General-Purpose Architectural Model for IoT-Based In-situ Monitoring Systems

Christian Nonyelum Ejike[1*], *Tebella* Mathaba[2], *Francois* Du Rand[3]

[1]Vaal University of Technology, Department of Electrical Engineering, Vanderbijlpark, South Africa
[2] Postgraduate School of Engineering Management, University of Johannesburg, South Africa,
[3] Vaal University of Technology, Department of Electrical Engineering, Vanderbijlpark, South Africa

Abstract. Over time, the Internet of things (IoT) discussion has come to prominence, and in-situ monitoring systems have been geared up with IoT services and deployed over IoT architectures. The integration of IoT services within system development has enriched many monitoring application studies but the architectural models used in majority of these studies trivializes several key components of a fully functional IoT architecture. This paper proposes a general-purpose architectural model (GPAM) that can be used in the deployment of any in-situ monitoring system. The proposed architectural model is successfully implemented using a single domain use-case (water assessment) as a conceptual proof. The traditional water assessment processes are transformed into IoT processes in an attempt to reach the same result in a more efficient manner. This template can help prospective developers build and engineer robust IoT monitoring systems.

1 Introduction

The Internet of Things (IoT) is a coined phrase for the concept of bringing animate and inmate objects and their environment together via the internet. There are generally three layers of an IoT system namely perception, communication and application layers which spans into several more sub-layers [1-5]. Continually, priority is given to one or two layers in many studies, while others are only mentioned in passing. The proposed architectural solutions of several researchers include few layers that do not capture the entire function of an IoT architecture and thus, cannot be re-used for other applications despite addressing similar problem domains. Since there is no guiding architecture standard [4], IoT system designs routinely follows a contextual architecture that fits a particular set of specifications and requirements such as home automation, healthcare systems, precision agriculture, environmental monitoring, etc. [6,7]. While the domain is not the issue here, most implementation do not capture all the necessary functionalities especially in in-situ monitoring.

* Corresponding author: chritsiane@vut.ac.za

Although studies may be presented for in-situ monitoring and ideally it is expected that different infrastructural elements and how they function together to achieve their monitoring goals are included in their publications, this is not usually the case. Most of the time, researchers are specialists in their fields and their research focus only on their areas of expertise. For example, a programmer would ideally focus his study on the programming aspect of the system; a researcher who is adept at mathematical algorithms may focus on the modelling aspect of sensors and energy harvesting [8-10], while totally ignoring other aspects like data storage and manipulation. A database engineer for instance may focus on data streamlining, while network engineers will typically present studies on the wireless communication and overlook the importance of modelling the sensors and energy harvesting for accuracy, or on the visualization aspect to enable target end-users' interaction with the system for better understanding and so on. This in actuality is an incomplete representation of a fully functional IoT system as the overlooked functional areas of their systems form part the crucial criteria (discussed in section 2) that qualifies a good IoT System architecture. Moreover, several architectures are not scalable, and most studies totally omit the cyber security aspect of their systems which poses a risk of data theft, or unintentional violation of regulatory rules like the Protection of Personal Information Act (POPI Act or POPIA) and expose their system to different cyber security threats.

For the above-mentioned issues, it is deemed necessary to investigate important aspect of a basic IoT architecture that can be reused for different in-situ monitoring applications. This study submits a bona fide, certifiable general-purpose architectural model (GPAM) that can be used to qualify an ideal IoT system design. The study aims to identify and enumerate necessary basic functions and components that can form part of any architectural model irrespective of domain application. The added value of the GPAM is demonstrated in the development and implementation of an in-situ water monitoring system that integrates IoT services. Each layer of the GPAM architectural model is meticulously simplified and the integration mechanism with other IoT systems is delineated. Data acquisition, storage, manipulation, transfer and protection is also discussed. Suggestions are also given on how to apply business logic, AI, smart algorithms, automation, real-time monitoring, analytics, alerts, and reports to the model. Industries, researchers, students, commercial and Do-It-Yourself (DIY) developers can use this model as a blueprint for future system designs

The rest of the paper is structured as follows: Section 2 outlines the proposed GPAM and an evaluation model to follow when implementing it. Section 3 describes the model implantation and shows how alignment to the model can be achieved. Finally, section 4 concludes the presented work and gives envision for future work.

