NTA-IoT: A Flexible and Modular Architecture for IoT Applications

Nelson Marcelo Romero Aquino, and Sidney Viana,
Fundación Parque Tecnológico Itaipu Paraguay (FPTI-PY)

***Abstract*—This work introduces the NTA-IoT, an abstract IoT architecture that is intended to be flexible and modular, in order to serve as a generic guide tool for developing IoT projects.The model is composed of two types of layers: base layers and support layers, with each layer providing a particular service in the IoT infrastructure. Some considerations and possible roles for building IoT projects following the NTA-IoT architecture are also detailed. Finally, this paper presents an example of a concrete implementation of the architecture on a home automation problem, based on the use of different technologies such as KNX, MQTT and the Outsystems platform to build a functional end-to-end IoT project. The correct functioning of this implementation shows that the architecture is consistent and well-suited to serve as strategic guide to carry IoT projects with satisfactory results. Although the example presented in this work is applied to home automation, the framework is hoped to be applicable to any type of project.**

***Index Terms*—Internet of Things, IoT Architecture.**

I. Introduction

THE Internet of Things (IoT) is based on the interconnection of everyday or industrial objects through various platforms that allow these objects to be managed and, mainly, to analyze the data generated in order to guide users in the decision making. There are two types of IoT, one based on the interconnection of elements through the Internet (Internet of Things) and one in which devices are integrated only using a local network, within a company, home or infrastructure (Intranet of Things).

IoT is considered the future of internet communication between machines [1]. The number of devices connected to the internet grows each day **introducing** the necessity to integrate those devices in order to have synchronized control systems and to gain insights from all the data collected by those devices. There is also an increased need for providing means of connectivity for those devices that originally do not have that capability. In this scope, there is a need for proper IoT architectures that establish standardized paths to be followed when carrying the development of an IoT project. Aspects such as the heterogeneity of the devices and their working protocols, and means of communication for these.

Following this tendency, during the recent years, several works introduced IoT architectures devised to solve different problems, with the common factor of the interconnection of devices. For instance, another research work [2] presented a architecture to perform real time interaction between mobile clients and smart things such as sensors and actuators. The method performs dynamic discovery of connected devices and endpoints, allowing to connect non-smart devices connected over Modbus [3]. The data from the devices are represented using the Sensor Markup Language (SenML), that had its capabilities extended to support actuator control from the clients. I. Ungurean and N. Gaitan [4] proposed an architecture based on OPC.NET specifications[1], which define a set of interfaces to communicate client and server applications through Windows Communication Foundation (WCF). The architecture is composed of two layers or modules: the data server and the application. The former acquires data from the sensors whilst the latter is a client executed on PCs, tablets or smart-phones. The paper also proposed the development of a gateway to perform real-time data acquisition, which can acquire data from different types of fieldbuses such as ZigBee, CANOpen, among others, and it can be connected to an OPC.NET server via Modbus TCP/IP or Modbus RTU. Following this line, based on defining an architecture considering the technology to be applied as base, an IoT architecture for automatically monitor and track patients, personnel and biomedical devices within hospital and nursing institutes was proposed [5]. The model is based on the use of RFID, WSN and smart mobile devices operating through an network infrastructure that combined CoAP, LoWPAN and REST. The architecture, named Smart Hospital System (SHS), which has its core in an IoT Gateway that provided access to the user interfaces with a layer that composed of a secure access manager, an application manager and a 2-way proxy, receiving and acting on the devices from the hybrid sensing network proposed in the work.

In contrast to these architectures, which are designed to be applicable for specific domains, other works introduced more general frameworks. For instance, [6], which introduced an architecture structured into a secure API, a backbone and device networks with interface to the backbone. The layers of the architecture are based on the OSI stack. A similar layered structure, was proposed by Dai and Y. Wang [7], with four layers: Application, Internet, Adaptation and Things. This abstract-layered architecture was also followed in another previously research work [8], introducing the Distributed Internet-like Architecture for Things (DIAT), devised to mainly address issues regarding the heterogeneity of IoT devices, by defining a three-layered model with a core that consists of an IoT Daemon, which also contains some internal layers. This architecture aimed at addressing aspects such as scalability,

**Email address:* nmarceloromero@gmail.com (Marcelo Romero).
sidney.viana@pti.org.py (Sidney Viana).

[1]https://opcfoundation.org/about/opc-technologies/opc-net-4-0/

interoperability, security, automation, among others.The model presents a high abstraction level that could add complexity to the process of developing an IoT project. Following this line, the research work proposed by J. Ren, H. Guo, C. Xu, and Y. Zhang [9] introduced a transparent computing based architecture, which consists in allowing users to select their desired services on-demand, avoiding them to be concerned about details such as their installation or configuration. It is highly focused on offering device management through the means of cloud computing. The proposed framework is composed of five layers: End User Layer, Edge Server Layer, Core Network Layer, Cloud Layer, and Management and Interface Layer.

