IoT Messaging Using Low-Cost COTS Devices

Francis Rousseau
IoT & Industry 4.0 department
Uzinakod
Montréal, Canada
francis.rousseau.1@ens.etsmtl.ca

Tony Wong
Systems Engineering Department
École de technologie supérieure
Montréal, Canada
tony.wong@etsmtl.ca

Abstract

Real time information and data's growing significance are changing the way we gather and interact with information. A mob of hardware and software innovations are pushing connectivity beyond today's known limits. This paper will give an overview of how Industry 4.0 typical processes from the edge to the cloud collects, aggregates and stores real-time information using low-cost commercial off-the-shelf (COTS) devices. A description of each layer and step guaranteeing the data flow will be analyzed to understand how each connected device create the Internet of things (IoT) environment. The presented platform used for this analysis will demonstrate each process and communication protocol involved like the Message Queuing Telemetry Transport (MQTT) and the Serial Peripheral Interface (SPI).

Keywords
Gateway, MQTT, SPI, IoT, Messages

1. Introduction

In a nutshell, the concept of Internet of Things refers to a network of interacting devices connected to the Internet (Atzori et al. 2017). The abstraction of devices as "objects" is possible due to the development of powerful embedded computers and microcontrollers with optimized energy consumption and small physical footprint. In a typical IoT architecture, the message exchange process is modeled by a connection layer which is often described as the input layer (Fatima et al. 2020). Messages arriving to the connection layer are then transformed into data and dispatched to other processing layers within the architecture (Lee et al. 2015 ; Ganguli et al. 2017).

It is common for businesses to track and analyze their operation processes. The introduction of IoT enables tracking, analysis and even forecasting in real time. This extends the Internet connectivity to a more detailed level than the one enjoyed by traditional computers and computer systems. According to (Karie et al. 2020), more than 38 billion objects will be connected to the Internet as soon as 2025. This new connectivity extension is not only confined to the IoT domain. In fact, this is already visible in areas such as information technologies, digital enterprises, data analytics and artificial intelligence. Furthermore, revenues generated by IoT platforms, edge computing and tier software worldwide are estimated to be 136 to 235 billion dollars in the next few years (Liu 2017).

Process management and monitoring is common practice in modern organizations. Technological evolution entails organizational adaptation in order to be efficient and competitive. Thus, the evolution of Business Process Management systems can be correlated with several industrial revolution phases that have occurred since the beginning of the 20th century. With information sharing, through IoT, industries are able to adapt and become responsive to the collected data (Lee et al. 2015). In this way, manufacturers can offer individualized products while maintaining manufacturing infrastructure capable of mass production (Lasi et al. 2014). There are at least four steps involved in most IoT messaging. These steps are: i) Acquisition; ii) Aggregation; iii) Transmission; v) Saving; vi) analyzing. Using sensing devices and controllers to collect raw data is the first step. The data are then grouped to form

a meaningful data packet. A computational device, acting as a gateway, encodes the data packet for transmission to cloud services. Lastly, data are stored and analyzed using cloud services.

In this paper, we present a design approach using low-cost commercial off-the-shelf devices for IoT messaging. In the next section, we examine the IoT platform and its setup. Section 3, focuses on the communication issues and data management. Finally, section 4, we discuss important IoT messaging steps.

2. IoT Platform

The structure of the IoT Platform used for this paper is shown in Figure 1. Data acquisition is accomplished via the controller by sending commands and reading values from the sensor modules. At this point, data exchange is often at a very low-level requiring hardware input-output port. The controller then processes the data into a usable form and forward them to the gateway. This processing is needed to smooth out sampled data for upstream signal filtering, intervals and statistical analysis. Finally, the gateway is responsible for the network interconnect and data exchange between devices by means of predefined high-level protocols.

