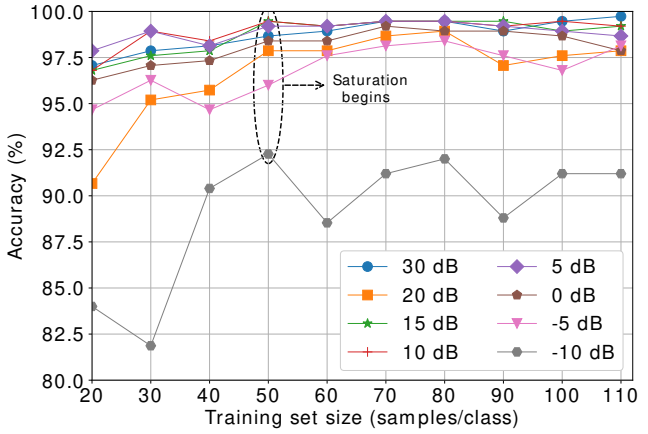


Fig. 8 – Classification accuracy as a function of training set size. Models give reasonable accuracies even with very low training data sizes. Saturation begins after 50 samples/class. We used 75 samples/class throughout this work.

5.3.2 *Model with a merged training set*

Even though the models trained with different single-SNR data sets give satisfactory results, this approach comes with a practical difficulty. We can use these models only if

we can measure the SNR of a received signal prior to classification. In order to get rid of this requirement and also to save time, we merged the training sets of different SNR levels to create a more generalized model. The truncation cut-off that gives the smallest average loss was found to be -10 dB/Hz, therefore we merged all the images for eight SNR levels denoised at this cut-off threshold. Training and test sets are eight times greater than those of the single SNR sets. A classification accuracy of 98.8% (across all SNR levels) is achieved when using this model.

A confusion matrix of the merged model is given in Fig. 7. The major deficiency of the model is observed at $(13, 3)$, where 14 out of 200 test data which belongs to class 3 is predicted as class 13. These two controllers belong to the same company and both their time-series plots and spectrograms show high virtual resemblance.

We also tested the merged model with images at intermediate SNR levels ranging from -12 dB to 22 dB with increments of 5 dB. We used 30 images for each class at each SNR, which add up to 3600 images, all previously unseen to the classifier, to test the model. Our model gives 96.9% accuracy, as shown in Table 4. In the case when we exclude the test data at -12 dB, the accuracy of the model increases up to 99.3%, which indicates that almost all the misclassification is associated with this particular SNR level.

5.4 Classification accuracy vs. training set size

There are popular CNN models in the literature that can be implemented to a wide variety of image classification problems via transfer learning, e.g., VGG16 or InceptionV3. These models have abundant hidden layers and have been trained over enormous data sets. Other than these models, it is more customary to come across CNN models that are deeper and trained on larger data sets in the literature. If the problem in hand is to accurately classify images of miscellaneous objects, e.g., humans, animals or cars, then a deep model with a very large number of training set should be required. This is because these images have more diversity in terms of position, angle, ambiance, lightning, etc. However, in our model, the set of images that we classify are generated by the well-defined methods that use the outputs of quite robust electronic circuitry. Thus, proposed models reach very high accuracy with as low as 100 training samples per class.

To better explain the sufficiency of a low number of samples for this particular problem, we examined the dependence of accuracy to the training data set size. Fig. 8 shows the accuracy of the classifier with respect to sample size per class for different SNR levels. In these simulations, the same models that are optimized for 100 samples per class are used for all cases. Note that, in this figure, x-axis denotes the size of the training sets only. We did not shrink the validation set while we tune the training set sizes. All models have been validated with a test

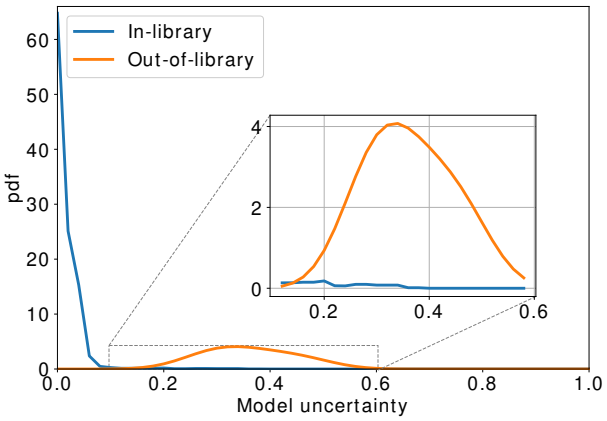


Fig. 9 – Pdfs of the model uncertainty for in-library and out-of-library UAV classes. The model predicts in-library signals with high certainty. The prediction uncertainty increases when the model encounters an out-of-library controller.

set of 25 samples per class unseen by the models before. Here it is seen that for small training set sizes, classification accuracy decreases as expected. After roughly 50 samples/class, the accuracy reaches saturation and begins to fluctuate. On the other hand, we see that the created models give reasonable accuracy even for a training set size of as low as 20 samples per class, which is because these samples are created by devices that have a high level of consistency.

For the practicality of the proposed system, the RF signal database should be updated as new products are introduced to the market. This also requires retraining of the CNN models, and hence fast training algorithms are needed. However, as explained above, the proposed system only requires a limited amount of training data, which in turn makes it a promising solution.

5.5 Out-of-library UAV controller signals

Finally, we investigate the behavior of the proposed algorithm when the receiver captures an out-of-library UAV controller signal. To do that, we tested our optimized CNN-based classifier for 40 signals from a Hubsan H501S X4 drone and compared the estimated probability distribution functions (pdfs) of the prediction uncertainty with those of the in-library test signals in Fig. 9. The output layer of the trained model gives a set of predictions for an incoming signal, where each element of the set corresponds to the estimated probability of that signal belonging to a particular class. A final decision on the class of the test signal is made based on the maximum probability, $p_{\max}$, in this set. We define the model uncertainty in Fig. 9 as $(1 - p_{\max})$.

We observe that the two classes (i.e., in-library and out-of-library UAVs) are well separated in terms of the model uncertainty associated with each of them, and out-of-library UAV signals can be easily identified by a simple thresholding mechanism. The threshold can be placed

based on the system requirements, i.e., the desired classification performance and false alarm rate. We recognize that a complete consideration of out-of-library classification requires adding out-of-library data in the training set or adaptation of open set recognition by introducing an OpenMax layer, which estimates the probability of an input being from an unknown class [38]. On the other hand, our proposed model gives very low model uncertainty for in-library signals, and this therefore still provides a reasonable solution described above to detect the out-of-library drones in practice.

6. CONCLUSION

In this study, we proposed a system that uses drone controller RF signals to classify drones of different makes and models for a wide variety of SNRs. We used CNN classifiers with two different sources for training the models: time-series images, and spectrogram images. We showed that the CNN model using the spectrogram images is more resilient to noise when compared with the time-series images based model. The proposed method that uses a merged training set of RF signals at different SNR levels along with the proposed denoising mechanism was shown to be effective for UAV classification even at SNRs not directly considered by the trained model. We also explored classification performance against training set size and showed that reasonable classification accuracy can still be obtained with limited training data. Consequently, adding new classes to the model (e.g., to include data from newly released drones) does not entail a high computation cost. Finally, we examined the model behavior with in-library and out-of-library drone signals and concluded that the proposed model shows a good performance identifying drones from an unknown class. Our future work includes comparing our results with federated learning techniques, and testing of the proposed CNN-based UAV classification technique at a larger scale, such as using the AERPAW experimental platform at NC State University.

ACKNOWLEDGMENT

This work has been supported in part by NASA under the Federal Award ID number NNX17AJ94A. The authors would like to thank Martins Ezuma at NC State University for providing the drone controller RF data set used in this study.

REFERENCES

[1] Hazim Shakhatreh, Ahmad H Sawalmeh, Ala Al-Fuqaha, Zuochao Dou, Eyad Almaita, Issa Khalil, Noor Shamsiah Othman, Abdallah Khreishah, and Mohsen Guizani. “Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges”. In: *IEEE Access* 7 (2019), pp. 48572–48634.

[2] Dimas Pristovani Riananda, Galih Nugraha, Harish Mahatma Putra, Muhammad Lukman Baidhowi, and Riza Alaudin Syah. “Smart pulley workflow in delivery drone for goods transportation”. In: *AIP Conference Proceedings*. Vol. 2226. 1. 2020, p. 060010.

[3] Connie A Lin, Karishma Shah, Lt Col Cherie Mauntel, and Sachin A Shah. “Drone delivery of medications: Review of the landscape and legal considerations”. In: *The Bulletin of the American Society of Hospital Pharmacists* 75.3 (2018), pp. 153–158.

[4] Vinay Chamola, Vikas Hassija, Vatsal Gupta, and Mohsen Guizani. “A Comprehensive Review of the COVID-19 Pandemic and the Role of IoT, Drones, AI, Blockchain, and 5G in Managing its Impact”. In: *IEEE Access* 8 (2020), pp. 90225–90265.

[5] Walid Saad, Mehdi Bennis, and Mingzhe Chen. “A Vision of 6G Wireless Systems: Applications, Trends, Technologies, and Open Research Problems”. In: *IEEE Network* 34.3 (2020), pp. 134–142. DOI: 10.1109/MNET.001.1900287.

[6] M. Ritchie, F. Fioranelli, H. Griffiths, and B. Torvik. “Micro-drone RCS analysis”. In: *Proc. IEEE Radar Conf.* Johannesburg, South Africa, Oct. 2015, pp. 452–456.

[7] Ismail Guvenc, Farshad Koohifar, Simran Singh, Mihail L Sichitiu, and David Matolak. “Detection, tracking, and interdiction for amateur drones”. In: *IEEE Commun. Mag.* 56.4 (2018), pp. 75–81.

[8] Crown Consulting. *NASA Urban Air Mobility (UAM) Market Study*. URL: https://ntrs.nasa.gov/citations/20190026762.

[9] Federal Aviation Agency. *Proposed Rule on Remote Identification of Unmanned Aircraft Systems*.

[10] H. Zhang, C. Cao, L. Xu, and T. A. Gulliver. “A UAV Detection Algorithm Based on an Artificial Neural Network”. In: *IEEE Access* 6 (May 2018), pp. 24720–24728.

[11] W. Zhou, L. Wang, B. Lu, N. Jin, L. Guo, J. Liu, H. Sun, and H. Liu. “Unmanned Aerial Vehicle Detection Based on Channel State Information”. In: *Proc. IEEE Int. Conf. Sensing Commun. Netw. (SECON)*. Hong Kong, China, June 2018, pp. 1–5.

[12] M. Z. Anwar, Z. Kaleem, and A. Jamalipour. “Machine Learning Inspired Sound-Based Amateur Drone Detection for Public Safety Applications”. In: *IEEE Trans. Veh. Technol.* 68.3 (Jan. 2019), pp. 2526–2534.

[13] Y. Seo, B. Jang, and S. Im. “Drone Detection Using Convolutional Neural Networks with Acoustic STFT Features”. In: *Proc. IEEE Int. Conf. Advanced Video Signal Based Surveillance (AVSS)*. Nov. 2018, pp. 1–6.

[14] V. Thai, W. Zhong, T. Pham, S. Alam, and V. Duong. "Detection, Tracking and Classification of Aircraft and Drones in Digital Towers Using Machine Learning on Motion Patterns". In: *Proc. Integrated Commun. Navig. Surveillance Conf. (ICNS)*. Herndon, VA, Apr. 2019, pp. 1–8.

[15] M. Saqib, S. Daud Khan, N. Sharma, and M. Blumenstein. "A study on detecting drones using deep convolutional neural networks". In: *Proc. IEEE Int. Conf. Advanced Video Signal Based Surveillance (AVSS)*. Lecce, Italy, Aug. 2017, pp. 1–5.

[16] J. Ren and X. Jiang. "Regularized 2-D Complex-Log Spectral Analysis and Subspace Reliability Analysis of Micro-Doppler Signature for UAV Detection". In: *Pattern Recognit.* 69 (Mar. 2017), pp. 225–237.

[17] M. Marco and G. Pinelli. "Classification of Drones with a Surveillance Radar Signal". In: *Proc. Int. Conf. Comput Vision Syst.* Thessaloniki, Greece, Sept. 2019, pp. 723–733.

[18] P. Zhang, L. Yang, G. Chen, and G. Li. "Classification of drones based on micro-Doppler signatures with dual-band radar sensors". In: *Proc. Progress Electromagn. Research Symp. (PIERS)*. Singapore, Singapore, Nov. 2017, pp. 638–643.

[19] A. Huizing, M. Heiligers, B. Dekker, J. de Wit, L. Cifola, and R. Harmanny. "Deep Learning for Classification of Mini-UAVs Using Micro-Doppler Spectrograms in Cognitive Radar". In: *IEEE Trans. Aerosp. Electron. Syst.* 34.11 (Nov. 2019), pp. 46–56.

[20] B. K. Kim, H. Kang, and S. Park. "Drone Classification Using Convolutional Neural Networks With Merged Doppler Images". In: *IEEE Geosci. Remote Sens. Lett.* 14.1 (Jan. 2017), pp. 38–42.

[21] B. Oh, X. Guo, F. Wan, K. Toh, and Z. Lin. "Micro-Doppler Mini-UAV Classification Using Empirical-Mode Decomposition Features". In: *IEEE Geosci. Remote Sens. Lett.* 15.2 (Feb. 2018), pp. 227–231.

[22] P. Molchanov, K. Egiazarian, J. Astola, R. I. A. Harmanny, and J. J. M. de Wit. "Classification of small UAVs and birds by micro-Doppler signatures". In: *Proc. Eur. Radar Conf.* Nuremberg, Germany, Oct. 2013, pp. 172–175.

[23] L. Wang, J. Tang, and Q. Liao. "A Study on Radar Target Detection Based on Deep Neural Networks". In: *IEEE Sens. Lett.* 3.3 (Jan. 2019), pp. 1–4.

[24] A. Alipour-Fanid, M. Dabaghchian, N. Wang, P. Wang, L. Zhao, and K. Zeng. "Machine Learning-Based Delay-Aware UAV Detection and Operation Mode Identification Over Encrypted Wi-Fi Traffic". In: *IEEE Trans. Inf. Forensics Security* 15 (Dec. 2019), pp. 2346–2360.

[25] N. Regev, I. Yoffe, and D. Wulich. "Classification of single and multi propelled miniature drones using multilayer perceptron artificial neural network". In: *Proc. Int. Conf. Radar Syst.* Belfast. UK, Jan. 2017, pp. 1–5.

[26] M. Ezuma, F. Erden, C. Kumar Anjinappa, O. Ozdemir, and I. Guvenc. "Detection and Classification of UAVs Using RF Fingerprints in the Presence of Wi-Fi and Bluetooth Interference". In: *IEEE Open J. Commun. Soc.* 1 (Nov. 2019), pp. 60–76.

[27] Martins Ezuma; Fatih Erden; Chethan K. Anjinappa; Ozgur Ozdemir; Ismail Guvenc. "Drone Remote Controller RF Signal Dataset". In: IEEE Dataport, 2020. DOI: 10 . 21227 / ss99 – 8d56. URL: https://dx.doi.org/10.21227/ss99-8d56.

[28] Z. Shi, X. Chang, C. Yang, Z. Wu, and J. Wu. "An Acoustic-Based Surveillance System for Amateur Drones Detection and Localization". In: *IEEE Trans. Veh. Technol.* 69.3 (Mar. 2020), pp. 2731–2739.

[29] C. Aker and S. Kalkan. "Using deep networks for drone detection". In: *Proc. IEEE Int. Conf. Advanced Video Signal Based Surveillance (AVSS)*. Lecce, Italy, Aug. 2017, pp. 1–6.

[30] Y. Zhao and Y. Su. "The Extraction of Micro-Doppler Signal With EMD Algorithm for Radar-Based Small UAVs' Detection". In: *IEEE Trans. Instrum. Meas.* 69.3 (Apr. 2020), pp. 929–940.

[31] B. Choi and D. Oh. "Classification of Drone Type Using Deep Convolutional Neural Networks Based on Micro-Doppler Simulation". In: *Proc. Int. Symp. Antennas Propag. (ISAP)*. Busan, South Korea, Oct. 2018, pp. 1–2.

[32] A. Shoufan, H. M. Al-Angari, M. F. A. Sheikh, and E. Damiani. "Drone Pilot Identification by Classifying Radio-Control Signals". In: *IEEE Trans. Inf. Forensics Security* 13.10 (Mar. 2018), pp. 2439–2447.

[33] Roberto Brunelli. *Template Matching Techniques in Computer Vision: Theory and Practice*. John Wiley & Sons, Ltd, 2009. ISBN: 9780470744055.

[34] P. Sysel and Z. Smékal. "Enhanced estimation of power spectral density of noise using the wavelet transform". In: *Pers. Wireless Commun.* Boston, MA: Springer US, 2007, pp. 521–532. ISBN: 978-0-387-74159-8.

[35] Francisco-Cervantes Cervantes, Jesus Gonzalez-Trejo, Cesar Real-Ramirez, and Hoyos-Reyes Luis. "Fractal dimension algorithms and their application to time series associated with natural phenomena". In: *J. Phys.: Conf. Ser.* 475 (Nov. 2013).

[36] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, 2019.

[37] A. Bhandari, H. Yin, Y. Liu, W. Yao, and L. Zhan. "Real-Time Signal-to-Noise Ratio Estimation by Universal Grid Analyzer". In: *Proc. Int. Conf. Smart Grid Sync. Meas. Anal. (SGSMA)*. College Station, TX, Aug. 2019, pp. 1–6.

[38] A. Bendale and T. E. Boult. "Towards Open Set Deep Networks". In: *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*. Las Vegas, NV, Dec. 2016, pp. 1563–1572.

AUTHORS

Ender Ozturk received his B.S. and M.S. degrees from Bilkent University, Ankara, Turkey, in 2005 and 2008, respectively, and the Ph.D. degree from Hacettepe University, Ankara, Turkey, in 2018, all in electrical and electronics engineering. He held different positions in industry as an engineer, senior engineer and team leader. He is currently working as a Postdoctoral Research Scholar at North Carolina State University.

Fatih Erden received the B.S. and M.S. degrees from Bilkent University, Ankara, Turkey, in 2007 and 2009, respectively, and the Ph.D. degree from Hacettepe University, Ankara, Turkey, in 2015, all in electrical and electronics engineering. From 2015 to 2016, he was an Assistant Professor at Atilim University, Ankara, Turkey. He is now working as a research associate at North Carolina State University. His research interests include signal and image processing, machine learning, time series analysis, mmWave communications, and UAVs.

Ismail Guvenc is Professor at North Carolina State University. His recent research interests include 5G wireless networks, UAV communications, and heterogeneous networks. He has published more than 200 conference/journal papers, several standardization contributions, three books, and over 30 U.S. patents. He is a recipient of the 2019 R. Ray Bennett Faculty Fellow Award, 2016 FIU COE Faculty Research Award, 2015 NSF CAREER Award, 2014 Ralph E. Powe Junior Faculty Award, and 2006 USF Outstanding Dissertation Award, and he is a senior member of the National Academy of Inventors.

3-OF-3 MULTISIGNATURE APPROACH FOR ENABLING LIGHTNING NETWORK MICRO-PAYMENTS ON IOT DEVICES*

Ahmet Kurt[1], Suat Mercan[1], Enes Erdin[2], Kemal Akkaya[1]
[1]Department of Electrical and Computer Engineering, Florida International University, Miami, FL 33174, United States,
[2]Department of Computer Science, University of Central Arkansas, Conway, AR 72035, United States

NOTE: Corresponding author: Ahmet Kurt, akurt005@fiu.edu

*A preliminary version of this paper was accepted as a poster paper to 2021 IEEE International Conference on Blockchain and Cryptocurrency (IEEE ICBC 2021).

Abstract – *Bitcoin's success as a cryptocurrency enabled it to penetrate into many daily life transactions. Its problems regarding the transaction fees and long validation times are addressed through an innovative concept called the Lightning Network (LN) which works on top of Bitcoin by leveraging off-chain transactions. This made Bitcoin an attractive micro-payment solution that can also be used within certain IoT applications (e.g., toll payments) since it eliminates the need for traditional centralized payment systems. Nevertheless, it is not possible to run LN and Bitcoin on resource-constrained IoT devices due to their storage, memory, and processing requirements. Therefore, in this paper, we propose an efficient and secure protocol that enables an IoT device to use LN's functions through a gateway LN node even if it is not trusted. The idea is to involve the IoT device only in signing operations, which is possible by replacing LN's original 2-of-2 multisignature channels with 3-of-3 multisignature channels. Once the gateway is delegated to open a channel for the IoT device in a secure manner, our protocol enforces the gateway to request the IoT device's cryptographic signature for all further operations on the channel such as sending payments or closing the channel. LN's Bitcoin transactions are revised to incorporate the 3-of-3 multisignature channels. In addition, we propose other changes to protect the IoT device's funds from getting stolen in possible revoked state broadcast attempts. We evaluated the proposed protocol using a Raspberry Pi considering a toll payment scenario. Our results show that timely payments can be sent and the computational and communication delays associated with the protocol are negligible.*

Keywords – Bitcoin, Internet of Things, lightning network, multisignature, payment channel networks

1. INTRODUCTION

The Internet of Things (IoT) has been adopted in various domains at a great pace in the last decade as it brings numerous opportunities and convenience [1]. In such applications, typically resource-constrained IoT devices supply data from their sensors to remote servers through a wireless connection. With their increased capabilities, we have been witnessing applications where an IoT device may need to do financial transactions. For instance, IoT devices may be used in commercial applications such as toll systems, where an on-board unit acting as an IoT device on a vehicle may need to do automatic payments as the vehicle passes through a toll gate [2]. Similarly, there are other cases such as automated vehicle charging, parking payment, sensor data selling, etc. where *micro-payments* need to be made [3, 4].

In these applications, the common feature is device-to-device (D2D) communication which may not involve any human intervention. Therefore, transactions should be automated. While these automated payments may be linked to credit card accounts of device owners, this is not only inconvenient but also requires the involvement of third parties that will bring additional management overhead. In this context, cryptocurrencies have great potential to provide a smooth payment automation. Thus, a successful merge of IoT and cryptocurrency technologies such as Bitcoin [5] and Ethereum [6] looks promising.

However, despite their popularity, mainstream cryptocurrencies such as Bitcoin and Ethereum suffer from scalability issues in terms of transaction confirmation times and throughput [7]. This increases the transaction fee and makes their adoption infeasible for micro-payments. The Payment Channel Network (PCN) idea has emerged as a second layer solution to address this problem by utilizing off-chain transactions [8]. For instance, Lightning Network (LN) [9] is the PCN solution designed for Bitcoin which exceeded 20,000 nodes in three years. While LN is a successful solution, the current LN protocol cannot be run on most IoT devices because of the computation, communication, and storage requirements [10]. As well known, IoT devices are mostly resource-constrained and most of them are not capable of running the LN protocol where a full Bitcoin node (e.g., as of today 349 GB of storage area is required) has to be running alongside an LN node. Additionally, a robust Internet connection and relatively high computation power are required to receive and verify new blocks for the Bitcoin node. Even if we can empower some IoT devices with the needed resources, these IoT devices still need to be always online to receive synchronization messages from both Bitcoin and LN, which is not realistic for IoT either.

Therefore, there is a need for a lightweight solution that will enable resource-constrained IoT devices to utilize LN for micro-payments. To this end, in this paper, we propose an efficient and secure protocol where an IoT device can connect to an *untrusted LN gateway* that already hosts the full LN and Bitcoin nodes and can: 1) open/close LN channels and 2) send payments on behalf of the IoT device when requested. Our approach is similar to a delegation approach which comes with almost negligible computation and communication overheads for the IoT devices. We are proposing to incentivize the LN gateway to participate in this payment service by letting it charge IoT devices for each payment it processes.

In our proposed protocol, we introduce the concept of *3-of-3 multisignature LN channels*, which involves signatures of all three channel parties (i.e., the IoT device, the LN gateway, and a bridge LN node to which the LN gateway opens a channel for the IoT device) to conduct any operation on the channel as opposed to using the LN's original 2-of-2 multisignature channels. This modification to the channels is possible by changing the LN's Bitcoin scripts which play a critical role in our protocol as it prevents the LN gateway from spending the IoT device's funds. More specifically, the LN gateway cannot spend the IoT device's funds in the channel without getting the IoT device's cryptographic signature which consequently means that the IoT device's funds are secure at all times. Since LN's original protocol is modified, this also necessitates revisiting the *revoked state broadcast* issue of LN. We offer revisions to protect the IoT device's funds in revoked state broadcast attempts when 3-of-3 multisignature LN channels are used.

To assess the effectiveness and overhead of the proposed protocol, we implemented it within a setup where a Raspberry Pi sends LN payments to a real LN node through a wireless connection. We considered two real-life cases in the experiments: a vehicle at a certain speed making a toll payment through a wireless connection and a customer paying for a coffee at a coffee shop using his/her smartwatch. We demonstrated that the proposed protocol enables the realization of timely payments with negligible computational and communication delays. We separately provide a security analysis of the proposed protocol.

The structure of the rest of the paper is as follows. In Section 2, we provide the related work. Section 3 describes the LN, its underlying mechanisms and its protocol specifications. System and threat model are explained in Section 4. We explain the proposed protocol in detail in Section 5. Security analysis against the assumed threats is provided in Section 6. We present our evaluation results for the proposed protocol in Section 7. Finally, the paper is concluded in Section 8.

2. RELATED WORK

The closest work to ours is from Hannon and Jin [11]. They propose to employ LN-like payment channels to give IoT devices the ability to perform off-chain transactions. Since an IoT device does not have access to the blockchain, they use a pool of two different third parties which are called the IoT payment gateway and watchdog to aid the IoT device in the process. Using game theory, they show that the protocol reaches an equilibrium given that the players follow the protocol. However, this approach has a major issue: They assume that the IoT device can actually open LN-like payment channels to the IoT gateway. While it is not clear how it can be done, the authors also do not have a proof of concept implementation of the approach which is another puzzling aspect of the work.

Robert et al. [12] proposed IoTBnB, a digital IoT marketplace, to let data trading between the data owners and the users. In their scheme, after the user selects which item to buy from the marketplace, it is redirected to an LN module that performs the payment. This LN module is hosting the full Bitcoin and LN nodes. However, the authors are focused on integrating LN into an existing IoT ecosystem rather than enabling individual IoT devices to use LN that are not part of such an ecosystem. Additionally, in their system, the IoT device's funds are held by wallets belonging to the ecosystem which raises security and privacy concerns. In our work, we cover a broader aspect of IoT applications and IoT's funds are not held by third parties.

