# Message from the Chairs
# SERP4IoT 2019

SERP4IoT 2019, the first edition of the International Workshop on Software Engineering Research & Practices for the Internet of Things has taken place in Montreal, Canada, on May 27th. It was held in conjunction with the 41st ACM/IEEE International Conference on Software Engineering (ICSE 2019).

It is our objective to establish SERP4IoT as an annual venue gathering researchers, industrials, and practitioners to share their vision, experience, and opinion on how to address the challenges of, find solutions for, and share experiences with the development, release, and testing of robust software for IoT devices.

To date, there is no precise definition of what constitutes IoT, since it encompasses many different aspects of software design, development, evolution, deployment and operation, including success, longevity, growth, resilience, survival, diversity, sustainability, transparency, privacy, security, etc.

Software engineering is vital for IoT to design systems that are secure, interoperable, modifiable, and scalable. However, there is no a consensus of crucial questions like what are the best practices for developing projects for IoT, how to select the best architecture, which communications protocols are the most suitable, and what are the best practices in terms of security.

The technical program of the workshop includes presentations of 16 papers, contributed by 53 authors from 13 different countries. Each paper was reviewed by at least three Program Committee members. All papers were evaluated according to their relevance to the workshop, soundness, maturity and presentation quality. Accepted long and short papers of high quality were invited for presentation and inclusion in the workshop proceedings, published by IEEE Xplore Digital Library, while rest were invited for presentation and the internal proceedings only.

Different from a typical conference type workshop, SERP4IoT counted we several activities to foster participation of the assistants in a constructive manner. We had Beach Ball Buzz game, affinity mapping and card sorting for discussing existing issues and challenges on IoT, and closed with an open *Fishbowl* to allow each attendant to give their opinion in an interactive way.

We were very proud to host three invited keynotes by Enrique Alba (University of Málaga, Spain), Hausi A. Müller (University of Victoria, Canada), and Mathieux Bergeron (Kinova Robotics).

Topic-wise, we received diverse submissions investigating a wide range of software engineering issues and challenges for the IoT: such as end user programming models, software architecture for interoperability, model driven engineering techniques, security, reliability, and diverse applications of IoT to support a variety of domains like pisciculture, gaming, etc.

We would like to thank everyone who made the workshop come to life, including the authors, the Program Committee members and the ICSE 2019 Workshop Chairs. We were glad to have your presence in the workshop and hope to meet you again at a future SERP4IoT edition!

| | |
|---|---|
|  | Danny Dig (Oregon State University, USA)<br>digd@eecs.oregonstate.edu |
|  | Rodrigo Morales (Concordia University, Canada)<br>rodrigo.morales2@acm.org |
|  | Rubén Saborido (Concordia University, Canada)<br>rsain@uma.es |
|  | Shah Rukh Humayoun (Tufts University, USA)<br>humayoun@cs.tufts.edu |
|  | Yael Dubinsky (Technion, Israel Institute of Technology, Israel)<br>yaeldubinsky3@gmail.com |
|  | Yann-Gaël Guéhéneuc (Concordia University, Canada)<br>yann-gael.gueheneuc@concordia.ca |

# Program Committee
# SERP4IoT 2019

| | | |
|---|---|---|
| **Amin** | **Hammad** | Concordia University |
| **Fabio** | **Petrillo** | Universite du Quebec a Chicoutimi |
| **Foutse** | **Khomh** | DGIGL, École Polytechnique de Montréal |
| **Francisco** | **Chicano** | University of Málaga |
| **Hamid** | **Mcheick** | University of Quebec At Chicoutimi |
| **Hironori** | **Washizaki** | Waseda University |
| **Juan Carlos** | **Herrera Lozada** | IPN - CIDETEC |
| **Lucas** | **Hof** | École de technologie supérieure |
| **Marios** | **Fokaefs** | École Polytechnique de Montréal |
| **Mónica** | **Pinto** | Universidad de Málaga |
| **Nobukazu** | **Yoshioka** | National Institute of Informatics |
| **Roberto Erick** | **Lopez-Herrejon** | École de Technologie Supérieure |

# Submission list

**Paper index:** [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16]

[1] Ramapriya Ranganath (Microsoft). *IoT Green Corridor .*

**Abstract.** This is a system that combines the existing trivial traffic signal lights with sensors, which are capable of synchronizing with each other, and take certain decisions on the switching of lights as per the given set of conditions. Synchronizing of sensors is possible only by creating a network among them. Hence, the sensors are mutually connected by means of internet using an IoT (Internet of Things) network. The Internet of Things (IoT) is a new and evolving concept that provides connectivity to the Internet via sensing devices to achieve intelligent identification and management in a heterogenous connectivity environment. The aim is to design a prototype of traffic junctions which can sense the presence of an Emergency Vehicle (EMV) within a threshold distance and change the traffic flow, so that the EMV passes the signal within less time and uninterrupted. This very prototype consists of sound sensors (amplified by IC LM 358), which are connected to a circuit comprising of PIC 12F683 microprocessor, that senses the presence of the EMV by detecting the frequency of its siren. Based on the output of this frequency detecting circuit, the other sound sensors placed at the respective lanes start sending data to an Intel Galileo Board, which is a microcontroller, that takes decisions on switching of traffic lights in the junction, by comparing the inputs given by the sound sensors. As soon as the EMV leaves the junction, The Galileo Board sends a signal to a Linkit ONE board (an IoT development board) which sends a value to the Cloud, which in this case is the Ubidots cloud computing platform, signalling that the EMV has passed the junction. As soon as the cloud receives this signal an event-based action occurs, which sends a signal to the next junction, according to the pre-defined path of the EMV, indicating that it is approaching the junction. This next junction consists of a similar setup of microcontrollers and sensors, which repeat the process. This occurs at each traffic junction till the EMV reaches its destination.

**Time:** Jan 23, 05:40
**Decision:** ACCEPT
**Keywords:** Emergency Vehicles, IoT, Green Corridor, Goertzel's Algorithm
**Paper:** ✔

[2] Steven Reiss (Brown University). *IoT End User Programming Models.*

**Abstract.** The advent of smart devices and sensors (the Internet of Things or IoT) will create increasing demands for the automation of devices based on sensor, time, and other inputs. This is essentially a programming task with all the problems and difficulties that programming entails, for example, modularity, feature interaction, debugging, and understanding. Moreover, much of the programming for smart devices is going to be done not by professional programmers but by end users, often end users without any programming experience or computational literacy. Our research is aimed at exploring the programming space and the associated issues using a case study of a smart sign that can be controlled using a variety of sensors. We have developed a general system for programming smart devices and, in this paper, explore a variety of different user interfaces for programming this system for our smart sign.

**Time:** Jan 28, 17:07
**Decision:** Accept and recommend
**Keywords:** Internet of things, end-user programming, debugging, program understanding.
**Paper:** ✔

[3] Aleksandr Lepekhin (Peter the Great St.Petersburg polytechnic university), Alexandra Borremans (Peter the Great St.Petersburg Polytechnic University), Sami Jantunen (LUT University) and Igor Ilin (Peter the Great St.Petersburg Polytechnic University). *A Systematic Mapping study on Internet of Things challenges .*

**Abstract.** The challenge of developing IoT-based systems has been found to be a complex problem. It is influenced by number of factors: heterogeneous devices/resources, various perception-action cycles and widely distributed devices and computing resources. Increasing complexity and immaturity to deal with it have resulted in growing range of problems and challenges in IoT development. This paper identifies essential IoT-related challenges by conducting a systematic mapping study of existing IoT literature. To this end, we distil information with respect to IoT-related: 1) challenges, 2) experimental studies, and 3) recommendations for future research. We then discuss our findings in order to understand better the general state of IoT research, potential gaps in research, and implications for future research.

**Time:** Jan 31, 19:14
**Decision:** Accept and recommend
**Keywords:** Internet of Thing, IoT, IoT Challenges, IoT Development, Systematic mapping study
**Paper:** ✔

[4] Zayan El-Kaheld (University of Quebec at Chicoutimi), Hamid Mcheick (University of Quebec at Chicoutimi) and Fabio Petrillo (University of Quebec at Chicoutimi). *WiFi coverage range characterization for smart space applications.*

**Abstract.** Recently, humans are more and more dependent to communication technologies (CT) in their everyday life to get services, exchange information and communicate with their relatives. Hence, many researches have been made in order to propose convenient and low-cost solutions compatible with the context of smart spaces. This paper analyzes a solution based on WiFi in order to provide a low-cost communication solution for the smart space. Technical characteristics are illustrated and the coverage range is characterized. Furthermore, a wide deployment network model is deducted on light of obtained results.

**Time:** Feb 01, 02:41
**Decision:** Accept and recommend
**Keywords:** Smart space, resilient network, WiFi, coverage range, radio link quality, wireless internet service
**Paper:** ✔

[5] Cristiano Politowski (Concordia University) and Fabio Petrillo (Universite du Quebec a Chicoutimi). *Improving engagement assessment in gameplay testing sessions using IoT sensors.*

**Abstract.** Video game industry is a multimillionaire market which makes solo indie developers millionaire in one day. However, success in the game industry it is not a coincidence. Video game development is an unusual kind of software that mix multidisciplinary teams, as software engineers, designer and artists. Further, for a video game be well received, it must be fun and polished, so exhaustively well tested. Testing in video game development ranges from different types, in different parts of the process. For example, measuring the engagement of players in a test session can drive the development drastically. The designers/developers analyze actions taken by players and how they behave facing each decision in the game. Based on that, they decide if that feature/level requires rework or cut it. It is very common to throw out many hours of man/work in a feature just because it is not fun. As the designers (usually) assess the gameplay session by hand, how can they be sure that specific feature is (or is not) good enough? If we could provide more meaningful data for the designers to review we can have a better assessment of what is happening in the gameplay, what could be wrong and if there is a real need to remove or rework. In this paper, we propose an IoT environment platform to assess the player's engagement in gameplay session by adding IoT sensors together with game devices which will produce a rich output for the designers.

**Time:** Feb 12, 17:18
**Decision:** ACCEPT
**Keywords:** Gameplay, Testing, Internet of Things, Software quality
**Paper:** ✔

[6] Andrew Truelove (University of Houston), Farah Naz Chowdhury (University of Houston), Omprakash Gnawali (University of Houston) and Mohammad Amin Alipour (University of Houston). *Users Issues in using the Internet of Things Systems.*

**Abstract.** Internet of Things (IoT) systems are bundles of networked sensors and actuators that are deployed in an environment and act upon the sensory data that they receive. These systems, especially consumer electronics, have two main cooperating components: a device and a mobile app. The unique combination of hardware and software in IoT systems presents challenges that are lesser known to mainstream software developers and might require innovative solutions to support the development and integration of such systems.

In this paper, we analyze the more than 90,000 reviews of ten IoT devices and their corresponding apps and extract the issues that users encountered in using these systems. Our results indicate that issues with \emph{connectivity}, \emph{timing}, and \emph{update} are particularly prevalent in the reviews. These results also call for a new software-hardware development framework to assist the development of reliable IoT systems.

**Time:** Feb 15, 23:36
**Decision:** Accept and recommend
**Keywords:** User Review Analysis, Internet of Things, Empirical Study
**Paper:** ✔

[7] Juan Moreno (Universidad Nacional Mayor de San Marcos), Felipe Moreno (Universidad Católica San Pablo) and Frank Moreno (Universidad Nacional de Ingeniería). *Proposal of a New software architecture for interoperability to improve the communication in the Edge layer of a smart IoT ecosystem.*

**Abstract.** In the current years, IoT has evolved to such an extent to extend to all corners of each place through devices, which connected to a network, either local or internet itself, generate information to be processed with a specific purpose, to this level there is a problem called interoperability of devices where not only is the compatibility of adding or removing devices to an ecosystem and there is compatibility, it is also expected that the information generated is standardized and optimized to transmit. This paper presents a new software architecture pattern for interoperability between devices that generate heterogeneous information in the edge layer of an IoT ecosystem.

**Time:** Feb 17, 06:35
**Decision:** ACCEPT
**Keywords:** Interoperability, IoT, data encode, data decode, protocol buffer, Edge Computing, REST API, Software Architecture, IoT Ecosystem
**Paper:** ✔

[8] Tansu Aşıcı (Ege University), Burak Karaduman (Ege University), Raheleh Eslampanah (Izmir University of Economics), Moharram Challenger (University of Antwerp), Joachim Denil (University of Antwerp) and Hans Vangheluwe (University of Antwerp and McGill University). *Applying Model Driven Engineering Techniques to the Development of Contiki-based IoT Systems.*

**Abstract.** The huge variety of smart devices and their communication models increases the development complexity of embedded software for the Internet of Things. As a consequence, development of these systems becomes more complex, error-prone, and costly. To tackle this problem, in this study, a model-driven approach is proposed for the development of Contiki-based IoT systems. To this end, the available Contiki metamodel in the literature is extended to include the elements of WiFi connectivity modules (such as ESP8266), IoT Log Manager, and information processing components (such as Raspberry Pi). Based on this new metamodel, a domain-specific modeling environment is developed in which visual symbols are used and static semantics (representing system constraints) are defined. Also, the architectural code for the computing components of the IoT system such as Contiki, ESP8266, and RaspberryPi are generated according to the developer's instance model. Finally, a Smart Fire Detection system is used to evaluate this study. By modeling the Contiki-based IoT system, we support model-driven development of the system, including WSN motes and sink nodes (with ContikiOS), WiFi modules and information processing components.

**Time:** Feb 17, 13:52
**Decision:** Accept and recommend
**Keywords:** Model-driven Engineering (MDE), Internet of Things (IoT), Embedded Software, Wireless Sensor Network, ContikiOS, Smart Fire Detection System
**Paper:** ✔

[9] Snehasis Banerjee (Tata Consultancy Services) and M Girish Chandra (Tata Consultancy Services). *A Software Framework for Procedural Knowledge based Collaborative Data Analytics for IoT.*

**Abstract.** The outburst of data generation by machines and humans, along with emergence of sophisticated data processing algorithms have created a demand for a wide number of data analytics based services and applications. The paper presents a collaborative framework and system to carry out a large number of data processing tasks based on semantic web technology and a combination of reasoning and data analysis approaches using software engineering guidelines. The paper serves as a first step for systematic fusion of symbolic and procedural reasoning that is programming language agnostic. This approach helps in reducing development time and increases developer's productivity. The proposed software system's logical functionality is explained with the help of a healthcare case study, and the same can be extended for other applications.

**Time:** Feb 17, 18:22
**Decision:** Accept and recommend
**Keywords:** Procedural Reasoning, IoT Analytics, Software Framework, Software Orchestration, Developer Productivity
**Paper:** ✔

[10] Mehboob Hasan Rohit (North South University), Zarin Tarannum Hoque (North South University), S M Mujibul Karim (North South University) and Dr. Shahnewaz Siddique (North South University). *Cost Efficient Automated Pisciculture Assistance System using Internet of things(IoT).*

**Abstract.** The more the technology is getting advanced, the more it makes the life of people depending on it and one of that technologies is automation. Internet of Things (IoT) is another term which envisions every physical object that are being connected to the internet and being able to identify themselves to other devices. This paper illustrates a methodology to provide a low cost Automated Pisciculture Assistance System for indoor fish production using Wireless Fidelity (Wi-Fi). Despite being ranked third in the world in terms of inland fish production [1], Bangladesh is presently using the method of Recirculation Aquaculture Systems (RAS), because pollution of pond water is a major factor posing significant danger to hygiene issue for fish population inhabiting in pond water. In this research we have developed a complete assistance system which gives update to the user of the conditions of the water through the sensors and operate the device remotely. The key components of this system are a pocket-sized Wi-Fi module, Message Queuing Telemetry Transport (MQTT) for monitoring, controlling the sensors and alerting the user through SMS, an Android application to visualize the data provided by the module and to operate the device. Our main objective was to design a system to overcome the downsides with minimal costing and easy installation process.

**Time:** Feb 17, 19:05
**Decision:** Accept and recommend
**Keywords:** Internet of Things (IoT), Wireless Fidelity (Wi-Fi), Pisciculture Automation System, Wi-Fi module, Android, Sensors
**Paper:** ✔

[11] Orges Cico (Norwegian University of Science and Technology). *Reliable IoT Systems for Improving Quality Of Life Through The Exploitation of Cloud, Mobile and BLE Low Energy Based Technologies. Case Study: Battery Charge Protect.*

**Abstract.** Connecting various smart objects within an intelligent ecosystem provides high capability of developing and integrating mobile, cloud and embedded device communications based on existing internet infrastructure. This has been the trend of Internet of Things (IoT), addressing health, quality of life, smart cities.

In this paper we are a focused on similar aspects with a proper case study in an embedded system, used in protecting batteries of mobile devices. Cloud/Mobile applications interconnected with embedded devices shall be presented. The research undertaken present not just the technological innovation, exploiting state of the art technology, but also the security benefits and prolongation of battery life. The system has been fully developed, tested and it's benefits evaluated throughout the paper. We have been able to integrate and provide a real life case study adopting this integrated architecture to be reused for future implementations.

**Time:** Feb 17, 21:02
**Decision:** Accept and recommend
**Keywords:** Internet-of-Things, Smart Devices, Fall detection, Energy efficiency, Wearable devices
**Paper:** ✔

[12] Nadine Kashmar (Université), Mehdi Adda (Université), Mirna Atieh (Lebanese University) and Hussein Ibrahim (Institut Technologique de Maintenance Industrielle (ITMI)). *A New Dynamic Smart-AC Model Methodology to Enforce Acess Control Policy in IoT layers.*

**Abstract.** Internet of Things (IoT) is the conversion of everyday tangible devices or machines to smart objects. This means that these objects would be able to think, sense and feel. For example, your home devices will be able to detect and feel your absence to turn off the lights of empty rooms, close doors, lock the gates, and other tasks. Thus, would it be acceptable to find intruders who might mess up your daily life style or control your home appliances? Absolutely not! The same idea for factories, they definitely reject to detect any unacceptable access from any foreigner to their logical/physical assets or machines who might be able to locally or remotely control, for example, any machine operation. This would cause a significant loss for their reputation or investments, since any vulnerability or attack can produce, for example, fault products. So far, IoT is considered as one of the most essential areas of future technologies, especially for the industries. Hence, finding an environment full of smart devices needs a smart security methodology to prevent any illegal access. In this domain, various researches are conducted to find Access Control (AC) models to enforce security policies that prevent any unauthorized detection of sensitive data and enable secure access of information. For this purpose, we present a new dynamic Smart-AC model methodology to enforce security policy in IoT layers.

**Time:** Feb 18, 00:03
**Decision:** Accept and recommend
**Keywords:** IoT, access control, smart, security, model, policy
**Paper:** ✔

[13] Elizabeth Reilly (Massachusetts Institute of Technology), Matthew Maloney (Massachusetts Institute of Technology), Michael Siegel (Massachusetts Institute of Technology) and Gregory Falco (Massachusetts Institute of Technology). *A Smart City IoT Integrity-First Communication Protocol via an Ethereum Blockchain Light Client.*

**Abstract.** Smart city IoT is responsible for communicating system-critical data about urban infrastructure that keeps our modern cities functioning. Today, IoT devices lack communication protocols with data integrity as a priority. Without data integrity, smart city infrastructure is at risk of actuating urban environments on compromised data. Attackers can use this IoT communication flaw to wage cyber-physical attacks on Smart Cities. We designed and developed an integrity-first communication protocol for IoT that is distributed and scalable based on the Ethereum blockchain. Our light client ensures data communication integrity for systems that require it most.

**Time:** Feb 18, 00:09
**Decision:** Accept and recommend
**Keywords:** IoT Security, Blockchain, Smart Cities, Communication Integrity, Ethereum, IoT Communication Protocol
**Paper:** ✔

[14] Davino Mauro Junior (CIn-UFPE), Walber Rodrigues (CIn-UFPE), Kiev Gama (CIn-UFPE), José A. Suruagy (CIn-UFPE) and Paulo André da S. Gonçalves (CIn-UFPE). *Towards a Multilayer Strategy Against Attacks on IoT Environments.*

**Abstract.** The Internet of Things market has seen a large growth in numbers in the recent past. With it, security is becoming a usual concern among consumers. Taking an already existing categorization of typical IoT attacks grouped by TCP/IP network layers as a starting point, we did a non-exhaustive search of solutions addressing each attack. We found that solutions are typically focused on a single layer or even a specific attack only. Furthermore, these solutions lack flexibility to incorporate new attacks. To avoid the non-practical approach of having multiple non-extensible tools, this paper presents an ongoing work that focuses on a multilayer approach to IoT threats. The proposed system leverages an autonomic architecture to analyze network traffic in a distributed manner, detecting suspicious behavior with preconfigured rules being applied by a Complex Event Processing (CEP) engine.

**Time:** Feb 18, 03:30
**Decision:** Accept and recommend
**Keywords:** Security, Internet of Things, MAPE-K
**Paper:** ✔

[15] Hironori Washizaki (Waseda University), Nobukazu Yoshioka (National), Atsuo Hazeyama (Tokyo Gakugei University), Takehisa Kato (Toshiba Digital Solutions Corporation), Haruhiko Kaiya (Kanagawa University), Shinpei Ogata (Shinshu University), Takao Okubo (Institute of Information Security) and Eduardo B. Fernandez (Florida Atlantic University). *Landscape of IoT Patterns.*

**Abstract.** Patterns are encapsulations of problems and solutions under specific contexts. As the industry is realizing many successes (and failures) in IoT systems development and operations, many IoT patterns have been published such as IoT design patterns and IoT architecture patterns. Because these patterns are not well classified, their adoption does not live up to their potential. To understand the reasons, this paper analyzes an extensive set of published IoT architecture and design patterns according to several dimensions and outlines directions for improvements in publishing and adopting IoT patterns.

**Time:** Feb 18, 10:43
**Decision:** Accept and recommend
**Keywords:** Patterns, Internet of Things (IoT), Design, Architecture, Survey, Systematic Literature Review (SLR)
**Paper:** ✔

[16] Narayana Sashi Rekha Dada (Tata Consultancy Services, Hyderabad) and Aditya Allamraju (Andhra University College of Engineering (A), Visakhapatnam). *IOT based Air Pollution Monitoring and Control Mechanism using UAVs.*

**Abstract.** Air pollution in India is a serious issue which is increasing rapidly and reaching to hazardous levels. One of the major sources being sulphur dioxide (SO2) is emitted from the combustion of fossil fuels like coal and petroleum. Traffic congestion in India's most noticeable cities and towns is severe. It is caused due to increase in number of vehicles per kilometer of available road, chaos due to poor enforcement of traffic laws, obstacles in the road causing a blockage and merger etc. Complete lack of traffic sense in Indian public is the main reason for the chaos on the roads. Traffic congestion reduces the average traffic speed. At low speeds, it is scientifically proved that vehicles burn fuel inefficiently and pollute more per trip. In most of the highly congested Indian city roads, the average trip is less than 20 kilometers per hour. At such speeds, vehicles emit air pollutants 4 to 8 times more than they would, with less traffic congestion; and also consume a lot more carbon footprint fuel per trip than they would if the traffic congestion was less.

In context with the above issue in India, the present idea is to trace the vehicles which are emitting carbon dioxide (CO2) and other hazardous pollutants beyond the permitted levels especially at the points of huge traffic congestion. The thermal sensors mounted on the Unmanned Aerial Vehicles (UAVs) or drones can be used to detect the vehicles that are emitting higher amount of pollutants (beyond threshold amount) and when detected should also snap an immediate picture of the vehicle registration number. The picture is then scanned using image scanning techniques and based on the registration number, the corresponding image will be uploaded into RTO website as evidence thereby updating the Government records. In parallel from the RTO website, the vehicle owner's details can be fetched. Email/phone number linked to any of the person's identification proof like that of Aadhar number/ PAN number or address linked to the registered number should be sent an E-mail/SMS/letter of the receipt of fine and asking them to get the vehicle repaired to emit lesser pollutants.

This system works with the assistance of the technology of Internet of Things (IOT) which is a rising technology based on the fusion of electronics and computer science. The embedded sensors in the system help to detect major air polluting gases such as CO2, SO2 and CO. The concept of IOT helps to access data from remote locations and save it in database so that we don't need to actually be present in that area. This would be a benefit to keep a constant check on pollution, in keeping the country greener and also to avoid corruption thereby saving a lot of money to the nation.

**Time:** Feb 21, 13:38
**Decision:** ACCEPT
**Keywords:** Traffic congestion, Unmanned Aerial Vehicles, Internet of Things
**Paper:** ✔

# IoT Green Corridor

**R Ramapriya**
ramapriya288@gmail.com

***Abstract*—This is a system that combines the existing trivial traffic signal lights with sensors, which are capable of synchronizing with each other, and take certain decisions on the switching of lights as per the given set of conditions. Synchronizing of sensors is possible only by creating a network among them. Hence, the sensors are mutually connected by means of internet using an IoT (Internet of Things) network. The Internet of Things (IoT) is a new and evolving concept that provides connectivity to the Internet via sensing devices to achieve intelligent identification and management in a heterogenous connectivity environment. The aim is to design a prototype of traffic junctions which can sense the presence of an Emergency Vehicle (EMV) within a threshold distance and change the traffic flow, so that the EMV passes the signal within less time and uninterrupted. This very prototype consists of sound sensors (amplified by IC LM 358), which are connected to a circuit comprising of PIC 12F683 microprocessor, that senses the presence of the EMV by detecting the frequency of its siren. Based on the output of this frequency detecting circuit, the other sound sensors placed at the respective lanes start sending data to an Intel Galileo Board, which is a microcontroller, that takes decisions on switching of traffic lights in the junction, by comparing the inputs given by the sound sensors. As soon as the EMV leaves the junction, The Galileo Board sends a signal to a Linkit ONE board (an IoT development board) which sends a value to the Cloud, which in this case is the Ubidots cloud computing platform, signalling that the EMV has passed the junction. As soon as the cloud receives this signal an event-based action occurs, which sends a signal to the next junction, according to the pre-defined path of the EMV, indicating that it is approaching the junction. This next junction consists of a similar setup of microcontrollers and sensors, which repeat the process. This occurs at each traffic junction till the EMV reaches its destination.**

***Index Terms*—Emergency Vehicles, IoT, Green Corridor, Goertzel's Algorithm**

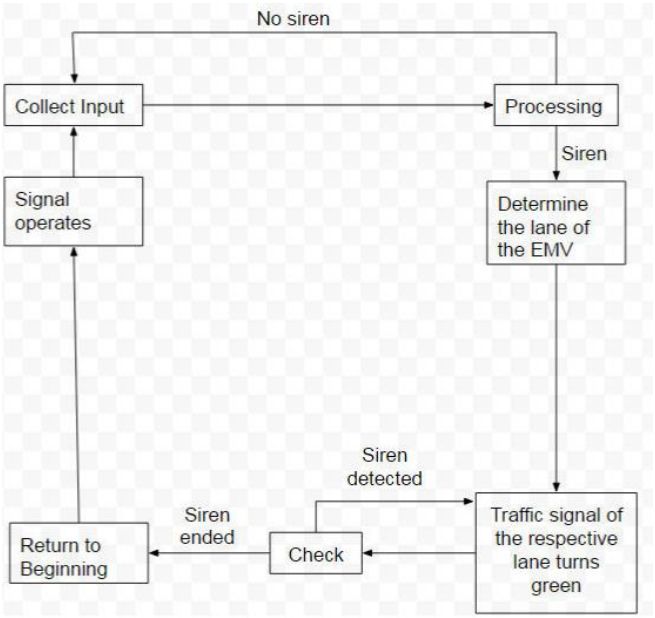
## I. INTRODUCTION

Every other day we are stuck in a jam, or we wait at a red signal and hear the siren of an Emergency Vehicle (EMV), such as an ambulance or a fire engine, going off behind us. But if there is no traffic police to take charge of the situation, we are helpless, and the EMV is forced to wait till the light goes green. Every year in India alone, several thousands of people lose their life because they were not given proper treatment on time, one of the prime reasons being the EMVs not able to reach their destination fast enough. The one thing that crosses our mind is, if only there was a system in place that would take care of this automatically, so people have no choice but to give the ambulance a free and clear path. This is exactly what this project aims to do. There is a mechanism in place at every traffic junction that detects the presence of an EMV and provides it a clear path. This mechanism can be efficient only if human-human or human-machine interaction is minimized, and hence the term IoT comes into play. IoT in layman's terms is the mutual interaction of objects with each other, causing event-based actions to take place. These objects may be sensors, microcontrollers, etc., and they communicate wirelessly thus creating a network, thus forming the Internet of Things The Internet of things (IoT) is the network of physical devices, vehicles, buildings and other items - embedded with electronics, software, sensors, actuators and network connectivity that enable these objects to collect and exchange data with the manufacturer, operator and/or other connected devices. Internet enabled objects, together with web services that interact with these objects, form this network. The Internet of Things enables objects to be sensed and controlled remotely across the existing network infrastructure, creating opportunities for more direct integration between the physical world and computer-based systems, and resulting in enhanced efficiency, accuracy and economic benefit. These devices collect useful data with the help of various existing technologies and then autonomously flow the data between other devices. In this project, a network of sensors is created at each junction. We all know that an ambulance or fire engine or any EMV for that matter has a characteristic siren. A sound sensor is connected in a circuit with a microprocessor programmed to detect this specific frequency. As soon as this frequency is detected, more sound sensors placed along the sides of the road are used to detect which lane the ambulance is in. Once this is known, the respective signal light goes green, and the others red, till the ambulance has crossed the junction. The sensors communicate wirelessly through the cloud. Each sensor sends its registered readings to the cloud and this data is analyzed and processed in the cloud. It then sends back specific data which causes the action of changing the lights to take place. The traffic from other lanes must be stopped until the emergency vehicle leaves the junction, hence, creating a green corridor for the EMV.

## II. DESIGN

The frequency sensor consists of a simple microphone with an amplifier connected in a simple circuit that processes the input collected by it. Following are the events that occur one after the other that lead to the changing of the signal light. The microphone of the frequency sensor keeps collecting input. The signals follow their default pattern. This input is processed by the PIC12F683 microprocessor using discrete Fourier transform. If the siren is not detected it repeats the process till a siren is detected. If a siren is detected, the readings from the amplitude sensors are then analyzed and the greatest one is found. Based on this the lane of the EMV is detected. Accordingly, the corresponding traffic light turns green and the others turn red. The light that turns green is the

one directly opposite the EMV. The frequency sensor continuously keeps checking if the conditions remain the same to keep the signal from changing till the ambulance has passed. If the siren is still present, it keeps the signal in the same state. If the siren is no longer loud enough to sense, indicating that the EMV has passed the junction, The default pattern is resumed. Also, as soon as it leaves the junction a value 1 is sent to a variable in Ubidots, the cloud computing platform used in this project. An event created in Ubidots then sets another variable to This is sent to the next junction to inform it that the EMV is approaching.




This is the basic System Block Diagram

## III. HARDWARE

**Intel Galileo Board:** It is a first in a line of Arduino-certified development boards based on Intel x86 architecture and is designed for the maker and education communities. Intel Galileo combines Intel technology with support for Arduino ready-made hardware expansion card (called "shields") and the Arduino software development environment and libraries. The development board runs an open source Linux operating system with the Arduino software libraries, enabling re-use of existing software called "sketches". Intel Galileo can be programmed through OS X, Microsoft Windows and Linux host operating software. The board is also designed to be hardware and software compatible with the Arduino shield ecosystem.

**Linkit ONE development board**: The Linkit ONE development platform is an open source, high performance board for prototyping wearables and IoT devices. It is based on the world's leading SoC for wearables, MediaTek Aster (MT2502) combined with high performance Wi-Fi (MT5931) and GPS (MT3332) chipsets to provide you with access to all the features of MediaTek Linkit. It also provides similar pin-out features to Arduino boards, making it easy to connect various sensors, peripherals and Arduino shields. Linkit ONE is an all-in-one prototyping board for IoT/ wearables devices. Integrating GSM, GPRS, Wi-Fi, GPS, Bluetooth features into a basic Arduino form factor.

**Grove Sound Sensor:** A Grove - Sound sensor can detect the sound strength of the environment. The main component of the module is a simple microphone, which is based on the LM358 amplifier and an electret microphone. This module's output is analog and can be easily sampled and tested by any Arduino compatible microcontroller board

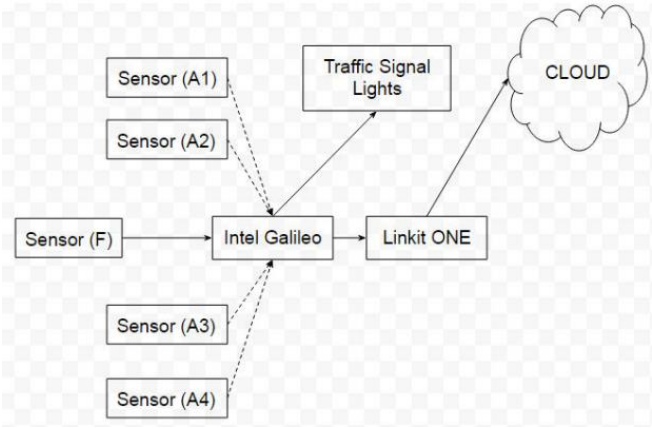
**PIC12F683 microprocessor:** These are the Low Pin-Count (8) PIC Flash Microcontroller products that offer all of the advantages of the well-recognized midrange x14 architecture with standardized features including a wide operating voltage of 2.0-5.5 volts, on-board EEPROM Data Memory and nanowatt Technology. Standard analog peripherals include up to 4 channels of 10-bit A/D, an analog comparator module with a single comparator, programmable on-chip voltage reference and a Standard Capture/Compare/PWM (CCP) module

**Base Shield:** The Grove base shield plugs into all Arduino compatible boards and is the foundation of the Grove system. All I/O ports of the controller board are exposed and adapted into Grove connectors which include digital I/O, analog I/O and specialized ports (I2C, SPI, UART).

Final Stage

## IV Hardware Block Diagram

A frequency sensor (consisting of a simple microphone in a circuit) first detects the presence of a sound of frequency 750 Hz. If it does detect a sound of this frequency, the analog inputs from the four amplitude sensors, each placed on the left side of the road is read into the Galileo. The Galileo then checks to see which sensor records the highest value, and accordingly gives a high output to the respective pin. This high output is sent to one of three analog input pins of linkit one. Depending on which pin turns high, the respective traffic signal turns GREEN and the others turn RED. This occurs as long as the readings of that amplitude sensor remains the highest. When the readings of all the sensors are then below a threshold, indicating that the ambulance has crossed the junction, the traffic signals continue to change as per the default cycle




The frequency detector circuit can be used to detect the presence of a certain frequency within an analog signal, such

as an audio signal. This circuit is based on 8 pin PIC 12F683 microprocessor. Only a few additional resistors and capacitors are needed to complete a circuit that will accept an analog signal and drive a microprocessor output pin high when the selected frequency is present in the signal. The steps that follow detail operation of the circuit and the program that runs on the processor. A description of the digital signal processing algorithm used by the PIC is included. The code for programming the processor is included as a .hex file and also the source code .as file is also provided. The operation of the circuit and function of each component is described This circuit can easily be built on a breadboard. You will need a programmer for PIC microprocessors. In this case we use PICkit 3 to program the microprocessor. We program the PIC processor using a PICkit 3 programmer and for this we need MPLABX IDE and MPLAB IPE software's.

## IV. GOERTZEL'S ALGORITHM

The Goertzel algorithm is a signal processing algorithm which is used for detecting a single frequency. It is derived from the Fourier transform. The algorithm acts as a very narrow band pass filter. It produces a very sharp response to frequencies within the pass band, and a much lower response for frequencies outside the pass band. The minute details of the Goertzel algorithm won't be covered here. The algorithm loop samples the input using the microprocessor built in A/D converter. The necessary mathematical operations are performed in the time between successive samples

$$\text{Cosine Coefficient} = \cos\left(2\pi\frac{f_{target}}{f_{sample}}\right)$$

$$\text{Sine Coefficient} = \sin\left(2\pi\frac{f_{target}}{f_{sample}}\right)$$

$$W_n = 0.54 + 0.46 \cdot \cos\left(2\pi\frac{n}{N}\right)$$

$$X_n = X_{n\_sample} \cdot W_n$$

$$Y_0 = X_n + Y_1 \cdot \text{Cosine Coefficient} - Y_2$$

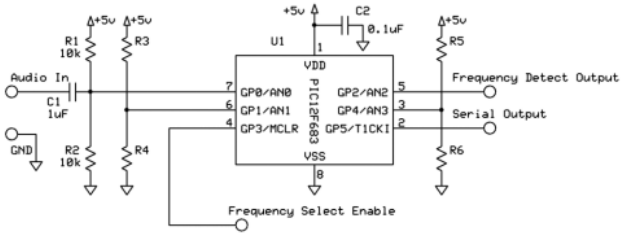
$$Real = Y_1 - Y_2 \cdot \text{Cosine Coefficient}$$

$$Imaginary = Y_2 \cdot \text{Sine Coefficient}$$

$$Magnitude = \sqrt{(Real)^2 + (Imaginary)^2}$$

Xn_sample=The latest sample from the A/D converter
Xn=The latest sample from the A/D converter multiplied by the window function.
Y0=The output value presently being computed.
Y1=The previously computed output value. (The output value from the previous iteration of the loop).
Y2=The output value from the iteration the iteration before the l previous iteration of the loop.
The figure below shows a list of the variables used in the algorithm. Refer to the flow chart in the blog to see how the Goertzel algorithm portion of the program executes on the processor. Once the algorithm has processed all the samples (the attached *.hex and *.as files use 200 samples), the real and imaginary components of the results are calculated. The real and imaginary portions are then used to compute the magnitude. The magnitude is a measure of the target frequency is present in the sampled data. The magnitude is then compared against the threshold. If the value is greater than the threshold value, the frequency is considered to be detected, and the output is set HIGH. If the magnitude is lower than the threshold, the output is set LOW. The sharpness of the filter response versus frequency is proportional to the number of samples taken. The response of the algorithm must be sharp enough that it responds to the target frequency but produces much lower response for frequencies outside the target. A value of 200 samples was found to produce a reasonably narrow response in experimentation.

## V. Building the Circuit




Pin 1: Vdd (+5 volts)
Pin 2: Serial Output
Pin 3: Detection Threshold Analog Input
Pin 4: Frequency Select Enable
Pin 5: Frequency Detect Output
Pin 6: Frequency Select Analog Input
Pin 7: Biased Signal Analog Input
Pin 8: VSS (Ground)

The circuit is built on the breadboard as the following circuit diagram. The programmed PIC microprocessor is used in the circuit as shown in the schematic. The A/D converter measures voltages between 0V and the supply voltage VDD, which is 5V in this case. The analog input may be a signal that swings above and below 0 volts, so its waveform will have positive and negative portions. In order to sample the input waveform with the A/D converter, the input signal needs to be shifted. The voltage divider created by resistors R1 and R2 sets the bias at half of the 0 to VDD A/D input range. The capacitor C1 couples the AC input signal to the A/D input, so now the input waveform swings about ½ VDD instead of swinging about 0 volts. R3 and R4 form a voltage divider used to set the target frequency. The processor reads the voltage at the frequency select pin and uses the value to determine the correct coefficients needed by the algorithm to determine if the target frequency is present. R5 and R6 form a voltage divider used to set detection threshold. This allows the user to select the magnitude of the algorithm output that is necessary to make the program turn the output HIGH. The processor reads the voltage at the detection threshold pin and uses the value to determine the detection threshold. C2 is a decoupling capacitor used with the microprocessor to keep the VDD supply free of spikes

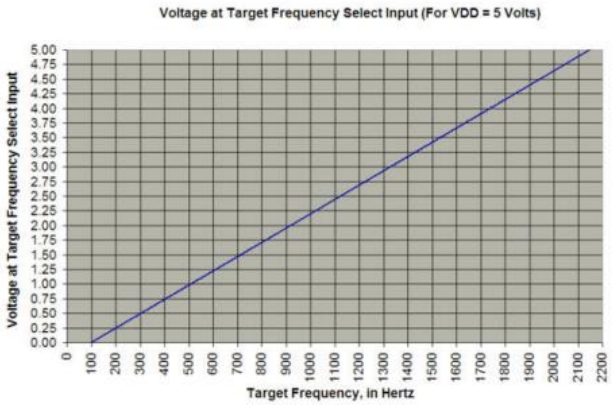
## VI. Target Frequency Selection

The target frequency is selected by means of a voltage applied to the frequency select input. The voltage on this input is ready by the microprocessor's A/D converter, and from this reading the program configures itself to decode the corresponding frequency. The frequency range that can be decoded is between 100Hz and 2148Hz. The program always reads the target frequency select input after power up. After that, the target frequency select input is ignored, unless the target frequency select input enable is brought low. So, to make changes to the target frequency during program operation, the enable input must be low. It needs to be kept low for at least 100ms, as the input is checked at the beginning of the program loop. Once the input is brought high again, the program will ignore the target frequency select input, and the target frequency in effect will be that established while the enable pin was low. A voltage divider is used to set the correct input voltage on the pin to select the desired target frequency.

V_target_frequency_select_input= VDD*(Target_Frequency -100)/ 2048

VDD is the power supply voltage (usually 5 Volts). The valid range of frequencies for the program as written is 100Hz to 2148 Hz. So, for a target frequency of 750Hz, and VDD of 5 Volts, the voltage applied to the frequency select input should be:

V_target_frequency_select_input = 5*(750 -100)/ 2048 = 1.59 volts

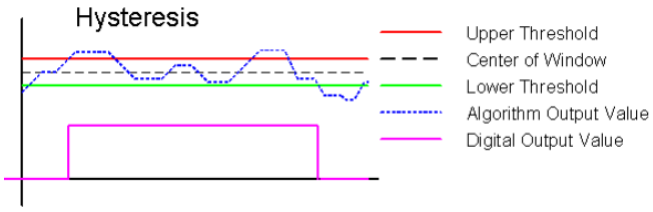



The graph shows the target frequency selected versus voltage on the target frequency select input for a VDD of 5 volts. The easiest way to set the voltage is using a voltage divider, as shown in the schematic.

The voltage at the output of a voltage divider is: Vout=VDD*[R4/(R3+R4)] The exact values of the resistors is not critical, only the ratio of their values. However, to prevent noise from upsetting the frequency select input, it is best to choose lower resistor values. For the prototype, we chose the values to be 1k and 2.2k.

## VII. Detection Threshold Settings:

The detection threshold is set by means of a voltage applied to the detection threshold input. The voltage on this input is ready by the microprocessor's A/D converter, and from this reading the program sets the detection threshold




The value read from the threshold setting input establishes the center value of a hysteresis window, with the upper and lower thresholds sitting symmetrically above and below the center. The output value from the algorithm must exceed the upper threshold to cause the detect output to go high, and the output value must drop below the lower threshold before the detect output will go low. The smaller variations in output value above and below the center of the window do not cause the detect output to change. The software does not allow the detection threshold to be set all the way to zero, as this would cause the detection output to go high for any small amount of noise on the signal input. The processor uses the VDD power supply on pin 1 as the reference for the A/D converter. If a different VDD is used the output of the algorithm for a given signal input level will change proportionately. A voltage divider is used to set the correct input voltage on the pin to select the desired detection threshold

To calculate the voltage to apply to the detection threshold input to establish the minimum upper threshold required to detect an input signal of a certain amplitude at the target frequency, use the following equation.

V_dection_threshold_input=VDD*[342*V_signal_at_target_freq – 27] / 1024

So, to set the detection threshold so that the detection output goes high for an input signal of 1 volt (peak) or greater at the target frequency, for a VDD of 5 volts, the voltage at the detection threshold input should be:

V_dection_threshold_input = 5 * [342 * 1 -27] / 1024 = 1.54 Volts

The graph below shows the upper and lower thresholds of the hysteresis window versus voltage on the detection threshold input for a VDD of 5 volts. The easiest way to set the voltage is using a voltage divider, as shown in the schematic. The voltage at the output of the voltage divider is:

Vout=VDD*[R6/(R5+R6)]

The exact values of the resistors is not critical, only the ratio of their values. However, to prevent noise from upsetting the value read by the input, it is best to choose lower resistor values. For the prototype we chose the values to be 1k and 2.2k.

## VIII. CONCLUSION.

A prototype of a system was designed and constructed that would detect the presence of an EMV, and turning the corresponding signal green, thus creating a green corridor for the EMV. For an emergency response vehicle, reaching its destination in the least amount of time is critical, and every second of journey time reduced makes a big difference. Hence by designing an IoT based system, which minimizes human-human and human-machine interaction, thus clearing

its path more efficiently, we believe that this is a step forward in saving thousands of lives every year, and at the same time, creating a smart city and hence a smart nation.

## REFERENCES


[1] F. Andronicus and Maheswaran. “Intelligent Ambulance Detection System”.

[2] Dimitris K. Fragoulis and John N. Avaritosiotis. “A Siren Detection System based on Mechanical Resonant Filters”.

[3] . F. Meurcci, L. Pierucci, E. Del Re, L. Lastrucci and P.Desii. “A REAL-TIME SIREN DETECTOR TO IMPROVE SAFETY OF GUIDE IN TRAFFIC ENVIRONMENT”

[4] Takuya Miyazaki, Yuhki Kitazono and Manabu Shimakawa. “Ambulance Siren Detector using FFT on dsPIC”

# IoT End User Programming Models

Steven P. Reiss
Department of Computer Science
Brown University
Providence, RI 02912
spr@cs.brown.edu

## ABSTRACT

The advent of smart devices and sensors (the Internet of Things or IoT) will create increasing demands for the automation of devices based on sensor, time, and other inputs. This is essentially a programming task with all the problems and difficulties that programming entails, for example, modularity, feature interaction, debugging, and understanding. Moreover, much of the programming for smart devices is going to be done not by professional programmers but by end users, often end users without any programming experience or computational literacy. Our research is aimed at exploring the programming space and the associated issues using a case study of a smart sign that can be controlled using a variety of sensors. We have developed a general system for programming smart devices and, in this paper, explore a variety of different user interfaces for programming this system for our smart sign.

## CCS CONCEPTS

**Software and its engineer**ing → **Software creation and management**; Software development techniques

## KEYWORDS

Internet of things; end-user programming; debugging; program understanding.

## 1 INTRODUCTION

Imagine a "smart house" where everything is driven by software. Lights are not directly wired to switches; thermostats are advisory and not directly connect to the furnace; buttons exist to request open windows or skylights; doors may be unlocked by RFID keys (as in today's automobiles); etc. The owners of such a house would need to program it and keep the program up-to-date as new devices are added or as their needs change. Moreover, the owners are generally not going to be software engineers and will not necessarily understand programming or programming concepts.

Programming for such a smart house raises several issues. The first is what form a program for such a smart house should take. We lean toward a rule-based system since rules are generally easy to specify and can map directly to user interactions. However, the conditions and rules that are needed can be complex and involved.

The second issue is handling conflicts when creating rules and when attempting to understand what will happen based on the current set of rules. Any programming system for smart devices needs to have a scheme for addressing conflicts and needs to make such conflicts explicit and understandable to the end user.

A third issue is that the users will not create the perfect program initially. This means that the user interface for programming needs to handle debugging, particularly debugging with respect to potential conflicts.

The fourth issue is one of scale. While our simple case study only involves twenty-some rules, a smart house with tens of devices might require hundreds of rules. Can an interface be designed that can handle this large a set? Will end users be able to understand a "program" at this scale? What techniques will help in such understanding?

This paper describes our initial explorations of how to program smart devices on the scale needed for a smart house. We start by describing some of the related work in Section 2. Then we describe a case study, our smart sign, in Section 3. Next we describe what we see as a practical programming framework for smart devices based on a rule-based model in Section 4. Next we provide a set of four exploratory solutions we developed using our smart sign as an example in Section 5. We conclude by describing our experiences.

## 2 RELATED WORK

Pervasive computing has been touted for a long time. In the past few years, this technology has become more evident in the "Internet of Things". Here everyday devices are web-enabled so they can talk and potentially control one another. Individual smart devices, e.g. windows that close themselves when it rains, coffee makers that turn on in the morning, thermostats that program themselves, are become more common. Internet-based control of individual devices in the home is becoming common with major companies such as Verizon producing practical control systems. Smart houses were first popularized in science fiction (e.g. the television series The Jetsons), but have now become a reality. This has opened many new research questions and directions [32].

The trends we see here are moving from physical controls (such as switches and locks) to virtual controls (such as a phone app); moving from user-programmed devices to learning-based programming; and moving from explicit control to automated control. We see the trend in terms of the Internet of Things moving to devices which essentially control themselves according to the user's wishes.

Most of the techniques for programming devices and home automation are based on production systems. Production system have been widely studied over the years, are well understood, and have been used in a wide variety of applications [22]. The most widely used rule-based model for programming devices is trig-

ger-based. This is exemplified by IFTTT ("IF This Then That") [17]. It is both widely used and has been extensively studied. For example, [46] show that it is easy to learn and to program. Our rule-based approach builds on this. However, purely trigger-based approaches do not really cover all aspects and in many cases are not natural ways of specifying interactions [46,47]. This led us to develop our more general framework.

Moving from individual web-enabled devices to a larger set of interacting devices, for example, in a smart house, is non-trivial. Different sensors might trigger the device in different ways at the same time; different device settings might conflict with each other; there can be implicit constraints (e.g. do not heat and cool at the same time; do not turn on both the coffee maker and the microwave since that will blow a fuse) that should be enforced. This problem, the complexity involved in interacting devices and conflicts, has been noted by several others [3,20,24]. Others have noted that it is important in the user interface to deal with unusual situations (i.e. deviations from the routine) [12,31,49]. Our interfaces attempt to make such conflicts and interactions explicit.

The DeLorean system takes another approach to handling conflicts [9]. It assumes a rule-based language with trigger-based rules with arbitrary conditions and actions. Based on this language, they use timed-automata to let the user fast-forward their system in order to understand conflicting behaviors. This, and other model-checking based approaches are aimed more at the problem of predicting unusual behaviors while we are more interested in letting do their own exploration. Our interfaces attempt to take a more practical approach to showing conflicts, but do not preclude the use of formal checking.

Explanations of the behavior of rule-based systems are common [4,13,14]. Lim, Dey, and Avrahami [30] provide the user with explanations of the behavior of complex systems that are rule or machine-learning based and note that it can be useful to both show why an action was taken and to show why and action was not taken. Explanations for debugging are explored in WhyLine [23]. Explanations can also be used for debugging machine learning approaches [25]. Many explanations, especially for debugging, involve reachability questions [27]. Our interfaces include the ability to show what will happen and can easily be extended to show full explanations.

Learning interfaces for IoT devices are becoming more common as in the Nest thermometer [26]. The utility and practicality of such interfaces has been studied where it was shown that they should be combined with other interactive technologies for handling exceptions, providing descriptions, and user engagement [49]. Rule recommendation systems work along similar lines [48]. Our approach includes the ability to learn new rules based on user requests.

End-user programming [7,8,19,35,37,38] has a long history. Spreadsheets are the prime example of a very successful end-user programming system, employing a metaphor that is easily understood and enabling a wide range of programs. There has been significant work on debugging spreadsheets from this perspective [1,5,6,25]. Teaching programming concepts, especially to children, has been one theme for end-user programming. This started with Logo and has progressed to languages such as Alice [11], Scratch [40], Squeak [18], and similar systems [21]. A recent example is Toque [45], which uses cooking as a basis. These systems attempt to make complex constructs intuitive to the user. End-user debugging is also used for web applications [16]. Natural programming languages and end-user oriented environments are another approach [34,36]. Our approach draws inspiration from all of these.


**FIGURE 1. The automated sign outside my office.**

Another theme of end user programming has been programming-by-example [2,15,29,44] or programming-by-demonstration [10,33,41-43]. This has been used for IoT devices to a limited extent [28,39] Our learning interface provides an basic implementation of programming-by-demonstration for IoT devices.

## 3 SMARTSIGN: A CASE STUDY

I have a sign outside my office. Actually its not a sign but a inexpensive 10" Android tablet running a kiosk application. (It was originally a Bluetooth digital picture frame.) The sign displays my current status, indicating whether I am available, whether I'm with someone, whether I'm on the phone, or, if I'm out of the office, when I'm likely to be back. It does this all automatically. The sign can be seen in Fig. 1. The sign has been running continually for about 3 years at this point and serves as an interesting, well-tested, case study.

The sign software detects whether I'm in my office by seeing if it can see my phone via Bluetooth. It determines if I have a visitor by using a motion detector aimed at my seating area. It determines if I'm on the phone using an off-the-hook circuit attached to my phone line. It accesses my Google calendar to determine if I'm at a meeting or on vacation and when I'll be back. It accesses the web to determine the current weather conditions and temperature.

The sign works off a set of about twenty-five rules. The rules have a relatively simple form, for example,

IF <in_office> AND <time between 8:30am and 5pm, Monday thru Friday> THEN <I'm available>
IF <visitor> THEN <With visitor>
IF <in_office> THEN <I'm hiding>

This form consists of a set of conditions and then the action to perform when the conditions hold. In the case of the sign, the action is simply to display an image (created as an SVG diagram) which is done by updating an image for a web page. The conditions include the basic ones determined by sensors (e.g. *in_office*), time-based conditions, Google-calendar based conditions, weather-based conditions, and some artificial conditions that

were created to simplify the rules. Examples of the latter include a condition which is true if I've been out of the office for less than five minutes (likely stepped out to use the facilities or get a soda), and one that is true if I've been in at all during the day.

The rules are given in priority order, so that the first rule whose conditions are satisfied is applied. Thus, with the above three rules, the last one is applied if I'm in the office and it is not during what I consider work hours.

In addition to displaying my current status on the sign display, the system also provides my current status on my personal web site (http://www.cs.brown.edu/people/spr/status.html).

## 4 PROGRAMMING CONSIDERATIONS

There are many different alternatives for control languages for embedded or smart-house applications. One could start with an actual programming language, for example Python, and add basic methods that check conditions and implement controls. This is what is done, for example, in Sikuli [50] for user interface testing. However, given the difficulties involved in understanding and teaching computation, this does not seem to be a feasible approach for the average homeowner who knows nothing about programming. Alternatively, one could work in terms of finite state machines, which are a natural model for many devices. However, these again present a not-easily-understood formalism that could be confusing to non-programmers and would require extensions to handle time-based conditions.

A simpler alternative is a rule-based approach, essentially a production system. Here there are a set of rules that are checked in some order. The rules consist of a condition and action. Rules are evaluated by checking the condition, and then, if it holds, taking the corresponding action. Even with this constrained framework, there are several design alternatives that need to be considered.

The first alternative is whether rules are considered as triggers or as conditions that are (in theory at least) continuously checked. An common implementation of a rule-based approach for devices or home automation is IFTTT ("IF This Then That") [17]. This consists of conditions that are triggers and an action that should be evaluated when the trigger conditions occur. This type of rule will be helpful in a smart house (e.g. send me a text message if the burglar alarm goes off), but is not sufficient. Most rules for our sign (and eventually for a smart house) are designed to be evaluated continuously. Consider, for example, "If I'm in the office then display 'I'm available'. This is more natural than saying "When I enter the office, display 'I'm available' since the condition is designed to hold whenever I'm in the office, not just when I enter it. Moreover, other displays are possible while I'm in the office (e.g. On the Phone, With a Visitor, Hiding) and defining trigger-based transitions for all such events would be tedious and error-prone. Moreover, such events also need to consider time (the actual rule includes normal office hours), which make a trigger-based approach even more complex. Moreover, one can easily create conditions where the number of trigger-only rules could grow quite large. For example, consider N light switches and N lights where each switch turns on a different subset the lights (e.g. each light is based on OR conditions over a subset of the switches).

Given that both trigger-based and continuous rules are needed, a second design alternative involves how to handle multiple triggers and how to distinguish between trigger and continous rules. It is not uncommon for users to try to create rules of the form "If the doorbell rings at 3:00pm" [47]. Such rules will almost never be triggered since it is unlikely that someone will ring the door at precisely (to the millisecond) 3:00pm. Similarly, using trigger-like actions (e.g. send an email) under continuous conditions could result in unexpected or unwanted results.

To avoid these situations, our approach distinguishes both conditions and actions as either continuous or trigger-based. A rule can either include zero or one trigger-based condition. If it has none, then the action has to be continuous. If it has one, then the action has to be trigger-based. The front end is also aware of these constraints and enforces them as the user creates rules.

The third design alternative is what types of conditions are allowed. It is easy to code the system to allow arbitrary logical conditions, i.e. combinations of AND, OR, and NOT. However, this quickly becomes confusing without parenthesis even to the experienced programmer since there is no implicit priority ordering for these operators as there is for addition and multiplication. Moreover, creating a easy-to-use interface that supports all three also becomes difficult. Based on our initial experiences attempting to develop interfaces for our sign, we determined that a more viable alternative was just to allow AND rules. This makes the rules easier to comprehend and the interface more tractable.

A problem with only allowing AND rules is that there are conditions that are naturally OR conditions, for example turn on light 2 if either switch A is on OR switch B is on. Simple OR conditions like this can easily be handled by using multiple rules. To handle more complex situations, our experience has shown that it is often easier to create new basic conditions (effectively pseudo-sensors) than to create more complex rules. For example, we have both an On-Phone condition and a Not-On-Phone condition. For the switch, one could create a new sensor which was "Either A or B is on".

New sensors are useful for other applications as well. For example, we have defined sensors that combine other sensor states with time for use with our sign. One is whether we have stepped out of the office for less than five minutes; another is whether we have come in at all during the day. Such sensors will also be useful within a smart house (i.e. for turning off lights when there is nobody around for some time or automatically setting the alarm to external-doors only if someone is in the house at night). We note that such new sensors can be used to turn a trigger-condition into a continuous condition when needed.

The current front end allows new sensors to be defined interactively for durations (some event occurring for either more than or less than a given interval); for latches (some event occurring at all with a given timeout or reset time); and for combinations (ORs of events). Other sensors, for example those based on RSS feeds and web pages are implemented, but the front end for defining them is still missing since we have not found them particularly useful for our smart sign. The implementation also includes a generic sequence sensor that detects and ordered sequence of events and maintains a corresponding internal state (essentially a finite state machine), although again, we have not implemented the corresponding interfaces for building such sensors. More complex new sensors can be defined programmatically as needed.

A fourth design alternative involves determining what rules (assuming a priority order) should be applied as conditions

change. With the sign, it is rather simple. The first rule (in priority order) whose condition is true will affect the sign and hence it is applied, and any subsequent rules are ignored since they would also affect the sign. However, in a smart house multiple rules affecting different, independent devices might be triggered by a single condition. For example, if it starts to rain each of the skylights should be closed. In this case, it seems logical to allow multiple active rules while still maintaining rule priorities for each device. One needs to make it explicit to the user, however, what rules preclude or do not preclude others as it might not be immediately obvious (for example if both close-window and close-curtain are triggered by different rules for the same window, would both occur or only one; similarly if a rule has multiple actions associated with it and one of those actions conflicts with a prior rule, should the other actions be taken or ignored).

A smart house will require a large rule set. It is essential for understandability that this rule set be organized in a meaningful way. The common notion in programming languages is to modularize the rules. This seems a logical approach. However, there are multiple ways that such modularization can be done. For example, one could show all rules that affect a given device, or one could show all rules that are affected by a given sensor, or all the rules that are might be in effect at a given time, or some combination of these. The fifth design alternative involves how to do this modularization, whether it should be done statically (i.e. rules are defined in modules), or dynamically (where the user specifies a device or condition or time and the system creates an implicit module for corresponding rules).

Modularization of the rules has other uses as well. It can, for example, make the implicit finite state machines embodied by the various devices in the house explicit. This is done by viewing each state in the machine as a "module" and letting the user define the rule set for just that state. This requires a flexible modularization approach and might require artificial sensors that reflect the implicit state of various devices.

A sixth design alternative involves handling complex sensors. With the sign we needed special sensors to handle time and access to ones Google calendar. Time was modeled by how events (and repeated events) are typically defined in modern calendar systems. This provides an interface that people might be accustomed to, but still might be overly complex for simple conditions. Events derived from a Google Calendar were a bit more complex since we had no example to build against. Our current implementation lets the user choose specific fields of the calendar event (e.g.Where, Who, State, Visibility, Title, Which Calendar) and define strings that should or should not occur in those fields. Again, for simplicity and understandability, the calendar event matches if all specified fields are matched. For example, we can create rules for meetings that are not in my office by looking for events where the WHERE field does not contain "403" (my office number). We expect that for a smart house there will be other sensors of similar complexity that will have to be defined and handled, for example using a camera to detect motion or the presence of people in a room.

We also have implemented sensors for accessing RSS feeds and web pages. The RSS feed sensor is triggered on each new feed item and makes the title and description available to rules. The web page sensor takes a URL, a check frequency, and a CSS-style selector for the text of interest. It is used to check weather conditions and temperatures.

## 5 SAMPLE PROGRAMMING INTERFACES

The programming considerations cited above need to be consolidated into a user interface or programming system that can be used to effectively control our sign or a smart house. As a first step toward exploring what is needed and what might work here, we prototyped a control system and four different web-based interfaces for programming the smart sign. These interfaces are designed to be automatically generated based on a description of the underlying conditions and available actions and hence should be able to be used for a smart house, provided they scale appropriately.

### 5.1 The Control System

To experiment with the different end-user programming interfaces we developed an underlying control system that supports a wide variety of rule types and thus can accommodate many of the alternatives described above. The rules supported by the system are based on the notion of input sensors and output actuators. While it is simple to state that the smart house consists of separate sets of sensors and actuators, this by itself is not a realistic model. Many devices are actually both sensors and actuators. Even a simple light bulb can provide input information as to whether it is burned out or not. A dimmer light switch might include a display showing the current light setting. An alarm panel will display the current state of its sensors and the alarm state as well as letting the user change state, bypass sensors, etc. For this reason, the control system is based on devices, with each device having 0 or more sensors which are externally viewable parameters, 0 or more internal parameters (hidden state), and 0 or more actions.

The control system supports a web-callable (RESTful) API that lets us define web-based front ends that can create arbitrary rules and get information about the system. To facilitate more complex interfaces, it includes the notion of a *hypothetical world*. Such a world can be created by cloning the current (real) world or an existing hypothetical world. Within a hypothetical world, conditions and time can be set arbitrarily, and actions can be taken without actually changing the real world. This lets external tools (and hence the user) explore the effect of the rule set without actually changing any devices. This can be used, for example, to provide an interface that would show the user what happen under different conditions, something that is very useful for both debugging and understanding the potential effect of new rules.