2 Proposed General-Purpose Architectural Model

An architecture refers to the high-level design model of a system. It is a depiction of how the base of the system is structured and maps out in detail all the individual elements that form part of the entire system. It describes how the elements are arranged and connected to deliver services and achieve the targeted functionality of the system. IoT architectures typically follow this structural pattern. Elements of an IoT system architectures are either centralized or decentralized [11]. In the centralized design, devices connect to IoT platforms over the internet and the services to these devices are managed centrally at the backend. Real-time analytics, event management, alerting is some of the core tasks performed by the said IoT platforms. On the other hand, decentralized architectures refer to those which necessitate autonomous communication between devices with no central management such as in peer-

to-peer communications. Here, system functionality can be achieved without the need for an internet connection or central management.

There is no one globally accepted IoT design that everyone agrees upon. Different scholars have presented several architectural layer designs ranging from three to six-layer design [1-7] and even a seven-layered breakdown have been presented [12]. However, the most accepted layer design is the three-layer architecture as shown in Figure 1.

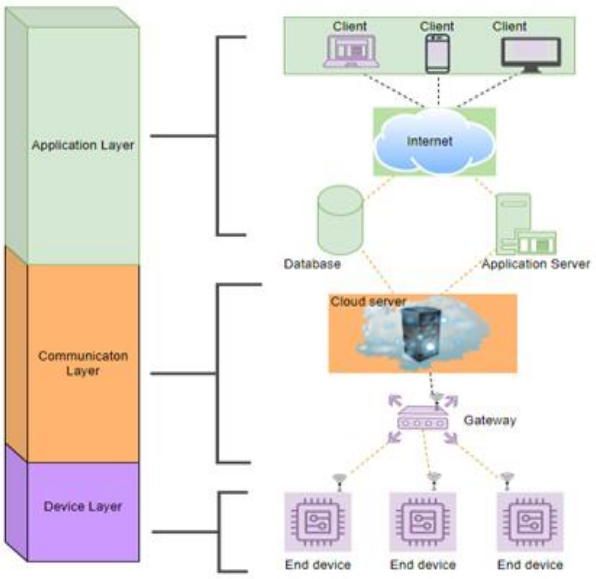

Figure 1: General-purpose architectural model

These layers are interdependent and sometimes components of each layer overlap to form the unified ecosystem solution. From a top-bottom view, this architectural model has three layers namely application, network and perception layers

(i) *Application Layer* - this layer provides a method for hosts to be interfaced to deliver specific services and manages end user interaction with the system. It handles data formatting and presentation. Over and above that, it defines the contextual in-situ monitoring domain for which the system is deployed and denotes the software through which devices operate. This layer also facilitates with the real-time analytics, integration, and reporting features.

(ii) *Network (Communication) layer* - this layer defines the agreed-upon mechanism for end-to-end and message exchange across the network. It is responsible for data transportation across all layers of the architecture and assures that data transmission from a sender to a receiver is achieved via appropriate routing and transport protocols. It is also referred to the middleware layer as it is the intermediary between the bottom and top layers of the IoT architecture. It is a core layer for the system without which any IoT system is considered futile. In the GPAM, it is referred to as the *Communication layer* for linguistic simplicity and to avoid any connotative misinterpretation. Sometimes the word "network" may suggest other explicit meaning that includes physical devices. For this

reason, "communication" is used emphasises on its function and separates the connotative confusion.

(iii) *Perception (Device) layer* - this layer embodies all the hardware elements (microcontroller, sensors, actuators and client gadgets etc.,) and their sundry dispersion to sense and gather information about their immediate environment and act as service executors such as actuating duties which are defined per configuration. It also enables localisation accuracy and geographical mapping which is an essential service for IoT applications. Device statuses and their physical location can be obtained and monitored irrespective of where they are around the world. In the GPAM, it is referred to as the *Device layer.*

Now that the GPAM architecture have been presented, the study aims to outline a set of criteria that must be satisfied and provide a checklist of basic questions that can help researchers achieve good IoT system functionality.

