IoT sensor integration to Node-RED platform

Milica Lekić[1], Gordana Gardašević[2]
Department of Telecommunications
Faculty of Electrical Engineering, University of Banja Luka
Banja Luka, Bosnia and Herzegovina
milica.lekic@etf.unibl.org [1], gordana.gardasevic@etf.unibl.org [2]

Abstract – This paper presents the implementation of an Internet of Things (IoT) application that performs the temperature and humidity sensing using DHT11 sensor on Raspberry Pi, and data transfer to the Cloud of the IBM Bluemix. The implementation is done using programming system Raspbian Stretch Lite on Raspberry Pi and IBM Internet of Things platform based on the Node-RED tool installed on the Raspberry Pi and the IBM Cloud.

Key words – IoT; Node-RED; DHT11 sensor; Raspberry Pi.

I. INTRODUCTION

The Internet of Things (IoT) concept enables connecting the physical things with embedded electronics, software, sensors and connectivity thus providing data exchange with manufactures, operators and/or other connected devices. In 2013, *Global Standards Initiative on Internet of Things* (*IoT-GSI*) defined IoT as "the global IT infrastructure that provides advanced services (physical and virtual) by networking of things, based on existing and interoperable information and communication technologies in the development" [1]. For this purpose, the term "thing" represents "the object of the physical world (physical things), the information or the word (virtual things), which can be identified and can be integrated into communication networks".

The IoT allows things to be noticed and remotely controlled by the existing network infrastructure, enabling more direct integration of the physical world and computer systems, resulting in increased efficiency, accuracy and economic benefits, while reducing human intervention. By adding sensors to the IoT concept, a variety of technologies are produced, such as smart homes, smart cities, smart grid, intelligent transport, and virtual power plants. Each thing can be uniquely identified through an embedded computer system and is interoperable within the existing internet infrastructure. The term "Internet of Things" was proposed by Kevin Ashton in 1999 [1].

It is estimated that by 2022 there will be over 50 billion interconnected devices [2]. The consequence of the increase in internet clients, and thus the increase in devices connected to the Internet is the adoption of IPv6 (*Internet Protocol version 6*), because each device needs the unique IP address to provide internet connectivity.

The use of IPv6 provides, in theory, $3{,}403 \times 10^{38}$ unique addresses. Due to the extremely large amount of data (*Big Data* concept) the sensors collect in real time, they are in most cases stored and processed on a Cloud [3]. Their processing gives us a much more detailed insight into how things around us work, where problems arise, and where and how to react. The IoT already transforms our everyday life since many objects are equipped with miniature identification devices or machine-readable identifiers.

Paper is organized as follows; Section II presents a brief introduction in Node-RED development tool. Section III gives an overview on the Cloud technology and Cloud platform (Platform as a Service). Section IV provides implementation details. Finally, Section V gives conclusion and future work.

II. NODE-RED DEVELOPMENT TOOL

Node-RED is an open source flow-based development tool for the integration of IoT hardware devices, APIs (Application Programming Interfaces) and online services developed by IBM Emerging Technology [4]. Node-RED is a free JavaScript-based tool, built on Node.js platform, which provides a visual browser-based flow editor. The system contains nodes that are represented by appropriate icons. It can operate in two ways: drag, drop and wire up nodes, or import JavaScript code.

The node provides different functions, such as to monitor the flow as the *debug out* node, or to read and write with GPIO pins of Raspberry Pi as the Raspberry Pi node. Created flows are stored using JSON (*JavaScript Object Notation*). Node-RED enables developers to wire up *input*, *output* and *processing* nodes in order to create flows for data processing, controlling things, or sending alerts [5]. It works on the following principle: it enables the connection of web services or customizes nodes to one another or to things, in order to perform functions such as sending sensor data via e-mail or to services like Twitter, easily performing complex analysis, etc.

Node-RED has three basic components (Fig.1):

1. Node Panel,
2. Flow Panel,
3. Info and Debug Panel.

Node-RED is a flexible and powerful tool that is used to create prototypes. This system allows quick creation of applications, especially applications that trigger on an event such as IoT applications. The essence of this tool is to enable engineers and technicians to simply create and configure real-time applications on end-devices.