The control system is designed to handle more than the smart sign. It includes authentication to provide limited or selective access to the rule set. It includes sensor and rule types that are not used for the smart sign, notably trigger rules (one-shot) and sensors that act as finite state automata and maintain an internal state. It can handle multiple output devices. New devices and sensors can be added dynamically.

Using this control system we have built four potential interfaces to explore different user programming models that might be used in coding a smart house. These are a programmer-oriented interface, and interface specialized to rule creation, an interface for learning rules, and an interface to explore modularity. While the interfaces have been explored for the smart sign application, they are all generated automatically from a description of the

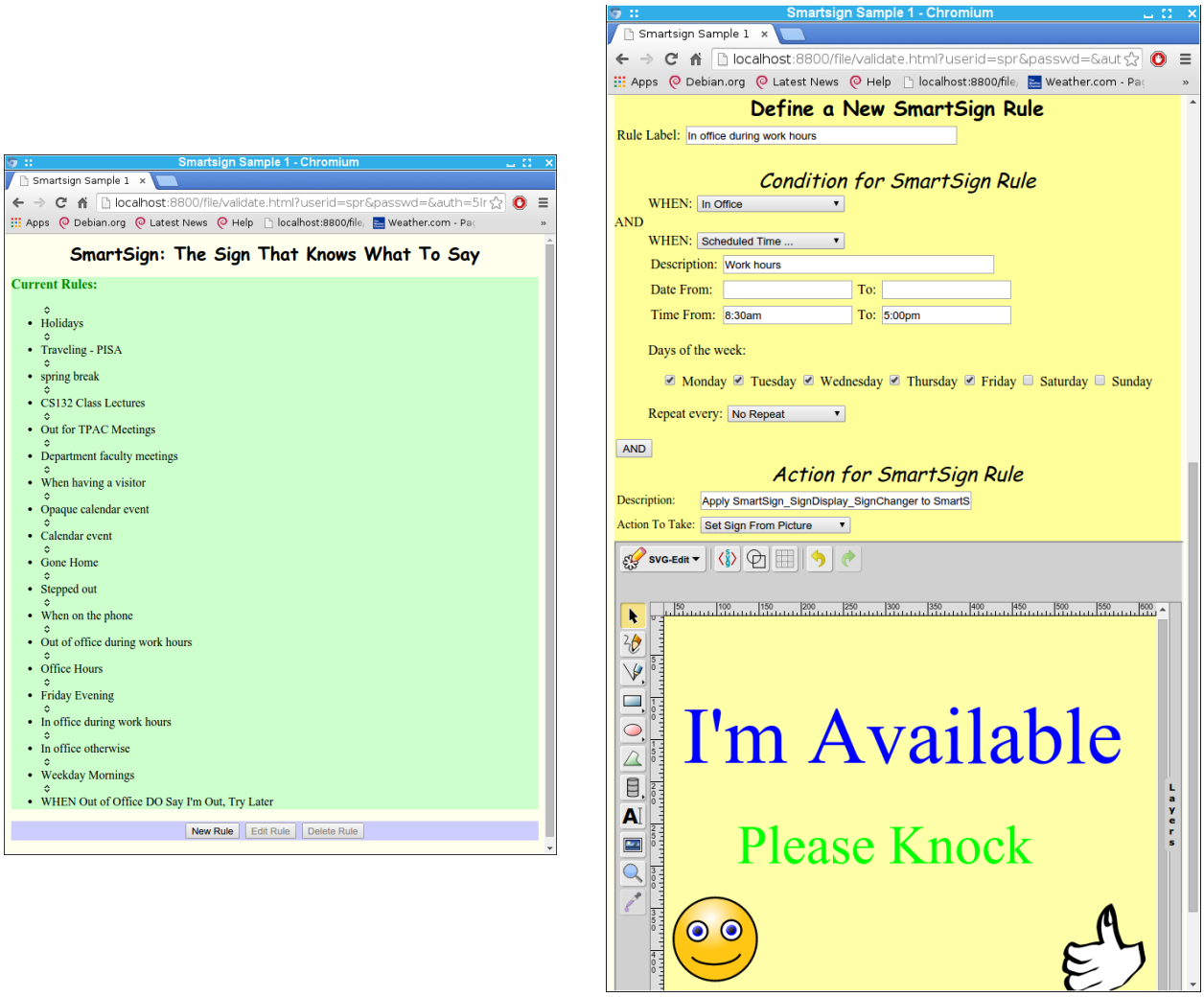



**FIGURE 2. Programmer Interface to the smart sign. The image on the left shows the current rules and lets the user drag and drop rules to change priorities. The image on the right shows the interface for defining a new rule or editing an existing one.**

devices and sensors and not specialized to the smart sign. We note that these are experimental prototypes and not professionally designed interfaces.

## 5.2 The Programmer Interface

The first interface, shown in Fig. 2, is what we call the programmer interface. It is designed for those who understand production systems and want complete control over the system. It consists of the list of all rules in priority order (using labels provided by the user when the rules were defined) where the user can drag rules around to change priorities. In addition, one can select a rule to edit, can delete a rule, and can define a new rule. Selecting "New Rule" or "Edit Rule" brings up a dynamic display seen on the right of the figure where the user can provide one or more conditions and then use an SVG editor to define the image for the sign.

This is the interface that we generally use personally when programming the sign (we are programmers after all). The rule listing part is relatively simple, and with twenty-some rules. The rules generally fit on one web page and it is relatively easy to find the rule one is looking for. The interface is not particularly good, however, if one is not already familiar with the existing rules since it provides no hints as to placement of a new rule. This has led to several occasions where we had to debug new rules because they didn't operate as intended in unusual situations. Moreover, this interface will not scale easily to systems with larger sets of rules.

## 5.3 The New-Rule Interface

The second interface, shown in Fig. 3, we call the new-rule interface since it emphasizes defining new rules and understanding their impact. It consists of the interface for creating a rule used in the programmer interface at the top and then a display of all the rules at the bottom that this potential new rule would conflict with and, by default, would be overridden if the rule were created as specified. The idea is that the user starts by specifying some basic trigger or condition for the new rule. As conditions are added, the set of conflicting rules is narrowed down. If the user decides that this rule should not override all of the conflicts that are shown, they can add additional conditions to specify when this rule should be applied or can select where the new rule should go in the listed hierarchy. When the user is satisfied that

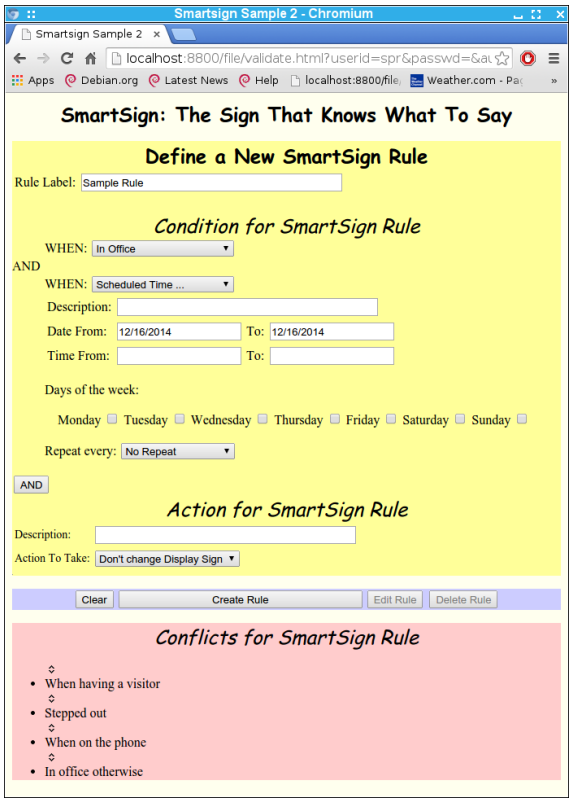



**FIGURE 3 New-Rule interface to the smart sign. This interface lets the user start defining a new rule by specifying its conditions. As they do so, the set of conflicting rules is displayed at the bottom.**

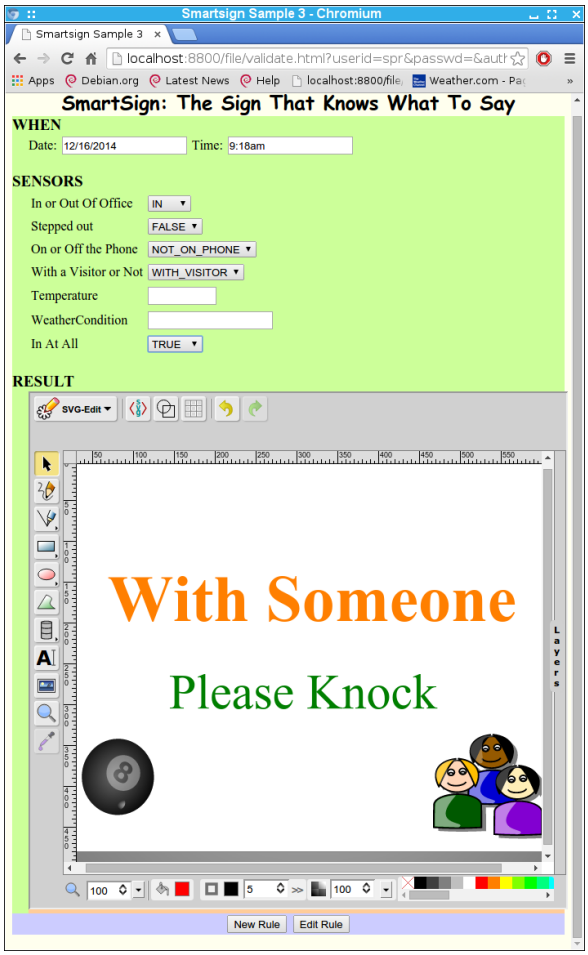



**FIGURE 4The learning interface. The display shows a set of conditions (by default the current ones), and then shows what the current rules would result in under those conditions. The user can then change the display to indicate what the state should be. The system will deduce new rules based on the user specification.**

the new rule is appropriate (i.e. that all the conflicting rules should indeed be overridden by this rule), they can click on the Create Rule button. Selecting an old rule before clicking this button causes the new rule to be added after that rule rather than before all the listed rules.

This interface scales somewhat better than the programmer interface. For the smart sign, it does reasonably well, because all the listed rules are essentially relevant to the sign and can quickly narrow the set of rules to understand conflicts. For a smart house, there might still be a large set of rules, many of which were irrelevant to what the user was intending. Another aspect of the interface is that the placement of the new rule is relative to the given rules, not absolute. This means that the actual priority of the new rule is not obvious to the user.

## 5.4 The Learning Interface

The third interface, shown in Fig. 4, we call the learning interface. It is loosely based on devices such as the Nest thermostat [26]. It starts with a display where the user can specify the time and a set of conditions. (Google calendar conditions are actually read from the user's calendar based on the time specification.) The time can be in the past or future and defaults to the current time. Based on the condition settings, the system computes what would be displayed and shows that to the user. (This uses the hypothetical worlds feature of the controller.) The user can then edit the displayed state and hit "New Rule". ("Edit Rule" changes the output for the rule that caused the display.) This causes the system to remember the conditions under which this rule should be applied. The system then builds a decision tree based on these conditions, any prior conditions that were defined using this interface, and the current set of rules defined using the other interfaces.

Creating a decision tree is done using the standard machine learning algorithm. The system uses the resultant decision tree to determine appropriate conditions for triggering all the rules created using this interface and then creates the corresponding rules with appropriate priorities. The decision tree approach is relatively straightforward, but there are unsolved problems.

The main problem is that the interface has the user specify a particular instance in time. This makes sense when exploring the rule set and for handling time-based external sensors such as cal-

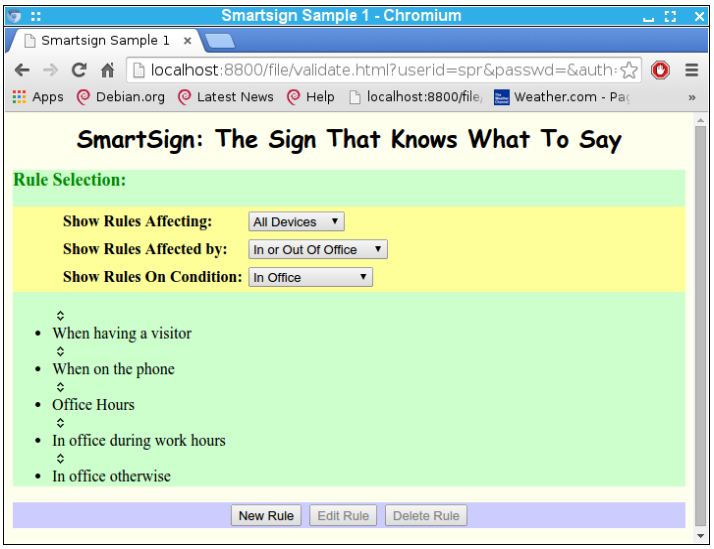



**FIGURE 5 The modular interface. Here the user chooses a device, sensor or condition and the system displays the rules relevant to that object. The user can then change the relative priority of the given rules, can edit one of the relevant rules, or can create a new rule relevant to the context.**

endar events. However, the rules are based on time intervals, not time instances. This leads to the difficult problem of converting the time instant from the specification into an appropriate time interval. This is currently done by keeping a history of sensor changes over time and using this history to determine what a reasonable interval should be for this rule and whether there are any repeats. Our experience to date is that this method is not particularly effective, possibly because our schedule is somewhat erratic.

Because many rules are time-based and the problem of inferring time is difficult, this interface has not been used extensively for creating rules. However, it has proven effective for understanding what this system is doing and hence for debugging purposes. One can use it to easily explore what will happen under different conditions, to see if the rules are correct, and to determine if new rules might be needed. It would be better for such exploration if it also showed which rule(s) were being triggered (although this is generally obvious for the sign from the display), and provided a means for checking why a particular rule was not triggered.

### 5.5 The Modular Interface

The fourth interface, shown in Fig. 5, we call the modular interface. It supports dynamic modularization of the rules sets. It starts by letting the user choose how to modularize, either by device, by sensor, or by condition, or some combination of these. (It currently does not support modularization by time.) Once the user chooses the modular condition, the rule set relevant to that condition is displayed as it was in the programmer interface. Now the user can rearrange the rules for this modularization, can edit one of these rules, or can create a new rule that is appropriate to the modularization.

This interface provides a convenient means for scaling the simple programmer interface to large sets of rules. However, it still requires understanding priorities and does not provide feedback on what will happen under different conditions or what rules conflict with each other. Another aspect of this interface that might be confusing to the user is what happens to the priority of a modular rule with respect to rules not displayed when the priority is changed.

A similar modular interface could be created for new rule and learning interfaces. Modularization in these cases would be most useful for particular devices.

## 6 EXPERIENCE

Our smart sign has been running for several years at this point, providing us with enough experience to understand many of the design issues and to realize potential next steps.

The current rule set is relatively stable. We edit the rules about once a month, generally to take into account travel, vacation, and recurrent events that change each semester, and then generally to reset the times associated with the affected rules. This generally involves editing existing (or previously used) rules, not creating new ones from scratch.

The underlying control system is stable and robust. The web front ends, a little less so. This is particularly true for the SVG editor, which is a bit quirky, and can create SVG that can not be converted to an image directly. The interfaces are still a bit primitive and could use a good redesign. The main continuing issues we have found are in the connectivity with the tablet display and the Bluetooth on our phone.

We see several directions for extending these efforts. One is to define meta-rules that can generate low-level rules automatically (e.g. in a room with a ceiling light and a wall switch, the switch should control the light). Another is to be able to specify properties and validate that the rule set enforces those properties. A third involves determining how to do learning effectively, particularly learning of time slots.

## 7 REFERENCES

1. Robin Abraham and Martin Erwig, "Goal-Directed Debugging of Spreadsheets," pp. 37-44, in *Proceedings of the 2005 IEEE Symposium on Visual Languages and Human-Centric Computing* (2005).
2. G. Attardi and M. Simi, "Extending the Power of Programming by Examples," in *Integrated Interactive Computing Systems*, ed. P. Degano and E. Sandewall, North-Holland (1982).

3. Kenneth H. Braithwaite and Joanne M. Atlee, "Towards automated detection of feature interactions," pp. 36-59, in *Feature Interactions in Telecommunications Systems*, ed. Weit Bouma and Hugo Velthuijsen, IOS Press (1994).

4. Andrea Bunt, Matthew Lount, and Catherine Lauzon, "Are explanations always important?: a study of deployed, low-cost intelligent interactive systems," pp. 169-178, in *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces* (2012).

5. Margaret Burnett, Bing Ren, Andrew Ko, Curtis Cook, and Gregg Rothermel, "Visually testing recursive programs in spreadsheet languages," *IEEE Symposium on Human- Centric Computing Languages and Environments* (September 2001).

6. Margaret Burnett, Curtis Cook, Omkar Pendse, Gregg Rothermel, Jay Summet, and Chris Wallace, "End-user software engineering with assertions in the spreadsheet paradigm," *International Conference on Software Engineering 2003* (May 2003).

7. Margaret M. Burnett and Brad A. Myers, "Future of end- user software engineering: beyond the silos," pp. 201-211, in *Proceedings of the on Future of Software Engineering* (2014).

8. Margeret M. Burnett, "End-user development," in *The Encyclopedia of Human-Computer Interaction, 2nd Edition*, ed. Mads Siegaard and Rikke Friis Dam, The Interaction Design Foundation (2014).

9. Jason Croft, Ratul Mahajan, Matthew Caesar, and Madan Musuvathi, "Back to the future: Forecasting program behavior in automated homes," *Microsoft Technical Report MSR-TR-2012-131* (May 2012).

10. Allen Cypher, Mira Dontcheva, Tessa Lau, and Jeffrey Nichols, *No Code Required: Giving Users Tools to Transform the Web*, p. 512, Morgan Kaufmann Publishers Inc. (2010).

11. Wanda P. Dann, Stephen Cooper, and Randy Pausch, *Learning To Program with Alice*, Prentice Hall Press (2008).

12. Scott Davidoff, Min Kyung Lee, Charles Yiu, John Zimmerman, and Anind K. Dey, "Principles of smart home control," *Proceedings of the 8th International Conference on Ubiquitous Computing*, pp. 19-34 (2006).

13. Kate Ehrlich, Susanna E. Kirk, John Patterson, Jamie C. Rasmussen, Steven I. Ross, and Daniel M. Gruen, "Taking advice from intelligent systems: the double-edged sword of explanations," pp. 125-134, in *Proceedings of the 16th International Conference on Intelligent User Interfaces* (2011).

14. Shirley Gregor and Izak Benbasat, "Explanations from intelligent systems: theoretical foundations and implications for practice," *MIS Quarterly* **23**(4), pp. 497-530 (1999).

15. D. C. Halbert, *An example of programming by example, Ph.D. dissertation, Department of Computer Science, University of California, Berkeley (1981).*

16. Mouna Hammoudi, Ali Alakeel, Brian Burg, Gigon Bae, and Gregg Rothermel, "Facilitating debugging of web applications through recording reduction," *Empirical Software Engineering* (2017).

17. IFTTT, "If This Then That," http://ifttt.com (2014).

18. Dan Ingalls, Ted Kaehler, John Maloney, Scott Wallace, and Alan Kay, "Back to the future: the story of Squeak, a practical Smalltalk written in itself," *Proceedings ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications ,97*, pp. 318-326 (1997).

19. Andri Ioannidou, Alexander Repenning, and David C. Webb, "AgentCubes: Incremental 3D end-user development," *J. Vis. Lang. Comput.* **20**(4), pp. 236-251, Academic Press, Inc. (2009).

20. Michael Jackson and Pamela Zave, "Distributed feature composition: a virtual architecture for telecommunications services," *IEEE Transactions on Software Engineering* **24**(10), pp. 831-847 (1998).

21. Caitlin Kelleher and Randy Pausch, "Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers," *ACM Computing Surveys* **37**(2), pp. 83-137 (June 2005).

22. D. Klahr, P. Langley, and R. Neches, *Production System Models of Learning and Development*, MIT Press (1987).

23. Andrew J. Ko and Brad A. Myers, "Debugging reinvented: asking and answering why and why not questions about program behavior," *International Conference on Software Engineering 2008*, pp. 301-310 (May 2008).

24. Mario Kolberg, Evan Magill, Dave Marples, and Simon Tsang, "Feature interactions in services for Internet personal appliances," *Proceedings of the IEEE International Conference on Softare Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, pp. 2613-2618 (2002).

25. Todd Kulesza, Margaret Burnett, Weng-Keen Wong, and Simone Stumpf, "Principles of Explanatory Debugging to Personalize Interactive Machine Learning," pp. 126-137, in *Proceedings of the 20th International Conference on Intelligent User Interfaces*, ACM, Atlanta, Georgia, USA (2015).

26. Nest Labs, "Nest Thermostat," http://nest.com/thermostat (2014).

27. Thomas D. LaToza and Brad A. Myers, "Developers ask reachability questions," pp. 185-194, in *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1*, Cape Town, South Africa (2010).

28. Toby Jaa-Jun Li, Yuanchun Li, Fanglijn Chen, and Brad A. Myers, "Programming IoT devices by demonstration using mobile apps," *International Symposium on End User Development (IS-EUD 2017)*, pp. 3-17 (June 2017).

29. H. Lieberman, "Designing Interactive Systems from the User,s Viewpoint," in *Integrated Interactive Computing Systems*, ed. P. Degano and E. Sandewall, North-Holland (1982).

30. Brian Y. Lim, Anind K. Dey, and Daniel Avrahami, "Why and why not explanations improve the intelligibility of context-aware intelligent systems," pp. 2119-2128, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2009).

31. Sarah Mennicken and Elaine M. Huang, "Hacking the natural habitat: an in-the-wild study of smart homes, their development, and the people who live in them," *Proceedings of the 10th International Conference on Pervasive Computing*, pp. 143-160 (2012).

32. Sarah Mennicken, Jo Vermeulen, and Elaine M. Huang, "From today,s augmented houses to tomorrow,s smart homes: new directions for home automation research," *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pp. 105-115 (2014).

33. Brad A. Myers, Brad Vander Zanden, and Roger B. Dannenberg, "Creating graphical interactive application objects by demonstration," *Proceedings UIST ,89*, pp. 95-104 (November 1989).

34. Brad A. Myers, John F. Pane, and Andy Ko, "Natural programming languages and environments," *Commun. ACM* ***47***(9), pp. 47-52, ACM (2004).

35. Bonnie A. Nardi, *A Small Matter of Programming: Perspectives on End User Computing*, MIT Press (1993).

36. Stephen Oney, Brad Myers, and Joel Brandt, "InterState: a language and environment for expressing interface behavior," pp. 263-272, in *Proceedings of the 27th annual ACM symposium on User interface software and technology*, ACM, Honolulu, Hawaii, USA (2014).

37. John F. Pane, Brad A. Myers, and Leah B. Miller, "Using HCI Techniques to Design a More Usable Programming System," p. 198, in *Proceedings of the IEEE 2002 Symposia on Human Centric Computing Languages and Environments (HCC,02)*, IEEE Computer Society (2002).

38. Fabio Paterno, "End user development: survey of an emerging field for empowering people," *ISRN Software Engineering article 532659* **2013** (2013).

39. Marissa Radensky, Toby Jia-Jun Li, and Brad A. Myers, "How end users express conditionals in programming by demonstration for mobile apps," *2018 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pp. 311-312 (October 2018).

40. Mitchel Resnick, John Maloney, Andres Monrow-Hernandez, and Natalie Rusk, "Scratch: programming for all," *Communications of the ACM* **52**(11), pp. 60-67 (Nov. 2009).

41. Robert V. Rubin, Eric J. Golin, and Steven P. Reiss, "ThinkPad: a graphical system for programming-by- demonstration," *IEEE Software* **2**(2), pp. 73-78 (March 1985).

42. Robert V. Rubin, Steven P. Reiss, and Eric J. Golin, "Compiler aspects of an environment for programming by demonstration," *Proceedings International Workshop on Visual Aids To Programming* (May 1986).

43. John Stasko, "Using direct manipulation to build algorithm animations by demonstration," *Proceedings SIGCHI Conference on Human Factors in Computing Systems*, pp. 307-314 (1991).

44. Phillip Dale Summers, "Program construction from examples," Yale research report 51 (1976).

45. Sureyya Tarkan, Vibha Sazawal, Allison Druin, Evan Golub, Elizabeth M. Bonsignore, Greg Walsh, and Zeina Atrash, "Toque: designing a cooking-based programming language for and with children," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2417-2426 (2010).

46. Blase Ur, Elyse McManus, Melwyn Pak Yong Ho, and Michael L. Littman, "Practical trigger-action programming in the smart home," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 803-812 (2014).

47. Blase Ur, Melwyn Pak Yong Ho, Stephen Brawner, Jiyun Lee, Sarah Mennicken, Noah Picard, Diane Schulze, and Michael L. Littman, "Trigger-Action Programming in the Wild: An Analysis of 200,000 IFTTT Recipes," pp. 3227-3231, in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, San Jose, California, USA (2016).

48. Beidou Wang, Xin Guo, Martin Ester, Ziyu Guan, Bandeep Singh, Yu Zhu, Jiajun Bu, and Deng Cai, "Device-Aware Rule Recommendation for the Internet of Things," pp. 2037-2045, in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, ACM, Torino, Italy (2018).

49. Rayoung Yang and Mark W. Newman, "Learning from a learning thermostat: lessons for intelligent systems for the home," *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pp. 93-102 (2013).

50. Tom Yeh, Tsung-Hsiang Chang, and Robert C. Miller, "Sikuli: using GUI screenshots for search and automation," *Proceedings UIST 2009*, pp. 183-192 (2009).

# A Systematic Mapping study on Internet of Things challenges

Aleksandr Lepekhin
Graduate School of Business and Management
Peter the Great St.Petersburg Polytechnic University
St.Petersburg, Russia
lepekhinalexander@gmail.com

Alexandra Borremans
Graduate School of Business and Management
Peter the Great St.Petersburg Polytechnic University
St.Petersburg, Russia
Borremans_ad@spbstu.ru

Igor Ilin
Graduate School of Business and Management
Peter the Great St.Petersburg Polytechnic University
St.Petersburg, Russia
Ivi2475@gmail.com

Sami Jantunen
School of Engineering Sciences
LUT University
Lappeenranta, Finland
sami.jantunen@lut.fi

***Abstract*—The challenge of developing IoT-based systems has been found to be a complex problem. It is influenced by number of factors: heterogeneous devices/resources, various perception-action cycles and widely distributed devices and computing resources. Increasing complexity and immaturity to deal with it have resulted in growing range of problems and challenges in IoT development. This paper identifies essential IoT-related challenges by conducting a systematic mapping study of existing IoT literature. To this end, we distil information with respect to IoT-related: 1) challenges, 2) experimental studies, and 3) recommendations for future research. We then discuss our findings in order to understand better the general state of IoT research, potential gaps in research, and implications for future research.**

***Keywords—Internet of Things, IoT, IoT Challenges, IoT Development, Systematic mapping study***

## I. INTRODUCTION

Due to advances in the field of Internet technologies and wireless sensor networks (WSN), a new trend in the era of ubiquity is being realized. The huge increase in internet users and the modification of interworking technologies allow the creation of networks of everyday objects called Internet of Things (IoT). Implementation of this technology is rapidly gaining momentum as technological, public and competitive pressures push firms to innovate and transform. The entire infrastructure of different domains changes with the introduction of IoT, creating a so-called intelligent infrastructure.

As IoT technology advances, an increasing number of firms have started to adopt the technology in different sectors: Health Care, Transportation, Waste Management, Food Supply Chains, and Energy & Utilities. IoT does not only bring certain changes in those domains, but it enables fundamentally new paradigm, which can be called as an extension and expansion of Internet-based network. It expands the communication from human and human to human and things or things and things. IoT is not subversive revolution over the existing technologies, it is comprehensive utilizations of existing technologies [1].

Development of IoT-based solutions is currently a widely discussed topic. The challenge of developing IoT-based systems has been found to be a complex problem. It is influenced by number of factors: heterogeneous devices/resources, various perception-action cycles and widely distributed devices and computing resources [2]. Another issue, pointed out by researchers, is the shift of focus in software development. It changes from goal-oriented to user-oriented, distributed intelligence and machine-to-machine and machine-to-human collaboration [3].

Increasing complexity and immaturity to deal with it have resulted in growing range of problems and challenges in IoT development. This paper identifies essential IoT-related challenges by conducting a systematic mapping study of existing IoT literature. To this end, we distil information with respect to IoT -related: 1) challenges, 2) experimental studies, and 3) recommendations for future research. We then discuss our findings in order to understand better the general state of IoT research, potential gaps in research, and implications for future research.

The structure of our study is as follows. Section 2 describes the research method. Section 3 presents the results of our systematic mapping study. Section 4 identifies recommendations for future IoT research and compares these findings with the results of Section 3. Finally, Section 5 concludes the paper.

## II. RESEARCH METHODS

We have conducted a systematic literature mapping [4]. This type of research requires a prior definition of the review protocol, that describes and justifies the research questions and outlines a method for studying, evaluating and synthesizing different types of research. In our study, following research questions were specified:

1. What IoT-related challenges have been reported?
2. How identified IoT challenges have been addressed?
3. What implications the findings have on future IoT research?

An important part of systematic mapping study is the specification of inclusion and exclusion criteria [5]. We decided to include Computer Science-related studies that described IoT challenges. Those studies that could not be accessed without additional investments were decided to be excluded from this mapping study.

Definition of the search strategy and data sources is the next step of systematic mapping study. Kitchenham [6]

suggests that researchers should indicate their rationale for using an electronic or manual search or a combination of both. In this study, we used electronic search procedure to identify the studies that help to answer the research questions and exclude human factor (when manual search is done). Electronic search procedure was chosen because it is represented as an essential contribution toward ensuring accuracy and completeness of the evidence [7]. The electronic search was applied on the Scopus search engine, because almost 80% of Scopus records include abstract, which makes analysis easier, and high quality of search outcomes [8]. The following search string was applied:

TITLE-ABS-KEY (iot AND challenges) AND (LIMIT-TO (ACCESSTYPE(OA))) AND (LIMIT-TO (SUBJAREA, "COMP"))

The search resulted with 79 open-access research papers (45 journal papers and 34 conference proceedings). Three papers were excluded from the analysis as they were country specific (India, South Korea, Netherlands). The sources of research papers are summarized in Table I.

TABLE I. IDENTIFIED PAPERS

| Publication Source (# of identified papers) | Included papers |
|---|---|
| ***Papers in Conference proceedings (34)*** | ***33*** |
| Procedia Computer Science (28) | 27 |
| Other Conference proceedings (6) | 6 |
| ***Papers in Journals (45)*** | ***43*** |
| Mobile Information Systems (8) | 8 |
| Eurasip Journal On Wireless Communication and Networking (6) | 6 |
| Digital Communications And Networks (3) | 3 |
| Procedia Manufacturing (3) | 3 |
| Other Journals (25) | 23 |
| **Total (79)** | **76** |

## III. RESULTS

Our systematic mapping revealed that the reviewed papers addressed six different categories of IoT challenges (Table II). This section describes these six types of IoT challenges and shows how the reviewed papers addressed these challenges.

TABLE II. IDENTIFIED CATEGORIES OF IOT CHALLENGES AND THE PAPERS ADDRESSING THESE CHALLENGES

| | |
|---|---|
| **IoT Opportunities in different contexts** | [9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25][26][27][28][29][30][31][32][33][34][35] |
| **Communication technologies** | [10][36][37][38][39][40][41][42][43][44][45][46][47][48][49][50][51][52][53][54][55][56] |
| **Interoperability** | [10][38][57][58][59][60][61][62][63][64][65] |
| **Security** | [10][11][13][34][66][67][68][69][70] |
| **Data and Privacy** | [3][15][16][22][30][63][68][71][72][73][74][75] |
| **IoT Development considerations** | [3][14][58][76][77][78][79][80][81][82][83] |

### *A. IoT Opportunities in different contexts*

IoT technology has already been utilized in a wide variety of contexts [9][10]. One of the most recognizable and notable contexts is healthcare and medicine [11][12][13][14][15][16][17][18][19]. IoT infrastructure has the potential to revolutionize the practice of medicine and transform how people manage their health [10]. Within this context privacy and security particularly important. To this end, [11] have developed a secure and efficient authentication and authorization architecture for IoT-based healthcare, and [16] sought to define a mechanism for prioritizing threats and aspects that are likely to be affected when devices may be added, removed, or changed. Since safety and privacy is heavily influenced by legislation, [13] analyzed relevant legislative frameworks and presented a case study how legislation affected the design of mobile application in digital health. Another challenge is to support the interoperability among several heterogeneous devices from different manufacturers. This challenge has been addressed by [12] with their implementation of a web middleware platform for connecting doctors and patients using attached body sensors. Such high degree of interconnectivity and sensitivity of data also raise ethical questions. These considerations are addressed in [14], which articulates various levels of trust among the concerned stakeholders in the service ecosystem and suggests value-sensitive design considerations, anchored on the principles of trust, for future IoT-enabled assistive care services.

Smart city is another context, where IoT is already heavily used. In some studies it is also called «Programmable city», highlighting its ability to develop smart physical infrastructure [20]. The amount of data in this context is considerably bigger than in any other context, bringing forward the challenges of data storage, management, security and analysis [21][22]. This challenge has been addressed by [23], with a framework proposal called Cloud-based Context-aware Internet of Things services in smart cities (C2IoT), that is intended to ease the deployment, development and offering of new smart services for the future smart city [23]. Urbanization context of IoT is linked to environmental issues, which created a concept of Green IoT [24][25]. IoT in Smart city is also considered in the context of transportation system. Researches analyze how intelligent transportation system can be implemented based on wireless networks [26][27].

IoT has also been argued to create opportunities to manufacturing industry for improving sustainability, performance and quality of production [28]. Such Industrial adaptation of IoT is often called Smart Factory and the trend of utilizing IoT in manufacturing is often referred as Industry 4.0 (German technological strategic initiative) [29][34][35]. A practical example of using IoT in manufacturing context is provided by [28], with a proposal of a smart injection molding system framework based on real-time manufacturing data considering the characteristics of injection molding processes, modules that compose the framework, and their detailed functions. As another example, [30] presented the design, implementation and proof of concept evaluation of an industrial, semantic Internet of Things positioning architecture, using low-power embedded wireless sensors.

IoT has also been used on more personal contexts. One of the well-known context is Smart Homes, that enables people to instrument their own homes [10][31]. The opportunities for using IoT in gaming context has been studied by [32]. Sports environment is viewed as another context of IoT adaptation, where IoT adoption could significantly improve

the sport experience and also the safety level of team sports [33].

### B. Communication technologies

Existing network protocols have largely been designed for stationary devices with reasonable computational and memory resources [10]. Since many IoT devices do not have access to a continuous power source, the way devices compute and communicate are significantly constrained. This creates needs for low-power communication. Furthermore, new network protocols and architectures need to also be able to support vast numbers of devices that are possibly mobile and that interact with the physical world, human users, and the cloud [10][36][37].

A review of currently used wireless data communication technologies have been provided in [38][39]. Our literature mapping identified several experimental studies that sought to improve communication in terms of scalability, power consumption and end-to-end delay by: 1) modifying Bluetooth low energy (BLE) retransmission model [40]; 2) reducing signaling cost of mobile IoT devices, by grouping devices with similar movement behavior with a PMIPv6-based group binding update method [41][42]; 3) reducing the number of transmitted packets by proposing Conditional Observation-function, where clients tell the servers the criteria for notifications [43]; 4) optimizing a Media Access Control (MAC) protocol for linear network topologies [44]; 5) introducing a new enhanced data streaming route optimization scheme that uses an optimized Transmission Control Protocol (TCP) realignment algorithm [45]; 6) developing an efficient scheme to estimate the number of unidentified tags for Dynamic Framed Slotted Aloha (DFSA) based RFID system [46]; 7) envisioning a scenario where many in-home sensors are communicating with a smart gateway over the BLE protocol, while at the same time harvesting RF energy transmitted from the gateway wirelessly via a dedicated radio interface [47]; 8) applying a multi-objective evolutionary optimization algorithm to reduce network energy consumption through optimization of sensor distribution [48]; 9) proposing a QoS based vertical handover scheme for M2M communications [49][50][51]; 10) presenting a novel data aggregation and multiplexing scheme for mobile M2M traffic [54]; 11) proposing novel access and scheduling schemes can reduce the collision rate dramatically during the IoT random access procedure and improve the performance of IoT communication [53]; 12) creating a scalable two-level index scheme (STLIS) for RDF data which can respond to the complex query in real time [55]; 13) developing an on-demand CSaaS to provide users with standardized access to various sensor networks and a level of abstraction that hides the underlying complexity [56].

The need to support Internet of Things (IoT) and device-to-device (D2D) communication also require a major paradigm shift in the way cellular networks have been planned in the past. This challenge has been addressed in [52].

### C. Interoperability

IoT would benefit from ways to let multiple applications (and sets of associated IoT devices) control their own fate over a shared network infrastructure [10]. This would lead to Internet of services that is highly dynamic and continuously changing due to constant degrade, vanish and possibly reappear of the devices [57]. In such scenario, we are shifting from code-heavy monolithic applications to a set of smaller, self-contained microservices, each offering narrowly focused, independently deployable services [58]. To this end, [59] presents challenges and technological enablers that will allow things to evolve and act in a more autonomous way, becoming more reliable and smarter.

Some of the essential challenges addressed in the reviewed papers are resource discovery and selection [60][61][57]. Another major challenge is to guarantee that the system is always configured correctly, despite of being in constant flux. It has been argued [10] that this can be achieved only with automation and with higher-level policy languages that allow users and administrators to specify high-level intent rather than configure low-level mechanisms. This challenge has been addressed by [59] with a presentation of architecture allowing things to learn based on others experiences and introducing situational knowledge acquisition and analysis techniques for things to be aware of conditions and events affecting IoT-based systems behavior.

To reduce the frequency and increase the effectiveness of debugging, IoT systems and devices will need to constantly monitor their own "health" [10]. To this end, IoT systems could adopt Structural Health Monitoring (SHM) processes of gathering basic information that allows detecting, locating and quantifying vulnerabilities early on (fatigue cracking, degradation of boundary conditions, etc.), thereby improving, the resilience of the IoT System [38]. When problems are detected, IoT systems need to be able to learn from errors and automatically repair themselves [10]. An example of developing methods to automatically achieve a self-configurable system, where associations and data can be adapted by self-learning is presented in [62]. Moreover, in order to provide heterogeneous connectivity, graph theory and system models can be applied for IoT network optimizations [64]. It was also noted that for supporting the task of high-level IoT applications the semantic-web based approach can be used [63]. To prevent compatibility problems, it is proposed to use the model-driven development approach at the stage of system modeling and create interoperability-specific models. Such models can support and be used in different applications that are allowing them to be used by both professional and non-expert application developers [65].

### D. Security

Traditional security solutions and protocols cannot be implemented well in IoT specific environment that is typically constrained by limited computing and power resources [66][67]. Consequently, devices and objects may interact together using many different security techniques and using different operational environments [68]. Added with the challenge of insufficient security expertise among device manufacturers and end users, IoT devices may not always defend themselves appropriately [10]. It is even more difficult to maintain and to improve security when there are frequent software changes, which is the case in agile environments [69].

Some of the well-known IoT-related security challenges include risks of attackers to [66]: 1) access devices which control physical access to home or business such as electronic doors, locks etc.; 2) access smart devices that can be potentially used to act as botnets; 3) exploit IoT devices

for purposes that they were not intended for, leading businesses to monetary losses; 4) monitor and collect valuable data from compromised IoT devices in a home or enterprise; 5) discover the location of the IoT device.

In a recent literature review, [69] identified 17 security challenges with regards to *technical*, *organizational* and *methodological* perspectives, where technical challenges relate to security concerns when designing and implementing IoT applications, organizational challenges relate to company's policies, factors about market and external stakeholders, and methodological challenges relate to the difficulties companies found in integrating security identification, analysis, testing and monitoring in their development methodology [69]. It has been argued that companies would need support with tailoring a process to address these security challenges [69] and that that security challenges can be effectively addressed, if security is introduced prior to development and deployment of IoT by relying on secure software, secure hardware and secure communications [66]. To this end, IoT applications need to consider a comprehensive system, from a cloud-based storage and data analysis, end user applications, middleware and hardware devices and their connectivity [69]. In this line of work, [68] have detailed a systemic and cognitive approach for the IoT, along with a description of: 1) the state-of-the-art of IoT security research activities, 2) the major technological solutions and projects according to this systemic and cognitive approach, and 3) the main standardization activities related to IoT security.

Due to the heterogeneous and irregular composition of IoT systems, it becomes necessary to define a different evaluation of trust for objects, humans, and services [68]. Furthermore, securing IoT devices requires deliberation to create balance between security mechanism and intended operation of a device [66]. Majority of security challenges can be addressed by considering appropriate protocols for the problem and by following best practices during design and development stage, such as [66]: 1) Categorizing IoT devices according to their functionality and communication requirements and then omitting all unnecessary communication modules from the devices; 2) Paying attention to access and authentication mechanisms; 3) Embedding cryptography functions on a hardware level; 4) Considering appropriate key distribution mechanism 5) Making policy decisions related to secure storage capability within the device; 6) Considering how to update or patch the device firmware.

As some practical security-related studies: [13] presents a case study on security and privacy implications on the design of a mobile application in digital health, which utilizes sensing technologies and capabilities of the Internet of Things (IoT), [11] describes an implementation of a secure and efficient authentication and authorization architecture for IoT-based healthcare, [70] proposes to optimize elliptic curve cryptography for 16-bit devices without hardware multiplier by shifting primes, [22] describes the model-based security toolkit, which is applied in a management framework for IoT devices, and [34] highlights Industry 4.0 - related security issues raising awareness for security good practices within Industry 4.0.

### *E. Data and Privacy*

Internet of Things (IoT) applications rely on networks composed of set of heterogeneous sensors and smart devices, which have the capability to constantly, observe the surroundings and generate massive amounts of data [63][71], that need to be processed and stored [3]. The heterogeneity of the devices leads to heterogeneity of generated raw data, which makes the task to interpret such data and detect events in the real world more complex [63][71]. Another challenge with data is the uncertainty caused by noise, sensor error or wireless communication techniques [72]. Data heterogeneity leads to interoperability problems between IoT applications. This challenge has been addressed by using Semantic Web (SW) technologies to model and integrate data from different sources on the web [63]. Semantically annotating contextual information to IoT data is a fundamental step toward developing smarter and interoperable IoT applications. This topic has been addressed by [73][63][74][30].

The necessity to manage data leads also to the challenge of privacy that can be further decomposed to the issues of data privacy and access privacy. Data privacy must be considered throughout the different phases of data usage, including collection, transmission, and storage. Access privacy emphasizes the manner in which people can access to personal information [68]. These topics have been discussed and addressed in [75][68][15][16].

### *F. IoT Development considerations*

Our study revealed several studies working towards new software development approaches. As practical examples, [76] investigated challenges of integrating IoT products into enterprise architecture, and [77] addressed issues related to battery life and power usage, arguing that these are fundamental elements of design in the IoT ecosystem that play a significant role in market success. The topic of guaranteeing the Quality of Service (QoS) was addressed by [58] [78].

IoT is accelerating the transition from application development approaches to application composition approaches [58]. To this end, [79] presents state-of-the-art survey in 13 existing Visual Programming Languages (VLS) being used for IoT application development. Furthermore, large number of connected heterogeneous things and objects require simplifying the development of new applications and services [3]. To address this challenge [80] studied middleware technologies, that are situated between things and applications and create a reliable platform for communication among things with different interfaces, operating systems, and architectures.

IoT is also shifting the development focus from one-time delivery towards continuous delivery, causing currently widely used development methods, such as waterfall or serial methods to crumble [81]. Another issue with traditional development approaches is that development tend to happen in silos, whereas developing IoT solutions would require close communication among relevant stakeholders [81]. It has been argued [81] that agile methodologies' ability to support collaboration and automated tools supporting continuous delivery are particularly suitable to deal with these new IoT-related demands, resulting with shorter lead times, faster overall development efficiency, and more product updates and releases [81]. Furthermore, [82] suggests that combining elements of universal and

participatory design provides users the opportunity to contribute to the invention of solutions that are more compatible with their daily lives.

Despite of advances in ways of developing IoT solutions, [3] argues that there is still a paucity of studies on the social, behavioral, economic, and managerial aspects of the IoT. In particular, the insufficient understanding of IoT business models and how they are connected to the underlying ecosystem calls for greater attention to emerging IoT ecosystems from the business perspective [83] and human relationships and trust within the ecosystem [14].

## IV. Discussion

Developing software for complex IoT systems require new approaches to software development including abstractions, tools, and practices [10]. However, best IoT development practices are still emerging, and suitable test and validation environments are still in their infancy [84]. Two of the reviewed papers [10][58] outlined recommendations for future IoT research. In this section, we will compare the findings of our mapping study with these recommendations.

In this literature mapping study, we identified several examples of utilizing IoT in different domains. Our observation from these studies is that they typically focus on a single domain (such as healthcare, smart city, etc.). Support for research that considers architectures and solutions transcending these specific application domains is hence needed [10].

We identified several studies improving communication technologies in terms of scalability, power consumption and end-to-end delay. This line of work require interdisciplinary collaboration in signal processing and wireless communication, as well as computer architecture and operating systems [10].

Our review related to interoperability challenges revealed a tendency towards microservices architecture, where IoT subsystems can control their own fate over a shared network infrastructure. This creates challenges of discovering relevant services and challenges of ensuring the integrity of system at any given time. Since IoT systems need to be continually modifiable and maintainable to meet changing requirements that were not envisaged at design time, we need novel methods to support software adaptability, scalability and maintainability [58]. These needs call for research to facilitate the construction, deployment, and automated analysis of multicomponent systems with complex and dynamic dependences [10]. Some of the main challenges include the development of models, methods and design tools for going beyond formal methods research to create abstractions and formalisms for constructing and reasoning about systems with diverse and more difficult-to-characterize components [10][58]. There is also a need to further research design patterns at the architectural level describing the obligations/constraints to be fulfilled by the system in which the software is running, and to validate and standardize them [58].

Security was identified to be one of the essential challenges of IoT systems. The reviewed papers suggested that this is best to be tackled in a holistic manner already at design stage. Further research is needed on the unique challenges and opportunities in IoT security, such as minimal operating systems to create IoT devices with smaller attack surfaces, new ways detect and prevent anomalous network traffic, and high-level policy languages for specifying permissible communication patterns [10]. On a more general level, we need research that addresses cross-cutting issues, such as: networking, security, privacy, and impact of the physical on the cyber, real-time. [10]. These aspects need to be taken into account when software development methodologies are evolving towards supporting IoT systems development [58].

The ability of IoT systems to generate wide range of data, not only raises challenges of turning raw data into meaningful knowledge and events, but also raises concerns of privacy. These concerns are closely related with the topic of security.

IoT systems consisting of independent microservices, that are possibly created and maintained by several different parties, create a complex and dynamic software-powered ecosystem that is flexible and constantly evolving. Existing Requirements Engineering approaches do not account well for such dynamicity of use and unknown requirements [58]. Hence, there is the need for a radically divergent approach to capture emerging behavior from systems and users [58]. The fact that we can now access enormous amount of data about the system creates an exciting new opportunity. We have now better possibilities to make informed decisions based on gathered user feedback and data showing the usage of the system [58]. This calls for novel software production methodologies to actually enable controlled management of feature experimentation with short development cycles [58]. An example of this kind of new methodology is Hypothesis Experiment Data-Driven Development (HYPEX-model) [85], that is used for initiating, conducting and evaluating feature experiments with customers with the intent to improve decision-making and prioritization within software development companies. However, more research is needed for supporting decision-making that is based on real data gathered from experiments with customers. Despite the ability to capture large amounts of data related to the behavior of an application, limited progress has been achieved in developing feedback analysis tools [43]. Moreover, in order to reduce the development time as much as possible, we need simulators, benchmarks, and code bases to be able to quickly conduct realistic (and repeatable) experiments and to evaluate new ideas with reasonable investments of time and money [10]. We also need approaches that can increase the anti-fragility of systems, reduce the meantime-to-restore-service (MTRS), and develop accelerated methodologies to test quality through staging and canary testbeds [58].

Our review also revealed that the diversity of stakeholders within the business ecosystem creates essential challenges that are not yet sufficiently researched, creating the need to study business aspects, value creation as well as social aspects within the ecosystem.

## V. Conclusions

We have in this paper conducted a systematic mapping study of existing IoT literature, with the intent to distill IoT - related: 1) challenges, 2) experimental studies, and 3) recommendations for future research. Our work resulted with the identification of six categories of IoT challenges and with identification of the papers addressing these challenges

(Table II). Our study provides details on the progress of research with respect of each type of IoT challenges and argues that software development style is likely to change towards experimentation and data-driven development. The paper also points out the insufficient attention to business and human aspects, when working within an IoT ecosystems.

## REFERENCES


[1] S. Chen, H. Xu, D. Liu, B. Hu, and H. Wang, "A Vision of IoT: Applications, Challenges, and Opportunities With China Perspective," *IEEE Internet Things J.*, vol. 1, no. 4, pp. 349–359, Aug. 2014.

[2] N. K. Giang, M. Blackstock, R. Lea, and V. C. M. Leung, "Developing IoT applications in the Fog: A Distributed Dataflow approach," in *2015 5th International Conference on the Internet of Things (IOT)*, Seoul, South Korea, 2015, pp. 155–162.

[3] I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," *Bus. Horiz.*, vol. 58, no. 4, pp. 431–440, Jul. 2015.

[4] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic Mapping Studies in Software Engineering.," in *EASE*, 2008, vol. 8, pp. 68–77.

[5] J. Popay *et al.*, "Guidance on the conduct of narrative synthesis in systematic reviews," *Prod. ESRC Methods Programme Version*, vol. 1, p. b92, 2006.

[6] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering–a systematic literature review," *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, 2009.

[7] M. Sampson, J. McGowan, E. Cogo, J. Grimshaw, D. Moher, and C. Lefebvre, "An evidence-based practice guideline for the peer review of electronic search strategies," *J. Clin. Epidemiol.*, vol. 62, no. 9, pp. 944–952, 2009.

[8] A. Aghaei Chadegani *et al.*, "A comparison between two main academic literature collections: Web of Science and Scopus databases," 2013.

[9] S. Albishi, B. Soh, A. Ullah, and F. Algarni, "Challenges and Solutions for Applications and Technologies in the Internet of Things," *Procedia Comput. Sci.*, vol. 124, pp. 608–614, Jan. 2017.

[10] "[1604.02980] Systems Computing Challenges in the Internet of Things." [Online]. Available: https://arxiv.org/abs/1604.02980. [Accessed: 24-Dec-2018].

[11] S. R. Moosavi *et al.*, "SEA: A Secure and Efficient Authentication and Authorization Architecture for IoT-Based Healthcare Using Smart Gateways," *Procedia Comput. Sci.*, vol. 52, pp. 452–459, Jan. 2015.

[12] P. Maia *et al.*, "A Web Platform for Interconnecting Body Sensors and Improving Health Care," *Procedia Comput. Sci.*, vol. 40, pp. 135–142, Jan. 2014.

[13] M. Volk, J. Sterle, and U. Sedlar, "Safety and Privacy Considerations for Mobile Application Design in Digital Healthcare," *Int. J. Distrib. Sens. Netw.*, vol. 11, no. 10, p. 549420, Oct. 2015.

[14] S. Bhattacharya, D. Wainwright, and J. Whalley, "Internet of Things (IoT) enabled assistive care services: Designing for value and trust," *Procedia Comput. Sci.*, vol. 113, pp. 659–664, Jan. 2017.

[15] Y. O'Connor, W. Rowan, L. Lynch, and C. Heavin, "Privacy by Design: Informed Consent and Internet of Things for Smart Health," *Procedia Comput. Sci.*, vol. 113, pp. 653–658, Jan. 2017.

[16] S. Darwish, I. Nouretdinov, and S. D. Wolthusen, "Towards Composable Threat Assessment for Medical IoT (MIoT)," *Procedia Comput. Sci.*, vol. 113, pp. 627–632, Jan. 2017.

[17] A. Abdelgawad and K. Yelamarthi, "Internet of things (IoT) platform for structure health monitoring," *Wirel. Commun. Mob. Comput.*, vol. 2017, 2017.

[18] Abinaya, V. Kumar, and Swathika, "Ontology based public healthcare system in internet of things (IoT)," presented at the Procedia Computer Science, 2015, vol. 50, pp. 99–102.

[19] R. Cortés, X. Bonnaire, O. Marin, and P. Sens, "Stream processing of healthcare sensor data: Studying user traces to identify challenges from a big data perspective," presented at the Procedia Computer Science, 2015, vol. 52, pp. 1004–1009.

[20] P. M. N. Martins and J. A. McCann, "The Programmable City," *Procedia Comput. Sci.*, vol. 52, pp. 334–341, Jan. 2015.

[21] L. Tian, H. Wang, Y. Zhou, and C. Peng, "Video big data in smart city: Background construction and optimization for surveillance video processing," *Future Gener. Comput. Syst.*, vol. 86, pp. 1371–1382, Sep. 2018.

[22] R. Neisse, G. Steri, I. N. Fovino, and G. Baldini, "SecKit: A Model-based Security Toolkit for the Internet of Things," *Comput. Secur.*, vol. 54, pp. 60–76, 2015.

[23] S. Faieq, R. Saidi, H. Elghazi, and M. D. Rahmani, "C2IoT: A framework for Cloud-based Context-aware Internet of Things services for smart cities," *Procedia Comput. Sci.*, vol. 110, pp. 151–158, Jan. 2017.

[24] H. Chen, H. Hui, Z. Su, D. Fang, and Y. Hui, "Real-Time Pricing Strategy Based on the Stability of Smart Grid for Green Internet of Things," *Mobile Information Systems*, 2017. [Online]. Available: https://www.hindawi.com/journals/misy/2017/5039702/abs/. [Accessed: 24-Dec-2018].

[25] J. Shuja *et al.*, "Greening emerging IT technologies: techniques and practices," *J. Internet Serv. Appl.*, vol. 8, no. 1, p. 9, Jul. 2017.

[26] A. Al-Saadi, R. Setchi, and Y. Hicks, "Cognitive network framework for heterogeneous wireless networks," *Procedia Comput. Sci.*, vol. 60, pp. 216–225, 2015.

[27] K. Ashokkumar, B. Sam, R. Arshadprabhu, and others, "Cloud based intelligent transport system," *Procedia Comput. Sci.*, vol. 50, pp. 58–63, 2015.

[28] H. Lee, K. Ryu, and Y. Cho, "A Framework of a Smart Injection Molding System Based on Real-time Data," *Procedia Manuf.*, vol. 11, pp. 1004–1011, Jan. 2017.

[29] S. Simons, P. Abé, and S. Neser, "Learning in the AutFab – The Fully Automated Industrie 4.0 Learning

Factory of the University of Applied Sciences Darmstadt," *Procedia Manuf.*, vol. 9, pp. 81–88, Jan. 2017.

[30] S. G. Pease, P. P. Conway, and A. A. West, "Hybrid ToF and RSSI real-time semantic tracking with an adaptive industrial internet of things architecture," *J. Netw. Comput. Appl.*, vol. 99, pp. 98–109, Dec. 2017.

[31] M. Spadacini, S. Savazzi, and M. Nicoli, "Wireless home automation networks for indoor surveillance: technologies and experiments," *EURASIP J. Wirel. Commun. Netw.*, vol. 2014, no. 1, p. 6, 2014.

[32] H.-Y. Kim, "A design and implementation of a framework for games in IoT," *J. Supercomput.*, vol. 74, no. 12, pp. 6516–6528, Dec. 2018.

[33] L. Catarinucci, D. De Donno, L. Mainetti, L. Patrono, M. L. Stefanizzi, and L. Tarricone, "An IoT-aware Architecture to improve Safety in Sports Environments," vol. 13, no. 2, pp. 44–52, Jun. 2017.

[34] T. Pereira, L. Barreto, and A. Amaral, "Network and information security challenges within Industry 4.0 paradigm," *Procedia Manuf.*, vol. 13, pp. 1253–1260, Jan. 2017.

[35] R. Y. Zhong, X. Xu, E. Klotz, and S. T. Newman, "Intelligent Manufacturing in the Context of Industry 4.0: A Review," *Engineering*, vol. 3, no. 5, pp. 616–630, 2017.

[36] P. P. Ray, "A survey on Internet of Things architectures," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 30, no. 3, pp. 291–319, 2018.

[37] B. Negash, A.-M. Rahmani, T. Westerlund, P. Liljeberg, and H. Tenhunen, "LISA: Lightweight internet of things service bus architecture," presented at the Procedia Computer Science, 2015, vol. 52, pp. 436–443.

[38] L. Alonso, J. Barbarán, J. Chen, M. Díaz, L. Llopis, and B. Rubio, "Middleware and communication technologies for structural health monitoring of critical infrastructures: A survey," *Comput. Stand. Interfaces*, vol. 56, pp. 83–100, Feb. 2018.

[39] P. Fremantle and P. Scott, "A survey of secure middleware for the internet of things," *PeerJ Comput. Sci.*, vol. 2017, no. 5, 2017.

[40] R. Rondón, M. Gidlund, and K. Landernäs, "Evaluating Bluetooth Low Energy Suitability for Time-Critical Industrial IoT Applications," *Int. J. Wirel. Inf. Netw.*, vol. 24, no. 3, pp. 278–290, Sep. 2017.

[41] J. Guan, I. You, C. Xu, and H. Zhang, "The PMIPv6-Based Group Binding Update for IoT Devices," *Mobile Information Systems*, 2016. [Online]. Available: https://www.hindawi.com/journals/misy/2016/7853219/abs/. [Accessed: 24-Dec-2018].

[42] S. M. Ghaleb, S. Subramaniam, Z. A. Zukarnain, and A. Muhammed, "Mobility management for IoT: a survey," *Eurasip J. Wirel. Commun. Netw.*, vol. 2016, no. 1, 2016.

[43] G. K. Teklemariam, J. Hoebeke, I. Moerman, and P. Demeester, "Facilitating the creation of IoT applications through conditional observations in CoAP," *EURASIP J. Wirel. Commun. Netw.*, vol. 2013, no. 1, p. 177, Jun. 2013.

[44] R. Sokullu and E. Demir, "Investigating Energy Efficiency and Timeliness for Linear Wireless Sensor Networks," *Procedia Comput. Sci.*, vol. 37, pp. 24–31, Jan. 2014.

[45] B. Park, A. Hwang, and H. Latchman, "Design of Optimized Multimedia Data Streaming Management Using OMDSM over Mobile Networks," *Mobile Information Systems*, 2017. [Online]. Available: https://www.hindawi.com/journals/misy/2017/2867127/abs/. [Accessed: 24-Dec-2018].

[46] H.-W. Wang, "Efficient DFSA Algorithm in RFID Systems for the Internet of Things," *Mobile Information Systems*, 2015. [Online]. Available: https://www.hindawi.com/journals/misy/2015/942858/abs/. [Accessed: 24-Dec-2018].

[47] O. Galinina, K. Mikhaylov, S. Andreev, A. Turlikov, and Y. Koucheryavy, "Smart home gateway system over Bluetooth low energy with wireless energy transfer capability," *EURASIP J. Wirel. Commun. Netw.*, vol. 2015, no. 1, p. 178, Jun. 2015.

[48] A. Rodriguez, A. Ordóñez, H. Ordoñez, and R. Segovia, "Adapting NSGA-II for Hierarchical Sensor Networks in the IoT," *Procedia Comput. Sci.*, vol. 61, pp. 355–360, Jan. 2015.

[49] A. Ahmad, A. Paul, M. M. Rathore, and S. Rho, "Power Aware Mobility Management of M2M for IoT Communications," *Mobile Information Systems*, 2015. [Online]. Available: https://www.hindawi.com/journals/misy/2015/521093/. [Accessed: 24-Dec-2018].

[50] Y. Mehmood, C. Görg, M. Muehleisen, and A. Timm-Giel, "Mobile M2M communication architectures, upcoming challenges, applications, and future directions," *Eurasip J. Wirel. Commun. Netw.*, vol. 2015, no. 1, pp. 1–37, 2015.

[51] A. Biral, M. Centenaro, A. Zanella, L. Vangelista, and M. Zorzi, "The challenges of M2M massive access in wireless cellular networks," *Digit. Commun. Netw.*, vol. 1, no. 1, pp. 1–19, 2015.

[52] A. Taufique, M. Jaber, A. Imran, Z. Dawy, and E. Yacoub, "Planning Wireless Cellular Networks of Future: Outlook, Challenges and Opportunities," *IEEE Access*, vol. 5, pp. 4821–4845, 2017.

[53] Z. Jiang, B. Han, P. Chen, F. Yang, and Q. Bi, "On Novel Access and Scheduling Schemes for IoT Communications," *Mob. Inf. Syst.*, vol. 2016, 2016.

[54] S. N. Khan Marwat, Y. Mehmood, C. Görg, and A. Timm-Giel, "Data aggregation of mobile M2M traffic in relay enhanced LTE-A networks," *EURASIP J. Wirel. Commun. Netw.*, vol. 2016, no. 1, p. 110, Apr. 2016.

[55] Y. Leng, Z. Chen, and Y. Hu, "STLIS: A Scalable Two-Level Index Scheme for Big Data in IoT," *Mob. Inf. Syst.*, vol. 2016, 2016.

[56] M. Kim, M. Asthana, S. Bhargava, K. K. Iyyer, R. Tangadpalliwar, and J. Gao, "Developing an on-demand cloud-based sensing-as-a-service system for internet of things," *J. Comput. Netw. Commun.*, vol. 2016, 2016.

[57] M. Ahmed, L. Liu, J. Hardy, B. Yuan, and N. Antonopoulos, "An Efficient Algorithm for Partially Matched Services in Internet of Services," *Pers.*

*Ubiquitous Comput*, vol. 20, no. 3, pp. 283–293, Jun. 2016.

[58] G. Casale *et al.*, "Current and Future Challenges of Software Engineering for Services and Applications," *Procedia Comput. Sci.*, vol. 97, pp. 34–42, Jan. 2016.

[59] D. Kyriazis and T. Varvarigou, "Smart, Autonomous and Reliable Internet of Things," *Procedia Comput. Sci.*, vol. 21, pp. 442–448, Jan. 2013.

[60] M. Blackstock and R. Lea, "IoT interoperability: A hub-based approach," in *2014 International Conference on the Internet of Things (IOT)*, 2014, pp. 79–84.