2.1 Evaluation Framework for GPAM

Again, there are basic, yet essential requirement needs to be met when designing and implementing a certifiable IoT system architecture. In this section, the authors present the *D.R.E.A.M.S* framework shown in Figure 2. It delineates those key requirements and provides suggestions on how to satisfy them. The *D.R.E.A.M.S* framework reduces the risks and increases the success rate for any in-situ project.

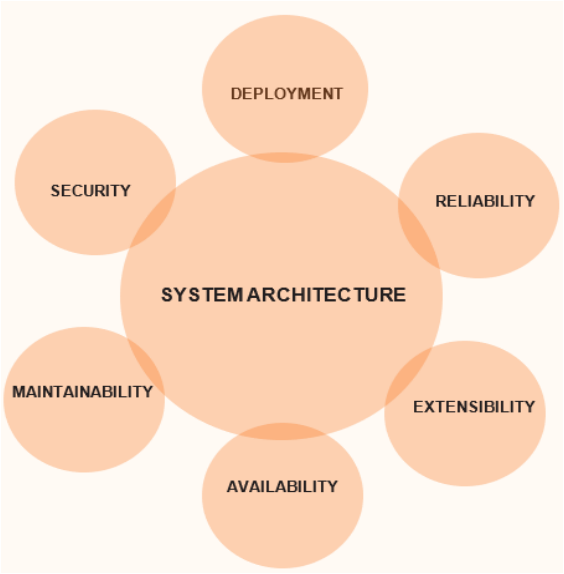

Figure 2: The D.R.E.A.M.S Framework

i. Deployment

It is believed that the expected number of devices connected to the internet may reach an amount that may exceed 1 billion devices over time [2, 3]. Thus, every IoT architecture should be designed and built to accommodate such an increase in device numbers. Similarly, when discussing IoT system applications, attainable distance and energy consumption

models are important facets. A defined deployment strategy is important to decide depending on the system application. System applications are diversifying over time, thus, it is important to plan a deployment strategy that meet these functional requirements. Several deployment concerns have emerged over different studies and the authors recommend considering the following: scalability, flexibility, interoperability, energy-efficiency and security. These are defined in Table 1.

Table 1: Consideration for better system deployment

Terms	Definition
Scalability	The ability of the architectural design to support more devices and architectural resources without any significant degradation in the system performance.
Flexibility	The ability for system to be used for different purposes without without any visible structural changes or system behaviour. Can be achieved through loose coupling.
Interoperability	The ability of the system to seamlessly communicate, integrate and share information with other related systems.
Energy-efficiency	The system should adhere to lower power consumption needs of IoT. The more power-saving mechanisms are applied, the better. Techniques such as device sleep routines, battery clustering, data normalisation, homogeneous data aggregation, coding modulations, use of renewable sources, asynchronous messaging etc., may be employed
Security	The ability to protect or guard against attacks and damage.

ii. Reliability

Just like friendships are strengthened by how dependable two people are to one another, good IoT systems architectural designs should have certain level of trustworthiness too. Reliability is a system's functional capacity to perform consistently under prescribed conditions over an extended length of time. For example, a system designed and expected to detect water contamination within a given timeframe should precisely return threshold level readings of the given water parameter for which the sensor(s) have been correctly interfaced and configured. Should the system return unknown or unrelated readings within that timeframe despite being correctly configured and deployed, the system cannot be trusted, and the data may be considered invalid.

Reliability is a determinant of the resilience, and consistency in which transactions are processed without any noticeable failure irrespective of the increase or decrease in number of events. It also provides an assurance of the system integrity as expected output results remain unchanged for the same input data. Reliability can be measured using certain metrics such as Functional Correctness, Mean Time Between Failures (MTBF); Responsiveness, Rate of User Error, Flaws Over Time etc. [13]. These are defined in Table 2.

Table 2: Metric consideration for better architectural reliability

Terms	Definition
Functional Correctness	Degree of accuracy and precision for system results
Mean Time Between Failures (MTBF)	The average time between system failures under normal operating conditions

Responsiveness	The reaction period to complete each system request
Mean Time to Repair (MTTR)	The average time to fix and reinstate system to optimal working condition after failures
Flaws Over Time	Overall system defects/vulnerability within a given time interval

iii. Extensibility

This is the ability to tolerate additional features to system functionality or modify existing system functions without impacting existing system performance and needing structural rework. It provides functional diversity, reusability of movable parts, plug-and-play device integration and bolster adaptability of the system. These are expanded upon in Table 3.