A different work focusing on Ethereum micro-payments rather than Bitcoin was proposed by Pouraghily and Wolf [13]. They employ a Ticket-Based Verification Protocol (TBVP) for a similar purpose, enabling IoT devices to perform financial transactions in an IoT ecosystem. By using two entities called contract manager and transaction verifier, attempts are made to reduce the performance requirements on the IoT devices. There are some issues with this approach: 1) It is mentioned that the IoT funds are deposited into an account jointly opened with a partner device. Since the details are not provided, it raises security and privacy concerns. 2) The scheme utilizes Ethereum smart contracts and TBVP was compared with μRaiden [14] which is an Ethereum payment channel framework targeting low-end devices. However, this comparison might not be reliable as μRaiden development was dormant for more than 2 years. In our work, we targeted Bitcoin and LN which are actively being developed and currently dominating the market.

A recent work by Profentzas et al. [15] proposes TinyEVM, an Ethereum based off-chain smart contract system to enable IoT devices to perform micro-payments. The authors tried to tackle the problem by modifying Ethereum virtual machine and running it on the IoT device. In our approach, we only require the IoT device to generate signatures, which is not a resource-intensive operation. Another work, [16], focuses on data transactions for IoT using payment channels. A slightly different work by

authors of [17] explores using LN for delivering patch updates to the IoT devices. They employ LN in the process of claiming rewards upon successful delivery of the patch updates.

There have been commercialized implementation efforts to create lighter versions of LN for low-resource devices. Neutrino [18] is one of them which is a Bitcoin light client specifically designed for LN. The idea is to use the block headers only as opposed to using the whole blockchain. Breez [19] is another example which a mobile client based on lnd [20] and Neutrino. While a portion of the IoT devices might be able to run these software, they still need to be online to synchronize block headers. Thus, we propose a solution that does not require staying online all the time or synchronizing any messages after coming online. This work is an extension to our poster paper [21] with a lot of new content. 1) The poster version does not have the full details of the proposed protocol. In this paper, we explain the protocol in detail and show the modifications to LN's existing specifications. 2) The poster version does not have related work and background sections. In this paper, we give comprehensive background information on LN and provide the related work. 3) In this journal version, we present the threat model and the security analysis that did not exist in the poster version. 4) The evaluation section in the poster version only has the WiFi experiments. In this paper, we present Bluetooth experiments in addition to the WiFi. Additionally, we present another use-case scenario in the experiments and provide a cost analysis.

3. BACKGROUND

In this section, we provide comprehensive background information on LN, its components and specifications to help understand our approach.

3.1 Lightning network

LN was introduced by Poon and Dryja in 2015 in a technical paper [9]. After 2 years of its introduction, it was implemented by Lightning Labs and started being used on Bitcoin mainnet [22]. The intuition behind developing LN which is a layer-2 payment channel network is to solve Bitcoin's *scalability* problem. Similar to Bitcoin, LN is a peer-to-peer distributed network but it is not standalone, rather it is operating on top of Bitcoin. The idea of creating networks on top of blockchains is not new [8]. For Bitcoin, it is possible to create and deploy such networks like LN by utilizing its *smart contract* feature [23]. In this way, secure *payment channels* can be established which can be used by users for instant and almost free Bitcoin transactions. When enough of these payment channels are opened by the users, they form a network of payment channels. Then, this network can be utilized by new users to route their payments to specific destinations. Such payments where the existing channels on LN are utilized are called multi-hop payments. This feature of LN eases the onboarding process for new users to start using the network for sending/receiving payments.

While LN is mostly serving end users and merchants, there are other entities within it that serve different purposes. For example, an end user trying to connect to the LN using a mobile device will be connecting to a *gateway node*. Such gateway nodes directly serve the end users and earn fees by routing the payments. After connecting to the LN, the end user's payments have to be conveniently routed to other participants of the network. *Bridge nodes* serve this purpose. They enable the connectivity between the existing gateway nodes and also earn routing fees. Even though they have different names, bridge nodes and gateway nodes are essentially all regular LN nodes distinguished by their roles in the network. Fig. 1 summarizes the virtual topology of the LN. The network is highly scalable and can support millions of transactions per second. After its creation, LN grew exponentially reaching 21,310 nodes maintaining 48,915 channels at the time of writing this paper [24].

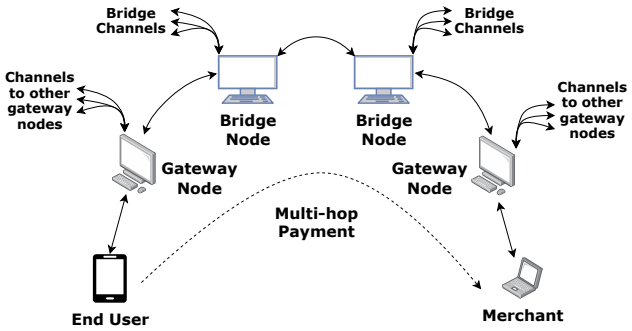


Fig. 1 – An illustration of the topology of LN adapted from [25]. Payments can be routed over existing channels.

3.2 Underlying LN mechanisms

In this section, we will briefly touch upon the key concepts of LN which are crucial to understanding the protocol descriptions. To explain these concepts, we use an example case where Alice opens an LN channel to Bob with the purpose of sending him LN payments.

Funding transaction: When Alice wants to open a channel to Bob, she needs to construct a proper funding transaction first. This on-chain transaction determines the channel capacity which is the amount of funds that will be committed to the channel. Once Alice creates the transaction, she sends the outpoint[1] to Bob. Receiving the outpoint, Bob can send a signature to Alice which will enable her to broadcast the funding transaction to the Bitcoin network. In Fig. 2, Alice opens a channel to Bob with a 5 Bitcoin capacity. Once funds are committed to the channel, she can send off-chain payments to Bob up to a total of 5 Bitcoins.

[1]Transaction outpoint is the combination of the transaction output and the output index.

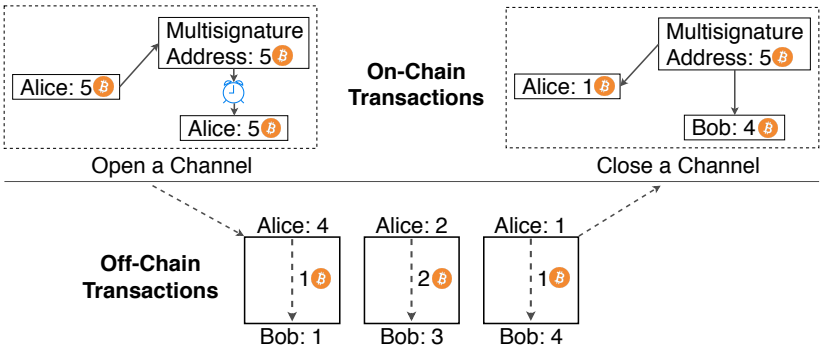


Fig. 2 – A depiction of the on-chain channel opening and closing transactions and off-chain payments in LN.

Commitment Transaction: When Alice starts sending Bob off-chain payments, her and Bob's balances on the channel will change. In Fig. 2, Alice sends 3 different payments to Bob with amounts 1 Bitcoin, 2 Bitcoins, 1 Bitcoin respectively. Since these transactions are not on-chain (i.e. not mined by miners thus not included in the blocks), there has to be a different mechanism to keep track of each parties' balances in the channel. This is done by the *commitment transactions*. A commitment transaction is a type of Bitcoin transaction specifically designed for LN. A payment channel consists of states, changing with each payment. In each state, parties have different balances which are recorded onto their commitment transactions. We illustrated Alice's commitment transaction in Fig. 3 after she initiates a payment of 1 Bitcoin to Bob. The inputs to her commitment transaction are: 1) funding transaction outpoint and a signature from Bob. She receives Bob's signature for each new state which enables her to broadcast her commitment transaction if required. As for the outputs, there are 3 of them as shown in Fig. 3. Output 1 is for Alice's balance on the channel. If she broadcasts her commitment transaction for any reason, she can claim her funds from this output after waiting k number of blocks. Output 2 is for Bob's balance on the channel which Bob can spend immediately. Finally, Output 3 is for the 1 Bitcoin payment Alice sent to Bob. We will explain it next.

Commitment Tx (Held by Alice)

Input	**Funding Transaction Outpoint, Bob's Signature**
Output 1	**Alice's balance on the channel. She can spend this output using her private key but has to wait *k* number of blocks** **or** **Bob can spend this output immediately using Alice's revocation private key if Alice cheated** **Amount: 4 BTC**
Output 2	**Bob's balance on the channel and it is immediately spendable by Bob** **Amount: 0 BTC**
Output 3	**Output for the HTLC payment and spendable by Bob if he knows the payment preimage *R*** **or** **Alice gets it back after block height *w*** **Amount: 1 BTC**

Fig. 3 – An illustration of the commitment transaction stored at Alice.

HTLCs: Hash Time Locked Contracts (HTLCs) are a core part of LN and they enable sending conditional payments. As the name suggests, an HTLC payment is hash and time locked. This means that the recipient of the payment has to redeem it within a certain period of time by revealing a secret otherwise the payment is returned to the sender. When Alice initiates a payment of 1 Bitcoin to Bob, she creates a new commitment transaction and adds an additional output to it called the *HTLC output* (Output 3 in Fig. 3). The steps taken in this process are as follows: Alice asks Bob to generate a secret called *preimage*. Bob takes a hash of the preimage and sends the hash to Alice. Alice creates an HTLC using this hash and sends the HTLC to Bob. Receiving the HTLC, Bob reveals the preimage to Alice to prove that he was the intended recipient of the payment. This finalizes the payment. The important detail here is that, if Bob does not reveal the preimage on time, 1 Bitcoin will be returned to Alice.

Revoked State Broadcast: We briefly mentioned in the commitment transaction part that, the first output of Alice's commitment transaction is conditional. If Alice broadcasts her commitment transaction, she can redeem her funds from Output 1 only after waiting k number of blocks, in other words, her funds are timelocked. This mechanism of LN is to prevent possible cheating attempts that might arise from broadcasting old commitment transactions. If Alice broadcasts a revoked (old) state, Bob can sweep Alice's funds in the channel (Output 1) by using the *revocation private key* of the respective channel state while Alice is waiting to redeem the funds for the duration of the timelock. Publishing an old state is tempting for Alice since she initially had 5 Bitcoins in the channel which is more than what she has now (i.e. 4 Bitcoins). However, this cheating attempt will result in Alice losing all her funds in the channel. Therefore, channel parties are disincentivized from cheating with this timelock mechanism in the commitment transactions.

3.3 Basis of lightning technology

Basis of Lightning Technology (BOLT) documents specify the LN's layer-2 protocol completely. Different implementations of LN follow these specifications to be compatible with each other. We propose changes to BOLT #2 and BOLT #3 to implement our protocol. These two specifications are explained below.

3.3.1 BOLT #2: Peer protocol for channel management

This protocol [26] explains how LN nodes handle channel related operations thus called the peer protocol for channel management. An LN channel has three phases which are *channel establishment*, *normal operation*, and *channel closing*. We propose modifications to this protocol to incorporate the IoT device in channel operations. High-level explanations related to each channel phase are given below.

Channel Establishment: To establish a channel, the funding node first sends an *open_channel* message to the fundee, who replies with an *accept_channel* message. Then, the funder creates the funding and the commitment transactions. It signs the fundee's commitment transaction and sends the signature to the fundee in a *funding_created* message. The fundee also sends its signature to the funder in a *funding_signed* message. Then, the funder broadcasts the funding transaction to open the channel which becomes usable after exchanging the *funding_locked* messages.

Normal Operation: Now that the channel is opened, it can be used to send/receive payments. Both nodes can offer each other HTLCs with *update_add_htlc* messages. The sender of the payment first updates the commitment transaction of the receiver, signs it, and sends the signature in a *commitment_signed* message. Receiving the signature, the receiver applies the changes and sends a *revoke_and_ack* message to the sender to revoke the old state. Finally, the sender also applies the changes to its own commitment transaction which completes the payment.

Channel Close: Both nodes can close the channel when they want. To initiate a mutual channel closing, one of the nodes sends a *shutdown* message which is replied by a *shutdown* message by the counterparty. This means no new HTLCs are going to be accepted. Then, the nodes start negotiating on a channel closing fee through *closing_signed* messages until they both agree on the same fee. Once negotiated, the mutually agreed channel closing transaction is broadcast to the Bitcoin network and the channel is closed. Nodes can also choose to close a channel by broadcasting their latest commitment transaction. However, this is not the preferred way to close a channel in LN and is only recommended to be done when the counterparty is not responding.

3.3.2 BOLT #3: Bitcoin transaction and script formats

This specification [27] explains the format of LN's Bitcoin on-chain transactions. Following this protocol is essential as it ensures that the generated signatures are valid. The transactions that are specified in this protocol are: 1) the funding transaction, 2) the commitment transactions and 3) the HTLC transactions. Before explaining them, we first describe some terminology related to Bitcoin.

SegWit: Segregated Witness (SegWit) was a Bitcoin softfork that became active in 2017. It fixed the transaction malleability problem. Before SegWit, it was possible to change the transaction ID (txid) of a transaction after it was created. Eliminating malleability also enabled deploying LN onto Bitcoin.

P2WSH: Pay-to-Witness-Script-Hash (P2WSH) is a type of Bitcoin transaction that uses SegWit. Before SegWit upgrade, Bitcoin was using Pay-to-Script-Hash (P2SH) transactions.

Witness: It is the part of a SegWit transaction that is not included when the transaction is hashed and signed. Thus, witness does not affect the txid.

Witness Script: This is the script that describes the conditions to spend a P2WSH output.

Bitcoin Opcodes: Operation codes (opcodes) specify the commands to be performed in computer languages. For Bitcoin, its scripting language has a number of opcodes that define various functions. For instance, `OP_CHECKSIG` checks whether a signature is valid.

Now, we briefly explain LN's Bitcoin transactions below as specified by BOLT #3.

Funding Transaction Output: This is a P2WSH output with a witness script: `2 <pubkey1> <pubkey2> 2 OP_CHECKMULTISIG`. It sends the funds to a 2-of-2 multisignature address which is generated from `pubkey1` and `pubkey2`. `pubkey` is also referred to as `funding_pubkey`.

Commitment Transaction Outputs: A commitment transaction currently can have up to 6 types of outputs. These outputs are:

1. `to_local` Output: This P2WSH output is for the funds of the owner of the commitment transaction and it is timelocked using `OP_CHECKSEQUENCEVERIFY`. It can also be claimed immediately by the remote node if it knows the revocation private key.
2. `to_remote` Output: This P2WPKH output is for the funds of the remote node which is immediately spendable by him/her.
3. `to_local_anchor` and `to_remote_anchor` Output: This P2WSH output was recently added to prevent cases where the commitment transaction cannot be mined by miners due to insufficient fees. With this non-timelocked output which both nodes can spend, the commitment transaction's fee can be changed for faster mining.
4. Offered HTLC Outputs: This is a P2WSH output for the HTLCs that are offered to the remote node. It either sends the funds to the HTLC-timeout transaction after the HTLC timeouts or to the remote node that can claim the funds using the payment preimage or the revocation key.
5. Received HTLC Outputs: This is a P2WSH output for the HTLCs that are received from the remote node. The remote node can claim the funds after the HTLC timeouts or by using the revocation key or; the funds are sent to an HTLC-success transaction which can be claimed by the local node with the payment preimage.

HTLC-Timeout and HTLC-Success Transactions: LN makes use of these two-stage HTLCs to protect nodes from losing any funds[2]. Both HTLC transactions are almost identical and can be spent with a valid revocation key in case of a cheating attempt.
The information provided in this section is essential to understand the modifications we propose to BOLT #3 in Section 5.4.

4. SYSTEM & THREAT MODEL

In this section, we present our system model and state the threat assumptions.

4.1 System model

There are five entities in our system which are **IoT device**, **IoT gateway**, **LN gateway**, **bridge LN node**, and **destination LN node** as shown in Fig. 4. The IoT device wants to send payments to the destination LN node for the goods/services. The IoT gateway acts as an access point connecting the IoT device to the Internet through a wireless communication standard such as WiFi, Bluetooth, 5G, etc. depending on the application. The IoT gateway provides connectivity when the IoT device is within its communication range. With this Internet connection, the IoT device communicates with the LN gateway which is running on the cloud. The LN gateway hosts the full Bitcoin and LN nodes to serve the IoT device and collects fees in return for its services. Upon a channel opening request from the IoT device, the LN gateway opens a channel to a bridge LN node that it selects from the existing LN nodes on the network. This bridge LN node is used to route the IoT device's payments to the destination LN node. While the LN gateway is free to choose any node, it should choose a highly connected node with many open channels to prevent routing failures. Our proposed system requires changes to the LN protocol. Therefore, the LN gateway and bridge LN node have to run the modified LN software.
We assume that both the IoT device and the LN gateway stay online during an LN operation such as sending a payment. The IoT device can be offline for the rest of the time.

4.2 Threat model

We make the following security related assumptions:

1. The LN gateway can post old (revoked) channel states to the blockchain. It can gather information about IoT devices and can try to steal/spend funds in the channel. However, it follows the proposed protocol specifications for communicating with the IoT device.
2. The IoT device and the LN gateway have an encrypted and authenticated communication channel between them (i.e., SSL/TLS).

[2]An explanation on why two-stage HTLCs are needed in LN can be found at `https://github.com/lnbook/lnbook/issues/187`.

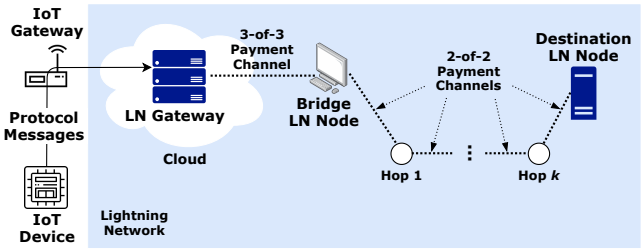


Fig. 4 – Illustration of our assumed system model.

We consider the following attacks to our proposed system:

Threat 1: Revoked State Broadcasts: The LN gateway and bridge LN node can broadcast revoked states to the blockchain to steal money from other parties.

Threat 2: Spending IoT Device's Funds: The LN gateway can spend the IoT device's funds that are committed to the channel by sending them to other LN nodes without the consent of the IoT device.

5. PROPOSED PROTOCOL DETAILS

In this section, we explain the details of the proposed channel opening, payment sending, and channel closing protocols. As mentioned in Section 3, we propose changes to LN's BOLT #2 and BOLT #3 and show these changes throughout the protocol description.

5.1 Channel opening process

As mentioned earlier, payment channels are created by the on-chain *funding transaction*. In our case, the IoT device does not have direct access to Bitcoin and LN thus cannot broadcast a funding transaction by itself to open a channel. For this reason, we propose that the IoT device securely initiates the channel opening process through the LN gateway and generates signatures whenever required. This necessitates modifying the LN's existing channel opening protocol. The main modification is to require a third signature which is going to be generated by the IoT device. Regular 2-of-2 multisignature LN channels are not suitable to accompany 3 signatures; therefore, we propose to change the 2-of-2 multisignature channels to 3-of-3 multisignature channels. With this change, an LN channel can securely be opened by involving 3 parties. This is the main novelty in our approach.
The steps of the proposed channel opening protocol are depicted in Fig. 5. We explain the protocol step by step below:

IoT Channel Opening Request: The IoT device sends an *OpenChannelRequest* message to the LN gateway to request a payment channel to be opened (#1 in Fig. 5). This message has the following ields: Type: *OpenChannelRequest, Channel Capacity. Channel Capacity* is speci ied by the IoT device and this amount of Bitcoin is taken from the IoT device's Bitcoin wallet as will be explained in the next steps.

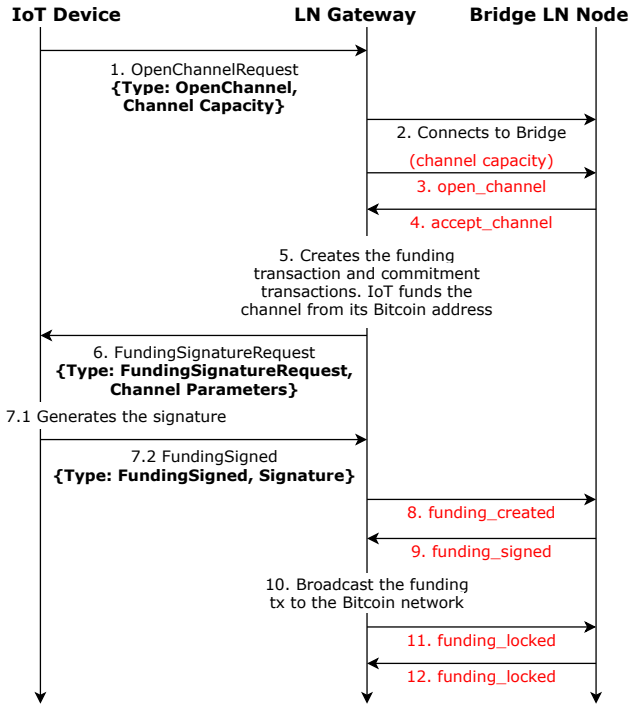


Fig. 5 – Protocol steps for opening a channel. Messages in red show the default messages in BOLT #2.

Channel Opening Initiation: The LN gateway initiates the channel opening process upon receiving the request from the IoT device. For this, it connects to a bridge LN node which it selects from the existing LN nodes on the network. Preferably, the LN gateway chooses a node with many active channels to have good chances of getting the IoT device's payments routed to the destination LN node. Upon establishing a connection, the LN gateway sends an *open_channel* message to the bridge LN node (#3 in Fig. 5). This message includes a parameter called the *funding_pubkey* which is a Bitcoin public key. Both channel parties provide their own *funding_pubkey* which are later used to construct the multisignature address of the channel. Here, we propose to add the *funding_pubkey* of the IoT device in *open_channel* message as well. Once the bridge LN node receives this message, it responds with an *accept_channel* message (#4 in Fig. 5) to acknowledge the channel opening request from the LN gateway. This message includes the *funding_pubkey* of the bridge LN node.

Creating the Funding and Commitment Transactions: Exchanging these messages locks the channel parameters and the LN gateway can now start creating the funding transaction. Using the *funding_pubkeys* of all 3 parties, the LN gateway creates a 3-of-3 multisignature address which will be used to store the channel funds. Then, the LN gateway creates a Bitcoin transaction from the IoT device's Bitcoin address to the 3-of-3 multisignature address it generated. The on-chain transaction fee for this transaction is deducted from the IoT device's Bitcoin address since it requested the channel opening. As a next step, the LN gateway creates the first versions of the commitment transactions for itself and the bridge LN node. Now, the LN gateway and the bridge LN node need to exchange signatures for the newly created commitment transactions. Specifically, the bridge LN node needs IoT device's and LN gateway's signatures. Similarly, the LN gateway needs the IoT device's and bridge LN node's signatures. The process starts with the LN gateway asking for a signature from the IoT device as explained next.

Getting Signature from the IoT Device: The LN gateway now has to request a signature from the IoT device to be sent to the bridge LN node in the *funding_created* message. For this, the IoT device needs the channel parameters that were agreed on earlier between the LN gateway and the bridge LN node. Thus, the LN gateway sends a *FundingSignatureRequest* message (#6 in Fig. 5) to the IoT device having the following fields: *Type: FundingSignatureRequest, Channel Parameters*. Having this information, the IoT device can generate the necessary signature and send it to the LN gateway in a *FundingSigned* message (#7.2 in Fig. 5) which has the following fields: *Type: FundingSigned, Signature*.

Exchanging Signatures with the Bridge LN Node: Now that the LN gateway has the IoT device's signature, it can generate its own signature as well and send these two signatures to the bridge LN node in the *funding_created* message (#8 in Fig. 5). This message also includes the funding transaction outpoint. Now, the bridge LN node is able to generate the signature for the LN gateway's commitment transaction and sends it to the LN gateway with the *funding_signed* message (#9 in Fig. 5). Note that, the LN gateway does not have the IoT device's signature for its own commitment transaction yet. In case the bridge LN node becomes unresponsive at this stage, the LN gateway can ask the IoT device for its signature to close the channel unilaterally.

Broadcasting the Funding Transaction: Now is the time for the LN gateway to broadcast the funding transaction to the Bitcoin network. After broadcasting the transaction, the LN gateway and the bridge LN node wait for the transaction to reach enough depth on the blockchain (typically 3). Once the required depth is reached, they send *funding_locked* messages (#11-12 in Fig. 5) to each other to lock the channel and begin using it.

5.2 Sending a payment

As in the case of channel opening, the introduction of the IoT device requires modifications to LN's payment sending protocol as well. We incorporate the IoT device in the process for signature generation. The details of the payment sending protocol are depicted in Fig. 6 and elaborated below:

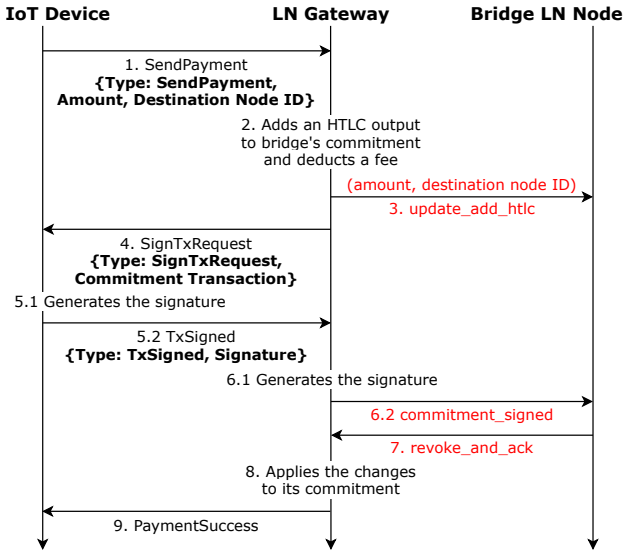


Fig. 6 – Protocol steps for sending a payment. Messages in red show the default messages in BOLT #2.

IoT Payment Sending Request: The IoT device sends a *SendPayment* message to the LN gateway to request a payment sending (#1 in Fig. 6). This message has the following fields: *Type: SendPayment, Amount, Destination Node ID*. Here, we assume that the IoT device receives the *destination node ID* in some form (i.e. QR code) from the service provider (i.e. toll company).