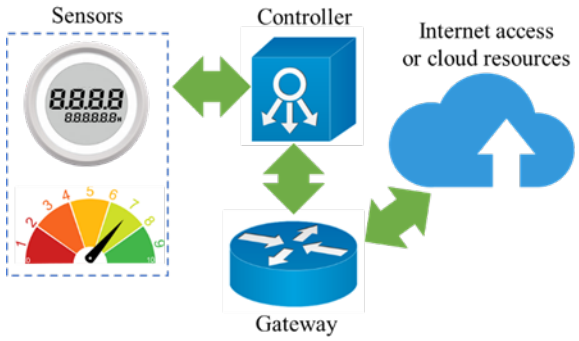

Figure 1. Structure and modules of the IoT platform

In this work, we deploy two sensor modules. The first one will be used to acquire sound signals and the second sensor will sample temperature and humidity. Each sensor is in turn controlled by an Arduino Uno R3 board (AUR3) (Arduino 2019). Even though recent AUR3 boards are network capable, it is not advisable to expose them directly to the Internet because of security concerns. The network interconnect is thus delegated to a single-board computer (SBC) namely the Raspberry Pi 4 (Raspberry 2021). The latter will manage high-level communications through the Internet to other cloud services.

2.1 Controllers and Gateways

Controllers often have the main task of assuring the control of data acquisition. Two AUR3 boards are tasked with sampling data from the sensors. They are connected and configured to operate through the Serial Peripheral Interface (SPI) bus (Grusin 2013).

The Raspberry Pi 4 (Pi4) is used to coordinate events within the IoT platform. This SBC has a low energy consumption level and a small footprint. It controls the communication link between the controllers and the cloud services. Unlike standard network nodes, a gateway can communicate using more than one protocol. The use of an SBC as a gateway enables its deployment close to the field devices.

A network bridge is created to allow Internet access. With this network infrastructure in place, the Pi4 can now transmit packets in addition to being accessible by the SSH protocol using a remote terminal.

2.2 Sensors

We used two sensors for the purpose of acquiring different types of data. The DHT11 digital humidity and temperature sensor is based on a capacitive humidity component with direct digital readouts. The other sensor is an Electret microphone with an adjustable gain and a 20 Hz to 20 kHz passband range. Analog signals can be easily converted to digital form using AUR3's onboard analog to digital converter (ADC). This sound signal can be transformed into equivalent sound exposure level L_{eq} as described by Eq. (1)

$$L_{eq} = 10log_{10}\left(\frac{1}{t_p}\sum_{i=1}^{n} t_i 10^{0.1L_i}\right) \#(1)$$

where t_p represents the total measurement duration, n is the number of acoustic pressure L_i with sampling rate t_i. The result represents a sound energy level measure in dB.

To compute the sound energy value from input signal x, we need to perform the following conversions presented in Fig. 2. First, the sound pressure level or SPL is amplified by the sensor gain. It is then converted into an electrical signal in dBV/Pa by the sensor. The electrical signal is sampled by the onboard 10-bit ADC into the numerical values. This is the value read by the controller.

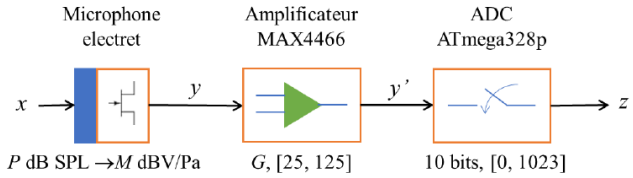

Figure 2. Sound energy level conversion and amplification process

By using the common acoustic pressure reference P_{SPL} = 20 µPa, it is possible to define that for P = 1 Pa rms the sensor will have a sensitivity around 94 dB SPL. This result in dBV/Pa is a direct application of the Eq. (2). That is to say,

$$dB_{SPL} = 10log_{10}\left(\frac{P}{P_{SPL}}\right)^2 .\#(2)$$

For this work, the gain will be set at G = 125 which does represent a value around 42 dBV. So, with this gain G, the sensitivity M in dBV/Pa and the pressure P in dB SPL, the acoustic pressure L_i can be calculated from the logarithmic scaling of dB:

$$L_i = dBV + P - M - G. \#(3)$$

To obtain the equivalent sound level. it suffices to accumulate n values of L_i such that

$$L_{eq} = L_1 + \cdots + L_n. \#(4)$$

The sampling of temperature, humidity and the sound pressure transformation occur at fixed intervals. The resulting values can be averaged over several sampling periods before transmitting to cloud services.