Table 3: Consideration for better architectural extensibility

Terms	Definition
Functional diversity	System functionality can be pivoted for different context requirements.
Reusability	The base architecture can be utilised repeatedly
Plug-and-play	System components can easily be added or removed through system peripheral
Adaptability	The system should cope with changing environment
Integration	The ability of a several dependent and independent subsystems and sub-components to combine into one larger system whilst maintaining cohesion

iv. Availability

The percentage of time the system infrastructure remains in an optimal operating state under ideal working conditions in order to serve its intended purpose. It is a good determinant of how well the system will deliver services in the future. Table 4 identifies system areas that help with ensuring system availability

Table 4: Authors considerations for better architectural availability

Terms	Definition
Storage	System data needs to be properly persisted into a database for trend analysis and real-time access.
Uptime	The system needs to be in readily functional in order to achieve constant in-situ monitoring. Any downtime can impact the availability of devices.
Fault Tolerance	System should be configured with a backup strategy to avoid and mitigate against degradation. Hardware and data redundancy may be applied
Risk monitoring	System health like microcontroller CPU usage should be monitored in real-time
Visibility checks	Apply status check and device discovery mechanisms to keep track of when devices are down

v. Maintainability

The ease with which system faults can be remediated. Table 5 outlines areas of consideration when planning for system maintenance.

Table 5: Authors consideration for better architectural maintainability

Terms	Definition
Cost	Financial implication of repairing the system within a set period of time
Management	Level of administrative controls that can be applied
Time	How long it takes to maintain the system
Modularity	System decomposition into smaller parts
Programmability	Ability to alter system behaviour though logic instructions

vi. Security

Every system must be protected against any form of vulnerability attack. This is of paramount importance and has become an inevitable requirement for any IoT environment [14-16]. A good IoT architecture must be built robustly to mitigate against common cyber security threats. It is advisable that security be applied at every possible layer of the entire system architecture.

Security is important for all IoT system architectures to mitigate against data and system loss, ensure data integrity and confidentiality. Poor security may lead to the theft of sensitive information. For instance, if a water monitoring system has been successfully bridged and data has been maliciously tampered without notice, it may mislead users, and communities about the safety of water supply in environmental pollution studies which in turn endangers lives. Security breaches can tarnish a reputation and jeopardize system trustworthiness if not properly protected. In addition, regulatory violations may attract penalties both financially and in form of permanent exclusions. Table 6 enlists some best practice to ensure security:

Table 6: Authors consideration for better architectural security

Measures	Definition
Clearly defined System Roles	User roles must be distinctly defined, and authentication mechanisms put in place to granting access. Mechanisms such as the AAA framework is a good example.
Compliance	System operations should comply with any set regional rules and regulations partaking to their system implementations
End-To-End Encryption	Secure communication should be implemented an used across all communication paths; Using a standard secure transport layer encryption technology like HTTPS, SSL, TLS etc.,

Hardening	Client applications should be reinforced using modern device hardening techniques [16] against hacks such as Session hijacking, Cross-site scripting, Injection, Insecure deserialization, Broken access control etc
Patch Management	There should be a strategy for update and security patches to components such as devices, software, operating system etc.

3 Model Implementation

To implement the model, the study focused on developing a water quality monitoring system. The system was developed using the proposed architecture and passed the testing operation successfully. Figure 3 shows a high-level conceptual model of the test system