Payment Processing at the LN Gateway: Upon receiving the request from the IoT device, the LN gateway adds an HTLC output to the bridge LN node's commitment transaction. When preparing the HTLC, the LN gateway *deducts a certain amount of fee* from the real payment amount the IoT device wants to send to the destination LN node. Therefore, the remaining Bitcoin is sent with the HTLC. Then, to actually offer the HTLC, the LN gateway sends an *update_add_htlc* message to the bridge LN node (#3 in Fig. 6). As explained in Section 3, this HTLC can be redeemed with the payment preimage. *Destination node ID* is embedded into the *onion routing packet* in the *update_add_htlc* message. And the *amount* is also sent in this message.

Getting Signature from the IoT Device: Now, the LN gateway can apply the changes to the bridge LN node's commitment transaction and get it ready for signing. As in the case of channel opening, two signatures are needed; one from the LN gateway and one from the IoT device. Thus, we propose the LN gateway requests a signature from the IoT device for the new commitment transaction. For this purpose, the LN gateway sends a *SignTxRequest* message (#4 in Fig. 6) to the IoT device which has the following fields: *Type: SignTxRequest, Commitment Transaction*. The IoT device generates a signature for this commitment transaction and sends it to the LN gateway in a *TxSigned* message having the following fields: *Type: TxSigned, Signature* (#5.2 in Fig. 6).

Exchanging Signatures with the Bridge LN Node: Upon receiving the signature from the IoT device, the LN gateway generates its own signature as well and sends these two signatures to the bridge LN node in a *commitment_signed* message (#6.2 in Fig. 6). The bridge LN node checks the correctness of the signatures and once it verifies that the signatures are valid, it replies to the LN gateway with a *revoke_and_ack* message (#7 in Fig. 6). This message includes the commitment secret of the old commitment transaction effectively revoking the old channel state.

Payment Sending Finalization: Now, the LN gateway can also apply the changes to its own commitment transaction. To notify the IoT device of the successful payment, the LN gateway sends a *PaymentSuccess* message (#9 in Fig. 6) to the IoT device which finalizes the payment sending process.

5.3 Channel closing process

We briefly mentioned LN's channel closing mechanism in Section 3.3.1. An LN channel can be closed: 1) unilaterally when one of the channel parties broadcasts its most recent commitment transaction or 2) mutually where channel parties agree on the closing fee and create and broadcast a closing transaction. In our case, all 3 parties of the channel namely; the IoT device, the LN gateway and the bridge LN node can close the channel. We explain each case separately below:

5.3.1 IoT device channel closure

To close the channel between the LN gateway and the bridge LN node which was opened due to a request from the IoT device, the IoT device follows the proposed protocol explained below, which is also shown in Fig. 7.

IoT Channel Closing Request: The IoT device sends a *ChannelClosingRequest* message to the LN gateway.

Mutual Close: The LN gateway can close the channel unilaterally or mutually. For the mutual close case, first, it sends a *shutdown* message to the bridge LN node to initiate the closing. If there are no pending HTLCs in the channel, the bridge LN node replies with a *shutdown* message. Now, the LN gateway and the bridge LN node start negotiating on the channel closing fee. Basically, both parties offer each other a fee that they think is fair until they both agree on the same fee. Each offer is done through a *closing_signed* message which includes the offering party's signature and the offered fee amount. When the LN gateway and the bridge LN node eventually agree on a fee, the resulting closing transaction will be broadcast by the LN gateway. Since the LN gateway only has the bridge LN node's signature for this closing transaction, it needs to also get the IoT device's signature before it can broadcast it. Thus, we propose the LN gateway to send a *ClosingTxRequest* message to the IoT device which has the following fields: *Type: ClosingTxRequest, Closing Transaction*. Upon receiving the request, the IoT device generates a signature for the closing transaction and sends it to the

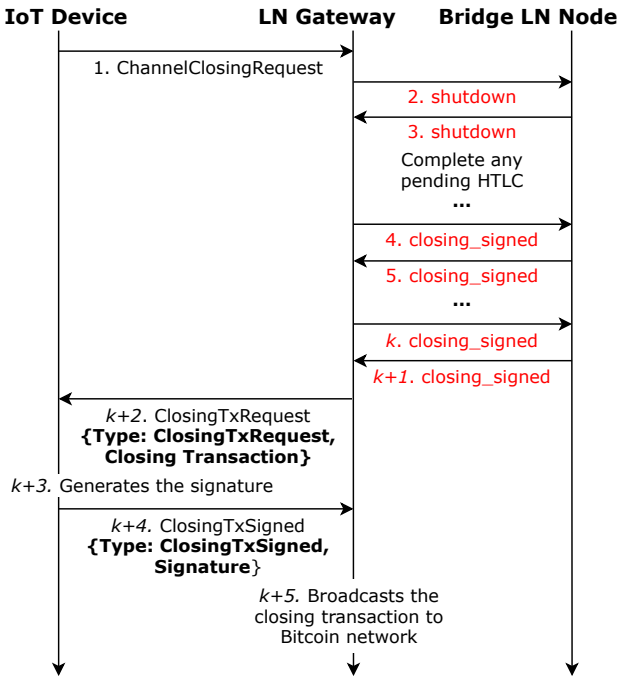


Fig. 7 – Protocol steps for the IoT device channel closure when the LN gateway performs a mutual close with the bridge LN node. Messages in red show the default messages in BOLT #2.

LN gateway in a *ClosingTxSigned* message that has the following fields: *Type: ClosingTxSigned, Signature*. With this signature, the LN gateway can now broadcast the closing transaction to the Bitcoin network and close the channel. The on-chain fee of this transaction is paid by the IoT device by deducting the fee from its balance on the channel. We illustrated the protocol steps for this mutual close case in Fig. 7.

Unilateral Close: For the unilateral close case, the LN gateway sends its most recent commitment transaction to the IoT device in a *SignTxRequest* message which has the following fields: *Type: SignTxRequest, Commitment Transaction*. Upon receiving the message, the IoT device generates a signature for this commitment transaction and sends it to the LN gateway in a *TxSigned* message that has the following fields: *Type: TxSigned, Signature*. After the LN gateway receives the signature, it can broadcast the commitment transaction to the Bitcoin network to close the channel. Similarly, the on-chain fee is paid by the IoT device.

5.3.2 LN gateway channel closure

The LN gateway can also close the channel it opened to the bridge LN node due to the IoT device's request. The steps are very similar to the IoT channel closure case with minor differences. We explain it below:

LN Gateway Channel Closing Request: The LN gateway sends a *ChannelClosingRequest* message to the IoT device.

Closing the Channel: The LN gateway can close the channel unilaterally or mutually and it needs the IoT device's signature to be able to close the channel. Therefore, it follows the exact same steps given for the IoT channel closure case above. The only difference is the on-chain fee part. This time, channel closing is requested by the LN gateway; therefore, it pays the on-chain fee. The same mechanism is used: the on-chain fee is deducted from the LN gateway's balance on the channel. There is a chance that the LN gateway might not have enough balance in the channel to cover the on-chain fee. Thus, we propose that the LN gateway does not attempt to close the channel without collecting enough service fees on the channel.

5.3.3 Bridge LN node channel closure

Similar to the LN gateway channel closure case, the bridge LN node can close the channel unilaterally or mutually. In both scenarios, the LN gateway will learn about closure of the channel. We propose that the LN gateway notifies the IoT device about the channel closure by sending it a *ChannelClosed* message. This serves as a notification to the IoT device so that it does not attempt to use the channel in the future.

5.4 Changes to LN's BOLT #3

We explained in Section 3.3.2 the BOLT #3 specification. Since the channels were made 3-of-3 multisignature with the introduction of the IoT device, it requires changes to the LN's Bitcoin scripts. In this section, we show the proposed changes to the funding transaction output, the commitment transactions, and the HTLC transactions.

Changes to the Funding Transaction Output: Instead of sending the funds to a 2-of-2 multisignature address, we propose to send them to a 3-of-3 multisignature address. Thus, the new proposed witness script of the funding transaction output is: `3 <pubkey1> <pubkey2> <pubkey3> 3 OP_CHECKMULTISIG`.

Changes to the Commitment Transaction Inputs: We propose to change the commitment transaction input witness to: `0 <sig_for_pubkey1> <sig_for_pubkey2> <sig_for_pubkey3>` as now 3 signatures are needed instead of 2.

Addition of `to_IoT` Output to the Commitment Transactions: The LN gateway is not funding the channel. Instead, the channel is funded by the IoT device. Therefore, the IoT device needs its own output in commitment transactions. Thus, we propose adding a `to_IoT` output to the LN gateway's and bridge LN node's commitment transactions. This output pays to the `IoT_pubkey` that the IoT device can spend with the witness `<IoT_sig>`.

Changes to `to_local` Output of the Commitment Transactions: Since the channel is funded by the IoT device, the `to_local` output in the LN gateway's commitment transaction only holds the LN gateway's service fees which are sent to the LN gateway's `<local_delayedpubkey>`. The bridge LN node's `to_local` output is not modified.

Changes to to_remote Output of the Commitment Transactions: There are no changes to the `to_remote` outputs. The LN gateway's `to_remote` output pays to the bridge LN node's `remotepubkey`, the bridge LN node's `to_remote` output pays to the LN gateway's `remotepubkey`.

Changes to Offered HTLC Outputs of the Commitment Transactions: The witness script of this output normally has `OP_DROP 2 OP_SWAP <local_htlcpubkey> 2 OP_CHECKMULTISIG` which sends the funds to the local node with the HTLC-timeout transaction. For the LN gateway's offered HTLC outputs, the local node is the IoT device instead of the LN gateway itself, thus `local_htlcpubkey` is changed to `IoT_htlcpubkey`. Our protocol does not support offered HTLC outputs for the bridge LN node's commitment transaction. This is because our protocol only supports payments from the IoT device to destination LN nodes. This is a limitation which we plan to address in the future.

Changes to Received HTLC Outputs of the Commitment Transactions: The current design does not support received HTLC outputs for the LN gateway's commitment transaction as the IoT device cannot receive payments on the channel. On the other hand, the bridge LN node's commitment transaction supports received HTLC outputs and we do not propose any changes.

Changes to HTLC-Timeout and HTLC-Success Transactions: As explained above, we do not support HTLC-success transactions for the LN gateway's commitment transaction as the IoT device cannot receive payment on the channel. Similarly, HTLC-timeout transactions are not supported for the bridge LN node's commitment transaction. For the HTLC-timeout transactions in the LN gateway's commitment transaction, we propose having 3 signatures instead of 2. Thus, the new transaction input witness will be: `0 <remotehtlcsig> <localhtlcsig> <IoThtlcsig> <>`. For the HTLC-success transactions in the bridge LN node's commitment transaction, we again propose having 3 signatures instead of 2. The new transaction input witness will be: `0 <remotehtlcsig> <IoThtlcsig> <localhtlcsig> <payment_preimage>`. Additionally, the `local_delayedpubkey` in the witness script for the output of the HTLC-timeout transaction in the LN gateway's commitment transaction is changed to `IoT_delayedpubkey`.

5.5 Handling revoked state broadcasts

We briefly mentioned in Section 3.2 that Alice and Bob can attempt to cheat by broadcasting revoked channel states to the blockchain. LN uses *timelocks* to address this issue. The idea is not to let the broadcasting party spend its funds immediately while letting the counterparty do so. Since we proposed changes to the LN protocol, it requires revisiting the revoked state cases. Specifically, the IoT device does not store any commitment transactions nor revocation keys. It is only involved in signing operations. Therefore, this situation should not result in any loss of funds for the IoT device in possible cheating attempts by other channel parties. We examine the possible revoked state broadcast cases by the LN gateway and bridge LN node separately below and show how both cases are handled properly.

5.5.1 Revoked state broadcast by the LN gateway

The LN gateway can broadcast revoked commitment transactions to the blockchain since they are already signed by everyone. This attempt has 2 possible outcomes: 1) The bridge LN node was offline for long enough to not realize the LN gateway was cheating. Thus, it loses some or all of its funds in the channel depending on the broadcast old state. 2) The bridge LN node was online during the LN gateway's cheating attempt thus, sweeps all the funds in the channel using the revocation private key of the respective old state.

The first scenario is the famous *being offline* issue for the LN nodes. All existing LN nodes are vulnerable to this attack when they are offline for extended periods of time [28]. Therefore, it is not specific to our protocol and addressing it is beyond the scope of this paper[3].

However, the second scenario jeopardizes the IoT device's funds if not addressed. To handle this case, we propose two modifications to the LN gateway's commitment transaction. The first modification is for the `to_IoT` output at which the IoT device's funds are held. We propose that even after a failed cheating attempt by the LN gateway, the bridge LN node cannot spend the `to_IoT` output. In other words, we propose to make this output spendable only by the IoT device at all times. In this way, the IoT device's funds in the channel will be protected. The second modification is proposed for the fee output (`to_local`) of the LN gateway. Normally, this output is spendable by the LN gateway only. We propose to turn it into a conditional output such that, if the LN gateway gets caught while cheating, the bridge LN node can spend this output using the revocation private key. In other words, the LN gateway will lose the fees it collected on the channel if it gets caught while cheating. This modification disincentivizes the LN gateway from attempting to cheat therefore addresses the revoked state broadcast issue. Consequently, with these 2 modifications, the IoT device's funds are protected and the LN gateway is disincentivized from broadcasting revoked states. We illustrate the LN gateway's modified commitment transaction in Fig. 8.

5.5.2 Revoked state broadcast by the bridge LN node

Similar to the LN gateway, the bridge LN node can also broadcast its revoked commitment transactions to the blockchain in an attempt to cheat. Depending on the

[3]Watchtowers [29] were proposed to protect LN nodes against this threat.

Commitment Tx (Held by Gateway)

Input	**Funding Transaction Outpoint, Bridge's Signature**
to_IoT	**IoT's balance on the channel and it is immediately spendable by IoT** **Amount: 4 BTC**
to_remote	**Bridge's balance on the channel and it is immediately spendable by bridge** **Amount: 0 BTC**
Offered HTLC	**- Output for the HTLC payment and spendable by the bridge if it knows the payment preimage *R*** **or** **- IoT gets it back after block height *w*** **Amount: 0.9 BTC**
to_local	**- Gateway's fees on the channel. It can spend this output using its private key but have to wait *k* number of blocks** **or** **- Bridge can spend this output immediately using its revocation private key if the gateway cheated** **Amount: 0.1 BTC**

Fig. 8 – An illustration of the commitment transaction stored at the LN gateway with the IoT output and the fee output modified (`to_IoT` and `to_local`, respectively). This commitment transaction is generated after the following events: A channel with 5 BTC capacity is opened; the IoT device initiated a 1 BTC payment to a destination; the LN gateway charged the IoT device a service fee of 0.1 BTC.

published old state, it might benefit the bridge LN node. However, this attempt will be successful only if the LN gateway is offline during the attempt. This brings us back to the *being offline* issue. Basically, it is the LN gateway's responsibility to stay online and protect itself against this attack. As this is a general LN issue, it is beyond the scope of this paper.

6. SECURITY ANALYSIS

In this section, we present how the attacks mentioned in Section 4.2 are mitigated in our approach.

Threat 1: Revoked State Broadcasts: For the attack where the LN gateway broadcasts a revoked state, our approach proposed punishing the LN gateway. Basically, the LN gateway loses the service fees it collected on the channel to bridge the LN node. With our punishment addition, the LN gateway is disincentivized from broadcasting a revoked state. Additionally, we proposed a protection mechanism for the IoT device's funds on the channel. In this way, even when the bridge LN node catches the LN gateway while cheating, the IoT device does not lose any funds. The other cases where the parties might lose funds because of being offline are not handled as they are general LN issues that are not related to our protocol.

Threat 2: Spending IoT Device's Funds: Using 3-of-3 multisignature channels secure the IoT device's funds in the channel since the LN gateway cannot spend them without getting the IoT device's cryptographic signature first. As shown in Fig. 6, the LN gateway sends the newly generated bridge LN node's version of the commitment transaction to the IoT device for signing in step 4. Without performing this step, the LN gateway cannot get a signature from the bridge LN node for its own version of the commitment transaction. Therefore, it is apparent that the LN gateway will not be able to complete a successful payment without getting a signature from the IoT device. On the other hand, if we were to use LN's original 2-of-2 multisignature channels in our system, the LN gateway could move the funds without needing a signature from the IoT device. Because, with a 2-of-2 multisignature channel, the LN gateway could send its own signature to the bridge LN node which would be enough for the bridge LN node to be able to spend its commitment transaction. Since the LN gateway cannot spend the IoT device's funds in the channel at its own will, the IoT device's funds are always protected and can be only spent when the IoT device provides its signature. Consequently, the usage of 3-of-3 multisignature channels protects the IoT device's funds from getting unwillingly spent by the LN gateway.

7. EVALUATION

In this section, we explain our experiment setup and present the performance results.

7.1 Experiment setup and metrics

To evaluate the proposed protocol, we created a setup where an IoT device connects to an LN gateway to send payments on the LN. To mimic the IoT device, we used a Raspberry Pi 3 Model B v1.2 and the LN gateway was set up on a desktop computer with an Intel(R Xeon(R CPU E5-2630 v4 and 32 GB of RAM. This desktop computer was in a remote location different from that of Pi's. For the full Bitcoin node installation, we used *bitcoind* [30] which is one of the implementations of the Bitcoin protocol. For the LN node, we used *lnd v0.11.0-beta.rc1* from Lightning Labs [20] which is a complete implementation of the LN protocol. Python was used to implement the protocol.
We used IEEE 802.11n (WiFi and Bluetooth Low Energy (BLE 4.0 to exchange protocol messages between the Raspberry Pi and the LN gateway. In the WiFi scenario, we created a server & client TCP socket application in Python. With this, Raspberry Pi and the remote desktop computer communicated with each other. Raspberry Pi was connected to the Internet through a regular Internet modem which acted as the IoT gateway. For Bluetooth experiments, we first paired the Raspberry Pi and the Bluetooth adapter of the IoT gateway. We used a laptop computer as the IoT gateway which had a Bluetooth adapter. Using Python's *bluetooth* library, Raspberry Pi and the laptop computer communicated with each other. The laptop computer was programmed to talk to the LN gateway over the same TCP socket application that was set up. In both cases, the LN gateway used gRPC API [31] of *lnd* to communicate with the LN node that was running on it.
To assess the performance of our protocol, we used the following metrics: 1) *Time* which refers to the total

computational and communication delays of the proposed protocol; and 2) *Cost* which refers to the total monetary cost associated with sending payments using the LN gateway.

To compare our approach to a baseline, we considered the case where the LN gateway sends the payments to a destination by itself. In other words, no IoT device is present, and all LN tasks are solely performed by the LN gateway.

7.2 Computational and communication delays

We first assessed the computational and communication delays of our proposed protocol. The computational delay of running the protocol on a Raspberry Pi comes from the AES encryption of the protocol messages and HMAC calculations. We used Python's *pycrypto* library to encrypt the protocol messages with AES-256 encryption. The encrypted data size for the messages was 24 bytes. For the HMAC calculations, we used *hmac module* in Python. The delays for these operations are shown in Table 1. As can be seen, they are negligible.

Table 1 – Computational delays on the IoT device

AES Encryption	HMAC Calculation	Total
15 ms	$<$ 1 ms	15 ms

We then measured the communication delays which are used in other experiments to evaluate the timeliness of the protocol. We define the communication delay as the delay of sending a protocol message from the Raspberry Pi to the IoT gateway and receiving an acknowledgment for that message from the IoT gateway or vice versa. This delay is also called the round-trip time of a protocol message. When WiFi was used for the connection, the measured round-trip delay of a message was approximately *9 ms*. When Bluetooth was used, the same delay was *0.8 seconds*. As expected, the message exchanges with Bluetooth are much slower compared to WiFi which is related to the bandwidth difference between the two technologies.

7.3 Toll payment use case evaluation of the protocol

In this part, we consider an example case where real-time response is critical. We assume a toll application where cars pass through a toll gate and pay the toll without stopping. For this, wireless technologies are used. The cars that enter the communication range of the toll gate's wireless system immediately initiates a payment to the toll company's LN node through the toll's LN gateway which is running on the cloud. Cars are notified upon a successful payment. In order for this process to work, payment sending has to be completed while the car is still in the communication range of the toll gate's wireless system.

As can be seen in Fig. 6, there are 4 protocol message exchanges between the IoT device and the LN gateway in our payment sending protocol. For this toll example, in each protocol message exchange, there are 2 corresponding communication delays which are between the car and the IoT gateway and between the IoT gateway and the LN gateway running on the cloud. We know the communication delays between the car and the IoT gateway from the previous section which were 9 ms and 0.8 seconds for WiFi and Bluetooth respectively. The delay of the communication between the IoT gateway and the LN gateway running on the cloud on the other hand will be the delay of a regular TCP communication between two computers. We measured an average of 123 ms delay for this communication. The actual LN payment on the other hand was sent in 2.1 seconds which is an average value calculated from 30 separate payments sent at different times throughout the day. There is also the 15 ms computational delay at the car for each protocol message it generates. Eventually, the *total payment sending time for WiFi was 2.658 seconds while BLE had 5.822 seconds*.

We mentioned earlier that 802.11n and BLE were used for the measurements. The advertised range of 802.11n is approximately 250 meters [32] and the advertised range of BLE is around 220 meters [33]. If cars pass through the toll gate with a speed of 50 miles per hour, there is around 11 seconds with WiFi and 10 seconds with Bluetooth available for them to complete the protocol message exchanges with the LN gateway for a successful toll payment. The results for varying vehicle speeds are shown in Table 2. As can be seen, for both WiFi and Bluetooth cases, our protocol meets the deadlines even under a high speed of 80 mph. Even in the case of Bluetooth where the data rates are low, the deadline can be met due to the long communication ranges of Bluetooth 4.0. If different technologies with limited ranges are used, cars' speed should be enforced accordingly.

Table 2 – Available time under different speeds to make a successful toll payment with WiFi and Bluetooth

	WiFi		Bluetooth	
Vehicle Speed	**Available Time**	**Satisfied?**	**Available Time**	**Satisfied?**
50 mph	11.2 s	Yes	9.8 s	Yes
60 mph	9.3 s	Yes	8.2 s	Yes
80 mph	7 s	Yes	6.2 s	Yes

7.4 Coffee shop example

As another real-life example, let us consider a case where customers pay for coffee at a coffee shop with their smartwatches using our protocol. This payment is less time-critical compared to the toll payments, since the customers can wait by the cashier until they get the payment confirmation from the coffee shop's LN node. We already measured the total payment sending time for WiFi and Bluetooth cases which were 2.658 seconds and 5.822 seconds, respectively. Therefore, again in this example, the use of our protocol through the WiFi and Bluetooth wireless technologies is feasible since the customers can wait for more than 5.822 seconds for payment confirmations.

As an alternative, customers can also pay for the coffee using an existing device in the coffee shop that is connected to the LN and ready to send payments using its existing LN channels. In this case, however, there is no IoT device involved; therefore, no wireless communication delays.

We call this option *No IoT Case*. Since there are no communication delays in this scenario, payment sending only takes 2.1 seconds. When compared to the case where the coffee is paid with a smartwatch, the difference in the payment sending time is the communication delays for the protocol message exchanges. The results of this coffee shop example are summarized in Table 3.

Table 3 – Total payment sending time comparison of all three cases for the coffee shop example

Our Approach - WiFi	Our Approach - BLE	No IoT Case
2.66 seconds	5.82 seconds	2.1 seconds

7.5 Cost analysis

We now investigate the total monthly payment sending cost of the IoT device. The only associated cost of the payment sending comes from the fees the LN gateway charges when it sends payments for the IoT device. We assume that the fee that will be charged totally depends on the LN gateway and specific use case of the service (i.e., paying for toll, paying for coffee, etc.). For the toll example, let us assume that a car passes through the toll 2 times a day. If the toll charges $1.5 per pass, the car pays $3 a day. The LN gateway also charges a *%k* fee on top of the toll. If we take k=10, then the car pays $3.3 in total, $0.3 of which goes to the LN gateway. The LN gateway's fee includes the LN's payment routing fees which are usually around a few satoshi per payment [34]. Then in one month, the car will pay $9 to the LN gateway for the fees. While it depends on the driver, we believe that the reflected cost is negligible considering the comfort of the fast toll payments.

8. CONCLUSION

In this paper, we proposed a secure and efficient protocol for enabling IoT devices to use Bitcoin's LN for sending payments. By modifying LN's existing peer protocol and on-chain Bitcoin transactions, a third peer (i.e. IoT device) was added to the LN channels. The purpose was to enable resource-constrained IoT devices that normally cannot interact with LN to interact with it and perform micro-payments with other users. The IoT device's interactions with LN are achieved through a gateway node that has access to LN and thus can provide LN services to it in return for a fee. In order to prevent possible threats that might arise from broadcasting old states, LN's commitment transactions were modified. Our evaluation results showed that the proposed protocol enables LN payments for the IoT devices with negligible delays.

REFERENCES

[1] Shancang Li, Li Da Xu, and Shanshan Zhao. The internet of things: a survey. *Information Systems Frontiers*, 17(2):243–259, 2015.

[2] Dražen Pašali , Branimir Cviji , Dušanka Bundalo, Zlatko Bundalo, and Radovan Stojanovi . Vehicle toll payment system based on internet of things concept. In 2016 5th Mediterranean Conference on Embedded Computing (MECO), pages 485–488. IEEE, 2016.

[3] Suat Mercan, Ahmet Kurt, Enes Erdin, and Kemal Akkaya. Cryptocurrency solutions to enable micropayments in consumer IoT. *IEEE Consumer Electronics Magazine*, 2021.

[4] Ahmet Kurt, Enes Erdin, Mumin Cebe, Kemal Akkaya, and A Selcuk Uluagac. LNBot: A covert hybrid botnet on bitcoin lightning network for fun and profit. In *European Symposium on Research in Computer Security*, pages 734–755. Springer, 2020.

[5] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Technical report, 2008. `https://bitcoin.org/bitcoin.pdf`.

[6] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. Technical report, 2014. `https://github.com/ethereum/yellowpaper`.

[7] Qiheng Zhou, Huawei Huang, Zibin Zheng, and Jing Bian. Solutions to scalability of blockchain: A survey. *IEEE Access*, 8:16440–16455, 2020.

[8] Christian Decker and Roger Wattenhofer. A fast and scalable payment network with bitcoin duplex micropayment channels. In *Symposium on Self-Stabilizing Systems*, pages 3–18. Springer, 2015.