Some architectures proposed in the works mentioned previously are focused on very concrete applications, defining fixed data formats and technical standards to be applied when they are implemented. This makes them strongly dependent on the technologies in which they are based on, making them useful only for problems of specific domains. On the other hand, other models provide a much higher abstraction level. However, most of them add complexity to IoT projects through layers that may not be concern-focused. Indeed, there is a need for a generic architecture focused on atomic and specific concerns, that could be further extended for specific applications, if needed.

To cope with the limitations described before, this paper proposes a generic, flexible and modular architecture, to serve as conceptual guide to develop IoT applications and projects with independence from the protocols, programming languages, operating systems or any other restriction derived from the state of the technology at a certain period of time. The architecture is intended to have a simple structure with a fair level of abstraction, differently from more recent architectures that could have high abstraction that adds more complexity to the model. For this purpose, our proposed architecture is designed as an evolution of the one presented by G. Dai and Y. Wang [7], introducing modifications such as adding two more layers that act as support for the main layers, and describing possible protocols and technologies that could be used when carrying out IoT projects. We also differentiate between two types of layers: the base layers (physical, aggregation, exchange and application) and the support layers (storage and security). The base layers are mandatory through the architecture, whilst the support layers may be included only if necessary, although their use is recommended. We consider that this could be a step forward towards simpler yet robust IoT projects based on the proposed architecture.

Considering the aspects discussed before, the highlights of this paper are:

- To provide a simple abstract IoT architecture, named NTA-IoT, to be guide the implementation of IoT projects.
- To introduce roles in an IoT project based on the NTA-IoT architecture.
- To describe a real-world implementation of NTA-IoT for a simple home automation scenario.

This paper is organized as follows. Section II describes the proposed architecture. Section III details recommendations when developing IoT projects and introduces the roles devised for the NTA-IoT framework. Finally, section presents conclusions and future research perspectives.

II. NTA-IoT Architecture

The proposed scheme is composed of four base layers: Physical, Aggregation, Exchanging and Application. There are two other layers, which we consider as support layers, acting as complement to the base layers: the Security layer, which will vary its scope and methods depending on the layer on which it acts, and the persistence layer (Storage), that allows to store the data in different manners according to which layer is applying the persistence strategies. Figure 1 presents an overview of the proposed architecture. The following sections present the characteristics of each layer individually.

In addition to the description of each layer, we introduce a concrete implementation example of the architecture for a Home Automation scenario. Home Automation Systems [10] aim at achieving full home control through different means. The main idea is to have control over physical equipment from a building, such as its lights, security systems, thermometers, air-conditioners, access control systems, among others. IoT pursuits the integration of all these devices physical device that may be smart or not, by connecting them through the internet or at least a local network. Figure **??** presents the concrete case and the technologies used to reach the solution.

A. *Physical*

The physical layer encompasses different devices such as sensors, actuators, mobile phones, GPS, surveillance cameras, among others. There are several manufacturers and even device standards that can be part of a devices ecosystem. The main characteristic of the components of this layer is its heterogeneity, there are several protocols to manage the devices, each with its own properties and functions that have to be taken into account when structuring an IoT project.

The devices could be smart, which implies that they could be directly connected to a network and they are usually programmable by providing user interfaces for such task; and they could also be not smart, which are non-programmable devices that do not have direct means to be connected to a network. Devices that are not smart require a gateway to be able to be connected to the other devices. This is tackled in the Aggregation layer, described in Section II-B. Note that this definition of smartness is specifically regarding connectivity capability, for a more detailed analysis of the characteristics of smart systems the reader can refer to [11].

This layer will address the installation, configuration and basic administration of those devices. It will also determine the grouping strategies and the basic operating rules, if it is necessary and if their respective protocols allow it. It is important to note that this layer does not deal with the interconnection of devices of different types or protocols, so each type of equipment is treated individually.

