Journal on Recent Technologies in Mechanical and Electrical Engineering
Volume: 5 Issue: 6

ISSN: 2349-7947
38 - 41

IOT Enabled Energy Efficient Wireless Sensor Network & Services – A Pathway towards Green Engineering - Part 1 (IoT Architecture)

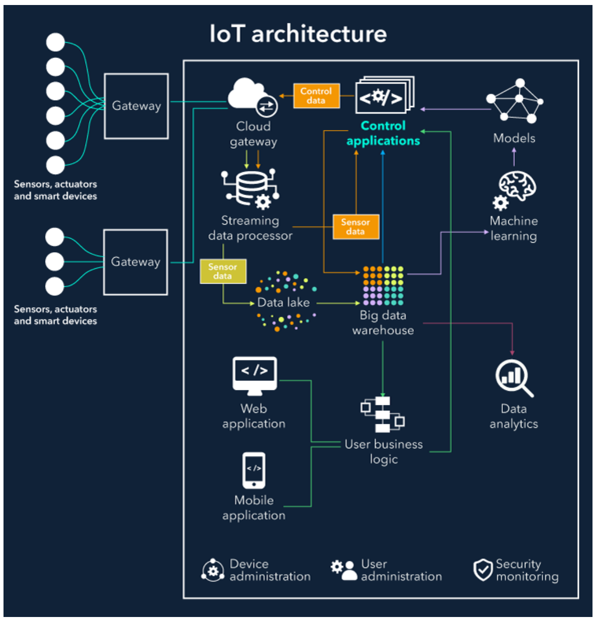
Prof K K Vyas, Director SIET Sikar
Amita Pareek, Scholar, MTech, SIET Sikar

Wireless Sensor Network (WSN) and low power smart device network (Internet of Things) are fast growing technologies used in wide range applications. In IoT nodes [1] are defined as sensor/actuators, which are highly constrained in nature (limited in energy and CPU usage). Designing energy efficient network architecture and routing mechanism is a great challenge in IoT networks. The network is said as energy efficient, based on node survivability. In the series of these exploratory papers part 1 to part 5 different aspects like Basic Architecture, WSN , Routing Algorithms for WSN and Applications of WSN etc will be reveled. Some concept & figure used from references with appropriate reference no. marked. Here energy efficient network architecture is proposed to prolong the network lifetime. Sensor nodes and relay nodes are placed in hierarchical manner to avoid uneven energy drainage (energy hole problem). In some architecture [1], sensors do sensing and relay nodes handle communication (data transmission from sensor to sink), which reduce the complexity of device. Relay node placement is done based on data traffic of the network. The aim of Energy efficient routing mechanism is to achieve balanced energy consumption.

*****

IoT architecture: building blocks and how they work
Can you imagine a huge variety of smart devices under the centralized control from one “brain”? To a certain extent, it’s possible with the evolvement of the Internet of Things - the network of physical objects with sensors and actuators, software and network connectivity that enable these objects to gather and transmit data and fulfill users’ tasks. The effectiveness and applicability of such a system directly correlate with the quality of its building blocks and the way they interact, and there are various approaches to IoT architecture.

Basic elements of IoT architecture
IoT architecture diagram[4] shows the building blocks of an IoT system and how they are connected to collect, store and process data.



Things. A “thing” is an object equipped with **sensors** that gather data which will be transferred over a network and **actuators** that allow things to act (for example, to switch on or off the light, to open or close a door, to increase or decrease engine rotation speed and more). This concept includes fridges, street lamps, buildings, vehicles, production machinery, rehabilitation equipment and everything else imaginable. Sensors are not in all cases physically attached to the things: sensors may need to monitor, for example, what happens in the closest environment to a thing.

Gateways. Data goes from things to the cloud and vice versa through the gateways. A gateway provides connectivity between things and the cloud part of the IoT solution, enables data preprocessing and filtering before moving it to the cloud (to reduce the volume of data for detailed processing and storing) and transmits control commands going from the cloud to things. Things then execute commands using their actuators.