3. Communication Network and data management

Our communication network is divided in two main parts. We have implemented the Serial Peripheral Interface (SPI) communication protocol to establish a wired synchronous communication between the controllers and the Gateway. This separation of the data wires and the clock wire makes it possible to have an infrastructure in which the AUR3s are automatically synchronized with the clock frequency defined and managed by the gateway.

A structure had to be put in place to define a communication standard. A list of commands was therefore defined to allow interactions between the controller and the gateway. For requests in which the Pi4 ask a return value, it needs to perform two transmissions in order to save the value received from the controller. The first transmission is used for sending the position of the desired value in the data register on the AUR3. Once the desired position in the register is received by the controller, the AUR3 sends back the requested value in the SPI Data Register (SPDR). Then, the Pi4 transmits an acknowledge byte enabling the controller to place the next value on the Master-In Slave-Out (MISO) line.

3.1 MQTT Transfer Protocol

At the gateway, the process and conditions of sending received data from both controllers are very simple. Since they sample data at a defined frequency, the Pi4 also sends data to the cloud based on a frequency defined by the user. Whether these receiving and sending frequencies are different or the same, the important thing is that each value from the controllers is available in the Pi4's internal memory when the send command is executed for all the collected data. This task is performed at a fixed frequency as long as the user doesn't stop the sending of the data. To send a message to our cloud platform, the Pi4 acts as an MQTT publisher (HiveMQ 2015). We use the MQTT server of the cloud platform provided by MathWorks (MathWorks 2019). Once the connection information has been received from the Web platform, the transmissions are carried out to the desired frequency and according to the receiving limitations of the platform.

3.2 Data Management

Sampled data need to be stored locally on the controller to permit reading by other platform components. At the controller's level, we have established an environment having a software class containing a public section in which functions are accessible and a private section in which data access is controlled. These public functions are the ones allowing access and modifications to the variables related to the temperature and humidity in the private section. Variables are stored in a data structure before any communications to the gateway. This procedure is the one used when we want to make the controller's variables available for the gateway. Using this communication scheme, it is essential that both Pi4 and AUR3 are programmed with the same data structure arrangement.

For the cloud platform, data management is managed by the concept of channels and fields. Temperature and humidity can be stored in the same channel but in a different field. It is also possible to store these data in different channels. Since data exchange using WEB services is slow, it is advised to tag each data point with a timestamp. Many cloud platforms also have the timestamping capability.

4. Application steps

In this section, we will go over the extent of the deployed platform and its programs. A review of the features and functionalities implemented at the gateway and controllers' level will be done to demonstrate how the platform orchestrates data ending from the sensors to the cloud. This overview of the established ecosystem will allow summarizing how software and hardware interact as a whole.

4.1 Startup and Data Acquisition

Since the Pi4 is the platform gateway, it is important to show the platform boot-up steps. For the controller, its startup process is rather simple. Once powered on, the controller will execute its initialization routine and enters a busy loop waiting for some events to occur. Note that the Pi4 startup is also made by powering the board itself.

At the beginning of the startup, the Pi4 transmits a stop command to disable data acquisition. This command ensures that the controllers are all in the same waiting state. Subsequently, the gateway transmits a set of SPI commands to set up acquisition variables and sampling rates as specified by the user. A start command is then transmitted to the controllers allowing them to start the sampling process. After these steps, both controllers will loop through their programs and execute the desired data acquisition operations. For example, the sampling rate for temperature and humidity (~ 30 seconds) can be much higher than the equivalent sound exposure level (~ 180 seconds). Once the temperature, humidity, and L_{eq} values were gathered, reading is done by the Pi4 before aggregating and sending all the values to the cloud.