Implementation Example. The physical protocol chosen for this layer is KNX[2], which is a communication standard

[2]https://www.knx.org

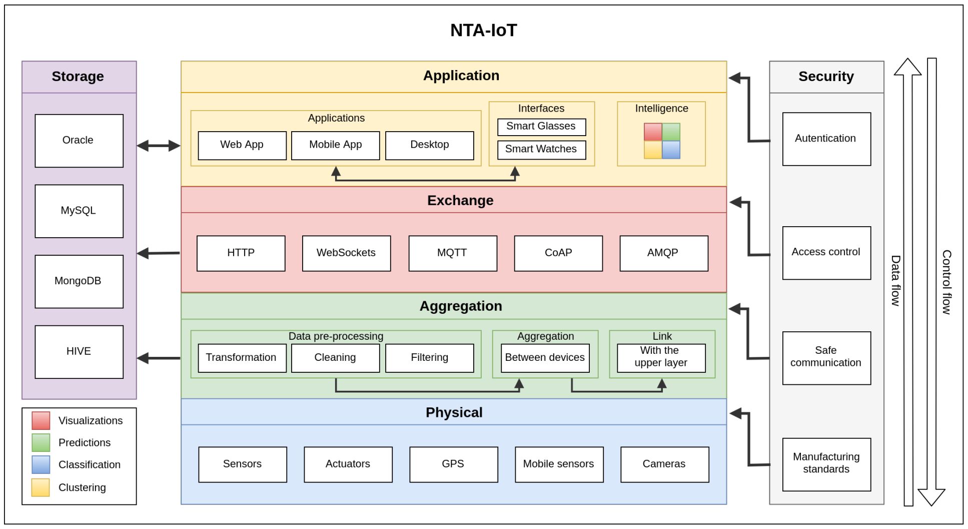

Fig. 1: Overview of the proposed framework.

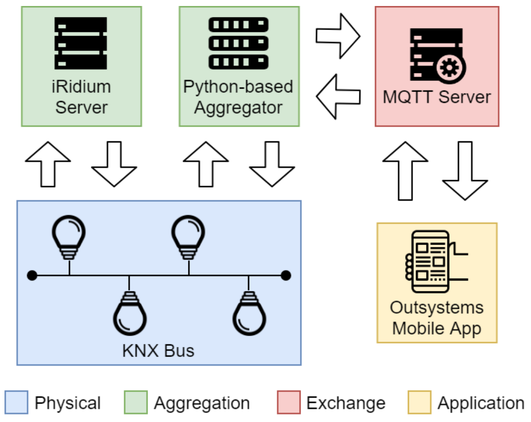

Fig. 2: Overview of the technologies used for the implementation of the example.

specifically devised for home automation. It defines specifications related to: the proper KNX protocol (including security), different media, configuration modes and applications. During the recent years, the KNX Association have been pursuing to integrate the KNX standard to be prepared to support IoT applications.

For our implementation, we use the lights of a room and a counter that defines the number of times a switch was pushed. The lights receive and send only 1-bit signals (OFF/ON), whilst the switch counter handles integer values of length equal to 8-bytes. The devices were configured using the ETS5[3] software for KNX, which allows to program devices connected to the KNX bus, and also assigning addresses for each on them. ETS5 also permits to create groups of devices that can be identified through a certain address, which is useful when the user aims at controlling several devices at once.

Among the several available KNX devices, a KNX/IP interface acts as a converter of signals coming from KNX devices to be able to communicate through some IP protocol (UDP/IP, specifically). This device was used to connect, through tunneling, the KNX devices to a server running in the Aggregation layer.

[3]https://www.knx.org/knx-en/for-professionals/software/ets-5-professional/index.php

B. Aggregation

The Aggregation layer groups several sub-layers, which in turn have their own processes, that can be executed partially or totally depending on the needs of the project.

In this layer it is possible, and even advisable, to pre-process the raw data coming from the sensors by applying different strategies such as conversion of formats, unification of values belonging from several sensors into a single variable, conversion of units of measurement, among other actions. It is also possible to filter data by establishing time intervals that must be met before sending data to the upper layer, applying multiple sampling, using low-pass type filters on the data flow, etc. Data cleaning could also be performed by removing noise, erroneous data or any other factor that could negatively affect the format of the data. All methods presented above are part of the Data pre-processing sub-layer.