[61] Y. Jung, M. Peradilla, and A. Saini, "Software-defined Naming, Discovery and Session Control for IoT Devices and Smart Phones in the Constraint Networks," *Procedia Comput. Sci.*, vol. 110, pp. 290–296, Jan. 2017.

[62] B. Xiao, T. Kanter, and R. Rahmani, "Constructing Context-centric Data Objects to Enhance Logical Associations for IoT Entities," *Procedia Comput. Sci.*, vol. 52, pp. 1095–1100, 2015.

[63] M. Al-Osta, B. Ahmed, and G. Abdelouahed, "A Lightweight Semantic Web-based Approach for Data Annotation on IoT Gateways," *Procedia Comput. Sci.*, vol. 113, pp. 186–193, Jan. 2017.

[64] I. Sohn, "Small-World and Scale-Free Network Models for IoT Systems," *Mobile Information Systems*, 2017. [Online]. Available: https://www.hindawi.com/journals/misy/2017/6752048/abs/. [Accessed: 24-Dec-2018].

[65] P. Grace, B. Pickering, and M. Surridge, "Model-driven interoperability: engineering heterogeneous IoT systems," *Ann. Telecommun.*, vol. 71, no. 3, pp. 141–150, Apr. 2016.

[66] "Internet of things (IoT) design considerations for developers and manufacturers - IEEE Conference Publication." [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7962762. [Accessed: 24-Dec-2018].

[67] C. Bekara, "Security issues and challenges for the IoT-based smart grid," *Procedia Comput. Sci.*, vol. 34, pp. 532–537, 2014.

[68] A. Riahi Sfar, E. Natalizio, Y. Challal, and Z. Chtourou, "A roadmap for security challenges in the Internet of Things," *Digit. Commun. Netw.*, vol. 4, no. 2, pp. 118–137, Apr. 2018.

[69] A. N. Duc, R. Jabangwe, P. Paul, and P. Abrahamsson, "Security challenges in IoT development: a software engineering perspective," presented at the Proceedings of the XP2017 Scientific Workshops, 2017, p. 11.

[70] L. Marin, A. Jara, and A. Skarmeta Gomez, "Shifting primes: Optimizing elliptic curve cryptography for 16-bit devices without hardware multiplier," *Math. Comput. Model.*, vol. 58, no. 5–6, pp. 1155–1174, 2013.

[71] F. Alam, R. Mehmood, I. Katib, and A. Albeshri, "Analysis of Eight Data Mining Algorithms for Smarter Internet of Things (IoT)," presented at the Procedia Computer Science, 2016, vol. 58, pp. 437–442.

[72] Y. H. Wang, K. Cao, and X. M. Zhang, "Complex event processing over distributed probabilistic event streams," *Comput. Math. Appl.*, vol. 66, no. 10, pp. 1808–1821, Dec. 2013.

[73] H. Rahman, R. Rahmani, and T. Kanter, "Multi-Modal Context-Aware reasoNer (CAN) at the Edge of IoT," *Procedia Comput. Sci.*, vol. 109, pp. 335–342, Jan. 2017.

[74] X. Su, H. Zhang, J. Riekki, A. Keränen, J. K. Nurminen, and L. Du, "Connecting IoT Sensors to Knowledge-based Systems by Transforming SenML to RDF," *Procedia Comput. Sci.*, vol. 32, pp. 215–222, Jan. 2014.

[75] A. Alkhalil and R. A. Ramadan, "IoT Data Provenance Implementation Challenges," *Procedia Comput. Sci.*, vol. 109, pp. 1134–1139, 2017.

[76] K. Julia, S. Kurt, and S. Ulf, "Challenges in Integrating Product-IT into Enterprise Architecture – a case study," *Procedia Comput. Sci.*, vol. 121, pp. 525–533, Jan. 2017.

[77] C. Armstrong, "Debug and analysis considerations for optimizing power in your internet of things design," in *2017 IEEE International Symposium on Electromagnetic Compatibility Signal/Power Integrity (EMCSI)*, 2017, pp. 15–19.

[78] G. White, V. Nallur, and S. Clarke, "Quality of service approaches in IoT: A systematic mapping," *J. Syst. Softw.*, vol. 132, pp. 186–203, Oct. 2017.

[79] P. P. Ray, "A Survey on Visual Programming Languages in Internet of Things," *Scientific Programming*, 2017. [Online]. Available: https://www.hindawi.com/journals/sp/2017/1231430/abs/. [Accessed: 24-Dec-2018].

[80] A. Farahzadi, P. Shams, J. Rezazadeh, and R. Farahbakhsh, "Middleware technologies for cloud of things: a survey," *Digit. Commun. Netw.*, vol. 4, no. 3, pp. 176–188, Aug. 2018.

[81] A. Wallgren, "As the IoT expands, agile is the way forward," *TechBeacon*. [Online]. Available: https://techbeacon.com/iot-expands-agile-way-forward. [Accessed: 24-Dec-2018].

[82] D. Gkouskos and J. Burgos, "I'm in! Towards participatory healthcare of elderly through IOT.," *Procedia Comput. Sci.*, vol. 113, pp. 647–652, Jan. 2017.

[83] S. Leminen, M. Westerlund, M. Rajahonka, and R. Siuruainen, "Towards IOT Ecosystems and Business Models," in *Internet of Things, Smart Spaces, and Next Generation Networking*, 2012, pp. 15–26.

[84] J. Favaro, "Strategic research challenges in the Internet of Things," in *IEEE International Conference on Research Challenges in Information Science (RCIS)*, 2015.

[85] H. H. Olsson and J. Bosch, "The HYPEX Model: From Opinions to Data-Driven Software Development," in *Continuous Software Engineering*, J. Bosch, Ed. Cham: Springer International Publishing, 2014, pp. 155–164.

# WiFi coverage range characterization for smart space applications

Zayan EL KHALED
Computer Science Department
University of Quebec at Chicoutimi
Chicoutimi, Quebec, Canada
Zayan.el-khaled1@uqac.ca

Hamid Mcheick
Computer Science Department
University of Quebec at Chicoutimi
Chicoutimi, Quebec, Canada
Hamid_mcheick@uqac.ca

Fabio Petrillo
Computer Science Department
University of Quebec at Chicoutimi
Chicoutimi, Quebec, Canada
Fabio_Petrillo@uqac.ca

***Abstract*—Recently, humans are more and more dependent to communication technologies (CT) in their everyday life to get services, exchange information and communicate with their relatives. Hence, many researches have been made in order to propose convenient and low-cost solutions compatible with the context of smart spaces. This paper characterizes the range of WiFi for outdoor applications in comparison with most known empirical path loss models, and analyzes its impact for smart space services like Internet of Things (IoT). The obtained range is 550m tested with a Samsung Galaxy S5 smartphone, and the comparison with empirical model showed a good difference. Hence the validity and accuracy of those models will be examined for this context, in order to develop an empirical model taking into account environmental effect during our future research. As solutions based on WiFi are generally low cost, its technical characteristics are illustrated and a wide deployment scenario based on this technology is explained on light of obtained results.**

***Keywords: Smart space, WiFi, coverage range, radio link quality, wireless internet service.***

## I. INTRODUCTION

Recently, people are more and more interacting with their environment through various devices or computing platforms thanks to technological progress. These entities are widely used in our everyday activities and spread in an omnipresent way around us, to get required services and information intuitively and easily, whatever is our place or time through many communication technologies (CT). Therefore, ubiquitous computing is the integration of computational power in all sides or objects of our life, and places, where these entities are integrated, and are called smart places or cities.

Thus, CT provided the means for smart spaces' applications and services to emerge by allowing their different entities to communicate together in order to monitor, process and react to environmental change or user needs. Hence, people are able to exchange data across thousands of Kilometers in less than a second, in such a way that world has become like a small village. As smart space applications are diversified such as smart home, smart healthcare, smart grids, and smart traffic; hence requirements of supporting CT are diversified too. Those requirements include various technical specifications such as the networking architecture, data rate, received signal level, the frequency band, coverage area, accessibility, capacity, security, robustness, privacy, and cost. Unfortunately, even with technological advancements, there is no unique communication solution to satisfy all smart space applications at the same time due to their diversified requirements and conditions [1, 2, 3, 4, 5].

Hence, researchers must find a fair equilibrium between applications' requirements, and achieved investments to develop convenient CT in order to support those services. Furthermore, they have to prioritize specifications that highly affect final performances. This goal is achieved by a deep comprehension of smart space conditions, needs, environments, and parameters, in addition to available communication solutions to adopt the most convenient technology to those needs.

In this context, wireless solutions are essential to provide services for people within smart spaces, in order to ease their life, improve their comfort and safety while moving or doing their daily activities. Thus, many researches have been accomplished in order to examine their different challenges and advantages. Those activities are justified by many advantages of wireless technologies such as reduced space overcrowding, lower cost, easy deployment, high coverage, high scalability, and dynamic network formation.

Among all wireless technologies, WiFi can offer a good compromise between the low power consumption of IoT compatible CT like Sigfox [43], low deployment and equipment cost of personal technologies such as Bluetooth, high data rate and wide coverage range of mobile communications such as LTE and WiMAX [44], in addition to its free of charge ISM (Industrial, Scientific and Medical) radio frequencies bands.

For those reasons, in addition to its popularity and availability in smartphones, the purpose of this paper is the evaluation of the coverage range of WiFi technology for outdoor applications as antennas manufacturing companies do not propose or provides the approach to use outdoor WiFi to directly connect the final consumer devices. Then we will discuss the effect of those results on the provision of wireless internet services (WIS) in the context of smart spaces applications like IoT. In addition, obtained results are compared with most known empirical path loss models in order to verify their accuracy in this context.

This paper is organized as follows: Section 2 discusses the state of the art by examining related work, and by giving an overview of wireless internet service and used technologies. In this section, a special focus is given to WiFi and technical characteristics of its versions and a brief description of many empirical path loss models is given. Section 3 describes the scenario we have used to characterize the WiFi range, Section 4 details results that we have obtained in comparison with empirical path loss model, Section 5 discusses the impact of obtained results on IoT applications, and Section 6 discussed threat of validity for obtained results. Finally, a conclusion is given in Section 7.

## II. STATE OF THE ART

### A. Related Work

In general, Smartphones are WiFi enabled, as a consequence many research activities tried to characterize this communication protocol. Berezin et al. [6] investigated if it is feasible to use WiFi in urban areas for mobile Internet access and which type of applications can benefit from the provided Internet access. They also analyzed the characteristics of WiFi coverage and connectivity of mobile users using different mobility speeds and varying some AP settings. Cai et al. [7] discussed an incentive mechanism for the operator to encourage users to delay their cellular

network services and switch to WiFi network in order to offload cellular mobile traffic. Lindner et al. [8] described four exemplary possibilities of exploiting the wireless local area network (WLAN) infrastructure for additional services, provided that WLAN is almost ubiquitous in a coherent area. In addition, many research activities have been made to evaluate and optimize the WiFi range for Sensor network applications [9-11]. The study in [12] is one of the first to analytically describe how businesses, governments, and other organizations are creating large WiFi and wireless internet clouds. This report also described the types of wireless coverage. Seneviratne et al. [13] characterized the WiFi connection set-up process in order to extend a known mathematical model, which will help in the dimensioning of WiFi networks for pedestrian smartphone users. As WiFi is mostly used for indoor applications, many papers tried to characterize its range or other properties for in-building use [14-16]. Actually many applications are using it such as Communication service for rural areas [17], Healthcareservices [18], during extreme events [19], mobile internet services [20], smart home applications [21], or smart city services [22]. In spite of the wide number of research made about communication technologies in smart spaces, there is a lack of characterization of the coverage range of WiFi for outdoor applications because it is mostly used for indoor services. Also, specification sheets of the majority of antennas manufacturing companies [40, 45, 46] do not propose or provide a synthesized process for the use of WiFi to directly connect the final consumer outdoor devices.

### *B. Wireless internet service (WIS)*

#### *1) Introduction*

Due to their countable advantages, wireless technologies (WT) knew a huge development and appear to be a really exciting analysis space for the researchers. Hence, WIS can be defined as the operation of wireless devices or systems used to connect two locations with a radio or other wireless links [23]. In addition, some communication providers call this technology as Wireless To The Home (WTTH) [70]. The deployment of WIS can be explained as a network of wireless access point (AP) having connectivity to the internet. Each AP can connect remotely many Customer Premises Equipments (CPE). A CPE can be installed on the roof of a home or any other high place on the location that we which to connect to the internet. Then CPE is connected to users inside equipments such as wireless routers, or computers by wires to link them to the internet.

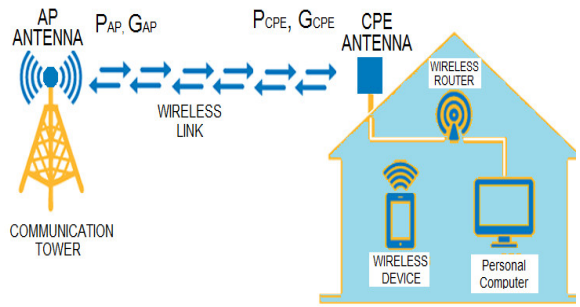



Figure 1. Deployment example of WIS network.

Actually, WIS has gained a wide place in the internet services to provide connectivity for population located in rural areas [24, 70, 72]. For example, WIS is available for 82% people around Canada [25]. Furthermore, this service is used widely in healthcare especially in remote monitoring of patients with chronic diseases while they do their daily routines. Its application in this domain aims to provide autonomy while increasing the quality of life for chronic disease patients and decrease the load of health establishments [26] at the same time. It can also be used during extreme events such as natural disaster as a secondary link to provide internet connection to rescue personnel or damaged population [19, 71].

#### *2) Used Communication Technology*

Many communication technologies are used by WIS or WTTH providers like WiMAX, LTE, Whitespace [27, 28]. But WiFi particularly attracted the attention, as it provides important characteristics and attributes that smart spaces require even in harsh environment [72]. Those critical requirements include:

- Coverage is a very interesting subject because the greater the scope of coverage, the concerned population is increased. A major advantage of WiFi network is its signal coverage [29], which is typically 100 m, it is also tested for outdoor use and its coverage reached 500m [19]. For point-to-point or point to multi-point links, its coverage can be extended until many tens of km [30] by using the configuration described in Figure 1.

TABLE I. A COMPARISON BETWEEN MOST RELEVANT CHARACTERISTICS OF POPULAR WIRELESS CT.

| | Bluetooth | WiFi | LORA | SIGFOX | WIMAX | LTE | 5G |
|---|---|---|---|---|---|---|---|
| Power | L | L | VL | VL | H | VH | M |
| Range | L | H | VH | VH | VH | VH | VH |
| Data rate | L | H | VL | VL | H | H | VH |
| Cost | VL | L | L | L | VH | VH | VH |
| Ref. | [47-50] | [30], [51-54] | [55, 56] | [57, 58] | [59-62] | [63-65] | [66-68] |

Very low: VL, Low: L, High: H, Medium : M, Very high: VH

Good:, Neutral:, Bad:

- WiFi is typically employed for its lower deployment cost [19, 28, 31]. In the context of smart space, a huge number of connected devices are required. Hence, a low-cost technology can decrease needed investments when covering new areas or while upgrading already covered areas to newer technologies. This quality is particularly important for developing countries where fewer resources are available. In this context, even if ad hoc technologies are less costly as they do not require infrastructures; but they still unusable due to reduced data rates and

increased latency when the number of nodes joining the network rises.

- The number of nodes is essential to connect the largest number of people or things especially in the context of smart spaces where a high number of IoT devices are interconnected. In this context, Lee et al. [32] confirm that the maximum number of nodes for WiFi ad hoc network could be unlimited.
- Data rate identifies the amount of information that the network can support. The higher the data rate the greater amount of information can be transmitted between devices. Hence, the data rate of the new version of WiFi surpasses easily 1 Gbps. This property is very important for smart spaces context where a huge amount of information of many types like video, images, text, IoT data or social networks applications are transferred through communication networks.
- As mobility becomes the major trend of our modern life, communication with less power becomes an essential subject to extend the life of batteries without the need for recharging or being connected to a power socket. It is stated that WiFi has the best energy efficiency [33] among WT; which is defined as the power consumed by a given CT in order to transfer a fixed amount of data.
- In general, even if a given technology is absolutely the most effective to support communication, but if it is not available on Smartphones, then it becomes useless in the context of smart spaces. According to information provided by Nokia [34] and Sony [35], Bluetooth and WiFi technologies are available on the majority of current Smartphones [36].
- The area to be covered may have many profiles of geography, vegetation and weather. As the wireless signal penetration can vary according to those environmental effects; the radio frequency is customized accordingly from lower frequency for high penetration through vegetation, buildings or hills to higher frequencies for line of sight links. WiFi can fulfill those requirements while being exempt of license fees at the same time, through the use of ISM radio frequency bands.
- Channel bandwidth need to be adjusted according to the data transfer need, from lower bandwidth for simple text application to high bandwidth for video streaming services. WiFi bandwidth range varies from 5 Mhz for low throughput applications to more than 160 MHz for high data transfer services.

As we can deduce, according to table I, WiFi seems to be among the best suitable choices for the smart space context. Its wide range and the big number of nodes allow covering a large geographic zone and a maximum number of devices. In addition, its high data rate maximizes information transfer and prevent network breakdown due to congestion. In addition, this technology is easy to deploy due to its availability on most smartphones and its reduced cost. Furthermore, it has low power consumption in order to extend the battery life of portable devices and improve users' mobility. Moreover, the frequency and channel bandwidth can be customized according to environmental conditions and throughput for various penetrations and data transfer profiles respectively.

*3) General description of WiFi(IEEE 802.11)*

Wireless fidelity (WiFi) is the radio technology for wireless devices operating in the IEEE 802.11. As the IEEE organization does not verify the compliance of conceived devices with its standards, hundreds of manufacturers around the world have met to form the WiFi alliance. Their aim is to verify the interoperability of their devices operating in 802.11 standards. Indoor WiFi is a single-hop network using IP stack to allow users to surf the Internet at broadband speed when they are connected to their wireless routers. The current solution to extend network range for outdoor use is to use repeaters or a wired LAN to interconnect a group of WiFi APs. New methods are designed by allowing stations to communicate directly without any AP [37]. Thus, WLAN can be formed without pre-planning; it is often referred to ad hoc network [38]. The drawback of this solution is the reduction of performances in terms of link's throughput and latency each time a new node is added to the network. Other solutions are proposed and used in many commercial applications through the use of high-gain antennas for AP and user side (CPE) [73, 74]. By using this solution, the range of WiFi networks can be extended to many tens of Kilometers [31].

*4) Technological Characteristics*

IEEE 802.11 includes a set of specifications of the physical layer and media access control (MAC) for the deployment of a wireless network. This standard allows many different radio frequencies bandwidth to be used within ISM bands for half duplex communications: 900 MHz, 2.4, 3.6, 5, and 60 GHz frequency bands. Wireless links using 2.4 and 5 GHz may probably suffer interference from many other devices operating in the same unlicensed frequencies such as Bluetooth devices or cordless phones. To reduce the interference, many techniques can be used such as orthogonal frequency division multiplexing (OFDM), dynamic frequency allocation or GPS synchronization.

TABLE II. TECHNICAL CHARACTERISTICS OF MOST USED 802.11 STANDARDS

| | **802.11a** | **802.11b** | **802.11g** | **802.11n** | **802.11ac** | **802.11ax** |
|---|---|---|---|---|---|---|
| **Frequency (GHz)** | 3.6, 5 | 2.4 | 2.4 | 2.4, 5 | 5 | 2.4, 5 |
| **Bandwidth (MHz)** | 20 | 22 | 20 | 20, 40 | 20, 40, 80, 160 | 20, 40, 80, 160 |
| **Data rate (Mbps)** | 6, 9, 12, 18, 24, 36, 48, 54 | 1, 2, 5.5, 11 | 6, 9, 12, 18, 24, 36, 48, 54 | Up to 600 for 40 MHz and 4 Streams | Up to 3466.8 for 160 MHz band and 4 Streams | Up to 10,530 for 160 MHz band and 4 Streams |
| **Modulation** | OFDM-64QAM | DSSS | OFDM-64QAM | OFDM-64QAM | OFDM-256QAM | OFDM-1024QAM |
| **MIMO streams** | - | - | - | 4 | 8 | 8 |

With technological advances, many versions of this standard have been made available to satisfy the diversified users' and industries' needs. The most used and known standards are the following: 802.11a, 802.11b, 802.11g, 802.11n, 802.11ac, 802.11ax (Wi-Fi 6), 802.11ad, and 802.11ah. The part of this standard dedicated to radio frequency spectrum allocation may vary according to countries, because each one applies its own strategies or regulations for bandwidth sharing, used frequency channels, maximum transmission power.

The 2.4 GHz frequency refers to the ISM band between 2.4 and 2.5 GHz. The 5 GHz frequency is the band between 4.915 and 5.825 GHz. The 3.6 frequency is between 3.65 and 3.7 GHz. The 900 MHz frequency corresponds to the bands 902-930 MHz. Each band is divided into many sub-channels with a precise central frequency, channel space, and bandwidth. For example, the 2.4 GHz is divided into 14 sub channel spaced of 5 MHz; the central frequency of the first channel is 2.412 GHz and 2.484 for the last channel. The availability of channels may differ between countries according to the spectrum allocations of various services made by regulation.

The bandwidth frequency allows more data rates, for example, if a 20 MHz bandwidth provides 300 Mbps, then the 40 MHz one allows 600 Mbps. Data rate refers to the speed of data transfer, each standard has its own data speed profile, the higher the received signal level, the higher the connection speed.

Furthermore, MIMO stream indicates the number of receiving or emitting antennas and it can make the same effect as the bandwidth makes to the data transfer rate. Hence, if 1 stream makes 300 Mbps then 2 streams are able to provide 600 Mbps.

### C. *Link quality and coverage range prediction*

The wireless link quality may be estimated by the link budget formulas [39]:

$$P_{RX}(\text{dBm}) = P_{TX}(\text{dBm}) + G_{TX}(\text{dB}) + G_{RX}(\text{dB}) - L(\text{dB}) \quad (1)$$

Where:

$P_{rx}$: is the received signal power by the receiver

$P_{tx}$: is the transmitted signal power by the transmitter

$G_{tx}$: is the gain of the transmitter antenna

$G_{rx}$: is the gain of the receiver antenna

L: is the total loss of the link; it includes internal device losses and path losses. Internal device losses can be known by their data sheets, such as connectors' losses. Path losses are the signal attenuation on his way between transmitter and receiver. A lot of research activities have been made to model the propagation effects on path losses, in order to ease network planning and performance prediction. Those models can be classified into three basic types: Empirical, deterministic, and stochastic [76]. Empirical models can give the best tradeoff between the accuracy of deterministic models and the decreased processing requirements off stochastic models. The free air loss [39] is the simplest empirical model and it is represented by the following formula:

$$L_{FS}(dB) = 20\log\frac{4\pi D}{\lambda} \quad (2)$$

Where D is the distance between transmitter and receiver, λ is the wave length of the radio frequency signal.

This model suppose that the attenuation of signal is due to the free air without taking into account other environmental effects such as buildings, multipath fading, antennas heights etc. Thus, several other models have been developed, the most known are the following ones [77]: Cost231, Stanford university interim, Ericson, Hata, Walfisch-Ikegami, Hata-Okumura. Unfortunately, each of these models has different limits of validity than the others and does not perfectly fit measured path losses without making some adjustments [78, 79]. During our research we will compare measured values to most known path loss empirical models in order to find which one fits the best the usage case for outdoor WiFi in the 2.4 GHz frequency band. In addition, environmental effects include all other losses such as attenuation of the wireless signal when it crosses the vegetation, snow, rain, or humidity. Those effects are not easy to estimate because they depend on many unpredictable and random parameters such as weather and leaf density. For the purpose of simplicity, we will consider only the line of sight study without taking into account environmental losses. However, we will consider the estimation of those effects during our future researches.

After the received signal is estimated for a given location, it will be compared with the sensitivity values of the receiver and then the data rate can be known by the nearest sensitivity value. For example, if the receiver is a Ubiquitous Rocket M2 radio [40], and the estimated received signal is -77 dBm, by using Table III, which is provided by the manufacturer of the receiver, we can estimate the data rate of the wireless link as MCS6.

Also, the maximum range can be estimated by the following formula [39]:

$$D = \frac{\lambda \times 10^{\frac{P_{TX}+G_{TX}+G_{RX}-P_{SENS}}{20}}}{4\pi} \quad (3)$$

Where Psens is the lowest detectable power by the receiver. When the receiver detects a radio signal with this power value, the data transfer speed of the link corresponds to the lowest possible data rate, MCS0, and the coverage range is the widest because Tx power is bigger and the sensitivity is lower. Thus if the receiver goes wider than this distance, the radio link fails and the receiver disconnects. Also, this power may correspond to wished operation data rate; in this case the distance is limited to the range where the received signal does not go lower to keep the data rate within specified limits. To extend the coverage while staying within this specification, the whole area is divided into many cells, where each one is covered by an AP as in Figure 3.

TABLE III. TRANSMIT POWER AND RECEIVER SENSITIVITIES FOR UBIQUITY ROCKET M2 RADIO [40] USING 802.11N PROTOCOL (2.4 GHZ) USING ONE MIMO STREAM.

| Data Rate | Avg. TX (dBm) | Sensitivity (dBm) |
|---|---|---|
| MCS0 | 28 | -96 |
| MCS1 | 28 | -95 |
| MCS2 | 28 | -92 |
| MCS3 | 28 | -90 |
| MCS4 | 27 | -86 |
| MCS5 | 25 | -83 |
| MCS6 | 23 | -77 |
| MCS7 | 22 | -74 |

## III. SCENARION OF USE

Our scenario is presented in Figure 2. It contains WiFi AP connected to a router or networking device that has a link to the internet, the covered zone by the AP contains WiFi-

enabled mobile devices. The aim of routing or switching device is to make firewalling, other network operations and to ensure the link between AP and the Internet.

In this scenario, AP may connect all WiFi enabled devices to cover the whole smart space area in order to provide mobile internet service. AP ensures links directly without intermediate devices, contrary to WiFi P2P networks which require many hopes before accessing to the internet [37]. By using high gain antennas, we can ensure a wide range capacity; AP can cover the whole zones including bridges, trees, buildings, and homes. The advantage of having direct connectivity to the Internet is the high data rate; contrarily to ad hoc devices where the network capability is decreased when traffic goes through intermediate devices. To ensure the widest range, it is recommended to install AP in a high place as in the top of a mountain, building or even in a balloon. Consequently, high antennas installation may provide a line of sight, reliable radio link and high data rate.

Our scenario has many advantages in comparison to the model proposed by Ajami and Mcheick [37] or other Ad hoc solutions, where a WiFi Peer to peer network technology is created in uncovered area. In their proposition, the coverage is extended from 100 meters to several kilometers by passing information from one WiFi P2P device to another. The major problem is that devices may not be compatible with each other and a solution that supports IOS devices have to be proposed. Another weakness of their model is the necessity of WiFi direct devices to be spread to all the area to ensure continuity of the coverage. Sometimes, the last condition is difficult to satisfy due to geographic issues. The short range of Ad hoc devices limited to about 100m may increase uncovered area. Unfortunately, we cannot extend this range because it depends on mobile phone devices' constructors and normalization organisms.

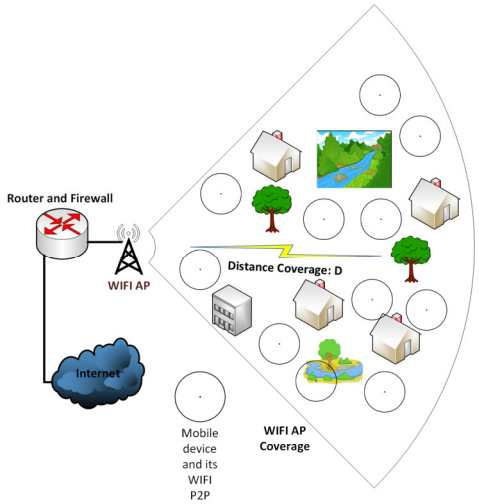



Figure 2. A scenario case study scheme.

AP antenna is chosen according to the area of smart space to be covered; in the market, we can find omnidirectional antennas if users are distributed uniformly around the antenna. If we need to cover a more specific zone, we can choose sector antennas; which have many radiation angles of 120, 90 or 60 degrees [73, 74]. As antenna gain decreases each time radiation angle increases, it is recommended to choose the lowest radiation angle, if possible, to widen the radiation distance.

The most used frequencies for WiFi are 2.4 and 5.8 GHz, which are available for most WiFi enabled devices. It is possible to use other frequencies like 900 MHz and 3.65 GHz but the network may require frequency conversion devices [75]. It is recommended to use 900 MHz and 2.4 GHz for AP due to their high penetration rate. Licensed 3 GHz is recommended to use for backbone link in high interference area. For ease line of sight backbone link, the 5 GHz is recommended for its high capacity and data rate.

## IV. VALIDATION AND RESULT

Our validation test includes the following devices:

- A smartphone: Samsung Galaxy S5
- AP antenna: Ubiquity Airmax AM-2G16-90 [41]
- Radio module with integrated router for AP: Rocket M2 [42]
- Power over Ethernet (POE) injector
- Two SMA pigtails
- AC, DC and RJ45 wires with Power supply

The AP antenna is installed in a way to give a direct line of sight to the test area. The test area has been chosen in an open air place, where the effect of vegetation and buildings is minimized. The AP antenna is connected to the radio through two SMA pigtails as it has two MIMO streams. The radio module includes an integrated router that is connected to the internet through RJ45 wire. It is powered through a POE injector from the router side. The radio is configured in a way to choose a clear bandwidth frequency by using its integrated snooping capability in order to minimize interference with nearby devices. It is configured with a Dynamic Host Configuration Protocol (DHCP) server in order to give a direct access to the internet for the Samsung Galaxy S5 smartphone.

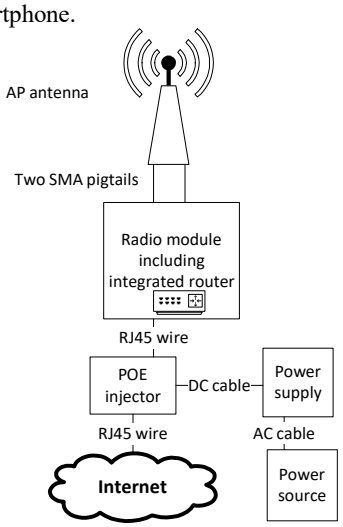



Figure 3. Diagram of the AP installation.

After powering on the power supply, we stayed near of the AP in order to insure that the smart phone is well connected to the internet through it. After then, we moved away gradually while checking the connectivity of the smartphone to the internet. The maximum distance we have gotten before it disconnects was 550m.

In Figure 4, the measured coverage is compared with many other prediction models. The radio characteristics of smartphone and AP are the same as those presented in [19], their height is 1.5 m. The Hata model gives the closest result with 0.258 Km. The free air loss gives a distance prediction

of 30.78 Km and Walfisch-Ikegami is 5.682 Km, we did not put their complete distance value on the graph for the sake of clarity for the results of other models.

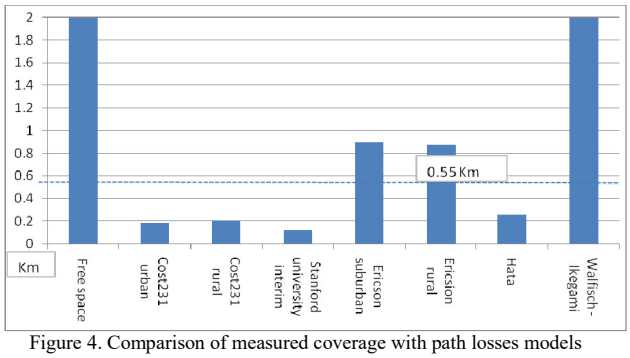



Figure 4. Comparison of measured coverage with path losses models

In most empiric models [77 - 79] the coverage range, depends on following parameters: Antennas gain, frequency, transmitting power, distance, height and obstacles. Those models are essentially developed and fitted basing on measured data in precise environmental conditions for a given technology. Thus, it is difficult to get very good prediction accuracy on all environments and all CT unless the limits of each model are respected. This observation can be justified according to Figure 4, where the best predicted distance has 40 % of difference from the measure, and in figure 5 the accuracy of result vary according to AP height. In the same way as AP height, all environmental conditions have an influence on prediction accuracy; during our future research we will detail, model and express their effects on outdoor WiFi for WIS application by using more data.

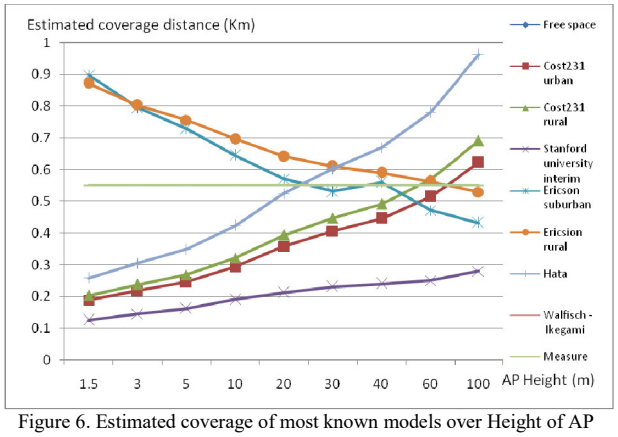



Figure 6. Estimated coverage of most known models over Height of AP

As a result, we can estimate the total area of a cell that can be covered by a WiFi AP. This area is similar to a circle, where its centre is the location of the WiFi AP and its diameter is about 1 km. If the smart space area is wider than the total coverage of a cell, we can install many AP in a way similar to Figure 3. Thus, the total area is divided into many sub-areas; each one is covered by a WiFi AP. The configuration can be done according to the table I, the frequency is chosen to reduce the interference between neighboring cells. The frequency bandwidth is configured according to the required global throughput; the population number in each cell can be a good index to estimate this parameter. An automatic configuration algorithm will be proposed for a similar context during our future researches.

## V. CONSEQUENCES FOR IOT APPLICATIONS

Actually, IoT is clearly a trend that cannot be ignored. According to CISCO [69], IoT connections are becoming bigger than other daily connections, and IoT traffic will double by 2020. Furthermore, IoT applications are diversified in terms of connectivity specifications such as data transfer, coverage range, energy efficiency and cost. The data transfer requirement vary from small and intermittent connectivity for some utility sensors and meters to large amount of continuous data flow such as video surveillance or drones activities. In addition, the power need may vary from high intensive power consumption applications to very high battery lifetime of many years' sensors. Equally, the coverage may vary from easy to access small indoor building into hard to access very wide area forests.

The WiFi connectivity is a good choice as it is almost ubiquitous for indoor buildings, campus environments and urban areas. In addition, it has the advantage of being able to support easily high data rate application, such as data streaming, with reasonable coverage range and power consumption with a lower cost. In addition, this technology can get a higher coverage range through the use of lower ISM frequency bands in order to improve its penetration through vegetation and hills for IoT devices to be used as fire forest sensors, covering a wide area and requiring lower power consumption and low data rate. Moreover, Lora and Sigfox technologies can be good alternatives for such IoT applications as they have extended battery life and convenient transmission capacity.

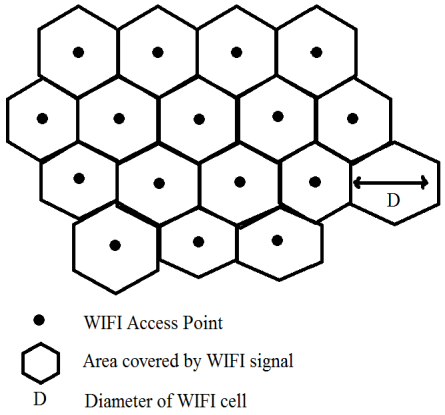



Figure 5. Wide range deployment model for WiFi.

At the other hand, for devices scattered across a large area and requiring a good throughput, such as drones broadcasting video, we can make a WiFi configuration like the one presented in figure 3. The advantage is that WiFi can support unlimited number of wireless connections, and it has many bandwidth frequency configurations in order to adjust throughput according to applications' needs while keeping a good mobility at the same time. Furthermore, this technology is well adapted for developing countries' context, as it does not require high investing as its infrastructure cost is low in addition to the free frequency licenses; which helps to spread IoT in poor countries.

## VI. THREATS TO VALIDITY

Our test has been done in an urban environment, hence the high spectrum congestion of WiFi inducing interference effects has been taken into account. Even if the test have been made for line of site configuration but fading and multi

path propagation of radio frequency signal have been taken into account. In addition, more measures will be done during our future research in order verify the accuracy of various path loss models and develop other models, if needed for the context of WIS deployment. Environmental effects such as seasons, weather, and vegetation will be taken into account during our future researches. A test configuration taking into account many AP will be proposed in our future researches.

## VII. CONCLUSION

This paper characterized the coverage range of WiFi for outdoor applications, and analyzed its consequences for smart space services like IoT. As WiFi is generally low cost, its technical characteristics are explained and a deployment scenario based on this technology is explained on light of obtained results. It is also available on all mobile devices and it is supported by a majority of used internet enabled objects. Therefore, investments to develop new compatible technologies on other devices are saved. In addition, this technology is well suited to resolve the small coverage of Ad hoc P2P networks and the necessity to have continuity of enabled P2P devices into the entire needed zone. Obtained outdoor coverage range is 550 m, it is characterized for a clear line of sight links. The comparison between the values of coverage distance obtained during measure with most known empirical path loss models showed an important difference. During our future researches we will verify the accuracy of empirical models for the context of outdoor WiFi on WIS deployment by using more measures. A new empirical model satisfying this context will be proposed by taking into account environmental effects such as weather, season and vegetation on the QoS of outdoor WiFi signals. In addition, we will propose a method to automate the configuration of WiFi APs covering a whole area in order to satisfy specific requirements and fasten their deployment.

## REFERENCES


[1] Hashem, Ibrahim & Chang, Victor & Anuar, Nor & S, Adewole & Yaqoob, Ibrar & Gani, Abdullah & Ahmed, Ejaz & Chiroma, Haruna. (2016). The Role of Big Data in Smart City. International Journal of Information Management. 36. 10.1016/j.ijinfomgt.2016.05.002.

[2] Yaqoob, Ibrar & Hashem, Ibrahim & Mehmood, Yasir & Gani, Abdullah & Mokhtar, Salimah & Guizani, Sghaier. (2017). Enabling Communication Technologies for Smart Cities. IEEE Communications Magazine. 55. 10.1109/MCOM.2017.1600232CM.

[3] Gharaibeh, Ammar & Salahuddin, Mohammad & J. Hussini, Sayed & Khreishah, Abdallah & Khalil, Issa & Guizani, Mohsen & Al-Fuqaha, Ala. (2017). Smart Cities: A Survey on Data Management, Security and Enabling Technologies. IEEE Communications Surveys & Tutorials. PP. 1-1. 10.1109/COMST.2017.2736886.

[4] Tiago D. P. Mendes, Radu Godina, Eduardo M. G. Rodrigues, João C. O. Matias and João P. S. Catalão: Smart Home Communication Technologies and Applications: Wireless Protocol Assessment for Home Area Network Resources, Energies 2015, 8, 7279-7311; doi:10.3390/en8077279

[5] Mohamed Daoud, Xavier Fernando:On the Communication Requirements for the Smart Grid, Energy and Power Engineering, 2011, 3, 53,60, doi:10.4236/epe.2011.31008 Published Online February 2011 (http://www.SciRP.org/journal/epe)

[6] Berezin, Maria & Rousseau, Franck & Duda, A. (2012). Citywide Mobile Internet Access Using Dense Urban WiFi Coverage. 31-36. 10.1145/2413236.2413244.

[7] Cai, S.-J & Xiao, L.-M & Wang, Jian & Zhou, S.-D. (2015). Incentive mechanism design for WiFi offloading with users' mobility. 37. 2431-2437. 10.11999/JEIT150285.

[8] Lindner, Thomas & Rannenberg, Kai & Fritsch, Lothar & Plank, Kilian. (2004). Exploitation of Public and Private WiFi Coverage for New Business Models. International Federation for Information Processing Digital Library; Building the E-Service Society;. 10.1007/1-4020-8155-3_8.

[9] Sharma, Rinku. (2018). Maximum Coverage Range Based Sensor Node Selection Approach to Optimize in WSN. International Journal for Research in Applied Science and Engineering Technology. 6. 1592-1596. 10.22214/ijraset.2018.4266.

[10] Sharma, Rinku. (2018). Maximum Coverage Range based Sensor Node Selection Approach to Optimize in WSN. International Journal for Research in Applied Science and Engineering Technology. 6. 597-601. 10.22214/ijraset.2018.5099.

[11] Godoi Vieira, Ricardo & Cunha, Adilson & Beatryz Ruiz, Linnyer & Camargo, Antonio. (2017). On the design of a long range WSN for Precision Irrigation. IEEE Sensors Journal. 1558-1748. 10.1109/JSEN.2017.2776859.

[12] A Shamp, Scott. (2018). WiFi Clouds and Zones: A Survey of Municipal Wireless Initiatives.

[13] Seneviratne, Suranga & Seneviratne, Aruna & Mohapatra, Prasant & Tournoux, Pierre-Ugo. (2013). Characterizing WiFi Connection and Its Impact on Mobile Users: Practical Insights. 81-88. 10.1145/2505469.2505480.

[14] Makki, Ahmed & Bleakley, Chris. (2018). WLAN Indoor Ranging Dataset for Evaluation of Time of Arrival Estimation Algorithms.

[15] Tervonen, Jouni & Hartikainen, Markku & Heikkilä, Marjo & Koskela, Marjut. (2016). Applying and Comparing Two Measurement Approaches for the Estimation of Indoor WiFi Coverage. 1-4. 10.1109/NTMS.2016.7792436.

[16] De, Guillaume & Roche, La & Jaffres-Runser, Katia & Gorce, Jean-Marie. (2007). On predicting in-building WiFi coverage with a fast discrete approach. Int. J. Mobile Network Design and Innovation. 2. 10.1504/IJMNDI.2007.013799.

[17] Rabin Krushnachandra Patra: A Multi-Tier Network Architecture for Long Distance Rural Wireless Networks in Developing Regions, thesis report, Electrical Engineering and Computer Sciences University of California at Berkeley, Hune 2009

[18] Mcheick, Hamid. (2017). Context Relevant Prediction Model for COPD Domain Using Bayesian Belief Network. Sensors 2017, 17(7), 1486; doi:10.3390/s17071486. 17. 24. 10.3390/s17071486.

[19] Zayan Elkhaled, Hamid Mcheick, and Hicham Ajami: Wide Range WiFi Network Formal Design Model for Ubiquitous Emergency Events. Submitted to The 11th International Conference on Future Networks and Communications, August 15-18, 2016, Montreal, Quebec, Canada.

[20] Ping, Zhang & Perkins, Charles. (2013). Mobile Internet [Guest Editorial]. Communications, China. 10. vii-viii. 10.1109/CC.2013.6549253.

[21] Ding, Jie & Li, Tian-Ran & Chen, Xue-Li. (2018). The Application of WiFi Technology in Smart Home. Journal of Physics: Conference Series. 1061. 012010. 10.1088/1742-6596/1061/1/012010.

[22] Garcia, Laura & Jimenez, Jose & Taha, Miran & Lloret, Jaime. (2018). Wireless Technologies for IoT in Smart Cities. Network Protocols and Algorithms. 10. 23. 10.5296/npa.v10i1.12798.

[23] "Wireless Broadband and Other Fixed-Wireless Systems" (PDF). networkcomputing.com. Retrieved 2008-02-20.

[24] Rabin Krushnachandra Patra: A Multi-Tier Network Architecture for Long Distance Rural Wireless Networks in Developing Regions, thesis report, Electrical Engineering and Computer Sciences University of California at Berkeley, Hune 2009

[25] Web site: https://crtc.gc.ca/eng/internet/internetcanada.htm visited on 20-09-2018

[26] Mcheick, Hamid. (2017). Context Relevant Prediction Model for COPD Domain Using Bayesian Belief Network. Sensors 2017, 17(7), 1486; doi:10.3390/s17071486. 17. 24. 10.3390/s17071486.

[27] Yaqoob, Ibrar & Hashem, Ibrahim & Mehmood, Yasir & Gani, Abdullah & Mokhtar, Salimah & Guizani, Sghaier. (2017). Enabling Communication Technologies for Smart Cities. IEEE Communications Magazine. 55. 10.1109/MCOM.2017.1600232CM.

[28] http://www.wispa.org

[29] OndreyHyncica et al, 2005, Wireless Standards for Mobile Platform, WSEAS, Stevens Point, Wisconsin, USA.

[30] Raman, Bhaskaran & Chebrolu, Kameswari. (2007). Experiences in using WiFi for rural internet in India. Communications Magazine, IEEE. 45. 104 - 110. 10.1109/MCOM.2007.284545.

[31] Patra, S. Nedevschi, S. Surana, A. Sheth, L. Subramanian and E. Brewer, "WiLdnet Design and implementation of high performanceWiFi based long distance networks," in Proceedings of the 4th USENIX conference on Networked systems design and implementation, 2007.

[32] Jin-Shyan Lee et al., 33rth Annual Conference of IECON, A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee and WiFi, November, 2007.

[33] N. Balasurbramanian, A. Balasubramanian and Arun Venkataramani, "Energy Consumption in Mobile Phones: A Measurement Study and Implications for Network Applications", Proceedings of the 9th ACM SIGCOMM conference on Internet measurement, Chicago, Illinois, USA, 4-6 November 2009, pp. 280-293. DOI: https://doi.org/ 10.1109/IWCMC.2013.6583599

[34] Nokia, http://www.developer.nokia.com/Devices/ Device_ specifications/N97/, May 2012.

[35] Sony, http://www.sonymobile.com/globalen/ products/phones/xperia-p/specifications/, May 2012.

[36] Apple Inc., http://www.apple.com/iphone/specs.html, May 2012

[37] Hicham Ajami and Hamid Mcheick, Emergency network Prototype, Proceeding of the 5Th Annual DIVA Workshop, NSERC-DIVA Network, DIVA research strategy network, Eds. A Boukerche& R. De Grande, pp. 52-56, February 16-18, 2016, Ottawa, Canada.

[38] Jin-Shyan Lee et al., 33rth Annual Conference of IECON, A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee and WiFi, November, 2007.

[39] Bernard Slar: Digital Communications: Fundamentals & Applications, Second Edition, December 2005

[40] https://www.ubnt.com/

[41] Ubiquity manufacturer, airmax access points https://dl.ubnt.com/datasheets/airmaxsector/airMAX_Sector_Antennas_DS.pdf

[42] Ubiquit manufacturer used as an example to show access point radio performance: https://dl.ubnt.com/datasheets/rocketm/RocketM_DS.pdf

[43] Sigfox Technical Overview, May 2017, www.sigfox.com

[44] A. Agarwal and K. Agarwal, "The Next Generation Mobile Wireless Cellular Networks–4G and Beyond", American Journal of Electrical and Electronic Engineering, Vol. 2, No. 3, pp. 92-97, 2014.

[45] https://www.manta.com/mb_35_E22A77N1_000/antennas_receiving

[46] http://www.listofcompaniesin.com/antenna-companies.html

[47] Siegemund F.: A Context-Aware Communication Platform for Smart Objects. In: Ferscha A., Mattern F. (eds) Pervasive Computing. Pervasive 2004. Lecture Notes in Computer Science, vol 3001. Springer, Berlin, Heidelberg

[48] K. Chang, "Bluetooth: a viable solution for IoT? [Industry Perspectives]," in IEEE Wireless Communications, vol. 21, no. 6, pp. 6-7, December 2014. doi: 10.1109/MWC.2014.7000963

[49] S. Vyas, U. Chaudhari, V. C. Nandini and B. Thakare, "State of the Art Literature Survey 2015 on Bluetooth", International Journal of Computer Applications, Vol. 131, no 8, December 2015, pp. 7-10. DOI: https://doi.org/ 10.5120/ijca2015907391

[50] S. Darroudi and C. Gomez, "Bluetooth Low Energy Mesh Networks: A Survey", Sensors, MDPI, Vol. 17, no 7, 22 June 2017, pp. 1467. DOI: https://doi.org/10.3390/s17071467

[51] Benny Bing:Wi-Fi Technologies and Applications: From 802.11ac and 802.11ax to Internet of Things, Drones, and Balloons Paperback, Amazon; 1st edition, 2017.

[52] A. Zanella, N. Bui, A. Castellani, L. Vangelista and M. Zorzi, "Internet of things for Smart Cities", IEEE Internet of Things Journal, Vol. 1, No. 1, pp. 22-32, February 2014. DOI: https://doi.org/ 10.1109/JIOT.2014.2306328

[53] R. Bhaskaran and C. Kameswari, "Design and evaluation of a new MAC protocol for long-distance 802.11 mesh networks," in Proceedings of the 11th annual international conference on Mobile computing and networking, 2005.

[54] S. Patra, S. Nedevschi, S. Surana, A. Sheth, L. Subramanian and E. Brewer, "WiLdnet Design and implementation of high performance WiFi based long distance networks," in Proceedings of the 4th USENIX conference on Networked systems design and implementation, 2007.

[55] A. Augustin, J. Yi, T. Clausen and W. M. Townsley, "A Study of LoRa: Long Range & Los Power Networks for the Internet of Things", SENSORS, Vol. 16, No. 9, p. 1466, 2016. DOI: https://doi.org/10.3390/s16091466

[56] https://lora-alliance.org/

[57] Kais Mekki, Eddy Bajic, Frederic Chaxel, Fernand Meyer: A comparative study of LPWAN technologies for large-scale IoT deployment, ICT Express, 2018, ISSN 2405-9595, https://doi.org/10.1016/j.icte.2017.12.005.

[58] Sigfox Technical Overview, May 2017, www.sigfox.com

[59] Sassan Ahmadi: Mobile WiMAX 1st Edition: A Systems Approach to Understanding IEEE 802.16m Radio Access Technology, Academic Press, 2011.

[60] H. L. Vu, S. Chan , L. L. H. Andrew, "Performance Analysis of Best-Effort Service in Saturated IEEE 802.16 Networks", IEEE Transactions on Vehicular Technology, Vol. 59, No. 1, pp. 460-472, 2010. DOI: https://doi.org/10.1109/TVT.2009.2033191

[61] A. Agarwal and K. Agarwal, "The Next Generation Mobile Wireless Cellular Networks–4G and Beyond", American Journal of Electrical and Electronic Engineering, Vol. 2, No. 3, pp. 92-97, 2014.

[62] T. Schwengler, Wireless & Cellular Communications - Class Notes for TLEN5510, 2014

[63] W. Jung and Y. Kwon, "Differences between LTE and 3G service customers: Business and policy implication" , Telematics and Informatics, Vol. 32, no. 4, pp. 667-680, 2015. DOI: https://doi.org/ 10.1016/j.tele.2015.03.001

[64] S. Ahmadi, "LTE-Advanced: a practical systems approach to understanding 3GPP LTE releases 10 and 11 radio access technologies", Academic Press, 2013.

[65] H. Ji, Y. Kim, J. Lee, E. Onggosanusi, Y. Nam, J. Zhang, B. Lee and B. Shim, "Overview of full-dimension MIMO in LTE-advanced pro", IEEE Communications Magazine, Vol. 55, No. 2, pp.176-184, 2016.

[66] Vincent W. S. Wong, Robert Schober, Derrick Wing Kwan Ng, Li-Chun Wang: Key Technologies for 5G Wireless Systems, Cambridge University Press, 2017.

[67] Mischa Dohler, Takehiro Nakamura, Afif Osseiran, Jose F. Monserrat, Patrick Marsch: 5G Mobile and Wireless Communications Technology, Cambridge University Press, 2016.

[68] E. Kammoun, F. Zarai, and M.S. Obaidat "Chapter 6 – LTE and 5G systems", Smart Cities and Homes, Key Enabling Technologies, pp. 111–129, 2016. DOI: https://doi.org/10.1016/B978-0-12-803454-5.00006-7

[69] https://www.cisco.com

[70] https://www.bell.ca/Bell_Internet/Wireless-to-the-Home

[71] El Khaled, Z. , & Mcheick, H. (2019). Case studies of communications systems during harsh environments: A review of approaches, weaknesses, and limitations to improve quality of service. International Journal of Distributed Sensor Networks. https://doi.org/10.1177/1550147719829960.

[72] http://digicom.ca/

[73] https://www.ui.com/

[74] https://connectivity.lairdtech.com/rf-antennas

[75] https://www.doodlelabs.com/products/fcs/

[76] Sati, Govind. (2014). A REVIEW ON OUTDOOR PROPAGATION MODELS IN RADIO COMMUNICATION. International Journal of Computer Engineering & Science. 4. 64-68.

[77] Phillips, Caleb & Sicker, Douglas & Grunwald, Dirk. (2013). A Survey of Wireless Path Loss Prediction and Coverage Mapping Methods. Communications Surveys & Tutorials, IEEE. 15. 255-270. 10.1109/SURV.2012.022412.00172.

[78] Ostlin, Erik & Suzuki, Hajime & Zepernick, Hans-Jürgen. (2008). Evaluation of the Propagation Model Recommendation ITU-R P.1546 for Mobile Services in Rural Australia. Vehicular Technology, IEEE Transactions on. 57. 38 - 51. 10.1109/TVT.2007.901902.

[79] Kumari, Manju & Yadav, Tilotma & Yadav, Pooja & Sharma, Purnima & Sharma, Dinesh. (2011). Comparative Study of Path Loss Models in Different Environments. International Journal of Engineering Science and Technology. 3.

# Improving engagement assessment in gameplay testing sessions using IoT sensors

Cristiano Politowski
*ORCiD: 0000-0002-0206-1056*
*Concordia University*
Montreal, Canada
c_polito@encs.concordia.ca

Fabio Petrillo
*ORCiD: 0000-0002-8355-1494*
*Université du Québec à Chicoutimi*
Chicoutimi, Canada
fabio@petrillo.com

***Abstract*—Video game industry is a multimillionaire market which makes solo indie developers millionaire in one day. However, success in the game industry it is not a coincidence. Video game development is an unusual kind of software that mix multidisciplinary teams, as software engineers, designer and artists. Further, for a video game be well received, it must be fun and polished, so exhaustively well tested.**

**Testing in video game development ranges from different types, in different parts of the process. For example, measuring the engagement of players in a test session can drive the development drastically. The designers/developers analyze actions taken by players and how they behave facing each decision in the game. Based on that, they decide if that feature/level requires rework or cut it. It is very common to throw out many hours of man/work in a feature just because it is not fun.**

**As the designers (usually) assess the gameplay session by hand, how can they be sure that specific feature is (or is not) good enough? If we could provide more meaningful data for the designers to review we can have a better assessment of what is happening in the gameplay, what could be wrong and if there is a real need to remove or rework.**

**In this paper, we propose an IoT environment platform to assess the player's engagement in gameplay session by adding IoT sensors together with game devices which will produce a rich output for the designers.**

***Index Terms*—iot, sensors, game development, testing, game design**

## I. Introduction

For decades, playing video games had been a joyful hobby for many people around the world [1]. It is not surprising this industry is multibillionaire surpassing the cinema and music combined [2]. However, the fantastic graphics and smooth gameplay hide constant and non-ending problems regarding the game development [3]. Most of which are related to bad practices and management, leaving a trail of burnout developers after long period of crunchs[1] [4, 5, 6, 7, 8, 9].

For a game to be well received by players, it must be a well-polished product. One key aspect of this process is testing. Testing a game implies not only searching for bugs but also finding the fun aspects. The aspects are how engaged a player (tester) is in the game regarding a specific feature or build. This task is the gameplay testing[2]. Gameplay testing means endless iterations by development teams in the last mile of the production. These iterations sometimes involve months of crunches by the team [10].

Gameplay testing sessions are crucial to delivering the fun (successful) game. In this sessions, testers play a specific build of the game, most of the time not knowing the game, on which, at the same time, the game designer assess the level of the game or a feature recently implemented. In the end, what the developers want to see is if the game is fun and if the players were engaged while playing.

Consequently, gameplay testing is hard. Video games are complex systems that include a large number of actions to be tracked [11]. As stated by Pascarella et al. [12], automate tests in the context of video game development is hard given its differences from traditional software. However, we claim the information gathered through a game session is not enough to explain the behaviour of the player/tester. In the majority of cases, the developers have only the actions performed by testers and their vocal or corporal expressions. It can lead to wrong interpretations which can cause delays in the development and, in the worst case, the commercial failure of the product.

Another issue is that game designers rely only upon their expertise and senses to asses the gameplay session. There is no metrics nor data to compare or be based. The lack of metrics happens because engagement and fun are abstract attributes and there is not (yet) a straightforward way to measure it.

To address all these concerns, we claim that a more rich set of data regarding the gameplay session can improve the game designer's judgment about the feature being tested. Which can contribute to the development process pace by (1) avoiding discarding an implemented feature; if the implemented feature was good enough, consequently increasing the reliance in the game success and improve the team performance; (2) raise the awareness of the path the game design is going to if the game/feature is not fun.

In this paper, we propose an IoT environment platform to assess the player's engagement in gameplay session by adding

[1]In video game development, crunch time is the period where developers work extra hours to complete the tasks and deliver the game in time.

[2]Gameplay testing could be playing a prototype, a slice of the final product or all the game.

IoT sensors together with game devices which will produce a rich output for the designers.

Several approaches are trying to measure the fun and engagement in video games (see section II) by creating a set of metrics, postmortems forms or biometric measures of the body. However, our solution aims to extend and improve those approaches by (1) using low-cost sensors allowing indie developers to afford and customize the solution; (2) aiming to specific requirements, that is, serve as a tool for video game designers; and (3) adding more sensors/information that are relevant to assess the gameplay testing session.

The proposal is to use biometric data, together with a screen recorder and joystick input tracker to enhance what the game designers can see about the behaviour of the testers. It can show spikes in the data related to a specific moment in the game.

The Figure 1 shows the workflow of the process we propose. The yellow blocks represent the game development team; the red block is regarding the test team, which may or may not be part of the main team; and finally, the blue blocks depict where our approach lies. As the development is iterative, we can consider the beginning when the development team produce a new build of the game (sometimes referred to as vertical slice). The tester then plays the game while the sensors capture the data, synchronize the information and compile a video with all the information. Lastly, the game developers analyze the video with the information and make his/her judgment. The process is then restarted as soon as the team decides to produce a new build or check a new feature.

The remainder of the paper is organized as follows. In the Section II we present the related works and how our proposal differ from it. In the Section III we describe how is the IoT architecture and how we are going to implement it. In Section IV we discuss the benefits of our approach and some threats. Finally, the summary of the paper in the Section V.

## II. Related Work

Some authors already tried to use biometrics feedback to assess the gameplay session. Clerico et al. [13] proposed a "predictive model" that show the fun experience of players based on the physiological responses by using biometric indicators as ECG, EMG, EDA and respiratory activity. To do so, they used a set of variables for each metric in a Support Vector Machine (SVM) to classify the fun. They validated it with a postmortem review of the gameplay made by the player after the session.

Martey et al. [14] attempted to measure the engagement of players using self-report, content analyses of videos, electrodermal activity, mouse movements, and click logs. They concluded that engagement is complex to be measure by a single attributes and should be measure by a set of metrics, that is, "engagement is a multi-dimensional construct".

Johnson et al. [15] tested how diversity in games affect players. They used psychophysiological measure like electrodermal activity (EDA) and heart-beat rate (ECG) as well as post-experimental forms in video game sessions. They found that more diversity in the game results in better enjoyment by the players.

Moura et al. [11] propose a method to better analyze players' behaviour in a specific set of games, in this case, RPGs or Action/Adventure where navigation, collection and talking with NPCs are important for the game. The tool proposed helps to visualize the players': time spent in each area, interaction with characters, if maps were activated, items collected, and characters' path. Their contributions allow the designers to: see the temporal progression of the player; compare the behaviors of different players to check the play styles. They do not explain how the tool was done, but we can infer that they collect data from the game directly.

Roose [16] proposed a method to evaluate the skill of players by using interviews (Cognitive Task Analysis) and eye tracking. Fowler et al. [17] used a tool called SMI RED500 to track the eye blinks of each participant with the goal to measure the learning curve in games. Saas et al. [18] explored the players' pattern by analyzing the meta-data from the games.

Some authors used non-biometric approaches to assess the gameplay. Fowler [19] proposed a method to qualify and quantify the learning aspect during video game sessions in children from 3 to 5 years old. They do not use bio-metric measures, where the assessment consisted only in normal observation. Ravaja et al. [20] investigated the emotional response patterns with 37 players by playing different games in random order. They assessed by using post-experiment forms. Desurvire et al. [21] tested the method of assessment of non-game usability professionals in a game context. To do so, they used forms after the game session. Tyack et al. [22] investigate what brings engagement to players using survey and interviews.

Some works tried to analyze the players behavior to improve the gameplay on-the-fly. Ang [23] investigated the impact of dynamic difficulty in games by using interviews to gather the feedback from the players. Some authors used AI in dialogs [24], wireless signals [25], and facial electromyography (EMG) [26], to improve the gameplay experience by detecting the players' emotions and changing the game accordingly.

There are some attempts to use IoT sensors to monitor ECG for the health care domain. Lacirignola and Pasero [27] created an ad-hoc solution for ECG monitoring with low energy consumption and less noise in the data. The authors made use of a self made board to this purpose.

Walke and Deshpande [28] proposed, although not validated, a architecture that makes use of cloud computing to gather, in real time, the data from ECG sensors.

Gia et al. [29] presented a low cost and energy efficiency wearable device to monitor the ECG, respiration rate, humidity, body and room temperature. Also, they presented a Fog node to display the data in graphical form. Their focus is on remote health monitoring. In this case, again, they used an ad-hoc approach, building the architecture from scratch.