[9] Joseph Poon and Thaddeus Dryja. The bitcoin lightning network: Scalable off-chain instant payments. Technical report, 2016. `http://lightning.network/lightning-network-paper.pdf`.

[10] Kenichi Kurimoto. Lightning network x IoT(LoT); potential, challenges and solutions, accessed 2021-03. `https://medium.com/nayuta-en/lightning-network-x-iot-lot-potential-challenges-and-solutions-6e4d8b4c252a`.

[11] Christopher Hannon and Dong Jin. Bitcoin payment-channels for resource limited IoT devices. In *Proceedings of the International Conference on Omni-Layer Intelligent Systems*, pages 50–57, 2019.

[12] Jérémy Robert, Sylvain Kubler, and Sankalp Ghatpande. Enhanced lightning network (off-chain)-based micropayment in IoT ecosystems. *Future Generation Computer Systems*, 2020.

[13] Arman Pouraghily and Tilman Wolf. A lightweight payment verification protocol for blockchain transactions on IoT devices. In *2019 International Conference on Computing, Networking and Communications (ICNC)*, pages 617–623. IEEE, 2019.

[14] Brainbot Labs. μraiden - a payment channel framework for fast & free off-chain ERC20 token transfers, accessed 2021-03. `https://raiden.network/micro.html`.

[15] Christos Profentzas, Magnus Almgren, and Olaf Landsiedel. TinyEVM: Off-chain smart contracts on low-power IoT devices. In *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, pages 507–518. IEEE, 2020.

[16] Dunfeng Li, Yong Feng, Yao Xiao, Mingjing Tang, and Xiaodong Fu. A data trading scheme based on payment channel network for internet of things. In *International Conference on Blockchain and Trustworthy Systems*, pages 319–332. Springer, 2020.

[17] Nachiket Tapas, Yechiav Yitzchak, Francesco Longo, Antonio Puliafito, and Asaf Shabtai. P4UIoT: Pay-per-piece patch update delivery for IoT using gradual release. *Sensors*, 20(7):2156, 2020.

[18] Lightning Labs. Neutrino: Privacy-preserving bitcoin light client, accessed 2021-03. https://github.com/lightninglabs/neutrino.

[19] Breez. Breez mobile client, accessed 2021-03. https://github.com/breez/breezmobile.

[20] Lightning Labs. Lightning network daemon, accessed 2021-03. https://github.com/lightningnetwork/lnd.

[21] Ahmet Kurt, Suat Mercan, Enes Erdin, and Kemal Akkaya. Enabling micro-payments on IoT devices using bitcoin lightning network. In *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 1–3. IEEE, 2021.

[22] Lightning Labs. Announcing our first lightning mainnet release, lnd 0.4-beta!, accessed 2021-03. https://blog.lightning.engineering/announcement/2018/03/15/lnd-beta.html.

[23] Bitcoin Wiki. Contract, accessed 2021-03. https://en.bitcoin.it/wiki/Contract.

[24] 1ml.com. Lightning network search and analysis engine, accessed 2021-03. https://1ml.com/.

[25] Bryan Vu. Exploring lightning network routing, accessed 2021-03. https://blog.lightning.engineering/posts/2018/05/30/routing.html.

[26] Lightning Labs. BOLT #2: Peer protocol for channel management, accessed 2021-03. https://github.com/lightningnetwork/lightning-rfc/blob/master/02-peer-protocol.md.

[27] Lightning Labs. BOLT #3: Bitcoin transaction and script formats, accessed 2021-03. https://github.com/lightningnetwork/lightning-rfc/blob/master/03-transactions.md.

[28] Ichiro Kuwahara. Operating lightning #0001 — basic challenges in operating lighting network nodes, accessed 2021-03. https://medium.com/crypto-garage/operating-lightning-0001-basic-challenges-in-operating-lighting-network-nodes-d04386ac931a.

[29] lnd. Private altruist watchtowers, accessed 2021-03. https://github.com/lightningnetwork/lnd/blob/master/docs/watchtower.md.

[30] bitcoin.org. Running a full node, accessed 2021-03. https://bitcoin.org/en/full-node.

[31] Lightning Labs. LND gRPC API reference, accessed 2021-03. https://api.lightning.community/.

[32] Akira Matsumoto, Kouichi Yoshimura, Stefan Aust, Tetsuya Ito, and Yoshihisa Kondo. Performance evaluation of IEEE 802.11n devices for vehicular networks. In *2009 IEEE 34th Conference on Local Computer Networks*, pages 669–670. IEEE, 2009.

[33] Heikki Karvonen, Carlos Pomalaza-Ráez, Konstantin Mikhaylov, Matti Hämäläinen, and Jari Iinatti. Experimental performance evaluation of BLE 4 versus BLE 5 in indoors and outdoors scenarios. In *Advances in Body Area Networks I*, pages 235–251, 2019.

[34] BitMEX Research. The lightning network (part 2) – routing fee economics, accessed 2021-03. https://blog.bitmex.com/the-lightning-network-part-2-routing-fee-economics/.

AUTHORS

Ahmet Kurt received two B.S. degrees from Antalya Bilim University, Antalya, Turkey in 2018. He is currently pursuing a Ph.D. degree in Electrical and Computer Engineering with the Florida International University, Miami, United States. His current research interests include Bitcoin's lightning network, Bitcoin and payment channel networks.

Suat Mercan is a postdoctoral researcher at Florida International University. He received his Ph.D. degree in Computer Science from the University of Nevada, Reno in 2011 and his M.S degree in Electrical and Computer Engineering from the University of South Alabama in 2007. His main research interests are blockchain, payment channel and peer-to-peer networks, cybersecurity, digital forensics, and content delivery.

Enes Erdin is an Assistant Professor at University of Central Arkansas. He received his Ph.D. degree in Electrical and Computer Engineering from Florida International University and he was an NSF CyberCorps Fellow. He conducts research in the areas of hardware security, blockchain technology, and cyber-physical systems.

Kemal Akkaya is a professor in the Department of Electrical and Computer Engineering at Florida International University. He leads the Advanced Wireless and Security Lab and is an area editor of the Elsevier Ad Hoc Networks Journal. His current research interests include security and privacy, and protocol design. He has published over 120 papers in peer-reviewed journals and conferences. He received the "Top Cited" article award from Elsevier in 2010.

SIOT FOR COGNITIVE LOGISTICS: LEVERAGING THE SOCIAL GRAPH OF DIGITAL TWINS FOR EFFECTIVE OPERATIONS ON REAL-TIME EVENTS

Miha Cimperman[1], Angela Dimitriou[2], Kostas Kalaboukas[3], Aziz S. Mousas[4], Salvatore Quattropani[5]
[1]Institute Jozef Stefan, Ljubljana, Slovenia , [2]Intrasoft International SA, Luxemburg, [3]Technical University of Crete, Greece, [4]SingularLogic SA, Greece, [5]CNIT - RU at the University of Catania, Italy

NOTE: Corresponding author: Salvatore Quattropani, Salvatore.quattropani@cnit.it

Abstract *– Over the years, with the migration of organizations towards the concepts of logistics 4.0, a paradigm shift was necessary to guarantee logistics efficiency. The challenge is to dynamically cope in real time with vast number of shipments and destinations, which need to be realigned both with a determined lead time and with a finite of available resources. Although a number of standards have already been adopted for the management of transport and logistics operations, taking advantage, for instance, of Decision Support Systems and Geographic Information Systems, new models are required for achieving effective handling of the dynamic logistics environment that is shaped today. In this paper, an integrated logistics framework addressing the previous challenges is presented, for the first time, as a result of the activities of the H2020 COG-LO project. This novel approach exploits Social Internet of Things (SIoT) and the digital twins technique to realize the concept of the Cognitive Logistics Object (CLO). A CLO is defined as an entity that is augmented with cognitive capabilities, it is autonomous, and bears social-like capabilities, which enable the formulation of ad hoc communities for negotiating optimal solutions in logistics operations.*

Keywords – Cognitive logistics, collaborative logistics, digital twins, Social Internet of Things

1. INTRODUCTION

The development of IoT and cyber-physical systems [1] technologies lay the foundations for the evolution of logistics 4.0, in which the way of organizing supply and production changed drastically. The digitization process pushes logistics operators and all interested parties to turn towards an approach that embraces all the technological innovations that the market offers. The aim is to reach the economic objectives faster and to improve logistics services' quality. The new emerging logistics scenario requires rapid information processing with a high level of security. The spread of big data, the expansion of the logistics chain, the problem of route optimization, the localization of resources, and the maximization of the load factor have become the main focus of research and development in the logistics sector. The challenges listed do not only require the introduction of new logistics concepts, but also need a significant effort to go beyond the common vision of the elements participating in the logistics chain. The COG-LO project [2] addresses these challenges by implementing innovative tools that enable logistics 4.0 [3]. Logistics entities become both cognitive and collaborative. Their level of interoperability increases significantly thanks to the digitization of all the actors participating in the supply chain (cargo, vehicle, warehouse, parking slot, other transport modes, systems, etc.). This is achieved by exploiting an ad hoc dynamic social network based on the Social Internet of Things (SIoT) paradigm [4]. SIoT enables these actors to interact and negotiate potential alternative solutions to emerging logistics requirements, taking into account their current status, needs and identified limitations. The proposed solution implements dynamic optimization strategies streamlining the burden of the decision-making process, enhancing the robustness of artificial intelligence systems, and thus allowing an increasingly pervasive approach of these techniques within the logistics 4.0 world.

The document is organized as follows: In Section 2, the necessary background information is provided. In Section 3, the COG-LO framework is introduced. In Section 4, the system's performance is evaluated. Finally, Section 5 concludes with some final remarks.

2. DIGITAL LOGISTICS INFRASTRUCTURE

In recent years, the issues of digital transformation of transport infrastructures have been of particular importance in the context of Industry 4.0. Numerous real-time planning algorithms have been developed by the logistics community over the last thirty years; these include Decision Support Systems (DSS) [5] and Geographic Information Systems (GIS), i.e. a group of procedures that provide input, data storage and retrieval, mapping and spatial analysis [6], and tools to support the organization's decision-making activities. In the past decades, several efforts have been made to integrate GIS with DSS, promoting the concept of collaborative GIS. The GS1 [7] system of standards is currently the definitive framework widely recognized in the field of logistics. It ensures a high degree of interoperability between stakeholders and provides a standard way to identify objects and locations.

Despite the enormous bene its of the aforementioned systems, there are still many limitations. Most dynamic routing algorithms are unable to analyze and distinguish the nature of external events affecting the supply chain, rendering therefore these systems unable to perform periodic re-optimizations [8]. In consequence, new models are needed that move away from the concepts on which traditional (functional) logistics are based.

2.1 Digital twin – Cognitive Logistic Object

The concept of Digital Twins (DT) has been introduced by M. Grieves [9] as a digital representation of a physical entity, for which a specific behaviour can be modelled. After performing appropriate simulations and calculations on this behaviour an action may be triggered on that entity. In the basis of the DTs, various scenarios have been introduced at the manufacturing, logistics sector [10], health and other sectors [11]. As the DT becomes a strong technological trend [12], with companies tending to invest on digital transformation, the transition to a DT modelling approach in logistics can offer new collaborative models and optimization procedures in the domain. The application of the DT paradigm in logistics has been introduced by the concept of a Cognitive Logistics Object [13]. A Cognitive Logistics Object (CLO) is a virtualized object (similar to a DT) or system that participates in the logistics process. It exhibits properties like autonomy, context awareness, responsiveness and learning ability. The CLO represents different actors such as cargo, truck, traffic light, supporting system, etc., with each one having different capabilities. In particular, a CLO is an autonomous object, reactive to changes in the environment and its context. It is able to learn, collaborate, decide on next actions, create social networks and solve local problems. Thanks to the virtualization of objects, communication between heterogeneous systems becomes possible, and each CLO action takes into account various variables such as business priorities, environmental conditions, traffic conditions, load information etc.. Furthermore, each virtualized entity implements the functionalities required for managing the entity's communications. A CLO exhibits social behaviour, which means that the digital counterpart of the logistics object implements a series of services offered by SIoT to establish relationships with other CLOs, dynamically exchange information and thus optimize logistics operations. The virtualization-based approach is quite common in the IoT domain [14] as it promotes interoperability and extends an object's physical and digital characteristics.

2.2 Social Internet of Things

The SIoT paradigm brings social network concepts to the IoT context. According to this paradigm, each object is characterized by a social behavior and therefore its digital twin is capable of creating social relations on the basis of common elements and affinity. In the resulting social network, any object looks for desired services by using its relationships, querying its friends and the friends of its friends in a distributed manner. This procedure guarantees an efficient and scalable discovery of CLOs and services following the same principles that characterize the social networks for humans. The following types of relationships are indicative:

- Ownership Object Relationship (OOR): created between objects that belong to the same owner.
- Co-location Object Relationship (CLOR): created between stationary devices located in the same place (also called Co-Geolocation CGLOR).
- Parental Object Relationship (POR): created between objects of the same model, producer and production batch.
- Co-Work Object Relationship (CWOR): created between objects that meet each other at the owner's workplace (e.g., two trucks parked at a depot).
- Social Object Relationship (SOR): created as a consequence of frequent meetings between objects.
- Transactional Object Relationship (TOR): established between devices that interact with each other frequently [15].
- Time Plan Object Relationship(TPOR): created between CLOs that have coincident or overlapping schedules.

The social graph generated by the SIoT in the COG-LO context is an undirected graph of CLOs connected with the aforementioned relationships. It is similar to a social graph generated by friendships between humans with common characteristics. The navigability problem of a social network has been widely addressed by Milgram, and the small world phenomenon [16] has been at the centre of social science research for decades. According to Milgram's hypothesis, even if a social graph is very large and two nodes are very distant from each other, it is possible, starting from one, to reach the other by surfing the net in less than 6 hops, thanks to the existence of short paths between pairs of nodes. The SIoT is responsible for establishing relationships on the basis of local information, therefore creating the necessary conditions for social navigation of the CLOs' graph. To achieve this, advanced storage techniques are adopted to facilitate navigability and ensure that the connections between the nodes of a graph on a logical level are equally accessible on the data plane [17]. To meet this challenge, SIoT exploits the metadata of the social nodes to efficiently index every single data, connection or path in the graph. SIoT can support an ML-based optimizer capable of pruning the social graph in order to generate a smaller subgraph, which represents some elements of Milgram's small world, where nodes have high correlation based on their local information. According to the principles explained above, each logistics object is associated with a CLO, which is the digital

representation of the physical entity that implements the functionality required for managing the communications and for supporting the common control plane [17]. A CLO carries metadata associated with a given object, such as information on object's nature, status and list of friends. Among the information characterizing the CLO, semantic descriptions facilitate the interactions between digital twins, supporting the detection and management operations despite devices heterogeneity. A CLO, as a software entity, can be implemented in SIoT repositories and hosted in the cloud or on the edge of the network infrastructure. SIoT is responsible for managing the life cycle of each digital twin, store and update their status information in real time and disseminating data from the digital twin to the physical world through a new data communication delivery method and scheme called Sociocast [18]. In particular, Sociocast leverages the SIoT concept to support group communications among nodes in an efficient and effective way on the control plane. With data driven logistics, dynamic optimization of basic logistics processes is at the forefront of the next generation of logistics services. Finding optimal routes for vehicles is a problem that has been studied for many decades from a theoretical and practical point of view. What is typically associated with the Vehicle Routing Problem (VRP) is a generalization of the Travelling Salesperson Problem, where multiple vehicles are available. This class of routing problems is notoriously hard; it not only falls into the class of NP-complete problems, but also it cannot be solved optimally in practice, even for moderate instance sizes. More importantly, processing VRP optimization on large graphs in real-time demands employing additional techniques, such as heuristics and/or graph pruning. Different clustering approaches have been used in pruning the input graph for VRP. For example, Ruhan et al. [19] uses k-means clustering in combination with a rebalancing algorithm to obtain areas with balanced numbers of customers. Bent et al. also study the benefits and limitations of vehicle and customer-based decomposition schemes [20], demonstrating better performance with the latter. In COG-LO, linear optimization was applied as an exact optimization approach for solving VRP. The combination of an enriched social-like behavior and instant-by-instant knowledge of an object's state, allows SIoT to support requests from optimization systems by effectively pruning CLO graphs on the basis of social relationships. Therefore, it proves to be a critical part for real-time optimization.

2.3 SIoT platform

Conceptually, the Social Internet of Things (SIoT) platform consists of various clusters of SIoT peers. Each SIoT peer is made up of different functional blocks and exposes its data and services via REST calls (Fig. 1).

The **Cognitive Logistic Objects Repository** (CLOR) is a data structure that contains all CLO information. **Friend Tables** (FTs) are the data structure where the CLO Friends information is hosted. The SIoT hosts only one Friend Table for each CLO. The FT content represents the CLO friends and contains information about the friendships. The **Relationship Manager** (RM) module implements the logic through which a relationship can be created, deleted or updated. It is responsible for providing rules for implementing the social relationships among ID-labelled CLOs. When a new relationship is created, the RM writes directly in the Friend Table distributed database of the involved CLOs. The **Relationship Browser** (RB) module deals with navigating the social graph while implementing the search algorithms. In particular, the RB deals with the analysis of the FTs to find a target destination/set of destinations to be reached among the friends of a given entity. The **Identity Service** (IS) module is responsible for managing the Identifiers lifecycle. IS manages the IDs of the created or removed CLOs. **Subscription Service** (SS) allows to the peer or cluster RM to subscribe to information about CLO changes (for example position) and provides support for pushing the relevant updates. SS is exploited whenever the friendship tables of the CLOR managed by two different RMs need to be updated. The **Virtual Instance** (VI) mdule represents the virtual instance of CLO: when a CLO has to interact, make decisions or perform computation, this component is instantiated. It is released when the CLO gets idle. It represents the digital twin of the CLO, extends its capabilities and carries out social behaviours.

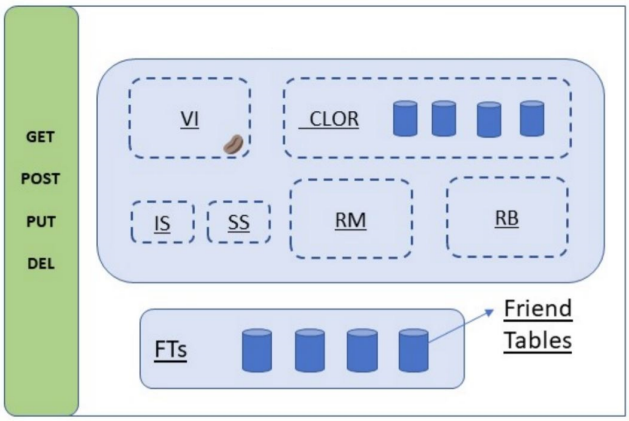


Fig. 1 – SIoT peer

3. COG-LO FRAMEWORK

The COG-LO framework aims to provide a holistic solution for handling the high operational dynamicity of the logistics environment. It follows a layered architecture and provides interoperable solutions regarding data models, information exchange and security mechanisms. This way it enables transparent coordination and exchange of information between different objects and systems based on heterogeneous access protocols in a secure manner.

3.1 Architecture

The physical entities that collectively carry out the actual logistics processes are denoted as the **Infrastructure layer**. These refer to a variety of concepts, including cargo (parcel, palette, container, etc.), transportation means (vehicles, trucks, trains, etc.), back-end ICT systems and services, as well as infrastructure components, like hubs, parking places, ports and other. The entities of the infrastructure are complemented by the datasources, including any source of data such as Enterprise Resource Planning, Warehouse Management System, Intelligent Transportation Systems, Traffic Information Systems, along with operational and configuration data that are essential for the operation of COG-LO.

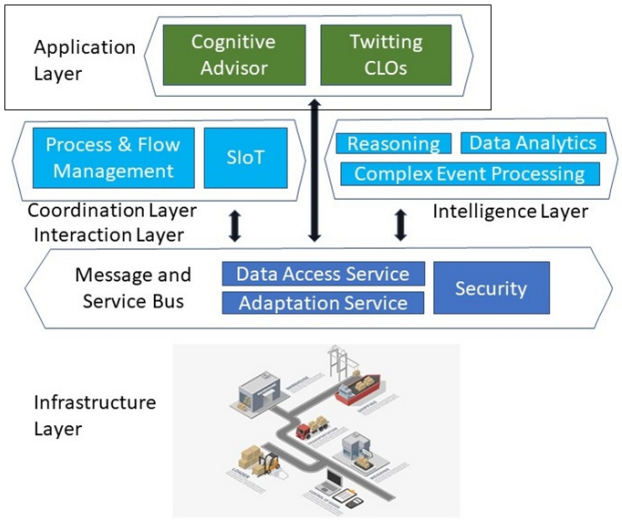


Fig. 2 – COG-LO Architecture

In order to provide for effective interaction, an **Interaction Layer** has been created. It includes a Message and Service Bus (MSB) which acts as mediation middleware between the components of the COG-LO ecosystem. The MSB comprises a message-oriented system providing both asynchronous and point-to-point message exchange between the system entities, circulation of events, and interaction between the CLOs. Furthermore, the MSB provides the integration of the infrastructure objects and data sources, by means of appropriate connectors. To this end, a fundamental duty of the MSB is the transformation of the Platform Independent Model (PIM) of the underlying components operational behaviour to the COG-LO Platform-Specific Model (PSM). The MSB incorporates also the functionality for the orchestration of COG-LO components and operations as regards the execution of workflows [21]. In addition, the MSB is the main COG-LO system entity for the enforcement of mechanisms for data security, privacy and trust.

The **Coordination Layer** incorporates the appropriate mechanisms both for the provisioning, management, monitoring and optimization of the Infrastructure Layer as well as for the effective coordination of resources and their actions towards business objectives. A key component of the Coordination Layer is thus the Social Internet of Things.

The **Intelligence Layer** provides the necessary logical inference mechanisms for knowledge extraction and formalisation, learning and reasoning, as well as cognitive behaviour of the underlying entities. To achieve this, the COG-LO intelligence Layer consists of multiple analytics technologies able to merge and aggregate data from different logistics entities and CLOs, to identify patterns, and to propose operation improvements. Secondly, predictive analytics with event processing enables foreseeing the impact of state changes of one or more operations and determining the corresponding effects on multiple stakeholders. COG-LO Intelligence Layer is coupled with optimization algorithms and heuristics for enabling CLOs adaptation to operational changes from the external environment in near-real time. In particular, COG-LO couples analytics and optimization for considering the effect of optimization control measures to the performance of operations in an environment with continuous external variations.

Finally, the **Application Layer** consists of the Cognitive Advisor (CA) and the Tweeting CLOs. The CA provides the logistics operator with visual decision support for routing optimisation. The CA interacts with the MSB to visualise the formalisation, reasoning and cognitive outputs of the Intelligence and Coordination Layers. The Tweeting CLO is the generic prototype consisting of the appropriate APIs that receive the messages from the logistics objects and vehicles and transmit them to the CLOs in the same network and to the CA via the MSB. Fostering interoperability, semantic data integration and operational effectiveness, COG-LO components rely on semantic ontologies for grounding data being collected, processed and disseminated, as well as establishing a common understanding between the collaborating entities.

3.2 Data model

The COG-LO integrates data from various sources that correspond to stakeholders with different roles in the logistics domain. The provenance of data and the use cases supported by this data exhibit great variety in their nature and cover a broad range of logistics services. Thus, the main challenges faced were (i) the design of a broad homogenized data model, which serves all purposes required, and at the same time (ii) the delimitation of the design to the context of the project.

The context of COG-LO demonstrates a dual nature, namely the physical and the digital one. In the physical context, objects perform normal logistics actions, while in the virtual context they act as virtualized entities with intelligence capabilities. In the physical context, logistics objects are parcels, containers, trucks, ships, trains,

employees with different duties in the logistics chain, postal offices, warehouses, ports, train stations, etc. The main set of capabilities exhibited by these objects is relevant with handling parcels. Logistics objects are related to organizations for which they execute pick-up and delivery plans. Properties of the logistics objects determine the planning procedure.

In the digital context, the situation is more dynamic. Sensors and smart devices attached to logistics objects, and which are assumed to be networked, transmit information about the underlying objects' state. The information is transmitted via messages that can be sent from one virtual object to the other. The range, to which the messages can be transmitted by a virtual object, is defined by social relationships, which are built and destroyed dynamically, and bear various "friendship" semantics among virtual logistics objects. As receptors of environmental information from the network, virtual objects can assess the state of their environment and cognitively react. Through optimization and decision-making capabilities, they can change their behaviour and suggest analogous actions to their social neighbours. Environmental changes may be populated by other virtual objects or by external components that contribute information to the network via appropriately formed messages encapsulating external events.

The design of the data model of COG-LO followed a top-down approach. Based on the identification of actors and their actions in the logistics setting of COG-LO, the base concepts both in the physical and the digital context were defined. In the physical context, there are four basic data types defined. The parcel is the principal unit of interest of logistics services. The existence of a parcel in the logistics chain starts with an order placed by a customer, and ends when the parcel is delivered to the recipient. Each parcel is attributed by a source location, a destination location, package dimensions and transportation type (e.g. normal, express etc.). The second concept is the container. Containers are placeholders for parcels. They are used to compose parcels into larger cargo units, and they may be encapsulated one inside the other. Besides similar attributes with parcels, containers have also a capacity associated with them, and also they are related with their content, i.e. the parcels or other containers they may carry. Transportation means, is the concept that identifies ships, trains and various types of vehicles. Sea, rail and road are the transportation modes considered in COG-LO, but air transportation could be represented similarly. Similarly to containers and parcels, there is a source and destination assigned to them at each point in time, when they are active in the logistics chain. Capacity is also associated with them and is measured in units corresponding to the amount of containers or parcels they can accommodate. Finally, a station is every establishment where a transportation means may stop and perform a cargo relevant action, that is load, unload, pack, unpack, customs' check etc. Such establishments are warehouses, ports, train stations, postal hubs and offices, end customers, customs and postal boxes.

In the digital context, the concepts of the physical context are mapped to their digital twins. They are attributed with capabilities of four levels: 1. transmit information, 2. receive information, 3. react to received information and 4. exhibit cognitive behaviour. Based on these capabilities, the digital concepts of VLO and CLO are defined as follows:

VLO definition

A Virtual Logistics Object is a virtual instance (VI) of a physical logistics object, which is composed by (i) the physical logistics object and (ii) the sensors and smart devices attached to it.

CLO definition

A Cognitive Logistics Object is a VLO for which the following hold: (i) the corresponding physical logistics object can perform at least one cargo action, (ii) it demonstrates at least 3rd level capabilities.