The two main functions of the Aggregation layer are to perform aggregation and to provide an IoT gateway (the Aggregation and Link sub-layers, respectively), with the second one being mandatory within this layer. The Aggregation consists of combining data from heterogeneous devices. By its nature, IoT involves the collection of a large amount of data that originate from devices of different types, manufacturers, protocols, and with differentiated functionalities. All these data must be unified using tools devised for such purpose, with the objective of ultimately transmitting the integrated data to the upper layer. In fact, to carry out this transmission, an IoT gateway must be available within the Aggregation layer, acting as a client of the data transmission protocol used in the upper layer.

Implementation Example. Two servers deployed to perform the actions defined in the aggregation layer are tested in order to assess the modularity of the architecture. The first one is a proprietary software named iRidium Pro, which allows to integrate data from different protocols such as Modbus, KNX, BACnet, among others. This feature is named iRidium Server, which is the only feature we used in this work. The iRidiurm Pro framework also provides other tools: iRidium Studio, an environment to develop interfaces and i3 Pro, a client application for controlling the devices and server projects.

On the other hand, we developed a simple yet intuitive integrator using Python, version 3.7, with the libraries KNXIP[4] and PAHO[5], the first allows to perform the tunneling between the Python application and the KNX/IP interface, whilst the last one provides an interface to connect to a MQTT server, which is the resource used in the Exchange layer.

The aggregation using both, iRidium Server and the Python Client, was performed by connecting the KNX addresses from the devices to MQTT topics. Hence, for a light with the KNX address $0/0/1$, the topic related related to that address will be, for instance, $building1/room1/light1$. It is worth mentioning that all the data processing and formatting is performed by iRidium Server or the Python Client. Both software are able to give the KNX data the proper format. In order to send that formatted data to the Exchange layer the payload was simply the raw value of the data in this case.

C. Exchange

This layer includes data transfer protocols such as Hypertext Transfer Protocol (HTTP) or Advanced Message Queuing Protocol (AMQP) that allow to exchange data provided by the IoT gateways of the Aggregation layer and the final applications, located in the Application layer. Although the choice of protocols to be used ultimately depends on the needs of each application, the recommended protocols for IoT are those based on the subscription-publication pattern such as Message Queuing Telemetry Transport (MQTT) or Constrained Application Protocol (CoAP).

The Exchange layer is the core of an IoT project, since it acts as a middleware devised to connect the Aggregation layer with the Application layer. This resource is usually publicly visible for Internet of Things applications. Hence, the proper authentication methods should be addressed when configuring it. It is highly recommended to use Transport Layer Security (TLS) in this layer in order to have end-to-end encryption of the transferred data.

Implementation Example. For this layer we deploy a MQTT[6] server with authentication and TLS, in order to have end-to-end encryption of the data. MQTT is a lightweight publish/subscribe messaging transport protocol. It receives the data sent to a certain topic and makes that data available for all clients that are subscribed to that topic.

D. Application

It includes the final applications that allow users to access the data coming from the IoT devices and in turn sends commands to manage them if necessary. Applications can be mobile, web, console, among other types. How the received data is used, the business logic to be applied on that data and which part of the data will be considered more relevant, are factors that are exclusively dependent on the final objective of the application. Therefore, no limitations or patterns of applications are established in this section. However, the emphasis is placed on applying a good user experience design [12], in order to produce intuitive applications with high usability.

In addition to traditional applications, it is also possible to use devices to use them as interfaces i order to access data. Devices such as smart watches or smart glasses could be useful as interfaces within the Application layer. Although at the same time they could also be devices from the Physical sending data to the upper layers.

The final goal of the Internet of Things is the analysis of the data obtained from the various devices used for decision making. Therefore, an Intelligence sub-layer is proposed in order to generate visualizations, predictions, classification and grouping of data.

Implementation Example. A mobile application is developed using the Outsystems platform[7]. The application is able to receive data from the MQTT server by acting as a MQTT client and subscribing for specific topics, and also act on the devices that provide such capability. Therefore, to control the light with the KNX address $0/0/1$, the application sends the proper value to the topic $building1/room1/light1$, which is the same configuration used in the Aggregation layer by the integrator (iRidium Server or the Python Client). Figure 3 presents screenshots of the application.