978-1-5386-4907-7/18/$31.00 ©2018 IEEE

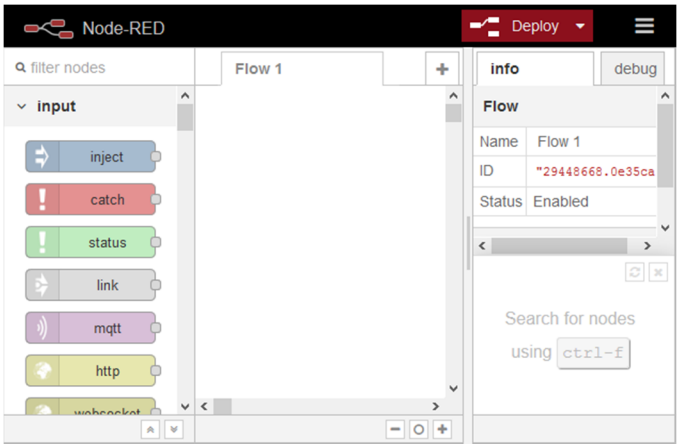

Figure 1. Node-RED editor.

Parts of program's code can be used multiple times, which allows wide usage [4]. IBM declared Node-RED for *the open source JS Foundation* project in 2016 [6].

III. Cloud Technology

Cloud technology represents the delivery of computing resources and storage capacity as a service for a heterogeneous group of end users. The concept of Cloud technology relies on sharing resources over the network, most often the Internet. Therefore, the Cloud in some way provides service delivery instead of the product itself. End-users access Cloud applications through a web browser or mobile application on a mobile phone, while software and user data are located on servers in a remote location [7].

The Cloud provides, at any time instance, the ability to access applications, data, data storage services, and does not require from the user to know the physical location of the service system. The Cloud is largely present in particular form, such as social networks, e-mail services or smart phones, etc. Moreover, the Cloud is a service that offers unlimited amounts of all resources (hard disk, processor, memory, etc.) when we really need it and to the extent that it suits our needs by doing everything we can to independently control it [7].

There are three types of Cloud technology:

- Software as a Service, SaaS,
- Platform as a Service, PaaS,
- Infrastructure as a Service, IaaS.

The basis of Cloud technology is a convergent infrastructure, made up of different IT technologies linked to one logical and functional entity, such as abstraction of physical resources by virtualization, as well as resource sharing. In the computing Cloud model, we distinguish two parts of the system: the *front end*, which is a user part and includes all the parts of the user-controlled infrastructure as well as the user's own way of accessing the service and the *back end*, which includes an infrastructure of Cloud providers.

A. Cloud platform (Platform as a Service)

Service refers to the development environment and the required package of software subsystems. The user can develop, test, and distribute his own applications that run on the infrastructure of a Cloud service provider. Provider is in charge of a platform and executable environment, which most often include: servers, network infrastructure, data center, operating systems, and programming languages.

The user has control over applications and the middle layer, while the Cloud service provider controls other layers of infrastructure, but the user can have the ability to choose the structure of the environment. In general, the team working on software development is not limited by the geographic location of resources or other team members. Examples of Cloud platforms are Amazon Elastic Beanstalk, IBM Bluemix, Google App Engine, and Microsoft Azure.

Platform services can be used for applications or to implement a private Cloud environment. The platform provides capabilities such as easy and fast scaling and expanding resources as needed, adding new servers or services, high availability and minimal downtime in the event of a failure some segment of the Cloud infrastructure itself or server, with the automatic migration and restart of the server on a healthy segment of the Cloud platform.

Cloud platform implies using the operating system via the Internet, without the need for download and installation. PaaS offers a variety of Cloud services to support all stages of the application development cycle, including the integrated development environment (IDE), source code control, version control, code tracking, interactive tests for multiple users, and more. PaaS solutions are development platforms in which development tools are located on Cloud and accessed using the web browser [7].