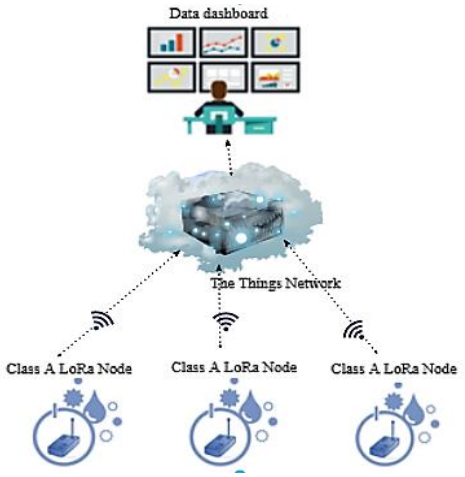

Figure 3: Conceptual implementation model

On a pragmatic level, the goal is to offer researchers, hobbyists, manufacturers, developers and entities alike who are interested in water quality monitoring, an innovative solution that could economically revolutionize the water assessment industry. For businesses who want to commercialize this model, the implementation is cost-effective as capital cost are low and majority of capital costs are low. Enthusiasts with financial constraints who which to replicate it for educational or personal purposes, can use low-cost technology and open-source IoT solutions as will be described in this section. On a technical level, the GPAM is a simplified set of architectural modules and is distributed over layers of the system's IoT infrastructure as presented in Figure 1. To explain how the tested system fits into the GPAM design, abstract elements of the system are mapped within the GPAM layers.

i. Application Layer composition – The Things Stack (TTS) [17] was used as the IoT application.

ii. Communication layer composition – the LoRaWAN protocol [18] was selected and used as the preferred data transmission mechanism for the system.

iii. Device Layer Composition – the hardware components of the system include the SEN0161 [19], the DS18B20 [20] and the SEN0189 [21] sensors which were used for water quality sensing. These sensors are responsible for sensing pH, temperature and turbidity respectively. The microcontroller used is the Semtech SX1276 ESP32 [22]. The microcontroller computed the data and prepared it prior to transmission. The client devices used are a smart phone and a laptop.

3.1 System Operation

The system logic was developed using the Arduino C programming language on a PC and the program was uploaded onto the ESP32 via console peripherals. The MCU was configured and connectivity to the public cloud TTS server was facilitated using Wi-Fi technology. On the TTS Application server, application data were processed, and some business logic were implemented. The LoRaWAN protocol used is one of the long power wireless area network (LPWAN) technologies leading the wireless network industry and its intelligent modulation techniques was crucial in maintaining energy efficiency.

TTS has all the necessary features to satisfy the functions of the Application layer within the system's IoT infrastructure. The TTS platform has a Graphical User Interface (GUI) for data presentation. It also has a built-in data logging unit which records activities within the system. Several IoT integrations with the TTS Application server were setup and activated such as DataLake and Blynk which were used to automate smart system alerts/notifications and scheduling reports. This integration allows client device applications to continuously receives real-time update of the sensor readings, and per system configuration, the alerts would be fired, and users would be instantly notified via an email, SMS, or on the application dashboard for them to take remedial actions to control the situation. In the test scenario, the alerts would only fire when predefined sensor reading thresholds are neared, reached or exceeded. Similarly, reports are scheduled on a weekly and monthly basis rather than hourly or daily to avoid inbox storming. The reports are important for trend analysis.

The system's compact size, modular components, and inexpensive cost make it portable, replaceable, and easy to maintain. Therefore, adding a degree of safety to the whole network. Real-time measurements and monitoring speed up turnaround and improve decision making, hence, confirming its reliability. Authors estimate that the system can continuously function for a lengthy period of over 3 years on a 12V powered battery and even longer on a grid power. This is achievable since the devices are programmed for deep sleep routine and only wake up to transmit when necessary.

The proposed design is very cohesive and loosely connected in areas such as communication via the Application layer. The proposed GPAM can be integrated with other existing IoT systems via the Application layer. Numerous protocols and modulation techniques can be applied on the communication layer, and several channels can be utilized for data transfer. More sensors can be added for other quality parameters. In addition, the system functionalities can be extended in multiple ways.

3.2 Result: System Alignment to Framework

The system follows the D.R.E.A.M.S framework in the following manners:

Table 7: Alignment for Deployment

Framework Element	Achievement strategy
Deployment	• More nodes were added successfully • Communication with other systems was achieved with ease • The data was persisted using TTS cloud database. In addition, a dedicated MySQL database also persisted data for backup purposes. • LoRa uses a chirp spread spectrum (CSS) modulation technique with is notable for its energy saving mechanism [16]. • Class A devices were deployed for optimal energy consumption. In addition, the microcontroller was configured with a sleep routine to conserve power usage. • Sensors of similar types were replaced without any configuration changes and system perform correctly • Different security measures were applied (refer to the Security)