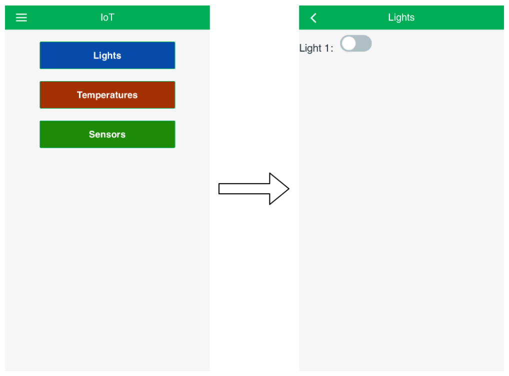

Fig. 3: Screenshots of the mobile application.

Note that this application could also be a web app or a console app. The only thing that must be considered is to properly provide the connection with the Exchange layer protocol, which is MQTT for this example. In fact, a Python MQTT Client could be configured to run in console in order to perform the same actions.

Although we did not use storage options in this layer, or in any other layer of the architecture, for this project, it is possible to do so by connecting the application to a data server in order to process and analyze that data afterwards. The application could also have authentication features to increase security.

[4]https://pypi.org/project/knxip/
[5]https://pypi.org/project/paho-mqtt/
[6]https://mqtt.org/
[7]https://www.outsystems.com/

E. Storage

It is possible to store the state of the data within the IoT flow, using different approaches depending on the needs of the project. Relational databases (Oracle, MySQL, etc.), non-relational databases (MongoDB, DynamoDB, etc.) and even persistence methods focused on Big Data (Hadoop) could be used. The storage method to be used would ultimately depend on the type and amount of data that will be processed. For applications that involve a large amount of data that have to be processed in real time, it is advised to store and process the data through big data solutions, such as the Hadooop framework[8]. On the other hand, applications that do not involve the processing of a high quantity of data could use traditional relational or non-relational databases.

The format and content of the data will depend on the layer in which the persistence operation is performed. For instance, the Application layer could incorporate a database to store the received data and use it within the applications to perform data analysis, presentation of visualizations and prediction of trends. Although using a database at the application level may seem redundant because in the Aggregation layer could already incorporate one, the Application layer could store the data in its final version, post-processed by the logic of the application and even store data referring to conclusions obtained from the data received from the Exchange layer. For example, a mobile application can receive the image of a security camera and a detection and recognition algorithm can be applied to people who are in the scene and store data such as time, date and their identities.

It is worth mentioning that only storing the data is not recommended. The data should be treated, analyzed and used to rise conclusions and considerations that could add value to the IoT project. Predictive analysis could be performed using the acquired data to create datasets useful to train intelligent systems able to perform such tasks, which is part of the Intelligence sub-layer described in Section II-D.

F. Security

Due to the nature of IoT, which rests on the interconnection of devices, security is an aspect of utmost importance when developing an application. It is important to have security schemes in each of the layers, because each one is susceptible to different types of attacks. This is even more relevant in case the architecture follows an Internet of Things scheme, that permits the devices of the physical layer to be connected to the Internet. For an Intranet of things scenario, the risks are lower, although they are also present. Hence, the proper security measures must be taken.

In an IoT scenario, there are two types of attacks: passive and active. The active attack directly blocks the operation of the devices while the passive one monitors the information of the IoT network without its presence being detected [13]. The proper measures to avoid these attacks must be taken.

The Physical layer should be based on choosing or developing devices that meet security standards. The capability to support the appropriate methods of authenticating users of the final applications should also be mandatory. Encryption, authentication and access control must be used in this layer. Regarding the security of the physical equipment used in this layer, it is recommended to use or develop devices that meet the standards defined in the good practices guide introduced by the Institute of Electrical and Electronic Engineering (IEEE) for IoT devices[9].

The Aggregation and Exchange layers, on the other hand, must use consistent protocols and software capable of identifying situations and behaviors that may represent potential threats. As mentioned before, since the Exchange layer is usually a public resource, the proper measures must be taken. Applying end-to-end data encryption in this layer is highly recommended.

In the Application layer, the emphasis should be focused on the availability of applications, taking into account the large amount of data coming from a high number of connected devices. In addition to that, there should be global policies and standards defined for authentication mechanisms and security strategies throughout all the client-side applications.

