IMPLEMENTATION OF NODE.JS TECHNOLOGY IN A CONCEPT BASED ON IOT

Bejtovic Muhamed[1], Aleksandar Žorić[1]
[1]*Faculty of technical sciences – University of Priština*

Abstract

With the development of Internet of Things (IoT) concept, it is possible to provide more efficient management and data exchange by connecting physical devices with sensors, actuators and the Internet. Experts estimate that IoT will make up to 30 billion items by 2020, and that the global market value of IoT will reach $ 7.1 trillion by 2020. Parallel to the popularization of the Web service, there is a significant development from the aspect of JavaScript server technology, which leads to the development of one of the most popular technologies, Node.js. This paper presents a solution that gives effective results at reducing the response time of physical devices connected to the Internet in the management / monitoring process and is based on the ATMega 2560 microcontroller and Node.js technology.

Keywords:. **ATMega2560 , JavaScript, Node.js,IoT**

INTRODUCTION

The IoT concept provides flexibility concerning communication and information. It is an approach that is relevant in many different environments. IoT solutions are applied in all areas of human life, from personal things such as "smart home", through solutions as "smart" cars, cities, medicine, agriculture, transport and environmental protection. Due to the development of modern technologies on the market, microcontroller platforms such as Arduino, BeagleBone or Raspberry, which are used in IoT technology, are available. Different conditions are set when designing the IoT system, depending on the specific problem and application area. In cases where a very short response time is required, a solution is proposed that is immediately applicable and in most cases economically viable.

The solution is based on a connection between the Arduino Mega 2560 microcontroller board [1], which in this case connects via usb cable to a PC / laptop, then a PC with Node.js installed [2], which represents the connection between the microcontroller board and Internet networks, and of course the Internet network that represents the connection between the user and the web server. The appropriate sensor and relay module are connected to the Arduino mega 2560 microcontroller plate.

When designing this IoT system, the goal for this solution was to have the multiple application areas that are without IT infrastructure, in order to maximize the utilization of existing resources, so that such a system is sustainable in the long run and that the initial investment is minimal.

SYSTEM ARCHITECTURE

The basic block diagram of the proposed system is shown in Figure 1. The microcontroller is used to obtain the physical condition value through the sensors associated with it. For programming microcontroller, the Arduino development environment "Arduino IDE" (Integrated Development Environment) [3] was used. Arduino IDE provides easy communication between microcontrollers and computers via a USB cable. The control software is programmed by the microcontroller on the ATMega 2560 development system, so it communicates with the outside world and the client.

Once the program is first entered into the microcontroller, another program can be re-entered, even the third and fourth ones, etc.

The system is designed to satisfy the following requirements:

- View the current state of the switch status
- Monitoring of temperature
- Monitoring humidity
- Motion detection monitoring.

Fig. 1.
Block diagram of the presented solution

The software part of the project was developed for the Windows platform in the HTML programming language, where CSS, JavaScript as well as Node.js technology are used, which is very customized for applications where it is necessary to maintain a permanent connection between the client and the server. In this case, the largest number of devices that can be controlled is eight, because the eight-relay module is used. A large increase in the popularity of the use of asynchronous techniques in the creation of software has enabled JavaScript to be executed in the Web browser, finally moving it away from the user's side to the server side. The most basic purpose of Node.js, which uses JavaScript for its work, is to serve to create a Web application server.

Together with the Google V8 JavaScript engine, it achieves a few times the speed of other scripting languages like Ruby and Python.Non-blocking architecture Node.js is ideal for creating applications that update client-side data in real-time. (eng.real-time). Applications can be run within Node.js runtime on OS X, Microsoft Windows, Linux, FreeBSD, and IBM platforms. Node.js very quickly processes a large number of requests at the same time.

The main part of the software code are embedded libraries for "HTTP" and "socket.io", which enable two-way communication in real time. For communication, the "Websocket" protocol is used, which enables one continuous TCP connection between the server and the client, and it is standardized as RFC 6455 in 2011. by IETF. Websockets are supported in all well-known Web browsers such as Google Chrome, Internet Explorer, Firefox, Safari and Opera.