B. IBM Bluemix Cloud platform

IBM Bluemix is a Cloud platform as a service (PaaS) developed by IBM. It is used to build, run, deploy and manage applications on the Cloud. Bluemix is based on Cloud Foundry open technology and runs on SoftLayer infrastructure [6]. Bluemix supports several programming languages and platforms including Java, Python, Node.js, PHP, Swift, Ruby Sinatra, Ruby on Rails, Go etc. It was announced as a public beta in February 2014 and generally available in June. On October 2017, IBM announced that they are merging the Bluemix brand with the IBM Cloud brand [8].

IV. Implementation

A. Installation of Node.js and Node-RED on Raspberry Pi 3, model B

Raspberry Pi is a single-board computer, size of credit card, developed in the United Kingdom by the Raspberry Pi Foundation. The Raspberry Pi 3 is the third-generation Raspberry Pi released in February 2016. This model has following characteristics [9]:

- Quad Core 1.2GHz Broadcom BCM2837 64bit CPU,
- 1GB RAM,

- BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board,
- 40-pin extended GPIO,
- 4 USB 2 ports,
- 4 Pole stereo output and composite video port,
- Full size HDMI,
- CSI camera port for connecting a Raspberry Pi camera,
- DSI display port for connecting a Raspberry Pi touchscreen display,
- Micro SD port for loading your operating system and storing data,
- Upgraded switched Micro USB power source up to 2.5A.

Before installing the Node.js module and Node-RED editor, it is necessary that Raspberry Pi has an operating system installed, Raspbian or NOOBS (*New Out Of Box Software*), which can be found on the official Raspberry Pi web page [9].

Raspbian is the Foundation's official supported Operating System. There are several versions of Raspbian including Raspbian Stretch and Raspbian Jessie. Due to the fact that it is made by Pi Foundation, Raspbian is versatile and has no competitors. Raspbian is based on the Debian Linux computer operating system and comes with over 35,000 packages pre-compiled software. It is estimated that there are over eighty Linux-based distributions for the Raspberry Pi, but there are also non Linux-based distributions. These range from Raspbian, Pidora, ArchLinux, to Linutop and PiBang for Linux-based distributions. But, not every Linux-based distribution works on every type of Raspberry Pi, since it depends on the architecture of integrated ARM central processing unit (CPU). NOOBS allows the user to install a couple of operating systems and to switch between them. The operating systems currently included in NOOBS are: Raspbian, Pidora, LibreELEC, OSMC, RISC OS and ArchLinux.

The command for installing Node.js module and Node-RED editor is as follows:

```
bash <(curl -sL https://raw.githubusercontent.com/node-red/raspbian-deb-package/master/resources/update-nodejs-and-nodered)
```

Node-RED runs from the terminal with command: **node-red-start**. Accessing over the browser by entering the IP address of Raspberry Pi and the 1880 port, a Node-RED interface will be displayed, confirming that the installation was successful. The appearance of the Node-RED editor is shown in Fig.1. Pressing CTRL + C stops the execution of the tool, and Node-RED is switched off by the command: **node-red-stop**.

B. DHT Sensor Node installation

The next step is to install the DHT sensor library written in JavaScript for the Node.js module. In order to allow direct access to data from the DHT11 sensor, the dependencies should be installed first, the BCM2835 library written in the C language and the Node.js module which is already installed. The commands below will install the BCM2835 library:

```
wget http://www.airspayce.com/mikem/bcm2835/bcm2835-1.52.tar.gz
tar zxvf bcm2835-1.52.tar.gz
cd bcm2835-1.52
./configure
make
sudo make check
sudo make install
```

With the installed library, Node.js module is added to ensure that Node-RED interacts with the BCM2835 library. By executing the following command, the DHT sensor library is installed:

```
cd ~/.node-red
npm install node-dht-sensor
```

The node-dht-sensor is configured in the Node-RED context by accessing the *settings.js* file and setting the Global Context function line. *Settings.js* file is accessed by the following command:

```
nano ~/node-red/settings.js
```

The setup of Global Context function is shown below:

```
functionGlobalContext: {
    sensorLib:require('node-dht-sensor')
},
```

Node-RED runs with the already known command: **node-red-start**, and then starts in the browser.