We also advise to follow the considerations presented by previously research work [14], which introduces several security measures with a layered-based security architecture, in which the measures for their Perception layer act over the Physical and Aggregation layers of our model, whilst those from the Transportation layer act over as our Exchange layer. Finally, the considerations regarding the Application layer should be applied over the final layer of our architecture, with the same name.

III. Considerations

A. Recommendations

The following considerations are recommended when establishing an IoT end-to-end infrastructure[10].

1) Data uniformity: there must be data format standards, mainly between the Application layer and the Exchange layer. In general, there is a pattern based on payloads (payloads) that contain the data to be transmitted in each exchange of messages.

2) Constant update: software and firmware of the devices must be kept constantly updated in order to provide the proper high-level security to the IoT infrastructure.

3) Resource centralization: it is relevant to provide decentralized servers, if necessary, so as not to suffer from bottlenecks in case that there is a large data flow.

4) Internet connectivity: internet connectivity must be used only when it is essential, in order to improve the security of the architecture. If it is used, it is recommended to apply the proper authentication mechanisms and end-to-end data encryption.

[8]http://hadoop.apache.org/

[9]https://internetinitiative.ieee.org/images/files/resources/white_papers/internet_of_things_feb2017.pdf

[10]https://www.ccn-cert.cni.es/informes/certifications-ccn-cert-buenas-practicas-bp.html

5) Data analysis: the behavior of the IoT devices is predictable, using Machine Learning [15] algorithms can be a useful option to identify and even predict abnormal situations or behaviors in order to respond to problems or threats, or in order to obtain conclusions that could aid at making decisions related to the project.

B. NTA-IoT Roles

The main idea of the development of a layer-based model is to decouple the components of each layer so that they could be independent from each other. This allows to achieve a modular structure, which is inherently. Hence, any modification on any of the layers will have no influence on the overall infrastructure or on the technologies used in the other layers. Likewise, expanding the features or the number of components of a layer will result in a minimal work of adaptation in the upper or lower layers, and in some cases even without any modification. Taking advantage of this structure, a division of roles is proposed when developing an IoT project:

1) Devices administrator: This role is based on the interaction with devices at a low level. The responsible must configure each device, groups of devices and their basic rules, if necessary. For not smart devices, this role must work on their connection to a proper IP gateway that will communicate with the service of the Aggregation layer through some IP protocol, such as TCP or UDP.

2) Aggregation manager: the aggregation manager will work in the connection between the physical and the digital part of the IoT architecture. This role will be the user, or even developer of software and equipment designed to aggregate data coming from the devices of the Physical layer. The methods and algorithms (and how they are going to be configured) that the aggregation service will apply to perform data pre-processing, data formatting, data filtering, aggregation and provide the gateway service must be configured by this role.

3) Exchange layer manager: the servers and gateways used in the Exchange layer must initially be configured and subsequently maintained by one or more managers.

4) Application developer: this role is based on the development of mobile, web or other types of applications that will be connected to the devices. They must follow the data patterns previously defined for the architecture. This role will focus on the design and implementation of the applications, on the manner that the received data will be processed during the application run-time and on providing an optimal user experience based on the available data.

5) Data Analyst: the data analyst will use the data obtained in each layer to apply data analysis, prediction and classification techniques, in order to obtain relevant information that will serve as guide for decision making. This role could also be in charge of the implementation of intelligent models able to inject value into the final client-applications applications.

It worth mentioning that each role must also take into account other common aspects such as security and availability. For the security layer, the figure of a security manager may be created, in order to manage and coordinate the security strategies used by each of the roles.

6) Roles in the Implementation Example: Note that in the context of the implementation example presented in the previous section, the authors performed the activities of all roles except for the Data Analyst, considering that no data analytics techniques were applied to obtain insights from the data gathered through the use of the application. Notwithstanding, this will be addressed in future work.

C. Characteristics and Limitations

As a layered-architecture, the NTA-IoT provides the separation of concern [16] among the components of each layer, which implies that the functioning of the components of each layer is independent from the others. This is how modularity is achieved. However, a layer depends on the output of the previous layer to be able to operate. Hence, issues within a layer that do not allow the data to flow into the next layer will have a repercussion on the following layers. On the other hand, the architecture is centered on the data flow between its origin (devices) and the final application, without considering the particularities of existing data in data warehouses, data lakes, etc., that will be used at the application level. The NTA-IoT architecture contemplates this aspect only superficially (with the Storage layer) although a proper framework with its definition must be adopted for that purpose. Finally, it is worth mentioning that the architecture is devised to support use cases from different domains, allowing to change or add new elements without significantly changing the architecture. This is how flexibility is achieved [17].