The Cognitive Logistics Object (CLO) constitutes the bridge between the physical context and the digital context of the COG-LO network. It is a virtual entity that corresponds to exactly one physical resource of the logistics network. For exemplifying the concept, let us assume a truck that transports parcels from one location to another. From the physical perspective, according to our previous analysis of capabilities' levels, it is a passive object. Nevertheless, when the truck is equipped with sensors measuring its weight and capacity, a GPS device tracking continuously its location, and we consider it being driven by a human with a smartphone, the perspective changes. This same object is now classified at least at the 3rd level of capabilities. It can inform constantly the network about the state of its deliveries, it can give an overview of the environment to the driver, and it can also let the driver assess a situation and resolve it, by following recommendations sent, or even by taking decisions based on situational awareness provided.

Each CLO is assigned with a plan. A plan is a sequence of physical actions assigned to the physical counterparts of CLOs. A plan comprises an ordered set of plan steps. Each plan step is executed at a specific location and consists of a series of cargo actions on parcels and containers: (i) loading/unloading, (ii) packing/unpacking (i.e., consolidation of parcels into containers) or (iii) customs checking. A plan step is associated with a cost estimation, which is what makes a plan get selected among a set of alternative ones during the optimization. Each plan is part of a recommendation produced by the Cognitive Advisor. A recommendation is generated as a response to an environmental change of COG-LO.

The COG-LO ontological model was defined, following the standard approach [22]. The development of the COG-LO ontology was performed using the Web Onotology Language (OWL) and the ontology is expressed in RDF/XML syntax. The data model of COG-LO is in detail presented by the COG-LO ontological framework [23].

3.2.1 Data storage, integration and exchanges

The main sources of information of the COG-LO framework are the databases of the pilot partners. They provide the core data set, based on which all processes of the system are executed. Each data model design, as well as the storage scheme of each source, is different and out of the scope of the project. The information integration is achieved through data connectors that are implemented as part of the Message and Service Bus of COG-LO. They are attached to the data sources and they assume the responsibility of transforming source data into the COG-LO data model.

The core system where the physical data model of COG-LO is deployed is the SIoT infrastructure. The social graph instantiates all virtual instances of the logistics objects, which are retrieved from the pilots' data sources, manipulates them and produces new data, i.e., their social relationships. The properties of the virtual objects as well as their interconnections in the social graph are available in real time to the Cognitive Advisor and the CLOs of the network. Various messages are exchanged (i) among CLOs, (ii) between CLOs and the Cognitive Advisor and (iii) between external event sources (e.g., Traffic Management System) and the Message and Service Bus. The structure of the messages follows the COG-LO data model.

The implementation of the SIoT data storage relies on the Apache Ignite platform [24]. It is a memory-centric distributed database, caching and processing platform for transactional, analytical and streaming workloads delivering in-memory speeds at petabyte scale. The inherent architectural design of Apache Ignite, which employs a distributed approach for both data storage and data caching, made it a natural solution for the implementation of the SIoT platform, where several social Cognitive Logistics Objects are required to be stored and updated in distributed fashion and simultaneously.

The SIoT data infrastructure is organized in a cluster of SIoT peers. One SIoT peer manages the data related to the VIs of the CLOs it is responsible for. Cluster nodes discover each other automatically enabling cluster scaling when necessary. The nodes are divided into two main categories: server and client. Server nodes are storage and computational units of the cluster that hold both data and indexes and process incoming requests along with computations. The platform is based on a durable memory architecture that allows storing and processing data and indexes both in-memory and on-disk, ensuring performance as well as durability.

3.3 Interoperability

The vastness of virtualized devices but above all the heterogeneity of their physical counterparts, requires sophisticated techniques to guarantee a high degree of interoperability in terms of communication and interaction.

The Message and Service Bus (MSB) plays a key role in this direction. It acts as the mediation middleware between the various components comprising the COG-LO ecosystem, and is assigned with the interaction, coordination and orchestration of COG-LO components and operations. In that respect, the MSB supports message exchange between system entities, circulation of events, interaction between CLOs. To this end, it facilitates cooperation within communities of CLOs and creation of ad hoc channels between CLOs by enabling the dynamic establishment of message topics.

Another fundamental challenge that the MSB tackles is the transformation of platform independent specifications of the underlying components operational behaviour to the COG-LO platform-specific model. The MSB provides for the integration of infrastructure entities and data sources, by means of data connectors.

In addition, the MSB is the main COG-LO system entity enforcing mechanisms for data security, privacy and trust, and enabling secure orchestration of COG-LO components as regards the execution of the necessary data flows, so that the reference operational scenarios are eventually fulfilled.

The Message and Service Bus provides a set of interfaces for the integration with COG-LO services, components, applications, as well as with external data sources and infrastructure entities:

- Entity management: this interface is used for adding, updating or deleting entities participating in the COG-LO system. It also provides a lookup method for getting details of COG-LO entities.
- Messaging: this interface is used by COG-LO services and applications for data communication. It also allows accessing information provided by various data sources, either internal or external to the COG-LO system.
- Data connector: this interface provides a unified solution for accessing information stored in heterogeneous systems, under a common transactional interface. It enables data interactions with the rest of the platform based on the COG-LO common semantic information model.

Data sources connected to the COG-LO platform (e.g., a traffic management system or public train timetables service) are initially registered to the MSB and data exchanges are handled by the corresponing data connectors. The latter consist of a set of data flows, which publish data to or ingest data from the platform, effectively hiding the implementation details of each data source.

3.3.1 Data flow management and orchestration

As part of the Message and Service Bus, the data flow management and orchestration solution enables end users to configure the way COG-LO components interoperate, in order to react to logistics events. It offers a user-friendly

way to connect COG-LO applications, platforms (e.g., SIoT, smart road infrastructure) and various data sources etc., on the grounds of the abstraction, adaptation and communication features of the MSB. The central point of the solution is its ability to continuously process streams of events and to orchestrate upon event receipt the operations that should be carried out, by which entities and in which order.

The data flow management and orchestration solution of the COG-LO system is based on the Apache NiFi integration platform [25]. Apache NiFi offers a visual command and control centre for designing, testing, deploying and monitoring data flows.

In order to integrate heterogeneous data sources, the following data flows types have been specified:

- Pre-Flow: transforms COG-LO domain requests to data source specific requests.
- Post-Flow: transforms the responses received by the data sources from the data source-specific model to the COG-LO domain model.
- Handler-flow: handles domain requests as they are received from the MSB and communicates with the underlying data source e.g. a database.
- Producer-flow: generates domain events after communicating with the underlying data source.

Outside of the context of a data connector the following data flow types are defined:

- Enforce-flow: processes domain data in a way that access control policies are applied to the data.
- Orchestration-flow: processes domain events and coordinates the chain of operations that need to be performed in order to fulfil given operational needs.

In order for the MSB to respond to an incoming request, for example in a point-to-point communication scenario, the MSB basically builds chains of data flow calls.

The COG-LO data flow management and orchestration solution employs data flows to model the interactions between COG-LO components, enabling end users to create custom data flows for handling incoming events e.g. traffic, emergency, general logistics events. This enables flexible integration of information systems and supports the realization of the COG-LO vision: facilitating the creation of ad hoc logistics collaboration by combining digital processes with physical procedures taking place at the level of actual cargo and means of transportation.

In order to support pilot operations and drive business scenarios a set of orchestrations have been deployed on the COG-LO platform. Their purpose is not limited to order management, but also to provide support for accidental events (e.g. vehicle breakdown) driving dynamic rescheduling of daily deliveries. The orchestrations leverage the Social Internet of Things for this task to intelligently select candidate objects, then contact the Cognitive Advisor to receive updated plans for the vehicles involved and finally initiate the negotiation orchestration.

3.4 Security, privacy and trust

The MSB as the core communication module of the system, enables access to both internal and external services and information through a unified interface. In this context, the mechanisms for security, privacy and trust cover all respective technologies, notably access and usage control, cryptography and trust infrastructure. Specifically, COG-LO provides a solution for identity management offering standard-based means for authenticating COG-LO actors; a policy-based access and usage control framework for regulating the circulation and usage of information. It additionally provides an architecture with a standard set of components, such as Policy Administration Point (PAP), Policy Decision Point (PDP), Policy Information Point (PIP) and Policy Enforcement Point (PEP) for evaluating access control policies.

The COG-LO platform adopts a token-based authentication solution offered by RedHat's KeyCloak identity and access management component [26], and is used for both user authentication as well as component (service) authentication.

For what concerns the access control, COG-LO adopts the Attribute-based Access Control (ABAC) paradigm and is established upon the XACML 3.0 language and reference architecture. In particular, the starting point for incorporating ABAC authorisation functionality within the MSB has been extensible Access Control Markup Language (XACML) that has been extensively used in academia and industry.

In line with the XACML reference architecture, the MSB, as the PEP, provides the mechanisms for enforcing the specified access and usage control policies when it comes to regulating message exchange between COG-LO components, services or CLOs. The MSB interacts with the PDP by providing attributes obtained by the original request, with the latter transformed in the XACML format that the PDP can process.

In the context of COG-LO, AuthZForce [29] has been selected as the policy decision engine as it implements the OASIS XACML 3.0 core specification, and provides an API to get authorisation decisions, based on authorisation policies, and authorisation requests from PEPs.

Cryptography traditionally represents the bottom line of data protection. Therefore, COG-LO puts in place a rich functional toolkit able to support all necessary cryptographic functionalities to foster data confidentiality. To this end, the COG-LO crypto-engine leverages a plethora of cryptographic primitives, both symmetric and asymmetric. Furthermore, COG-LO adopts the advanced technology of Attribute-Based Encryption (ABE), targeting the cryptographic enforcement of data disclosure policies by leveraging the attributes assigned to entities, being people or systems.

Concerning aspects such as secure channel establishment, COG-LO identified a series of shortcomings to the cornerstone technologies that facilitate secure information exchange over the Internet, namely the Public Key Infrastructure (PKI) and associated X.509 certificate standard [30]. Specifically, as it has been recently shown, PKIs are exposed to risks due to errors or breaches involving Certification Authorities (CAs), resulting in unauthorised certificates being issued and compromising thus the security of the corresponding end users. In light of the above, COG-LO adopts a novel blockchain-based solution enabled by the Hyperledger [27] family of technologies, namely the Hyperledger Indy and the Hyperledger Aries frameworks in order to establish secure cross-organisational communication. The aforementioned solution, being based on the blockchain technology, inherits inevitably its advantages. The solid basis of the public append-only log (past logs cannot be changed unless the blockchain is subverted by a dishonest network majority), eliminate the single-point-of-failure issue and enables rapid reaction to identity revocations since DIDs can be validated on the distributed ledger.

4. PERFORMANCE ANALYSIS

The fundamental aspect that is used to evaluate the performance of the COG-LO framework is the time consumption of the algorithms implemented within the Social Internet of Things (SIoT) and the optimizer.

More specifically, the scalability of these components is addressed by observing how the computational time for object digitalization varies as the number of objects increases, how the social graph surfing time varies when a friend must be discovered as the size of social graph varies, and finally by the complexity of the optimization algorithms in large environments, where the problem of a large amount of data and variables to be analyzed must be faced.

The Social Internet of Things (SIoT) architecture consists of various SIoT clusters. Each cluster is implemented using Apache Ignite to support a memory-centric distributed database, caching and processing platform [17]. Each SIoT peer can automatically discover each other in order to create a cluster or to browse another peer's social graph. In the first performance study of the SIoT, the response time for the creation of a SIoT social graph was observed, in relation to the size of the graph (number of CLOs). The initialization process entails the instantiation of the digital twins of logistics objects, along with their relevant data structures [28]. Fig. 3 demonstrates how this process scales in time increasing the number of CLOs forming the social graph.

The SIoT graph receives update requests every time the state of CLOs changes. These updates require recalculating the relationships between the CLOs. In this experiment, the time required for a CLO state update and the creation of social relationships between a CLO and N friends CLOs is observed. Fig. 4 depicts the results and demonstrates the time necessary to modify the data structures of both the CLO that wants to establish a friendship and the data structures of all involved friends (since the social relations are bidirectional).

As usual in traditional distributed deployments of systems like the SIoT platform used in this project, different servers are used to share the load of traffic and computation. Each server can be configured to work following a full replication or a partial replication scheme, or even as a totally independent system with non-replicated data. Using the first approach all the data is replicated or copied to all the participating nodes in the cluster. Otherwise, using the second approach, the entire data is split equally into partitions and is stored in the participating nodes, thereby creating a distributed storage of data. The total storage space depends on the total memory available across the peer. As shown in Fig. 5, the replicated mode allows the speed-up of the discovery process, since the information is immediately available in the peer from which the search is being performed. While this approach has benefits in terms of time, it also requires an increased use of resources.

The optimization performed by the CA is based on the graph representation of the CLOs network. Upon the occurrence of a disruption event (e.g., an ad hoc order, a traffic event etc.), the graph gets pruned by SIoT to include only CLOs in the vicinity of the event. The size of the graph, i.e., the number of vehicles included in the optimization impacts greatly the response time.

The performance of the optimization algorithms is presented in Table 1. It clearly shows the importance of graph pruning to achieve real-time responsiveness to disruption events. Table 1 shows the performance time for optimization processing on pruned graphs. An optimization algorithm uses exact methods, with linear solver, where using a large number of CLOs exponentially increases processing time. For optimization processing, the total infrastructure graph is clustered into regional representation with graph sizes of 300-500 CLOs (postal offices, vehicles, parcels). The processing time in Table 1 clearly shows that without pruning the graph and omitting the number of CLOs included in event handling, the system would not be able to create real-time responses.

Table 1 – VRP optimization response time, based on the number of CLOs included

vehicles/post offices	20 CLOs	25 CLOs	30 CLOs	40 CLOs
2 CLOs	4.9s	12.4 s	28.7s	95.1s
3 CLOs	9.9s	26.3s	43.4s	168.3s
4 CLOs	18.1s	38.2s	78.5s	258.2s
5 CLOs	27.5s	52.6s	127.2s	378.4s
6 CLOs	40.7s	117.4s	228.4s	592.2s
7 CLOs	52.8s	172.1s	415.6s	865.4s
8 CLOs	74.3s	230.6s	720.1s	1923.3s

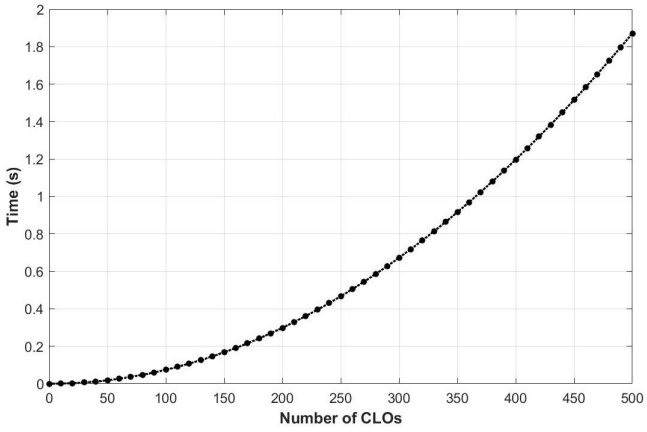


Fig. 3 – Required time for digital twin creation and SIoT graph initialization varying the number of CLOs

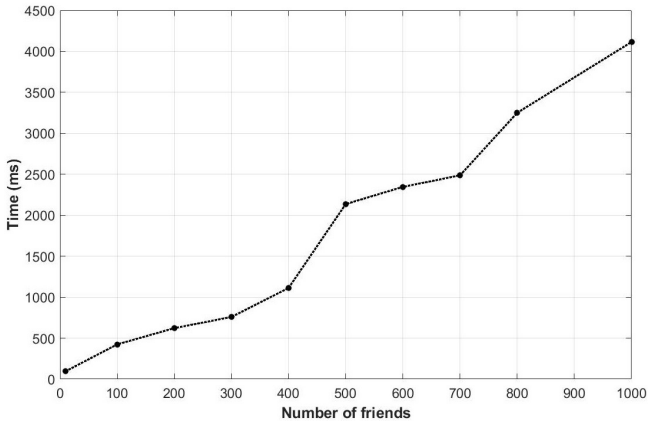


Fig. 4 – Time consumption for SIoT graph updates on creating new friendships between a CLO and varying number of friends

5. CONCLUSION

In this article, the innovative concept of the Cognitive Logistics Object, introduced within the COG-LO project, is presented. CLOs represent all the entities involved in logistics processes. They are made autonomous and dynamically reactive to the surrounding environment thanks to their cognitive capabilities. CLOs are able to collaborate and implement functionalities stemming from social networks thanks to virtualization techniques and the properties of the Social Internet of Things (SIoT) framework that has been exploited. The efficient architecture that implements the COG-LO framework has experimentally been tested, demonstrating that the proposed solution allows for optimization operations in responsive time outperforming in-place logistics procedures.

The results achieved through the application of SIoT and the relevant virtualization techniques are encouraging. The significant benefits in terms of interoperability, motivates for exploring the topic on a large scale. The adoption of the SIoT and the CLO concept proves to effectively mit- igate the optimization problem complexity, which allows the use of classical optimization techniques for real-time events processing. The exploitation of SIoT and CLO used in the COG-LO project offers an effective approach in managing new emerging problems in logistics infrastructure.

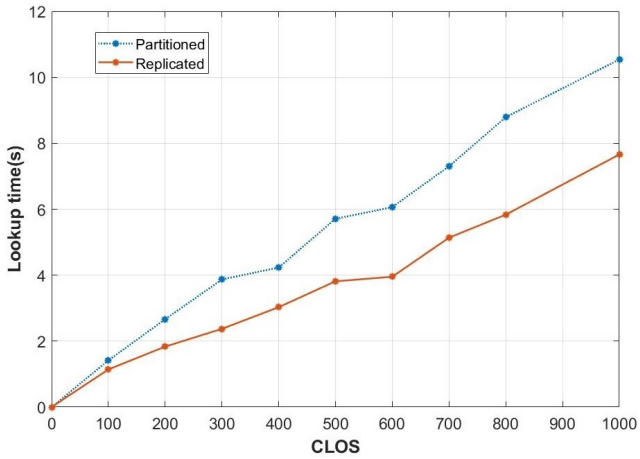


Fig. 5 – SIoT lookup time varying the number of CLO friends searched for replicated vs partitioned memory mode

ACKNOWLEDGEMENT

This work was partially supported by the European Union's Horizon 2020 research and innovation program under the COG-LO project (grant agreement no. 769141).

REFERENCES

[1] Walid, Taha, Abd-Elhamid, Thunberg, Johan, *What is a Cyber-Physical System?*, 10.1007/978-3-030-36071-9-1, 2020.

[2] http://www.cog-lo.eu/.

[3] Evtodieva, T. Chernova, D. Ivanova, N. Kisteneva, *Logistics 4.0*, 10.1007/978-3-030-11754-2-16, 2019.

[4] L. Atzori, et al., *The social internet of things (SIoT) - when social networks meet the internet of things: Concept, architecture and network characterization*, Computer Networks 56(16), pp. 3594-3608, 2012.

[5] S. Belardo, H. L. Pazer, *Scope/Complexity: A Framework for the Classification and Analysis of Information-Decision Systems*, J. of Management Information Systems 2(2): 55-72 (2015).

[6] Psaraftis HN. *Dynamic vehicle routing: Status and prospects*, Annals of Operations Research, 61:143-164, 1995.

[7] GS1, Logistics Interoperability Model, Version 1, Issue 1.0, http://www.gs1.org/lim, August 2007.

[8] Pillac, Victor, et al., *A review of dynamic vehicle routing problems*, European Journal of Operational Research 225.1: 1-11, 2013.

[9] M. Grieves, *Digital twin: Manufacturing excellence through virtual factory replication*, 2014.

[10] M. Heutger and M. Kuechelhaus, *Digital twins in Logistics: a DHL perspective on the impact of digital twins in the logistics industry*, DHL, 2019.

[11] R. Minerva, G. M. Lee and N. Crespi, *Digital Twin in the IoT Context: A Survey on Technical Features, Scenarios and Architectural Models*, Proceedings of the IEEE, vol. 108, no. 10, October 2020.

[12] Gartner, *Gartner Identifies the Top 10 Strategic Technology Trends for 2019*, 2018: https://www.gartner.com/en/newsroom/press-releases/2018-10-15-gartner-identifies-the-top-10-strategic-technology-trends-for-2019.

[13] K. Kalaboukas, G. Lioudakis, M. Koukovini, E. Papagiannakopoulou, G. Morabito , N. Dellas, M. Zacharias, S. Quattropani, M. Samarotto, M. Jermol, M. Cimperman, L. Stopar, M. Senozetnik, S. Bratusa, A. Kahvedzic, D. Justament, I.-K. Buntic, H. Marentakis, A. Maragakis, E. Al, E. Özsalih, A. Kiousi, A. Dimitriou, G. Galli, E. Pastori, E. Gualandi, F. Alesiani, G. Ermis, T. Jacobs, I. Mourtos, S. Lounis and G. Zois, *Cognitive Logistics Operations through Secure, Dynamic and ad-hoc Collaborative Networks: The COG-LO project*, London, 2019.

[14] M. Nitti, et al., *The virtual object as a major element of the internet of things: a survey*, IEEE CS and T, 18(2), pp. 1228-1240, 2016.

[15] L.Atzori, C. Campolo, et. al., *Enhancing Identifier/Locator Splitting through Social Internet of Things*, IEEE Internet of Things Journal, 2018.

[16] J. Travers, S. Milgram, *An experimental study of the small world problem. Sociometry*, 32, pp. 425–443, 1969.

[17] http://www.cog-lo.eu/sites/default/files/documents/2020-05/COG-LO-D3.6-COG-LO architecture definition (final version) – v1.0.pdf

[18] Atzori Luigi, Campolo Claudia, Iera Antonio, Milotta G., Morabito Giacomo , Quattropani, S., *Sociocast: Design, Implementation and Experimentation of a New Communication Method for the Internet of Things*, 662-667. 10.1109/WF-IoT.2019.8767348, 2019.

[19] He, Ruhan, et al., *Balanced k-means algorithm for partitioning areas in large-scale vehicle routing problem*, 2009 Third International Symposium on Intelligent Information Technology Application. Vol. 3. IEEE, 2009.

[20] Bent, Russell, and Pascal Van Hentenryck, *Spatial, temporal, and hybrid decompositions for large-scale vehicle routing with time windows*. International Conference on Principles and Practice of Constraint Programming. Springer, Berlin, Heidelberg, 2010.

[21] http://www.cog-lo.eu/sites/default/files/documents/2020-11/D4.4 Coordination Mechanisms (final).pdf

[22] S. Negru, F. Haag, and T. Ertl S. Lohmann, *Visualizing Ontologies with VOWL*, Semantic Web Journal, 2015.

[23] http://www.cog-lo.eu/sites/default/files/documents/2020-05/COG-LO-D3.4-COG-LO_dataModel_ontologicalFramework-final.pdf

[24] https://ignite.apache.org/

[25] https://nifi.apache.org/

[26] https://www.keycloak.org/

[27] https://www.hyperledger.org/

[28] http://www.cog-lo.eu/sites/default/files/documents/2020-02/COG-LO-D3.3-COG-LO_dataModel_ontologicalFramework-final.pdf

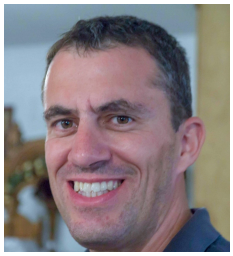
[29] https://authzforce.ow2.org/

[30] Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, IETF RFC 5280, May 2008, https://tools.ietf.org/html/rfc5280.

AUTHORS

Miha Cimperman, Ph.D, is a researcher and project lead at the Jozef Stefan Institute, Laboratory for Artificial Intelligence, Slovenia. His work focuses on research in the field of knowledge services development for critical infrastructure, namely: logistics, energy and finance infrastructure. His research is focused on data modelling and analytical solutions design based on sensor IoT data and text data.

Angela Dimitriou, is an Electrical and Computer Engineer, graduated from the National Technical University of Athens, Greece. She received her Ph.D. in the scientific domain of databases, collaborating with the Knowledge and Database Systems Laboratory of NTUA, Greece. Her research interests include data management, semi-structured data, keyword search, information retrieval, graph data and algorithms. She works at the National Technical University of Athens and has been collaborating with INTRASOFT International for several European research projects.



Kostas Kalaboukas, is Technology Transfer and New Business Development Manager at Gruppo Maggioli and candidate Ph.D student at the Technical University of Crete. His research focuses on digital supply chains and smart cities and has participated in many research projects in the past. He has studied Production Engineering and Management at the Technical University of Crete (1991-1996) and obtained an MSc in Total Quality Management at the University of Piraeus (2000). He has wide experience in business analysis, quality assurance and management of complex software projects.

Aziz S. Mousas, received his diploma in Applied Mathematics from the school of Applied Mathematics and Physics of the National Technical University of Athens (NTUA) in 2008, and his Dr.-Ing. degree in Electrical and Computer Engineering from NTUA in 2014. He has participated in several European and national R&D projects as a researcher, software engineer, project manager and ICT consultant. His research and professional interests include security and privacy protection, software engineering, middleware and distributed systems, as well as the applications of semantic ontologies in these fields. He has more than 20 publications in international journals, conferences and books related to these areas, as well as contributions to the MPEG-21 and MPEG-M ISO standards.

Salvatore Quattropani, received BSc. and MSc. degrees in Computer Engineering in 2016 and 2017, respectively, from University of Catania, Italy. Since August 2017 he is with the Consorzio Nazionale Interuniversitario per le Telecomunicazioni as a research engineer. His research interests focus on IoT connectivity and computing approaches.