C. Circuit construction

The DHT11 sensor has four pins: V_{CC}, signal, no connection, and GND. V_{CC} pin is connected to Raspberry Pi GPIO (*General Purpose Input/Output*) pin of 5V, the signal pin is connected to GPIO 04 pin and GND pin of the sensor to the GND GPIO pin of the Raspberry Pi, as shown on Fig.2.

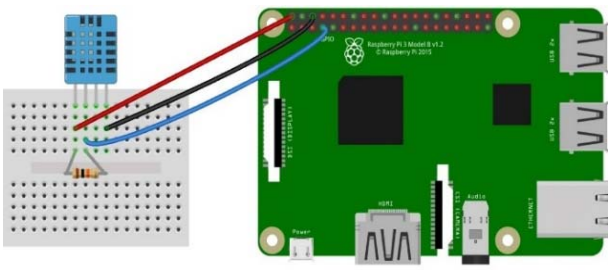

Figure 2. DHT11 sensor and Raspberry Pi circuit.

D. Flow construction in Node-RED editor

The flow construction is shown on Fig.3. The configuration of input *inject* node is shown below:

```
Payload: timestamp
Repeat: interval
every 3 seconds
```

The function of this node is to trigger a process that starts an algorithm which is executed by the sensor. The trigger is set to 3 seconds. *f* node named *read-sensor* represents a function block that allows programming in JavaScript language.

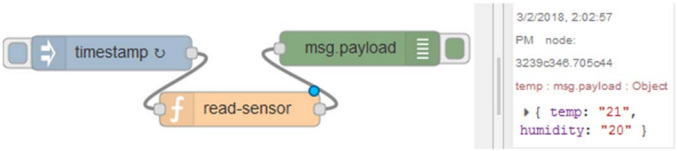

Figure 3. Flow construction.

The input *inject* node starts the function block, which transmits the message. The source code of the function is as follows:

```
var sensor = context.global.sensorLib;
var readout = sensor.read(11, 4);
console.log('temp: ' + readout.temperature.toFixed(1) + '°C,' +
'humidity: ' + readout.humidity.toFixed(1) + '%');
msg.topic = "temp";
msg.payload = {
temp: readout.temperature.toFixed(0),
humidity: readout.humidity.toFixed(0)};
return msg;
```

The last part of the flow is a *debug* node. The *debug* node shows the data received from the sensor that was processed by the function block, in the debug tab. After creating a flow, the Deploy button is pressed and the data received from the sensor, e.g. temperature and humidity should be displayed in the debug tab.

E. Application construction on Bluemix Cloud

The IBM Bluemix Cloud requires an account to create an application. Different types of applications can be created, one of them is Node-RED. Bluemix Cloud provides the URL of created Node-RED application (in this particular case: https://dht11sensor.eu-gb.mybluemix.net/). The application is accessed in the browser with the obtained URL. A flow that receives data from the Node-RED on Raspberry Pi is generated. The connection is made using a *websocket* node. In Nod-RED application, *websocket in* node is used as a server, and in Node-RED on Raspberry Pi, *websocket out* node is used as a client. The configuration of *websocket in* node, as a server, is shown below:

```
Type: Listen on
Path: /ws/temp
```

The configuration of *websocket out* node, as a client, is as follows:

```
Type: Connect to
URL: wss://dht11sensor.eu-gb.mybluemix.net/ws/temp
Send/Received payload
```

The URL refers to the URL of *websocket* node as a server, i.e. the URL of *websocket* node in the Node-RED application. The WebSocket Protocol provides full-duplex communication between a client running untrusted code in a controlled environment, and a remote host [10]. It provides creating persistent and low latency connection which supports the transaction initiated by client or server. The security model used for this is the one commonly used by web browsers. The protocol consists of a connection establishment process known as an opening handshake followed by basic message framing, layered over TCP (*Transmission Control Protocol*). Data are transferred as *messages*, each of them consists of *frames* containing the data that is being sent (the payload). Properly message reconstruction should be ensured when message reaches the client. Therefore, each frame has data prefix, with length of 4-12 bytes. This frame-based messaging system benefits in significant latency reduction. The communication uses TCP port number 80 which is appropriate for the environments that do not allow non-web internet connections with a firewall. The essence of this technology is to provide a mechanism for browser-based applications that need full-duplex communication with servers that does not rely on opening multiple HTTP (*Hyper Text Transfer Protocol*) connections.