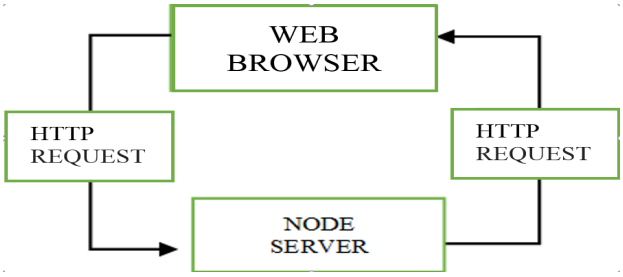

Fig. 2. *The process of opening a website*

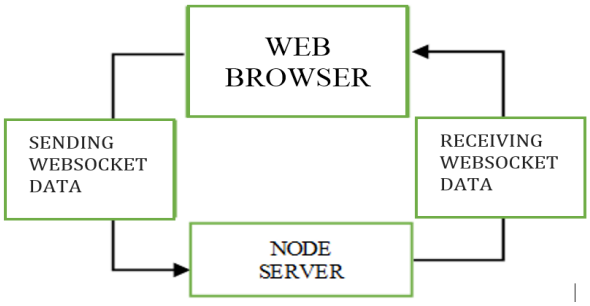

Fig. 3. *Process during some action in the website*

Node.js can control the simultaneous copying of HTTP and WebSecure, using only one TCP / IP port as shown in Figures 2 and 3. It is important to note that when installing on Node.js, modules are installed to enable HTTP functionality.

Comparative analysis [4] showed that on average one HTTP request lasted about 107ms and Socket.io required 83ms. For a number of parallel requests, things began to look quite different, so for fifty requests, the request lasts about 5 seconds for the HTTP request and about 180ms for the Socket.io request. Total HTTP enabled it to complete about 10 requests per second while Socket.io can, at the same time, submit nearly 4,000 requests. The main reason for this big difference is that the browser limits the number of concurrent HTTP connections (6 by default in Chrome)

HARDWARE REALIZATION

In this paper, the main part is the software part of the project, will be given only the basic characteristics of the hardware components used in the project.

Basically, these components are divided into:

• Controllers: to make choices based on advance programmed rules and events.

• Sensors: they serve to monitor and forward messages in case of a change in condition.

• Actuators: used to perform physical actions.

• Networks: allow communication between units and environment.

• Interface: for communication between the user and the system.

Different sensors and actuators can significantly expand the capabilities of the described system.

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila. It is shown in figure 4.

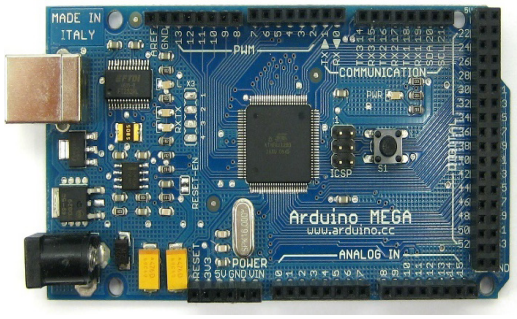

Fig. 4. *Arduino Mega 2560 Microcontroller Platform*

Also the Arduino relay module is used, with 8 relays and smd LEDs for input indication [5]. Relays are with switching contact (one open, one closed). The relay characteristics are: 230 VAC / 10 A (30VDC / 10A). The input is TTL with 3.3-5V, and the module also requires an external power supply of 5 - 7.5 V.

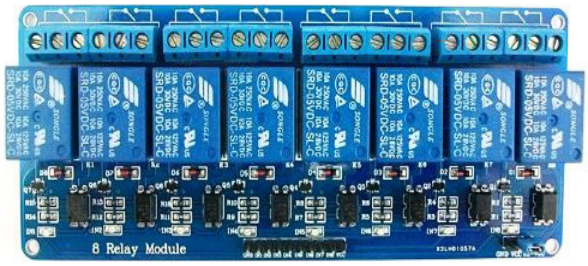

Fig. 5. *8-relay module*

In order to make some more serious device control system sense and working, it is necessary to install the sensors from which the necessary data will be obtained.

The project used:

• Motion detector - the passive infrared sensor (PIR). Its function is to detect motion [6]. The angle the sensor sees is 120 degrees and reaches up to 7 meters. It has two potentiometers: for sensitivity detection and delay (used to check how offen there is movement).

Fig. 6. *PIR sensor*

Fig. 7. *DHT11 sensor*

• DHT11 [7] uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin (no analog input pins needed). It's fairly simple to use, but requires careful timing to grab data. The only real downside of this sensor is one can only get new data from it once every 2 seconds.

In order to implement and demonstrate the system developed theoretically, we created a prototype. Thus the whole system that is being developed is given below figure 8.