Cloud gateway facilitates data compression and secure data transmission between field gateways and cloud IoT servers. It also ensures compatibility with various protocols and communicates with field gateways using different protocols depending on what protocol is supported by gateways.

Streaming data processor ensures effective transition of input data to a data lake and control applications. No data can be occasionally lost or corrupted.

Data lake. A data lake is used for storing the data generated by connected devices in its natural format. Big data comes in "batches" or in “streams”. When the data is needed for meaningful insights it’s extracted from a data lake and loaded to a big data warehouse.

Big data warehouse: Filtered and preprocessed data needed for meaningful insights is extracted from a data lake to a big data warehouse. A big data warehouse contains only cleaned, structured and matched data (compared to a data lake which contains all sorts of data generated by sensors). Also, data warehouse stores context information about things and sensors (for example, where sensors are installed) and the commands control applications send to things.

Data analytics: Data analysts can use data from the big data warehouse to find trends and gain actionable insights. When analyzed (and in many cases – visualized in schemes, diagrams, infographics) big data show, for example, the performance of devices, help identify inefficiencies and work out the ways to improve an IoT system (make it more reliable, more customer-oriented). Also, the correlations and patterns found manually can further contribute to creating algorithms for control applications.

Machine learning and the models ML generates: With machine learning, there is an opportunity to create more precise and more efficient models for control applications. Models are regularly updated (for example, once in a week or once in a month) based on the historical data accumulated in a big data warehouse. When the applicability and efficiency of new models are tested and approved by data analysts, new models are used by control applications.

Control applications send automatic commands and alerts to actuators, for example:

- Windows of a smart home can receive an automatic command to open or close depending on the forecasts taken from the weather service.
- When sensors show that the soil is dry, watering systems get an automatic command to water plants.
- Sensors help monitor the state of industrial equipment, and in case of a pre-failure situation, an IoT system generates and sends automatic notifications to field engineers.

The commands sent by control apps to actuators can be also additionally stored in a big data warehouse. This may help investigate problematic cases (for example, a control app sends commands, but they are not performed by actuators – then connectivity, gateways and actuators need to be checked). On the other side, storing commands from control apps may contribute to security, as an IoT system can identify that some commands are too strange or come in too big amounts which may evidence security breaches (as well as other problems which need investigation and corrective measures).

Control applications can be either rule-based or machine-learning based. In the first case, control apps work according to the rules stated by specialists. In the second case, control apps are using models which are regularly updated (once in a week, once in a month depending on the specifics of an IoT system) with the historical data stored in a big data warehouse.

Although control apps ensure better automation of an IoT system, there should be always an option for users to influence the behavior of such applications (for example, in cases of emergency or when it turns out that an IoT system is badly tuned to perform certain actions).

User applications are a software component of an IoT system which enables the connection of users to an IoT system and gives the options to monitor and control their smart things (while they are connected to a network of similar things, for example, homes or cars and controlled by a central system). With a mobile or web app, users can monitor the state of their things, send commands to control applications, set the options of automatic behavior (automatic notifications and actions when certain data comes from sensors).

Device management

To ensure sufficient functioning of IoT devices, it's far not enough to install them and let things go their way. There are some procedures required to manage the performance of connected devices (facilitate the interaction between devices, ensure secure data transmission and more):

- **Device identification** to establish the identity of the device to be sure that it's a genuine device with trusted software transmitting reliable data.
- **Configuration and control** to tune devices according to the purposes of an IoT system. Some parameters need to be written once a device is installed (for example, unique device ID). Other settings might need updates (for example, the time between sending messages with data).
- **Monitoring and diagnostics** to ensure smooth and secure performance of every device in a network and reduce the risk of breakdowns.
- **Software updates and maintenance** to add functionality, fix bugs, address security vulnerabilities.

User management

Alongside with device management, it's important to provide control over the users having access to an IoT system.

User management involves identifying users, their roles, access levels and ownership in a system. It includes such options as adding and removing users, managing user settings, controlling access of various users to certain information, as well as the permission to perform certain operations within a system, controlling and recording user activities and more.