IV. Conclusion

This work introduced a flexible and modular framework for developing IoT applications. The architecture is based on four base layers (Physical, Aggregation, Exchange and Application) and two support layers that act on the overall scheme as auxiliary and non-mandatory resources. This simple and abstract architecture is named NTA-IoT. We also introduced suggested roles to be applied to lead IoT projects based on the proposed architecture. Finally, we presented a practical example of the implementation of the framework for a home automation problem.

It is expected that the flexibility and modularity of the proposed architecture will lead to achieve satisfactory results when leading IoT projects. On the other hand, the proposed architecture is focused only on data streams between the devices and the final application, any other data source that already exist in the Application layer should be properly treated following a more specific framework. Future work may aim at exploring other architectures derived from the NTA-IoT and to apply the framework to case studies in areas such as healthcare, industrial automation, commerce, among others.

References

[1] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, "Future internet: The internet of things architecture, possible applications and key challenges," in *2012 10th International Conference on Frontiers of Information Technology*, Dec 2012, pp. 257–260.

[2] S. K. Datta, C. Bonnet, and N. Nikaein, "An iot gateway centric architecture to provide novel m2m services," in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, March 2014, pp. 514–519.

[3] I. Modbus, "Modbus application protocol specification v1. 1a," *North Grafton, Massachusetts (www. modbus. org/specs. php)*, 2004.

[4] I. Ungurean, N. Gaitan, and V. G. Gaitan, "An iot architecture for things from industrial environment," in *2014 10th International Conference on Communications (COMM)*, May 2014, pp. 1–4.

[5] L. Catarinucci, D. de Donno, L. Mainetti, L. Palano, L. Patrono, M. L. Stefanizzi, and L. Tarricone, "An iot-aware architecture for smart healthcare systems," *IEEE Internet of Things Journal*, vol. 2, no. 6, pp. 515–526, Dec 2015.

[6] I. Gronbak, "Architecture for the internet of things (iot): Api and interconnect," in *2008 Second International Conference on Sensor Technologies and Applications (sensorcomm 2008)*, Aug 2008, pp. 802–807.

[7] G. Dai and Y. Wang, "Design on architecture of internet of things," in *Advances in Computer Science and Information Engineering*. Springer, 2012, pp. 1–7.

[8] C. Sarkar, A. U. N. S. N., R. V. Prasad, A. Rahim, R. Neisse, and G. Baldini, "Diat: A scalable distributed architecture for iot," *IEEE Internet of Things Journal*, vol. 2, no. 3, pp. 230–239, June 2015.

[9] J. Ren, H. Guo, C. Xu, and Y. Zhang, "Serving at the edge: A scalable iot architecture based on transparent computing," *IEEE Network*, vol. 31, no. 5, pp. 96–105, 2017.

[10] R. Teymourzadeh, S. A. Ahmed, K. W. Chan, and M. V. Hoong, "Smart gsm based home automation system," *arXiv preprint arXiv:1806.03715*, 2018.

[11] M. Romero, W. Guédria, H. Panetto, and B. Barafort, "Towards a characterisation of smart systems: A systematic literature review," *Computers in Industry*, 2020.

[12] C. Moser, "User experience design," in *User experience design*. Springer, 2013, pp. 1–22.

[13] R. Mahmoud, T. Yousuf, F. Aloul, and I. Zualkernan, "Internet of things (iot) security: Current status, challenges and prospective measures," in *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*, Dec 2015, pp. 336–341.

[14] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the internet of things: perspectives and challenges," *Wireless Networks*, vol. 20, no. 8, pp. 2481–2501, Nov 2014. [Online]. Available: https://doi.org/10.1007/s11276-014-0761-7

[15] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[16] C. SantAnna, E. Figueiredo, A. Garcia, and C. Lucena, "On the modularity assessment of software architectures: Do my architectural concerns count," in *Proc. International Workshop on Aspects in Architecture Descriptions (AARCH. 07), AOSD*, vol. 7, 2007.

[17] A. Aerts, N. B. Szirbik, and J. B. Goossenaerts, "A flexible, agent-based ict architecture for virtual enterprises," *Computers in Industry*, vol. 49, no. 3, pp. 311–327, 2002.