An interesting proposal is from Aleman-Soler et al. [30] which used *Arduino* and *Libelium* for build a low cost solution

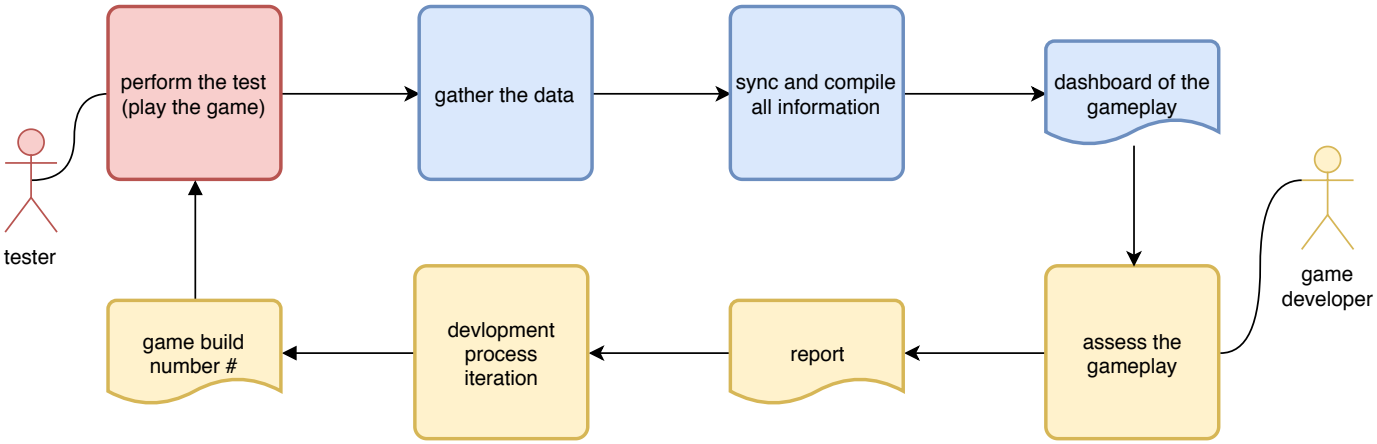



Figure 1: Workflow of the framework.

to monitor biomedical signals as Electromyogram (EMG), Electrocardiogram (ECG) and the Galvanic Skin Response (GSR). The goal of their approach is gather different biomedical signal to improve health problems detection.

> Our approach borrow some of these ideas and brings it to the light of IoT and Software Engineering. We are trying to extend the analysis of gameplay sessions by combining more sensors. Also, we do not rely on players feedback using forms or interviews. Many, if not all approaches cited, use a off-the-shelf solution (black box), which is expensive and not extensive/customized solution. Our focus is on low cost sensors and programmable devices where the development team can modify by their needs. Finally, we are focusing in a tool to aid developers, more specifically, game designers. Aside form the work from Moura et al. [11], other authors had other objectives in theirs papers.

## III. Implementing the Architecture

The Figure 2 shows the proposed architecture in UML2 component diagram. The whole system is based on low cost sensors and *Arduino*[3]. The basic setup for a game session is the **screen**, **game system** (console, computer, etc.) plus the game (version of the build to be tested) and a input device, **joystick** in this case.

The **screen recorder** here is described as a software, but there are some specific devices for this purpose, however, as we are focusing in low cost sensors, we decided to use a simple screen recorder software. The same for the **input tracker** in the joystick. We use a software to map the commands. The data from both are put together by the main server and then stored.

As for the **camera**, there is no caveats. It is a camera focusing on the tester, all the time. The data is then sent directly to the main server as it does not need treatment.

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

The **force sensitive resistor** is attached to the joystick to capture the pressure force made by the tester. It them send this data to the **edge node** to be normalized and then stored in the main server.

Finally, the bio-metrics are taken by a set of sensors, **ECG**, **EMG**, **EDA**, and **GSR**. These sensors are connected with an *Arduino* board which is responsible for gather this data and send them to the **edge node** for pre-processing. After that, the edge send it to the main server to be stored.

We will briefly describe each sensor, however, their functionalities as well how to measure their output are not in the scope of the article.

### A. Electrocardiography (ECG)

Electrocardiography (ECG or EKG) is the process of recording the electrical activity of the heart over a period of time using electrodes placed over the skin [31]. The electrocardiogram (ECG) is the graphical representation of the electrical activity of the heart, as the Figure 3 shows.

In this regard, we can add electrodes on the skin of the tester to monitor the changes provoked by the hear beat. As the output is line with a pattern, we can check the variance and spikes and, therefore, correlate with the level/area in the game. More research must be done to list what each pattern in the line means. As an example, the AD8232 sensor for *Arduino* is a low cost alternative.

### B. Electrodermal Activity (EDA)

The principle of Electrodermal Activity (EDA) is that skin resistance varies with the state of sweat glands in the skin, which sweating is controlled by the sympathetic nervous system. In this way, skin conductance can be a measure of emotional and sympathetic responses [32].

EDA is associated with emotion and cognitive processing, moreover, some emotional responses, like threat, anticipation, salience, and novelty, may occur unconsciously [33]. Additionally, EDA peak (height and rate) (see Figure 4) describe the stress level of a person [34].

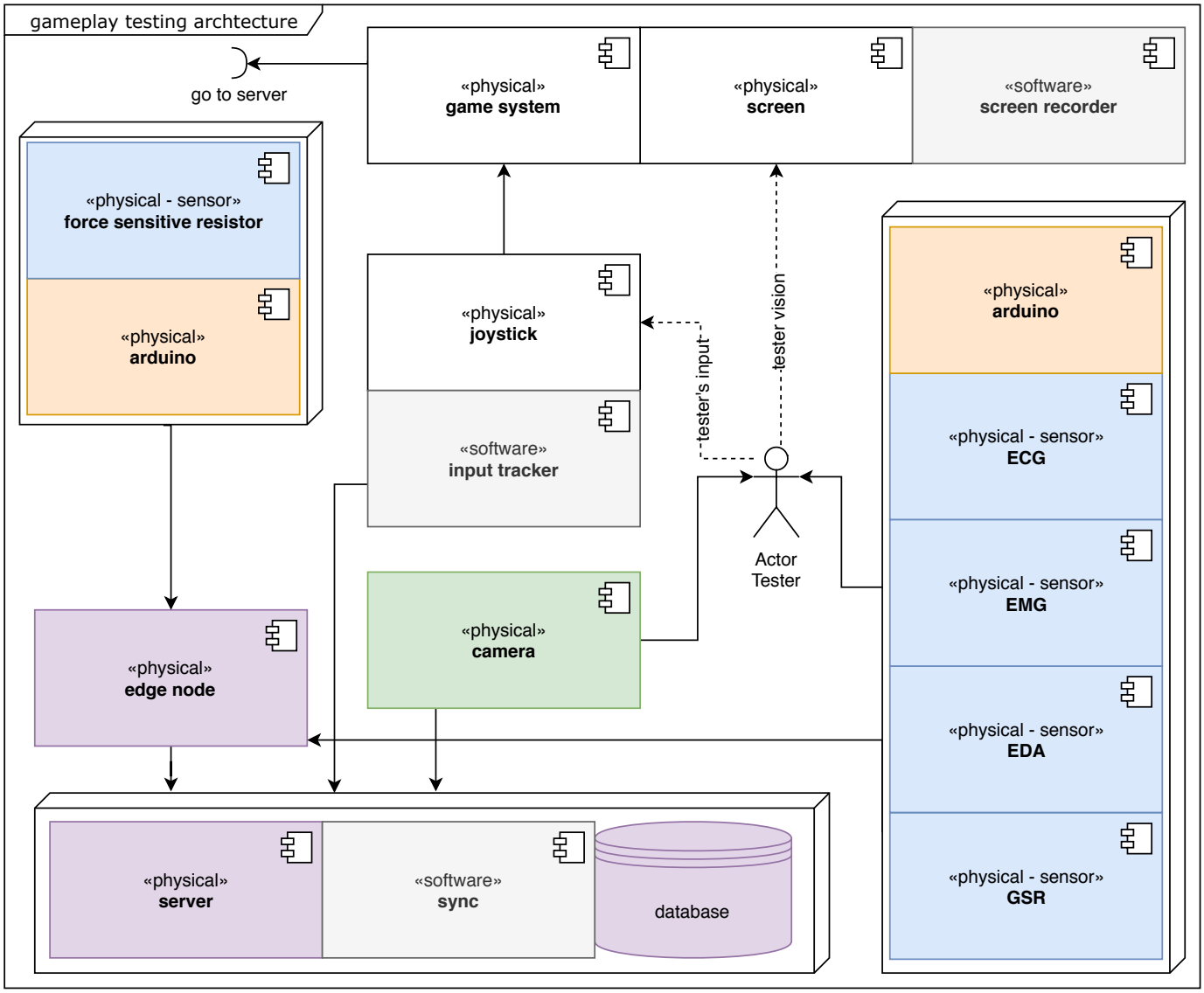



Figure 2: Proposed architecture in UML2 component diagram.

With this method we can measure the excitement and arousal of the tester during the session. A lower cost option is using two electrodes in a *Arduino*, inducing voltages through one electrode and measuring using a second one.

### C. Electromyography (EMG)

Electromyography (EMG) is a technique that record electrical activity produced by skeletal muscles [36]. The output of this measure is the electromyogram (see Figure 5). It detects potential difference that actives the muscle cells, which can be used to detect abnormalities in the movement. The less invasive method to measure EMG is using electrodes to control the overall activation of the muscle [30]. A lower cost sensor for *Arduino* is the MyoWare Muscle Sensor.

### D. Galvanic Skin Response (GSR)

Galvanic Skin Response (GSR) is the property of the human body that causes continuous variation in the electrical characteristics of the skin. GSR measure the electrical condictance of the skin, which varies according to sweat glands, which in turn is controled by the sympathetic nervous system, finally indicating psychological or physiological arousal [30]

The measurement is made putting sensors in two fingers. The more the subject sweat level increases, the bigger is the conductivity, as, for example, in the graph output in Figure 6.

### E. Force Sensitive Resistor (FSR)

Force Sensitive Resistor (FSR) is a sensor that allow to detect physical pressure, squeezing and weight. We can attach this sensor on the joystick and measure with which intensity the tester are holding it. It can show how he behave facing certain scenarios of the game. With a proper baseline, we can even infer the player boredom and excitement.

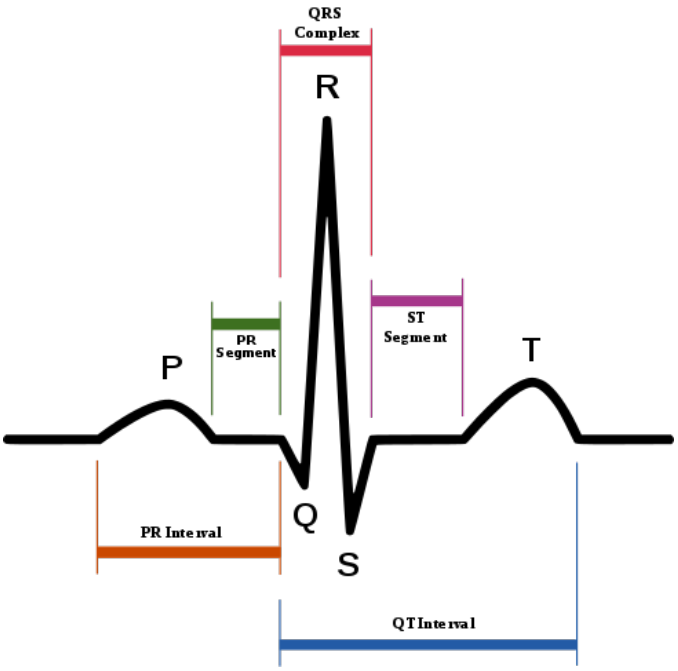



Figure 3: Schematic diagram of normal sinus rhythm for a human heart as seen on ECG [31].

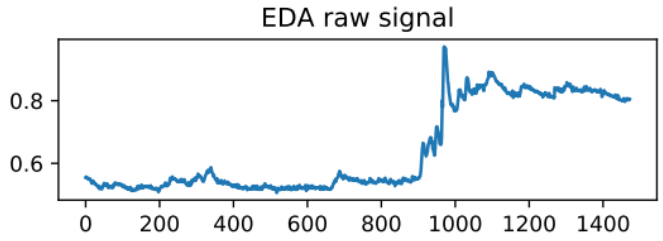



Figure 4: The decomposition of an EDA signal [35].

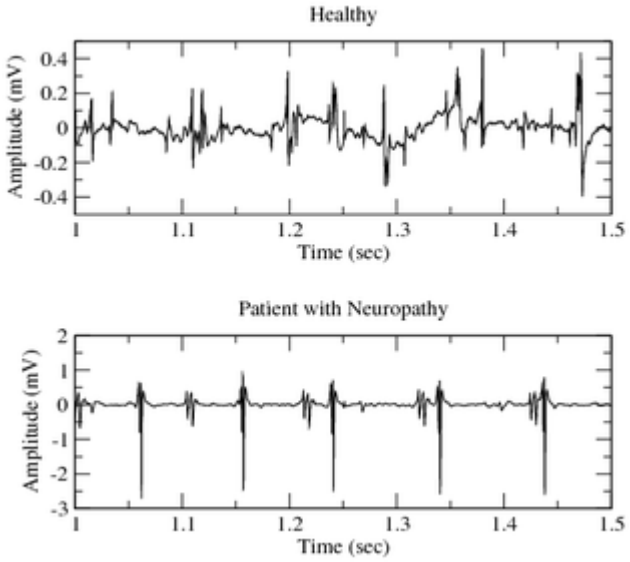



Figure 5: Electromyogram output example [37].

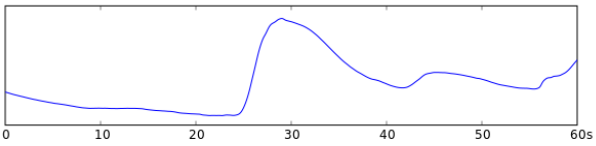



Figure 6: A sample GSR signal of 60 seconds duration [38].

### *F. Joystick Input Tracker, Camera and Screen Recorder*

The idea of the **Joystick Input Tracker** is to get all the commands (input) performed by the tester and map it to a virtual representation of the same and display it on the screen. It can help keep track of some detected bug of failure captured during the gameplay session and reproduce it during the development.

The purpose of the **camera** is simple: record the tester and to observe his reactions. The body expressions might reveal interesting things about the tester emotions. Moreover, we can apply some image pattern recognition.

Finally, the **screen recorder**, which is the main link between all the sensors and data. Without the gameplay video the game designers cannot correlate the gathered information and the part of the game.

### *G. Dashboard*

The Figure 7 show the dashboard comprising the output of the sensors after synchronizing with the gameplay video. Aside of the gameplay scene, it shows the graph, in real-time, of the data from all the sensors (ECG, EMG, EDA, GSR, and FSR) in the upper left corner. The footage of the tester is on the lower left corner with the virtual joystick inputs. We are adding the FPS (frames per second) and APM (actions per minute) to enhance the information set.

## IV. Discussion and Threats

Our approach aims to use the concepts of finding fun and engagement by making use of biometrics to aid game designers to assess their game features, on the development phase, during the gameplay sessions. It focuses on low-cost sensors and an extensible platform.

The idea also can be applied for the validation of a concept, during the pre-production, where the developers test new ideas with prototypes. In this case, a more robust measurement can prevent many months of rework or even years of development.

Although the related problem is to help video game designers in their task, the underlying issues on how to build and synchronize the IoT architecture is real. The related works that used biometrics to asses the gameplay focused on black-box solutions, which prevent any change or adding new sensors that the developers might want to use.

With the amount of information gathered from the sensors, we can apply machine learning to cluster some of the attributes (pattern recognition) and transform them into metrics. For example, a determined type of spikes in the ECG graph can imply a specific emotion or difficulty in the game. A bug, for instance, can induce a typical reaction of the tester, and


Figure 7: Dashboard of the prototype. The image is from the game Darkest Dungeon.

with this, we can generate a report with all possible bugs to investigate.

Aside from testing new features, this kind of enhanced feedback can bring new light to the game design. By observing the tester reaction and data, we can make correlations with parts of the gameplay that it enjoyed more. With that, developers can extract the core mechanic and apply to other games. It can become a library of core mechanics, rated by "fun level", that can be useful in new projects.

The output of this project can be interesting for researches in the design field as well. There are many attempts to measure the engagement in video games, and a platform that is customizable should help them to propose and test new hypotheses.

As threats, we need to mention that maybe it will not be possible to use only one *Arduino* to manage all sensors at once. Moreover, the setup of the board with the sensor may disturb the tester during the session. Finally, the synchronization of all the sensors data is very sensible, the delay of seconds can lead to a wrong interpretation of the results.

## V. Preliminary Conclusions

In this paper, we present an IoT architecture to assess video game testing. It is composed of a set of sensors and applications forming a low-cost framework which can be afforded by independent studios and developers. The goal of this approach is to provide a contextual output of the gameplay session, that is, besides the gameplay video, information regarding the biometrics of the tester or user as well as technical details of the game. With such solution, game developers (especially game designers) can use and customize their game projects, by gathering a more rich set of data from the gameplay test sessions, and, therefore, improve the quality of their games.

Although our approach project is large, it can be made in modules, so one sensor/module by the time. For example, the first step can be the set up the screen recorder, the edge node and server. Then adding the input tracker for the joystick and so on.

Although the contribution of this approach aims to solve a problem in video game development, the solution comes from IoT area. The challenge of creating and implementing this architecture and synchronize all the data is, in our words, relevant to the IoT community.

Last but not least, the outcome this tool can provide is broad regarding what can be done with the gathered data. By reasoning on the extracted information, we can create a model to evaluate, with a set of metrics, the gameplay session attributes, like engagement and fun. Then, our approach can improve the time to assess the gameplay session and its efficiency.

## References

[1] Entertainment Software Association - ESA, "Esa's essential facts about the computer and video game industry report," 2019, [Online; accessed 7-

February-2019]. [Online]. Available: http://www.theesa.com/wp-content/uploads/2018/05/EF2018_FINAL.pdf

[2] Newzoo, "2018 global games market report," 2019, [Online; accessed 7-February-2019]. [Online]. Available: https://resources.newzoo.com/hubfs/Reports/Newzoo_2018_Global_Games_Market_Report_Light.pdf

[3] F. Petrillo, M. Pimenta, F. Trindade, and C. Dietrich, "What went wrong? a survey of problems in game development," *Computers in Entertainment*, vol. 7, no. 1, p. 1, Feb. 2009.

[4] J. Schreier, "Inside rockstar games' culture of crunch," https://kotaku.com/inside-rockstar-games-culture-of-crunch-1829936466, accessed: 2019-02-06.

[5] T. Phillips, "The human cost of red dead redemption 2 according to the people who made it," https://kotaku.com/inside-rockstar-games-culture-of-crunch-1829936466, accessed: 2019-02-06.

[6] H. Edholm, M. Lidstrom, J.-P. Steghofer, and H. Burden, "Crunch Time: The Reasons and Effects of Unpaid Overtime in the Games Industry," in *2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP)*. IEEE, May 2017.

[7] A. Peticca-Harris, J. Weststar, and S. McKenna, "The perils of project-based work : Attempting resistance to extreme work practices in video game development," *Organization*, vol. 22, no. 4, pp. 570–587, Jun. 2015.

[8] J. R. Whitson, "What can we learn from studio studies ethnographies? a "messy" account of game development materiality, learning, and expertise," *Games and Culture*, vol. 0, Jun. 2018. [Online]. Available: https://doi.org/10.1177/1555412018783320

[9] C. Politowski, L. M. Fontoura, F. Petrillo, and Y.-G. Guéhéneuc, "Learning from the past: A process recommendation system for video game projects using postmortems experiences," *Information and Software Technology*, vol. 100, pp. 103–118, aug 2018.

[10] J. Schreier, *Blood, Sweat, and Pixels: The Triumphant, Turbulent Stories Behind How Video Games Are Made*. HarperCollins, 2017. [Online]. Available: https://books.google.ca/books?id=-bK-DQAAQBAJ

[11] D. Moura, M. S. el Nasr, and C. D. Shaw, "Visualizing and understanding players' behavior in video games," in *Proceedings of the 2011 ACM SIGGRAPH Symposium on Video Games - Sandbox '11*. ACM Press, 2011, pp. 11–16. [Online]. Available: http://odin.opendatawatch.com/Downloads/otherFiles/ODIN-2016-Methodology.pdf

[12] L. Pascarella, F. Palomba, M. D. Penta, and A. Bacchelli, "How Is Video Game Development Different from Software Development in Open Source?" no. May, 2018.

[13] A. Clerico, C. Chamberland, M. Parent, P. E. Michon, S. Tremblay, T. H. Falk, J. C. Gagnon, and P. Jackson, "Biometrics and classifier fusion to predict the fun-factor in video gaming," *IEEE Conference on Computatonal Intelligence and Games, CIG*, 2017.

[14] R. M. Martey, K. Kenski, J. Folkestad, L. Feldman, E. Gordis, A. Shaw, J. Stromer-Galley, B. Clegg, H. Zhang, N. Kaufman, A. N. Rabkin, S. Shaikh, and T. Strzalkowski, "Measuring Game Engagement: Multiple Methods and Construct Complexity," *Simulation and Gaming*, vol. 45, pp. 528–547, 2014.

[15] D. Johnson, M. Klarkowski, K. Vella, C. Phillips, M. McEwan, and C. N. Watling, "Greater rewards in videogames lead to more presence, enjoyment and effort," *Computers in Human Behavior*, vol. 87, no. March, pp. 66–74, oct 2018.

[16] K. Roose, "The Tracer Method : The Dynamic Duo Combining Cognitive Task Analysis and Eye Tracking," pp. 585–593, 2018.

[17] A. Fowler, B. Cusack, and A. Canossa, "Measuring learning in video games," in *Proceedings of the 2014 Conference on Interactive Entertainment - IE2014*, vol. 02-03-Dece. ACM Press, 2014. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-84955586007{&}partnerID=40{&}md5=1940d797d50f04b5ea38aa6ab3d33663

[18] A. Saas, A. Guitart, and A. Perianez, "Discovering playing patterns: Time series clustering of free-to-play game data," *IEEE Conference on Computatonal Intelligence and Games, CIG*, pp. 1–8, 2017.

[19] A. Fowler, "Measuring learning and fun in video games for young children: A proposed method," *ACM International Conference Proceeding Series*, pp. 639–642, 2013. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-84880556499{&}partnerID=40{&}md5=5ad7fa6ce0daa09705d9de437df526ce

[20] N. Ravaja, M. Salminen, J. Holopainen, T. Saari, J. Laarni, and A. Jrvinen, "Emotional response patterns and sense of presence during video games," in *Proceedings of the third Nordic conference on Human-computer interaction - NordiCHI '04*. ACM Press, 2004, pp. 339–347. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1028014.1028068

[21] H. Desurvire, M. Caplan, and J. A. Toth, "Using heuristics to evaluate the playability of games," in *Extended abstracts of the 2004 conference on Human factors and computing systems - CHI '04*, vol. 1, no. 2. ACM Press, 2004, pp. 64–75. [Online]. Available: http://www.upassoc.org/upa{_}publications/jus/2006{_}february/usability{_}game{_}development.html

[22] A. Tyack, P. Wyeth, and D. Johnson, "The appeal of MOBA games," in *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play - CHI PLAY '16*. ACM Press, 2016, pp. 313–325. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2967934.2968098

[23] D. Ang, "Difficulty in Video Games: Understanding the Effects of Dynamic Difficulty Adjustment in Video Games on Player Experience," *Proceedings of the 2017 ACM SIGCHI Conference on Creativity and Cognition*,

pp. 544—-550, 2017.

[24] J. Fraser, I. Papaioannou, and O. Lemon, "Spoken conversational AI in video games," in *Proceedings of the 18th International Conference on Intelligent Virtual Agents - IVA '18*. ACM Press, 2018, pp. 179–184.

[25] M. Zhao, F. Adib, and D. Katabi, "Emotion recognition using wireless signals," in *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking - MobiCom '16*. ACM Press, 2016, pp. 95–108. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2973750.2973762

[26] R. L. Hazlett, "Measuring emotional valence during interactive experiences," in *Proceedings of the SIGCHI conference on Human Factors in computing systems - CHI '06*. ACM Press, 2006, p. 1023. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1124772.1124925

[27] F. Lacirignola and E. Pasero, "Hardware design of a wearable ECG-sensor: Strategies implementation for improving CMRR and reducing noise," in *2017 European Conference on Circuit Theory and Design (ECCTD)*. IEEE, Sep. 2017, pp. 2–5.

[28] S. M. Walke and R. S. Deshpande, "On-line real time feature extraction of ECG signal: Recent advances & survey," in *2015 International Conference on Information Processing (ICIP)*. IEEE, Dec. 2015, pp. 211–216.

[29] T. N. Gia, M. Jiang, V. K. Sarker, A. M. Rahmani, T. Westerlund, P. Liljeberg, and H. Tenhunen, "Low-cost fog-assisted health-care IoT system with energy-efficient sensor nodes," in *2017 13$^{th}$ International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, Jun. 2017, pp. 1765–1770.

[30] N. M. Aleman-Soler, C. M. Travieso, E. Guerra-Segura, J. B. Alonso, M. K. Dutta, and A. Singh, "Biometric approach based on physiological human signals," in *2016 3$^{rd}$ International Conference on Signal Processing and Integrated Networks (SPIN)*. IEEE, Feb. 2016, pp. 681–686.

[31] Wikipedia contributors, "Electrocardiography — Wikipedia, the free encyclopedia," https://en.wikipedia.org/w/index.php?title=Electrocardiography&oldid=882128916, 2019, [Online; accessed 7-February-2019].

[32] W. Boucsein, *Electrodermal activity*. Springer Science & Business Media, 2012.

[33] A. Greco, A. Lanata, L. Citi, N. Vanello, G. Valenza, and E. Scilingo, "Skin admittance measurement for emotion recognition: A study over frequency sweep," *Electronics*, vol. 5, no. 4, p. 46, Aug. 2016. [Online]. Available: http://www.mdpi.com/2079-9292/5/3/46

[34] C. Setz, B. Arnrich, J. Schumm, R. L. Marca, G. Troster, and U. Ehlert, "Discriminating stress from cognitive load using a wearable EDA device," *IEEE Transactions on Information Technology in Biomedicine*, vol. 14, no. 2, pp. 410–417, Mar. 2010.

[35] S. Jacob, S. Ishimaru, and A. Dengel, "Interest detection while reading newspaper articles by utilizing a physiological sensing wristband," in *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers - UbiComp '18*. ACM Press, 2018, pp. 78–81.

[36] G. Robertson, G. Caldwell, J. Hamill, G. Kamen, and S. Whittlesey, *Research methods in biomechanics, 2E*. Human Kinetics, 2013.

[37] W. Commons, "File:electromyogram.png — wikimedia commons, the free media repository," https://commons.wikimedia.org/w/index.php?title=File:Electromyogram.png&oldid=199110369, 2016, [Online; accessed 16-February-2019].

[38] Wikipedia contributors, "Electrodermal activity — Wikipedia, the free encyclopedia," https://en.wikipedia.org/w/index.php?title=Electrodermal_activity&oldid=877603128, 2019, [Online; accessed 16-February-2019].

# Users Issues in using the Internet of Things Systems

Andrew Truelove, Farah Naz Chowdhury, Omprakash Gnawali, and Mohammad Amin Alipour
Department of Computer Science, University of Houston, United States

***Abstract*—Internet of Things (IoT) systems are bundles of networked sensors and actuators that are deployed in an environment and act upon the sensory data that they receive. These systems, especially consumer electronics, have two main cooperating components: a device and a mobile app. The unique combination of hardware and software in IoT systems presents challenges that are lesser known to mainstream software developers and might require innovative solutions to support the development and integration of such systems.**

**In this paper, we analyze the more than 90,000 reviews of ten IoT devices and their corresponding apps and extract the issues that users encountered in using these systems. Our results indicate that issues with *connectivity*, *timing*, and *update* are particularly prevalent in the reviews. Our results call for a new software-hardware development framework to assist the development of reliable IoT systems.**

## I. Introduction

Internet of Things systems (IoT) are sets of interconnected sensors and actuators that potentially backed and managed by servers on the Internet. These systems are becoming part of "smart" solutions to the everyday life of users. For example, traditional thermostat, a solution for controlling the room temperature, can be replaced by smart thermostat that can extract the users' preferences and can be controlled remotely.

Despite the popularity of IoT solutions, the development of such systems still seems to a form of art, and the potential issues facing users are largely unknown. A systematic identification of problems would enable researchers to devise tools, techniques, and frameworks to support effective development of such systems. In this paper, we use the users review on Amazon and Google marketplaces to elicit the issues in IoT systems. We particularly focus on IoT consumer electronics that are used by home users. Most consumer electronics have two main components: a physical device, and a mobile app. Marketplaces such as Amazon.com and app stores allow users to leave reviews about devices and the mobile apps.

In this paper, we analyze over 90,000 reviews from ten IoT consumer electronic systems to understand what are the common issues that users are facing. We evaluate all reviews from January to mid-October 2018 for ten popular devices from Amazon.com and their corresponding Android apps from Google Play. Our results indicate that issues with *connectivity*, *timing*, and *update* are particularly prevalent in the reviews. The results call for new software-hardware development framework to assist development of reliable IoT systems.

**Contributions.** This paper makes the following contributions.

- We identify technical issues in ten consumer IoT systems by analyzing users reviews on Amazon and Google Play.
- We make data and analysis code available.

## II. Related Work

There is a large body of work in analyzing users' reviews to elicit the issues in software systems. To the best of our knowledge, extracting users' issues in IoT technology, at least in the form of consumer electronics, have not been explored.

Atrozi et al. [4] survey the definitions, architecture, fundamental technologies, and applications of the Internet of Things. They note that IoT has been deployed in the area of mobile apps and that mobile devices will expand the IoT market as they continue to develop. Alur et al. [3] provide a list of challenges in development of IoT systems. Fu et al. [7] report the potential safety and security issues in IoT systems.

Maalej and Nabil [14] used the probabilistic technique to automatically classify app reviews into one of four types: bug reports, feature requests, user experience and ratings. Hoon et al. [11] studied how reviews evolve over time and the characteristics of the reviews. Pagano and Maalej [16] explore how and when users leave their feedback and also analyzed the content of the review. AppEcho allows users to leave feedback in situ that is exactly when the user discovered an issue [17]. AR-Miner extracts informative information from the reviews using topic modeling [5].

Hermanson [10] looked at whether perceived ease of use and perceived usefulness were widely discernable in user reviews submitted to apps on the Google Play Store. The author collected 13,099 reviews from the Google Play Store and found that only 3% of the reviews had information relating to perceived usefulness and that less than 1% of the reviews had any mention of perceived ease of use.

CLAP is a tool to help developers parse through app reviews when rolling out an update [18]. CLAP categorizes user reviews based on the information they contain. It groups related reviews together, and then automatically prioritize which groups should be prioritized for the next app update. Gu and Kim [9] propose SUR-Miner, a pattern –based parsing technique which parses aspect-opinion pairs from review sentences to produce effective user review summarization. Di Sorbo et al. [6] propose a tool called SURF which summarizes thousands of app reviews and generates a detailed interactive agenda on recommended updates and changes to the app. Licorish et al. [13] used content analysis and regression to provide insights into the nature of reviews provided by the users. Mujahid et al. [15] looked at user reviews of wearable apps. The authors manually sampled and categorized 6 android wearable apps. They found that the most frequent complaints involved functional errors, lack of functionality and cost.

TABLE I: IoT Devices and Applications Used in this Study

| First Row: Name of App<br>Second Row: Name of Device | Description |
|---|---|
| Amazon Alexa<br>Amazon Echo Dot (2nd Gen) | A virtual assistant. App connects to a variety of devices with speakers and microphones that allows the user to interface with the service. |
| ecobee<br>ecobee4 Smart Thermostat | Connects to a thermostat that can be controlled by the app. |
| Google Home<br>Google WiFi System, 1-Pack | A virtual assistant. App connects to a variety of devices with speakers and microphones that allows the user to interface with the service. |
| Insteon for Hub<br>Insteon Hub | Connects to a hub device that, in turn, connects to a number of other Insteon devices, including light switches, lamps, and security camera. Through the hub, the user can control all connected devices with the app. |
| Kevo<br>Kevo Lock (2nd Gen) | Connects to a door lock that can be installed in the user's door. Lock can be controlled with the app. |
| Nest<br>Nest T3007ES Thermostat | Connects to a thermostat that can be controlled by the app. |
| Philips Hue<br>Philips Hue Starter Kit | Connects to light bulbs whose intensity and color are controlled by the app. |
| SmartThings (Samsung Connect)<br>SmartThingsSmart Home Hub | Connects to a variety of Samsung-branded devices. These devices can be controlled through the app. |
| Tile<br>Tile Mate | Connects to a small, square-shaped device that can be attached to a number of personal belongings. The device connects to the internet, allowing its location to be tracked through the app. |
| WeMo<br>WeMo Mini Smart Plug | Connects to a number of WeMo-branded devices, including cameras, light bulbs, and electrical plugs. These devices can be controlled through the app. |

TABLE II: Characteristics of Reviews Considered in this Study

| System | | Total | Review Length (char) | | | | |
|---|---|---|---|---|---|---|---|
| | | | Min. | 25% | 50% | 75% | Max |
| Amazon Alexa | App Reviews | 5,785 | 1 | 18 | 56 | 135 | 2,027 |
| | Device Reviews | 54,289 | 3 | 44 | 92 | 192 | 7,632 |
| ecobee | App Reviews | 917 | 4 | 68 | 133 | 229 | 1,572 |
| | Device Reviews | 598 | 14 | 148.8 | 336.5 | 644 | 12,390 |
| Google Home | App Reviews | 7,051 | 2 | 26 | 73 | 157 | 1,996 |
| | Device Reviews | 1,859 | 9 | 102 | 240 | 468 | 9,526 |
| Insteon | App Reviews | 70 | 7 | 71.5 | 118.5 | 264.8 | 532 |
| | Device Reviews | 121 | 19 | 113 | 316 | 621 | 2,232 |
| Kevo | App Reviews | 461 | 3 | 33 | 93 | 206 | 1,724 |
| | Device Reviews | 296 | 15 | 154.8 | 337 | 719.2 | 5,016 |
| Nest | App Reviews | 1,798 | 3 | 61 | 135 | 242 | 1,877 |
| | Device Reviews | 1,431 | 9 | 83.5 | 210 | 462 | 5,139 |
| Philips Hue | App Reviews | 1,231 | 3 | 64 | 137 | 248 | 1,553 |
| | Device Reviews | 667 | 9 | 69 | 146 | 303.5 | 4,833 |
| SmartThings | App Reviews | 9,973 | 2 | 18 | 58 | 139 | 2,662 |
| | Device Reviews | 417 | 7 | 89 | 214 | 487 | 3,998 |
| Tile | App Reviews | 1,480 | 2 | 34 | 90.5 | 194 | 1,718 |
| | Device Reviews | 2,149 | 7 | 62 | 137 | 256 | 3,209 |
| WeMo | App Reviews | 3,177 | 2 | 40 | 85 | 177 | 1,833 |
| | Device Reviews | 2,013 | 5 | 100 | 215 | 385 | 7,841 |

## III. Method

In this section, we first describe the data selection and characteristics of the review data used in this study.

### A. Characteristics of Data

Table I lists the IoT systems (Devices and their corresponding apps) used in this study. These systems encompass a wide domain including conversational assistants, thermostats, electronic locks, and tracking devices. The price of the devices ranged from about $25 to $200 at the time of writing.Six of these systems were used in a previous study of IoT apps by Kaaz et al. [12], and the remaining four systems are based on a Google search for popular IoT apps. For each system, we found device on Amazon.com and the corresponding app on Google Play. We note that for some of the devices there are multiple versions of the products on Amazon website. In such cases, we chose the ones which had more reviews.

For each system, we extracted the reviews from the Amazon website and the corresponding app reviews from the Google Play Store. We collected reviews that were posted during a 10-month period starting from the beginning of January 2018 to mid-October of the same year.

Table II shows statistics about the number and length of reviews for products and apps. The table provides some noteworthy insights. For instance, with all IoT systems, the maximum review length was always higher in the device reviews than in the app reviews. It is possible that Amazon allows a higher character limit in its reviews than the Google Play Store. Moreover, users have to use a mobile phone to enter the app reviews, but they can use computers for leaving reviews for the devices on Amazon. Typing on a computer can be easier for many users than on phones.

For seven out of ten systems, more reviews were collected from the Google Play Store than Amazon. The three exceptions to this pattern are Amazon Alexa, Insteon, and Tile. With Amazon Alexa, this could be explained by the fact that Amazon is both the creator of the device and the curator of the storefront. As a first-party product, the Echo Dot likely receives some level of favoritism, likely expressed through increased promotion on the Amazon.com web site. This promotion could lead to more purchases and ultimately, more reviews. This favoritism may also explain why the Google Home app received so many more reviews than the Google Home device. The reason Insteon is an exception is probably due to the fact that it received fewer reviews overall. There is only a difference of 51 reviews between the app reviews and the device reviews. If Insteon had received more reviews during the time frame studied, the number of reviews may have more closely matched the pattern of the other systems. With Tile, no explanation for its anomalous behavior is immediately apparent. It is worth noting that Tile, as an IoT system, is fairly unique out of all the systems studied. These facts will be explored in more depth and explained in detail in the later sections of the paper.

### B. Topic Modeling

We used Latent Dirichlet Allocation (LDA) to identify the most important topics users feel most strongly about [8]. By creating topics from the text of these reviews, it is possible, that some topics will be comprised of words that speak to a component of the app or device that users are complaining about. For example, if a topic contains the words "bad", "battery", and "drain", then we could infer that complaints about battery life are a significant topic in the user reviews. We used Gensim library [1] with the default configurations to generate a list of topics. For each review, we used LDA to generate three topics and return the ten words for each topic that contributed the most to that topic.

## IV. Issues mentioned in IoT System Reviews

This section describes the result of analysis of users reviews for the systems in our study. For each IoT system, we generated three topics made up of ten words. Our results listed these ten words in the order of how much they contributed to that topic. For brevity, we discuss the analysis of two systems

in details here, and we add the results of the topics discussed in the reviews of other systems in Appendix A.

Tables III and IV depict the words for each topic for the Amazon Alexa and SmartThings apps. Tables V and VI display the topics for the corresponding devices. Beside each word is a number from 0 to 1 that reflects the magnitude at which that word contributed to the topic. When it comes to interpreting the LDA results, it was clear some words in a list appeared to be more important than others. Determining the usefulness of a word was based on a combination of its position in the list and the magnitude value the word had been assigned. A higher magnitude means a word contributed to the topic more strongly, meaning it is likely to be more integral in identifying the topic created by the LDA. At the same time, each topic list spans a different range of values between the magnitude of the first word and the magnitude of the tenth word. In some cases, the final few words had magnitudes so low to appear almost negligible, but in other cases, the final words carried magnitudes not all that lower than the value for the first word in that list.

For example, in Table V, the eighth word in Topic 3 is "christmas", which has a magnitude of 0.017. Though its position near the end of the list means this word may be one of the least important words in Topic 3, its impact is not entirely negligible. Compare the magnitude value of "christmas" in Topic 3 to the magnitudes found in Topic 1. The only word in Topic 1 with a magnitude higher than 0.017 is the first word, "speaker", which has a magnitude value of 0.021. Every word following has a lower magnitude value than "christmas". This means that "christmas" had more of an impact on its topic than nine of the ten words listed for Topic 1. This would suggest that the magnitude values of each word relative to the other magnitude values in the same topic carry more importance than the absolute position in any list.

If an IoT system is receiving significantly different rating distributions from the app store page and device store page, perhaps the kinds of topics generated from the app reviews and the device reviews may illustrate why.

### *A. Apps vs. Device*

In a very general sense, the topics for the apps had more instances of words with negative sentiment than the topics for the devices. Though there are plenty of positive words in both the app and device topics, when a negative word like slow, bad, waste, or useless does appear, it seems to be more likely to be in an app review topic. Additionally, words such as control and connect appear more prominently in the app review topics, which may be an indicator of what issues users are running into when suing the app. The word update is particularly common in the app review topics.

**Observation 1:** Topics for the apps had more instances of words with negative sentiment than the topics for the devices.

As an example, none of the topics for the SmartThings Hub device contain any significantly negative language (Table VI).

TABLE III: Amazon Alexa App LDA Topics

| Topic 1 Words | Topic 1 Magnitude | Topic 2 Words | Topic 2 Magnitude | Topic 3 Words | Topic 3 Magnitude |
|---|---|---|---|---|---|
| good | 0.037 | love | 0.021 | connect | 0.018 |
| music | 0.026 | device | 0.018 | time | 0.016 |
| play | 0.019 | update | 0.016 | wifi | 0.015 |
| great | 0.015 | slow | 0.013 | phone | 0.014 |
| nice | 0.011 | home | 0.011 | keep | 0.013 |
| amazing | 0.008 | list | 0.010 | update | 0.012 |
| control | 0.008 | awesome | 0.007 | android | 0.009 |
| song | 0.007 | take | 0.007 | device | 0.009 |
| voice | 0.006 | please | 0.007 | tried | 0.008 |
| time | 0.006 | phone | 0.007 | best | 0.008 |

TABLE IV: SmartThings App LDA Topics

| Topic 1 Words | Topic 1 Magnitude | Topic 2 Words | Topic 2 Magnitude | Topic 3 Words | Topic 3 Magnitude |
|---|---|---|---|---|---|
| great | 0.035 | phone | 0.036 | tv | 0.044 |
| love | 0.024 | uninstall | 0.030 | connect | 0.028 |
| smartthings | 0.023 | permission | 0.019 | good | 0.026 |
| device | 0.022 | update | 0.015 | device | 0.020 |
| easy | 0.016 | bloatware | 0.015 | phone | 0.017 |
| home | 0.014 | disable | 0.014 | smart | 0.015 |
| smart | 0.013 | apps | 0.014 | time | 0.013 |
| classic | 0.013 | remove | 0.012 | bluetooth | 0.011 |
| useful | 0.011 | device | 0.011 | update | 0.011 |
| awesome | 0.009 | delete | 0.011 | remote | 0.009 |
| Topic 1 Summary: Ease of Use | | Topic 2 Summary: Desire to Remove App from Device | | Topic 3 Summary: Connecting Phone with App | |

The only instance of somewhat negative language comes from a single appearance of the word issue in Topic 3, and even then, the word has a fairly low magnitude value of 0.006. Meanwhile, the topics for the SmartThings app (Table IV) contain significantly more negative language, particularly in Topic 2, where words like uninstall, bloatware, remove, and delete are all found. The presence of the words permission and update in this topic suggest that something about the SmartThings app's permission requirements and updates is being associated with users wanting to remove the app from their device.

Overall, the observations that can be made from these LDA results are fairly general. There are exceptions to the general observations identified above; some negative words do appear in topics for the device reviews, for example. Though the topics provide some guidance as to what kinds of issues users of the apps are facing, it may be possible to refine the results to make these issues more apparent. We decided to see if running an LDA specifically on the app reviews that came with a low star rating might provide more helpful information.

TABLE V: Amazon Echo Dot LDA Topics

| Topic 1 | Topic 1 Magnitude | Topic 2 | Topic 2 Magnitude | Topic 3 | Topic 3 Magnitude |
|---|---|---|---|---|---|
| speaker | 0.021 | music | 0.045 | love | 0.139 |
| device | 0.016 | play | 0.025 | great | 0.084 |
| sound | 0.016 | fun | 0.024 | easy | 0.032 |
| good | 0.015 | question | 0.019 | gift | 0.023 |
| home | 0.011 | ask | 0.017 | product | 0.020 |
| smart | 0.010 | weather | 0.015 | bought | 0.020 |
| better | 0.009 | thing | 0.015 | room | 0.017 |
| time | 0.008 | time | 0.013 | christmas | 0.017 |
| voice | 0.008 | answer | 0.013 | house | 0.015 |
| quality | 0.007 | know | 0.013 | family | 0.012 |
| Topic 1 Summary: Good Sound Quality | | Topic 2 Summary: Capbility of Features | | Topic 3 Summary: Good Gift for Family | |

TABLE VI: SmartThings Hub LDA Topics

| Topic 1 Words | Topic 1 Magnitude | Topic 2 Words | Topic 2 Magnitude | Topic 3 Words | Topic 3 Magnitude |
|---|---|---|---|---|---|
| light | 0.011 | device | 0.020 | device | 0.021 |
| home | 0.008 | smartthings | 0.016 | smart | 0.014 |
| smart | 0.007 | home | 0.010 | home | 0.013 |
| time | 0.006 | product | 0.009 | new | 0.007 |
| smartthings | 0.006 | light | 0.009 | easy | 0.007 |
| lock | 0.005 | time | 0.008 | smartthings | 0.007 |
| sensor | 0.005 | smart | 0.008 | issue | 0.006 |
| product | 0.005 | support | 0.007 | automation | 0.005 |
| device | 0.005 | zwave | 0.007 | control | 0.005 |
| back | 0.004 | great | 0.007 | phone | 0.005 |

TABLE VII: Prominent Words from LDA Topics of 1-Star App Reviews

| System | Words | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| Amazon Alexa | hate | device | time | update | useless |
| ecobee | update | thermostat | time | internet | connection |
| Google Home | music | chromecast | time | device | update |
| Insteon | device | time | find | waste | version |
| Kevo | lock | update | door | phone | time |
| Nest | camera | thermostat | update | home | time |
| Philips Hue | light | update | bridge | time | connection |
| SmartThings | phone | permission | uninstall | access | connect |
| Tile | phone | time | find | battery | key |
| WeMo | device | time | product | switch | update |

### *B. Issues in low-rated systems*

We filtered the app reviews so only reviews that had a minimal 1-star rating were left in the text. The goal behind running the LDA on only the 1-star reviews was to see if it was possible to identify the aspects of the app and devices that were leaving users with a negative impression. As such, we did not focus on words dealing with sentiment or emotion. Instead, we looked at words related to the functionality and features of the apps and devices. Table VII shows some of the noteworthy words that appeared in the topics for each app and Table VIII shows the same for device when LDA looked only at the 1-star reviews. These are words that had high magnitude values or that appeared in multiple topics.

Going over all the topics, a handful of relevant words seemed to appear with a greater frequency than others in the apps. For example, for all apps except for Kevo, at least one topic contained either the word "connect" or "connection". The prevalence of these words suggests that users of these apps have experienced some issue with connecting their phone to another device or network. The frequency in which "connect" and "connection" appears can mean that these connection issues are perhaps a greater source of frustration for users of IoT apps in general. Another noteworthy word was update. This word appeared in topics for all apps except for Insteon for Hub and Tile. It is important to note that the context for this word may not be the same in every appearance in the tables. For example, it is possible that some topics use "update", because an update was the source of a problem. It is also possible that the word appears in the context of users requesting an update to fix a problem with the app. However, the prevalence of the word does indicate that updates are an important part of app development and care should be taken in determining how they are implemented.

Home was another common word that appeared for six apps. With Google Home, this is not all that surprising, since home is part of the app's name. As for the other apps, the frequency of the word might suggest that many of these apps are indeed utilized for personal, home use. Making sure that these apps remain suited to this kind of use is another important thing for developers to keep in mind.

The word that appeared with the greatest frequency, however, was "time". This word appeared for all 10 apps; for most of the apps. With the exceptions of ecobee and Insteon, "time" actually appeared in at least two of the three topics for every app. Similar to "update", "time" does not necessarily have a single meaning in every one of its appearances. For apps like Philips Hue, the word appears to refer to the user's ability to configure through the app the time in which their light bulbs are set to turn on, turn off, change color, and so on. In these cases, the word "time" seems to relate more to scheduling functions of the app. In other cases, such as with Amazon Alexa, "time" appears in conjunction with words like "slow". Here, "time" is used to refer more to the duration of a function. The word appears in at least one of these contexts for every app. The prevalence of the word suggests that issues involving time are also an important element of these low-rated reviews. Resolving issues involved with timing settings as well as working to reduce the duration of app functions appear to both be issues app developers may want to pay attention to.

**Observation 2:** Issues with *connectivity*, *timing* and *update* are prevalent in the reviews of apps.

In the 1-star rated device reviews, in addition to timing and connectivity issues were also prominent. Anothe topic that seems to frustrate users in half of the devices is "support". Closer investigation of term "support" in the reviews revealed that in Insteon this term is largely referring to the issue of discontiuation of support of certain device, e.g., "INSTEON has stopped supporting their first cameras". In a fast-paced market such as IoT, abandaning of product might happen, but it is far from ideal. It indicates that the design of systems does not afford an efficient maintanence of the systems. Unsupported devices also known as zombie devices pose serious security, privacy and safety threats to the users [7] In ecobee, Google Home, Nest, , and Philips Hue, the term "support" was mostly refering to the customer support.

**Observation 3:** Issues with *connectivity*, *timing*, and *support* are prevalent in the reviews of the device.

## V. Discussion

The intent behind running topic modeling on the app and device reviews was to help identify those functions and features of the IoT system that appeared to be the most important to its users. After seeing the greater distribution of 1-star reviews in the apps compared to the devices, we were interested in discovering whether the LDA results would in particular help

TABLE VIII: Prominent Words from LDA Topics of 1-Star Device Reviews

| System | Words | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| Amazon Alexa | time | device | star | music | sound |
| ecobee | thermostat | support | product | system | temperature |
| Google Home | wifi | device | router | product | support |
| Insteon | support | device | customer | sensor | year |
| Kevo | lock | door | phone | time | product |
| Nest | thermostat | support | product | time | heat |
| Philips Hue | bulb | light | bridge | support | turn |
| SmartThings | product | device | home | time | new |
| Tile | phone | battery | key | time | product |
| WeMo | device | switch | connect | smart | time |

identify the characteristics of the apps that were causing users to leave negative reviews. The topics generated by the LDA from the full review texts provided fairly general information. Negative words appeared to be more common in the app review topics than in the device review topics, for example.

Running LDA on the 1-star app reviews only seemed to produce slightly more tangible results. Words like "time", "update", and "connect" were particularly frequent among these topics. Each of these words is related to different aspects of an app's functionality that can be a focus for developers. Though it is likely that the process can be refined further to be more effective, the results suggest that topic modeling approaches such as LDA can be used to help identify issues users may be dealing with when using the app.

The three prominent issues of timing, connectivity, and update shed lights on some facets of IoT systems that are rarely encountered in developing mainstream software systems. Powerful processors, abundant memory, optimizing compilers have largely resolved the problem of timing and efficiency in the development of software. However, in systems that work on limited processing power and memory such as IoT devices and the mobile systems, efficiency has become an issue.

Moreover, fast, reliable networks with negligible latency is a given in the development of traditional software systems. It has been achieved by development of technologies and tools that reduce the latency of network connections; for example, nowadays, almost all cloud service providers automatically move the running instances of applications to data centers closer clients. It seems that we need new technologies to address this problem for IoT systems.

The problem of automatic update and backward compatibility in traditional software systems have been under investigation for many years. Nowadays, thanks to standardization of operating systems and protocols there are frameworks that strive to (almost) seamless update of software. For example, Android, Windows, and MacOS allow developers to update their applications using the corresponding app stores. However, update for IoT systems that a large portion of the hardware and protocols have not been standardized poses new challenges that require new tools and techniques.

Understanding issues and obstacles in operational IoT system allows us to devise techniques and tools to support effective development of these systems. We believe that analysis of user reveiews can contribute in better understanding of these systems by extracting first-hand experiences of users. We released the dataset and the source code of this study at https://github.com/atruelove/AppReviewAnalysis to replicate the study and to facilitate further analysis of the reviews.

## VI. Threats to Validity

There are the following main threats to the validity of this study. First, our analysis was small in scope, we only used relatively recent reviews of a small number of IoT systems in our study. We also included the reviews from the Google Play app store not other app stores. Although small in scope, we believe that this study will provide the first glimpse of the users' issues in IoT systems. Second, we used LDA for topic modeling. It is known that LDA suffers from some limitations such as order effect [2]. To address these limitations, for given proposed words as topics, we manually checked the words to understand the intended meaning in the reviews and make sense of them.

## VII. Conclusion

In this paper, we analyzed the reviews of ten IoT devices from Amazon and the corresponding apps from the Google Play Store. To the best of our knowledge, it is the first analysis of such systems. Our results suggest that (1) there are more negative topics in the mobile apps than the devices, and (2) efficiency, connectivity, and update seem to be prevalent issues in such systems. Our results call for the development of new tools and techniques to support practitioners to address these issues. We released the dataset and the source code of this study at https://github.com/atruelove/AppReviewAnalysis to facilitate further analysis of the reviews.

## References


[1] gensim: topic modelling for humans. https://radimrehurek.com/gensim/, 9 2018.

[2] Agrawal, A., Fu, W., and Menzies, T. What is wrong with topic modeling? and how to fix it using search-based software engineering. *Information & Software Technology 98* (2018), 74–88.

[3] Alur, R., Berger, E., Drobnis, A. W., Fix, L., Fu, K., Hager, G. D., Lopresti, D., Nahrstedt, K., Mynatt, E., Patel, S., et al. Systems computing challenges in the internet of things. *Computing Community Consortium (CCC) Technical Report* (2016).

[4] Atzori, L., Iera, A., and Morabito, G. The internet of things: A survey. *Computer Networks 54*, 15 (2010), 2787 – 2805.

[5] Chen, N., Lin, J., Hoi, S. C. H., Xiao, X., and Zhang, B. AR-miner: mining informative reviews for developers from mobile app marketplace. In *Proceedings of the 36th International Conference on Software Engineering - ICSE 2014* (New York, New York, USA, 2014), ACM Press, pp. 767–778.

[6] Di Sorbo, A., Panichella, S., Alexandru, C. V., Shimagaki, J., Visaggio, C. A., Canfora, G., and Gall, H. C. What would users change in my app? summarizing app reviews for recommending software changes. In *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering - FSE 2016* (New York, New York, USA, 2016), ACM Press, pp. 499–510.

[7] Fu, K., Kohno, T., Lopresti, D., Mynatt, E., Nahrstedt, K., Patel, S., Richardson, D., and Zorn, B. Safety, security, and privacy threats posed by accelerating trends in the internet of things. *Computing Community Consortium (CCC) Technical Report 29*, 3 (2017).

[8] FUJINO, I. Refining lda results and ranking topics in order of quantity and quality with an application to twitter streaming data. In *2014 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)* (2014), IEEE, pp. 209–216.

[9] GU, X., AND KIM, S. " what parts of your apps are loved by users?"(t). In *Automated Software Engineering (ASE), 2015 30th IEEE/ACM International Conference on* (2015), IEEE, pp. 760–770.

[10] HERMANSON, D. New directions: Exploring Google Play mobile app user feedback in terms of perceived ease of use and perceived usefulness.

[11] HOON, L., VASA, R., SCHNEIDER, J.-G., AND GRUNDY, J. An analysis of the mobile app review landscape: trends and implications. *Technical report, Swinburne University of Technology* (2013), 1–23.

[12] KAAZ, K. J., HOFFER, A., SAEIDI, M., SARMA, A., AND BOBBA, R. B. Understanding user perceptions of privacy, and configuration challenges in home automation. In *Visual Languages and Human-Centric Computing (VL/HCC), 2017 IEEE Symposium on* (2017), IEEE, pp. 297–301.

[13] LICORISH, S. A., SAVARIMUTHU, B. T. R., AND KEERTIPATI, S. Attributes that predict which features to fix: Lessons for app store mining. In *Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering* (2017), ACM, pp. 108–117.

[14] MAALEJ, W., AND NABIL, H. Bug report, feature request, or simply praise? On automatically classifying app reviews. In *2015 IEEE 23rd International Requirements Engineering Conference, RE 2015 - Proceedings* (aug 2015), IEEE, pp. 116–125.

[15] MUJAHID, S., SIERRA, G., ABDALKAREEM, R., SHIHAB, E., AND SHANG, W. Examining User Complaints of Wearable Apps: A Case Study on Android Wear. *Proceedings - 2017 IEEE/ACM 4th International Conference on Mobile Software Engineering and Systems, MOBILESoft 2017*, August (2017).

[16] PAGANO, D., AND MAALEJ, W. User Feedback in the AppStore: An Empirical Study (submitted). *RE '13: Proceedings of the 21st International Requirements Engineering Conference* (2013), 125–134.

[17] SEYFF, N., OLLMANN, G., AND BORTENSCHLAGER, M. AppEcho: A User-Driven, In Situ Feedback Approach for Mobile Platforms and Applications. In *Proceedings of the 1st International Conference on Mobile Software Engineering and Systems - MOBILESoft 2014* (New York, New York, USA, 2014), ACM Press, pp. 99–108.

[18] VILLARROEL, L., BAVOTA, G., RUSSO, B., OLIVETO, R., AND DI PENTA, M. Release planning of mobile apps based on user reviews. *Proceedings of the 38th International Conference on Software Engineering - ICSE '16* (2016), 14–24.

## APPENDIX

### A. Results of topic modeling for IoT apps and devices

TABLE XIV: Philips Hue App

| Topic 1 Words | Topic 1 Magnitude | Topic 2 Words | Topic 2 Magnitude | Topic 3 Words | Topic 3 Magnitude |
|---|---|---|---|---|---|
| ”light” | 0.041 | ”version” | 0.011 | ”update” | 0.031 |
| ”room” | 0.016 | ”light” | 0.010 | ”bridge” | 0.028 |
| ”great” | 0.014 | ”color” | 0.007 | ”light” | 0.023 |
| ”update” | 0.013 | ”location” | 0.007 | ”home” | 0.018 |
| ”scene” | 0.012 | ”scene” | 0.007 | ”connect” | 0.014 |
| ”time” | 0.011 | ”routine” | 0.007 | ”time” | 0.008 |
| ”bulb” | 0.010 | ”good” | 0.007 | ”control” | 0.008 |
| ”new” | 0.010 | ”gen” | 0.006 | ”connection” | 0.008 |
| ”turn” | 0.010 | ”easy” | 0.006 | ”new” | 0.007 |
| ”feature” | 0.009 | ”feature” | 0.006 | ”find” | 0.006 |

TABLE XV: Tile App

| Topic 1 Words | Topic 1 Magnitude | Topic 2 Words | Topic 2 Magnitude | Topic 3 Words | Topic 3 Magnitude |
|---|---|---|---|---|---|
| ”great” | 0.021 | ”love” | 0.020 | ”phone” | 0.039 |
| ”tile” | 0.018 | ”phone” | 0.018 | ”key” | 0.029 |
| ”find” | 0.016 | ”found” | 0.010 | ”find” | 0.020 |
| ”battery” | 0.016 | ”key” | 0.009 | ”time” | 0.017 |
| ”time” | 0.014 | ”easy” | 0.008 | ”tile” | 0.013 |
| ”phone” | 0.014 | ”find” | 0.008 | ”keep” | 0.010 |
| ”key” | 0.011 | ”location” | 0.008 | ”lost” | 0.009 |
| ”product” | 0.010 | ”great” | 0.008 | ”location” | 0.008 |
| ”never” | 0.009 | ”best” | 0.008 | ”bluetooth” | 0.008 |
| ”year” | 0.008 | ”good” | 0.007 | ”ring” | 0.008 |

TABLE XVI: WeMo App

| Topic 1 Words | Topic 1 Magnitude | Topic 2 Words | Topic 2 Magnitude | Topic 3 Words | Topic 3 Magnitude |
|---|---|---|---|---|---|
| ”switch” | 0.025 | ”great” | 0.037 | ”device” | 0.035 |
| ”light” | 0.024 | ”easy” | 0.033 | ”time” | 0.023 |
| ”update” | 0.021 | ”time” | 0.027 | ”wifi” | 0.019 |
| ”home” | 0.021 | ”device” | 0.018 | ”connect” | 0.019 |
| ”turn” | 0.017 | ”setup” | 0.017 | ”switch” | 0.013 |
| ”device” | 0.015 | ”good” | 0.015 | ”setup” | 0.012 |
| ”smart” | 0.012 | ”love” | 0.013 | ”rule” | 0.011 |
| ”plug” | 0.011 | ”buggy” | 0.012 | ”network” | 0.011 |
| ”firmware” | 0.011 | ”slow” | 0.009 | ”plug” | 0.011 |
| ”product” | 0.011 | ”product” | 0.008 | ”reset” | 0.009 |

TABLE XVII: ecobee4 Smart Thermostat

| Topic 1 Words | Topic 1 Magnitude | Topic 2 Words | Topic 2 Magnitude | Topic 3 Words | Topic 3 Magnitude |
|---|---|---|---|---|---|
| ”thermostat” | 0.017 | ”thermostat” | 0.019 | ”thermostat” | 0.022 |
| ”sensor” | 0.015 | ”temperature” | 0.011 | ”easy” | 0.012 |
| ”room” | 0.011 | ”home” | 0.008 | ”great” | 0.011 |
| ”device” | 0.009 | ”house” | 0.008 | ”support” | 0.010 |
| ”great” | 0.008 | ”feature” | 0.007 | ”install” | 0.008 |
| ”home” | 0.008 | ”smart” | 0.007 | ”product” | 0.008 |
| ”support” | 0.007 | ”love” | 0.007 | ”sensor” | 0.007 |
| ”temperature” | 0.006 | ”sensor” | 0.006 | ”system” | 0.006 |
| ”smart” | 0.006 | ”time” | 0.006 | ”wire” | 0.006 |
| ”house” | 0.006 | ”room” | 0.006 | ”house” | 0.006 |

TABLE XVIII: Google WiFi System

| Topic 1 Words | Topic 1 Magnitude | Topic 2 Words | Topic 2 Magnitude | Topic 3 Words | Topic 3 Magnitude |
|---|---|---|---|---|---|
| ”wifi” | 0.041 | ”router” | 0.022 | ”wifi” | 0.026 |
| ”house” | 0.025 | ”easy” | 0.019 | ”router” | 0.018 |
| ”system” | 0.014 | ”device” | 0.016 | ”device” | 0.017 |
| ”easy” | 0.014 | ”wifi” | 0.014 | ”network” | 0.016 |
| ”router” | 0.014 | ”setup” | 0.011 | ”mesh” | 0.007 |
| ”great” | 0.013 | ”great” | 0.011 | ”point” | 0.007 |
| ”home” | 0.013 | ”speed” | 0.010 | ”house” | 0.007 |
| ”signal” | 0.013 | ”love” | 0.008 | ”setup” | 0.007 |
| ”speed” | 0.012 | ”house” | 0.008 | ”time” | 0.006 |
| ”product” | 0.009 | ”network” | 0.007 | ”speed” | 0.006 |

TABLE XIX: Insteon Hub Device

| Topic 1 Words | Topic 1 Magnitude | Topic 2 Words | Topic 2 Magnitude | Topic 3 Words | Topic 3 Magnitude |
|---|---|---|---|---|---|
| ”switch” | 0.015 | ”device” | 0.016 | ”device” | 0.024 |
| ”control” | 0.008 | ”year” | 0.014 | ”new” | 0.016 |
| ”product” | 0.007 | ”product” | 0.011 | ”scene” | 0.012 |
| ”purchase” | 0.007 | ”support” | 0.010 | ”sensor” | 0.011 |
| ”buy” | 0.006 | ”new” | 0.009 | ”light” | 0.011 |
| ”de” | 0.006 | ”account” | 0.009 | ”switch” | 0.008 |
| ”house” | 0.005 | ”month” | 0.008 | ”old” | 0.008 |
| ”device” | 0.005 | ”died” | 0.007 | ”year” | 0.008 |
| ”return” | 0.005 | ”control” | 0.007 | ”home” | 0.007 |
| ”time” | 0.005 | ”bought” | 0.006 | ”product” | 0.007 |

TABLE XX: Kevo Lock (2nd Gen)

| Topic 1 Words | Topic 1 Magnitude | Topic 2 Words | Topic 2 Magnitude | Topic 3 Words | Topic 3 Magnitude |
|---|---|---|---|---|---|
| ”lock” | 0.029 | ”lock” | 0.032 | ”lock” | 0.046 |
| ”door” | 0.014 | ”phone” | 0.018 | ”time” | 0.018 |
| ”key” | 0.013 | ”door” | 0.017 | ”door” | 0.012 |
| ”phone” | 0.009 | ”key” | 0.011 | ”key” | 0.012 |
| ”time” | 0.008 | ”time” | 0.009 | ”battery” | 0.011 |
| ”product” | 0.008 | ”unlock” | 0.008 | ”phone” | 0.009 |
| ”open” | 0.007 | ”open” | 0.007 | ”open” | 0.007 |
| ”touch” | 0.005 | ”plus” | 0.007 | ”unlock” | 0.006 |
| ”kwikset” | 0.005 | ”great” | 0.006 | ”great” | 0.005 |
| ”bluetooth” | 0.004 | ”bluetooth” | 0.006 | ”fob” | 0.005 |

TABLE XXI: Nest T3007ES Thermostat

| Topic 1 Words | Topic 1 Magnitude | Topic 2 Words | Topic 2 Magnitude | Topic 3 Words | Topic 3 Magnitude |
|---|---|---|---|---|---|
| ”easy” | 0.027 | ”thermostat” | 0.023 | ”thermostat” | 0.020 |
| ”thermostat” | 0.026 | ”wire” | 0.018 | ”time” | 0.009 |
| ”great” | 0.019 | ”unit” | 0.011 | ”house” | 0.009 |
| ”love” | 0.018 | ”system” | 0.010 | ”home” | 0.009 |
| ”install” | 0.017 | ”installed” | 0.009 | ”day” | 0.008 |
| ”home” | 0.015 | ”old” | 0.007 | ”support” | 0.008 |
| ”temperature” | 0.013 | ”hvac” | 0.007 | ”product” | 0.007 |
| ”control” | 0.009 | ”support” | 0.006 | ”heat” | 0.007 |
| ”product” | 0.008 | ”new” | 0.006 | ”ac” | 0.007 |
| ”house” | 0.008 | ”install” | 0.005 | ”degree” | 0.006 |

TABLE XXII: Philips Hue Starter Kit

| Topic 1 Words | Topic 1 Magnitude | Topic 2 Words | Topic 2 Magnitude | Topic 3 Words | Topic 3 Magnitude |
|---|---|---|---|---|---|
| ”product” | 0.017 | ”light” | 0.040 | ”light” | 0.044 |
| ”bulb” | 0.016 | ”home” | 0.020 | ”bulb” | 0.034 |
| ”great” | 0.016 | ”bulb” | 0.018 | ”great” | 0.018 |
| ”light” | 0.011 | ”control” | 0.012 | ”love” | 0.017 |
| ”easy” | 0.009 | ”switch” | 0.011 | ”turn” | 0.016 |
| ”router” | 0.008 | ”easy” | 0.009 | ”easy” | 0.014 |
| ”home” | 0.007 | ”great” | 0.009 | ”home” | 0.013 |
| ”bridge” | 0.007 | ”smart” | 0.009 | ”setup” | 0.008 |
| ”turn” | 0.007 | ”turn” | 0.009 | ”smart” | 0.008 |
| ”good” | 0.007 | ”kit” | 0.007 | ”room” | 0.006 |

TABLE XXIII: Tile Mate

| Topic 1 Words | Topic 1 Magnitude | Topic 2 Words | Topic 2 Magnitude | Topic 3 Words | Topic 3 Magnitude |
|---|---|---|---|---|---|
| ”battery” | 0.034 | ”phone” | 0.056 | ”key” | 0.064 |
| ”year” | 0.021 | ”great” | 0.034 | ”time” | 0.028 |
| ”tile” | 0.013 | ”find” | 0.034 | ”find” | 0.020 |
| ”new” | 0.011 | ”key” | 0.026 | ”phone” | 0.018 |
| ”month” | 0.010 | ”love” | 0.021 | ”lost” | 0.016 |
| ”product” | 0.010 | ”time” | 0.012 | ”wallet” | 0.014 |
| ”replace” | 0.007 | ”bluetooth” | 0.011 | ”easy” | 0.013 |
| ”location” | 0.007 | ”product” | 0.011 | ”lose” | 0.009 |
| ”time” | 0.006 | ”bought” | 0.011 | ”bought” | 0.008 |
| ”buy” | 0.006 | ”gift” | 0.010 | ”month” | 0.008 |

# Proposal of a New software architecture for interoperability to improve the communication in the Edge layer of a smart IoT ecosystem

1st Juan Moreno-Motta
*School of Informatics and System Engineering*
*Universidad Nacional Mayor de San Marcos*
Lima, Perú
juan.moreno2@unmsm.edu.pe

2nd Felipe Moreno-Vera
*Deparment of Computer Science*
*Universidad Católica San Pablo*
Arequipa, Perú
felipe.moreno@ucsp.edu.pe

3rd Frank Moreno-Vera
*School of Electronic Engineering*
*Universidad Nacional de Ingeniera*
Lima, Perú
fmorenov@uni.pe

***Abstract*—In the current years, IoT has evolved to such an extent to extend to all corners of each place through devices, which connected to a network, either local or internet itself, generate information to be processed with a specific purpose, to this level there is a problem called interoperability of devices where not only is the compatibility of adding or removing devices to an ecosystem and there is compatibility, it is also expected that the information generated is standardized and optimized to transmit. This paper presents a new software architecture pattern for interoperability between devices that generate heterogeneous information in the edge layer of an IoT ecosystem.**

***Index Terms*—Interoperability, IoT, data encode, data decode, protocol buffer, Edge Computing, REST API, Software Architecture, IoT Ecosystem.**

## I. Introduction

### A. Background

The Internet of Things (from now until the end IoT) is a technology that has been emerging and converging of many technologies generating new paradigms to implement these architectures.