Security monitoring

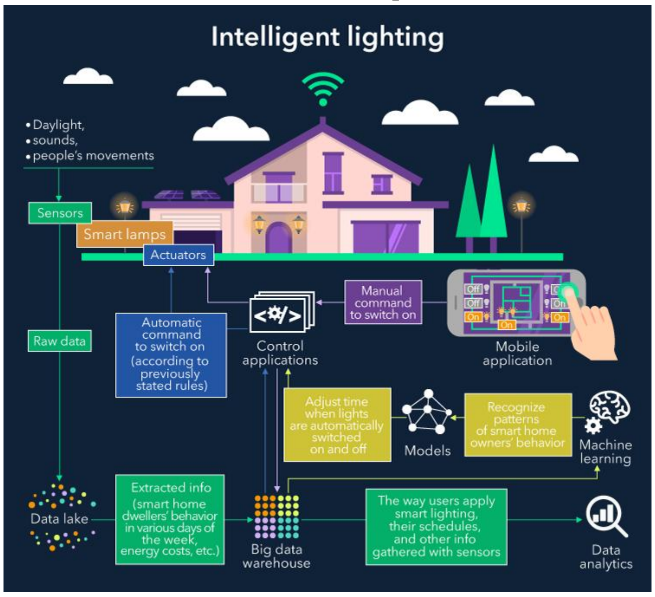
Security is one of the top concerns in the internet of things. Connected things produce huge volumes of data, which need to be securely transmitted and protected from cyber-criminals. Another side is that the things connected to the Internet can be entry points for villains. What is more, cyber-criminals can get the access to the "brain" of the whole IoT system and take control of it.

To prevent such problems, it makes sense to log and analyze the commands sent by control applications to things, monitor the actions of users and store all these data in the cloud. With such an approach, it's possible to address security breaches at the earliest stages and take measures to reduce their influence on an IoT system (for example, block certain commands coming from control applications).

Also, it's possible to identify the patterns of suspicious behavior, store these samples and compare them with the logs generated by an IoT systems to prevent potential penetrations and minimize their impact on an IoT system.

IoT architecture example – Intelligent lighting

Let's see how our IoT architecture elements work together by the example of smart yard lighting as a part of a smart home – a bright illustration of how an IoT solution simultaneously contributes to user convenience and energy efficiency. There are various ways a smart lighting system can function, and we'll cover basic options.


Intelligent lighting
• Daylight,
• sounds,
• people's movements
Sensors
Smart lamps
Actuators
Manual command to switch on
Off
Off
On
On
On
On
Mobile application
Control applications
Raw data
Automatic command to switch on (according to previously stated rules)
Adjust time when lights are automatically switched on and off
Models
Recognize patterns of smart home owners' behavior
Machine learning
Data lake
Extracted info (smart home dwellers' behavior in various days of the week, energy costs, etc.)
Big data warehouse
The way users apply smart lighting, their schedules, and other info gathered with sensors
Data analytics

Basic components [4]

Sensors take data from the environment (for example, daylight, sounds, people's movements). Lamps are equipped with the actuators to switch the light on and off. A ***data lake*** stores raw data coming from sensors. A ***big data warehouse*** contains the extracted info smart home dwellers' behavior in various days of the week, energy costs and more.

Manual monitoring and manual control

Users control smart lighting system with a ***mobile app*** featuring the map of the yard. With the app users can see which lights are on and off and send commands to the control applications that further transmit them to lamp actuators. Such an app can also show which lamps are about to be out of order.

Data analytics

Analyzing the way users apply smart lighting, their schedules (either provided by users or identified by the smart system) and other info gathered with sensors, data analysts can make and update the algorithms for control applications.

Data analytics also helps in assessing the effectiveness of the IoT system and revealing problems in the way the system works. For example, if a user switches off the light right after a system automatically switches it on and vice versa, there might be gaps in the algorithms, and it's necessary to address them as soon as possible.