Table 8: Alignment for Reliability

Framework Element	Achievement strategy
Reliability	• The sensor readings are obtained in real-time with an interval of 10 seconds for each parameter measurement. Each sensor test was repeated 500 times amounting to 5000 seconds and a total of 15000 seconds for the three sensors. The system performed consistently without any functional error or noticeable degradation. • No system defects were detected for the entire testing duration of 2 months • The system could detect signals as low as -120 dBm • Sensors of similar types were replaced without any configuration changes and system perform correctly • Communication from sender to receiver took less than 60 microseconds and delays did not exceed 60 seconds

Table 9: Alignment for Extensibility

Framework Element	Achievement strategy
Extensibility	• System was successfully integrated with DataLake and Blynk which were used to automate smart system alert/notifications and scheduling reports. • Three different water sensors were used simultaneously on each node, and they were able to accurately measure and obtain different water quality parameters.

	• In total there are 15 general peripheral input/output (GPIO) ports on the microcontroller, which can be used to add more sensors. • Sensors of similar types were replaced without any configuration changes and system perform correctly • The base architecture can be used for a completely new environmental monitoring application such as air quality monitoring, smart farming or medical system applications by replacing the sensors.

Table 10: Alignment for Availability

Framework Element	Achievement strategy
Availability	• System remained up and functional throughout the test period. Every downtime was planned like in the case of component replacement • The TTS application portal has a dashboard that automatically detect and indicate device status. • LoRaWAN application portal has a geo-localisation setting which autodetect and maps device locations • System was successfully integrated with DataLake and Blynk which were used to automate smart system alert/notifications and scheduling reports.

Table 11: Alignment for Maintainability

Framework Element	Achievement strategy
Maintainability	• Low-cost devices and open-source software were used • The system has an administrative portal through its application where controls may be applied and only users with administrative rights may access it. • The authors tried to replace some components and were able to reinstate system functions within 5 minutes. • The system consists of different subsystem and components with distinct functions. • Multiple logic instructions were tested on the system, and all were successfully executed. In addition, different programming language were tested such as C++ and Python

Table 12: Alignment for Security

Framework Element	Achievement Strategy
Security	• LoRaWAN provides a wide range of authentication and authorisation tools, approaches, and procedures. • The system security features, such as the AES-128 cryptography algorithm and join authentication, enable end-to-end data encryption against cyber security threats, ensuring that only authorized devices may communicate • Different user profiles were defined with different access and functional rights

	• The system is compliant with regional frequency usage of South Africa. LoRa is configured to operate within the EU863 - EU870 (MHz) ISM band • Error checks are applied though the LoRaWAN CRC header • Form validations were implemented to harden the system and prevent cross-site scripting and injection attacks • System is physically protected against adverse environmental conditions using an IP68 waterproof enclosure

4 Conclusion

This paper proposed and presented a general-purpose architectural model (GPAM) for IoT systems and described how to apply it for different application domains. The study reviewed different architectural designs for IoT systems, and their layers were summarised into a concise, easy to follow model. In addition, the authors extracted basic functional requirements and common system behaviours for in-situ monitoring systems. This led to the development of a novel criteria framework they called D.R.E.A.M.S. Thereafter, the viability of the model was tested using a developed water monitoring system in order to ensure it performs as expected in a real-world environment. The result is presented in tabular form in Section 3.2 which shows the system's practical alignment with the framework. The Achievement strategy describes how each component of the conceptual model satisfies the GPAM criteria and how easy integration of the functional data with third-party applications is achieved. The result is satisfactory, the architectural model is technology-agnostic and reusable for different application domains thus it is easy to replace or replicate using different IoT components. The paper forms a foundation for a future project which will include layout a rating system for IoT alignment with the framework. It also provide both novice and established researchers ideas on how to integrate our architecture to their system and tailor it to align with our proposed framework. Finally, our proposed architectural and framework solution can be pivoted to cover several contextual domains and meet different business requirements across different industries.