In 2014, Oliver Kleine [1] estimated that IoT devices will be increased at least 1.7 billions. In the same year, Utkarshani Jaimini [2] said the storage for those devices surpass 150 Exabytes in 2017.

According to Guinard et al [3], IoT is a system of physical objects that can be discovered, controlled or interacted with electronic devices that communicate through various network interfaces and can finally be connected to the Internet.

### B. Present context

In 2017, Talavera et al. [4] study all IoT solution in topics like Agribusiness and Environment implementations, They found around 72 projects those represents all developed before, with this study they propose a general architecture for this solutions but they consider as challenge the way to implement a standard, compatibility and security guaranty between devices in the edges and cloud services.

Also, in 2017 Woznowski et al. [5] develop SPHERE (A Sensor Platform for Healthcare in a Residential Environment), in this project they have determined that there are 9 requirements that the IoT needs to cover to implement a smart city, but the most difficult requeriment to garanty and implement was the Interoperability.

Elsts et al. [6] based on SPHERE, consider that systems must comply with existing low-power IoT standards and protocols to (1) be susceptible to future extensions with third-party components; (2) reduce learning time for new staff.

Alkhalil et al. [7] mentions that the challenges related to the origin of the data, despite the current techniques, remains very challenging in its implementation and optimization. Therefore, this challenge of data origin, of the 7 main ones, is directly linked to interoperability, making it more difficult to deploy heterogeneous systems.

In two research about this, Pace et al. [8] and Madaan et al. [9] said that is very complicate to integrate all information generated by all systems with different devices, brands, hardware design, protocols, body message encode-decode, different programming languages, different data structs, etc.

### C. Problematic and proposes to solve it

After of analysis all previous works above, we note the problematic was centering in how to implement a secure system that manage interoperability without complications.

Yacchirema et al. [10] present a platform oriented to Ambient Assisted Living (AAL), that platform called AAL-IoTSys is a prototype based on Wireless Sensor Network (WSN) with heterogeneous devices using Binary encoder to transfer information between low power devices.

Androec et al. [11] propose web semantic to allow IoT interoperability using JSON-LD. with this method they can create a description about data information and link objects and properties in a JSON file.

Sun et al. [12] proposes a REST API for Web of Things which work with JSON and compare Micro-services architecture against Monolithic architecture. They use REST API to communicate all devices including the IoT ecosystem, the

control of the environment through the central service is dynamic.

Lim et al. [13] and Malik et al. [14] propose and compare architectures based on SOAP (XML) and REST API (JSON) services applied to try to improve scalability of interoperability, but they do not have success.

Kum et al. [15] based on previous works (some of them was mentioned above) they propose an architecture for Fog Computing applications, in this case, Fog Computing allows to manage better the information between edge layer and cloud services trough middle servers called ”Fog servers” (See Fig.1).

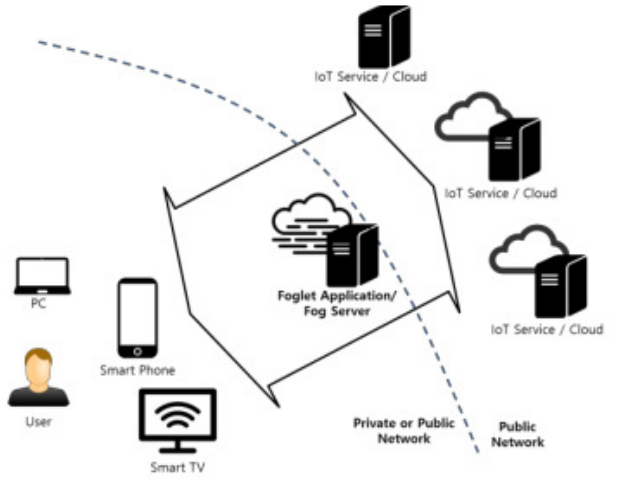



Fig. 1. kum et al. Architecture [15].

Luan et al. [16] propose to create a routing gateway to manage all information and communication between devices (edges), fog nodes and cloud servers. They work with a network topology which improve and reduce time to transfer data. The aim was achieve to devices to communicate all servers and nodes was mobiles, the architecture transmits data trough mobiles (5G infrastructure) as end users or edge layer (See Fig.2).

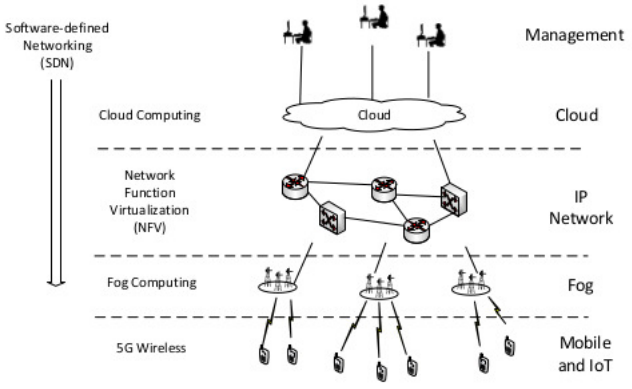



Fig. 2. luan et al. Architecture [16].

Lysogor et al. [17] conduct a study on the transfer and exchange of data in heterogeneous networks where there is an absence of network infrastructure and their research focuses on the use of satellite networks, however, there is a great limitation on the size of the data transmitted. They show that the binary format generated by Protocol Buffer allows more bytes to be transferred than the JSON format (See Fig.3).

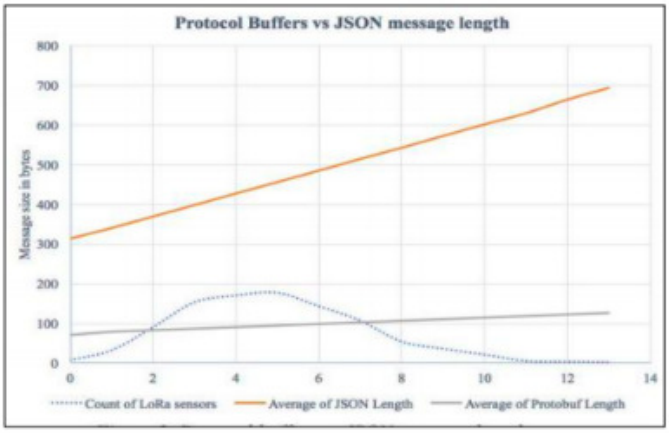



Fig. 3. lysogor et al. performance chart [17].

Nitin Naik [18] write a complete document about protocols, applications, operating systems, and other metric for different situations in IoT environments. In this document they present HTTP, AMQP, MQTT and CoAP protocols (commonly used in IoT solutions) (See Fig.4). According authors, the user can decide their relevant usage in IoT systems based on their requirements and suitability.

| Criteria | MQTT | CoAP | AMQP | HTTP |
|---|---|---|---|---|
| 1. Year | 1999 | 2010 | 2003 | 1997 |
| 2. Architecture | Client/Broker | Client/Server or Client/Broker | Client/Broker or Client/Server | Client/Server |
| 3. Abstraction | Publish/Subscribe | Request/Response or Publish/Subscribe | Publish/Subscribe or Request/Response | Request/Response |
| 4. Header Size | 2 Byte | 4 Byte | 8 Byte | Undefined |
| 5. Message Size | Small and Undefined (up to 256 MB maximum size) | Small and Undefined (normally small to fit in single IP datagram) | Negotiable and Undefined | Large and Undefined (depends on the web server or the programming technology) |
| 6. Semantics/ Methods | Connect, Disconnect, Publish, Subscribe, Unsubscribe, Close | Get, Post, Put, Delete | Consume, Deliver, Publish, Get, Select, Ack, Delete, Nack, Recover, Reject, Open, Close | Get, Post, Head, Put, Patch, Options, Connect, Delete |
| 7. Cache and Proxy Support | Partial | Yes | Yes | Yes |
| 8. Quality of Service (QoS)/ Reliability | QoS 0 - At most once (Fire-and-Forget), QoS 1 - At least once, QoS 2 - Exactly once | Confirmable Message (similar to At most once) or Non-confirmable Message (similar to At least once) | Settle Format (similar to At most once) or Unsettle Format (similar to At least once) | Limited (via Transport Protocol - TCP) |
| 9. Standards | OASIS, Eclipse Foundations | IETF, Eclipse Foundation | OASIS, ISO/IEC | IETF and W3C |
| 10. Transport Protocol | TCP (MQTT-SN can use UDP) | UDP, SCTP | TCP, SCTP | TCP |
| 11. Security | TLS/SSL | DTLS, IPSec | TLS/SSL, IPSec, SASL | TLS/SSL |
| 12. Default Port | 1883/ 8883 (TLS/SSL) | 5683 (UDP Port)/ 5684 (DLTS) | 5671 (TLS/SSL), 5672 | 80/ 443 (TLS/SSL) |
| 13. Encoding Format | Binary | Binary | Binary | Text |

Fig. 4. Nitin comparison table [18].

Petersen et al. [19] make a demonstration of the performance of the different formats, arriving at the conclusion that the binary format generated with Protobuff developed by Google is the one of better performance and less memory use (See Fig.5), It is observed that Protocol Buffer for any

communication protocol can serialize many more messages per second, being one of the protocols with better ZeroMQ performance.

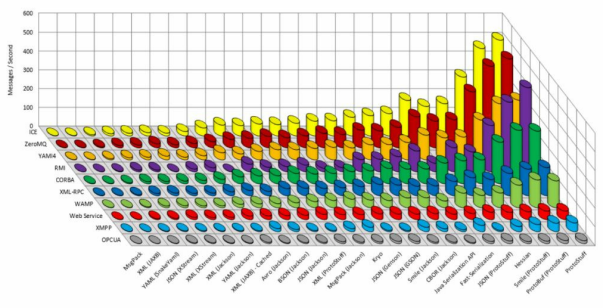


Fig. 5. petersen et al. performance chart [19].

## II. Proposal

After to identify all applications and architectures proposed to solve the challenge of devices interoperability, we propose our architecture to manage in a better way this problem, using Fog Computing to manage IoT solutions as components, we define an Ecosystem as the process to communicate data between devices or different IoT solutions or different Frameworks (See Fig.6) based on encode-decode data serialization.

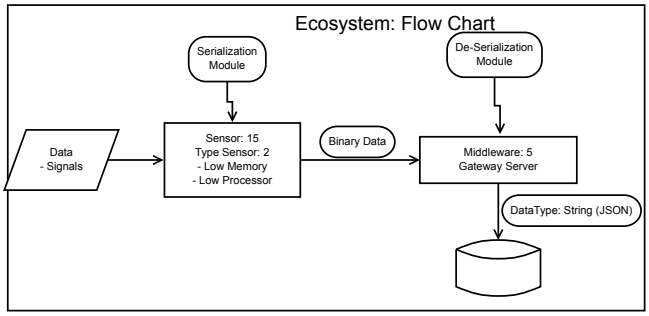



Fig. 6. Ecosystem design.

We define Ecosystems like a process to encode-decode data, based on Micro-services architecture. The data exchange between devices is very important when you need to get or analyses by sectors and that the reason to divide all interaction for a better management of the complete system.

Then, to manage all communication and interaction between devices we define our Fog Ecosystem (Ecosystem - Fog Computing) subdivide in 7 important components (See Fig.7):

- Services: This component is the service layer that Fog Computing can deliver so that other external devices can receive information (from sensors) or send an action to be executed (actuators).
- Repository: This component is of persistence where de-serialized data frames will be stored in JSON format.
- Processing: This component is responsible for processing the binary data formats that are received from the component ”Device”, previously identifies the type (category to identify encode-decode algorithm, origin, etc) of device and the device that comes as data within the frame.
- Management interface: This component provides an administration interface. This administrator registers the types of devices and devices, as well as the Middleware and the ecosystem to which one or more Middleware belongs.
- Security: This component is used by the Middleware to validate that the devices that are connecting have the required authorization and can thus receive the data frames in binary format that is sent by the devices.
- Middle-Ware: This component is responsible for receiving the binary format, makes use of the ”Processing” component that has greater processing capacity since its operation is that of an exclusive processing server. It also makes use of the File System component as an ecosystem configuration file.
- File System: This component is the Fog Ecosystem logger. Also, storage all data decoded.

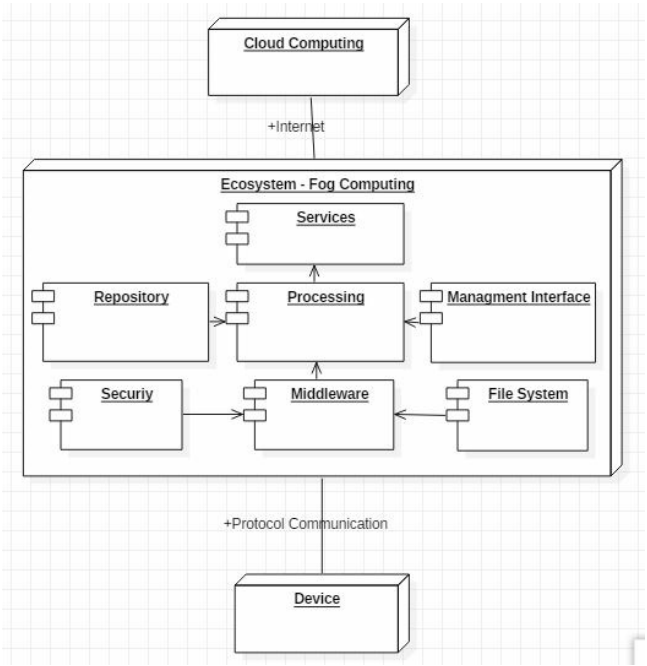



Fig. 7. Architecture proposed.

Then we have 2 more components: devices and Cloud Computing with protocols like internet (for Cloud services) and Protocol buffer to interact with devices.

- First: ”devices component” represent all devices which will be connect to Ecosystem trought Middle-Ware component, in this part is implemented encode-decode data (based on protocol buffer), in general could be sensors or any devices which generate data.
- Second: ”Cloud Computing” represent the final repository of data analysis of all information collected and pre-processed from Fog Ecosystem.

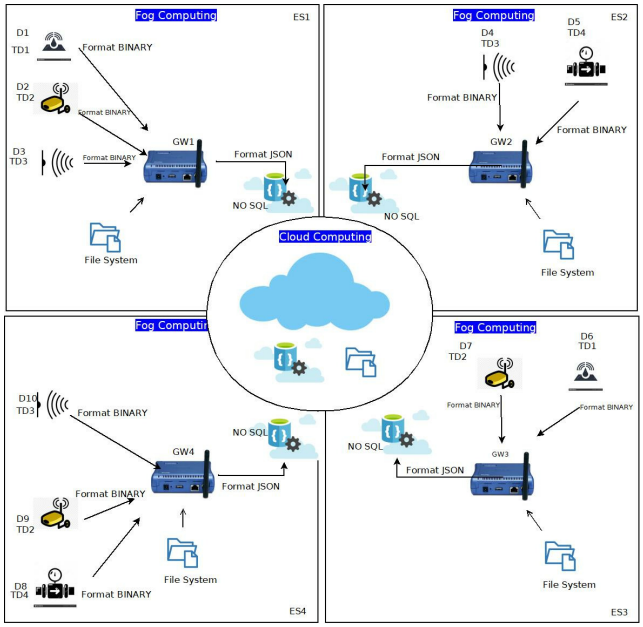



Fig. 8. Architecture proposed for multiple solutions.

## III. Conclusions

In this paper, we propose a new architecture for interoperability using Fog and Edge computing to manage and improve smartly communication and transfer information between devices at edge-fog level in the IoT ecosystem that can be integrated with other ecosystems achieving scalability and identifying the origin of the data, keeping them all ordered and communicated among themselves.

To achieve this, we use Protocol buffers with our data standardized format for all devices (or at least most of them) to get a better performance and get less latency transferring data and a high flexible fog architecture to manage all dynamic changes in devices.

## Future work

We are implementing this architecture in real time applications to get our proof of concept of our proposed architecture, our test is a real time system monitors of data measure from environment, we use Temperature, Humidity, Sound and Monoxide sensors, Raspberry Pi 3 Model B (1.4GH, Quad Core) board, Arduino board, SQLite, protocol buffer to encode-decode data and Cloud services like Firebase to storage data and analyze, compare, measure latency, compatibility and processing or organizing data speed between each layer of the ecosystem (See Fig.8).

## Acknowledgment

This work was supported by grant 234-2015-FONDECYT (Master Program) from CienciActiva of the National Council for Science,Technology and Technological Innovation (CONCYTEC-PERU). Also, thanks to SCOTIABANK MASTER PROGRAM GRANT for the financial support to Juan Moreno.

## Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this article.

## References

[1] Oliver Kleine, CoAP Endpoint Identification - A Protocol Extension for Crowd Sensing in the mobile Internet, 2014 IEEE International Conference on Internet of Things (iThings 2014), Green Computing and Communications (GreenCom 2014), and Cyber-Physical-Social Computing (CPSCom 2014).

[2] Jaimini, U. (2017). PhD Forum: Multimodal IoT and EMR based Smart Health Application for Asthma Management in Children.

[3] Guinard, D. D. and Trifa, V. M. (2016). Building the Web of Things. NY 11964: MANNING.

[4] Talavera, J. M., Tobn, L. E., Gmez, J. A., Alejandro, M. A., Aranda, J. M., Parra, D. T., Garreta, L. E. (2017). Review of IoT applications in agro-industrial and environmental fields.

[5] Przemyslaw Woznowski, Alison Burrows, Tom Diethe, Xenofon Fafoutis, Jake Hall, Sion Hannuna, Massimo Camplani, Niall Twomey, Michal Kozlowski, Bo Tan, Ni Zhu, Atis Elsts, Antonis Vafeas, Adeline Paiement, Lili Tao, Majid Mirmehdi, Tilo Burghardt, Dima Damen, Peter Flach, Robert Piechocki, Ian Craddock, George Oikonomou, SPHERE: A Sensor Platform for Healthcare in a Residential Environment. https://link.springer.com/chapter/10.1007/978-3-319-44924-1_14. Last visited dec 2018.

[6] Elsts, A., Oikonomou, G., Fafoutis, X. and Piechocki, R. (2017). Internet of Things for Smart Homes: Lessons Learned from the SPHERE Case Study.

[7] Alkhalil, A. and Ramadan, R. A. (2017). IoT Data Provenance Implementation Challenges.

[8] Pace, P., Gravina, R., Aloi, G., Fortino, G., Fides-Valero, A., Ibaez-Sanchez, G., Yacchirema, D. (2017). IoT platforms interoperability for Active and Assisted Living Healthcare services support.

[9] Madaan, N., Ahad, M. A. and Sastry, S. M. (2017). Data integration in IoT ecosystem: Information linkage as a privacy threat.

[10] Yacchirema, D. C., Palau, C. E. and Esteve, M. (2017). Enable IoT Interoperability in Ambient Assisted Living: Active and Healthy Aging Scenarios.

[11] Androec, D., Toma, ,. B. and Kiasondi, T. (2017). Interoperability and Lightweight Security for Simple IoT Devices.

[12] Sun, L., Li, Y. and Memon, R. A. (2017). An Open IoT Framework Based on Microservices Architecture.

[13] Lim, N., Majumdar, S. and Nandy, B. (2010). Providing Interoperability for Resource Access Using Web Services.

[14] Malik, S. and Kim, D.-H. (2017). A Comparison of RESTful vs. SOAP Web Services in Actuator Networks.

[15] Kum, S. W., Moon, J. and Lim, T.-B. (2017). Design of Fog Computing based IoT Application Architecture.

[16] Tom H. Luan, Longxiang Gao, Zhi Li, Yang Xiang, Guiyi We and Limin Sun (2016). Fog Computing: Focusing on Mobile Users at the Edge.

[17] Lysogor, I., Voskov, L. and Efremov, S. (2018). Survey of Data Exchange Formats for Heterogeneous LPWAN-Satellite IoT Networks.

[18] Nitin Naik. Choice of effective messaging protocols for iot systems: Mqtt,coap, amqp and http.In2017 IEEE International Systems EngineeringSymposium (ISSE), pages 17, 2017

[19] Petersen, B., Bindner, H., You, S. and Poulsen, B. (2017). Smart Grid Serialization Comparison.

# Applying Model Driven Engineering Techniques to the Development of Contiki-based IoT Systems

Tansu Zafer Asici
*International Computer Institute*
*Ege University*, Izmir, Turkey
tansu.asici@gmail.com

Burak Karaduman
*International Computer Institute*
*Ege University*, Izmir, Turkey
bburakkaraduman@gmail.com

Raheleh Eslampanah
*Electric and Electronics Engineering*
*Izmir University of Economics*, Izmir, Turkey
raheleh.eslampanah@ieu.edu.tr

Moharram Challenger
*Electronics and ICT Department*
*University of Antwerp*, Belgium
and *Flanders Make*, Belgium
moharram.challenger@uantwerpen.be

Joachim Denil
*Electronics and ICT Department*
*University of Antwerp*, Belgium
and *Flanders Make*, Belgium
joachim.denil@uantwerpen.be

Hans Vangheluwe
*Mathematics and Computer Science Dept.*
*University of Antwerp*, Belgium
and *Flanders Make*, Belgium
hans.vangheluwe@uantwerpen.be

***Abstract*—The huge variety of smart devices and their communication models increases the development complexity of embedded software for the Internet of Things. As a consequence, development of these systems becomes more complex, error-prone, and costly. To tackle this problem, in this study, a model-driven approach is proposed for the development of Contiki-based IoT systems. To this end, the available Contiki metamodel in the literature is extended to include the elements of WiFi connectivity modules (such as ESP8266), IoT Log Manager, and information processing components (such as Raspberry Pi). Based on this new metamodel, a domain-specific modeling environment is developed in which visual symbols are used and static semantics (representing system constraints) are defined. Also, the architectural code for the computing components of the IoT system such as Contiki, ESP8266, and RaspberryPi are generated according to the developer's instance model. Finally, a Smart Fire Detection system is used to evaluate this study. By modeling the Contiki-based IoT system, we support model-driven development of the system, including WSN motes and sink nodes (with ContikiOS), WiFi modules and information processing components.**

***Index Terms*—Model-driven Engineering (MDE), Internet of Things (IoT), Embedded Software, Wireless Sensor Network, ContikiOS, Smart Fire Detection System**

## I. Introduction

Internet of Things (IoT) systems are rapidly taking their place in different technologies and markets, such as home appliances, smart buildings, and Industry 4.0 applications. In these systems, devices communicate with each other and work in coordination, to create an intelligent environment. Generally, these devices can communicate in two ways. The first method is used by devices with Internet connection capability (e.g., WiFi connection, using the IEEE 802.11 protocol) where they communicate by connecting directly to the Internet. WiFi connection suffers from various problems such as limited access range to cover large areas (introducing the need for range extenders which results in extra costs and more setup complexity). The second method is used by low power devices using 802.15.4 protocol and point to point communication, such as Wireless Sensor Networks (WSN). This communication protocol can route the transmitting data through neighbor nodes (also called sensor motes). Thus, they are spreadable and can cover wide areas. These two communication protocols can be combined so new IoT systems can have both accessibility via Internet and have wide area coverage.

However, these communication protocols need different underlying technologies and therefore, different programming languages, which make the embedded software development complex, error-prone, and costly. Also, these systems usually require other, complementary parts. For example, a WSN based IoT system may require a gateway to transmit the data to the Internet using the IEEE 802.15.4 protocol. Furthermore, an information management sub-system may be needed. This information management sub-systems should process the information received from the WiFi modules and WSN devices. It can be implemented using a desktop application, a web application with access to a database, or even by a cloud based system. Moreover, this application needs to be written in a yet another programming language. Designing and developing such a system adds even more complexity.

To address this complexity, a Model-Driven Engineering (MDE) approach can be used to increase the level of abstraction and automatically synthesize some of the artifacts. For this purpose, in this study, a modeling environment is proposed for the development of Contiki [8] and ESP8266 (a low cost WiFi transceiver module)[1] based IoT systems which are using both WSN and WiFi communication. As MDE raises the abstraction level of the software development process, this approach can reduce the development cost and time as well as decrease the number of errors which in turn leads to a shorter testing time.

To apply the MDE approach, domain-specific modeling

*This study is partially funded by the Scientific Research Project No 17-UBE-002 at EGE University, Izmir-Turkey.*

[1]https://www.espressif.com/en/products/hardware/esp8266ex/overview

languages can be used [6]. Domain-specific modeling allows domain-experts to reason in their problem domain rather than in the solution domain. Domain-specific languages (DSL) are defined by specifying the concepts and relations between components within the domain by using a meta-model. The concrete syntax of the language (i.e. used in the language editor) can be textual and/or graphical. Finally, the semantics of the language can be defined by using a set of model to code/text transformation rules.

In our previous study [9], a meta-model was proposed for ContikiOS. The idea was purely about modeling the ContikiOS programs. In this study, this meta-model is extended, a new graphical editor editor is developed and the transformation rules are defined for code generation. The modeling environment is extended to support the other required IoT components and systems such as WiFi based modules/micro-controllers (i.e. ESP8266) and WSN gateway devices (such as a Java program on a Raspberry Pi[2]) as well as the data processing and repository such as a IoT Log Manager System. A fire detection system [1] is used as a case study to evaluate the proposed language.

This paper is organized as follows: Section II discusses the related work. The extended meta-model is introduced in Section III. The modeling environment, including the graphical concrete syntax and the static semantics (constraints), is described in Section IV . The Code generation mechanism for both target domain and model checking tools is presented in Section V. Section VI demonstrates the case study and Section VII evaluates the study and discusses the results. The paper is concluded in section VIII.

## II. Related Work

In the literature, some MDE studies have been conducted in order to facilitate the design, development and implementation of WSN and IoT systems. Two surveys ( [10] [14]) carried out a systematic mapping study of this domain.

According to the above-mentioned surveys, different MDE-based languages developed between the years 2007 and 2015. Furthermore, the modeling motivation of most of these studies is code-generation to increase productivity [10]. For example, the nesC language has a code generation capability for TinyOS. However, none of these studies address ContikiOS.

LwiSSy is a DSL [4] to model Wireless Sensor and Actuators Network (WSAN) systems. It allows the separation of responsibilities between domain experts and network experts. It also considers the separation of structure, behavior, and optimization concerns by using multiple views. In the study of [15], a model-driven architecture (MDA) is proposed which provides platform independent modeling (PIM), platform-specific modeling (PSM), and transformation rules for WSAN application development. Doddapaneni et al. [7] proposed a framework to separately model the software components and their interactions, the low-level and hardware specification of the nodes, and the physical environment where the nodes are deployed. This multi-view architectural approach requires linking the models together for mapping the models.

Tei et al. [17] propose a process that enables step-wise refinement to separately address data processing-related and network-related concerns. Their approach is similar to the modeling purpose of Rodrigues et al. [15]. They focus on the separation of responsibilities between domain experts and network experts for the model-based engineering of systems working on TinyOS. However, Tei et al. [17] have limited support for experts. PSM is not supported in their study and the experts simply create templates over platform independent models.

Our work contributes to the aforementioned noteworthy studies in the way of providing a model-driven engineering method for developing Contiki based IoT systems. To the best of our knowledge, currently no study addresses modeling of WSNs based on ContikiOS and required components to build a IoT system. In Durmaz et al. [9], an MDE approach is applied for Contiki-based devices by providing a meta-model for components of Contiki operating system. This meta-model is adopted and extended in our study. The unique lightweight thread structure of Contiki makes it more useful in the implementation of the WSN systems [12], [13]. This can be considered as a reason for the developers to prefer ContikiOS instead of the other existing platforms such as TinyOS. Hence, providing a modeling language as proposed in this study can facilitate the efficient development of IoT systems based on WSN.

Many systems and solutions have been designed for the fire detection system case study we address in this paper. In [11] the k-coverage algorithm is used for fire detection. In study [2], WSN devices communicate using mesh network and measure smoke, humidity and temperature sensors and these values are interpreted then concluded about fire risk. Unlike creating a mesh network like [2], the study in [5] uses GPS technology to transmit data directly to the base station and the data be accessed using web browsers. In another study [16], ad-hoc and multi-hop networks are used to detect fires in nature and send alarm messages using the shortest path algorithm. In our previous studies, [13] and [12], a cloud based library fire detection system is designed and implemented with single hop and multi hop WSNs using ContikiOS. However, none of these studies have been done using domain-specific modeling in the design and development steps.

In conclusion, this study includes the modeling of Contiki operating system, WiFi connectivity devices and other required components to develop the IoT system. The WiFi module addressed in this study is ESP8266 and the complementary components are IoT Log Manager and information processing device (such as the program on a Raspberry Pi). Furthermore, this study proposes a meta-model including the required elements of the system, graphical editor, static semantic, and code generation for the modeling environment.

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

## III. The Extended Meta-model

The meta-model introduced in this Section represents the abstract syntax of the domain-specific language proposed in this study. Generally, the meta-model defines the elements, the relations between the elements, and possibly the cardinally constraints of the relations.

In this study, we first update the meta-model in [9] by adding the missing features of the ContikiOS such as *Process Event* and *Process Post*. Then, the meta-model is extended with the elements and relations of the other components in the IoT such as ESP8266 representing a WiFi connectivity module, RaspberryPi representing a gateway, and IoT Log Manager system. In general, the proposed meta-model in this study is the result of analyzing different WSN programs working in IoT system. This task is done in close collaboration with WSN group.

WSN elements are represented with the ContikiOS elements in the meta-model including *Events*, *UDP/TCP protocols*, *Processes*, *Timers*, etc. For details about the ContikiOS part of the meta-model, the interested readers can refer to our previous study in [9]. Considering the space limitations, the extended meta-model is not shown in this paper. However, it is available on our online materials related to this paper at: Link. However, the key elements of extended parts are briefly described in this section.

The required elements and relations of the ESP8266 are added to the meta-model. All the elements of the ESP8266 are connected to a main element called *ESP* in the meta-model. There are five attributes in *ESP* element: *Name* attribute for device name, *SSID* and *Password* attributes are the credentials to connect a access point. *Host* attribute is a key parameter (API Key) to connect to the IoT Log Manager, and *SerialNo* attribute is a defining value (Tag name) to send the data generated by ESP8266 to the IoT Log Manager. The *HttpClient* element has an object *Name* attribute to make requests to the server; and *WebServer* element has a *Name* and *PortNo* attributes to create a web server inside ESP8266 and open a listening port for communication. *Wifi* element has *Mode* attribute that specifies in which WiFi mode ESP8266 operates and *Status* attribute represents the status of the established connection. *WifiMulti* element is used to connect ESP8266 to a access point in WiFi range. Moreover, a *servomotor* element is considered in the meta-model as an actuator, e.g., in the fire detection system to open doors automatically during alarm state. The *Port* attribute is indicating the listening port to interpret incoming *POST* requests. *Angle* attribute specifies rotation angle for servo motor, to open or close the door. A servo motor can be controlled by a ESP8266 using *Network* element, and it is accessible by specifying the parameters of *IP*, *Gateway* and *Sub-net* which are sub-elements of the *Network* element.

Elements of the *application* (e.g., a Java application) on a computing device such as Raspberry Pi provides connection between the WSN Nodes (specifically the Sink Node) which are using Contiki operating system and the parts of the system. *PortName* attribute indicates *Serial-port* name which physically connects RaspberryPi and Sink Node. *URLConnection* element is an element which provides network connection between RaspberryPi and the IoT Log Manager. *BufferedReader* and *OutputStreamWriter* supply RaspberryPi to read, and parse and match the data which comes from Sink node. This data usually contains node id, sensor values, and IPv6 address. Additionally, *JavaTimer* element allows the developer to call desired function in specified periods (in seconds).

The IoT Log Manager receives the sensor data from the sink nodes via gateway and also directly from ESP8266. It is possible to create various events in the IoT Log Manager. The events can be created by the end users. *Channel* is the element where data for a specific IoT system is preserved. The users can create as many *Channel*s as they need and each *Channel* has its own special *read* and *write* API key. Using these API keys, data can be sent to the channel or it can be received from channel by *POST* requests.

Finally, the main element of the system meta-model is *IoTSystem* which contains *Contiki*, *ESP8266*, *RaspberryPi* and *IoT Log Manager* elements, and it has *Name* attribute to specify the system name. The meta-model is implemented using the Eclipse Modeling Framework (EMF) Ecore [3] and is serialized in an XMI format.

## IV. The Concrete Syntax and Static Semantics

In this section, the graphical concrete syntax (i.e. the Graphical Editor) for modeling and development of ContikiOS based IoT systems is introduced. This graphical editor is developed using the Eclipse Sirius Framework [4].

Generally, the concrete syntax provides a mapping between some graphical notations and abstract elements. In this study, the graphical notations used to represent the elements in the editor are shown in Figure 1.

The developed domain-specific modeling environment consists of a multi-view and layered structure. The top layer contains the general view in the modeling of the system, and the lower layer includes Contiki view, RaspberryPi view, Log Manager view and ESP8266 model view. This reduces the complexity of the system design as the designer works with a specific view at a time. Thus, the developed graphical concrete syntax has five main views. The first view is for System layer which provides the high level view of the system. This view includes representative elements for the other view in the lower layer. An instance of System view is shown in Figure 2.

In Figure 2, the *Contiki* element is connected to *RaspberryPi* via *Contiki* relation to specify that the nodes with ContikiOS send the data to the RaspberryPi and it forwards to the corresponding channel. To transfer the data to a channel in IoT Log Manager system, *RaspberryPi* uses related *Tags* for each sensor node.

*ESP8266* has a direct connection with the *Log Manager* since it transfers the data directly to the *Log Manager* using

[3] https://www.eclipse.org/modeling/emf/

[4] https://www.eclipse.org/sirius/

| Icon | Concept | Icon | Concept |
|---|---|---|---|
| Br | BufferedReader | | ProcessPost |
| | Channel | | ProcessThread |
| | Class | | Psock |
| | ClientConnection | | PT |
| | Ctimer | | PTThread |
| | Data | | PT ThreadCall |
| | ESP | | Raspberry Pi |
| | Etimer | | Rtimer |
| | Evet | | ServerConnection |
| | Field | | ServoMotor |
| | HttpClient | | Stimer |
| | IoTSystem | | Tag |
| | JavaTimer | | TCPIPEvent |
| | LogManager | | TimeEvent |
| | Multithreading | | TimeoutEvent |
| | Network | | Timer |
| | Node | | URLConnection |
| OSw | OutputStreamWriter | | WebServer |
| | Pin | | WiFi |
| | Contiki | | WiFiClient |
| | Port | | WiFiMulti |
| | ProcessEvent | | |

Fig. 1. The graphical notations of the elements in the editor

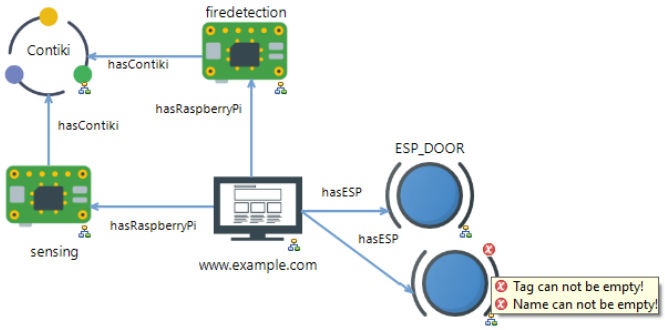



Fig. 2. High-level system view in the modeling environment

*POST* requests. The *Log Manager* is the central component of the system where all the data transfers end. It store the data and use them to make a decision to trigger another device, such as an actuator, or to do reporting, such as displaying some graphs/diagrams to the user.

Another view is *Contiki* view in which *WSN mote* elements are located. In this view, the designer can use the concepts and relations of the WSN to model the ad-hoc communication part of the IoT system. These elements are shown in Figure 1.

Figure 3 shows the *ESP8266* (on the top) and *RaspberryPi* (at the bottom) views. It includes the palette elements and relations at the right hand side, and simple instance models at the left.

For the *ESP* view, some of the main graphical notations

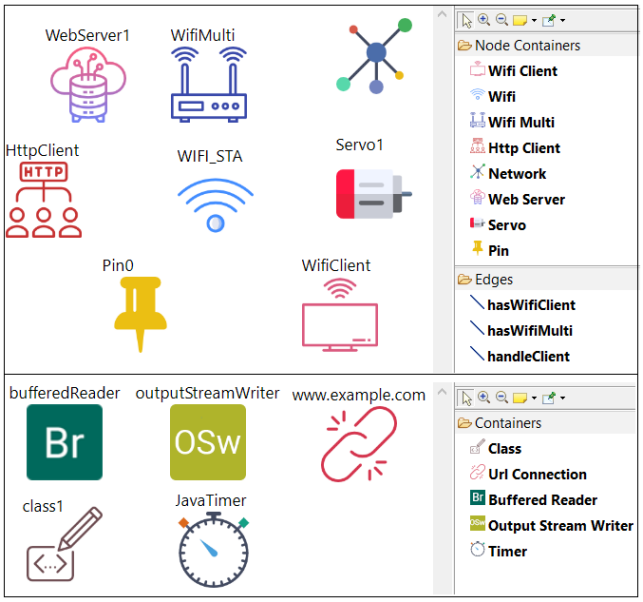



Fig. 3. Elements in ESP and RaspberryPi Views

(icons) are the *Web Server* representing the server and cloud, *WiFi Multi* representing multiple WiFi connections, *Servo* representing a Servo motor, and *Pin* representing the pin which *ESP8266* actively uses.

For the *RaspberryPi*, as it is programmed in Java, the elements in this view belong to Java programming language. For example, *JavaTimer* element (see Figure 3) specifies the Timer in Java and *UrlConnection* element is used to connect to a specific URL, in our use case RaspberryPi is connected to the IoT Log Manager web address. *BufferedReader* is used to read the data from *SinkNode* and *OutputStreamWriter* is used to write the data to *IoT Log Manager*.

The last view is the model of *IoT Log Manager*. This view can be accessed via the *IoT Log Manager* element in the *System* view. The *IoT Log Manager* elements such as *Channel* and *Tag* are shown in Figure 4 for a fire detection channel and sensing channel with some tags for specific sensors. These elements are part of the case study discussed in Section VI.

Basically, *ESP8266* and *RaspberryPi* transfer the data to a specific predefined channel using tags. For example, in a system with two channels, each with three sensors (see Figure 4), the data can be transferred to *firedection* channel using tags *sensor1*, *sensor2* and *sensor3*. The tags should be labeled in a unique way which is controlled by the *IoT Log Manager*.

To apply the domain rules in the modeling environment and help the user to design more accurate models, some semantic constraints are implemented in the framework using Acceleo Query Language (AQL)[5] which is built-in the Eclipse Sirius Framework. Some of these semantic constraints are presented bellow for the system view of our modeling environment.

Example rule 1: $aql : self.Name.size() > 0$

Example rule 2: $aql : self.esp\_tag- > size() > 0$

[5]https://www.eclipse.org/sirius/doc/specifier/general/Writing_Queries.html

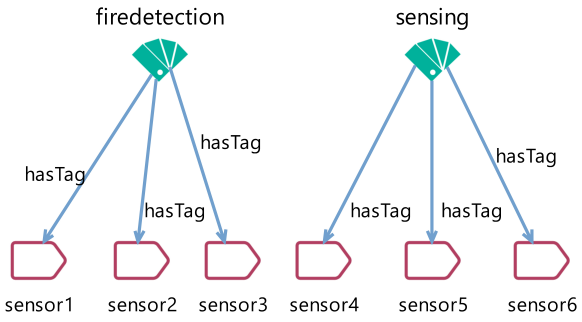



Fig. 4. IoT Log Manager View

The example rule 1 checks whether the *Name* attribute is blank or not, and the example rule 2 checks whether the tags which are necessary to connect the elements to the *IoT Log Manager* are available or not. As the result of these example rules, the framework forces the designer to fill the name spaces and provide the related tags. If the conditions are not satisfied then error messages appear in the user model as shown in Figure 2. This leads to improve the model before going to the next phase, the code and specification generation.

It is worth to note that, although the *Contiki* view is rather comprehensive and represents the main features of this WSN operating system, the other views including *ESP* and *RaspberryPi* are representative as they can be extended based on the required components (sensors and actuators) in the application domain (and related product family).

## V. Translational Semantics

The semantics of the proposed modeling language is defined by translation of the user models to both a specification for design verification and to the target domain artifacts. These transformations are realized using the Acceleo [6] template-based model transformation engine.

### A. Generation of a Design Verification Model

It is important to verify the system design model before it is implemented. To this end, the design model needs to be transformed to a formal specification/model for which required properties can be checked. This helps the designer to inspect the model and to check the structure or behavior of the system to conform to domain rules. In this way, unpredictable problems and errors can be detected during the design phase.

Petri-nets is a formalism used for verification of pertinent properties of automation systems, embedded systems, and network systems. In this study, we use Petri-nets to check, a.o., for the occurrence of bottlenecks as well as for reachability or (un)desirable states.

To this end, the user instance models (the whole IoT system model) are automatically transformed to Petri-net stochastic models in the Pipe tool to check the required properties. For this purpose, the instance model elements are transformed to places; relations are transformed to arcs; sensors' data is transformed to tokens; and transitions are added to combine two or more arcs.

[6]https://www.eclipse.org/acceleo/

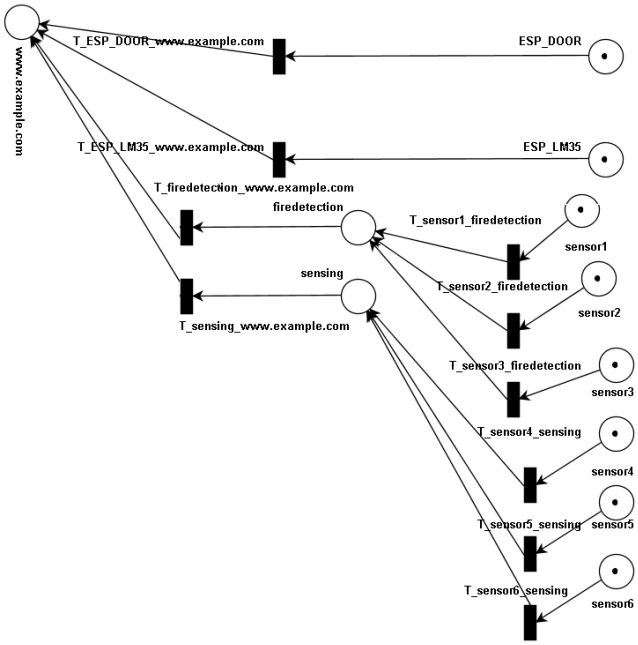



Fig. 5. A Petri-net Model generated from an Instance Model

In Figure 5, the elements which are used in Figures 2 and 4 are represented by a Petri-net. As seen in Figure 5, ESP8266 modules directly send messages and sensor nodes send messages via RaspberryPi to the IoT Log Manager (named www.example.com).

In the transformation rules, the places represent RasperryPi(s), ESP8266(s), sensor nodes(s) and IoT Log Manager. The routes of the messages are encoded by arcs, and message transitions are added to the Petri-net model. The messages are represented by tokens. In Listing 1, an excerpt of the Acceleo transformation rule is given to generate the Petri-net model as an XML file for the Pipe Petri-net analysis tool.

Listing 1. Code excerpt from Petri-net .xml file in Acceleo

```
1[file(Name.concat('.xml'),false,'UTF-8')]
2<?xml version="1.0"encoding="ISO-8859-1"?><pnml>
3<net id="Net-One"type="P/T_net">
4[for(l:LogMan|aIoTSystem.logman)]
5  [for(e:ESP|aIoTSystem.esp)]
6    <place id="[e.Name/]">
7      <graphics><position x="720"y="[90*i/]"/></graphics>
8      <name>
9        <value>[e.Name/]</value>
10       <graphics><offset x="0" y="0"/></graphics>
11     </name>
12     <initialMarking>
13       <value>1</value>
14       <graphics><offset x="0" y="0"/></graphics>
15     </initialMarking>
16     <capacity><value></value></capacity>
17   </place>
18   <transition id="T_[e.Name/]_[l.URL/]">
19   <graphics><position x="435" y="[(i*90)/]"/></graphics>
```

### B. Target Domain Artifact Generation

In this study, the translation to the target domain, so called code generation, provides the architectural code for Mote (SourceNode), SinkNode, RaspberryPi (Java Code), ESP8266 (Arduino code), and configuration for IoT Log Manager.

The code generation starts with creating the files of the elements in the IoT system. This is done using a for loop in Acceleo. For example, in Listing 2 Lines 2-4, the Java files for the RaspberryPi elements used in the model are generated. Part of the code is not directly modelled in the design model but it can be concluded indirectly, e.g., importing libraries. In Listing 2 Line 6-9, if the RaspberryPi the model includes a JavaTimer element then the rule will be executed and the import statements will be generated.

Similarly, if the model has ESP8266 element(s), then the related file(s) is/are created with "ESP8266.Name.ino" file name(s). Depending to the ESP8266 elements, the importing rule for that element, such as the one in Lines 11-13, is run and the related library is imported.

For each Contiki model/view, a C file is created. However, two types of nodes, Mote and Sink-Node, can be implemented using ContikiOS for each of which separated code file must be generated. To specify this separation, SinkNode has Server-Connection and Mote has ClientConnection element. In an instance model, if a node has a relation with a ServerConnection element, it is a SinkNode. If it does not have it, then it must be determined as Mote. This rule is shown in Listing 2 Lines 15-19.

Listing 2. Excerpt of the Acceleo rules for generating some of the files and importing their libraries

```
1[template public generateElement(aIoTSystem:IoTSystem)]
2[for(c: Class|aIoTSystem.raspberrypi.class)]
3  [file(c.Name.concat('.java'),false,'UTF-8']
4  package[aIoTSystem.raspberrypi.raspberry_channel.Name/];
5...
6[if(aIoTSystem.raspberrypi.javatimer->size()>0)]
7  import java.util.Timer;
8  import java.util.TimerTask;
9[/if]
10...
11[if(e.servo.Name.toString()<>'invalid')]
12  #include <Servo.h>
13[/if]
14...
15[for(n:Node|aIoTSystem.platform.node)]
16  [for(p:Process_Thread|n.processThread)]
17    [if(p.hasServerConnection.Name=
18      aIoTSystem.platform.serverConnection.Name)]
19      [file(n.Name.concat('.c'), false, 'UTF-8')]
```

To use UDP or TCP communication in the Contiki between the WSN motes, the required code is generated using the rule listed in Listing 3.

To create reply message a TCPIP_Event element must be added to the model. Since Contiki is an event based operating system, if any UDP or TCP packet is received, an event named tcipip_event is generated by the operating system. Therefore sending the reply message is bounded to this event.

Listing 3. Send reply according to UDP/TCP connection

```
1[let a:String=n.processThread.handlesEvent->
2  select(event|event.eClass().name='TCPIP_Event')->
3  at(i).eGet('replyMessage')]
4[if(n.UDP)] //NOTE:Add Server Connection to the model
5  PRINTF("DATA sending reply\n");
6  uip_ipaddr_copy(&[p.hasServerConnection.Name/]->
7    ripaddr, &UIP_IP_BUF->srcipaddr);
8  uip_udp_packet_send([p.hasServerConnection.Name/],
9    "[a/]", sizeof("[a/]"));
10 uip_create_unspecified(
11   &[p.hasServerConnection.Name/]->ripaddr);
12[/if]
13[if(n.TCP)] //TCPIP Reply
14  sprintf(buf, "[a/]"(%d)",1);
15  PRINTF(buf);
16  PRINTF("\n\r");
17  uip_send(buf,strlen(buf));
18[/if]
19[/let]
```

Using these rules the required files and their architectural code are generated automatically for different components of the IoT system. These artifacts should be completed using the user's delta code defining the domain-specific business logic such as the controlling conditions to have a fully functional system.

## VI. Case Study: Smart Fire Detection System

The proposed modeling framework is evaluated using a use case called fire detection system. In this domain, the IoT components are used to recognize the signs of a fire in a large area and to react instantly.

To achieve this, sensor nodes sample the temperature in each 4 seconds. Thus, the fire can be prevented at the earliest possible time. In the system view of this case study, see Figure 6, the IoT Log Manager is located in the center of the system. SinkNode, ESP8266 Module (with its LM35 sensor), and RaspberryPi are connected to the IoT Log Manager. Motes can send packets to a sink node using multi-hop communication. The SinkNode takes the data from the closest source node (Mote) and transfers the data to RaspberryPi. Since RaspberryPi has direct access to IoT Log Manager, it transmits the data which is collected from sensors to a relevant channel by matching sensor's Tag. ESP8266 with a LM35 sensor, measures the temperature of its surrounding, and sends the temperature data to the IoT Log Manager.

The IoT Log Manager keeps the received data in PostgreSQL database, and visualizes them using the Blazer query tool (see Figure 7) based on the user query script. This helps the user to provide the desired reports.

In IoT Log Manager, a channel is defined by giving a Channel name and Channel description and the events are created in this Channel to control the conditions and trigger proper actuators. The end users can use his/her logical comparisons for events or write their own complex scripts to create various events. The IoT Log Manager compares the values which comes from RaspberryPi and ESP8266 and if the temperature is above a set-point, then it sends an e-mail to alert the people who are in charge. Moreover, Notifications are sent to smartphones via Pushbullet[7] application to alert the user about the danger.

There are also two ESP8266 modules with servo motors in the system that control the door locks. In the case of

[7]https://www.pushbullet.com/

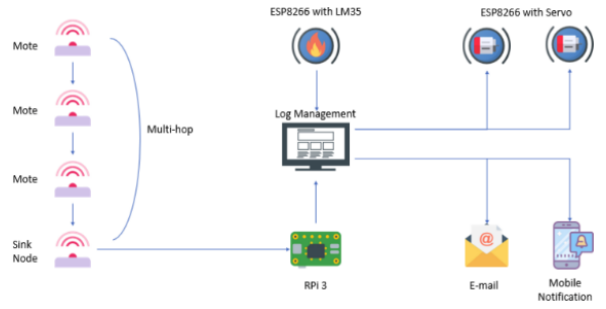



Fig. 6. Design of smart fire detection system


Fig. 7. Report and graph generation screen with query tool

emergency, the IoT Log Manager sends POST messages to the ESP8266(s) to release the door locks (to open the doors). Of course, there can be any number of doors or any other devices which can be controlled by an event. Finally, the people in the building can send their status, such as pushing the Safe button in the app, as the feedback to the system.

Due to page limitations, some of the other views and models of this case study are shown as examples in the previous sections. For example, the IoT Log Manager view of this case study is shown in Figure 4 modeling the related channels and sensor tags. Also, in Figure 5 the generated Petri-net model for this case study is demonstrated. Using this model, the bottleneck analysis is done for this system. Also, the complete model of Contiki view is not shown in this paper due to its size. However, it is available in the online bundle of the paper discussed in Section III.

Finally, for testing the use case, the motes were placed at a distant where they can communicate with each other and create a mesh network. While RaspberryPi was receiving sensor data which was collected by Sink Nodes and ESP8266 modules from the environment, one of the nodes was heated by a lighter. When the IoT Log Manager has detected that the temperature is rising, a callback has been sent to the ESP8266 which controls the servo motor to open the door lock. At the same time, a notification has been sent to the smart phone and an e-mail has been sent to the user to warn about danger.

## VII. Evaluation

The general evaluation of the modeling platform is done via assessing its generation performance. To this end, we

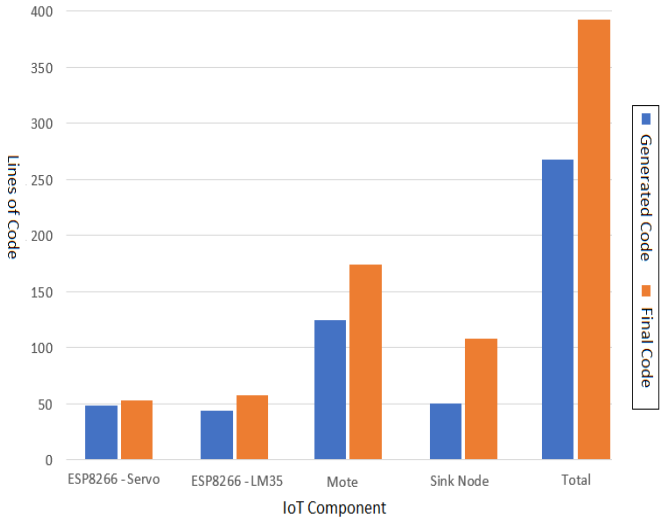



Fig. 8. Generation performance of the proposed modeling framework

have analyzed the generation capability of the framework by comparing amount of the generated architectural code with the final code (the code after adding the end-user's delta code).

We are aware that the evaluation with single use case has the external threat to the validity of the results which can risk the generalization of the results. However, a systematic and extensive evaluation of the approach with multi case study will be done in our next study which will focus both the generation performance and development time performance [3].

Figure 8 shows the comparison of generated lines of code and lines of the final code for different components of the IoT system. According to these figures, the generation performance of the IoT component in this study are about 96% for ESP8266-Servo, 78% for ESP8266-LM35, 71% for Motes, 47% for Sink-Node, and 70% in the total. These numbers show the rate of generated code considering the final code.

As you can see, the Sink Node has the lowest generation rate of almost half of the code, and ESP8266-Servo has the highest with almost all of the code generated automatically. But the Sink Node is not the biggest part of the total code. The biggest part of the code belongs to the Mote with 71% of generation. As the result, more that two third of the whole code in the smart fire detection system is generated.

It is worth to discuss that, generally, some parts of the code cannot be generated in this modeling environment, as they are very specific for the scenarios and the model does not cover that much detail. In fact, the level of abstraction in the modeling language is a design choice. If you involve too much details in the model, the higher level of abstraction can be violated, as the model can be as complex as the target code itself. This is a trade-off which should be assessed by evaluations to reach the optimum level of abstraction which considers both ease of design (less design complexity and less design time) and higher generation performance.

## VIII. Conclusion and Future Work

In this study, a domain specific modeling language and its supporting tools are developed for design and implementation of IoT systems to address the increasing complexity and difficulty of these systems. The code generation mechanism provides the architectural codes for ESP8266, ContikiOS and RaspberryPi as well as the specification for design verification in Petri-net.

To this end, the Contiki meta-model available in the literature (implemented in EMF) is extended by the other components required for an IoT system such as Sensors connected to a transceiver module (ESP8266), a computation device as the gateway, and a Log Manager. This meta-model is used as an abstract syntax of the DSML and a set of graphical notations as well as some domain constraints are used to develop the graphical editor for the proposed modeling environment.

ContikiOS provides a light-weight multi-threading mechanism for bi-directional and multi-hop communication of sensor nodes which enables them to cover a large area. RaspberryPi has the important role of bridging the different communication approaches which are used in Contiki based WSNs (IEEE 802.15.4 protocol) and WiFi Modules such as ESP8266 transceiver module (IEEE 802.11 protocol). Finally, Log Manager is the event based central data repository of the system which also handles the triggering the actuators and reporting the data.

Furthermore, the system automatically generated Petri-net model helps the end user or IoT developer to analyze and verify the design model at the early phases of the development, before any implementation, to check the critical properties for the IoT system under development.

Finally, based on the evaluation results, the proposed approach for the development of IoT systems, using the provided generative modeling environment, can reduce the development time and cost.

As a future work, the evaluation of the performance of the proposed approach will be done in a systematic way, using multi case studies and two groups of developers. In this way, we will be able to evaluate the development-time efficiency of the approach as well as generation-performance of the modeling framework.

As another future work, the level of abstraction for the modeling environment is going to be increased to also support Platform Independent Level. The current study, focuses on a set of specific target platforms (e.g., ContikiOS) and provides a Platform-specific Modeling environment for the development of IoT systems using WSNs and WiFi modules. This future work will extend the current work to support modeling of IoT systems independent of their target platforms. This will be done using commonality and variability analysis of WSN Operating Systems such as ContikiOS, TinyOs, RIOT and so on, and also using model to model transformations.

## Acknowledgment

Some of the ideas in this work were developed during Short Term Scientific Missions (STSMs Nos. 41940 and 41967) by two of the authors, within the IC1404 Multi-Paradigm Modelling for Cyber-Physical Systems (MPM4CPS) COST Action.

## References

[1] S. Arslan, M. Challenger, and O. Dagdeviren. Wireless sensor network based fire detection system for libraries. In *Computer Science and Engineering (UBMK), 2017 International Conference on*, pages 271–276. IEEE, 2017.