4.2 Reading, Aggregating and Sending

Each value coming from the controllers is stored in the form of a byte array in the gateway's memory. We only need to define a read command and define the array size in order to receive and manage incoming data.

To read sensor data, the Pi4 gateway can sample the controllers using the SPI port. Data are aggregated after each read operation. It is an essential task to perform before sending data over the Internet. We accomplish this data aggregation using struct.unpack() command. This aggregation procedure also converts data bit patterns into programming language data. Ultimately, data values are further converted into character strings and transmitted to cloud services using the MQTT protocol.

4.3 Dashboards

Raw data transmitted to cloud services are usually displayed on dashboards. Figures 3, 4 and 5 show a typical setup showing temperature, humidity and equivalent sound exposure level.

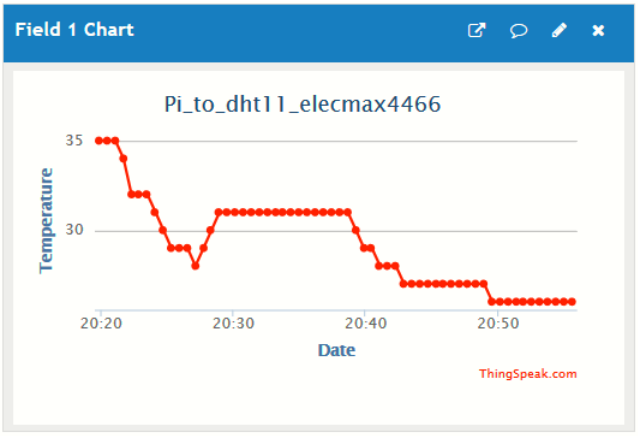

Figure 3. Dashboard showing Temperature values

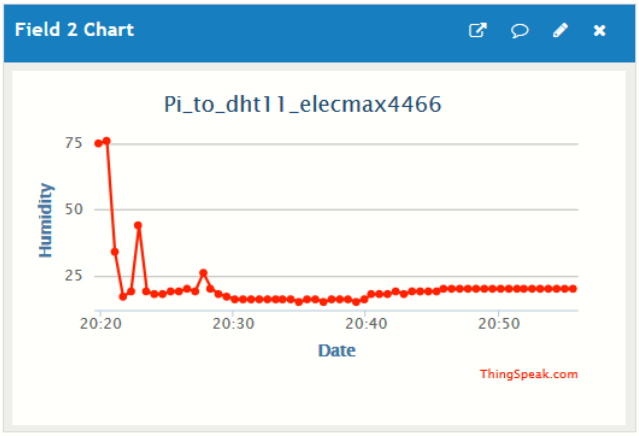

Figure 4. Humidity Level.

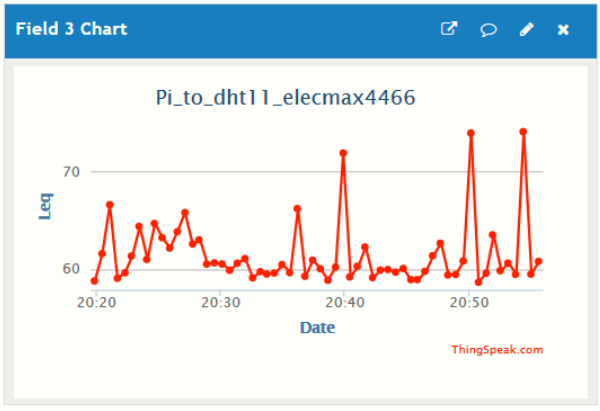

Figure 5. Sound exposure level.