LYSIS CHATBOT: A VIRTUAL ASSISTANT FOR IOT PLATFORMS

Raimondo Cossu[1], Roberto Girau[2], Luigi Atzori[3]

[1,3]Dept. of Electrical and Electronic Engineering, University of Cagliari and National Telecommunication Inter-University Consortium (CNIT), Research Unit of Cagliari, Italy., [2]Dept. of Computer Science and Engineering, University of Bologna, Italy

NOTE: Corresponding author: Roberto Girau, roberto.girau@unibo.it

Abstract – *The configuration and management of devices and applications in Internet of Things (IoT) platforms may be very complicated for a user, which may limit the usage of relevant functionalities and which does not allow its full potential to be exploited. To address this issue, in this paper we present a new chatbot which is intended to assist the user in interacting with an IoT platform and allow them to use and exploit its full potential. The requirements for a user-centric design of the chatbot are first analyzed, then a proper solution is designed which exploits a serverless approach and makes extensive use of Artificial Intelligence (AI) tools. The developed chatbot is integrated with Telegram to message between the user and the Lysis IoT platform. The performance of the developed chatbot is analyzed to assess its effectiveness when accessing the platform, set the main devices' parameters and request data of interest.*

Keywords – Chatbot, IoT platform, Lysis IoT, user experience

1. INTRODUCTION

In recent years, the development and deployment of chatbots to be used in several scenarios have risen significantly, and many businesses have opted to use these in their services. In many sectors, such as e-commerce, insurance, banking, healthcare, finance, legal, and others, chatbots are currently used to support the execution of a variety of business activities. Gartner Summits [1] predicts that over 70 % of customer interactions will involve emerging technologies such as Machine Learning (ML) applications, chatbots and mobile messaging by 2022. The objective of a chatbot is to emulate the conversational capabilities of humans so that when a person interacts with a chatbot they behave as if they were interacting with a peer. This is possible because the chatbot goes through a series of steps to process human data and then determine an appropriate response or action based on the user's query. There are already various examples of Artificial Intelligence (AI)-based chatbots, for example: Cleverbot, Cortana or Tay. First of all, Tay [2], Microsoft's first public experiment involving the test of a bot on Twitter, was so successful that it began to behave like its followers. Over time, however, after just 16 hours of activity it was necessary to turn it off because they she had begun to exhibit xenophobic, feminist and racist behavior. It was a similar ending for the conversation between two AI entities developed in the Facebook labs, trying to make them talk to each other, after some time they began to speak a language that was known only to them. While the epilogue was not what was expected, these experiments showed how much the technology around smart chatbots had evolved. Chatbots are a hot topic among tech behemoths like Facebook and Microsoft, as well as smaller messaging platforms like Telegram and Slack, which have made their frameworks available to developers to ensure smooth development.

A chatbot can be used for a variety of purposes, and the Internet of Things (IoT) can easily be added to this list. One of the reasons why the IoT is struggling to take off is the difficulty of less experienced users installing or configuring their devices, as well as solving small, common problems. This forces users to rely on qualified staff to fix simple problems on a regular basis, which makes the user experience frustrating. In this scenario, a chatbot can have a vital role in improving the user experience, as when properly programmed and inserted within the reference IoT platform, it would give the user the necessary support when dealing with complicated actions thus fulfilling the lack of skills. Obviously, it would not only provide some help in performing specific actions but it would also provide information that will be specifically requested by the user, such as, for example, about the status of their car, home, work and so on, significantly reducing the barriers between the user and connected objects. All in natural language (human language) rather than relying on navigation through a graphical interface in a mobile application or website. Unfortunately, progress is currently being made at a snail's pace. The truly conversational chatbot, which will be able to autonomously interpret user inputs, is still a long way off, but several research efforts are moving towards the unification of an ecosystem that is currently very fragmented. The IoT space is at an inflection point, with conversational user interfaces at the forefront. This process is becoming more achievable day-to-day with services that help companies to easily integrate natural language understanding into their products. The scope for conversational user interfaces is enormous, and it continues to expand. With a variety of technology available for the implementation of such systems, the next step is to figure out where machine learning strategies make more sense than other technologies and whether they can potentially save us time and enable people to focus on more valuable tasks.

This work will focus on the interfaces between users and IoT-enhanced environments. Indeed, the chatbot will provide users with a user-friendly interface to assist them in creating their profile and managing their services and objects. The virtualization of the user is also introduced to suggest relevant services that are expected to be the most suitable and interesting for each user based on their profile and thanks to context-awareness mechanisms. The use of chatbots in the IoT already has a history, but the main limitation comes from the fact that these are vertical and domain-specific solutions. The solution we propose, thanks also to the intermediation of user/device virtualizations, allows the reuse of the same chatbot interface on different IoT applications.
Accordingly, the contributions of the paper are as follows:

- We analyze the key requirements for the development of a chatbot for the IoT scenario and provide a description of the architectural components of the chatbot-enabled user virtualization;
- We discuss the integration of the chatbot system in a fully distributed virtualization-based IoT architecture;
- We provide the details of the implementation that have been carried out to develop a prototype;
- We present some experimental results for the evaluation of the capability of the proposed solution to identify correctly the user intention interacting in the IoT environment.

The paper is structured as follows. Section 2 discusses the major related works in this area. Section 3 presents the key requirements in designing a chatbot system in a virtualization-based IoT platform. The system architecure is presented in Section 4, while in Section 5 we illustrate a use case to better understand the reference scenario. The implementation and the experimental results are shown in Section 6 and Section 7 respectively. Finally, Section 8 concludes the paper.

2. STATE OF ART

The first entertainment chatbot was developed in 1966, it was called ELIZA [3] and was a parody of a psychotherapist who answered the patient's questions with other questions, obtained by rephrasing the patient's questions. In 1995, Richard S. Wallace built A.L.I.C.E. [4] a chatbot made entirely with open source software that uses the AIML language, child of the XML language from which it inherits extensibility, which thus allows the chatbot to hold a conversation. With the growing interest in artificial intelligence and with the idea of simplifying the interaction between man and machine, more and more companies, have developed or directed part of their research on chatbots.

In industry and in particular in the IoT field, chatbots are entering in workflows in a capillary way. This is because thanks to their characteristics they allow to stem the difficulties of configuring and troubleshooting devices encountered by operators and users.
The first problem that arises when new solutions have to be introduced into existing systems is to understand the impact in terms of complexity. To understand the complexity in [5] the authors analyzed the possibility of creating a general architecture that would allow the integration between chatbot and IoT systems in a simple way. The study found that what chatbots and IoT have in common is that they adopt their services through relatively simple, often RESTful, web APIs. In this scenario, adopting a service-oriented development approach to development, integration is feasible thanks to RESTful HTTP standards and protocols. In this case the ISO/OSI application level is the only level concerned, without having to go down to the underlying levels. It is therefore clear that with design precautions, the integration between chatbot and IoT platforms is extremely simple.
In the literature there are several examples of systems that use chatbots to interact with IoT devices. In [6] the authors implement a chatbot integrated with an agricultural plant monitoring system. In their implementation they use fuzzy logic and Natural Language Processing (NLP) to interpret user inputs. The user asks the plant a question and it answers. An orchid was used for the experiments. The success rate of the interaction between question and answer was 71%.
An interesting proposal is presented in [7], an IoT system with AI chatbots for plant monitoring capable of monitoring various parameters useful for knowing the health of houseplants. Alongside the IoT system, we implement a chatbot to inform the owner about the current conditions of the plant and its current needs. The data is also stored and through the bot the user is able to analyze the graph and determine the level of wellbeing of the plant and any problems.
In [8] an integrated Chatbot-IoT system is implemented to make the monitoring and improvement of water quality quick and efficient. For monitoring, a network of IoT sensors was created, supported by a cloud platform. Inside, a chatbot has been integrated that uses text mining techniques to interpret user inputs. The result showed excellent performance with high precision and recall for each class.
In [9] and in [10] two IoT platforms for home monitoring and remote control are presented. They have a built-in chatbot that can understand text or voice commands using NLP. Using different APIs and protocols, the authors have obtained user-friendly systems for controlling home devices. They also demonstrated how an architecture structured on multiple services is effective and easy to implement.

The authors in [11] focus on integrating chatbots and IoT to address a critical problem such as air quality awareness. In this case, the chatbot not only provides users with information on air quality, temperature and humidity, but also provides services such as subscriptions to preferred air quality monitoring points. Furthermore, advanced functions have been implemented that can be managed entirely via chat such as: alarm services, threshold settings, geoquery and advice based on pollutant levels.
In [12], a healthcare prognosis chatbot based on AI-IoT and with adaptive learning capabilities is proposed. The aim of the system is to provide medical diagnoses in real time and to support patients in the absence of healthcare professionals. The interactive system provides tools to collect data, answer general medical questions, provide assistance and provide alerts to remind patients that they need to take their medication. The system in question has shown an accuracy of 90% of the answers. Similar to the previous one, in [13] is presented a chatbot designed to increase the capacity of health services so as to reduce the management costs for medical consultancy services. Unlike the [12] proposal, this chatbot is paired with an IoT device for detecting vital signs. This combination can help people know their health status.
With COVID-19 social stress has grown exponentially, the proposed work in [14] uses a chatbot to defeat the stress of individuals during the period of isolation. This chatbot allows persons to interface with remote clinical specialists. In this case, artificial intelligence and NPL techniques combined with a clinical chatbot. This will understand if it is enough to continue the conversation with the bot or if the user needs to interact with a human professional. From what has been analyzed, it can be seen that integration with IoT systems is very useful for enabling inexperienced users to use advanced features in a simple way. Our proposal is to insert a support chatbot within a Social IoT (SIoT) platform. In this way, all the applications that will be hosted within it will be easily usable and configurable even by the less experienced.

3. BACKGROUND

Currently, there are multiple architectural solutions for IoT (vertical, horizontal, centralized or distributed solutions, etc.) with involvement at various levels of the user in interacting with devices that surround them. The design of an intuitive interface requires key requirements that best fit the chosen architectural solution. The following subsections show the technological needs in the design of a chatbot and the chosen reference IoT architecture.

3.1 Key requirements in designing a chatbot system

In human-interaction-based applications the processing time and the latency in general are key requirements. Similarly, in a chatbot application, users expect immediate responses in comparison to other web and mobile applications. The processing time should not increase directly or exponentially with the number of users, but rather should be constant and perform at its best almost regardless of the workload. To get high scalability, we can rely on serverless cloud services such as Amazon AWS, Google Cloud Platform, IBM OpenWhisk or Microsoft Azure. On these platforms, we are able to develop lightweight event-based architectures so as an event can have more than one handler and is also able to start the execution of short isolated parts of codes written in order to perform specific atomic tasks. Additionally, each event handler can create one or more event after processing the event data. Function as a Service (FaaS) is a cloud service model based on serverless architecture that allows developers to build a flexible system that fits well to pulling entire functions up and down for each request. In chat applications, the speed with which applications are instantiated is crucial to reduce latency times. In an FaaS solution, the platform manages the loads at the level of individual requests, optimizing in terms of performance and costs. However, it is not possible to implement a chatbot system entirely in FaaS, as there are other features that require other service models to ensure, for example, data persistence, back end to an IoT platform or front end for user interface rendering. And it is not recommended to use exclusively a container-based service model (Containers as a Service (CaaS)) even if currently Kubernetes, at the level of scaling, is approaching FaaS solutions thanks to intelligent traffic management based on analysis models that imply FaaS features. Based on the application context, however, we can think of a hybrid use of containers and FaaS, which is the solution we adopted in our system.

The use of a pay-per-use model reduces operating costs compared to a traditional system that requires the allocation of the resources of one or more processing instances. In fact, the FaaS model allows you to activate the necessary functions on request and to release them immediately after the execution of the tasks. So we can see that, given the speed with which requests must be processed and given the conditions of traffic non-uniformity that make it impossible to estimate the users who will actually request the service, the most convenient solution for the implementation of the chatbot is the serverless one. In addition to scalability, it is also necessary to pay attention to the latency at start-up, that is the time that a FaaS function takes to respond to requests. Typically in all the platforms mentioned they take from a few milliseconds to a few minutes, this time is variable and depends on various factors, such as programming languages, for example.

Another key requirement is the management of the state, a serverless system by its nature is stateless, to overcome this problem we will rely on an instance of a database that will take care of saving both the state and further inputs that will be used to manage the requests.

3.2 Reference IoT architecture

Recently, several studies have looked at the problems of managing and effectively using large numbers of heterogeneous devices, and have found a solution in the use of social networking principles and technologies. In [15], the definition of the Social IoT (SIoT) has been formalized, and it is intended to be a social network in which each node is an entity capable of forming social relationships with other things on its own, according to the rules set by the owner. The proposed model is based on the Lysis cloud SIoT architecture [16], which incorporates virtual objects as digital counterparts to physical objects to enhance their capabilities in a transparent manner to users. Lysis architecture foresees a four level structure of independent modules. Its lowest layer is populated by Real World Objects (RWO), i.e. physical IoT devices able to perform basic tasks. On top of this, the virtualization layer, directly interfaces with the real world and is populated by Social Virtual Objects (SVO), which are VOs with socialization capabilities. The aggregation layer is responsible for composing several SVOs into entities with extended capabilities, called Micro-Engines (MEs). Finally, at the application layer, user-oriented macro-services are provided (APP).

Socialization algorithms implemented in the first two levels allow for the creation of social relations as foreseen in the SIoT paradigm. The resulting social graph is exploited to find the required resources.

3.3 User virtualization in IoT

The widespread presence of connected objects throughout daily life has allowed the Internet of Things (IoT) to spread. The IoT vision forms a collaborative ecosystem for a multitude of heterogeneous objects with different connectivity and computing capabilities to achieve the common purpose of providing user services.

At the current time, the IoT platforms seem to present several pending issues that prevent a full spread of IoT applications. Indeed, services are mostly configured manually by users, according to preferences that could be shared among similar or cross-domain services (e.g., preferences about ambient temperature at home and at work to manage HVAC systems). Secondly, the users that access an IoT platform need to autonomously look for the required services among a plethora of them. The risk is in a decrease in the quality and reliability perceived by users, who therefore risk being discouraged from using IoT applications. Our system has the objective of exploiting the concept of Virtual User (VU)[17] to improve the user experience and, at the same time, enhance the efficiency and usability of the IoT platforms and services.

The VU is the virtualization of a user, and it is represented by an agent that enables the following major benefits: providing users with a user-friendly interface that enables automatic or assisted setup of their profile, objects and services; proposing the services that are expected to be the most suitable and interesting for each user, based on their profile and context: and enabling objects to be as much plug & play as possible.

The specific focus of this work will be on the interfaces between the users and IoT enriched environments. Indeed, a user-friendly interface will be provided by means of the chatbot to users to assist them to create their profile and manage their services and objects. Based on their profile and thanks to context-awareness mechanisms, the VU will be able to suggest relevant services that are expected to be the most suitable and interesting for each user, and settings will be automatically configured.

4. PROPOSED ARCHITECTURE

The proposed solution is aimed at designing and experimenting a chatbot system that simplies the interaction of the users with the VU in an IoT platform by means of text messaging. As previously explained in Section 3, the VU is the virtualization of the user and takes decisions on their behalf for known activities; as such, it interacts with all the modules of the Lysis IoT architecture [17]. The VU was not introduced specifically for the Lysis platform. In fact, it follows the more general concept of virtualization in the IoT and of virtual objects. The concept of VU arises from the need to provide a virtualization element that constantly deals with the context of the user it represents, their interaction interfaces and their IoT services. In this scenario, the VU is a standalone element in a distributed virtualization system. The Lysis platform, which we use as a development environment, is precisely a distributed system of elements that allows for the creation of a social network among virtual objects in order to facilitate their interaction. The VU could be used in a centralized system, possibly vertical; however, in this case it would not bring all the advantages that characterize the implementation in a distributed system. Furthermore, any IoT platform that is a candidate for the integration of the VU and its interfaces, such as the chatbot, should provide open APIs that allow for full integration.

Fig.1 shows the components of the overall architecture according to the Lysis model. The VU communicates with all levels to provide user preference information to build tailored IoT services. The chatbot system is a back-end service for proxies the communication between the users the the VU.

Fig. 2 shows the architecture of the proposed solution that has been designed to address the requirements that have been previously discussed.

The upper layer implements the functionalities to receive and transmit requests and data. The requests are generated by either web apps or (proprietary / non-proprietary) messaging services which are used by the user for sending and receiving messages. Each request contains the *intent*, i.e., the action that the user would like to take, which is written in natural language. The intents are then received by the chatbot API gateway to be sent to an AI-based service that interprets the intents to

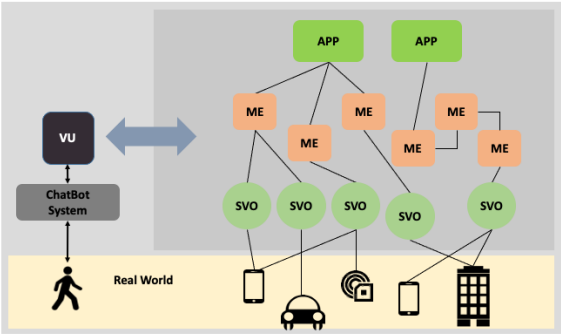


Fig. 1 – ChatBot-enabled VU in the Lysis architecture

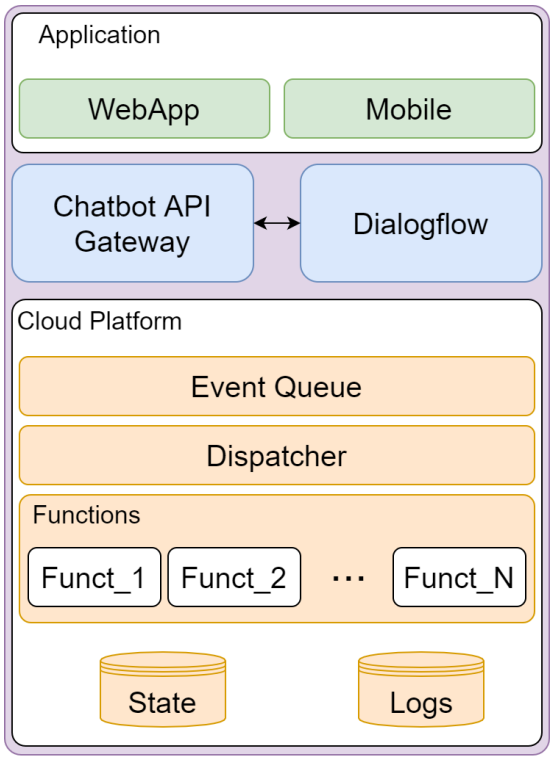


Fig. 2 – The propose chatbot architecture

understand what events are associated with these. In particular, in the developed solution, we have adopted the Dialogflow service. The resulting event is then communicated to the gateway (through the edge layer), which takes care of sending it to the interface that then stores it in an event queue waiting for the dispatcher to direct it to the needed function. Indeed, there is a different function for each event or group of events that is activated when a given event is received. At this moment, a search is then carried out to understand if there is already an active instance for this function; if not, a new instance of this function is activated, the event status is temporarily saved and then finally the function instance is destroyed once it is no longer needed. The functions to be implemented do not have to be all accessible via a gateway route; in fact, it will be sufficient to make them accessible via a specific topic. In this way it is easier to manage the planned events also considering the interaction between the same functions. The gateway is an HTTP server in which routes and endpoints have been defined, where each route is associated with a FaaS function. When the gateway receives a request, it identifies the corresponding routing configuration by calling the relevant FaaS function. Fig. 3 shows the gateway workflow. When the user at a given time needs to request information, they will forward a message to the gateway. Herein, let us assume that the user is already authenticated; at the time of sending the request, a POST call is made to the URL / API / createHook. This URL takes care of creating the WebHook to use for communicating (conversing) with the cloud functions. The WebHook consists of two basic parts: a token and an event. The token is generated during the creation of the WebHook and is used to authenticate the communication, lasting for a predefined amount of time; if this cannot be authenticated the call is stopped. The event, on the other hand, becomes the topic. Whenever the endpoint receives signed data from the chat service correctly, the gateway has to respond immediately with an HTTP status code; if everything went smoothly, it generates a code 200 (OK), 201 (created) or 202 (accepted). However, if the data is not signed correctly or even the signature is missing, it responds with a 403 code (forbidden) and does not provide the broker with the needed data. To protect transactions between servers, it is convenient to use SSL / TLS.

This described architecture is integrated in the Lysis IoT platform so that the outcome of the functions is taken by the VU to perform the resulting tasks. The following are the different needed functions: Message Handler; Action Service; Save Conversation; Send Message; Failure Handler; and Save Logs. These are briefly presented in the following.

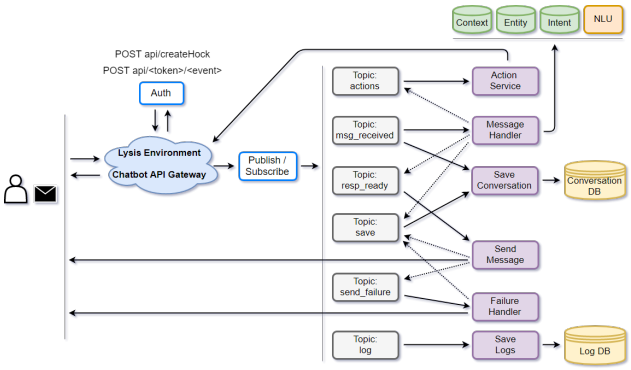


Fig. 3 – Process flow for the whole system

4.1 Message Handler

If the gateway gives the green light, it stores the input in an event queue and then processes it as soon as possible. The various topics provided allow for calling the related functions. In the case of a request generated by the user, then the topic will be *msg_ received* which calls the Message Handler function; this is the only function that can be called directly from the gateway interfaces. This function processes the request and creates the output to be provided to the user. The construction of the output takes place in various stages and on the basis of the user's request.

As a representative case, let us consider the user that wants to enable remote surveillance of her home (at the moment this functionality is off). The user sends the following text "Set video On"; in order to understand what to do with this command, the message must be broken down and analyzed. From here, therefore, the function has to understand the entity, the user intent and the context using an NLU (Natural Language Understanding) algorithm. Obviously, in order for the analysis to be relevant to what is requested in the messages, the system must be able to store and analyze the status otherwise, if this was not the case, the responses and actions taken would not be relevant to the general context of the conversation. On the basis of the results obtained then, if an intervention or a reading is required in a given device, you will have to be able to invoke, through an appropriate topic, a function that will implement or request what the user needs and then return it to the Message Handler. Once all the data has been obtained, it is possible to create the reply and make it available for sending.

4.2 Action Service

When the Massage Handler function needs data that is on the platform, it must be able to retrieve it. The simplest way is to rely on a second function with this objective. The Action Service function takes care of retrieving the requested data. Once the Message Handler has processed the request and on the basis of the NLU algorithm has understood the actions to be taken, it activates the Action Service which fetches to the platform requesting data or making settings. In the event of an error, the identification code will be returned.

4.3 Save Conversation

This function is invoked when the save event occurs. This can be invoked by the Message Handler, Send Handler, and Failure Handler functions. In the first case, as soon as the message is received, this (in addition to being taken over by the Message Handler function) also passes through the function in question which will create a record containing the request and the status of the conversation. The second case is similar to the first but now it takes care of saving the response produced by the Message Handler function; however, if the sending fails, the save event cannot be invoked and *send_failure* will be invoked in its place. The last case is equal to the second except for the fact that now failing or not, a record is still created which will be the message produced in the case of success or an error message in the case of failure. Each time you save the conversation, the status is also saved.

4.4 Send Message

The sending function is the one that takes care of forwarding the response to the user through the chosen messaging service. It could be for example a telegram rather than a proprietary application created ad hoc.

Whatever the application chosen, this function first of all sets up the WebHook and then, if the setting is successful, sends the response produced to the user; if the setting fails, it contacts the Failure Handler function to manage the mistakes. A best practice is to use separate functions and topics for receiving, error handling and sending. This way there won't be problems in contacting the correct endpoints; accordingly, operations such as save and retry won't be taken over by this function.

4.5 Failure Handler

If the topic becomes *send_failure*, it means that there was a problem sending data to the user. To manage these types of problems there is the need to rely on a special function. When the Send Message function fails the first attempt, it contacts the Failure Handler function passing the message and the error code returned by the attempted send. The function is encoded in order to retry the sending for a certain number of times after which the user will be notified that there is a problem in satisfying their request. If, on the other hand, the sending is completed within the established number of attempts, the message is delivered in a totally transparent way to the user.

4.6 Save Logs

This is used to archive all messages related to the system in general; in this way it is possible to monitor the flow and see if there are any problems or if some parts need some actions to be performed. This is a feature that can be implemented by relying on the logging of activities of the cloud platform. In addition, some platforms such as AWS or Google Cloud Platform allow for saving in the log file, in addition to the default entries, new fields at the user's discretion. In so doing, by integrating the system logs with those of the requests to the bot, it is possible to have complete and detailed logs.

5. SCENARIO

The IoT Lysis platform currently does not provide any help for the user either with regard to the deployment of SVO or with regard to the resolution of any problems such as failures, unresponsive devices and so on. To simplify the user-platform-SVO interaction, the intention is to insert a bot within the platform that guides the user in carrying out those activities that are currently cumbersome or even impossible to perform remotely:

- SVO deployment
- Problem resolution
- Inquire of devices
- Setting
- Task automation

To create the bot, the Google Cloud platform was chosen, in which all the back end and the various functions are hosted, alongside the DialogFlow service provided by Google that offers a retrieval-model-based technique for matching responses with the aid of machine algorithms learning, where the latter can be enabled at the user's discretion. This function, if enabled, allows us to have some flexibility in the interpretation of the user's requests as the answers are given based on the best score obtained from a classification prior to the choice of the answer. In this way, therefore, it is possible to manage any spelling or form problems that might cause errors in recognizing the correct intent for the request made.

The proposed solution allows for a dynamic composition of the services that can be provided, given the ability of the bot to query any SVO owned by the user present on the platform. When the user queries the chatbot, they will be offered various choices and based on the SVOs that are selected, a service is composed with only the choices made by the user. For example, in the car, in addition to the SVOs relating to the car, you may also need SVOs relating to other environments, for the purpose of continuous monitoring, the service offered by the bot therefore includes the data from these SVOs. In addition, you would also have the possibility to save them and re-propose them at a later time as favorite services. Fig. 4 shows the sequence diagram which illustrates the simple steps that take place when a request is sent to the bot.

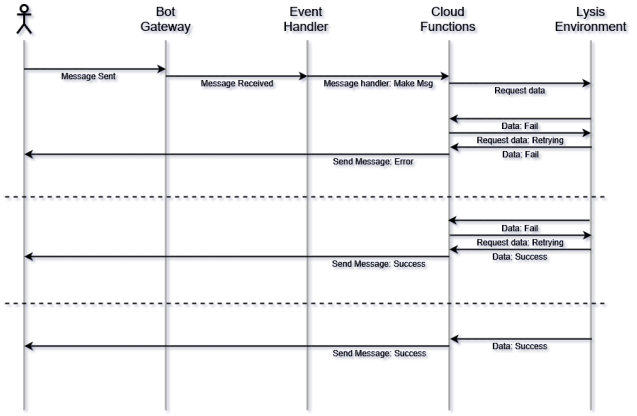


Fig. 4 – Use case diagram

When the user accesses the chatbot interface, they are presented with the various options. Once the desired option is selected, the bot will send a message to the bot gateway, who will take care of handling the request, labeling it and sorting it to an Event Handler (EH).