[2] Z. Chaczko and F. Ahmad. Wireless sensor network based system for fire endangered areas. In *Third International Conference on Information Technology and Applications (ICITA'05)*, volume 2, pages 203–207, July 2005.

[3] M. Challenger, G. Kardas, and B. Tekinerdogan. A systematic approach to evaluating domain-specific modeling language environments for multi-agent systems. *Software Quality Journal*, 24(3):755–795, 2016.

[4] P. Dantas, T. Rodrigues, T. Batista, F. C. Delicato, P. F. Pires, W. Li, and A. Y. Zomaya. Lwissy: A domain specific language to model wireless sensor and actuators network systems. In *2013 4th International Workshop on Software Engineering for Sensor Network Applications (SESENA)*, pages 7–12, May 2013.

[5] N. S. David M. Doolin. Wireless sensors for wildfire monitoring. In *Proc.SPIE*, volume 5765, pages 5765 – 5765 – 8, 2005.

[6] S. Demirkol, M. Challenger, S. Getir, T. Kosar, G. Kardas, and M. Mernik. Sea_l: a domain-specific language for semantic web enabled multi-agent systems. In *Computer Science and Information Systems (FedCSIS), 2012 Federated Conference on*, pages 1373–1380. IEEE, 2012.

[7] K. Doddapaneni, E. Ever, O. Gemikonakli, I. Malavolta, L. Mostarda, and H. Muccini. A model-driven engineering framework for architecting and analysing wireless sensor networks. In *Proceedings of the Third International Workshop on Software Engineering for Sensor Network Applications*, SESENA '12, pages 1–7, Piscataway, NJ, USA, 2012. IEEE Press.

[8] A. Dunkels, B. Gronvall, and T. Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. In *29th Annual IEEE International Conference on Local Computer Networks*, pages 455–462, Nov 2004.

[9] C. Durmaz, M. Challenger, O. Dagdeviren, and G. Kardas. Modelling Contiki-Based IoT Systems. In *6th Symposium on Languages, Applications and Technologies (SLATE 2017)*, volume 56 of *OpenAccess Series in Informatics (OASIcs)*, pages 5:1–5:13, Dagstuhl, Germany, 2017.

[10] F. Essaadi, Y. Ben Maissa, and M. Dahchour. Mde-based languages for wireless sensor networks modeling: A systematic mapping study. In R. El-Azouzi, D. S. Menasche, E. Sabir, F. De Pellegrini, and M. Benjillali, editors, *Advances in Ubiquitous Networking 2*, pages 331–346, Singapore, 2017. Springer Singapore.

[11] M. Hefeeda and M. Bagheri. Wireless sensor networks for early detection of forest fires. In *2007 IEEE International Conference on Mobile Adhoc and Sensor Systems*, pages 1–6, Oct 2007.

[12] B. Karaduman, T. Aşıcı, M. Challenger, and R. Eslampanah. A cloud and contiki based fire detection system using multi-hop wireless sensor networks. In *Proceedings of the Fourth International Conference on Engineering & MIS 2018*, page 66. ACM, 2018.

[13] B. Karaduman, M. Challenger, and R. Eslampanah. Contikios based library fire detection system. In *2018 5th International Conference on Electrical and Electronic Engineering (ICEEE)*, pages 247–251. IEEE, 2018.

[14] I. Malavolta and H. Muccini. A study on mde approaches for engineering wireless sensor networks. In *2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications*, pages 149–157, Aug 2014.

[15] T. Rodrigues, F. C. Delicato, T. Batista, P. F. Pires, and L. Pirmez. An approach based on the domain perspective to develop wsan applications. *Software & Systems Modeling*, 16(4):949–977, 2017.

[16] B. Son, Y.-s. Her, and J.-G. Kim. A design and implementation of forest-fires surveillance system based on wireless sensor networks for south korea mountains. *International Journal of Computer Science and Network Security (IJCSNS)*, 6(9):124–130, 2006.

[17] K. Tei, R. Shimizu, Y. Fukazawa, and S. Honiden. Model-driven-development-based stepwise software development process for wireless sensor networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(4):675–687, April 2015.

# A Software framework for Procedural Knowledge based Collaborative Data Analytics for IoT

Snehasis Banerjee
*TCS Research & Innovation*
*Tata Consultancy Services*
Kolkata, India
snehasis.banerjee@tcs.com

M Girish Chandra
*TCS Research & Innovation*
*Tata Consultancy Services*
Bangalore, India
m.gchandra@tcs.com

***Abstract*—The outburst of data generation by machines and humans, along with emergence of sophisticated data processing algorithms have created a demand for a wide number of data analytics based services and applications. The paper presents a collaborative framework and system to carry out a large number of data processing tasks based on semantic web technology and a combination of reasoning and data analysis approaches using software engineering guidelines. The paper serves as a first step for systematic fusion of symbolic and procedural reasoning that is programming language agnostic. This approach helps in reducing development time and increases developer's productivity. The proposed software system's logical functionality is explained with the help of a healthcare case study, and the same can be extended for other applications.**

***Index Terms*—Procedural Reasoning, IoT Analytics, Software Framework, Software Orchestration**

## I. Introduction

Internet-of-Things (IoT) [1] and the services around it is poised for a disruptive growth in near future mainly because of huge number of sensor deployments and advances in networking technologies. In fact, Gartner predicts 21 Billion IoT Devices to be deployed by 2020 [2]. As data is considered the new 'oil' (fuel) for innovations in Industry 4.0 [3], it has been found that data analytics on IoT is in huge demand across all business domains and services around it [4].

Among the various sensors that are deployed (apart from camera and allied sensors) most produce one-dimensional (1-D) readings (say temperature sensor) or multiples of 1-D readings in combination (say accelerometer sensor). Doing analytics on such type of sensor data can be mapped to modules of algorithms belonging to signal processing and machine learning fields. This can be abstractly formalized in a flow diagram for IoT Analytics (Fig. 1).

A typical IoT workflow will comprise (a) Data retrieval from sensors (b) Pre-processing the data by applying various methods like formatting, noise cleaning and anomaly removal (c) Applying data transformation to another representation form like Fourier transform or Wavelet transform (d) Applying Feature Engineering principles of feature extraction from a region of interest followed by selecting features by application of a set of feature selection algorithms (e) finding a suitable Model for the data and problem post tuning modeling algorithm parameters (f) optional inferencing step to derive high level deductions from discovered model (g) Visualizing the results (in a way domain experts can understand).

All of the aforementioned steps need a repository of algorithms for each module and invocation as per need. The traditional approaches for IoT Analytics have confined themselves to quantitative analysis of the data based on the problem at hand. A standing issue is that different suites of algorithms are available in different programming language platforms and this leads to manual intervention or asynchronous and sequential analysis of an IoT Analytics task. The work presented here tries to address this problem. Some work [5] [6] [7] has been done on realizing a data analysis framework for IoT, however they mostly lack the semantic angle and efforts to create a generic framework where problems of different domains can be plugged. Utilization of semantic technologies in IoT has been surveyed in [8], however, only a few IoT applications are found to utilize semantic technologies or in general symbolic techniques of analysis. In this work, an attempt has been made to combine the symbolic and quantitative techniques suitable for a typical IoT Analytics workflow execution. To make this happen, the proposed system has both a Quantitative Analytics Engine and a Symbolic Reasoning Engine that works in synergy by an implicit central controller mechanism. Another burden in a typical IoT Analytics problem is the variety of domain dependent solutions, which is overcome by separating the logic from generic procedures and storing them in rules, ontologies and knowledge stores. This lays foundation for a domain-independent general purpose reasoning based software framework for each IoT Analytics module. A major issue in a Reasoning enabled Analytics system is lack of support of procedural evaluations and keeping check of facts that have turned obsolete or false. This is overcome by (a) sticking to non-monotonous reasoning following the philosophy of truth maintenance systems [9] and (b) allowing seamless integration of procedural execution into the heart of reasoning following software engineering principles.

The main contributions of this work are:
(1) a collaborative software framework and system is presented that leverages semantic techniques (2) enabling seamless procedural evaluation embedded in semantic rules targeted for IoT data analysis tasks (3) a case study in health-care that illustrates the utility of this approach.


Fig. 1. A typical IoT Analytics Workflow most suited for one-dimensional sensor data

Section 2 discusses about the background for IoT Analytics work-flow from a knowledge based perspective. Section 3 describes the proposed system framework and section 4 shows the solution approach taken. Section 5 illustrates procedural reasoning approach of the system and its implications with a healthcare case study. Finally section 6 concludes the paper after laying down future scope of work.

## II. Background

The semantic web technologies that can be used in developing IoT Analytics applications are enlisted here:

a) RDF[1]: A RDF triple, also called a fact, contains three components: (a) subject, which is an RDF URI reference or a blank node (b) predicate, which is an RDF URI reference (c) object, which is RDF URI reference, literal or blank node.
An example in triple (<subject><predicate><object>) form:
<sensor:TemperatureSensor><rdf:property><temp:Celsius>

b) Ontologies: used for defining class relationships and set theoretic constraints on relationships. This is typically manifested in OWL[2], a knowledge representation language in machine interpretable form. OWL enables reasoning from RDF enabled sources. Some of the well known ontologies in IoT space are OGC[3], SensorML of SWE[4] and SSN[5]; eg. :
<sml:input name="SurfaceTemperature"> <sml:
ObservableProperty definition="http://sensors.ws/ont/
NMMO/sensor/SeaSurfaceTemperature"/ > < /sml:input>

c) SPARQL[6]: used to query RDF models; eg. :
SELECT avg(?value) WHERE {
<sensor:TemperatureSensor> prop:hasValueInCelsius ?value}

d) Semantic Rules: Reasoning with rules is typically based on first-order predicate logic or Description Logic (DL) to make conclusions from a sequence of statements (premises) derived by predefined rules. A reasoner deduces facts from existing semantic data and ontologies based on predefined rules. Common reasoning and inference engines such as Jena[7] and Pellet[8] are based on different rule languages and have support for ontologies and OWL. Some of the reasoners support SWRL[9] and RIF[10] rule languages, whereas others have implemented their own human readable rule syntaxes. For reasoning on IoT Analytics workflow, Jena was found suitable for its support of custom rules and ease of implementation of extensions. A Jena Rule looks like:
[transitiveRule: (?P rdf:type set:TransProp)
(?A ?P ?B) (?B ?P ?C) − > (?A ?P ?C) ]

e) Semantic Graph Databases: used when large amounts of facts are to be stored and processed either in batch or on-demand. These are also called RDF Databases and popular ones include Virtuoso[11], Jena TDB[12] and AllegroGraph[13].

### A. Reasoning

Computer Reasoning relies on precise rules leading from a set of well-formed statements [10]. One way to classify reasoning approaches is in terms of monotonicity. In case of Non-monotonic reasoning (such as truth maintenance system), facts can become false in future, which directly relates to practical scenarios. This requires monitoring of facts so that when a fact becomes invalid because of a context change, facts derived as a consequence of that fact is removed and so on as a chain reaction. Reasoning techniques involves working with predicates and facts. A predicate on a domain, S, maps every element of S to either true or false, that is one of T,F, which can be viewed as a relation on S $\times$ {T,F}. A function on a domain, S, maps every element of S to a unique element of the co-domain (or range), R, so can be viewed as a relation on S$\times$R. Any function, f : S$\rightarrow$R, can be converted to a predicate, p : S$\times$R by defining p(s,r) $\Leftrightarrow$ f(s) = r. Similarly, any predicate p:S can be converted to a function f : S$\rightarrow${T,F} by defining f(s) = T $\Leftrightarrow$ p(s) and f(s) = F $\Leftrightarrow$ $\neg$ p(s). Functional predicates are useful when in a reasoning enabled system, procedural evaluation is needed to evaluate inferences based on dynamic facts.

### B. Knowledge based Data Analytics

Knowledge has been found to play a crucial role in data analytics. Most of the time, the solutions are dependent on knowledge of domain, algorithms and application. The data analysts use these varied knowledge to filter out non-consistent models as well as apply correct steps in deriving inferences from data at hand. A survey [11] on Semantic Web and Data Analytics convey with examples how semantic web technologies can become an integral part of a typical data analysis workflow. It specifically mentions 3 types of ontologies namely (a) Domain: that exxpress background knowledge about the application domain, (b) Ontologies for data analytics process: that define knowledge about the data mining process, its steps and algorithms and their possible parameters, (c) Metadata ontologies: Describe meta knowledge about the data, such as provenance, format and source.

[1]Resource Description Framework: https://www.w3.org/RDF/
[2]Web Ontology Language: https://www.w3.org/OWL/
[3]Open Geospatial Consortium: www.opengeospatial.org
[4]Sensor Web Enablement: www.sensorml.com/standards.html
[5]Semantic Sensor Networks: w3.org/2005/Incubator/ssn/ssnx/ssn
[6]SPARQL Protocol and RDF Query Language: www.w3.org/sparql
[7]Jena under Apache License: https://jena.apache.org
[8]Pellet Reasoner: https://www.w3.org/2001/sw/wiki/Pellet
[9]Semantic Web Rule Language: www.w3.org/Submission/SWRL
[10]Rule Interchange Format: www.w3.org/TR/rif-overview
[11]OpenLink Virtuoso: https://virtuoso.openlinksw.com/
[12]TDB: https://jena.apache.org/documentation/tdb
[13]Franz AllegroGraph: https://allegrograph.com

Some recent works on knowledge based data analytics are: (a) [12] Publish-Subscribe based Knowledge Analytics that uses automated schema mapping to overcome data heterogeneity and uses semantic inferences on ontology enhanced data, (b) [13] shows how Knowledge graphs can aid in knowledge discovery for data mining, (c) [14] shows how formal semantics in ontologies can be incorporated into data analytics as the formal structure of ontology makes it a natural way to encode domain knowledge, (d) [15] describes an ontology-based system that deals with instance-level integration and data preprocessing. It uses a preprocessing ontology to store information about the required transformations.

## III. System Description

The proposed system can be represented as a tuple $\{$ D, M, W, O, R, Q, K, A, L $\} \rightarrow$ I, where D = data, M = metadata, W = workflow templates, O = ontologies, R = rules, Q = queries, K = other knowledge sources, A = algorithm repository of registered procedural evaluation, L = learning algorithm recommendation knowledge and I = inferences drawn post application of reasoning and learning.

The major modules of the system (Fig. 2) are as follows:

1) Data Handler: the purpose of this module is to (a) handle bursts of data when running in live mode by using priority queues (b) using batch to process large data files when in offline mode. The module checks any inconsistency in data (such as mis matched rows and columns, data format support) and raises a flag to the user accordingly.

2) Workflow Manager: this dedicated module comprises of a workflow engine whose main task is to compose and execute data analytics workflows. This is achieved by composing an apt workflow from templates based on given data, meta-data and knowledge around it. The workflow manager controls the workflow instance in execution and can enable to and fro data and control exchange between Analytics Engine module and Reasoning Engine module. Further details of this module are kept out of scope to focus on the knowledge aspects.

3) Metadata store: this module contains both specific and generic metadata around algorithms used for analytics. Also meta-data related to given dataset is used as a knowledge input at different stages of analysis.

4) Analytics Engine: this module comprises of the list of analytics algorithm instances (such as those drawn from machine learning and statistical analysis) and their mappings to specific data analytics task. The workflow in execution links with this module when any data analytics task (such as classification) or sub task (such as data transformation) needs to be carried out.

5) Knowledge Store: this module consists of knowledge that is related to datasets, application, domain, workflow templates and algorithms. The knowledge can be in form of (a) ontologies such as OWL (b) facts such as RDF (c) formatted schematic files such as XML (d) flat files (such as csv) with tags or mappings for machine understanding of content (e) RDF stores (f) external databases with defined schema (for relational) or entities (for graph based).

6) Data Unifier: this module is responsible for unification of different data and knowledge formats into a consistent form, which is required for reasoning. Based on standard templates and keywords, mappings are carried out to represent data and knowledge into standard semantic form like RDF and OWL. One important thing in this respect is that the raw data is not converted into RDF form, only the meta-data around the data such as inherent columns and externally supplied metadata is. This saves overhead of re-converting the RDF facts to the initial numeric or categorical values. Each numeric value is tagged as a literal (such as string, double, etc.) in RDF, that adds individual literal tags to each instance of data value. RDF schemas are descriptive such as vocabulary definition, not prescriptive like relational schemas where data must take the prescribed form. This will create a large RDF data file - hence this approach is avoided. The pointers to data sources are kept as RDF facts in the current working memory. However, aggregation of data values that may lead to meaningful inferences can be stored as RDF facts such as mean, median, standard deviation of data fields.

7) Ontology Repository: this module maintains a list of references to all ontology type knowledge files with description. Users are free to add their own custom ontologies or link to an ontology on the web.

8) Rule Repository: this module contains all rules registered in the system via Rule Registry, including procedural rules. Rules have names and are mapped to general usage or specific applications so that selective invocation can be done on demand. System has a large repository of in-built rules which include set theoretic operations as well as commonly used logical expressions. Rules need to be binded to a Reasoner instance for execution. There is a provision to add custom rules including those having procedural functions.

9) Query Repository: this module contains queries which are used in order to view the inferences drawn via rule firing. The repository is pre-filled with simple template SPARQL like semantic queries which can be extended by user as per application demand. Some logging queries are pre-registered that track execution of the Reasoning engine.

10) Reasoning Engine: this important module is concerned with three principal tasks: a) supporting procedural evaluation when firing rules b) provision of apt workflow recommendation based on inferences drawn on given problem and dataset c) algorithm selection for each of the steps of a workflow in execution. The 'Knowledge Bus' connects the Reasoning module to the different knowledge repositories discussed above. A Listener service is registered in the Knowledge Bus that checks if any change occurs in any of the repositories. A provision is kept to store inferred knowledge and facts in their apt repositories for future use.

Reasoning engine is composed of the following sub-modules:

(a) Working Memories: all the relevant facts and ontologies are loaded in a dedicated memory space which is called working memory in computer reasoning terminology. It usually has a limited capacity that is responsible for temporarily holding

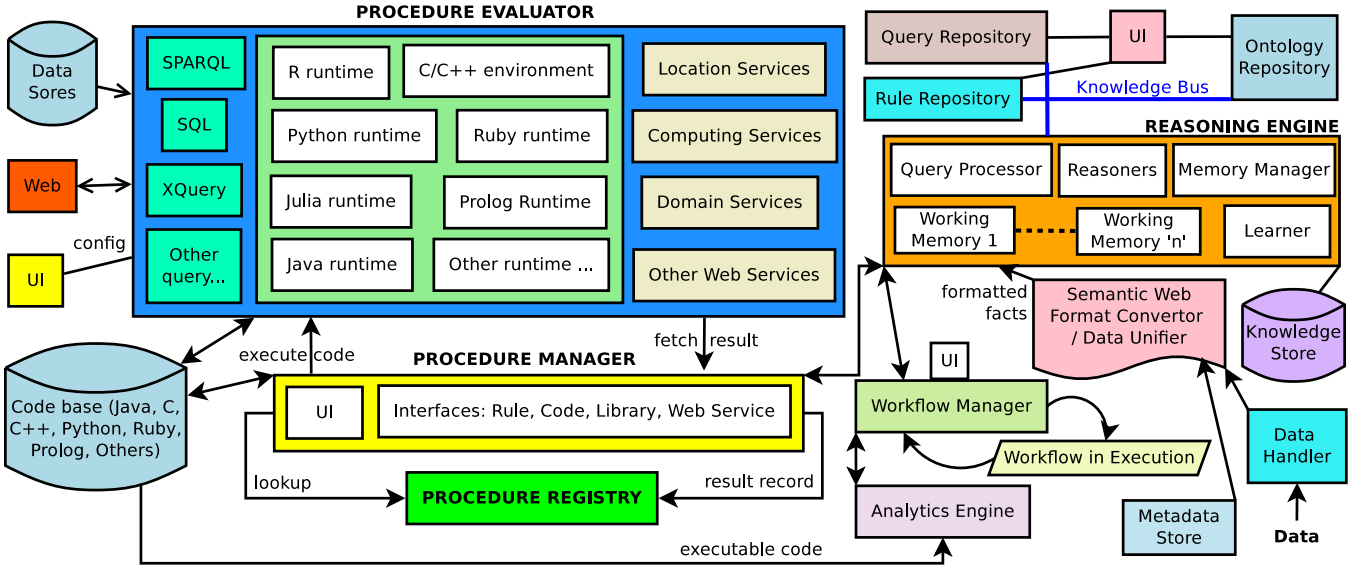



Fig. 2. Proposed Collaborative Framework for Data Analytics based on Hybrid Knowledge Processing

information available for processing. With advancements in distributed computing and memory technology, it is possible to hold large amounts of facts in memory for fast processing. Facts (or tuples) lying in a working memory under the non-monotonous reasoning paradigm are considered as true, and obsolete facts are deleted from it. In other words, working memory represent the current logical state of a system. The patterns of facts loaded in a working memory is based on the patterns existing in left hand side of rules (productions) and query expressions for a specified workflow execution.

(b) Query Processor: this module supports two types of queries: 1) Ad-hoc: this can be issued by users on demand to query the working memory for situations like result visualization 2) Registered queries: this are pattern seeking queries already linked to the working memory and trigger at regular intervals or on some condition satisfaction.

(c) Reasoners: this module consists of a library of 1) predefined standard reasoners such as RDFS or OWL reasoners, where logical inferences are fixed 2) custom reasoners made by binding custom rules including procedural rules to embed logic as per need. A choice of hybrid, forward and backward chaining are available to the user to choose. If the number of rules for a problem is large, then forward chaining is better as rules get triggered only when matching patterns in data are observed. Forward chaining being data driven, rule processing continues until no more rules can be applied or some cycle limit is met. As this can generate a huge number of inferences (new RDF facts) that can bloat working memory, forward chaining is advised for usage in event driven scenarios. Backward chaining on the contrast is a goal driven inference technique. It starts from possible conclusion or goal and aims for necessary data. This makes backward chaining suitable when number of rules are few and when working memory has limitations in capacity. For small datasets and rules, the earlier two approaches or a hybrid one usually suffice, but for large amounts of data and large number of rules, Rete is the optimal choice of reasoning. The Rete algorithm provides a generalized logical description of an implementation of functionality responsible for matching facts against rules in a pattern-matching production system, ie. a rule engine. Rete avoids iterating through the data elements by storing the current contents of the conflict set in memory, only adding and deleting items form it as data elements are added and deleted from memory. Rete algorithm is designed to sacrifice memory (by creating and maintaining Rete Network of Alpha and Beta Nodes) for increased speed. Rete has three components: 1) Alpha Network - left side of the node graph (rule patterns) that forms a discrimination network and selects elements from working memory based on simple conditions 2) Beta Network - the right side of node graph which mainly performs joins between elements in working memory, 3) Agenda - typically implemented as prioritized queues, that finds the order in which a rule is fired in a match-resolve-act cycle.

(d) Memory Manager: this module retrieves resources from various repositories via Knowledge Bus. It manages the working memories and allocates and deallocates memory areas to individual reasoner instances based on need. What to load in the working memory is defined based on the dataset, application and problem description set by the user, apart from support of direct manual loading from the user's end. The system dedicates a working memory for each application instance. Due to the nature of knowledge and applications, it is often found that there is a significant overlap of the knowledge needed across use cases. Hence, keeping redundant facts across disjoint working memories adds to overhead in processing as well as space. This can be overcome by having multiple working memories with sharing support [16].

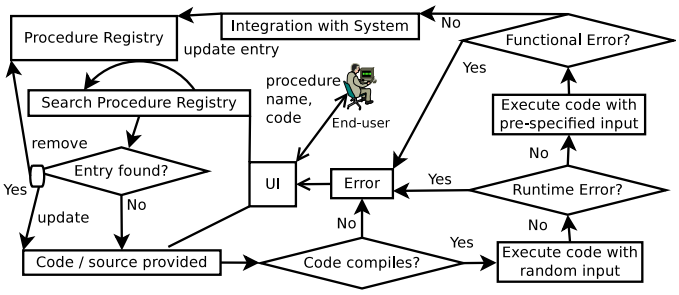



Fig. 3. Flowchart of Procedure Update Validation

(e) Learning Module (Learner): This module learns based on execution logs of the reasoner. System learns which facts and rule bindings need to be kept in memory in cache based on frequency of usage.

11) Procedure Manager: this module is the central control to manage procedural evaluations while reasoner is in execution. When a new procedural rule needs to be registered, the rule binding happens in the reasoner (implicit reasoner registry) while the procedure mapping happens in Procedure Registry. Procedure Registry maintains list of algorithm codes with pointer, names, arguments, parameters and meta-data. Users can define their own procedures to extend the built in functions of the reasoning vocabulary of keywords and pointers to executable methods. Fig 4. illustrates the user's perspective on semantic rule formation as per demand of the solution logic. The registry contains following types of functions (methods or procedures):
a) Standard functions: some frequently used functions like standard deviation b) Defined functions: functions added by the user c) Shared functions: functions added by other users and shared for re-usability. d) Web Service functions: web service APIs registered in system e) External functions: external tools and database connections binded to registry.

12) Code Base: the executable files are kept here. Support for various languages such as Java, C/C++, Ruby, Python, Julia, Prolog, R is kept as a) different languages has different functionality that can be leveraged, b) users usually have their own preference of languages, c) different library of functions are available for each language and a specific required code library may be available only in one language.

13) Procedure Evaluator: this is composed of runtime environments of different programming languages, where the values are passed and code gets executed. Connectivity to external databases (such as those listed in LOADB[14] and DBPedia[15]) supporting queries via SPARQL, SQL and other querying methods are maintained to link aptly at the time of procedural evaluation. Various services and their APIs are kept registered to use the functionalities offered. Services include Location (like Google Distance Matrix API), Computing (like evaluation via Wolfram Alpha) and Domain (like information

[14]http://www.loadb.org
[15]https://wiki.dbpedia.org/

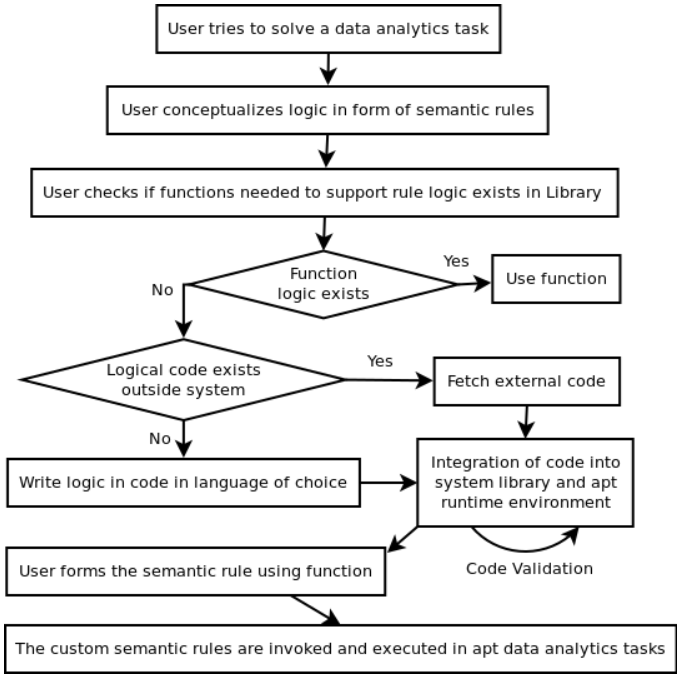



Fig. 4. Flowchart of custom semantic rules formation from user perspective

around disease outbreaks). Inter connections between runtimes are kept to minimize context switching.

The system has multiple front end user interfaces binded to specific modules that serve as means to update files, configurations and defining relations and placeholders in the system. Due to limited space, discussion on UI is skipped.

## IV. Method Description

The proposed system's underlying principle has been chosen to be non-monotonic forward chaining deductive reasoning supporting functional predicates to suit needs of IoT workflows. Logic can be expressed both in form of semantic rules and SPARQL like graph pattern queries. However, to maintain the truth maintenance aspect, logic in this case is restricted to semantic rules having provision for fact removal. SPARQL is used as a way to view and retrieve answers to queries (patterns under scrutiny). Rete [17] and its improved variations [18] [19] have stood the test of time for handling fast pattern matching in a non-monotonous reasoning scenario. For IoT based workflow execution, knowledge decisions based on Jena's implementation of Rete forward reasoner was found suitable as Rete is able to handle large number of rules and fast fact deletion based on its alpha-beta networks and efficient conflict resolution strategies.

### A. Handling External Procedures

External procedures in any supported programming language can be added to the system. Fig 3. shows how a user can register a procedure to be binded in Procedure Registry. User starts with searching for a procedure name in the existing registry and if entry found, can view the code if it is public and can update (also remove) the same if access rights are

granted. Else, user creates a new entry of procedure by linking or uploading code in language of choice. The system checks (in the runtime environment) if the code gets compiled and executes code with random input to see if any runtime errors or exceptions happen. Next, system asks user to provide a set of input and expected output and sees if the procedure yields functionally correct results. Next, the procedure gets integrated with the Registry and a user will be able to use the function names and arguments in defining custom rules. Any errors in the process are shown to user for rectification.

### *B. Collaborative Development*

The users can do the following in the UI connected to Procedure Registry: a) Updating / adding/ removing custom functions (algorithmic methods) b) Sharing functions with all users or a group or keeping it private c) Viewing source code or other implementation details of shared functions with apt permissions d) Commenting, rating and providing suggestions for different Registry functions e) Tag the self created functions with meaningful meta-data f) In the particular session, users can turn on and off certain functions. This comes handy in case of overriding functions like the different interpretations of a function bearing the same name.

Fig 4. illustrates the user's perspective on semantic rule formation as per demand of the solution logic. In order to solve a data analytics task, user thinks of the problem at hand and how to conceptualize a solution about it. Usually the path taken to solve that problem can be thought of semantically connected steps whose logic can be expressed in rules. User next checks in the system where building blocks of rules (combining which a complete rule expression will be formed) exist in the system, that were registered by other developers. If a building block (in this case, a functionality) exists in the rule processing system, then the user composes a rule based on logical requirement of the task. Else, the user is given a choice to add required procedural logic as building blocks of rules to be added to the system. If the procedure already exists outside the system, user can just fetch the code and let the system integrate it automatically. Else the user can write the code in an editor and integrate in the system. Code is validated before registering the function with a name in the system. User forms a set of rule expressions to execute the same in the system to serve a specific task on a given data.

Functions or methods stored in the system are analyzed about their time and space requirements, frequency of usage, modification, extension and other details. This aids in creating effective recommendation of functions when a user searches the Registry for existing procedural functions. User created functions can be rated, shared and extended to for a collaborative development ecosystem. Functionality of added functions can be tested and validated, with feedback given to the initial procedural code provider.

System creates a new URI (uniform resource identifier) for the new function being added with its programming language as a sub part if URI. This helps in distinguishing functions with same names based on domain and nomenclature. This URI assignment is done automatically following standard pre-defined standard templates. An example is shown:- http://com.algorepo/number/c/2/1/resample
where number is the domain, c represents programming language; 2 shows number of input arguments; 1 means number of declared return values; finally the name of the function. System creates a field and knowledge store entry to check the number of accesses to the calling function (this is updated on each call as hit) to monitor popularity in collaborative platform. Another field is created to monitor average memory and time requirement the given function consumes. A trailing number is added to the function name at the end of generated code that can be used to call the function when writing the rule so that the user do not have to type the whole URI to refer to that code. This comes as a overlay in the UI when function is used in a rule. Example: *resample2*, when two functions (may be expressed in different languages) already exist. This works like: function name + previous counter value + unit increment.

## V. Procedural Reasoning

While using Semantic Web technology for reasoning covers the description logic aspects, enabling procedure calls from bounded variables while reasoner is executing makes a reasoner powerful in handling procedural reasoning. Some example procedural reasoning scenarios along with corresponding logic expressions in rule syntax is shown with the help of a healthcare use case.

The task of 2016 PhysioNet / CinC Challenge[16] was to classify heart sound recordings (phonocardiogram / PCG) into normal and abnormal recordings. Main steps of the task are pre-processing, segmentation and modeling (classification). The following logic snippets show the approach's usefulness for pre-processing step for illustration purpose.

A section of domain knowledge used for the problem:
1. <problem:heartSound><sig:windowSizeInSeconds> 5
2. <problem:heartSound><sig:upperBoundFreqInHz> 400

A section of meta-data facts supplied for the problem:
1. <data:dataInstance><sig:samplingRateInHz> 2000
2. <data:dataInstance><file:format><file:wav>
3. <data:dataInstance><data:hasSource> 'file path'

A section of algorithmic knowledge stored in ontology:
1. <sig:MedianFilter><sig:type><sig:NonLinearFilter>
2. <sig:LowPassFilter><sig:type><sig:LinearFilter>
3. <sig:LinearFilter><sig:type><sig:Filter>
4. <task:PreProcessing><task:subTask><sig:Filter>

A section of procedural rules (custom define) used:
1. [ AutoResample: (?data <data:hasSource> ?url ) . (?data <problem:category> ?p) . (?data <sig:samplingRateInHz> ?val1) . (?data <sig:upperBoundFreqInHz> ?val2) -> c:resampleA(?url, ?val1, ?val2, ?urln) . (?p <task:completion>
<sig:AutoResample>) . (?data <data:newSource> ?urln) ]
2. [ LowPassFilter: (?data <data:hasSource> ?url ) . (?data <problem:category> ?p) . (?p <sig:upperBoundFreqInHz>

[16]https://www.physionet.org/challenge/2016/

?val) -> c:lowpass(?url, ?val, ?urln) . (?p <task:completion> <sig:LowPassFilter>) . (?data <data:newSource> ?urln) ]

The above procedural rule no. 1 attempts to resample data using a custom algorithm. Resampling data to a lower rate without missing patterns is recommended to enable faster processing. The auto resampling algorithm takes in sampling rate and the upper frequency range with in which meaningful frequencies exist and using principles of Nyquist Theorem and sampling [20], resamples data to an optimal sampling rate. The rule takes the data instance source, checks the problem tagging and sees if any facts about sampling rate and upper bound of frequency exists, calls the autoresampling algorithm (c: is the initial prefix notation to denote custom procedural calls) coded and registered by user and outputs completion fact and processed data source. Procedural rule no. 2 takes a data source, checks the problem tagging and sees if any domain knowledge of cutoff frequency exists, and next does a procedural call to low pass filter algorithm and gives out a link to processed data as well as adding a fact in working memory that lowpass filter operation is done. In this case, an available algorithm is used that is found in standard signal processing libraries of Matlab or Python. Hence it is showcased how symbolic and procedural knowledge based processing can be used for data analysis tasks.

## VI. Conclusion

The paper has presented a collaborative framework and system to carry out a large number of data processing tasks based on semantic web technology with support of procedural rules that is programming language agnostic. Example procedural rules were explained with the help of a healthcare use case. Future work will include extensions to 2-D (image) and 3-D (video) sensor data. Fast and incremental reasoning is a need of the system. Hence, to suit event-based IoT scenarios an improved Rete [19] [18] algorithm in the reasoning module is planned for integration. Finally, the system will be tested for different Analytics pipelines to gather further information on usability of procedural rules. The approach reuses other developer's efforts in semantic rule formation, thereby reducing development time.

## References


[1] P. Sethi and S. R. Sarangi, "Internet of things: architectures, protocols, and applications," *Journal of Electrical and Computer Engineering*, vol. 2017, 2017.

[2] N. Eddy, "Gartner: 21 billion iot devices to invade by 2020," *InformationWeek, Nov*, vol. 10, 2015.

[3] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, "Industry 4.0," *Business & Information Systems Engineering*, vol. 6, no. 4, pp. 239–242, 2014.

[4] M. S. Mahdavinejad, M. Rezvan, M. Barekatain, P. Adibi, P. Barnaghi, and A. P. Sheth, "Machine learning for internet of things data analysis: A survey," *Digital Communications and Networks*, 2017.

[5] M. Strohbach, H. Ziekow, V. Gazis, and N. Akiva, "Towards a big data analytics framework for iot and smart city applications," in *Modeling and processing for next-generation big-data technologies*. Springer, 2015, pp. 257–282.

[6] K. Hwang and M. Chen, *Big-data analytics for cloud, IoT and cognitive computing*. John Wiley & Sons, 2017.

[7] D. Pizzolli, G. Cossu, D. Santoro, L. Capra, C. Dupont, D. Charalampos, F. De Pellegrini, F. Antonelli, and S. Cretti, "Cloud4iot: A heterogeneous, distributed and autonomic cloud platform for the iot," in *2016 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*. IEEE, 2016, pp. 476–479.

[8] P. Barnaghi, W. Wang, C. Henson, and K. Taylor, "Semantics for the internet of things: early progress and back to the future," *International Journal on Semantic Web and Information Systems (IJSWIS)*, vol. 8, no. 1, pp. 1–21, 2012.

[9] H. Beck, "Reviewing justification-based truth maintenance systems from a logic programming perspective," Tech. Rep. INFSYS RR-1843-17-02, Institute of Information Systems, TU Vienna. July, Tech. Rep., 2017.

[10] R. S. Michalski, "Inferential theory of learning as a conceptual basis for multistrategy learning," *Machine Learning*, vol. 11, no. 2-3, pp. 111–151, 1993.

[11] P. Ristoski and H. Paulheim, "Semantic web in data mining and knowledge discovery: A comprehensive survey," *Web semantics: science, services and agents on the World Wide Web*, vol. 36, pp. 1–22, 2016.

[12] C. Esposito, M. Ficco, F. Palmieri, and A. Castiglione, "A knowledge-based platform for big data analytics based on publish/subscribe services and stream processing," *Knowledge-Based Systems*, vol. 79, pp. 3–17, 2015.

[13] P. Ristoski, "Exploiting semantic web knowledge graphs in data mining," Ph.D. dissertation, 2018.

[14] D. Dou, H. Wang, and H. Liu, "Semantic data mining: A survey of ontology-based approaches," in *Semantic Computing (ICSC), 2015*. IEEE, 2015, pp. 244–251.

[15] D. Perez-Rey, A. Anguita, and J. Crespo, "Ontodataclean: Ontology-based integration and preprocessing of distributed data," in *International Symposium on Biological and Medical Data Analysis*. Springer, 2006, pp. 262–272.

[16] S. Banerjee and D. Mukherjee, "Towards a universal notification system," in *Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)-Volume 03*. IEEE Computer Society, 2013, pp. 286–287.

[17] C. L. Forgy, "Rete: A fast algorithm for the many pattern/many object pattern match problem," in *Readings in Artificial Intelligence and Databases*. Elsevier, 1988, pp. 547–559.

[18] B. Berstel, "Extending the rete algorithm for event management," in *Temporal Representation and Reasoning, 2002. TIME 2002. Proceedings. Ninth International Symposium on*. IEEE, 2002, pp. 49–51.

[19] T. Gao, X. Qiu, and L. He, "Improved rete algorithm in context reasoning for web of things environments," in *Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing*. IEEE, 2013, pp. 1044–1049.

[20] E. J. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on information theory*, vol. 52, no. 2, pp. 489–509, 2006.

# Cost Efficient Automated Pisciculture Assistance System using Internet of things(IoT)

Mehboob Hasan Rohit
Department of Electrical and Computer Engineering
North South University
Dhaka,Bangladesh
mehboob.hasan@northsouth.edu

Zarin Tarannum Hoque
Department of Electrical and Computer Engineering
North South University
Dhaka,Bangladesh
zarinesterik@gmail.com

S M Mujibul Karim
Department of Electrical and Computer Engineering
North South University
Dhaka,Bangladesh
mujibul.karim@northsouth.edu

Dr.Shahnewaz Siddique
Department of Electrical and Computer Engineering
North South University
Dhaka,Bangladesh
shahnewaz.siddique@northsouth.edu

***Abstract*— The more the technology is getting advanced, the more it makes the life of people depending on it and one of that technologies is automation. Internet of Things (IoT) is another term which envisions every physical object that are being connected to the internet and being able to identify themselves to other devices. This paper illustrates a methodology to provide a low cost Automated Pisciculture Assistance System for indoor fish production using Wireless Fidelity (Wi-Fi). Despite being ranked third in the world in terms of inland fish production [1], Bangladesh is presently using the method of Recirculation Aquaculture Systems (RAS), because pollution of pond water is a major factor posing significant danger to hygiene issue for fish population inhabiting in pond water. In this research we have developed a complete assistance system which gives update to the user of the conditions of the water through the sensors and operate the device remotely. The key components of this system are a pocket-sized Wi-Fi module, Message Queuing Telemetry Transport (MQTT) for monitoring, controlling the sensors and alerting the user through SMS, an Android application to visualize the data provided by the module and to operate the device. Our main objective was to design a system to overcome the downsides with minimal costing and easy installation process.**

***Keywords*— *Internet of Things (IoT), Wireless Fidelity (Wi-Fi), Pisciculture Automation System, Wi-Fi module, Android, Sensors.***

## I. INTRODUCTION

Bangladesh is a low-lying riverine country located in southern Asia. About seven percent of the total area of Bangladesh is covered with rivers and inland water bodies [4]. The favorable geographic position of Bangladesh comes with a large number of aquatic species and provides plenty of resources to support fisheries potential. Fish is a popular complement to rice in the national diet, giving rise to the adage “a Bengali is made of fish and rice.” Not only a major part of the population lives on fish but also Bangladesh earns a considerable amount of foreign currencies by exporting fish and other fisheries products. There are, however, serious concerns surrounding the slow decline in the condition of open water fish stocks which have been negatively impacted upon through a series of natural and anthropogenic induced changes including large scale abstraction of water for irrigation and the construction of water barrages and dams, human activity resulting in the over exploitation of stocks, the unregulated introduction of exotic stocks and pollution from industry.

Now a day it has been very difficult to maintain a healthy water for fish production, for that people are now showing more interest to grow fish in a confined environment. One of the methods to grow fish in an indoor environment is Recirculating Aquaculture System (RAS) which rears fish at high densities, in indoor tanks with a controlled environment. Recirculating systems filter and clean the water for recycling back through fish culture tanks. New water is added to the tank only to make up for splash out and evaporation and for that used to flush waste materials. Fish grown in RAS must be supplied with all the conditions necessary to remain healthy and grow. They need a continuous supply of clean water at a temperature and dissolved oxygen content that is optimum for growth. Water pH need to be maintained in order not to be acidic water due to the excretion of fishes. The fish must be fed a nutritionally-complete feed on a daily basis to encourage fast growth and high survival. There are machines which already serves these necessities, but they are heavy weight and bulky in size and they are operated manually. It would have been much more comfortable if it could be both monitored and operated automatically.

Internet helps us to bring in with immediate solution for many problems and also able to connect from any of the remote places which contributes to overall cost reduction and energy consumption. IoT is a system that uses computer or mobile devices to control basic functions and features automatically through internet from anywhere around the world. With the introduction of IoTs, the research and development of automation are becoming popular in the recent days. Thus, the idea came up to make a unique system that can give update about the condition of the water in an Android application that would be sensed through sensors(temperature,pH,turbidity) and operate the appliances that are involved with fish production(oxygen pump,light,feeder,heater) remotely and as added bonus features ,scheduling the events was also included.

## II. PROBLEM STATEMENT

Problems related to the quality of water in pisciculture systems are very diverse. A careful balance of nutrients and other factors is essential to maintain the optimum culture conditions required for fish health. These factors include temperature, pH, turbidity, transparency, hardness, the balance between oxygen and carbon dioxide etc. In our system we have mainly focused on those parameters of which fish health is mostly dependent.

Temperature affects the water quality. The metabolic rate of fish increases as with the temperature. Therefore, temperature has a direct effect on important factors such as growth, oxygen demand, food requirements and food conversion efficiency. The higher the temperature, the greater the requirement for oxygen and food and the faster the growth rate. If the temperature is low, it can hamper the growth along with the health of the fishes by various diseases [6]. For that a standard temperature should be maintained as of mentioned in Table I.

Changes in pH also affects the fishes. Lower concentration of pH can accelerate the release of metals (copper and other heavy metals) from rocks and sediments which can affect the metabolism of the fish and its ability to take up water through gill [8]. Fishes are prone to attack of parasites and diseases in acidic waters. When pH rises over 11, the gills and lens and cornea of fish eyes are destroyed [7]. Due to waste discharge of the factories the required pH of the water gets reduced [3], the temperature of water fluctuates [2] which causes the death of fish. High pH also makes the toxic form of ammonia more prevalent.

TABLE I. Water quality standards for fish production [5]

| **Parameters** | **Acceptable Concentration** |
|---|---|
| DO | >5 mg/L |
| pH | 6.5-8.5 |
| Temperature | >20° C for warm water species<br>15-20° C for cool water species |
| COD | 20-30 mg/L |
| TSS | >80 mg/L |
| Nitrite | <0.02 |
| Nitrate | 0-100 |
| TAN | 0-0.2 |

Dissolved oxygen is considered as one of the important aspects in pisciculture. There is an inverse relationship of oxygen and temperature. At day time there is an increase of oxygen but at night there is a decrease of oxygen. It is needed by fish to respire and perform metabolic activities. Thus, low levels of dissolved oxygen are often linked to fish kill accidents. On the other hand, optimum levels can result to good growth, thus result to high production yield.

Fishes need to be fed a fix amount of food daily. The metabolism of fish is controlled by their surrounding temperature, the lower the temperature lower the digestive time. Also, most of the fishes do quite well on one feeding per day. So, if excessive of food is given to fishes, they may die of eating a lot at once.

Turbidity is a measure of how particles suspended in water affect water clarity. It is an important indicator of suspended sediment and erosion levels. Sometimes the water become muddy due to excessive rainfall or water becomes green because of way too much algae in water which affects the fish health.

The optimum fish production is totally dependent on the physical, chemical, and biological qualities of water to most of the extent. Hence successful pisciculture management require an understanding of water quality and to provide high quality water we have built this automated assistance system.

## III. EXISTING SYSTEM

The industry and the Academicians have worked together in perfect harmony to make great advances in the field of automation system. There are many technologies that are already involved with pisciculture but none of them are automated which requires more human effort. To monitor and operate the user needs to go in person which takes a lot of time as well. There already exists such components that can be used individually for that certain purpose. But they are not functionable in a single device which makes them costly in order to use in an indoor fish farm. Most of the systems are very bulky and heavy weight. In order to accommodate in an indoor environment, they require large space which sometimes becomes very hard to provide.

Use of IoT technology to reduce energy consumption is one of the challenging tasks because it becomes more hectic in busy life if user fails to turn off the appliances which may create the problem of loss of electricity. To achieve effective solution to these problem one such automation is required which allows the user to manage appliances remotely without their physical presence. Also, there are problems with automation as it faces the main problems costing, manageability, and security.

## IV. PROPOSED SYSTEM

In this paper we have introduced a system which is very light weight and easy to carry. It can be installed anywhere without having less complications as it is a compact device having four small sensors and a relay with which the other appliances are going to be connected that could be operated remotely.

A user can be updated about the conditions of the water anytime as it will be showing the temperature, pH, turbidity, and percentage of water damage in an android application and if required he/she can activate the switches of certain appliances of whose functionalities need to be performed by using the app.User can also schedule the events when he/she is away for a long time. If any abnormalities are seen in the water the user will be notified through an SMS.

## V.SYSTEM DESIGN AND IMPLEMENTATION DETAILS

### *A. Hardware Part*

In this system NodeMCU is at the center of the system which is based on ESP8266-12E Wi-Fi module shown in Fig. 1. It is a low cost and highly integrated chip that can be configured to connect to the Internet for Internet of Things (IoT) technology. The temperature sensor, turbidity sensor,4 channel relay is connected to it.

Fig 1. Node MCU

The temperature sensor DS18B20 is a one wire temperature sensor that measure temperature with a minimal amount of hardware and wiring. This sensor uses a digital protocol to send accurate temperature readings directly to the NodeMCU without the need of an analog to digital converter or other extra hardware. We have made this sensor waterproof by inserting it into a small waterproof stainless package just like Fig. 2

Fig 2. DS18B20

An analog pH meter is used here which has a built-in simple, convenient, and practical connection and features. It has an LED which works as the power indicator, a bayonet neill–concelman (BNC) connector and pH2.0 sensor interface just like Fig 3. To use the pH sensor, need to be connected with BNC connector and plug the pH2.0 interface into an analog input port. It needs to be calibrated whenever it is going to be used in a new environment.

Fig 3. pH sensor

We have used water turbidity sensor to detect water quality by measuring the levels of turbidity. It uses light to detect suspended particles in water by measuring the light transmittance and scattering rate, which changes with the amount of total suspended solids (TSS) in water. As the TSS increases, the liquid turbidity level increases.

Fig 4. Turbidity sensor

Here, 4 channel 5v opto isolated relay is used to connect oxygen pump, light, feeder, and heater in the system. These appliances will be activated whenever they will receive data from the NodeMCU through android application and also, they will be activated automatically if the events(on/off) of these appliances are scheduled.

Fig 5. 4 channel relays

To design an automated water pump we have used a 12v relay, a standalone microcontroller and liquid level probe. There are 3 wires in the liquid level probe. Among them, one is for detecting when the water tank is full, so that the water pump will be turned off automatically and other one is turning on the water pump whenever the water gets down of it as shown in Fig. 6

Fig 6. Automated water pump mechanism

### *B. Proposed IoT Architecture*

In this system NodeMCU is the main controlling unit which can connect with the smart phone application with the use of internet to control the peripheral devices along with getting the data from the sensors as shown in Fig. 7. With the help of internet, MQTT [9] is used to deliver the data from the NodeMCU to the android application and the website of it. It is used for setting the triggering conditions and controlling the appliances as well if any abnormalities happen in the water health, the user will be notified by an SMS. For customizing the messages, we have integrated the SMS service of Webhook applet of IFTT which supports the IoT based systems. [10]

The turbidity sensor is calibrated within the system with the help of MQTT in the range of 0-5v shown in Fig. 9. As shown in Fig. 8, the water turbidity has a relation with the voltage which is applied in our system to calibrate it.

To get analog pins, ADS1115, an analog to digital converter is used here as analog sensors (pH, turbidity) need to be connected to NodeMCU. Also, the analog pins of Node MCU serves 10-bit data and ADS1115 serves 16 bit-data for which it gives more accurate and efficient data. Thus, ADS1115 is used in the system for getting accurate data from the sensors. Also, a logic level shifter is used in the system as ADS1115 does serial communication and it is obvious the other device connected to it need to be operated in the same voltage. But NodeMCU can operate only in 3v but to serial communicate with ADS1115 it needs to be operated in 5v as well as to get the data from the sensors which is of 5v.

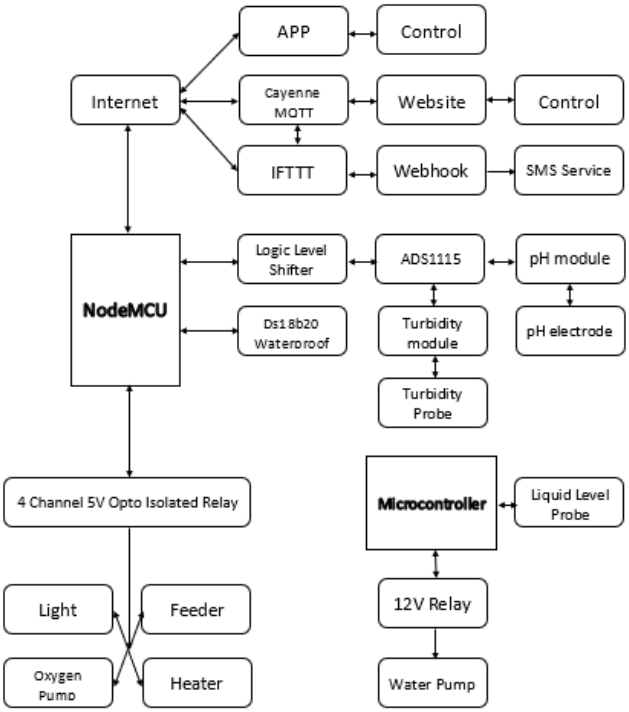



Fig 7. IoT architecture of the proposed system

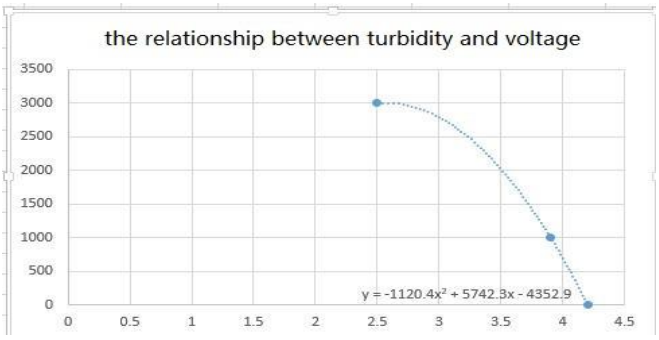



Fig 8. Relationship between turbidity and voltage [11]

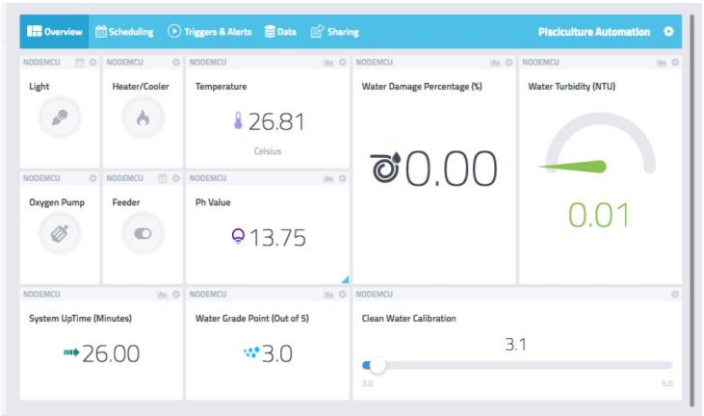



Fig 9. MQTT broker interface of the system

### *C. User Interface*

To operate the device remotely and to visualize the readings that we will be getting from the sensors in real time there is an app designed for the user. As shown in Fig. 10, in the first page of the app the user will be updated about the different parameters of the water e.g. temperature, pH, turbidity, water damage percentage through internet and in the second page of the app there are buttons of four appliances to turn on/off them remotely.

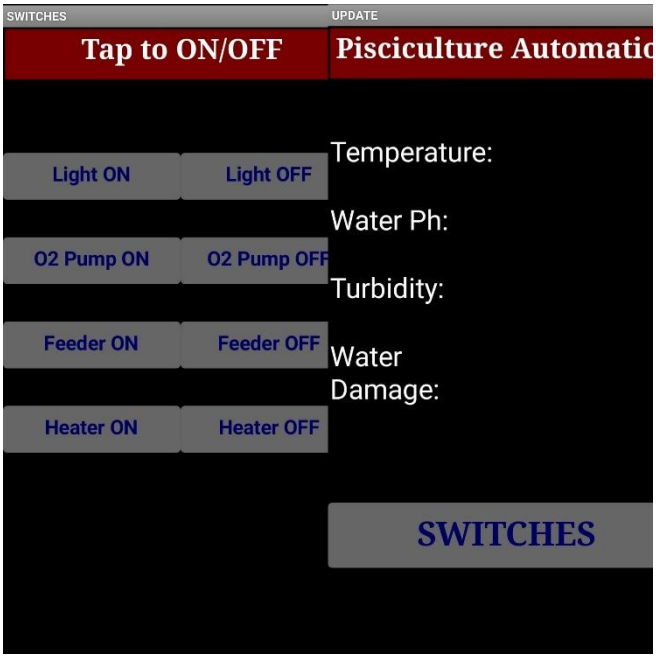



Fig 10. Android app for the system

## VI.RESULTS AND DISCUSSIONS

Main aim of the system was to make it cost efficient and small in size, so that anyone can afford it. Because most of the people who are involved in pisciculture are from poor family or middle-class family. In order to make it cost efficient as well as a compact device we have implemented the hardware part shown in Fig. 11, according to the design of the system, so that in can be installed anywhere very easily.

The pH sensor that we have used is an analog sensor which has pH module within which there are two pins-offset and limit. For every different type environment water this sensor needs to be calibrated by varying the limit pin.

We have tested our system against different type of environment water and satisfactory results were obtained as the system was built according to the plan. In the android application update of different parameters of the water were correctly displayed, shown in Fig. 12. The water pump worked very smoothly as per the need and also the appliances were operated remotely without having any difficulty. As all the outputs are shown in the smartphone instead of LED screen on dashboard which reduces the cost and makes the system installation easy.

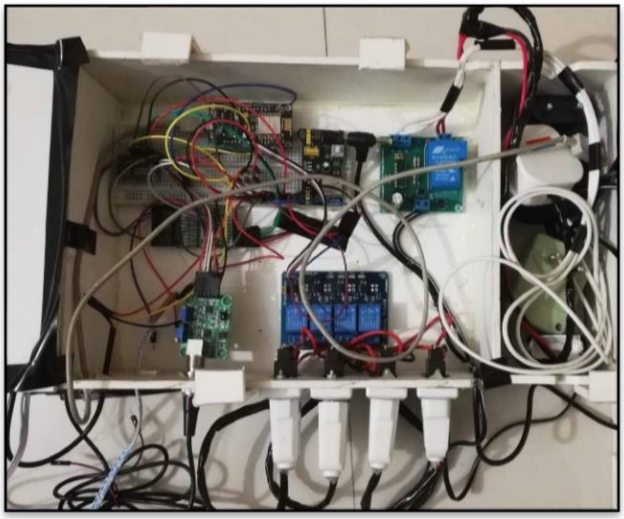


Fig11. Hardware implementation of the system

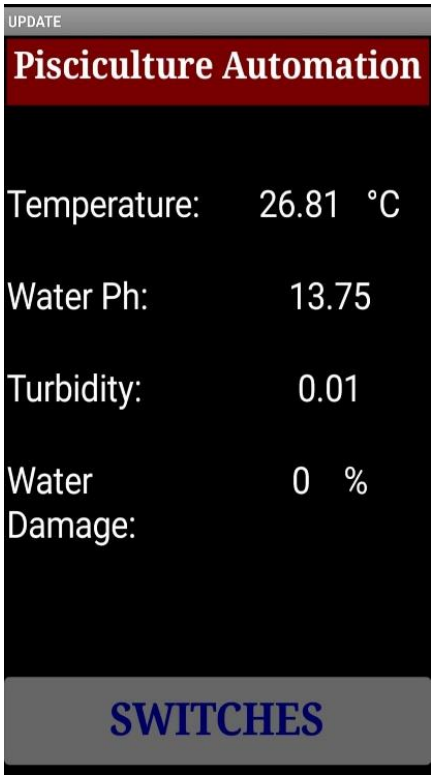



Fig12. Update of water health

The triggering conditions that we have set in the system which can detect any kind of abnormalities in the water and aware the user by sending an SMS was working satisfactorily, shown in Fig13.The system automatically takes necessary actions that we have already set in the system when it meet its triggering conditions.Also,the events can be scheduled rightly if the user is away for a long time, shown in Fig14.So,there is no option left that can harm the fishes anymore.

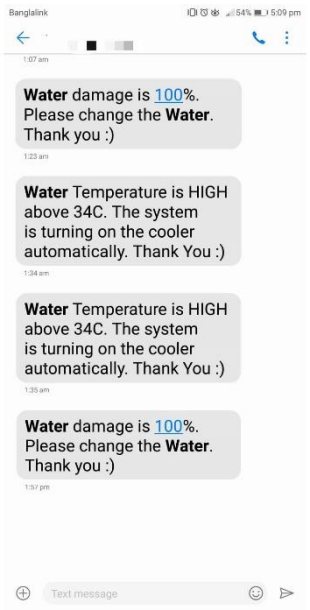



Fig13. Notification message received on user's cell phone

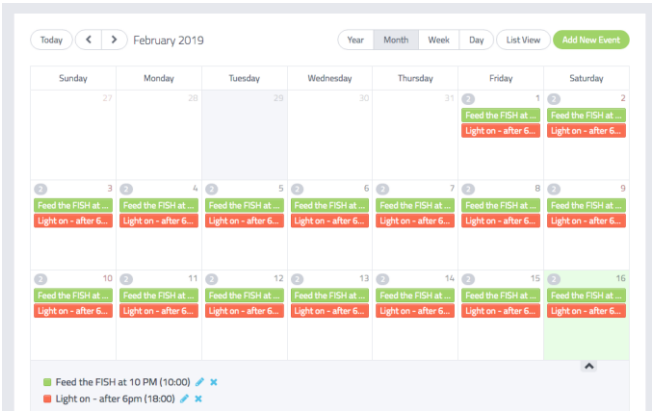



Fig14. Scheduled events.

This system can be used in outdoor fish farming as well by developing a bit. As it requires very less human engagement, it saves a lot of time as well as ensuring the proper growth and health of the fishes.

## VII.CONCLUSION

This system can significantly improve the overall pisciculture system. The usage of smartphones which are used by almost everyone nowadays due to lesser cost and some easily available devices makes the system affordable. The user-friendly application interface conveniently helps the

user to be in control of most situations by assisting him/her in every way. Our goal is to ensure healthy water for fishes altogether but even in case of unhealthy situations taking proper actions will become easier than ever.

## REFERENCES


[1] "FAO: Bangladesh ranks 3rd in inland fish production", Dhaka Tribune. Retrieve at July 18,2018.www.dhakatribune.com/feature/food/2018/07/18/fao-bangladesh-ranks-3rd-in-inland-fish-production

[2] Sudden Temperature Fall: Eye Disease Attacks Catla Fish in Bangladesh", 2013. http://en.bdfish.org/2013/10/sudden-temperature-fall-eye-disease-attacks-catla-fish-in-bangladesh/

[3] Havoc in Haors: Fish dying further down", The Daily Star. Retrieve at April 23,2017. www.thedailystar.net/frontpage/havoc-haors-fish-dying-further-down-1395001

[4] "Bangladesh", Irrigation in Southern and Eastern Asia in figures-AQUASTAT Survey,2011 www.fao.org/nr/water/aquastat/countries_regions/BGD/BGD-CP_eng.pdf

[5] Tasmiah Tabassom, Seminar paper on "Recirculating Aquaculture System", Bangabandhu Sheikh Mujibur Rahman Agricultural University,2018.

[6] Anita Bhatnagar, Pooja Devi," Water Quality Guideline for The Management of Pond Fish Culture", International Journal of Environmental Sciences, Volume 3, No. 6(2013)

[7] M. Delwar Hossain, M. Kabil Hossain, M. Habibur Rahman," Water Quality Parameters and Incidence of Fish Diseases in Some Water Bodies in Natore, Bangladesh", J. Life Earth Sci, Vol. 2(2) 27-30(2007)

[8] PHILIMANAQ," Water Quality Criteria and Standards for Freshwater and Marine Aquaculture", Annex 2.