References

1. Burhan, M., Rehman, R.A., Khan, B. and Kim, B.S. 2018. IoT elements, layered architectures and security issues: A comprehensive survey. *Sensors*, *18*(9), p.2796.
2. Sethi, P. and Sarangi, S.R. 2017. Internet of things: architectures, protocols, and applications. *Journal of Electrical and Computer Engineering*, *2017*.
3. Madakam, S., Lake, V., Lake, V. and Lake, V. 2015. Internet of Things (IoT): A literature review. *Journal of Computer and Communications*, *3*(05), p.164.
4. Weyrich, M. and Ebert, C. 2015. Reference architectures for the internet of things. *IEEE Software*, *33*(1), pp.112-116.
5. Wu, M., Lu, T.J., Ling, F.Y., Sun, J. and Du, H.Y., 2010, August. Research on the architecture of Internet of Things. In *2010 3rd international conference on advanced computer theory and engineering (ICACTE)* (Vol. 5, pp. V5-484). IEEE.

6. Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H. and Zhao, W. 2017. A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications. *IEEE internet of things journal*, *4*(5), pp.1125-1142.
7. Ray, P.P. 2018. A survey on Internet of Things architectures. *Journal of King Saud University-Computer and Information Sciences*, *30*(3), pp.291-319
8. Cloete, N.A., Malekian, R. and Nair, L., 2016. Design of smart sensors for real-time water quality monitoring. *IEEE access*, *4*, pp.3975-3990.
9. Lambrou, T.P., Anastasiou, C.C., Panayiotou, C.G. and Polycarpou, M.M., 2014. A low-cost sensor network for real-time monitoring and contamination detection in drinking water distribution systems. *IEEE sensors journal*, *14*(8), pp.2765-2772.
10. Ulukus, S., Yener, A., Erkip, E., Simeone, O., Zorzi, M., Grover, P. and Huang, K., 2015. Energy harvesting wireless communications: A review of recent advances. *IEEE Journal on Selected Areas in Communications*, *33*(3), pp.360-381.
11. Yaqoob, I., Ahmed, E., Hashem, I.A.T., Ahmed, A.I.A., Gani, A., Imran, M. and Guizani, M., 2017. Internet of things architecture: Recent advances, taxonomy, requirements, and open challenges. *IEEE wireless communications*, *24*(3), pp.10-16.
12. Lombardi, M., Pascale, F. and Santaniello, D., 2021. Internet of things: A general overview between architectures, protocols and applications. *Information*, *12*(2), p.87.
13. Klima, M., Rechtberger, V., Bures, M., Bellekens, X., Hindy, H. and Ahmed, B.S., 2020, December. Quality and reliability metrics for IoT systems: a consolidated view. In *International Summit Smart City 360°* (pp. 635-650). Springer, Cham.
14. Olivier, F., Carlos, G. and Florent, N., 2015. New security architecture for IoT network. *Procedia Computer Science*, *52*, pp.1028-1033.
15. Varshney, T., Sharma, N., Kaushik, I. and Bhushan, B., 2019, October. Architectural Model of Security Threats & theirCountermeasures in IoT. In *2019 international conference on computing, communication, and intelligent systems (ICCCIS)* (pp. 424-429). IEEE.
16. Woschek, M., 2015. Owasp cheat sheets. OWASP Foundation, pp.1-315.
17. The Things Industries. The Things Stack. https://www.thethingsindustries.com/docs/. [Accessed May 2022]. 2022 (cit. on p. 21).
18. LoRa Alliance, 2015. White paper: A technical overview of LoRa and LoRaWAN. *The LoRa Alliance*: San Ramon, CA, USA, pp.7-11.
19. DFRobot. 2019. *"PH meter (SKU: SEN0161),"* [Online]. Available at: https://www.dfrobot.com/wiki/index.php/PH_meter(SKU:_SEN0161)#pH_Electrode_Characteristics.
20. Maxim Integrated, 2018. *"DS18B20 Programmable Resolution 1-Wire Digital Thermometer."* [Online]. Available: https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf. Accessed Jan. 1, 2021
21. DFRobot. 2019. *Turbidity Sensor SKU: SEN0189.* [Online]. Available at : https://www.dfrobot.com/wiki/index.php/Turbidity_sensor_SKU:_SEN0189.
22. Anzeigen, M., 2018. ESP32 TTGO. [online] Esp32-ttgo.blogspot.com. Available at: http://esp32-ttgo.blogspot.com.. [Accessed 22 July 2021].