The Message Handler will recognize that a message has arrived and delivers it to a cloud function that takes care of the part of creating the response message. Then, it has to collect the data in addition to the textual answers by querying the platform that contains the SVOs necessary for the composition of the answer. At this moment, we may have two different scenarios that we analyze below:

- The data request fails
- The data request is achieved

We can see in Fig. 5 the workflow of requests in a possible user interface. All these interventions are immediate, the most expensive response in terms of timing is the one in which the data is requested, in this case the video stream. But in principle, the time between a send-reply is given by the user interaction time with the device plus the delay introduced by the bot to reply, which is a maximum of a few seconds.

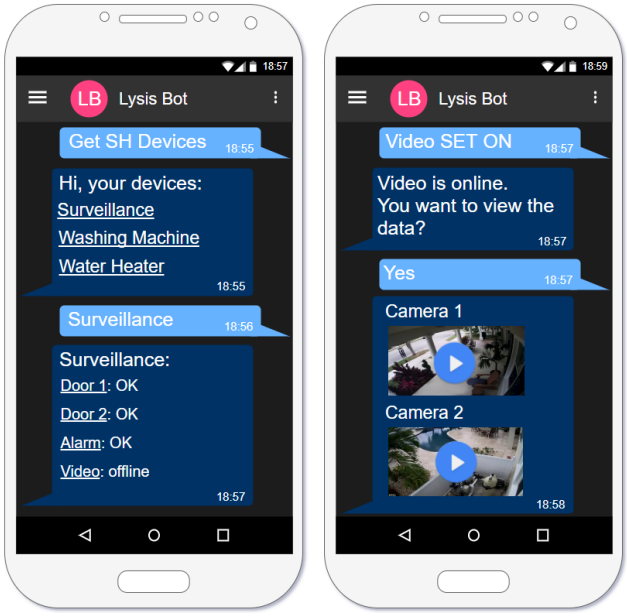


Fig. 5 – Representative flow of messages exchanged between the user and the bot

We have implemented the chatbot system that allows for interacting with the platform and for setting and sending requests to the devices. The queries to the bot are made in natural language and are taken over and processed by the DialogFlow platform, with has been integrated in our system. The messaging system selected has been the Telegram messaging client which is used by the user.

The development of the bot was divided into two parts: design and development of all the components necessary for the NLU functionalities; development of the gateway and the functions necessary to handle the events and associated data in the chatbot platform. An agent has been created within the DialogFlow platform. Agents are NLU modules that deal with transforming user requests, expressed in natural language, into *usable* data, i.e., data that can be associated to actions to be activated. To ensure that the requests are interpreted correctly, all the possible *intents* and *entities* have been loaded into the agent. Intents are JSON files and have been designed and built in order to map the user's requests with the actions to be performed in the best possible way. In order to have the best match between request and intent, new entities (all synonyms for a given word are associated to an intent) have been developed, in addition to those made available by the platform, which were able to best characterize the IoT context

of use we needed. Therefore, various attributes have been created within each entity, with their synonyms, in order to be able to extract the values of the input parameters without these being necessarily identical to those we had foreseen in the phase of creating the intents. In this way, every time an intent is activated, the platform will return a JSON file with the information on: Intents; Action; Event; Response; Contexts; Parameters; Score.

At each request, these parameters are updated based on the intent that is activated at that time so that, at the next call, the new intent to be activated is also chosen based on the previous parameters. In this way, it is possible to completely contextualize the conversation. In fact, the contexts section contains all the active contexts in that call ordered according to their lifetime. With this mechanism it was therefore possible to implement a management of the state of the bot allowing for the exchange of variables between subsequent requests.

6. IMPLEMENTATION

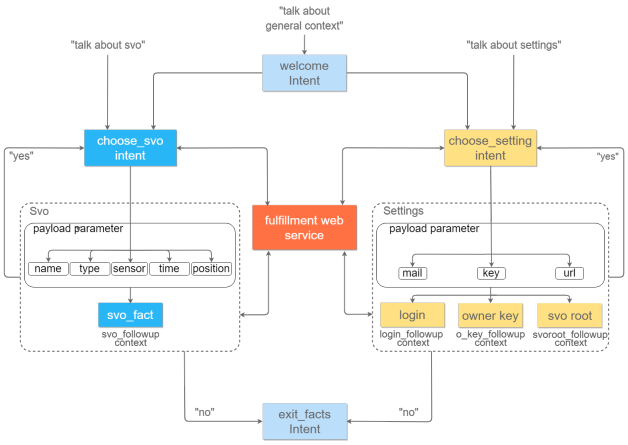


Fig. 6 – Flow diagram of the bot

As we can see in the diagram in Fig. 6, it is possible to enter one of the intents based on what is asked. Let's refer to our use case previously described related to the surveillance. As we said, the user has to ask the bot for sending the video stream from the video camera device; if it is turned off, it will first be asked to turn on and then send the data. When the user needs to use the bot, they have to log in and then send the message "show me what happens at home". This activates the "svo_facts" intent through the "svo_facts" event and setting "svo _followup context" with a certain lifetime set as the current context. The bot then responds by giving the list of objects that are indexed in their home. By selecting the video surveillance, the user remains within the "svo_followup context" and then activates the loop indicated with "yes" in the diagram; by reactivating the svo_facts intent, the context is updated again and the bot responds by displaying the state of the object. On the basis of this, then it allows for the choice whether to activate it or not. Once concluded, if the user decides to perform different actions to the question "do you want to do other operations?" answering "no", they activate the reset of the contexts and the "exit_facts" intent, initializing the bot for new requests.

7. EXPERIMENTAL RESULTS

The experiments have been conducted to assess how effective the chatbot was in understanding the user requests and perform action accordingly. In the following section we describe the performed tests with reference to the access to the platform and perform device setting and request data of interest. The performance of query matching results has been also analysed.

7.1 Access to the platform

Fig. 7 shows the interaction with the chatbot with the intent of accessing the Lysis platform. The figure shows the flow of questions and answers between the bot and the user. Remembering that it is necessary to authenticate, to be able to use both the chatbot services and to have access to the resources made available by the platform, the first question asked was on how it was possible to authenticate. The bot's response was a message with the instructions on how to log in and, once logged in, it suggested to the user that it was necessary to enter some additional information to complete the configuration. The user then asked how to enter the owner key and the SVO root, to which the bot answered by providing a description of where to find them and information on how to enter this information. This was possible thanks to the fact that when you ask for information either on the key or on the root SVO, the respective context is activated allowing you to keep track of what was previously requested. After completing the configuration and logging in again, you can see that the welcome message is simply given, a sign that the configuration was successful.

7.2 Setting the devices and data retrieval

We also tested the ability to request data from the platform and apply the desired settings to the available devices, all with the most natural language possible. Fig. 8 shows how it was simple to request the list of devices and their current status. If you want to switch a device on or off, simply specify which of these actions should be applied and the setting will be performed. The request for data was also handled in a similar way; in the sentence it is just needed to specify which data is needed and from which environment to obtain the requested data.

7.3 Elaboration time and latency evaluation

The time spent processing the information sent to the chatbot was very low. This happens thanks to the use of an ML engine that is fully running in external services and for the efficient implementation of the sample questions on the platform. This allows for a low latency between a request and the response and this makes the user

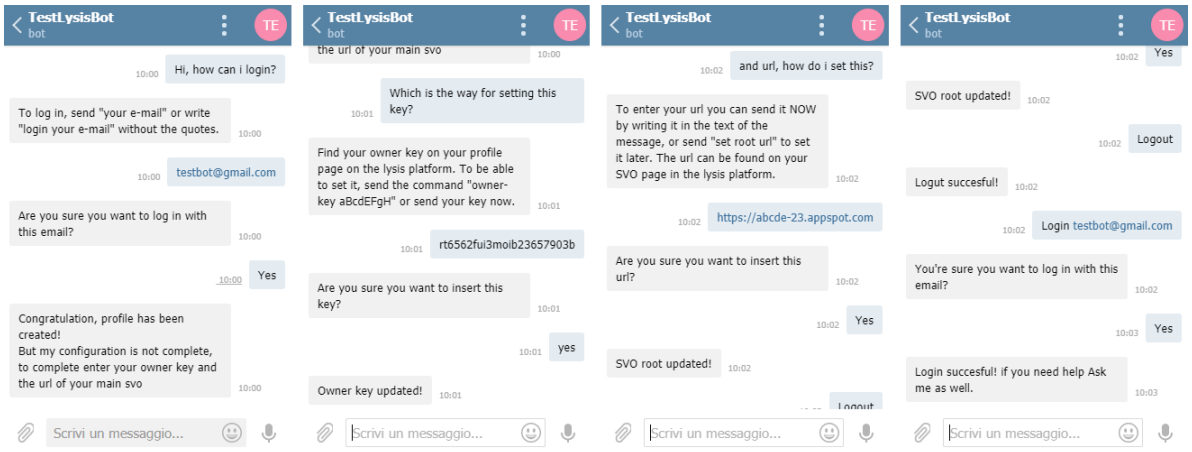


Fig. 7 – Bot tests: access to the platform

experience a smooth interaction. For instance, using a virtual machine instance with 1vCPU and 512 Mb of RAM, the latency values of the system to process and provide a response for a single request are between 100 ms and 200 ms, to which the delay introduced by the network should be added. This brings to an overall round trip time of less than a second. Furthermore, thanks to the ductility of the serverless system, by appropriately configuring the load balancing rules, when all the service instances are occupied a new instance can be started to automatically lighten the load of others.

7.4 Analysis of the questions matching scores

We have also analyzed the relevance of the questions submitted to the bot with the patterns inserted in the intents, created on the Dialogflow platform. The score calculated by DialogFlow was used for this purpose. This evaluates the level of confidence of the question submitted to the bot with the example ones present in the platform. This confidence level is calculated based on the state of the conversation and exploiting the Term Reinforcement techniques. These techniques allow for a greater weight to certain words through their repetition or the use of synonyms. Score values range from 0.0 (completely uncertain) to 1.0 (completely certain). In the proposed implementation, once a question is evaluated, there are two possible outcomes: a) if the question achieves a confidence match score greater than or equal to the classification threshold setting, the higher confidence intent is triggered; b) if no intent meets the threshold, no match is returned. In this case the threshold was set to 0.7. The score plotted in Fig. 9 and Fig. 10 indicates the quality of the match between the ideal question (the one contained in the intent) with the real question (the one generated by the user). Obviously, the sentences inserted within the intent are constructed, with the help of the entities, in such a way so as to be as general as possible, so they are not strictly meaningful sentences but rather they are composed only of the words actually necessary to give a meaning to the sentence so as to be able to guarantee the best match even with requests that are not well formulated, albeit with a lower score than the optimal one.

In Fig. 9 we can see the scores of the flow of requests that have been submitted to the bot during the configuration phase in two distinct cases. The first case, called “Best”, was produced by submitting to the bot the sentences formulated as similar as possible to how they were inserted into the intents, trying to make them as close as possible to natural language. The second case, called “Worst”, on the other hand was formulated using the synonym of the keywords and looking for a grammatical form quite different from the one used in the previous case. Similarly, in Fig. 10, the same analysis was performed for the second test, where device setting and data request were performed. Sentences 4 and 7 in 9, sentences 4 and 10 in 10 are cases in which the match between sentences is not accurate. This phenomenon is governed both by the number of synonyms that have been associated with the entities, and by the level of similarity between the various intents implemented and their length. For example, if you have two intents that trigger two different events but are very similar in natural language, the classifier will be less accurate about which one to choose. In Table 1 we also show the average values which demonstrate that there is not a big difference between the “Best” and “Worst” cases; indeed, in both cases it was possible to configure the bot, request data and set the devices smoothly without any issue about possible request misunderstanding. Obviously, the better the intents are constructed, the easier it will be to get accurate matches by submitting questions that are apparently different but express the same concept.

Table 1 – Comparison between the average values of the scores obtained for the two considered scenarios

	$\overline{S}_{Best}$	$\overline{S}_{Worst}$
Platform access	0.956	0.844
Device configuration	0.925	0.819

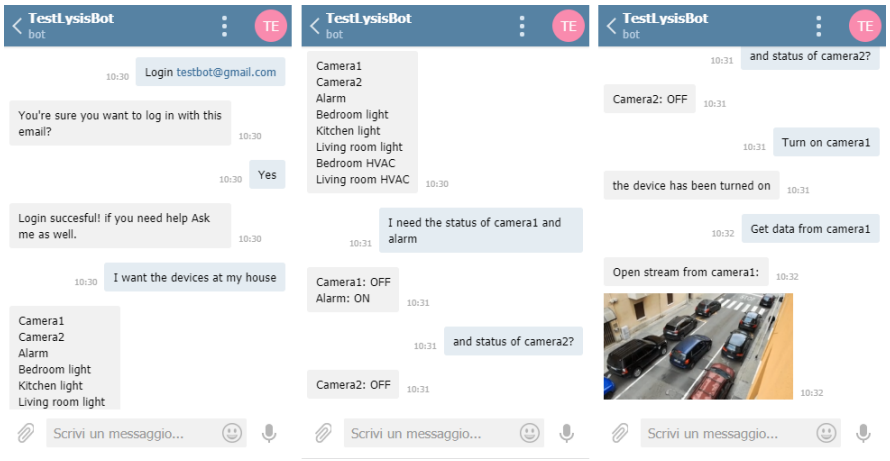


Fig. 8 – Bot tests: request of data and setting of the device

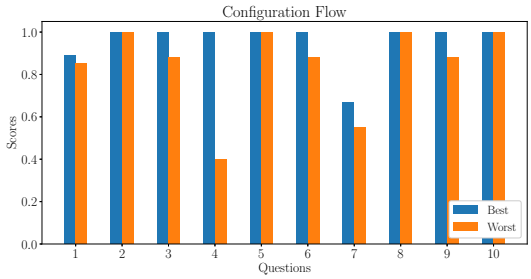


Fig. 9 – Scores that have been obtained by matching the queries with the intent during the platform accessing activities

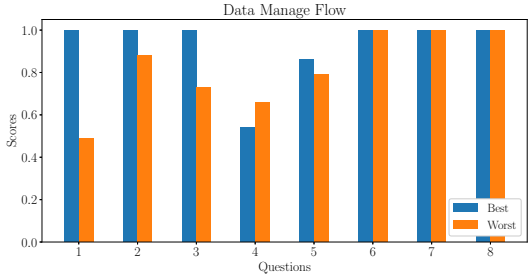


Fig. 10 – Scores that have been obtained by matching the queries with the intent during the setting of the devices and data retrieval

8. CONCLUSIONS

This study has investigated the possibility of integrating a virtual assistant, developed in the form of a chatbot, within an IoT platform to help and guide the user to easily carry out the various operations that would otherwise be cumbersome and sometimes complicated. This need, as we know, derives from the fact that the configurations and requests for data, for an inexperienced user, are not immediate but may require various steps to be completed and may be frustrating.

A bot has been then developed which, thanks to a natural language understanding engine, is able to process the user's requests formulated in a natural language. The bot essentially works as a mediator between the real world and the virtual world. In the experiments that have been carried out it has been possible to see how simple it is to configure the bot and use it to interact naturally with the IoT platform. We specifically focused on platform access, device setting and data request. It has to be said that for these experiments the operations carried out were simple but still encouraging for future developments. One of the most interesting actions is certainly the ability to deploy applications quickly and easily as well as being able to use the bot as a guide for troubleshooting, knowing in real time if the various devices are faulty or malfunctioning, so as to restart them automatically.

ACKNOWLEDGEMENT

This work has been partially found by the POR FESR Sardegna 2014 with the project Farmainforma (RICERCA_1C-38).

REFERENCES

[1] i. Gartner. 2020. URL: https://www.gartner.com/smarterwithgartner/top-cx-trends-for-cios-to-watch/#:~:text=Chatbots%2C%20virtual%20assistants%20and%20robots,up%20from%2015%25%20in%202018..

[2] G. Neff and P. Nagy. "Automation, Algorithms, and Politics Talking to Bots: Symbiotic Agency and the Case of Tay". In: *International Journal of Communication* 10.0 (2016).

[3] J. Weizenbaum. "ELIZA—a Computer Program for the Study of Natural Language Communication between Man and Machine". In: *Commun. ACM* 9.1 (1966), pp. 36–45.

[4] R. Wallace. "The anatomy of A.L.I.C.E". In: 2009, pp. 181–210.

[5] R. Kar and R. Haldar. "Applying chatbots to the internet of things: Opportunities and architectural elements". In: *arXiv preprint arXiv:1611.03799* (2016).

[6] S. Wiangsamut, P. Chomphuwiset, and S. Khummanee. "Chatting with Plants (Orchids) in Automated Smart Farming using IoT, Fuzzy Logic and Chatbot". In: *Advances in Science, Technology and Engineering Systems Journal* 4 (2019), pp. 163–173.

[7] M. H. A. Fadzil and D. Ab Kadir. "Plant Monitoring with Artificial Intelligence Chatbot". In: *Journal of Computing Technologies and Creative Content (JTec)* 5.2 (2020), pp. 34–38.

[8] M. U. H. Al Rasyid et al. "Integration of IoT and chatbot for aquaculture with natural language processing". In: *Telkomnika (Telecommunication Comput. Electron. Control* 18.2 (2020), pp. 640–648.

[9] G. Alexakis et al. "Control of smart home operations using natural language processing, voice recognition and IoT technologies in a multi-tier architecture". In: *Designs* 3.3 (2019), p. 32.

[10] S. Ahmed et al. "Smart Home Shield and Automation System Using Facebook Messenger Chatbot". In: *2020 IEEE Region 10 Symposium (TENSYMP)*. IEEE. 2020, pp. 1791–1794.

[11] S. Mahajan et al. "Design and implementation of IoT-enabled personal air quality assistant on instant messenger". In: *Proceedings of the 10th International Conference on Management of Digital EcoSystems*. 2018, pp. 165–170.

[12] J. E. P. Reddy et al. "AI-IoT based Healthcare Prognosis Interactive System". In: *2020 IEEE International Conference for Innovation in Technology (INOCON)*. IEEE. 2020, pp. 1–5.

[13] K. Sivaraj et al. "Medibot: End to end voice based AI medical chatbot with a smart watch". In: 9 (2021), pp. 201–206.

[14] C Balasubramaniam, S Velmurugan, and M Saravanan. "DESIGN AND DEVELOPMENT OFSMART HEALTHCARE CHATBOT APPLICATION USING AI-ML". In: *Journal of Natural Remedies* 21.7 (S1) (2020), pp. 13–20.

[15] L. Atzori, A. Iera, and G. Morabito. "SIoT: Giving a Social Structure to the Internet of Things". In: *Communications Letters, IEEE* 15 (2011).

[16] R. Girau, S. Martis, and L. Atzori. "Lysis: a platform for IoT distributed applications over socially connected objects". In: *IEEE Internet of Things Journal* PP.99 (2016), pp. 1–1.

[17] R. Girau et al. "Virtual User in the IoT: Definition, Technologies and Experiments". In: *Sensors* 19.20 (2019), p. 4489.

AUTHORS

Raimondo Cossu is a telecommunications engineer. As soon as he graduated he did an internship at Avanade SRL where he acquired IT skills. After the internship he was hired as a collaborator in the MCLab DIEE laboratory at the University of Cagliari, where he currently works. His field of research and development is focused on the Internet of Things, cloud computing and distributed systems. Currently he is also CTO in WiData SRL, a company that deals with data analysis.

Roberto Girau is a research fellow at University of Bologna, Department of Computer Science and Engineering since 2021. He received an M.S. degree in telecommunication engineering and his Ph.D. degree in electronic engineering and computer science from the University of Cagliari, Italy in 2012 and in 2017, respectively.
From 2012 to 2020, he worked as researcher at the Department of Electrical and Electronic Engineering of the University of Cagliari, developing an experimental platform for the social Internet of Things.
His main research areas of interest are IoT with particular emphasis on its integration with social networks, software engineering, smart cities and cloud computing.

Luigi Atzori is Full Professor at the Department of Electrical and Electronic Engineering, University of Cagliari (Italy) and Research Associate at the Multimedia Communications Laboratory of CNIT (Consorzio Nazionale Inter-universitario per le Telecomunicazioni). His research interests are in multimedia communications and computer networking, with emphasis on multimedia QoE, multimedia streaming, NGN service management, service management in wireless sensor networks, architecture and services in the Internet of Things. He has been the associate and guest editor for several journals, included: ACM/Springer Wireless Networks Journal, IEEE IoT journal, IEEE Comm. Magazine, the Springer Monet Journal, Elsevier Ad Hoc Networks, and the Elsevier Signal Processing: Image Communications Journal.

RESOURCE TOKENIZATION FOR CROWDFUNDING OF WIRELESS NETWORKS

Dr. Volkan Sevindik
Digimetrik LLC., 11654 Plaza America Dr., Virginia, 20194, USA

NOTE: Corresponding author: vsevindik@gmail.com

Abstract *– This paper presents a novel blockchain-based spectrum tokenization method used to crowdsource wireless network deployment projects. Crowdsourcing is a method of financing certain projects and ideas through the funds collected by individuals or businesses in an open marketplace. The method presented in this paper finances the wireless network deployment projects belonging to service providers or governments. The method tokenizes proposed novel wireless resource units, and sells these units to investors. A new Value Unit Per User (VUPU) resource unit is introduced with a new pricing scheme depending on a load of a base station. A novel Proof of Data Load (PoDLO) consensus algorithm is proposed which is used to verify data and traffic load of a base station. Device Diversity Factor (DDF) and Subscriber Unique Permanent Identifier (SUPI) Factor (SUF) are proposed new ways to determine the value of a base station and a network cluster.*

Keywords – Blockchain, crowdfunding, resource tokenization

1. INTRODUCTION

As a telecommunication industry, we are at the verge of deploying 5G wireless networks. With increased Spectral Efficiency (SE) gains promised by new wireless standards, 5G/NR standard also promises delivering higher SE compared to 4G/LTE. Higher SE enables the transmission of more data to subscribers, increasing revenue for service providers. During network design, service providers calculate how much capacity is needed at certain markets, and coverging the whole market with superior data speeds is the key to success. With mmWave spectrum and with above 6 GHz spectrum bands, reaching to far distances is very challenging, if not impossible. Therefore, wireless operators started to question the investment in 5G technology. Unlimited data plans, decreasing outdoor data usage, and increased competition will force wireless service providers to postpone their infrastructure investments for future wireless network technologies. In the near future, we will see a plateau on wireless data speeds, and innovation because of decreasing network investment.

The other side of the coin is that retail investors or individuals look for more tangible assets to invest. Therefore, a method of investing wireless communication systems' infrastructure can be appealing to the general public. However, wireless network resources should be defined and divided into investable pieces. Tokenization is an important first step of converting a wireless network infrastructure into investable pieces. The open research question is how to tokenize wireless resources so that it provides capital for network investment, and it also provides income to its investors with a minimum risk.

The main purpose of this paper is to share research findings on blockchain-based crowdsourcing platforms. For that purpose, an automated network tokenization method with blockchain is proposed. Section 2 defines novel wireless network resources and a tokenization method, Section 3 defines blockchain and smart contracts used in small-cell blockchain, Section 4 presents newly introduced the Proof of Data Load (PoDLO) consensus algorithm and defines a block creation rate formula, and Section 5 concludes the paper with important next steps.

2. WIRELESS NETWORK RESOURCES

2.1 Network architecture

Fig. 1 shows 5G network architecture and network interfaces [1]. A base station is connected to a 5G core network via an N2 interface, and this interface carries all control information from the base station to core network. N3 and N6 interfaces are used to carry user information. When a base station is deployed at a particular location, a backhaul connection is needed to connect to the core network [1].

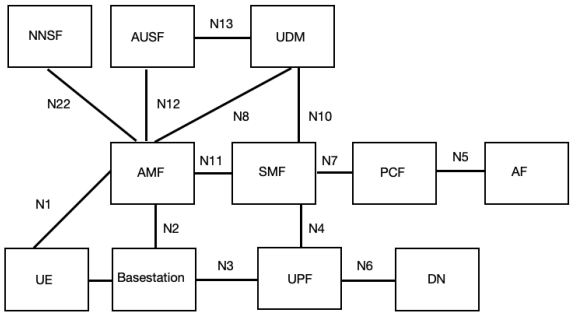


Fig. 1 – 5G network architecture and network interfaces

A backhaul connection can be any type of wireline or wireless connection.

2.2 Wireless network resource utilization

2.2.1 Data tonnage resources

Spectrum is the most important resource in a wireless network. In this paper, spectrum refers to licensed spectrum. And licensed spectrum is more valuable than unlicensed spectrum. Spectrum is divided into a number of physical resource blocks in 4G/LTE and 5N/NR standards [1],[3-6]. However, depending on the Modulation and Coding Scheme (MCS) selected, the amount of bits that can be carried by one Physical Resource Block (PRB) can change significantly. A resource block is defined as 12 consecutive OFDM symbols in frequency in both 4G and 5G standards. One OFDM symbol occupies 15 kHz in frequency in 4G/LTE and this is the same for 5G/NR with 15 kHz subcarrier spacing. 5G/NR supports multiple OFDM subcarrier spacings. In 5G/NR, 15, 30, 60, 120 kHz Subcarrier Spacings (SCS) are supported. Thus, one resource block occupies 12 x 15 kHz = 180 kHz of spectrum, and 10 resource block occupies 1.8 MHz of spectrum in a frequency domain.

$$N_{PRB,U} = N^{1}_{PRB,U} \cdot \frac{(SA\,.\,SSO)}{(12\,.15)\,.\,SC_F\,.\,(N_{SU})} \quad (1)$$

where $N_{PRB,U}$ represents number of PRBs per subscriber, $N^{1}_{PRB,U}$ is the number of PRBs of subscriber U in 1 msec., SA means total spectrum amount, SSO stands for symbol spacing overhead, SC_F is the subcarrier factor for 5G and N_{SU} is the number of simultaneous users. SC_F takes values of 1, 2, 4, or 8 for 15, 30, 60, 120 kHz SCSs respectively.

$$T_U = N_{PRB,U}\,.N_{OFDM}\,.N_{bits}\,.M_U \quad (2)$$

where T_U means tonnage per subscriber in megabytes (MBs), $N_{PRB,U}$ is the number of PRBs/ subscriber, N_{OFDM} is the number of OFDM symbols per PRB, N_{bits} is the number of bits per ODFM symbol, M_U is the number of connected minutes per user. SSO = 0.9, Number of RBs/1msec. = 2, Number of Simultaneous Users = 4, Number of Connected Minutes/User in an hour, which is assumed to be 3 minutes/hour, or 180 seconds/hr. In commercial wireless networks, the simultaneous number of subscriber per Transmission Time Interval (TTI) changes between 2 and 4 for 4G and between 4 and 8 for 5G. In this article, we assume there are 4 simultaneous subscribers per TTI. And the number of available PRBs is divided by 4 at each TTI.