[9] Cayenne MQTT. www.mydevices.com/for-hardware-manufacturers/iot-ready-program

[10] Webhook and IFTT. https://community.mydevices.com/t/webhook-and-ifttt/10182

[11] Turbidity sensor. www.dfrobot.com/wiki/index.php/Turbidity_sensor_SKU:_SEN0189

# Reliable IoT Systems for Improving Quality Of Life Through The Exploitation of Cloud, Mobile and BLE Low Energy Based Technologies. Case Study: Battery Charge Protect

***Abstract*—Connecting various smart objects within an intelligent ecosystem provides high capability of developing and integrating mobile, cloud and embedded device communications based on existing internet infrastructure. This has been the trend of Internet of Things (IoT), addressing health, quality of life, smart cities. In this paper we are a focused on similar aspects with a proper case study in an embedded system, used in protecting batteries of mobile devices. Cloud/Mobile applications interconnected with embedded devices shall be presented. The research undertaken present not just the technological innovation, exploiting state of the art technology, but also the security benefits and prolongation of battery life. The system has been fully developed, tested and it's benefits evaluated throughout the paper. We have been able to integrate and provide a real life case study adopting this integrated architecture to be reused for future implementations.**

***Index Terms*—Internet-of-Things, Smart Devices, Fall detection, Energy eciency, Wearable devices**

## I. Introduction

Internet of Things (IoT) concept and paradigm brings together existing internet infrastructure with everyday embedded devices such as smartphones or other micro-controllers in a common playground to improve people everyday life and provide smart innovative services. Information technology relying on Internet has widely grown during the past decades [1]. Thus all services and information are exchanged on World Wide Web (www). However a new concept arises with the need to provide interconnection not just between web browsers and cloud based servers / clusters with robust ISO protocols, but also between smart embedded devices. The latter might rely on independent protocols with respect to the ones used for interconnecting Web applications. The main goal of the IoT is to bring together both paradigms of the 21st Century to collaborate in creating larger grids of intelligent services and devices. Thus a mobile / cloud / embedded device interconnection through existing state of the art technologies shall be presented along this paper work. A development approach and methodology will be described and adopted to real case study addressing primarily health related issues. Thus a system monitoring the battery through a mobile application and BLE device is unfolded in the coming section. The main purpose of the Charge Protect Application is to control, predict and suggest the battery level of charge

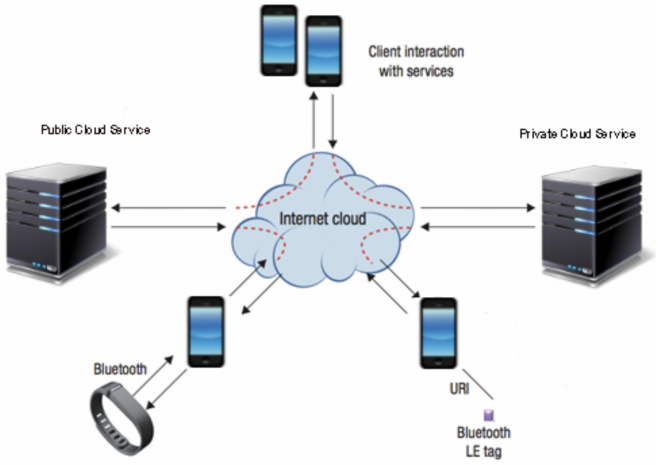



Fig. 1. Adopted architecture to interconnect the existing cloud infrastructure with embedded devices relying on Bluetooth Low Energy (BLE) communication protocol [1]

(SoC), providing not just useful tips but also security benefits which could incur from battery issues. After results have been gathered in the last section outcomes and challenges related to technological restrictions and decisions undertaken will be evaluated together with future proposals of improving existing approaches.

## II. Background

World Wide Web services are obviously the primary source and model of interconnection between services and information. Several existing embedded devices can become part, interact or controlled from existing web technologies. This would result into a combination between the previous two thus deriving the concept of Physical Web = Web Technology + IoT [2]. Figure 1 represents an overview of the possible physical web architecture to be adopted.

Existing concepts in web/cloud protocols will be extensively exploited along having a stable and robust infrastructure in storing data, sharing/retrieving information as well as guaranteeing high availability and scalability. Thus, URI (Unique Resource Identifiers) and RESTful API (REST standing for

Representational State Transfer) will be part of the system architectures. Moreover further protocols are required in order to fill the communication gap between smartphones and embedded devices relying on state of the art micro-controllers. One of the largely adopted technology is the one based on Nordic Semiconductors, which has a built-in BLE embedded inside its System on Chip (SoC) micro-controllers. Also the Bluetooth communication protocol relies on Generic Access Profile (GAP) and Generic Attribute Profile. GAP covers the usage model of the lower-level radio protocols to define roles, procedures, and modes that allow devices to broadcast data, discover devices, establish connections, manage connections, and negotiate security levels, GAP is, in essence, the topmost control layer of BLE. This profile is mandatory for all BLE devices, and all must comply with it. GATT deals with data exchange in BLE, GATT defines a basic data model and procedures to allow devices to discover, read, write, and push data elements between them. It is, in essence, the topmost data layer of BLE. Both protocols assure a stable communication of data and services with SmartPhones having BLE technology built-in. (Integrated from Bluetooth vs. 4.0). This has result the most promising standard starting from 2010.

## III. Literature Review and Motivation

The paper addresses not just concerns related to technology but also the health benefits related to the exploitation of BLE embedded devices. Many publications concerning UV (Ultraviolet) dosage intake have been a major concern related to skin health issues. One of the major risk being skin cancer [...]. While several articles during the past five years have been published related to development of embedded devices based on different micro-controller technologies operating in different settings including health benefits [3, 4, 5]. One of the most promising seems to be Nordic Semiconductors where BLE integration as a communication protocol has been made easy [6]. Several authors have presented the possibility of integrating into the Physical Web approach Cloud/Mobile and Embedded Devices, with high proficiency, security, availability and easiness of usage [7]. After analyzing charge current as function of charge Voltage. We made a simple adjustable charger circuit to analyze how different phones responded to different charge voltages. The test charger had adjustable output voltage from 4.0V to 5.3V. Max current 2A I had five different devices to test; HTC Desire, Samsung Galaxy S8, iPhone 6, iPad 2, iPhone 7). Had to spend some time to analyze how the Phones could detect the max current the different charges could deliver. Found that USB data lines D- and D+ are used to tell the charged device how many Ampere they can deliver. There are different standards (Note that iPhone do not follow these standards, but have made their own) how the phone can identify the chargers max charge current by using resistor networks in the charger to set different voltages on D- and D+ signals. See below for some different D+/D- configurations. Samsung used shorted D- to D+ and voltage set to 1.2V for 1A charger and Logic Low (must test) for 2A charger. iPhone has D+=2V and D-=2.7V for 1A and D+=2.7V and D-= 2.0V for 2A charger: Each phone was connected to the charger with correct D-/D+ voltage for max current charge and charge current was recorded for each 100mV starting at 5.3V and down to 4.0V. Table I. shows the Charge Current as function of Charge Voltage, while Figure 2 shows the corresponding plotting.

TABLE I
Component-based software engineering of the system

| | Phones and init charge level [%] | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 20 % | 80 % | 51 % | 24 % | 16 % | 54 % | 16 % | 74 % |
| Voltage | HTC Desire [mA] | Galaxy S3 [mA] | Galaxy S5-1 [mA] | Galaxy S5-2 [mA] | Galaxy S6 [mA] | iPhone 4 [mA] | iPhone S6 [mA] | iPad [mA] |
| 5,30 | | | | | | 970 | 1020 | 1530 |
| 5,20 | 904 | 950 | 1256 | 950 | 1193 | 895 | 1020 | 1412 |
| 5,10 | 899 | 942 | 1193 | 950 | 1193 | 828 | 1018 | 1300 |
| 5,00 | 890 | 670 | 1045 | 890 | 1196 | 758 | 1021 | 1185 |
| 4,90 | 870 | 484 | 947 | 754 | 899 | 690 | 908 | 1048 |
| 4,80 | 811 | 483 | 753 | 650 | 714 | 626 | 814 | 903 |
| 4,70 | 733 | 299 | 558 | 521 | 510 | 551 | 706 | 762 |
| 4,60 | 650 | 116 | 459 | 420 | 413 | 418 | 606 | 620 |
| 4,50 | 576 | 113 | 313 | 310 | 314 | 257 | 504 | 501 |
| 4,40 | 518 | 111 | 202 | 182 | 180 | 0 | 418 | 381 |
| 4,30 | 448 | 0 | 0 | 18 | 0 | 0 | 339 | 362 |
| 4,20 | 380 | 0 | 0 | 0 | 0 | 0 | 209 | 328 |
| 4,10 | 303 | 0 | 0 | 0 | 0 | 0 | 166 | 50 |
| 4,00 | 223 | 0 | 0 | 0 | 0 | 0 | 88 | 50 |

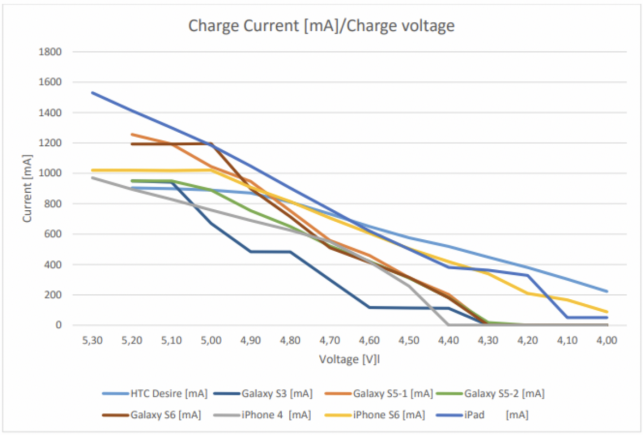



Fig. 2. Plot showing charge current as function of Charge VoltagePlot showing charge current as function of Charge Voltage

Initially we had an idea using a current limiter for low current charging for Long life mode, but after analyzing the charge current, I think the most optimal design will be to set low charge to 4.6V, medium charge to 4.8V and 5V for max current (what the charger can deliver). 4.6V will set max current to 600mA and a linear LDO regulator could be used instead of a Buck step down DC-DC converter. LDO are cheaper, fewer components and we will eliminate EMI emission issues. Maximum power loss using LDO will be 0.4V x 0.6A= 0.24W. Well below what could cause Dongle thermal issues.

### A. Charge Current Detection Coding

Device makers being a competitive lot, theres no one standard negotiation its a bit like having to speak six languages. While the USB forum released a generic signature standard, other manufacturers came up with their own signatures and in the end, there are at least six D+/D- signatures in the wild:

- 2.0V/2.0V low power (500mA)
- 2.0V/2.7V Apple iPhone (1000mA/5-watt)
- 2.7V/2.0V Apple iPad (2100mA/10-watt)
- 2.7V/2.7V 12-watt (2400mA, possibly used by Blackberry)
- D+/D- shorted together USB-IF BC 1.2 standard
- 1.2V/1.2V Samsung devices

These days, resistance-based voltage sensing options like the first four are described as legacy modes and all new devices we believe use chip-based detection.

In the following sections we will describe the methodology and the case study to address the problem.

### B. Methodology

Development of an integrated environment of Cloud / Mobile and Embedded wearable devices consists in the following steps:

1) Development of the Wearable Device (Dongle)
2) Firmware Development providing the communication protocol between the mobile
3) Mobile App Development (Android / iOS)
4) Google Cloud Communication

The system is general and abstract set of hardware and software making use of the common software engineering concept CBSE (Component-based software engineering) shown in Figure 3. The benefits of such an approach is that each component is reusable and highly configurable. The system architecture can be described as an onion with a core containing the abstract and general building blocks while more outer layers build upon the inner layers becomes more and more concrete and specific the further out in the onions layers.

The device micro-controller is based on Nordic Semiconductor technology with nRF51 Development Kit, from which a firmware development in C programming language is made possible. Charge Guard is basically a charge limiter for your phone battery. The intended purpose of the product is to limit the charge level of the phone battery to a certain level in order to prolong battery lifetime. The reason for why this works is that the higher the voltage of the Lithium-ion battery gets; the more wear the cells will be exposed to. By preventing the cell voltage of getting to its maximum charged level the wear on the battery cells are reduced and the battery lifetime will be prolonged. Figure 4 shows the intercommunication intended between the Charger/Dongle/SmartPhone.

A Client / Server (C/S) Architecture is adopted for broadcasting services and exchanging data between the Mobile Application and the Wearable Device. Firmware acts primarily as the Server while the mobile application developed in Android/iOS etc. shall act as a client communicating with

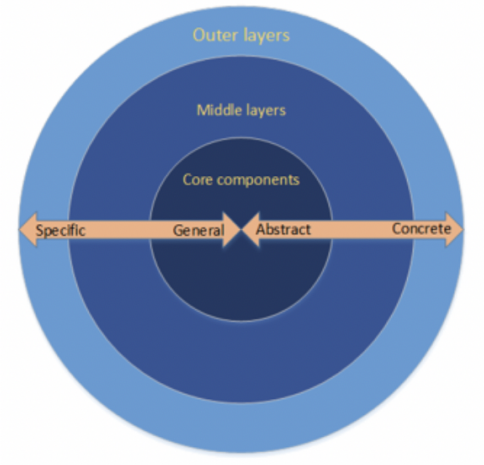



Fig. 3. Component-based software engineering of the system

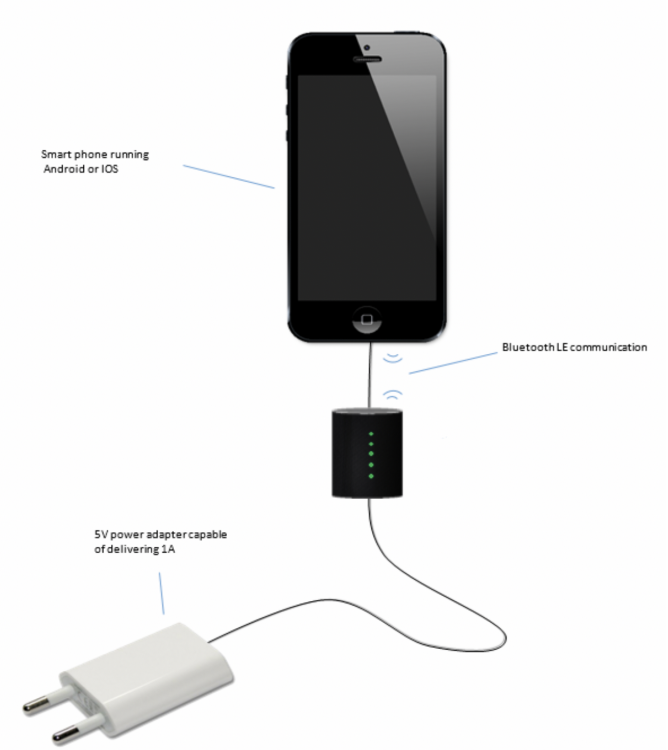



Fig. 4. Dongle interconnection between charger and device

the services offered from the server based on the GAP/GATT protocol. Further status of the current charge is updated on the Google Cloud from the mobile application. The overall architecture is described in Figure 5.

## IV. Case Study

Charge Guard is basically a charge limiter for your phone battery. The intended purpose of the product is to limit the charge level of the phone battery to a certain level in order to prolong battery lifetime. The reason for why this works is

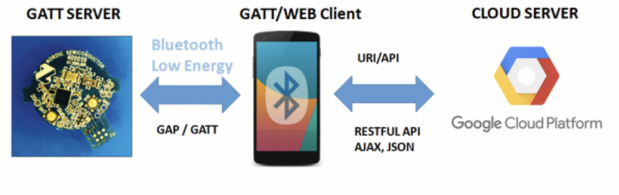



Fig. 5. IoT System Architecture

that the higher the voltage of the Lithium-ion battery gets; the more wear the cells will be exposed to. By preventing the cell voltage of getting to its maximum charged level the wear on the battery cells are reduced and the battery lifetime will be prolonged.

It will also give user an alarm when battery reach the predefined low battery level to start charging. Based on user setting the Charge Guard will also enable fast (high current charge) or slow (low current charge). Slow charge will charge with a lower current and prevent Battery from reaching high temperature that also can shorten the battery life. The last function is to monitor Battery temperature and charge current. It shall also activate multiple alarms (Vibration, sound, light) for user to act in order to avoid any hazard and turn off the charge current by internal switches. There are two main domains within the charge guard domain; The phone app and the charge limiting dongle itself. The respective features of each of them is described in the sections below.

### A. *HW Functionality Description*

The CG unit is a Battery charging protection device designed to prevent overcharging batteries in order to extend the battery lifetime. The CG is controlled by an App installed on the unit to be charged. The CG/App has several built in functions:

1) Fast or slow charge mode. The App has a user selectable charge speed (slow/fast) switch.
2) Battery under/over charge mode. The App has user defined lower and upper charge limits. Low setting will initiate start charge message to the user. Upper charge limit is defined by user and will turn of power to the Phone/PAD when limit is reached. A predefined default setting is set, but can easily be adjusted by user.
3) Charger Emulation. The CG will read charger signature set on USB d-/d+ lines and convert the levels to match the Phone/Pad brand and model for optimal charge current. This function will enable phones of different brands to use charger from other brands without risk for damaging charger or phone/pad.
4) Safety function will monitor voltages from charger and to phone in addition to charge current and temperature. If any of these are outside defined levels the Device will immediately abort the charging sequence and activate multiple alarms as visual, audible and vibration.
5) Allow normal charging current provided by manufacturer and charger specification.

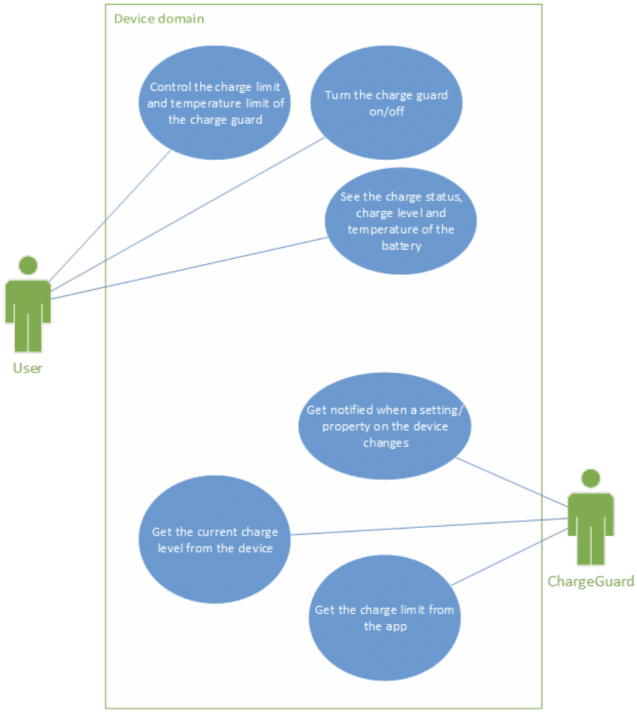



Fig. 6. System usability between the Dongle and the User

6) Allow low/medium charge current by setting charge voltage to 4.6V or 4.8V
7) Monitor Dongle temperature, charge voltage and current.
8) Send Alarm message if Dongle temp reach 60 oC
9) Emulate other brand chargers
10) The Charge guard shall be connected to the Charger and with cable to the device/phone/Pad.
11) Connectors USB-A is a universal solution

The usability of the system is represented in Figure 6

The dongle operates based on the implementation of a firmware dedicated exclusively to it. While based on the architecture presented in the methodology we implemented the mobile application in Android. The standard interface is through USB-A cable. While communication interfaces are between the charging dongle and the phone. BLE communication is established between dongle/Device pairing by Dongle Output Voltage cycling through a sequence changing output voltage between 4.6 4.8 5V to confirm correct dongle connected to the Device. If output voltage does not change according the programmed sequence the Paring must repeat this HW handshake for each dongle until match is detected. The Dongle includes programmable HW voltage design to supply 0V, 4.6V, 4.8V and 5V (the Charger output voltage) to the Charging Device (Phone/Pad or eq). It also monitors Charger Voltage and Voltage fed to Phone to prevent any Hazards by turning off output Voltage when abnormal voltage/current is detected. Current measurement is added to the Output voltage for safety and user information. The Current sense must read current up to min 3A. Simulation in Fig. shows Current Sense

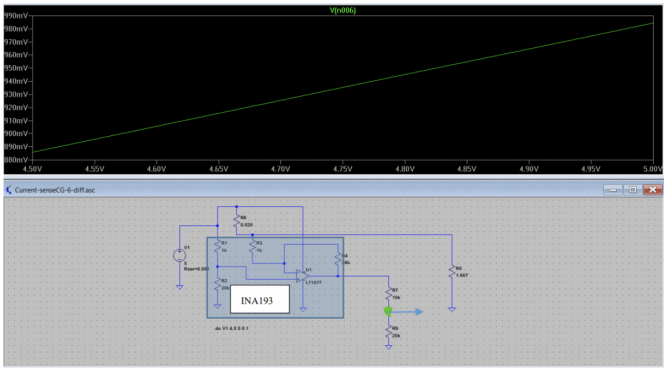



Fig. 7. Simulations for Current Sense design using INA193-198 Current sense Monitor for High side current sense

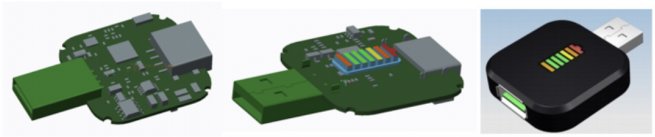


Fig. 8. Hardware industrial design of the dongle

design using INA193-198 Current sense Monitor for High side current sense. Select INA193/196 with Gain=20 and divide by 10k/20k to give 3A=1V output to nRF ADC, Figure 7

### *B. HW Detail Design*

The hardware industrial PCB design of the dongle is represented in Figure 8 where physical dimensions are:

1) Width: 35 mm
2) Depth: 60 mm (including connectors)
3) Height: 20 mm

The Dongle has LED based color coded charge level. The Tri-LED is also used for illumination (Alarm flashing and low ambient light lamp). Six LED shall be included for Battery status level to user covering the range from 0 to 100%. The Battery indicator have the color coding as in Table II.

While interfaces of communication between the charger and Dongle are shown in Figure 9

The Table III represents the electrical parameters for the connectors.

### *C. HW/FW functional description*

Table IV shows the functional dependencies between Hardware/Firmware developed for the dongle, while communication requirements are described in Table V

Development of the firmware is based on Nordic Semiconductors with nRF51 development kit [reference] based on C programming language. In Figure 10 can be observed the programming board interconnected to the dongle prototype (to the right) so that the developed firmware can be deployed on it. The firmware will execute on a microcontroller in an embedded system without the use of an operating system. The microcontroller chosen for the project is a Texas Instruments (CC2564) SOC (System on Chip) BLE enabled processor.

After being programmed the dongle allows for charge voltage controlled programming as described in Table VI

TABLE II
ELECTRICAL INTERFACES FOR CONNECTORS

| Charge Level [%] | LED Color |
|---|---|
| 100 | RED |
| 80 | GREEN |
| 60 | ORANGE |
| 40 | GREEN |
| 20 | GREEN |
| 0 | RED |

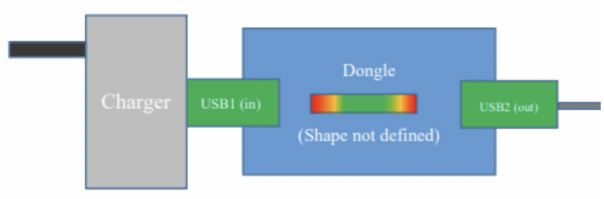
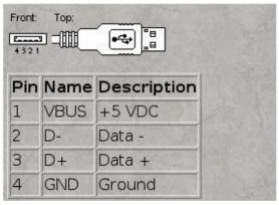



Fig. 9. Communication interfaces between Dongle and Charger

TABLE III
ELECTRICAL INTERFACES FOR CONNECTORS

| Connector Description | Des | Type | Comments |
|---|---|---|---|
| Programming Connector | J1 | 10 pin 1.27mm | Dual row |
| USB Charger side | USB1 | USB-A Male | Input |
| USB Phone/Tablet Side | USB2 | USB-A Female | Output |




| Pin | Name | Description |
|---|---|---|
| 1 | VBUS | +5 VDC |
| 2 | D- | Data - |
| 3 | D+ | Data + |
| 4 | GND | Ground |

### *D. Mobile and Cloud Functionality Description*

Using the phone app user interface the user must be able to set a level of charge that the phone should alert the user

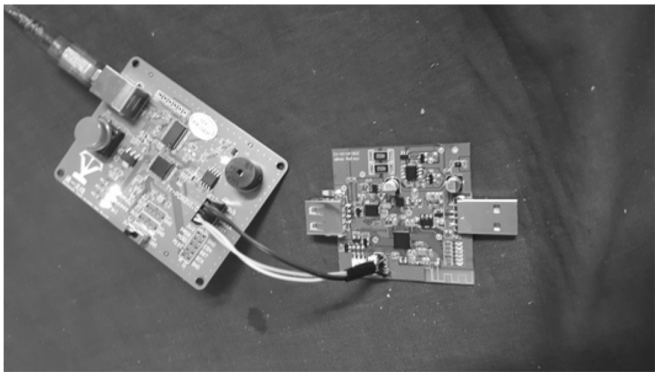


Fig. 10. Programming the device with Nordic Semiconductor Development Kit

TABLE IV
HW/FW DEPENDENCIES

| Signal | Function | U1 - pin | I/O | Signal level active | Comments |
|---|---|---|---|---|---|
| T1 | Turn on Max Voltage | 28 | O | High | Set output voltage to charger output voltage |
| T2 | Turn on 4.6V output Voltage | 21 | O | Low | |
| T2 | Turn on 4.8V output Voltage | 21 | O | High | |
| ENABLE | LDO Enable signal | 22 | O | High | |
| LED1 | LED RD or Tri Color | 43 | O-PWM | Low | LED not defined |
| LED2 | LED OR or Tri Color | 44 | O-PWM | Low | LED not defined |
| LED3 | LED OR or Tri Color | 45 | O-PWM | Low | LED not defined |
| LED4 | LED OR or Tri Color | 46 | O-PWM | Low | LED not defined |
| LED5 | LED GN or Tri Color | 47 | O-PWM | Low | LED not defined |
| LED6 | LED GN or Tri Color | 48 | O-PWM | Low | LED not defined |
| LED9 | Tri Color LED | 18, 19, 20 | O-PWM | High | Transistor inverter |
| PWM d- | Opamp | 15 | O | PWM | |
| PWM d+ | Opamp | 16 | O | PWM | |
| MUX Addr.A | HCT4053 | 25 | O | X,X,X | Binary Addressing |
| MUX Addr B | HCT4053 | 26 | O | X,X,X | Binary Addressing |
| MUX Addr C | HCT4053 | 27 | O | X,X,X | Binary Addressing |
| Current Sense | Read Current to Phone/Tablet | 7 | I | A/D 0-1.2V | Analogue input |
| Uin | Input Voltage | 9 | I | A/D 0-1.2V | Analogue input |
| Uout | Output Voltage | 8 | I | A/D 0-1.2V | Analogue input |
| USB D- | Input to nRF for Charger ID reading | 6 | I/O | A/A 0-5V | Used to decode charger and emulate diff types |
| USB D+ | Input to nRF for Charger ID reading | 5 | I/O | A/A 0-5V | Used to decode charger and emulate diff types |
| SWDIO | Programming Data In | 23 | I | Data | Flash programming |
| SWDCLK | Programming Clock | 24 | I | Clk | Flash programming |
| USB Shield GND | USB Shield not connected with signal GND | | - | | Follow USB spec for PCB design. Connect USB1 Shield to USB2 Shield only |
| X-TAL | | 36,37 | I/O | | MCU clock |
| BLE antenna | | 30-32 | I/O | | |

of when it is reached. When connected to a Charge Guard charging dongle the charge limit should be reflected in the hardware dongle.

The app must continuously monitor the level of charge in order to check whether the charge level is reached triggering a phone alarm/notification. If the app is connected to a Charge Guard dongle must be notified on order to stop the charging by cutting the power to the device.

The app will let the user determine a limit for max charging current and temperature limit (only in cases where the phone operating system supports retrieving it).

So in summary the mobile application associated to the system serves following purposes:

- Limit the charge to a certain user specified level
- Battery level monitoring
- Let the user specify the max charge limit
- Present a user friendly User Interface (UI)

The use cases representing the system usability are represented in Figure 11

The operating system requirements for the devices are:

- Android Minimum supported version greater or equal 4.1.x API version 16
- GATT server must run in background

The mobile application interfaces are represented in Figure 12

While Google Cloud Application is running on Django framework 1.11 and Python 2.7. The purpose of the cloud system is to keep track of the battery status sent from the

TABLE V
HW/FW COMMUNICATION

| *Function* | *APP SW* | *Dongle FW* | *Signal levels* | *A/D conv. input* | *Comment* |
|---|---|---|---|---|---|
| BLE Pairing | Send Uout Sequence to CG and pair when HW responds to request | Toggle Uout as defined in order to identify connected device until match. | 4.6 to 5VDC | | |
| Charger Signature ID (d+/d-) | Send Phone ID to CG to set correct Charger signature | Read Charger d-/d+ voltage and set correct d-/d+ to match charger and phone current | 0 to 5VDC | 5V=1.2V to nRF (divide input to 1.2V=2FF) | Probably risk to add this but seems to could work. Must be tested |
| Uout Level | Send Uout CMD according user requirement setting or Power monitoring | Set T1, T2 and Enable According Table 5.3.1.9 | L/H | | |
| Uin/Uout | Request Uin/Uout value | Reply UI/Uout value when requested by App and update display every 10 sec. Turn off charger if Values are outside spec. | 0 - 5V | 200mV/V 6V=2FF (Res divider 6V to 1.2V) | |
| Iout | Request Iout | Reply Iout when requested by App and update display every 10 sec. | 0 - 3A | 1V=3A Vref 1.2V 3.6A=2FF | |
| Temp | Request Dongle internal Temperature | Send Temp Alarm to App when required. App must turn off charging current and display Alarm message | Use nRF51822 internal temp sensor | | |
| LED – 1-6 | Send Battery level to turn LED on corresponding battery charge level from 0 – 100% | Activate LED as defined by App. See table 5.3.1.6 | L=Active | | |
| Alarm/Ambient LED | Send Alarm LED color and intensity | FW set LED as defined by App | H=Active | | Optional |
| | | | | | |

TABLE VI
CHARGE VOLTAGE CONTROL PROGRAMMING

| Charge Current | U out [V] | T1 | T2 | EN | Comments |
|---|---|---|---|---|---|
| Off | 0 | 0 | 0 | 0 | Turn off Q1 and LDO |
| Low Charge current | 4.6 | 0 | 0 | 1 | Select Resistor for 4.6V output |
| Med Charge current | 4.8 | 0 | 1 | 1 | Select Resistor for 4.8V output |
| Max Charge current | 5.0 | 1 | 0 | 0 | Turn on Q1 for max voltage and disable LDO |
| | | | | | |

mobile application described in the next section. Figure 13 shows the deployment diagram of the cloud application.

The cloud infrastructure offers a reliable way of developing storing and sharing information for different users. It also offers more cost effective, and other benefits such as security or privacy, with little latency drawback. Thus providing the possibility of a personalized battery charging and protection plan. The end user also has little or no knowledge at all about the system behind the scenes but its offered a robust yet easy to use and comprehensive battery protection for their devices.

### *E. System architecture and communication interfaces*

In this section we describe the communication interfaces between the mobile application and the dongle, relying on Bluetooth low energy technology, Figure 14

The GATT (Generic Attribute Profile) service is the core of the communication interface between the hardware dongle and the app. It defines the characteristics and attributes needed

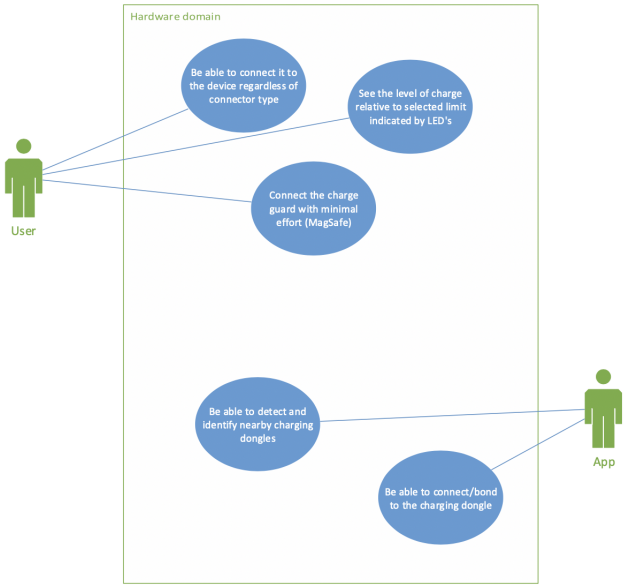



Fig. 11. System usability between the Mobile Application and the User


Fig. 12. Mobile Application UI

in order to transfer data between the phone and the charging dongle. The Universal Unique Identifier (UUID) types in this GATT service table only contains the last 16 bytes of the entire UUID type identifier. The company identifier given by the registration in Bluetooth SIG must be prepended in the application in order to correctly identify the service, characteristic and attributes of this service. The characteristics of the service are:

**Level of charge**

The level of charge represents the charge level of the phone battery represented as a percentage value between 0 and 100 (integer). The phone gets this value through the APIs presented by the respective operating system. This value is only readable

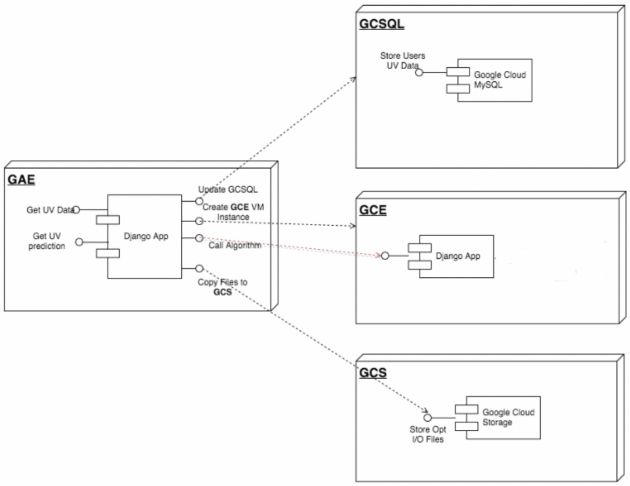



Fig. 13. Google cloud application deployment diagram

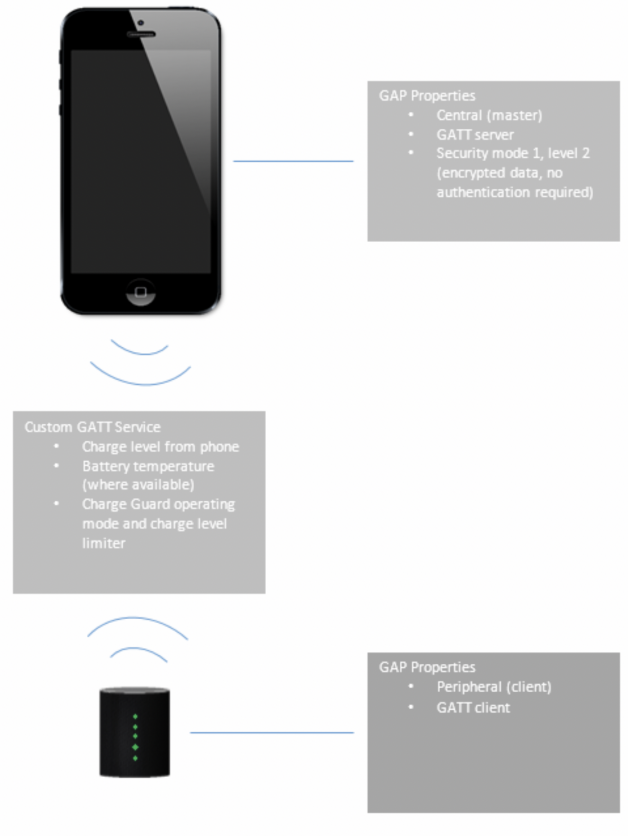



Fig. 14. Communication Interfaces

by the charging dongle (GATT client).

**Battery temp**

The battery temp represents the current battery temperature of the phone represented as degrees of Celsius scale as an integer. The phones that support this feature will provide the

TABLE VII
GATT SERVICE TABLE

| Name (Service 1) | Handle | UUID Type | Permissions | Value | Lenght |
|---|---|---|---|---|---|
| Service | 0x0090 | 0x2800 (Primary service) | Read | 0x1234 | 2 bytes |
| **Name (Characteristic1)** | **Handle** | **UUID Type** | **Permissions** | **Value** | **Value lenght** |
| Level of charge (declaration attribute) | 0x0091 | 0x2803 | READ | 0x0092 | 2 bytes |
| Level of charge (value attribute) | 0x0092 | 0x0111 | READ\|NONE | Actual value as double | 8 bytes |
| Level of charge (descriptor) | 0x0093 | 0x2902 | READ WRITE | 0x0001\| 0x0000 | 2 bytes |
| **Name (Characteristic 2)** | **Handle** | **UUID Type** | **Permissions** | **Value** | **Value lenght** |
| Battery temp (declaration attribute) | 0x0094 | 0x2803 | READ | 0x0095 | 2 bytes |
| Battery temp (value attribute) | 0x0095 | 0x0112 | READ\|NONE | Actual value as double | 8 bytes |
| Battery temp (descriptor) | 0x0096 | 0x2902 | READ WRITE | 0x0001\| 0x0000 | 2 bytes |
| **Name (Characteristic 2)** | **Handle** | **UUID Type** | **Permissions** | **Value** | **Value lenght** |
| Operation Mode (declaration attribute) | 0x0097 | 0x2803 | READ | 0x0098 | 2 bytes |
| Operation Mode (value attribute) | 0x0098 | 0x0113 | READ\|NONE | Charge Threshold 0-8 byte 9nth byte Bitfield matrix | 9 bytes |
| Operation Mode (descriptor) | 0x0099 | 0x2902 | READ WRITE | 0x0001\| 0x0000 | 2 bytes |

battery temp through the phones API. If the phones API doesnt support retrieval of battery temperature the value should be set to 10 000 so that the charging dongle can identify that the value is not provided and can be overseen.

**Operation mode**

The operation mode characteristic value is a multipurpose field which represents the following information:

Description is performed MSB:

- Byte 0: Charge threshold (uint8)
- Byte 1: Temp threshold (uint8)
- Byte 2: Current limit (boolean)
- Byte 3-9: Unassigned for possible future use

While the GATT table offering the services of communication among the mobile application and the dongle are presented in Table VII

## V. Conclusions

In this paper we have shown the possibility of combining Cloud / Mobile / Wearable devices into IoT grid offering safety benefits to the end users while charging their mobile phones. Battery life prolongation has been discussed. The system can have two operational modes, connected or not with the dongle, but also a well-structured synchronization logic. Main benefits are related to easiness of use, robust, reliable, secure and scalable system, yet with very little latency concerns. However still practical issues remain to be addressed such as device proximity awareness, pairing and handshake protocols, centralized vs. peer to peer (p2p) approach, low latency and real time concerns and integration of devices with very low computational power. A possible continuity could be evaluating the challenges of pairing a large set of devices within small areas.

## References


[1] Roy Want, Bill N Schilit, and Scott Jenson. “Enabling the internet of things”. In: *Computer* 1 (2015), pp. 28–35.

[2] Stephen S Yau and Arun Balaji Buduru. “Intelligent planning for developing mobile IoT applications using cloud systems”. In: *2014 IEEE International Conference on Mobile Services*. IEEE. 2014, pp. 55–62.

[3] Hannu Sillanpää et al. “Integration of inkjet and RF SoC technologies to fabricate wireless physiological monitoring system”. In: *Proceedings of the 5th Electronics System-integration Technology Conference (ESTC)*. IEEE. 2014, pp. 1–5.

[4] Muthuraman Thangaraj, Pichaiah Punitha Ponmalar, and Subramanian Anuradha. “Internet of things (iot) enabled smart autonomous hospital management system-a real world health care use case with the technology drivers”. In: *2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*. IEEE. 2015, pp. 1–8.

[5] Geng Yang et al. “A health-IoT platform based on the integration of intelligent packaging, unobtrusive bio-sensor, and intelligent medicine box”. In: *IEEE transactions on industrial informatics* 10.4 (2014), pp. 2180–2191.

[6] *Nordic Semiconductors:* https://www.nordicsemi.com/. 2019.

[7] Nan Chen, Xiaodan Li, and Ralph Deters. “Collaboration & mobile cloud-computing: using CoAP to enable resource-sharing between clouds of mobile devices”. In: *2015 IEEE Conference on Collaboration and Internet Computing (CIC)*. IEEE. 2015, pp. 119–124.

# A New Dynamic Smart-AC Model Methodology to Enforce Acess Control Policy in IoT layers

Nadine Kashmar*, Mehdi Adda
Département de Mathématiques, Informatique et Génie
Université du Québec à Rimouski
Rimouski, Québec
kasn0002@uqar.ca*, mehdi_adda@uqar.ca

Mirna Atieh
Business Computer Department, Faculty of Economic Sciences and Administration
Lebanese University
Hadat, Lebanon
matieh@ul.edu.lb

Hussein Ibrahim
Institut Technologique de Maintenance Industrielle (ITMI)
Sept-Îles, Québec
hussein.ibrahim@itmi.ca

***Abstract*—Internet of Things (IoT) is the conversion of everyday tangible devices or machines to smart objects. This means that these objects would be able to think, sense and feel. For example, your home devices will be able to detect and feel your absence to turn off the lights of empty rooms, close doors, lock the gates, and other tasks. Thus, would it be acceptable to find intruders who might mess up your daily life style or control your home appliances? Absolutely not! The same idea for factories, they definitely reject to detect any unacceptable access from any foreigner to their logical/physical assets or machines who might be able to locally or remotely control, for example, any machine operation. This would cause a significant loss for their reputation or investments, since any vulnerability or attack can produce, for example, fault products. So far, IoT is considered as one of the most essential areas of future technologies, especially for the industries. Hence, finding an environment full of smart devices needs a smart security methodology to prevent any illegal access. In this domain, various researches are conducted to find Access Control (AC) models to enforce security policies that prevent any unauthorized detection of sensitive data and enable secure access of information. For this purpose, we present a new dynamic Smart-AC model methodology to enforce security policy in IoT layers.**

***Keywords-IoT; access control; smart; security; model; policy***

## I. INTRODUCTION

In literature various Internet of Things (IoT) definitions are provided due to the integration of different technologies. The significant amount of IoT definitions can be summarized as a huge number of objects and devices with different technologies and platforms that are connected to the internet via heterogeneous networks (3G, LTE, WiFi, ZigBee …). Figure 1 illustrates this definition. It is the integration of several technologies, such as wired and wireless sensor networks, identification and tracking technologies, communication protocols shared with the next generation internet, and distributed intelligence for smart objects [1].

The big question in IoT is how to allow a variety of objects, such as PCs, mobile phones, sensors, etc. to interact and cooperate with each other transparently and securely to attain certain tasks related to consumers, companies or industry sectors. In this domain, the main concern for them is to keep their zone of interconnected devices and which are connected to the internet, secure, private and controlled only by them. Thus, finding an environment full of smart devices needs a smart security methodology to prevent any illegal access and enforce policy and security requirements. In this paper, we tackle IoT security and Access Control (AC) related issues and present a new methodology to enforce AC policy in different IoT layers.

The paper is organized as follows: section II briefly presents IoT limitations and challenges. Section III summarizes the existing AC models for IoT. The architecture of IoT layers and our methodology of integrating Smart-AC model in IoT are explained in section IV. Section V concludes this paper and presents future perspectives.

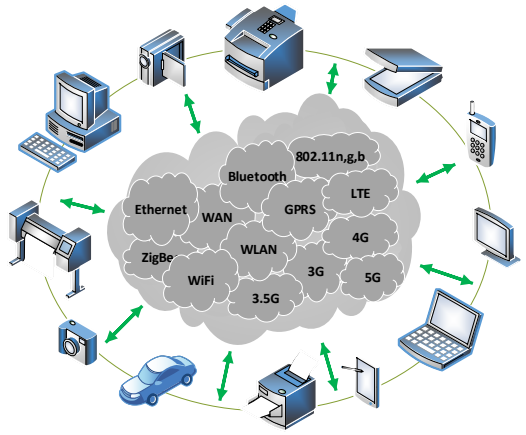



Figure 1. Heterogeous IoT platforms, devices and internet networks

## II. IOT LIMITATIONS AND CHALLENGES

The IoT is penetrating a wide range of domains including cities, homes, industry, healthcare, appliances, and much more, to make everything smart, adaptable and easy. In each area there are various opportunities and challenges. For this purpose, different plans are conducted, various applications are developed, and different software and hardware technologies are cooperatively used. The core

player in IoT technologies are the wireless technologies, where sensor networks play a critical role for linking the physical world with the digital world. For example, e-health applications, environmental monitoring (temperature, plants, weather), intelligent transportation systems, etc.

Although IoT is a popular topic, many challenging problems still need to be addressed, specially the technological and social aspects, before being the IoT idea widely accepted. The IoT challenges can be summarized under the following categories: free internet connectivity, security and all related issues, acceptability among the society, storage and computational ability, scalability, and power consumption [2, 3]. Among other challenges that are also mentioned in [4] we have: data management challenge, data mining challenge, privacy and security challenge, and chaos challenge. All these challenges and limitations open wide research issues, suggestions, methodologies, architectures, and others to address each of them. Since all IoT devices are connected to the internet, they are vulnerable to attacks and security threats. Hence, the core concern is the importance of finding methods to enforce AC policies to prevent any untrusted access and control access to the resources. Accordingly, the acceptance or rejection of this technology is determined by security and privacy.

## III. Related Work

The challenging IoT heterogeneous environments of interconnected networks and distributed systems, the heterogeneity of platforms and applications, and the diversity of users, force the necessity to design a well coherent AC architecture to enforce AC policy and administer security features in IoT. In this context, several IoT authentication and AC methods are offered by researchers to integrate security features with this technology from two or more AC models. The most famous AC models that are presented comprehensively and reviewed in literature are: Discretionary Access Control (DAC), Mandatory Access Control (MAC), Role Based Access Control (RBAC), Organization Based Access Control (OrBAC), and Attribute Based Access Control (ABAC) [5-7].

However, Ouaddah et. al in [8] propose SmartOrBAC AC framework for IoT environment. It is based on OrBAC, but due to some OrBAC limitations, it is coupled with the RESTFUL web services mechanisms due to its preferability for the low power constrained environment. The result is the use of web service technologies to implement secure collaboration between organizations. The basic AC requirements in IoT are analyzed by Hussein et. al in [9]. The summarized AC requirements are thin clients and server architecture, autonomous and self-contained AC, infrastructure integration, and attributes centric AC. However, authors adopt the community-based AC architecture (COBAC) to administer the AC in IoT, as a solution for these requirements. IoTCollab framework is developed by Adda et. al in [10], its aim is to ease the collaboration and data sharing in IoT. Based on IoTCollab, an extended study of RBAC and ABAC models are proposed in [11] to manipulate the security concerns of a data sharing framework. A collaborative RBAC (CollRBAC) and collaborative ABAC (CollABAC) models are described. Then, CollRBAC and CollABAC are compared in the light of IoT and IoTCollab requirements. Moreover, a model to find a secure communication between things is proposed by Liu et al. [12]. The main idea is to verify identities between two IoT devices by implementing authentication protocol in the presentation layer where identification key establishment occurs. For Authorization, authors adopt RBAC's authorization concept, implement Elliptic Curve Cryptosystem (ECC) for secure key establishment. A smart contract-based framework is proposed in [13] to implement distributed and trustworthy AC for the IoT by applying smart contract-enabled blockchain technology. It contains: multiple Access Control Contracts (ACCs), one Judge Contract (JC), and one Register Contract (RC). For AC between subjects and objects, ACCs are implemented. To judge the unpleasant behavior of a subject during the AC, JC is used. To manage the ACCs and JC, RC is used. In this context authors address different case studies to demonstrate the feasibility of the framework. As well, some researches address the different layers of IoT architecture, then propose an AC model. Authors in [14] mention that, even there is no consensus on a wide IoT architectures, they are generally comprised of three main components: an object layer, a middle layer(s), and an application layer. The difference between these architectures relies in the middle layer(s). Hence, authors propose a cloud-enabled IoT with four-layer AC Oriented (ACO) architecture which are: an object, a virtual object layer, a cloud service layer, and an application layer. Their purpose is to establish a framework to find AC models for cloud enabled IoT.

However, none of the proposed AC models encompass a general structure or methodology. Each model addresses certain case and implemented based on some features of different models (RBAC, OrBAC …), knowing that these models have limitations and deficiencies [5]. Also, none of them consider the continuous technological changes, and it is built for specific IoT architecture or structure. Thus, two key points inspire us to develop a smart AC methodology which can be implemented to find a Smart-AC model:

- The first is the word "SMART". While we think about IoT, we think about the smart objects which are able to sense and communicate within the IoT environment. Hence, we come up with the idea of a "Smart Access Control model".
- The second is the term "HETEROGENEOUS". As we know, IoT world is heterogeneous, starting from the devices, types of networks, platforms, reaching to the applications. As presented in literature, there are various AC models implemented from two or more AC models, for various cases and studies, and they are also heterogeneous. Thus, finding a Smart-AC model that is capable to include all AC features and can be dynamic enough to be implemented in any IoT environment becomes our objective.

The aim is to find a general and dynamic AC model structure which allow building other AC models to enforce AC policies in IoT, regardless of its architecture and type of application (smart home, smart industry …). For example, in

the field of smart industry or industrial IoT, the new factories of electricity rely on IoT applications. Any intrusion for such applications can cause hazardous consequences, such as cutting off the power for hospitals, ministries and even cities.

## IV. IoT Architecture and the Proposed Smart-AC Model Methodology

### A. *IoT Layers*

In [13], authors mention that there is no consensus on a wide IoT architecture. In this context, [3, 14-16] state that IoT architectures, are generally comprised of three components: 1) the object or the perception layer, 2) the middle or network layer(s), and 3) the application or presentation layer (Figure 2). The object layer contains Internet-enabled devices (cameras, sensors, …) to gather and exchange information with other devices through the Internet. The middle layer works as agent to transfer the collected data from an object layer to a specific destination in the application layer. In this layer, different network communication technologies are used for this purpose, such as Bluetooth, ZigBee, WiFi, 4G, etc. The application layer is where information is received and processed. Also, it is proposed that in some IoT architectures, the middle layer consists of two layers: the network and the service layers. Similarly, in some other researches it is proposed that the middle layer consists of three layers: object abstraction, service management, and service composition layers.

### B. *Smart-Access Control Model features and methodology*

Various AC models are presented in literature, MAC, DAC, RBAC, etc. each model is developed either to overcome some limitations found in preceding models or as a solution for a specific use case and application. Moreover, some other AC models have some combined features from some models to enhance some service features. Hence, our aim is to find a Smart-AC model with the following features:

- Generic enough to include all features offered by the existing AC models.
- Serves as a basis for specifying any AC policy.
- Eases the migration from an AC model to another.
- Handles multiple AC models and find advanced security features and operations.
- Works as a guard to restrict accesses starting from the physical locations reaching to the end user.
- Dynamic enough to handle the diverse needs, use cases, and applications for AC especially with the rapid propagation and evolution of information technologies (cloud computing, IoT …), and others.

The features of each AC model are summarized in [5]. DAC model includes three key components: a set of objects, a set of subjects, and a matrix. AC rights of subject(s) over object(s) are specified and represented as Capability Lists (CLs) and AC Lists (ACLs), which are represented as a matrix. In MAC, AC policy is managed in a centralized way, where security levels are associated with each subject and object, then permissions and actions are derived. As an alternative for DAC and MAC, RBAC model is developed, where users can be assigned some roles and a role can be associated to many users, and a role is a group of rights to use some object(s). OrBAC model is implemented to overcome some of the limitations in DAC, MAC and RBAC, and to find a more abstract control policy.

However, Figure 3 shows our vision for a Smart-AC model with the above mentioned, and AC models features. It illustrates the features and parameters for all models (subjets, objects …) with the ability to define new ones (e.g. X, Y …) and find the needed mappings and associations between each model entities. Thus, any AC model can be developed by combining features from the existing models, in addition to the ability to add or define new ones. Hence, various models can be implemented, migrated, and used dynamically to enforce AC policy based on the needed security requirements. In IoT, this approach is practical, due to its dynamicity and ability to be upgraded based on the continuous technological changes. Hence, the dynamic and generic properties of such model, if implemented, 1) will help constructing new AC models based on a general AC

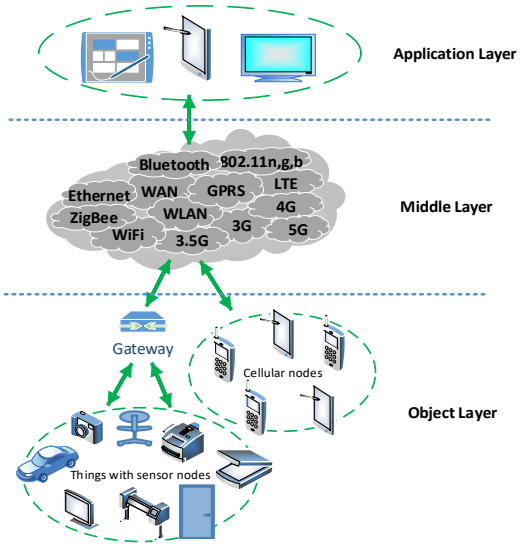



Figure 2. General IoT Layers

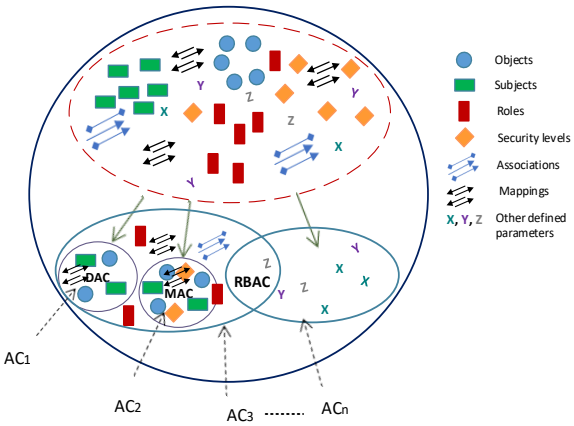



Figure 3. The vision of Smart-AC model

model concept. 2) Migrating AC policies from one model to another will also help companies or industry sectors reduce the complexity and cost in this domain. 3) Different AC models can cooperate within the same company and this would be effective for IoT applications.

Each IoT layer needs the integration of AC model(s) to enforce AC policy and find secure communication environment. Various access types might exist in each IoT layer, based on the existing objects and subjects, and the needed security requirement to deny any illegal access and determine who can access what and when. Based on general architecture of IoT layers and our illustrated vision for the Smart-AC model, Figure 4 shows how any AC model can be combined with IoT layers (AC1, AC2, … $AC_n$...), which is derived/defined from the smart-AC model.

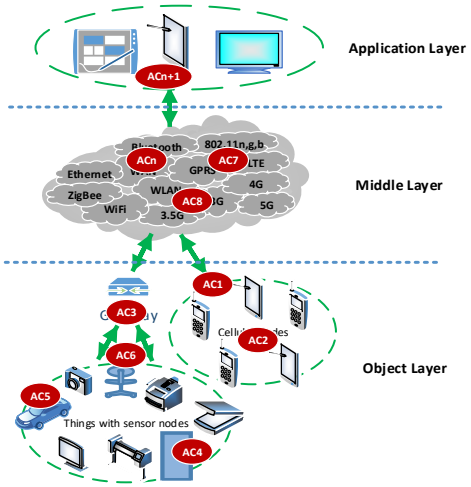



Figure 4. The vision of Smart-AC model

## V. Conclusion and Future Perspectives

IoT is an emerging phenomenon, since various technologies and applications are combined to find a real intelligent world. Finding a secure IoT environment is a challenging issue. Various AC models are presented in literature to enforce AC policy in IoT, but they lack the idea of being upgradable or dynamic to follow the progression of technological changes and upgrades. For this purpose, we present the headlines of new methodology to define a generic Smart-AC model, its concept is dynamic enough to define other AC models based on the needed security requirements. As future perspective, our aim is to define the presented headlines, of this paper, as a formal steps or guidelines and develop a general structure of the proposed methodology. Also, we will consider Industrial IoT (IIoT) or Industry 4.0 as an example to implement our methodology.

## Acknowledgment

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) and Fonds Québécois de la Recherche sur la Nature et les Technologies (FQRNT).

## References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," Computer networks, vol. 54, no. 15, pp. 2787-2805, 2010.

[2] S. C. Mukhopadhyay and N. K. Suryadevara, "Internet of things: Challenges and opportunities," in Internet of Things: Springer, 2014, pp. 1-17.

[3] K. Ahmad, O. Mohammad, M. Atieh, and H. Ramadan, "IoT: Architecture, Challenges, and Solutions using Fog Network and Application Classification," presented at the 19th International Arab Conference on Information Technology (ACIT 2018), Lebanon, Nov. 2018.

[4] I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," Business Horizons, vol. 58, no. 4, pp. 431-440, 2015.

[5] N. Kashmar, M. Adda, and M. Atieh, "From Access Control Models to Access Control Metamodels: A Survey," the Future of Information and Communication Conference (FICC) 2019, US, San Francisco, March 14-15, 2019. accepted

[6] V. C. Hu, D. F. Ferraiolo, R. Chandramouli, and D. R. Kuhn, Attribute-Based Access Control. London: Artech Hous, 2018.

[7] M. Ennahbaoui and S. Elhajji, "Study of access control models," in Proceedings of the World Congress on Engineering, 2013, vol. 2, pp. 3-5.

[8] A. Ouaddah, I. Bouij-Pasquier, A. A. Elkalam, and A. A. Ouahman, "Security analysis and proposal of new access control model in the Internet of Thing," in 2015 international conference on electrical and information technologies (ICEIT), 2015, pp. 30-35: IEEE.

[9] D. Hussein, E. Bertin, and V. Frey, "Access control in IoT: From requirements to a candidate vision," in 2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN), 2017, pp. 328-330: IEEE.

[10] M. Adda and R. Saad, "A data sharing strategy and a DSL for service discovery, selection and consumption for the IoT," Procedia Computer Science, vol. 37, pp. 92-100, 2014.

[11] M. Adda, J. Abdelaziz, H. Mcheick, and R. Saad, "Toward an access control model for IOTCollab," Procedia Computer Science, vol. 52, pp. 428-435, 2015.

[12] J. Liu, Y. Xiao, and C. P. Chen, "Authentication and access control in the internet of things," in 2012 32nd International Conference on Distributed Computing Systems Workshops, 2012, pp. 588-592: IEEE.

[13] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, and J. Wan, "Smart contract-based access control for the internet of things," IEEE Internet of Things Journal, 2018.

[14] A. Alshehri and R. Sandhu, "Access control models for cloud-enabled internet of things: A proposed architecture and research agenda," in 2016 IEEE 2nd International Conference on Collaboration and Internet Computing (CIC), 2016, pp. 530-538: IEEE.

[15] S. Talari, M. Shafie-Khah, P. Siano, V. Loia, A. Tommasetti, and J. Catalão, "A review of smart cities based on the internet of things concept," Energies, vol. 10, no. 4, p. 421, 2017.

[16] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," Future generation computer systems, vol. 29, no. 7, pp. 1645-1660, 2013.

# A Smart City IoT Integrity-First Communication Protocol via an Ethereum Blockchain Light Client

Elizabeth Reilly *, Matthew Maloney *, Michael Siegel *, Gregory Falco *
*Massachusetts Institute of Technology
{reillye, maloneym, msiegel, gfalco}@mit.edu

***Abstract*—Smart city IoT is responsible for communicating system-critical data about urban infrastructure that keeps our modern cities functioning. Today, IoT devices lack communication protocols with data integrity as a priority. Without data integrity, smart city infrastructure is at risk of actuating urban environments on compromised data. Attackers can use this IoT communication flaw to wage cyber-physical attacks on Smart Cities. We designed and developed an integrity-first communication protocol for IoT that is distributed and scalable based on the Ethereum blockchain. Our light client ensures data communication integrity for systems that require it most.**

***Keywords*—** IoT Security, Blockchain, Smart Cities, Communication Integrity, Ethereum, IoT Communication Protocol

## I. Introduction

Despite the prevalence and growing popularity of smart city IoT devices such as smart meters or CCTV security cameras, many of these devices lack security [9]. There are hundreds of videos dedicated to demonstrating how to hack into IoT devices in minutes [4]. The lack of communication standards and various IoT device manufacturers further complicates the security of these memory and processing power-constrained devices [8]. Our research focuses on a specific security gap for urban critical infrastructure IoT - improving communication integrity. Prioritizing integrity, what we call integrity-first communication, is important for smart city IoT because integrity failures can result in cyber-physical damages.

Because many IoT devices have different operating systems and configurations, it is hard to establish one single communication protocol that can be universally applied [6]. IoT devices with limited memory and computational resources pose an even greater concern as they often do not have the resources to host a communication protocol at all [12].

Several IoT communication protocols have been proposed that attempt to prioridize communication integrity, yet they are not scalable [13].

Given the distributed nature of IoT devices and their scale, a suitable integral communication protocol is needed. We propose using the blockchain for this purpose. Blockchain provides a scalable, distributed ledger that requires consensus across all participating nodes [14]. Once a transaction has been approved by the majority of nodes, it cannot be tampered with or erased unless an attacker has control over 51% of the nodes. Therefore, blockchain is a good candidate for IoT communication integrity, assuming the blockchain is sufficiently large.

However, hosting a blockchain node on IoT has been problematic because each node in the network must store the chain of transactions. Many IoT devices have memory and computational limits which keep them from being able to store entire chain data. Current solutions such as the tangle, IoT chain, IoTex and NeuroMesh have tried to overcome this issue by using lighter clients or hardware solutions [15][5][1][8]. However, each has limitations. We have designed a light version of the Ethereum blockchain that overcomes these outstanding issues. Our light client provides integrity-first communication for IoT devices.

## II. Related Works

Today, there is no industry standard for how IoT devices should communicate securely. Many current communication protocols for IoT devices either poorly address security, or are not scalable. Researchers have only recently begun investigating blockchain as a potential IoT communication protocol. However, most IoT blockchain implementations are either too large, too centralized, too expensive or use hardware solutions.