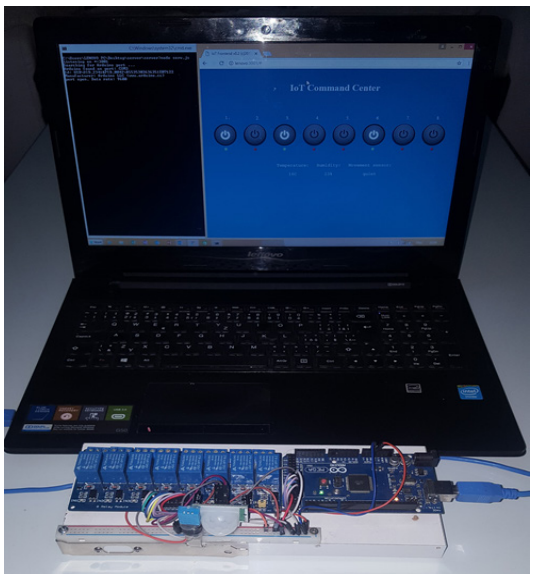

Fig. 8. *Prototype of the whole system*

Devices can be connected to the relay module, where we can manage their turning on or off using the Web site.

The process starts by running the Node.JS server, as shown in the fig.8. below.

```
C:\Users\LENOVO PC\Desktop\FOLDERS\server\server>node serv.js
listening on *:3001
Searching for Arduino port ...
Arduino found on port: COM3
Id: USB\VID_2341&PID_0042\85535303636351E07122
Manufacturer: Arduino LLC (www.arduino.cc)
port open. Data rate: 9600
```

Fig. 9. *The process starts by running the Node.JS server*

Therefore, after accessing the Web site, the appearance which is shown in figure 10, in one of the Internet browsers (supported by Google Chrome, Internet Explorer, Firefox, Safari and Opera), from the left to the right are the "keys" that can be started and extinguish the relays that are connected to the Arduino MEGA microcontroller board.

At the same time, the relay statuses are visually displayed: red means that the relay is turned off and the green is turned on.

The visual effect of the status is also added on the buttons on which the relays are switched on and off, as well as the sound of the "push off a button" at the moment when the relays are on and off.

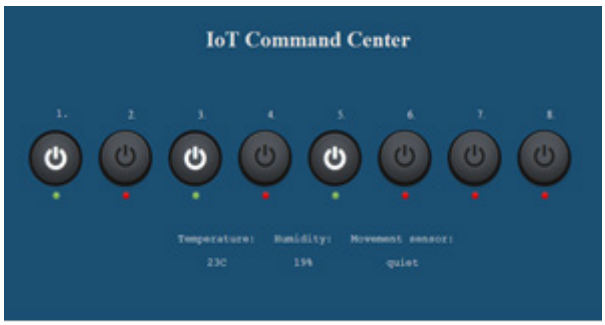

Fig. 10. *Website layout*

The second line shows the data from the connected sensors:

- Temperature (DHT11),
- Humidity (DHT11),
- Movement sensor (PIR Sensor).

CONCLUSION

The application of this IoT solution in operation is characterized by further optimization and efficiency in the device's contours, but alongside other hardware components of the request and the computer / laptop on which it will be installed and running Node.js.

So, in the domain of application where there are requirements for the speed of control over a particular device, the proposed solution gives effective results, and presents a different approach to how to solve the problem. The advantage of the proposed solution is certainly that the computer enables the processing of the received data from the sensor, and of course the ability to use computer resources, and to easily upgrade the system. Arduino is also open-source, which means it is constantly evolving and is accompanied by a lot of technical documentation. Arduino is ideal for starting up with microcontrollers as well as for complex projects. For most people, perhaps the most important advantage is Arduino's price. Although cheap it is very reliable and gives good performance.

ACKNOWLEDGEMENT

This work was supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia within the project III 47016.

REFERENCE

[1] Charalampos Doukas, “Building Internet of Things with the Arduino”, CreateSpace Publications, 2012

[2] Node.js, "Home page of Node.js," Joyent, 2016.Available:https://nodejs.org/en/. [Accessed 01 May 2016].

[3] Arduino, Available:http://www.arduino.cc/

[4] https://blog.feathersjs.com/http-vs-websockets-a-performance-comparison-da2533f13a77

[5] https://www.jaycar.com.au/medias/sys_master/images/8955340357662/XC4418-dataSheetMain.pdf

[6] https://cdnlearn.adafruit.com/downloads/pdf/pir-passive-infrared-proximity-motion-sensor.pdf

[7] Robotics D, “DHT11 Humidity & TemperatureSensor”,2010,www.micro4you.com/files/sensor/DHT11.pdf