189 MBs is the unit resource of data tonnage in a wireless network, which is called Value Unit Per User (VUPU). The number of VUPUs will increase when a wireless service provider deploys a network with high-capacity base stations. A wireless provider owning 40 MHz of spectrum, and hundreds of base stations at a particular region will afford more VUPUs than a wireless provider owning 20 MHz of spectrum and a smaller number of base stations. VUPUs will be sold to investors, and this will be the first tokenized unit. The concept of mining using base stations is described in [13]. A base station keeps the record of successfully created/delivered VUPU to each user terminal. And this information is recorded as part of a block in a blockchain held by the base station. Each user is assumed to consume 150 units of VUPU, and these units are tradable. The higher the number of subscribers, the more VUPUs, and the more investment that wireless service provider will have. This will leave more capital for investment, and will motivate service providers to deploy more base stations. Current unlimited data plans cost, on average, 80$ per 20 GBs of data delivered to a subscriber. This equals 80$/20 GB = 4$/1 GB = 0.4$/100 MB, and this is called Revenue per User (RPU).

$$BET = max\left\{\left(\sum_{i=180k}^{T=180(k+1)}(RPU_i\,.\,N)\right) - CoD - CoP, 0\right\} \quad (3)$$

where BET is the *Break Even Time* of a base station from a financial perspective, T is the time that base station is live and in an operational state, k = 0,1,2, M. N is the average number of connected users per base station during time duration of {180k, 180(k+1)}, *CoD* refers to *Cost of Base Station Deployment*, CoP refers to *Cost of Operation*, and

RPU is revenue per connected user, which is calculated at every 15 minutes of connection time. If we assume that one base station capacity is 600 connected users, then total revenue per base station will be 600 Connected Users x 60$. And this calculation assumes that users will stay connected for 24 hours per day and 30 days per month.

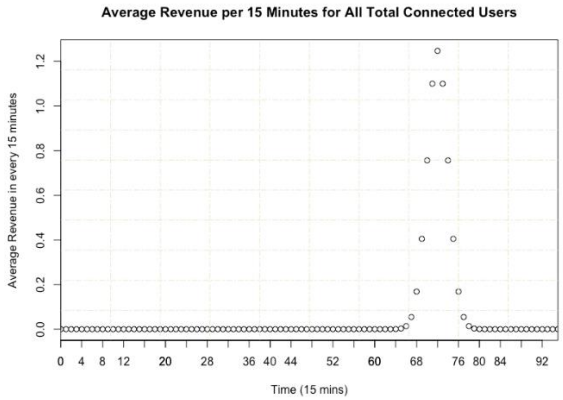


Fig. 2 – Average revenue at each 15 minutes

Fig. 2 shows one sample distribution at busy hours of a day. Fig. 3 shows the revenue of a base station for every 15-minute interval in a day, and one day consists of 96 15-minute intervals. Fig. 5 shows the average revenue per day of a base station.

Fig. 6 shows the average revenue per day for randomly selected three base stations in a network cluster. When a base station is in operational state for a long time, it will be financially faster to break-even and to reach positive revenue.

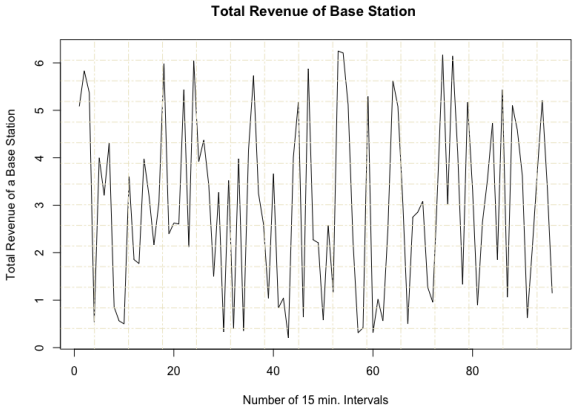


Fig. 3 – Average revenue at each 15 minutes

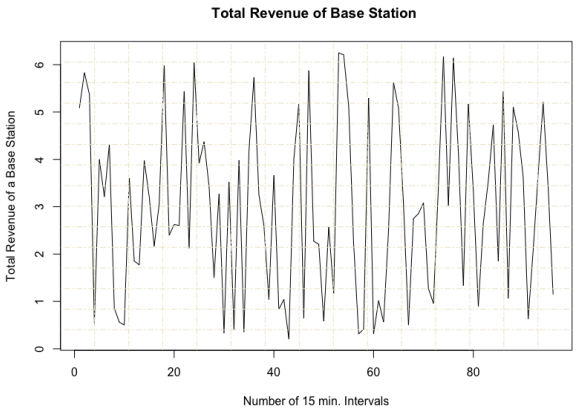


Fig. 4 – Total revenue of base station

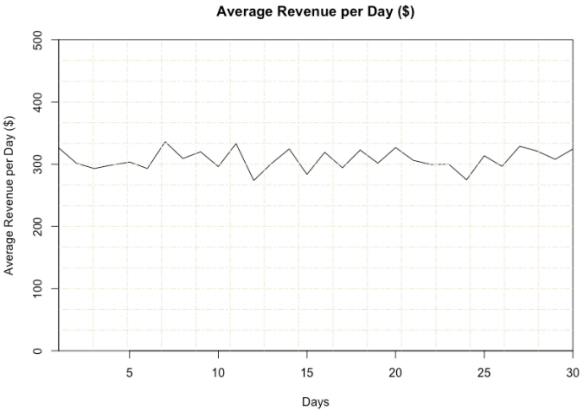


Fig. 5 – Average revenue per day

VUPU investment will start when BET ≠ 0, and the following formula is used to determine the price of one VUPU:

$$P(T)_{\text{VUPU}} = \frac{\sum_{n=1}^{B}\left(\sum_{i=180k}^{T=180(k+1)}\left(RPU_{i,n}\,.\,N_n\right)\right)+\max\{TI,0\}}{S} \quad (4)$$

where $P(T)_{\text{VUPU}}$ is the price of VUPU at time T, and k value ranges from 0 to 959. S is the number of shares issued by the wireless service provider, B is the number of base stations in a certain cluster, or in a certain geographical region.

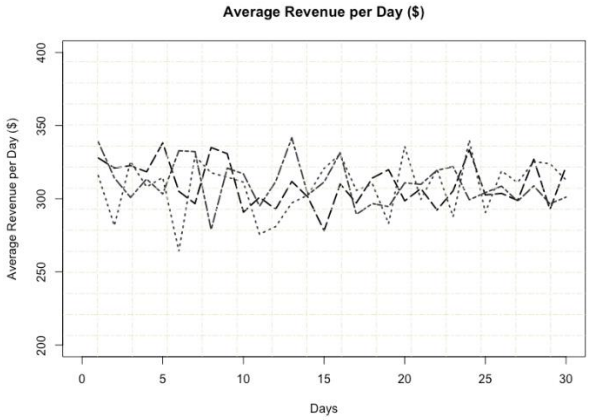


Fig. 6 – Average revenue per day of different base stations

TI represents the total investment in the same network cluster or in the same geographical region. Since the VUPU price is time dependent, price is updated with the following formula:

$$P(T)_{\text{VUPU}} = \left(1-\frac{1}{C}\right)P(T-m)_{\text{VUPU}} + \left(\frac{1}{C}\right)P(T)_{\text{VUPU}} \quad (5)$$

where C is the constant used to update the price. With lower C values, a recent calculated price has more impact on the average price, and vice versa. With a newly deployed network cluster, big C values are selected in order to converge to a certain price.

2.2.2 Mobile device diversity factor

A base station serves different types of terminals depending on the time and location. A more diverse set of terminals increases the value of the base station, and augments the value of theVUPU. The following formula is used to summarize the impact of terminal diversity:

$$DDF_n = 1 + \sum_{TT=2}^{Z}\left(\frac{1}{TT}\right) \quad (6)$$

where DDF_n is the device diversity factor of base station *n, TT* is number of different traffic types served by the base station in a 24-hr window. For example, if the base station serves only one traffic type, DDF_n = 1; and if the base station serves one different traffic type in addition to mobile traffic, then $DDF_n = 1 + 0.5 = 1.5$. Fig. 7 shows DDF values corresponding to different traffic types.

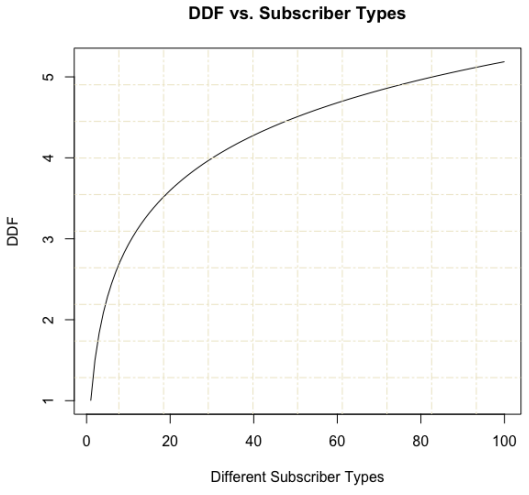


Fig. 7 – Increasing DDF with increading subscriber types

The formula is updated as follows:

$$P(T)_{\text{VUPU}} = \frac{\sum_{n=1}^{B}\left(DDF_n \cdot \left(\sum_{i=180k}^{T=180(k+1)}\left(RPU_{i,n}\cdot N_n\right)\right)\right)+\max\{TI,0\}}{S}. \quad (7)$$

2.2.3 Subscriber Unique Permanent Identifier (SUPI) Factor (SUF)

SUF_n means the number of different SUPIs (5G), IMSIs (4G) that are recorded by base station n. It is defined as follows:

$$SUF_n = C + \log(N_{SUPI}). \quad (8)$$

If SUF_n = 1, this means the same users connect to the same base station n. For modeling the SUPI factor, any logaritmic function can be used. The reason for using the logarithmic function is that the increasing number of diversity in the network will have lower impact on the value since the base station will be overloaded after some point leading the total service quality degradation for all terminals connected to the base station. In a commercial network, user capacity per base station ranges between 600 and 1200 connected users for 5G base stations. And, with SUF = 1, there are 600 different SUPIs recorded by the base station. If SUF = 2, this means there are 1200 unique SUPIs recorded by the base station. With SUF, the new formula for the price of VUPU will be

$$P(T)_{\text{VUPU}} = \frac{\sum_{n=1}^{B}\left(DDF_n \cdot SUF_n \cdot \left(\sum_{i=180k}^{T=180(k+1)}\left(RPU_{i,n}\cdot N_n\right)\right)\right)}{S^{\left(\frac{1}{DDF\cdot SUF}\right)}} + \frac{\max\{TI,0\}}{S^{\left(\frac{1}{DDF\cdot SUF}\right)}}. \quad (9)$$

The DDF value curve can have any shape depending on the number of hand-offs. Fig. 8 shows the DDF of a base station serving both mobile and nomadic users. Since subscribers/users are mobile, subscribers/users will hand off from one base station to another base station during the day. When users are stationary or nomadic, depending on the location of the user, Wifi might be available. Thus, users will use Wifi connection especially on these occasions. This shows that the real value of a cellular communication system is to provide high data speeds during mobility or in other words, to provide mobile data to mobile users.

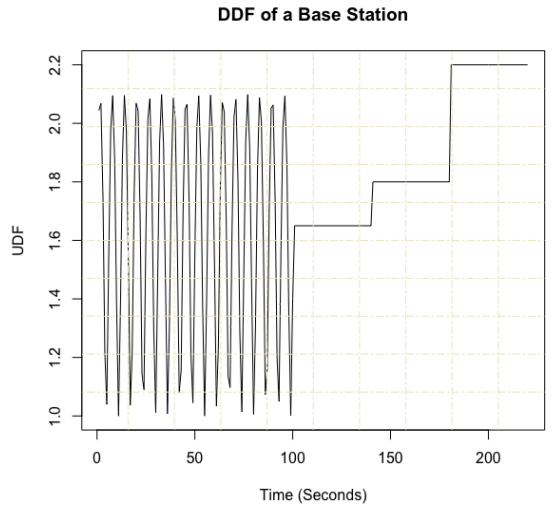


Fig. 8 – DDF of a base station based on mobility

Fig. 9 shows real wireless network deployment. Base station 1 and base station 2 are co-located, and they provide coverage using different operational spectrum. This model is called the 'overlay' coverage method, where one base station uses a lower frequency band to provide greater coverage than a co-located base station using a mid or high frequency band. The higher the mobility in the network, the higher the value that network delivers to its subscribers. Hence, the DDF factor is included in the pricing model since the network will be more valuable if there are a high number of mobile users or highly mobile users in the network.

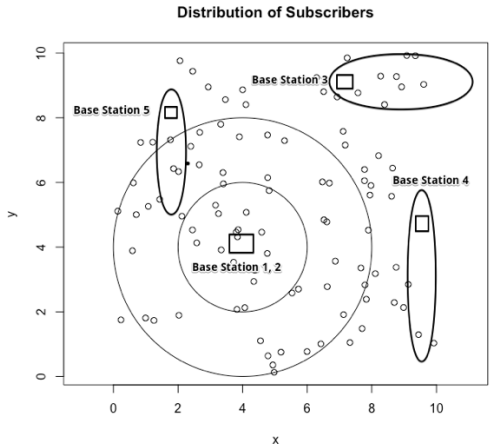


Fig. 9 – Distribution of subscribers In a real network cluster

3. BLOCKCHAIN

3.1 What is blockchain?

Blockchain is a distributed ledger technology that records transactions securely in an untrusted system [2], [13]. Blockchain records each transaction as a part of an existing block or as a new block. The first block in a blockchain is called the genesis block, and this is where blockchain starts. Each block generates encrypted input to the next block, and this creates extremely secure chains of blocks, where one unauthorized change in a block will make all blocks invalid [2], [12-13].

Each block in a blockchain is generated by a group of computing nodes called miners [2]. Miners collaboratively generate each block by following strict rules of generating a block, and by using strong cryptographic hash algorithms such as SHA-256 [10], [13]. Each VUPU created is recorded in a blockchain and any transaction performed on VUPU is also recorded in a blockchain. A transaction can be a sell transaction, a buy transaction, or an exchange transaction. Since VUPUs are recorded in blockchain, it is impossible to change any transaction details.

3.2 Small-cell blockchain

In a small-cell blockchain, each event in the network is recorded as a part of an existing block or part of a new block [2], [12]. An event in the network is defined as any action occurring in the network. For instance, a newly changed small-cell parameter is an event, or the amount of data transmitted to a user in the downlink is also an event. The higher the number of small-cells, the higher the number of events happening in the network; however, block generation will be also faster since a high number of small-cells will mine a block in small-cell blockchain [2].

4. BASE STATION TOKENS

The formula for the price of VUPU is

$$P(T)_{\text{VUPU}} = \frac{\sum_{n=1}^{B}\left(DDF_n \,.\, SUF_n \,.\, \left(\sum_{i=180k}^{T=180(k+1)}\left(RPU_{i,n} \,.\, N_n\right)\right)\right)}{S^{\left(\frac{1}{DDF \,.\, SUF}\right)}} + \frac{\max\{TI,0\}}{S^{\left(\frac{1}{DDF \,.\, SUF}\right)}}. \quad (10)$$

The first set of shares of a cluster will be 1000 shares, and issue time will depend on network performance. Naturally, shares will be issued when they are the most valuable, and also shares will be sold by investors when they are the most valuable. Fig. 10 shows the impact of DDF on the number of distributed shares of a cluster. When DDF increases, the value of the network cluster will increase and the price of the issued share will also increase. If the wireless network serves a diverse set of traffic at the beginning of network operation, then the share price will increase faster and will stabilize later (DDF = 2, 3). An additional set of traffic served by the network will also increase the price, however with a decreasing step-size (DDF = 4). In this article, the demand of investors on the price of shares is not investigated, and it will be studied in the next article. Each network cluster has a cluster head. When VUPUs are issued for a network cluster, VUPUs are recoded in small-cell blockchain as part of an existing block or as a new block. Blocks are generated by each miner base station in a network cluster, and each network cluster has a different number of miners. A consensus algorithm is used to verify the load of each base station in the network. Proof of work is used in bitcoin network [2], and Proof of stake is used in an Ethereum network [12].

The Proof of Data Load (PoDLO) consensus algorithm is a newly introduced consensus algorithm and is used to verify the load of a base station so that the correct VUPUs are calculated. A cluster head sends load information to each base station in the cluster, and there might be a direct connection between base stations. If there is no direct connection between two base stations, messages are relayed through base station(s) in the middle.

There is a total of 6 messages exchanged between base stations for a network shown in Fig. 11. Fig. 12 shows that there are 2 . N +N . (N-1) = N . (N+1) messages exchange between nodes, where N is the total number of base stations except the cluster head. This is a logical network architecture with one cluster head controlling two base stations. Since each VUPU transaction is recorded in a block, the block creation rate is very crucial. And the block is created when the network cluster has received votes from each base station.

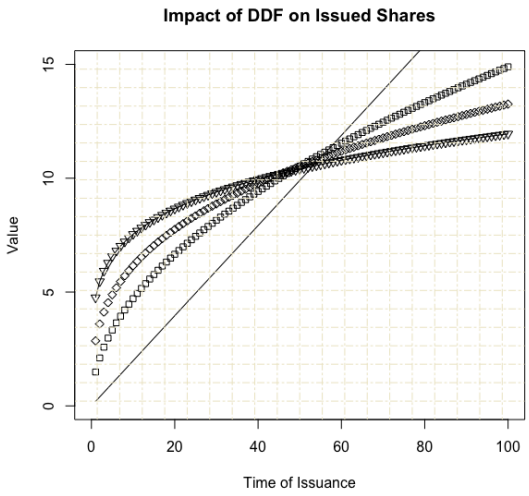


Fig. 10 – Impact of DDF on the issued VUPU shares

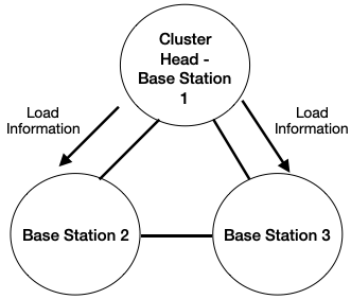


Fig. 11 – Cluster head, base stations, and messaging

Vote means 'load verification' messages that each base station sends to the network cluster. There is one cluster head running the consensus algorithm, and each base station belonging to the cluster sends its load verification votes to the network cluster head. Load verification means the verifying of a load of a base station in the network by another base station so that accurate load information can be collected for better price calculation of network resources. When a base station verifies a load of another base station, the verifier base station sends a 'Yes' vote to the cluster head, and otherwise the verifier base station sends a 'No' vote to the cluster head. If the number of 'Yes' votes is larger than 'No' votes, then a new block is created that is used to record all VUPUs related network transactions.

$$BCR = (C\,.M.DTL) + (BPS^{(\frac{1}{TDF})}) \qquad (11)$$

where BCR is the block creation rate, C is the constant, M is the number of nodes except the cluster head, DTL is data transmission latency, and BPS is the base station's processing speed. BPS can be represented with a logarithmic function since processing speed goes down with an increasing

number of processes. Each base station consists of central processing unit(s), and memory units the power consumption of which increases with the increasing amount of load. Increasing power consumption also increases the heat on the device, which in turn decreases the processing capability of the device.

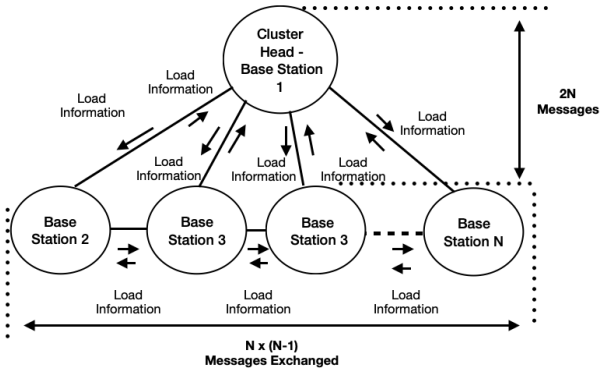


Fig. 12 – Network cluster message exchange model

One proposed model for BPS is

$$BPS = \left(\frac{1}{a}\right).\left(V^{(\log_{10}(\frac{V}{M}))}\right) + C \qquad (12)$$

where 'a' is the number of processors of base station hardware, or number of cores in a single processor. 'V' is the number of processes, threads processed by hardware of the base station and 'C' is the number of processors or processor cores assigned to the mining process.

Fig. 13 shows data processing latency, that is (1-DPS), of a base station with one processor (a = 1), two processors (a = 2), and four processors (a = 4). M is the constant related to the CPU model, speed, and impact of heat on CPU performance. M is a direct function of CPU processing power/speed, and the load on a CPU. With increasing CPU load and the data processing speed, M will also increase.

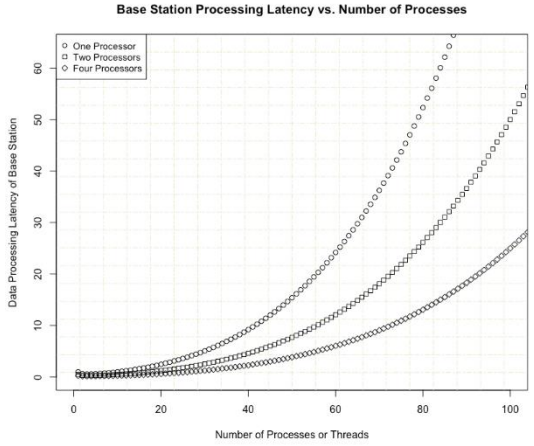


Fig. 13 – Base station processing latency

M value is usually given by the manufacturer of the CPU and it is unique to CPU model, type, and family. When a certain percentage of a base station's resources is assigned to the mining process, BCR will increase since miner base stations will only use these resources to create a block in small-cell blockchain. LTDS depends on the type of the link changing between (1msec., 10 msec.). With an increasing number of nodes, BCR also increases. The VUPU transaction rate will decrease which is not acceptable since a fast VUPU transaction record rate should be very fast so that buy and sell orders can be executed in almost real time. On the contrary, bigger N values will lead to a higher VUPU value. With one cluster head in a network cluster with 100 base stations each with a single processor, with TDF = 2.5, the BCR is calculated as follows:

$$BCR = (2\ x\ 100\ x\ 10\ msec) + \left(DPS^{\left(\frac{1}{2.5}\right)}\right) =$$

$$2\ \text{sec} + 40\ msec = 2.04\ sec. \qquad (13)$$

By changing the cluster size, base station type, transmission link type between base stations, BCR can be changed, adjusted and be made location and time dependent.

5. CONCLUSION

This paper presented a novel blockchain-based tokenization method for wireless network resources. A tokenization method tokenizes network reources, and determines the price of each unit of these resources. Tokenized network resources can be transacted by network investors, and collected funds can be used by service providers to finance new 5G and 6G network deployment.

This crowdfunding method enables network investment by individuals and companies. VUPU, DDF and SUF terms are newly introduced in this paper. The price of VUPUs can be calculated differently depending on how fast the investment fund should be sourced. A new formula to calculate the price of VUPU is disclosed. With increasing DDF and SUF values, the VUPU price also increases. VUPUs can be bought and sold at any time, and each VUPU transaction is recorded in a blockchain.

REFERENCES

[1] 3GPP Technical Standard Release 16. Available:https://www.3gpp.org/release-16

[2] Bitcoin: A Peer-to-Peer Electronic Cash System. Available: https://bitcoin.org/bitcoin.pdf

[3] J. Wang, J. Weitzen, V. Sevindik, O. Bayat, M. Li, "Dynamic Centralized Interference Coordination in Femto Cell Network with QoS Provision", Proceedings of the 18th International Conference on Communications (CSCC '14), pp. 81-86, July 2014

[4] V. Sevindik, Wireless Spectrum Trading: Techniques and Key Performance Indicators, Amazon Digital Services LLC, 2016

[5] V. Sevindik, Systems and methods for configuring a scheduler at an access node, US Patent No: 9894677

[6] V. Sevindik, Systems and methods for scheduling transmissions from an access node, US Patent No: 9525535

[7] V. Sevindik, J. Wang, O. Bayat, J. Weitzen, "Performance Evaluation of a Real Long Term Evolution (LTE) Network", 37th Annual IEEE Conference on Local Computer Networks", Fenruary 2013.

[8] V. Sevindik, O. Bayat, J. Weitzen, "Radio Resource Management and packet scheduling in femtocell networks", International Symposium of Modeling and Optimization of Mobile, Ad Hoc and Wireless Networks, May 2011.

[9] 3GPP Specification #36.423. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2452

[10] Cryptographic Hash Function. Available: https://en.wikipedia.org/wiki/Cryptographic_hash_function

[11] 3GPP Specification #36.410. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2443

[12] Ethereum Smart Contract. Available: https://www.ethereum.org/

[13] Smallcellcoin. Available: https://www.smallcellcoin.com

AUTHOR

Dr. Volkan Sevindik received a B.S. degree in electronics and telecommunication engineering with distinction (Summa Cum Laude) from Istanbul Technical University, Istanbul, Turkey, and an M.S. degree in electronics and electrical engineering from Bogazici University, Istanbul, Turkey, and a Ph.D. degree in electrical and computer engineering from University of Massachusetts, Lowell, MA, USA. Dr. Sevindik has held numerous managerial, and technical consultant roles in tier-1 wireless service operators, and well known consulting companies all over the world. He is the founder of Smallcellcoin Inc., where he has been conducting research on blockchain-based automated wireless network deployment, and management techniques since 2016. Dr. Sevindik holds 35 granted US patents with a total of 62 US patent applications. He has published 23 conference and journal papers, 1 book, 2 book chapters. He is an active member of IEEE, and he is actively providing technical consulting services on wireless network architecture, network financial planning, spectrum acquisition, and pricing.

INDEX OF AUTHORS

International
Telecommunication
Union

Telecommunication
Standardization Bureau (TSB)

Place des Nations
CH-1211 Geneva 20
Switzerland

ISSN: 2616-8375
Published in Switzerland
Geneva, September 2021