The WebSocket Protocol is used due to its simplicity, but the optimal solution for IoT applications should be based on more robust protocols such as MQTT (*MQ Telemetry Transport*) [11]. Fig. 4 and Fig.5 show relevant flows, where the result of the connection is shown on Fig.5 in debug tab.

This IoT application, implemented by using IBM Bluemix Cloud, could be run on any device that has Internet connection (access over web-browser). Moreover, a mobile application that use data from Node RED can be accessed from user's mobile phone.

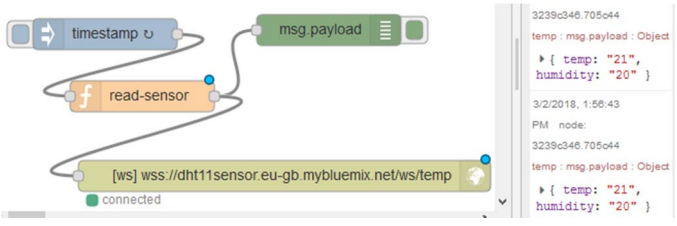

Figure 4. Flow on Node-RED of Raspberry Pi and the result of connection with the sensor in debug tab.

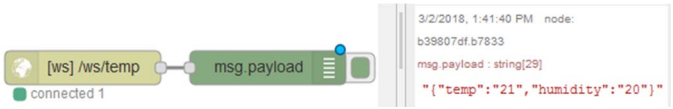

Figure 5. Flow of Node-RED application and the result of connection with Node-RED on Raspberry Pi in debug tab.

V. CONCLUSION AND FUTURE WORK

The paper presents the implementation of an IoT application, created by using Raspberry Pi 3 model B, DHT11 temperature and humidity sensor, and IBM Bluemix Cloud. The application on Cloud shows data collected from Raspberry Pi and DHT11 sensor using Node-RED and WebSocket Protocol. With installed Node-RED and the DHT Sensor Library, flows which allow devices to send data and process commands can be built. Node-RED enables collecting big data in the IoT systems and therefore provides a server function. Data on Node-RED are stored in internal data base. The collected data can be stored in data base created on some DataBase Server using nodes in Node-RED that enable it. Therefore, application retrieves data from that data base. Furthermore, collected data can be processed to obtain the desired information and to carry out statistics and analyses. Future work relates to creating and investigating various application scenarios using different sensors, in order to test different IoT concepts [12].

REFERENCES

[1] M.Ruggieri and H.Nikookar, "Internet of Things – From Research and Innovation to Market Deployment".

[2] E. Park, Y. Cho, J. Han, and S. J. Kwon, "Comprehensive Approaches to User Acceptance of Internet of Things in a Smart Home Environment," IEEE Internet of Things Journal, pp. 2342-50, 2017.

[3] S. H. Shah and I. Yaqoob, "A survey: Internet of Things (IOT) technologies, applications and challenges," IEEE SEGE, pp. 381-5, 2016.

[4] Node-RED guide, Web site: http://noderedguide.com/

[5] G. Gardasevic, H. Fotouhi, I. Tomasic, M. Vahabi, M. Bjorkman, M. Linden, "A Heterogeneous IoT-based Architecture for Remote Monitoring of Physiological and Environmental Parameters," 4th EAI International Conference on IoT Technologies for HealthCare, Angers, France, Oct. 24-25, 2017.

[6] IBM Bluemix, Official Web site: https://console.ng.bluemix.net/

[7] IBM Cloud, Web site: https://www.ibm.com/developerworks/cloud/library/cl-cloud-technology-basics/index.html

[8] IBM, Official Web site: https://www.ibm.com/us-en/

[9] Raspberry Pi, Official Web site: https://www.raspberrypi.org/

[10] The WebSocket Protocol, IETF RFC 6455, December 2011.

[11] MQTT, Official Web site: http://mqtt.org/

[12] M. Serrano, H. N. M. Quoc, and M. Hauswirth, "Open services for IoT cloud applications in the future internet," IEEE 14th International Symposium on WoWMoM, pp. 1-6, 2013.