### A. Legacy Communication Protocols

Modbus was one of the original proposed communication mechanisms for IoT devices [10]. Modbus was originally designed for isolated systems so the integrity of messages was not taken into consideration [10]. DNP3 is another early communication protocol for IoT devices built upon TCP/IP. It has a master-slave structure and likewise does not provide integrity of messages and no authentication mechanism between master and slave [10]. Both of these systems do not ensure integrity-first communication and were designed for early industrial IoT.

### B. Modern IoT Communication Protocols

IoT devices with limited resources are often referred to as constrained nodes. Currently, DTLS is the default security protocol used for application messages between constrained nodes [2]. DTLS is built upon the UDP protocol [12]. However, DTLS lacks scalability. Similarly, the IETF has attempted to create a standard for communication protocols for constrained nodes. One such protocol is CoAP, which is also built upon UDP and includes a subset of HTTP functions that are optimized for resource scarce environments [16]. However, similar to DTLS, this protocol does not perform well under high load or congestion [16]. Also, UDP is known for being unreliable and messages are often lost [1]. Therefore, none of the security protocols are able to achieve a distributed, scalable form of integral communication.

MQTT is another communication protocol popular with constrained nodes. It is built on TCP and IP [17]. MQTT has 3 different reliability standards called Quality of Service Levels [17]. The first level is 0, in which a message is delivered at most once and no acknowledgement of reception is required [17]. The second is level 1, in which every messaged is delivered at least once and acknowledgement of reception is required [17]. The third level is 2, in which a four-way handshake mechanism is used to deliver a message exactly once. Despite these different levels of reliability, MQTT also lacks scalability as more nodes connect to a centralized broker that must perform handshake procedures [13].

### C. IoT Blockchains

Blockchain has been proven to be highly scalable but has not been largely applied to IoT devices nor explored as a source of integrity-first communication. The most well known variation of blockchain specifically designed for IoT is the tangle, which is accompanied by the IOTA coin [15]. The tangle stores the chain in a DAG structure to reduce space and also has a light variation specifically for small devices [15]. In order for nodes to get their transactions approved, they must approve 2 other transactions first [15]. This makes the transactions fee-less but also means that there are no miners in the system. Without miners, the system has less incentive to grow. Furthermore, nodes can decide for themselves which transactions to approve. This can prevent certain transactions from being approved. Therefore, a coordinator that decides the order in which transactions will be approved is currently used. This is problematic because it creates a level of centralization and single point of failure within the network [2].Also, the tangle is a relatively small blockchain variation making it vulnerable to a 51% takeover attack.

### D. IoT Chain

IoT chain uses a DAG structure similar to the tangle and also uses Simplified Payment Verification (SPV) to facilitate operations on smaller devices [5]. SPV allows devices to conduct payment verification without maintaining complete blockchain information as long as block headers are preserved [5]. IoT chain also uses practical byzantine fault tolerance (PBFT) to ensure fast consensus [5]. PBFT is a state machine replication algorithm that is based on the consistency of message passing [5]. The combination of the DAG structure and PBFT allow transaction times for IoT chain to be milliseconds [5]. Despite these benefits, the primary limitation of IoT chain is that it requires a specific operating system and linking module [5]. This makes it a hardware solution which is more difficult to integrate with older devices. Also, the relatively small size of IoT chain makes it vulnerable to a 51% attack.

### E. IoTex

IoTex similarly uses PBFT and SPV to ensure fast transaction times and limited storage space [1]. The key concept for IoTex is the idea of blockchains within blockchains [1]. Essentially, different blockchains are used for different kinds of IoT devices [1]. This is done because different devices have different features and by separating these features into different blockchains, no one device has to store large chains [1]. For example, one chain might record transactions and another one might have turing-complete contracts on it [1]. There is a root blockchain that manages all the blockchains [1]. However, this root blockchain is problematic because it introduces a layer of centralization. Like the tangle and IoT chain, the small size of IoTex makes it vulnerable to a 51% attack.

### F. NeuroMesh

One blockchain that successfully provides secure communication for IoT devices, which is most similar to our design, is NeuroMesh [8]. NeuroMesh functions as a "friendly" botnet to fight against other botnets and communicates security commands to IoT devices using the Bitcoin blockchain as the communication protocol [8]. While this technology does provide integral communication because of the size of the Bitcoin network (minimizing the 51% attack risk), it has several operational constraints. First, NeuroMesh is only able to send 80 bytes of characters at a time [8]. This might be fine for sending security commands to IoT devices, but is insufficient to handle substantial data transfers. Second, the Bitcoin network is slow and expensive compared with Ethereum [18]. Finally, while NeuroMesh is only 1MB, it uses SPV which must store block headers [8][14]. This limits how small it can become.

## III. Design Parameters

For our light client, we focused on allowing constrained nodes to participate in global blockchain networks without needing to host an entire node. There were many parameters we had to consider in this light design. First, we needed to select an implementation of blockchain to use. There are many different implementations of blockchain, including both Bitcoin and Ethereum [18][14]. We initially considered creating our own blockchain, however, small blockchain networks are often vulnerable to 51% attacks. Also, we wanted to choose a blockchain implementation that was fast, inexpensive and reliable. Therefore, we decided on Ethereum.

Next, we had to find a way to scale down the size of Ethereum, both in terms of code size and chain data. We chose not to store the chain data at all and to reduce the code size using compiler tricks as well as manual stripping of the code.

Finally, we had to figure out how to avoid storing the chain data while also maintaining the correct parameters of the network, such as nonce and gas price. We achieved this by first storing these parameters locally, and occasionally querying the network to ensure that locally stored data matched global data. With all of these considerations in mind, we have implemented a light client version of Ethereum that is able to send and receive data from other devices.

## IV. Light Client Implementation

### A. Using Ethereum as a Base

As previously mentioned, the integrity of blockchain communication comes from the size of the network. For example,

when the number of participants in the blockchain is small, it is easy for a malicious actor to execute a 51% attack. While Bitcoin is the largest blockchain network, we sought a large blockchain that does not limit data transfer to 80 bytes nor has a high cost of transaction. For these reasons we chose the Ethereum blockchain [18]. Like Bitcoin, Ethereum uses proof of work as a decentralized consensus algorithm to hash blocks to the blockchain [18]. The coins exchanged in the Ethereum network are ethers rather than bitcoins [18].

### B. Avoiding Storing Chain Data

A considerable challenge was avoiding the need to store the Ethereum chain data (including headers) on constrained nodes so that they still could participate in the Ethereum network. Nodes rely on chain data to determine their nonce [18]. A node's nonce is a count of its outgoing transactions [18]. Every time a node wants to make a new transaction it must include its current nonce, otherwise the transaction will be rejected. Keeping the nonce consistent is crucial to the function of the light client.

In the light client implementation, nodes keep track of their nonce locally and occasionally query other nodes in the network to gain a consensus on their correct nonce.

In addition to the nonce, nodes need to have correct gas limit and gas price values in order to ensure that their transactions are mined and stored in the chain. Gas pays for the computation of the transaction regardless of whether it is accepted or not. This transaction fee is equal to gas limit * gas price. These values have been hard coded at a gas price of 30 Gwei (equal to 0.00000003 ether) and a gas limit of 210000 units of gas because these values are highly likely to get transactions added to the chain. Therefore, light nodes do not need to query for this data. With the gas and nonce set correctly, the light node is able to exchange transactions without storing chain data.

### C. Reducing Code size

Eliminating the need to store chain data greatly reduces the size of the code that needs to be stored on the node. However, the current open source code for Ethereum is still 30 MB. Therefore, we have stripped as much code as possible from the node in order to reduce its size. Since the light nodes are not acting as miners in the system, there are many aspects of the code base that can be removed for the light client, such as any code related to mining. Furthermore, there are a few compiler flags that we also added in order to cut down on size. To date, we have stripped the code base to 5 MB while preserving the ability to send transactions and query other nodes. Although 5 MB is sufficient for many smart city devices (many CCTVs), we aim to reduce this size even further.

### D. Architecture and Communication

As seen in Figure 1, the light client architecture interacts with full nodes and other light nodes in a distributed manner. The light nodes do not store any chain data but can both send and receive data as indicated by the two-way transaction broadcast arrows. The light nodes send data by adding that data to a transaction and then sending the transaction to another node. When the light node wishes to send a transaction, it pings all Ethereum nodes, just like any full node would. Only a full node can process the light node's data, which is indicated by the red "x" shown between light node communications. When a full node successfully mines the light node's block (indicated by a checkmark), other full nodes will no longer be able to mine the same block (indicated by an "x"). After the light node's block is successfully mined it is posted to the Ethereum network as an immutable value on the ledger. The light client must pay a small gas fee should it wish to send data, usually on the order of a few cents [11].

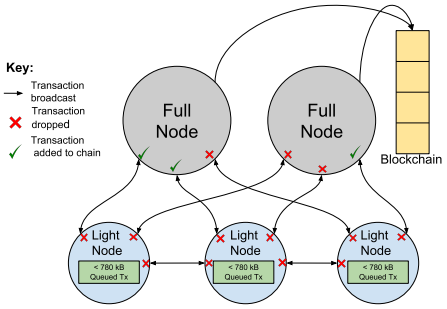



Fig. 1. Distributed Node Architecture

Light clients are able to receive data from the full nodes by pulling data from the blockchain. Although these light nodes will not store the blockchain themselves, they can query other nodes for information about the chain. Therefore, these nodes can query to see whether any recent transactions have been sent to them and then download these specific, small transactions. These transactions are signed and can be verified using cryptography [18]. Therefore, the light nodes and full nodes are able to send and receive messages effectively, and the small node does not need to store the entire chain data.

If the light client has a large amount of information it wants to send all at once, it can queue transactions and aggregate the transfer. The gas limit of an Ethereum transaction is about 3,000,000 gas which allows up to 780 kB of data per transaction [3]. Therefore, at any given time we queue up to this amount. The extent of transaction queuing varies by the amount of available storage on the device.

## V. Discussion

Our Ethereum light client communication protocol for smart city IoT devices provides assurance that data was not compromised in transit. The light client authenticates the origin of a data source based on the unique Ethereum address that initiated the transaction. This can be trusted because of the proof of work consensus algorithm and the size of the Ethereum chain. Many urban critical infrastructure sectors rely on integral communications. For example, electric grid infrastructure requires device state information to be transferred over networks at regular intervals. If an attacker compromised the integrity of state data, there could be cyber-physical damages.

Despite the benefits of using the light client for communications, there are limitations of this approach which restrict its practical implementation to certain IoT use cases.First, the light client will require an operating system to run. This considerably limits the number of devices it is applicable for. Also, the block time of our light client is on average 15 seconds meaning that there is a transaction delay for data [11]. This is superior to Bitcoin's block time which is closer to 10 minutes [11]. Therefore, devices requiring real-time data transfer should not use this protocol. Another consideration is that the average cost to send an Ethereum transaction using our light client implementation is 10 cents for gas [11]. This is considerably less expensive compared to Bitcoin-based NeuroMesh which costs an average of 75 cents per transaction [7]. However, for IoT use cases that require constant transactions, this could become very costly over time. While our Ethereum light client is both faster and less expensive than NeuroMesh, we acknowledge that it is not optimal for all types of use cases.

Another advantage of our IoT blockchain compared with IoTex, IoT chain and the tangle is that the light client resides on a major blockchain. Ethereum is widely used - over 10,000 transactions are sent every hour and within a 24 hour time period, around 200,000 addresses will be active [11]. Because of the high volume of users, it would be difficult for an attacker to gain control of over 51% of the network, unlike other IoT blockchain implementations.

To date, we have tested the light client on less than 100 devices. The transactions have reliably been sent and received within the expected constraints of the Ethereum network. We plan to install this communication protocol on more devices to test the light client at scale and under failure conditions, such as when a large portion of the network crashes or a node's transaction is continually rejected. To achieve scale, we will be publicly release the light client to the research community for testing.

## VI. Conclusion and Future Work

Our research in developing a light client, hopes to address the problems with communication integrity for IoT devices. Public blockchains such as Ethereum have given us the ability to disseminate data in a scalable and distributed fashion. Our future work on the light client will include further reducing its size and establishing optimal conditions for its function on smart city IoT devices. We also aim to develop an agent that will interpret data from the light client and implement commands on the IoT endpoint. Building an agent for the light client will be the basis for an integrity-driven approach to performing updates for IoT devices at scale.

## References


[1] Iotex: A decentralized network for internet of things. 2018. https://iotex.io/white-paper.

[2] Our response to 'a cryptocurrency without a blockchain has been built to outperform bitcoin. 2018.

[3] *Ethereum Blockchain App Platform*, (Accessed: 2019-02-13). https://www.ethereum.org.

[4] *Shodan*, (accessed April, 2018). www.shodan.io/.

[5] Iotchain: A blockchain security architecture for the internet of things. *In Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC 2018)*, pages 1–6, April 2018.

[6] Riccardo Bonetto, Nicola Bui, Vishwas Lakkundi, Alexis Olivereau, Alexandru Serbanati, and Michele Rossi. Secure communication for smart iot objects: Protocol stacks, use cases and practical examples. *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2012.

[7] David Easley, Maureen O'Hara, and Soumya Basu. From mining to markets: The evolution of bitcoin transaction fees. (May 1, 2018). https://ssrn.com/abstract=3055380.

[8] Gregory Falco, Caleb Li, Pavel Fedorov, Carlos Caldera, Rahul Arora, and Kelly Jackson. Neuromesh: Iot security enabled by a blockchain powered botnet vaccine. *ACM Proceedings: International Conference on Omni-Layer Intelligent Systems (COINS)*, 2019.

[9] Gregory Falco, Arun Viswanathan, Carlos Caldera, and Howard Shrobe. A master attack methodology for an ai-based automated attack planner for smart cities. *IEEE Access*, pages 48360–48373, August 28, 2018.

[10] Igor Fovino, Andrea Carcano, Thibault Murel, and Alberto Trombetta. Modbus/dnp3 state-based intrusion detection system. *2010 24th IEEE International Conference on Advanced Information Networking and Applications*, 2010.

[11] Adam Gencer, Soumya Basul, Ittay Eyall, Robert van Renessel, and Emin Sirer. Decentralization in bitcoin and ethereum networks. *arXiv preprint arXiv:1801.03998*, 2018.

[12] Jiyong Han, Minkeun Ha, and Daeyoung Kim. Practical security analysis for the constrained node networks: Focusing on the dtls protocol. *2015 5th International Conference on the Internet of Things (IOT)*, 2015.

[13] Andrew Minteer. *Analytics for the Internet of Things (IoT)*.

[14] Satoshi Nakamotoi. Bitcoin: A peer-to-peer electronic cash system. 2008. https://bitcoin.org/bitcoin.pdf.

[15] Popov Sergui. The tangle. 2015. https://iota.org/IOTAWhitepaper.pdf.

[16] Zhengguo Sheng, Shusen Yang, Yifan Yu, Athanasios Vasilakos, Julie McCann, and Kin Leung. A survey on the ietf protocol suite for the internet of things: standards, challenges, and opportunities. *IEEE Wireless Communications*, pages 91–98, 2013.

[17] Dinesh Thangavel, Xiaoping Ma, Alvin Valera, Hwee-Xian Tan, and Colin Tan. Performance evaluation of mqtt and coap via a common middleware. *2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2014.

[18] Gavin Wood. Ethereum (eth) – whitepaper. 2015. http://gavwood.com/paper.pdf.

# Towards a Multilayer Strategy Against Attacks on IoT Environments *

Davino Mauro Junior, Walber Rodrigues, Kiev Gama, José A. Suruagy, Paulo Andre da S. Gonçalves
Universidade Federal de Pernambuco - Centro de Informática (CIn/UFPE)
Recife, Brazil
{dmtsj, wmr, kiev, suruagy, pasg}@cin.ufpe.br

***Abstract*—The Internet of Things market has seen a large growth in numbers in the recent past. With it, security is becoming a usual concern among consumers. By taking as a starting point an already existing categorization of typical IoT attacks grouped by TCP/IP network layers, we did a non-exhaustive search of solutions addressing each attack. We found that solutions are typically focused on a single layer or even a specific attack only. Furthermore, these solutions lack flexibility to incorporate new attacks. To avoid the non-practical approach of having multiple non-extensible tools, this paper presents an ongoing work that focuses on a multilayer approach to IoT threats. The proposed system leverages an autonomic architecture to analyze network traffic in a distributed manner, detecting suspicious behavior with preconfigured rules being applied by a Complex Event Processing (CEP) engine.**

***Index Terms*—Security; Internet of Things; MAPE-K;**

## I. Introduction

The Internet of Things (IoT) is growing rapidly, with over 24 billion devices expected to be installed by 2020 [1]. This exponential growth reflects on various ecosystems; from industrial to connected houses. Security solutions focused on these ecosystems are crucial as these devices often have access to sensitive information.

The Smart Home ecosystem represents a big part of the IoT market share [2], thus requiring special attention. Devices which on a recent past were considered rare, e.g., smart cameras and smart locks, are now actively present in consumer's houses, leading to new challenges both in security and privacy domains. In terms of security, there are multiple scenarios that causes concern among users who wish to purchase these devices. For example, Fernandes et al. showed how a compromised smart lock from the IoT Smart Home platform Smart Things could allow an attacker to enter the house without dweller consent [3].

Privacy is also a serious concern among users. For instance, Nest smart cameras, Google's solution for IoT indoor and outdoor surveillance, were recently targeted by hackers. After compromising a Nest baby monitor, an attacker did broadcasts of voice messages threatening parents, who experienced a dreadful situation, feeling invaded and unconfortable [4]. This is just one, among many examples, but cases like this demonstrate the importance of addressing the security concerns involving these smart devices.

Different approaches and tools addressing these attacks were presented in the recent past. Due to the large number of IoT attacks, it is important to categorize them to have a better understanding to appropriately provide solutions to detect and neutralize such threats. Nawir et al. [5] and Ashraf et al. [6] proposed a categorization for the different types of IoT attacks. The former uses a fine-grained classification based on TCP/IP network layers, while the latter provides a coarse-grained perspective of three layers (Cloud, Network and M2M layers). Since we are focused on home IoT, we focused on the fine-grained one.

In section II-B, we did a non-exhaustive search of IoT threat defense tools. We matched them with their respective attacks grouped under different network layers, as presented by Nawir et al. [5]. This categorization brought up two main limitations of these solutions. First, these solutions often act on a single network layer, forcing the user to have multiple tools in place as to prevent the attacks. Second, most of these solutions do not provide any means to extend the tool in question for new attacks, lacking flexibility.

Thus, there is an open challenge concerning the integration and flexibility of the approaches to many IoT threats into a tool that could concentrate these defense mechanisms. It would allow an easier adoption as well as the ability to easily incorporate solutions to new threat patterns that may appear. As an attempt to address these limitations, we present our ongoing work on the *IoT-Flows* platform, which consists on a new security system for IoT environments focusing on two limitations of the state of the art: **extensibility** and **multilayer defense**. Our approach followed the MAPE-K reference architecture [7], typically used in autonomous systems.

## II. Background and Related Work

In this section, we revisit a study on the taxonomy of IoT attacks presented by Nawir et al. [5], using it as baseline for a non-exhaustive search on IoT solutions for each attack.

### A. Categorization of IoT Attacks

Nawir et al. propose a security attack categorization for each network layer [5]. Table I shows the attacks distributed on the different layers. In the **Physical** layer, the study categorizes jamming and tampering attacks. *Jamming* consists in constantly, deceptively or randomly compromising the communication channel with meaningful data, whereas *Tampering* is physically attacking the device [8]. On the **Data Link** layer, attacks are classified as collision, resource exhaustion and unfairness. *Collision* consists on sending packets at the same time of legitimate data packets, harming specific packets instead of the whole channel [8]. Resource *Exhaustion* are attacks that force the device to consume its resources deliberately, for example,

This material is based upon work supported by RNP (Brazil) and NSF (USA) under Grant Nos. 1740897 and 1740916.

sending multiple requests to devices that uses batteries until the battery dies. Some MAC protocols give priority to devices that are very low on battery, the *Unfairness* attack consists in creating a low battery device to have priority when sending packets denying real traffic [8].

The majority of attacks are reported on the **Network** layer, i.e., spoofed, altered or replayed routing information, selective forwarding, sinkhole, sybil, wormholes, HELLO flood and acknowledgement spoofing. The attacks based on *Spoofed, Altered or Replayed routing information* are based on unprotected ad-hoc networks, where the routing can be compromised. *Selective forwarding* is related to a denial of service on a specific node with packets being dropped. *Sinkhole*, as described by [9], consists on a network node pretending to have the shortest path to other node ir order to drop packets, modify data or interfere in clustering algorithms. *Sybil* attacks are based on a node obtaining multiple fake identities and misleading other nodes on the network. *Wormholes* attacks are based on different devices on distinct connected networks, where the data from one network is sent to another in order to create real, but misleading information. *HELLO* messages are used by some protocols to establish connection or neighborhood relation and are used with a high power transmitter to create fake proximity relations [8]. *Acknowledgement spoofing* uses ack messages to pretend that a disabled node is alive or the connection between two nodes is strong than it apparently is.

On the **Transport** layer, there are flooding and de-synchronization attacks. *Flooding* consists on exploring the natural vulnerabilities of TCP and UDP protocols as the UDP have no flow control and the TCP protocol is vulnerable to SYN (new connection request) flood. De-synchronization is the interference of an attacker in order to interrupt an active connection between two nodes, using fake packets containing error or specific control flags [10]. On the **Application** layer, the vulnerabilities reported are related to system reliability and cloning attack [5]. *Reliability* issues are often related to execution problems like buffer overflow. *Cloning* is the capability of attackers steal information from devices or steal the device credentials.

### B. Categorization of Solutions

Building on the taxonomy of the IoT attacks previously described, we did a non-exhaustive search with the main goal of distributing the state-of-the-art solutions on the different network layers presented. Table I shows the results.

Starting on the **Physical** layer, Namvar et al. presented a novel anti *Jamming* strategy which promotes an IoT controller to protect the IoT devices against malicious radio jammers [11]. For *Tampering* attacks, a team at the NEC Corporation developed a lightweight-architecture for tampering detection on IoT devices, using real-time inspection with no impact on the device normal operations [12]. On the **Data Link** layer, *Collision* attacks are similar to jamming, thus inheriting the same solution. Ruckebusch et al. presented a new architecture designed with IoT in mind that mitigates *Exhaustion* attacks, an after-effect of the previous attacks [13]. Regarding *Unfairness* attacks, Djedjig et al. presented a trust-based defense model to detect malicious behaviour, calculating trust levels for participating nodes [14].

On the **Network** layer, solutions often tackle more than one type of attack. SVELTE is one such case, applying real-time intrusion detection to detect *Spoofing* and *Sinkhole* attacks [15]. Huijuan et al. describes a system that uses watermarked packets to identify whether a network node is doing a *Selective Forwarding* attack, using a trust value to identify how many marked packets are dropped related to normal packet loss, skipping nodes with low trust [16]. Due to the fact of *Sybil* attacks are based on creating fake nodes in networks, Sohail et al. developed a system based on device mobility that is capable of differentiating if a node is fake or not by reading the RSSI (Received Signal Strength Indication) pattern [17]. For Pongle et al., the *Wormhole attack* is very location related, so the developed system periodically broadcasts the nearby RSSI. Then, this information is used by other devices to infer if a node is nearby or not, classifying it as compromised [18]. Singh et al. also used the RSSI but to mitigate *HELLO* attacks, considering that devices have a homogeneous signal strength, any other value too different is considered strange. If the value is close to the standard, the node will be asked to solve a puzzle that increases increases exponentially in difficulty per HELLO message. If the node fails to answer in an assigned time, the node is labeled as strange [19].

*Flooding* attack is commonly used on the **Transport** layer for distributed denial-of-service (DDoS) in IoT environments. Dao et al. presented how attack behaviour learning can be used to detect flooding attacks with smart filters distributed on the network [20]. *Desynchronization* attacks can be mitigated using authentication protocols like the one proposed by Fan et al. using a RFID protection scheme [21]. In the **Application** layer, common defenses include access policies to control information flows between applications and IoTdevices. One such solution was presented by Demetriou et al. with HanGuard, applying SDNs to enforce policies on Smart Home networks [22]. Another approach involves increasing data security on IoT applications. Fernandes et al. developed FloFence, a framework that enables developers to secure function executions involving sensitive data in Android-OS's created processes [23].

Although these solutions give powerful tools to secure smart devices, they are not designed to incorporate future attacks into their security models, therefore lacking flexibility when a new attack appears and not making difficult to improve the tools in an easy way. Also, the solutions do not provide a multilayer defense against the attacks, forcing the user/consumer to have multiple tools in place to improve security of their devices. In the following, we propose a system that focus on solving both of these problems.

## III. The IoT-Flows Platform

### A. Approach

The importance of autonomic systems for IoT security was already pointed out in a review that gathers different solutions emphasizing on autonomic computing to mitigate IoT threats [6], although with a limitation of typically addressing just one or sometimes two layers from TCP/IP. Besides the architectural approach, the concepts of self-healing and self-protection are the main aspects taken from the *self-** principle.

Table I: Categorization of State of the art solutions under the IoT Attacks taxonomy defined by Nawir et al. [5]

| Layer | Attacks | Methods/Strategies | State-of-the-art solution |
|---|---|---|---|
| Physical | Jamming | Creates radio interference and exhaustion on IoT devices | Namvar et al. [11] |
| | Tampering | Creates compromised nodes | NEC Corporation [12] |
| Data Link | Collision | Simultaneous transmission of two nodes on the same frequency | Namvar et al. [11] |
| | Exhaustion | By repetetively colliding the nodes | Ruckebusch et al. [13] |
| | Unfairness | Using above link layer attacks | Nabil et al. [14] |
| Network | Spoofed, altered or replayed routing information | Creates routing loops, extend or shortening sources routes, attracting or repelling network from select nodes. | Raza et al. [15] |
| | Selective forwarding | Choose what information to gather before transmitting. | Deng et al. [16] |
| | Sinkhole | Compromised node tries to attract network traffic by fake advertising its fake routing update | Raza et al. [15] |
| | Sybil | Single node duplicates its node to be in multiple locations. | Abbas et al. [17] |
| | Wormholes | Selectively tunneling or retransmit information to the IoT devices. | Pongle et al. [18] |
| | HELLO flood | Uses HELLO packets as weapon to launch the attack on IoT system | Singh et al. [19] |
| | Acknowledgement spoofing | Spoof the link layer acknowledgments for overhead packets. | Raza et al. [15] |
| Transport | Flooding | Repeat the request of a new connection until the IoT system reaches maximum level. | Dao et al. [20] |
| | De-synchronization | Disruption of an existing connection. | Fan et al. [21] |
| Application | Clock skewing, Selective message forwarding, Data aggregation distortion | The adversaries usually masquerade like normal behavior in IoT system. Attackers also can still choose a message that he/she intend in the IoT system and launched their own malicious activities. | Demetriou et al. [22] Fernandes et al. [23] |

We propose the *IoT-Flows* platform, which is a system that employs this autonomic approach in a distributed architecture with multiple components, each one possessing a specific and unique responsibility. Following an old design principle– Separation of Concerns–one could see each component as a module addressing a different concern, i.e., a problem [24]. A component could, for example, deal with the problem of monitoring the network in a distributed manner and filter the traffic data, while another component would analyze this data and look for signs of suspicious behaviour. This separation of concerns alleviates the complexity of a security enforcement system monitoring different networks with heterogeneous devices. The architecture of the system is based on the MAPE-K, a traditional architecture blueprint originally designed for self-adaptive systems. For brevity reasons, we will not discuss the architecture on this paper, please refer to the original white paper for a detailed discussion [7].

*IoT-Flows* focuses on securing communication in-between IoT devices. It acts on the different network layers, providing a **multilayer defense** for IoT environments. The system is able to monitor the traffic on the different WiFi networks that the smart devices are connected to. It also provides **extensibility**, allowing the user of the system to incorporate new attacks into the defense model, in the form of Complex Event Processing (CEP) rules. Currently we have developed patterns against attacks in three TCP/IP layers: **Network**, **Transport**, and **Application** layers. For instance, while monitoring the network, the system is able to detect that an IoT device is being targeted for *Acknowledgement Spoofing* with a fake device trying to masquerade the official device. At the same time, on the transport layer, the attacker would be flooding the IoT device with multiple requests and could also act on the application layer, trying to masquerade normal behaviour requests, like turning the device on or off. The system is able to detect any of these behaviours while monitoring the network traffic and applying pre-configured rules that analyze the packets being sniffed. Once a suspicious behavior is detected, the system can alert the user or block all requests directed to the IoT device in question, therefore stopping the attack.

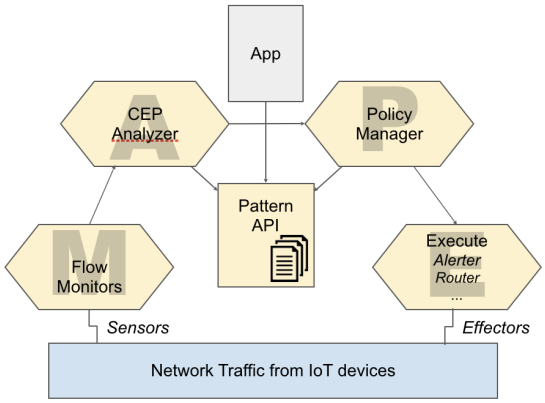



Figure 1: Architecture of the IoT-Flows platform.

*IoT-Flows* allows the user to download new security patches that provides detection of new attacks while also providing manual configuration, if needed. We have tested the approach of having an extensible mechanism based on Complex Event Processing rules that allows to easily include the identification of new attacks. Some drawbacks are the need to understand the rule language of the CEP Engine (Esper[1]) and understanding the metadata of the packet structure in order to write the rules.

### B. Architecture

Based on the architectural parts of the MAPE-K blueprint [7], we define the following components (shown in Figure 1): (i) Flow Monitor(s), (ii) Pattern API, (iii) CEP Analyzer, (iv) Policy manager, (v) Execute components (Alerter/Router/etc). We briefly describe each component below according to their matching role on the MAPE-K architecture, however, MAPE-K's Knowledge component is currently not implemented thus we do not describe it. Implementation details were omitted due to space limitations.

**Network Monitor.** The network monitor relates to the *Monitor* component on the MAPE-Karchitecture. Its main

[1] www.espertech.com/esper

responsibility is to aggreggate and filter traffic data, i.e., the resource that is monitored, generating "events" to be analyzed by the next component.

**Pattern API.** This is a core component of the system where the rules are stored and relates to the Analysis and Plan parts of the MAPE-K. A *Pattern* maps a rule to a certain action. For example, one could create a rule to block flows from a certain IoT device to an unwanted address. The system would detect this behavior trying to match the pre-defined rules to the network data being analyzed by the *Monitors*. Then, once matched, the system would then perform a pre-configured action, e.g., alert the user or block the network request. A support Web App allows platform users to write these rules and deploy them into *IoT-Flows*.

**CEP Analyzer.** This component is based on the Analysis part of the MAPE-K architecture. Its main responsibility is to receive the aggregated traffic data from the monitors and apply Complex Event Processing (CEP) to match these data against pre-defined patterns.

**Policy Manager.** This component acts as a bridge between the *CEP Analyzer* and the *Execute* components, i.e., the Alerter, Router, etc. It maps a pattern to a pre-defined action. For example, once the system detects a suspicious behavior that needs a certain action, say a DDoS attack that needs to trigger an email alert for certain users and also to block requests, this component would both

**Execute Components.** These components are logically one in responsibility. The *Alerter* component is responsible for generating an email or SMS alert to the user after a suspicious activity is detected or the component is configured to do so under certain circumstances, e.g., if a smartphone tries to connect to an IoT device. The *Router* component is responsible for applying enforcement policies to the devices, e.g., it could block a request to an unwanted endpoint originating from an IoT device on the network.

## IV. Conclusions and Future Work

Securing IoT devices against both old and new network attacks is crucial. After revisiting the different types of IoT attacks categorized in literature, we did a non-exhaustive search for state-of-the-art solutions targeting those attacks and grouped them under the respective target network layers. We found that solutions for the different types of IoT attacks exist, but they lack flexibility and are often bound to a single network layer. Trying to address these limitations, we presented our ongoing work on the *IoT-Flows* platform. It consists of a system acting as a line of defense that is able to integrate defense mechanisms for IoT attacks expanding across different network layers. The proposed system is also extensible, giving flexibility to incorporate patterns that can identify new attacks and act upon their detection. Our approach is based on the MAPE-K reference architecture for autonomous systems.

In the future, apart from completing and evaluating the proposed system, we envision a crosslayer solution for IoT environments using input from different layers in a simultaneous manner as to prevent complex attacks. In addition, we plan to implement MAPE-K's Knowledge component by employing machine learning techniques to analyze traffic history.

## References


[1] Business Insider, "There will be 24 billion iot devices installed on earth by 2020," 2016. [Online]. Available: https://www.businessinsider.com/there-will-be-34-billion-iot-devices-installed-on-earth-by-2020-2016-5

[2] Forbes, "2017 roundup of internet of things forecasts," 2005. [Online]. Available: https://www.forbes.com/sites/louiscolumbus/2017/12/10/2017-roundup-of-internet-of-things-forecasts/#7b71aae11480

[3] Michigan News, "Hacking into homes: 'smart home' security flaws found in popular system," 2016. [Online]. Available: https://news.umich.edu/hacking-into-homes-smart-home-security-flaws-found-in-popular-system/

[4] Washington Post, "'I'm in your baby's room': A hacker took over a baby monitor and broadcast threats, parents say," 2018. [Online]. Available: https://www.washingtonpost.com/technology/2018/12/20/nest-cam-baby-monitor-hacked-kidnap-threat-came-device-parents-say/

[5] M. Nawir, A. Amir, N. Yaakob, and O. B. Lynn, "Internet of things (iot): Taxonomy of security attacks," in *Electronic Design (ICED), 2016 3rd International Conference on*. IEEE, 2016, pp. 321–326.

[6] Q. M. Ashraf and M. H. Habaebi, "Autonomic schemes for threat mitigation in internet of things," *Journal of Network and Computer Applications*, vol. 49, pp. 112–127, 2015.

[7] IBM, "An architectural blueprint for autonomic computing,," 2005. [Online]. Available: https://www-03.ibm.com/autonomic/pdfs/AC%20Blueprint%20White%20Paper%20V7.pdf

[8] A. Tayebi, S. Berber, and A. Swain, "Wireless sensor network attacks: An overview and critical analysis," in *Sensing Technology (ICST), 2013 Seventh International Conference on*. IEEE, 2013, pp. 97–102.

[9] H. Shafiei, A. Khonsari, H. Derakhshi, and P. Mousavi, "Detection and mitigation of sinkhole attacks in wireless sensor networks," *Journal of Computer and System Sciences*, vol. 80, no. 3, pp. 644–653, 2014.

[10] D. R. Raymond and S. F. Midkiff, "Denial-of-service in wireless sensor networks: Attacks and defenses," *IEEE Pervasive Computing*, no. 1, pp. 74–81, 2008.

[11] N. Namvar, W. Saad, N. Bahadori, and B. Kelley, "Jamming in the internet of things: A game-theoretic perspective," in *2016 IEEE Global Communications Conference (GLOBECOM)*, 2016, pp. 1–6.

[12] NEC Corporation, "Lightweight-architecture tamper detection technology to protect iot devices," 2018. [Online]. Available: https://www.nec.com/en/global/rd/technologies/falsification_find/index.html

[13] P. Ruckebusch, E. De Poorter, C. Fortuna, and I. Moerman, "Gitar: Generic extension for internet-of-things architectures enabling dynamic updates of network and application modules," *Ad Hoc Networks*, vol. 36, pp. 127–151, 2016.

[14] D. Nabil, D. Tandjaoui, I. Romdhani, and F. Medjek, "Trust-based defence model against mac unfairness attacks for iot," 07 2017.

[15] S. Raza, L. Wallgren, and T. Voigt, "Svelte: Real-time intrusion detection in the internet of things," *Ad hoc networks*, vol. 11, no. 8, pp. 2661–2674, 2013.

[16] H. Deng, X. Sun, B. Wang, and Y. Cao, "Selective forwarding attack detection using watermark in wsns," in *Computing, Communication, Control, and Management, 2009. CCCM 2009. ISECS International Colloquium on*, vol. 3. IEEE, 2009, pp. 109–113.

[17] S. Abbas, M. Merabti, D. Llewellyn-Jones, and K. Kifayat, "Lightweight sybil attack detection in manets," *IEEE systems journal*, vol. 7, no. 2, pp. 236–248, 2013.

[18] P. Pongle and G. Chavan, "Real time intrusion and wormhole attack detection in internet of things," *International Journal of Computer Applications*, vol. 121, no. 9, 2015.

[19] V. P. Singh, S. Jain, and J. Singhai, "Hello flood attack and its countermeasures in wireless sensor networks," *International Journal of Computer Science Issues (IJCSI)*, vol. 7, no. 3, p. 23, 2010.

[20] N. Dao, T. V. Phan, U. Sa'ad, J. Kim, T. Bauschert, and S. Cho, "Securing heterogeneous iot with intelligent ddos attack behavior learning," *CoRR*, vol. abs/1711.06041, 2017. [Online]. Available: http://arxiv.org/abs/1711.06041

[21] K. Fan, W. Jiang, H. Li, and Y. Yang, "Lightweight rfid protocol for medical privacy protection in iot," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1656–1665, April 2018.

[22] S. Demetriou, N. Zhang, Y. Lee, X. Wang, C. A. Gunter, X. Zhou, and M. Grace, "Hanguard: Sdn-driven protection of smart home wifi devices from malicious mobile apps," in *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. ACM, 2017, pp. 122–133.

[23] E. Fernandes, J. Paupore, A. Rahmati, D. Simionato, M. Conti, and A. Prakash, "Flowfence: Practical data protection for emerging iot application frameworks." in *USENIX Security Symposium*, 2016, pp. 531–548.

[24] R. J. Mitchell, *Managing complexity in software engineering*. IET, 1990, no. 17.

# Landscape of IoT Patterns

Hironori Washizaki
*Waseda University / NII / SYSTEM INFORMATION / eXmotion,* Tokyo, Japan
washizaki@waseda.jp

Nobukazu Yoshioka
*National Institute of Informatics*
Tokyo, Japan

Atsuo Hazeyama
Tokyo Gakugei University
Tokyo, Japan

Takehisa Kato
Toshiba Digital Solutions Corporation
Kanagawa, Japan

Haruhiko Kaiya
Kanagawa University
Kanagawa, Japan

Shinpei Ogata
Shinshu University
Nagano, Japan

Takao Okubo
Institute of Information Security
Kanagawa, Japan

Eduardo B. Fernandez
Florida Atlantic University
Boca Raton, USA

***Abstract*—Patterns are encapsulations of problems and solutions under specific contexts. As the industry is realizing many successes (and failures) in IoT systems development and operations, many IoT patterns have been published such as IoT design patterns and IoT architecture patterns. Because these patterns are not well classified, their adoption does not live up to their potential. To understand the reasons, this paper analyzes an extensive set of published IoT architecture and design patterns according to several dimensions and outlines directions for improvements in publishing and adopting IoT patterns.**

***Keywords—Patterns, Internet of Things (IoT), Design, Architecture, Survey, Systematic Literature Review (SLR)***

## I. INTRODUCTION

The Internet of Things (IoT) aims to bring connectivity to almost every object (i.e., things found in physical space). Although it extends the connectivity to everyday things, such an increase in connectivity creates many challenges [1]. Patterns are encapsulations of reusable common problems and solutions under specific contexts. As the industry is realizing many successes (and failures) in IoT systems development and operations, many IoT patterns have been published such as IoT design patterns and IoT architecture patterns. However, these patterns are not well classified. Consequently, their adoption does not live up to their potential.

The contributions of this paper are an overview of the current landscape of the IoT architecture and design patterns, identification of shortcomings, and suggestions to improve publishing and adoption of IoT patterns. Here, a complete (to the authors' knowledge) set of IoT patterns that is available in the literature is analyzed. The authors surveyed 33 papers published from 2014–2018. For each question below, the directions for improvement are outlined constructively.

**RQ1.** ***What are the publication trends of IoT patterns?*** To answer this question, we identified the publication years and venues of the 33 papers surveyed.

**RQ2.** ***Are all existing IoT patterns really IoT patterns?*** To answer this question, we confirmed whether or not each proposed or used pattern the IoT context is actually a pattern addressing specific problems and solutions in IoT.

**RQ3.** ***Do the IoT pattern appropriately cover the issues in IoT development and operations?*** To answer this question, we built a classification scheme for IoT patterns and classified each IoT pattern based on the scheme, which includes abstraction level, domain specificity, and quality characteristics.

The rest of the paper is structured as follows. Section II presents the classification scheme. Section III introduces the main sources and analysis process used in this study. Section IV analyzes the literature according to the above-mentioned scheme and process. Finally, Section V presents the conclusions and future work.

## II. IOT PATTERN CLASSIFICATION

To classify IoT patterns, we identified three dimensions: abstraction level, domain specificity, and quality characteristic.

### *A. Abstraction Level*

In general, the IoT systems development process has several major phases with abstraction levels. From the most to the least abstract level, they are analysis, system and software architecture design, detail design and construction. According to these phases, IoT design patterns can be classified into the following three types in terms of abstraction level.

1) High: Reference architectures are architectural models that specify architectural elements and their connections at a very high abstraction level without much consideration of concrete platforms or detailed design issues. These are often used at early phases such as analysis and architecture design.

2) Middle: Unlike reference architectures, there are recommended concrete architecture design of IoT systems and software to address repeated concrete architectural problems such as ensuring interoperability among heterogeneous devices. These architectures are often documented as architecture patterns that encapsulate contexts, recurrent problems and their corresponding solutions. We regarded the abstraction level of the architecture patterns as between high and low.

3) Low: There are recommended detailed designs to address recurrent detailed design problems such as enabling proper communications among software modules while keeping a high extensibility. Since these patterns target specific modules or limited parts of entire system and software, and not the entire software and systems, we regarded the abstraction level of the design patterns as low. These are often used at detail design and construction phases.

### B. Domain Specificity

Domain specificity is important to examine the applicability and reusability of each IoT pattern. It is divided into three types: any, general IoT and specific IoT.

1) Any: General systems and software architecture patterns as well as design patterns that can be adopted to design IoT systems and software if their contexts and problems match the patterns' contexts and problems.

2) General IoT: IoT architecture and design patterns, which are applicable to any IoT systems and software.

3) Specific IoT: IoT architecture and design patterns that address specific problem domains (such as the healthcare) and technical domains (such as the brain-computer interaction).

### C. Quality Characteristic

All systems and software design patterns address some quality characteristics. Basically, IoT design patterns should address interoperability, which is defined as a sub-characteristic of compatibility in ISO/IEC 25010:2010 [2]. We use all quality characteristics except for the functional suitability defined in ISO/IEC 25010, which is a well-accepted quality model systems and software engineering. Additionally, there are other emerging characteristics that are not defined in ISO/IEC 25010 but are common in IoT development and operation. Possible candidates are scalability and privacy.

## III. Analysis Process

We use a systematic literature review (SLR) to evaluate relevant publications on IoT patterns. A SLR aims to assess scientific papers to group concepts around a topic.

1) Initial Search: We used Scopus (https://www.scopus.com/) as a well-accepted reliable scientific databases and indexing systems. For consistency, we executed the following query on titles, abstracts, and keywords of papers regardless of time and subject area. We found 63 papers published from 2014–2018.

"IoT" AND ( "design pattern" OR "architecture pattern" )

2) Impurity Removal: Due to the nature of the involved data source, the search results included some elements that are clearly not research papers such as abstracts and international standards. By removing these results, we got 56 papers.

3) Inclusion and Exclusion Criteria: Applying the following criteria reduced the number of papers to 33 [3-35].

- Inclusion: Papers addressing patterns for designing IoT systems and software, and papers written in English
- Exclusion: Papers that focus on IoT but do not explicitly deal with architecture and design patterns, and papers that are duplicates of other studies

4) Data Extraction: The following information was collected from each paper to answer the research questions: Publication title, publication year, publication venue, types of patterns proposed or used, pattern names, domain names in the case of Specific IoT patterns, and quality characteristics addressed.

## IV. Result and Discussion

### A. Publication (RQ1)

Table 1 presents the distribution of publications over time. The most common publication types are conference papers (17), journals (8), workshops (5), symposiums (2), and refereed book chapter (1). The high numbers of conference papers and journal papers suggest IoT architecture and design patterns are maturing. Since 2016, IoT patterns have become an important and eye-catching aspect of research, and interest has been expanding each year.

Table 1. Primary studies by publication type and year

| | Workshop | Symposium | Conference | Book chapter | Journal | Total |
|---|---|---|---|---|---|---|
| 2014 | | | 1 | | | 1 |
| 2015 | | | 1 | | | 1 |
| 2016 | | 1 | 3 | | 2 | 6 |
| 2017 | 1 | 1 | 7 | | 3 | 12 |
| 2018 | 4 | | 5 | 1 | 3 | 13 |
| Total | 5 | 2 | 17 | 1 | 8 | 33 |

### B. IoT Patterns (RQ2)

We identified 136 patterns mentioned in 33 papers. Among them, 75 patterns (55%) are classified as "Any" in terms of domain specificity. Major non-IoT patterns that appear in multiple papers include *Publish-Subscribe* [6,7,24,33,34], *Client-Server* [24,33,34], *Peer-to-Peer* [24,33], *REpresentational State Transfer (REST)*[33,34], *Service Oriented Architecture (SOA)*[24,34], *Role Based Access Control (RBAC)*[12,15], *Model-View-Controller (MVC)*[20,28], and *Reflection*[8,27]. The other 67 patterns appear in one paper only. Surprisingly, 14 papers used such non-IoT patterns only. According to these results, we confirmed that IoT systems and software are often designed via conventional architecture and design patterns that are not specific to IoT design.

There are 61 IoT patterns (i.e., 45%) in 19 papers that address specific problems and solutions in IoT. The details are discussed in the subsequent section.

### C. IoT Pattern Classification (RQ3)

Table 2 presents the distribution of IoT and non-IoT patterns by abstraction level and domain specificity. Table 3 presents the list of 61 IoT architecture and design patterns.

Surprisingly, only one pattern *Operator-Controller-Module (OCM)* is mentioned in multiple papers [4,32]. The remaining appear once in different papers. This indicates that IoT patterns are not shared or recognized by different research groups. This may be due to its short history. Potential pattern authors are encouraged to carefully check existing IoT patterns before publishing their own "new" patterns.

In terms of abstraction level, about half of 61 IoT patterns are IoT architecture patterns (i.e., 48%), 27 cover IoT design patterns (44%), and 5 reference architectures (8%). We confirmed that IoT architecture patterns and IoT design patterns are almost equally proposed or used.

In terms of domain specificity, 41 patterns (i.e., 67%) are general IoT, while the remaining 20 patterns (33%) are specific to a problem or technical domain shown in Table 3.

Reviewing the combinations of abstraction level and domain specificity, most of IoT design patterns are applicable to any domain but many IoT architecture patterns exist for specific domains. This implies that the unique nature of IoT adoption in specific domains often appears at the architecture level. Design details seem to be commonly addressed by general IoT design patterns or even non-IoT design patterns. In the future, the number of specific IoT design patterns may increase as more domains adopt IoT.

In terms of quality characteristics, many IoT patterns address performance efficiency, compatibility (including interoperability as a sub-characteristic), usability, reliability, and maintainability. This finding is quite natural since major concerns in IoT adoption revolve around these characteristics. Consequently, other quality characteristics remain to be researched. A few number of IoT patterns address security, portability (including adaptability as a sub-characteristic) and scalability. Privacy is rarely addressed by IoT patterns.

Table 2. Patterns by abstraction level and domain specificity

| Abstraction level | Domain specificity | | | |
|---|---|---|---|---|
| | Any | General IoT | Specific IoT | Total |
| High | 20 | 3 | 2 | 25 |
| Middle | 14 | 14 | 15 | 43 |
| Low | 41 | 24 | 3 | 68 |
| Total | 75 | 41 | 20 | 136 |

### *D. Limitations*

The classification of the patterns was conducted by all authors except for the last author of this paper and reviewed by the first author. It is possible that our classification results may not be completely correct. To mitigate this threat to validity, we will open the classification results to the public and call for comments at our Website.

We used Scopus as the initial document base of the SLR. Although it is adopted in other SLRs, relevant papers may be missed. To mitigate this threat, we plan to use other databases, extend our SLR, and elicit public review of the revised results.

## V. Conclusion and Future Work

To overview the current landscape of IoT architecture and design patterns, we surveyed about 136 patterns mentioned in 33 papers published between 2014–2018 according to several dimensions. Most of IoT design patterns are applicable to any domain but many IoT architecture patterns exist for specific domains. In the future, the number of specific IoT design patterns may increase as more domains adopt IoT. In terms of quality characteristics, many IoT patterns address performance efficiency, compatibility, usability, reliability, and maintainability. Consequently, other quality characteristics remain to be researched. Our future works include further analysis on IoT patterns using additional dimensions such as relationships among patterns and writing quality of patterns.

## References


[1] Mohab Aly, Foutse Khomh, Yann-Gaël Guéhéneuc, Hironori Washizaki, Soumaya Yacout, "Is Fragmentation a Threat to the Success of Internet of Things?", IEEE Internet of Things Journal, 2019

[2] ISO/IEC 25010:2011 Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality model

[3] W.T. Lee and P.J. Law, "A case study in applying security design patterns for iot software system," ICASI 2017

[4] C. Wolff, et al., "A layered software architecture for a flexible and smart organic rankine cycle (orc) turbine - solutions and case study," Information Technology and Control, 47(2), 2018

[5] M. Syed, et al., "A pattern for fog computing," VikingPLoP 2016.

[6] L. Roffia, et al., "A semantic publish-subscribe architecture for the internet of things," IEEE Internet of Things Journal, 3(6), 2016

[7] N. Ntuli and A. Abu-Mahfouz, "A simple security architecture for smart water management system," IST-AWSN 2016

[8] E. Jung, I. Cho, and S. Kang, "An agent modeling for overcoming the heterogeneity in the iot with design patterns," MUSIC 2013

[9] C. Pahl, N. Ioini, S. Helmer, and B. Lee, "An architecture pattern for trusted orchestration in iot edge clouds," FMEC 2018

[10] V. Charpenay, S. Kabisch, D. Anicic, and H. Kosch, "An ontology design pattern for iot device tagging systems," IoT 2015

[11] S. Pape and K. Rannenberg, "Applying privacy patterns to the internet of things' (iot) architecture," Mobile Networks and Applications, 2018

[12] I. Ali and M. Asif, "Applying security patterns for authorization of users in iot based applications," ICEET 2018

[13] S. Mendez and J. Zao, "Bci ontology: A context-based sense and actuation model for braincomputer interactions," ISWC-SSN 2018

[14] R. Tkaczyk, et al., "Cataloging design patterns for internet of things artifact integration," ICC Workshops 2018

[15] K. Periyasamy, V. Alagar, and K. Wan, "Dependable design for elderly health care," FedCSIS 2017

[16] G. Bloom, B. Alsulami, E. Nwafor, and I. Bertolotti, "Design patterns for the industrial internet of things," WFCS 2018

[17] T. Spieldenner, et al., "Eca2ld: From entity-component-attribute runtimes to linked data applications," SWeTI 2018

[18] S. Chen, B. Liu, X. Chen, Y. Zhang, and G. Huang, "Framework for adaptive computation offloading in iot applications," Internetware 2017

[19] S. Qanbari, et al., "Iot design patterns: Computational constructs to design, build and engineer edge applications," IoTDI 2016

[20] M. Shopov, "Iot gateway for smart metering in electrical power systems - software architecture," MIPRO 2017

[21] A. Gill, N. Phennel, D. Lane, and V. Phung, "Iot-enabled emergency information supply chain architecture for elderly people: The australian context," Information Systems, 58, 2016

[22] S. Vorapojpisut, "Model-based design of iot/wsn nodes: Device driver implementation," ICESIT-ICICTES 2018

[23] M. Brambilla, et al., "Model-driven development of user interfaces for iot systems via domain-specific components and patterns," Journal of Internet Services and Applications, 8(1), 2017

[24] B. Tekinerdogan and O. Koksal, "Pattern based integration of internet of things systems," ICIOT 2018

[25] M. Walker, et al., "Platibart: A platform for transactive iot blockchain applications with repeatable testing," M4IoT 2017

[26] V. Cardellini, et al., "Qos-based elasticity for service chains in distributed edge cloud environments," LNCS 10768, 2018

[27] M. Mongiello, G. Boggia, and E. Di Sciascio, "Reios: Reflective architecting in the internet of objects," MODELSWARD 2016

[28] M. Al-Taee, W. Al-Nuaimy, Z. Muhsin, and A. Al-Ataby, "Robot assistant in management of diabetes in children based on the internet of things," IEEE Internet of Things Journal, 4(2), 2017

[29] H. Khazaei, H. Bannazadeh, and A. Leon-Garcia, "Savi-iot: A self-managing containerized iot platform," FiCloud 2017

[30] A. Mazayev, et al., "Semantic web thing architecture," exp.at 2017

[31] A. Auger, E. Exposito, and E. Lochin, "Sensor observation streams within cloud-based iot platforms: Challenges and directions," ICIN 2017

[32] C. Wolff, et al., "Software architecture for an orc turbine - case study for an intelligent technical system in the era of the internet of things," ICIST 2017

[33] P. Jacob and P. Mani, "Software architecture pattern selection model for internet of things based systems," IET Software, 12(5), 2018

[34] V. Taratukhin, et al., "The internet of things prototyping platform under the design thinking methodology," ASEE 2018

[35] J. Holler, et al., "Toward a machine intelligence layer for diverse industrial iot use cases," IEEE Intelligent Systems, 32(4), 2017

Table 3. Detailed list of IoT patterns (Pe: Performance efficiency, C: Compatibility, U: Usability, R: Reliability, Se: Security, M: Maintainability, Po: Portability, Sc: Scalability, Pr: Privacy)

| Abstraction | Specificity | Pattern | Domain | Pe | C | U | R | Se | M | Po | Sc | Pr | Paper |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| High | General | Layered architecture for IoT applications | | X | | | X | X | | | X | X | [29] |
| High | General | Lambda-style architecture | | X | | | X | | X | X | X | | [31] |
| High | General | Kappa-style architecture | | X | | | X | | X | X | X | | [31] |
| High | Specific | Security Architecture (for Smart Water Management) | Smart Water Management | | | | | X | X | | | | [7] |
| High | Specific | Machine intelligence layer for industrial IoT | Machine Intelligence | | X | | | | | | X | | [35] |
| Middle | General | Alignment-based Translation | | | X | | | | X | | | | [14] |
| Middle | General | AS2AS Discovery of IoT Services | | | X | | | | X | | | | [14] |
| Middle | General | AS2AS Flow-based Service Composition | | | X | | | | X | | | | [14] |
| Middle | General | AS2AS Service Orchestration | | | X | | | | X | | | | [14] |
| Middle | General | D2D REST Request/Response | | | X | | | | X | | | | [14] |
| Middle | General | IoT artifact's Middleware Message Broker | | | | | | X | X | | | | [14] |
| Middle | General | IoT Artifact's Middleware Message Translator | | | X | | | | X | | | | [14] |
| Middle | General | IoT Artifact's Middleware Self-contained Message | | | | | | | X | | | | [14] |
| Middle | General | IoT Artifact's Middleware Simple Component | | | | | X | | X | | | | [14] |
| Middle | General | IoT Gateway Event Subscription | | | X | | | | X | | | | [14] |
| Middle | General | Orchestration of SDN Network Elements | | | X | | | | X | | | | [14] |
| Middle | General | IoT SSL CROSS-Layer Secure Access | | | X | | | X | X | | | | [14] |
| Middle | General | Translation with Central Ontology | | | X | | | | X | | X | | [14] |
| Middle | General | Entity-Component-Attribute | | | X | | | | X | X | | | [17] |
| Middle | Specific | Blockchain-based Architecture | Trusted Orchestration Management | X | | | X | X | X | | X | | [9] |
| Middle | Specific | Closed-Loop: Classical Closed-Loop Control | Industrial IoT | X | | | X | | | | | | [16] |
| Middle | Specific | Cloud-in-the-Loop: Closed-Loop Control via the Cloud | Industrial IoT | X | | | X | | | | | | [16] |
| Middle | Specific | Cloud-on-the-Loop: Cloud-configured Control | Industrial IoT | X | | | X | | | | | | [16] |
| Middle | Specific | Device-to-Device (D2D): Local Coordination | Industrial IoT | X | | | X | | | | | | [16] |
| Middle | Specific | Open-Loop: Classical Open-Loop Control | Industrial IoT | X | | | X | | | | | | [16] |
| Middle | Specific | Publisher: Sensor Data Publication | Industrial IoT | | X | | | | X | | | | [16] |
| Middle | Specific | Design pattern for computation offloading | Computation Offloading | X | | | | | X | X | | | [18] |
| Middle | Specific | Landline Interception | Emergency Information Delivery | | | X | | | | | | | [21] |
| Middle | Specific | SIM Equipped Device | Emergency Information Delivery | | | X | | | | | | | [21] |
| Middle | Specific | SMS to Display over Bluetooth/Wi-Fi | Emergency Information Delivery | | | X | | | | | | | [21] |
| Middle | Specific | SMS to Mobile Application | Emergency Information Delivery | | | X | | | | | | | [21] |
| Middle | Specific | Web System (for Emergency Information Delivery) | Emergency Information Delivery | | | X | | | | | | | [21] |
| Middle | Specific | Actor | Blockchain | | X | | | X | | | | | [25] |
| Middle | Specific | Operator-Controller-Module (OCM) | Organic Rankine Cycle Turbine | X | | | | | X | | | | [4, 32] |
| Low | General | Edge Code Deployment | | | | | | | X | X | | | [19] |
| Low | General | Edge Diameter of Things (DOT) | | | | | | | X | | | | [19] |
| Low | General | Edge Orchestration | | | | | X | | X | X | X | | [19] |
| Low | General | Edge Provisioning | | | | | X | | X | | | | [19] |
| Low | General | Frame Buffer | | X | | | X | | | | | | [22] |
| Low | General | Slot Buffer | | X | | | X | | | | | | [22] |
| Low | General | Application launch | | | | X | | | | | | | [23] |
| Low | General | Get details of a device | | | | X | | | | | | | [23] |
| Low | General | Get Information for one category | | | | X | | | | | | | [23] |
| Low | General | Get information from the device | | | | X | | | | | | | [23] |
| Low | General | Get state of the device | | | | X | | | | | | | [23] |
| Low | General | More devices more operations | | | | X | | | | | | | [23] |
| Low | General | More devices one operation | | | | X | | | | | | | [23] |
| Low | General | Nearby devices | | | | X | | | | | | | [23] |
| Low | General | One category more operations | | | | X | | | | | | | [23] |
| Low | General | One device more operations | | | | X | | | | | | | [23] |
| Low | General | One device one operation | | | | X | | | | | | | [23] |
| Low | General | One device one program | | | | X | | | | | | | [23] |
| Low | General | Pull information | | | | X | | | | | | | [23] |
| Low | General | Push information | | | | X | | | | | | | [23] |
| Low | General | Search device | | | | X | | | | | | | [23] |
| Low | General | Action interaction | | | | | | | X | | | | [30] |
| Low | General | Event interaction | | | | | | | X | | | | [30] |
| Low | General | Property interaction | | | | | | | X | | | | [30] |
| Low | Specific | Ontology Design Pattern for IoT Device Tagging Systems | Building Automation | | X | | | | X | | | | [10] |
| Low | Specific | Actuation-Actuator-Effect (AAE) ontology design | Brain-Computer Interaction | | X | | | | X | | | | [13] |
| Low | Specific | Stimulus-Sensor-Observation (SSO) ontology design | Brain-Computer Interaction | | X | | | | X | | | | [13] |
| Number of patterns that address the corresponding quality characteristic | | | | 13 | 17 | 20 | 14 | 6 | 31 | 6 | 8 | 1 | |

# 1st International Workshop on Software Engineering Research & Practices for the Internet of Things (SERP4IoT 2019)

**Title:** IOT based Air Pollution Monitoring and Control Mechanism using UAVs

**Authors:** [1]Dada Narayana Sashi Rekha, [2]Aditya Allamraju

Air pollution in India is a serious issue which is increasing rapidly and reaching to hazardous levels. One of the major sources being sulphur dioxide is emitted from the combustion of fossil fuels like coal and petroleum. Traffic congestion in India's most noticeable cities and towns is severe. It is caused due to increase in number of vehicles per kilometre of available road, chaos due to poor enforcement of traffic laws, obstacles in the road causing a blockage and merger etc. Complete lack of traffic sense in Indian public is the main reason for the chaos on the roads. Traffic congestion reduces the average traffic speed. At low speeds, it is scientifically proved that vehicles burn fuel inefficiently and pollute more per trip. In most of the highly congested Indian city roads, the average trip is less than 20 kilometres per hour. At such speeds, vehicles emit air pollutants 4 to 8 times more than they would, with less traffic congestion; and also consume a lot more carbon footprint fuel per trip than they would if the traffic congestion was less.

In context with the above issue in India, the present idea is to trace the vehicles which are emitting carbon dioxide ($CO_2$) and other hazardous pollutants beyond the permitted levels especially at the points of huge traffic congestion. The thermal sensors mounted on the Unmanned Aerial Vehicles (UAVs) or drones can be used to detect the vehicles that are emitting higher amount of pollutants (beyond threshold amount) and when detected should also snap an immediate picture of the vehicle registration number. The picture is then scanned using image scanning techniques and based on the registration number, the corresponding image will be uploaded into RTO website as evidence thereby updating the Government records. In parallel from the RTO website, the vehicle owner's details can be fetched. Email/phone number linked to any of the person's identification proof like that of Aadhar number/ PAN number or address linked to the registered number should be sent an E-mail/SMS/letter of the receipt of fine and asking them to get the vehicle repaired to emit lesser pollutants.

This system works with the assistance of the technology of Internet of Things (IOT) which is a rising technology based on the fusion of electronics and computer science. The embedded sensors in the system help to detect major air polluting gases such as $CO_2$, $SO_2$ and CO. The concept of IOT helps to access data from remote locations and save it in database so that we don't need to actually be present in that area. This would be a benefit to keep a constant check on pollution, in keeping the country greener and also to avoid corruption thereby saving a lot of money to the nation.

**Keywords:** Traffic congestion, Unmanned Aerial Vehicles, Internet of Things