Automatic control's options and pitfalls

The sensors monitoring natural light send the data about the light to the cloud. When the daylight is not enough (according to previously stated threshold), the control apps send automatic commands to the actuators to switch on the lamps. The rest of the time the lamps are switched off.

However, a lighting system can be "baffled" by the street illumination, lamps from neighboring yards and any other sources. Extraneous light captured by sensors can make the smart system conclude that it's enough light, and lighting should be switched off. Thus, it makes sense to give the smart system a better understanding of the factors that influence lighting and accumulate these data in the cloud.

When sensors monitor motions and sounds, it's not enough just to switch on the light when movements or sounds are identified in the yard or switch all the lamps off in the silence. Movements and sounds can be produced, for example, by pets, and cloud applications should distinguish between human voices and movements and those of pets. The same is about the noises coming from the street and neighboring houses and other sounds. To address this issue, it's possible to store the examples of various sounds in the cloud and compare them with the sounds coming from the sensors.

Machine learning

Intelligent lighting can apply models generated by machine learning, for example, to recognize the patterns of smart home owners' behavior (leaving home at 8 am, coming back at 7 pm) and accordingly adjust the time when lights are switched on and off (for example, switch the lamps on 5 minutes before they will be needed).

Analyzing users' behavior in long-time perspective, a smart system can develop advanced behavior. For example, when sensors don't identify typical movements and voices of home inhabitants, a smart system can "suppose" that smart home dwellers are on a holiday and adjust the behavior: for example, occasionally switch on the lights to give the impression that the house is not empty (for security reasons), but do not keep the lights on all the time to reduce energy consumption.

User management options

To ensure efficient **user management**, the smart lighting system can be designed for several users with role distribution: for example, owner, inhabitants, guests. In this case, the user with the title "*owner*" will have full control over the system (including changing the patterns of smart light behavior and monitoring the status of the yard lamps) and priorities in giving commands (when several users give contradicting commands, an owner's command will be the one control apps execute), while other users will have access to a limited number of the system's functions. *"Inhabitants"* will be enabled to switch on and off the lamps with no opportunity to change settings. *"Guests"* will be able to switch on and off the light in some parts of the house and have no access to controlling the lights, for example, near the garage.

Apart from role distribution, it's essential to consider ownership (as soon as one system can control over 100 thousand of households, and it's important that a dweller of a smart home manages the lighting in his yard, and not the one of a neighbor).

In a nutshell, IoT architecture contains the following components:

- **Things** equipped with **sensors** to gather data and **actuators** to perform commands received from the cloud.
- **Gateways** for data filtering, preprocessing and moving it to the cloud and vice versa, – receiving commands from the cloud.
- **Cloud gateways** to ensure data transition between field gateways and central IoT servers.
- **Streaming data processors** to distribute the data coming from sensors among relevant IoT solution's components.
- **Data lake** for storing all the data of defined and undefined value.
- **Big data warehouse** for collecting valuable data.
- **Control applications** to send commands to actuators.
- **Machine learning** to generate the models which are then used by control applications.
- **User applications** to enable users to monitor control their connected things.
- **Data analytics** for manual data processing.

IoT architecture also contains device and user management components to provide stable and secure functioning of things and control user access issues.

References :

[1]. Energy efficient network architecture for IoT applications, Sarwesh P ; N.Shekar V. Shet ; Chandrasekaran K
[2]. Energy efficiency of IOT, Technology & Energy Efficiency report, April 2016.
[3]. https://www.elprocus.com/architecture-of-wireless-sensor-network-and-applications/
[4]. https://www.scnsoft.com/blog/iot-architecture-in-a-nutshell-and-how-it-works
[5]. Energy Efficiency of IOT, EDNA,
[6]. Design & Analysis of Energy Efficient Routing protocol – R D Deokar
[7]. Energy Efficiency in WSN N K Pour, Dec 2015,
[8]. https://arxiv.org/ftp/arxiv/papers/1605/1605.02393.pdf
[9]. https://www.analyticsvidhya.com/blog/2016/08/10-youtube-videos-explaining-the-real-world-applications-of-internet-of-things-iot/