As mentioned earlier, the task of the cloud platform, in this interconnected environment, is to gather and store raw data coming from the controllers. Changes in data values are easily trackable by the use of a dashboard. It is also possible to get a timeline of when they were saved. With this additional information, we get a key-value pair allowing us to time track each saved value.

5. Conclusion

The IoT platform created in this paper using COTS low-cost devices brought raw data generated by analog and digital sensors from the edge to the cloud seamlessly. Establishing a connected infrastructure of this kind is complex but by defining the intended role and technical limitations of individual component such as the gateway or the controllers helped with the implementation of the functionalities at their respective layer. The ability to accurately assess these hardware and software limitations is the key factor to have a highly detailed view of the complex ecosystem. Visualizing each layer where value is added to the raw data and becoming a meaningful data packet greatly helps estimating the final informative potential of the data packet. Furthermore, the communication links used to support the data flows and the cloud connectivity is what defines the collaborative capacities of the entire platform. We can therefore demonstrate that the implementation of a connected environment combining several IoT technologies is a contextual task that needs a serious in-depth pre-implementation analysis of the needs. With rapidly evolving connected technologies, an IoT platform shouldn't simply be deployed and forget without searching at new technologies which could influence today's processes and connectivity in the near future. Once collected data reach the cloud, it must transform its informative potential into contextual actions within or outside the connected ecosystem.

References

Atzori, L., Iera, A., and Morabito, G., Understanding the Internet of Things: definition, potentials, and societal role of a fast evolving paradigm, *Ad Hoc Networks*, vol. 56, pp. 122-140, 2017.

Fatima, I., Malik, S. U. R., Anjum, A., and Ahmad, N., Cyber Physical Systems and IoT: Architectural Practices, Interoperability, and Transformation, IT Professional, vol. 22, no. 3, pp. 46-54, 2020.

Lee, J., Bagheri, B., and Kao, H.-A., A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems, Manufacturing Letters, vol. 3, no. F0020004, pp. 18-23, 2015.

Ganguli S., and Friedman, T., IoT Technology Disruptions: A Gartner Trend Insight Report, Gartner, 2017.

Karie, N. M., Sahri, N. M., and Haskell-Dowland, P., IoT Threat Detection Advances, Challenges and Future Directions, Workshop on Emerging Technologies for Security in IoT (ETSecIoT), pp. 22-29, 2020.

Liu, V., Business Benefits of the Internet of Things: A Gartner Trend Insight Report, 2017.

Lasi, H., Fettke, P., Kemper, H. G., Feld, T., and Hoffmann, H., Industry 4.0, Business & Information Systems Engineering, vol. 6, no. 4, pp. 239-242, 2014.

Arduino., Arduino Uno Rev 3. Available: https://store.arduino.cc/usa/arduino-uno-rev3, 2019.

Raspberry Pi Foundation., Raspberry Pi 4. Available: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/, 2021.

Grusin. M., Serial Peripheral Interface (SPI). Available: https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all

The HiveMQ Team., Quality of Service 0,1 & 2 - MQTT Essentials: Part 6. Available: https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/, 2015.

MathWorks., MQTT Basics. Available: https://www.mathworks.com/help/thingspeak/mqtt-basics.html#bvzu07x, 2019.

Biographies

Francis Rousseau holds a B.Eng and M.Eng. degrees in systems engineering from École de technologie supérieure. With a background from the aerospace, automotive, software and manufacturing industries. M. Rousseau joined Uzinakod in 2021. His interests cover the spheres of automation, internet of things, networking, robotics, and data-driven manufacturing process optimization.

Tony Wong holds a B.Eng. and M.Eng. degrees from École de technologie supérieure in electrical engineering. He received his Ph.D. degree in computer engineering from École Polytechnique de Montréal. Dr. Wong joined the Systems Engineering department of École de technologie supérieure in 1999. His research interests are multi-objective chance-constrained evolutionary algorithms, machine learning methods, and their application to service and manufacturing problems.