

IRID: Infraestructura para la recolección inalámbrica de datos provistos por sensores en el marco de robótica educativa en nivel secundario**Alumno:** Juan Pablo Lozano Arce¹**Directores:** Alejandra Beatriz Lliteras^{1,2} y Andrés Rodríguez¹¹ UNLP. Facultad de Informática, Centro LIFIA ² CICPBA

Contacto {jlozano, alejandra.lliteras, andres.rodriguez}@lifia.info.unlp.edu.ar

Tipo de Trabajo: Tesina de grado en curso**Palabras Claves:** MicroBit, ESP, WiFi, Protocolo, Servidor, API Rest, Programación basada en Bloques, Comunicación Inalámbrica, Recolección de Datos, Visualización de Datos**Síntesis**

En este trabajo se propone una infraestructura que combina software y hardware para que sea posible, por un lado, recolectar datos mediante sensores conectados a placas MicroBit (V2), y por otro, enviar dichos datos, vía Wifi, a un servidor para su almacenamiento y posterior visualización. El objetivo es que esta infraestructura pueda ser usada de manera sencilla y guiada por docentes y estudiantes de nivel secundario, y que permita trabajar aspectos de pensamiento computacional empleando programación basada en bloques. En este trabajo se propone: a) el diseño e implementación de un servidor que recibe, valida, almacena y permite visualizar datos, b) un protocolo de comunicación entre placas MicroBit, ESP8266 y un servidor Web, c) una extensión al entorno MakeCode de programación basada en bloques para placas MicroBit d) guías para la preconfiguración de la placa ESP y para desplegar el servidor

1 Motivación

El desarrollo del Pensamiento Computacional al considerar temas de robótica o IoT en el área de educación, suele trabajarse a partir de kits que ya vienen pre armados y con instrucciones concretas para que los docentes puedan trabajar en el aula usando placas como, por ejemplo, MicroBit¹ o Arduino². Para bajar la complejidad que implican las placas Arduino para estudiantes de secundarios no técnicos, Pech y Novák sugieren el uso de placas MicroBit por sobre las otras ya que, entre otros aspectos, traen componentes integrados.

Al conectar sensores a una placa MicroBit y querer enviar vía Wifi los datos recolectados, surge la necesidad de conectarla a una placa ESP para que sea esta quien realice el

¹ <https://microbit.org/>² <https://www.arduino.cc/>

envío. En particular, se espera visualizar mediante gráficos los datos recolectados y que tanto la recolección de los datos, como su envío vía Wifi pueda ser programado de manera intuitiva usando programación basada en bloques. Es objetivo de este trabajo que la infraestructura formada por sensores, placa MicroBit, ESP y servidor Web, pueda ser replicados en diferentes espacios de trabajo, como por ejemplo escuelas, sin necesidad de expertos en tecnología.

2 Aporte

El trabajo presenta diferentes aportes, a continuación, se detalla cada uno de ellos

2.1 Servidor:

Para este trabajo se propone un servidor web. La Fig. 1 muestra en esquema de la solución del trabajo, en esta sección se hace foco en el servidor. A continuación, se describen los diferentes componentes.

FastApi (REST) es el componente encargado de recibir, analizar y tratar la información enviada por la ESP8266 vía wifi. Está desarrollado en el lenguaje de programación Python. *MongoDB* es el componente encargado de guardar la información recibida por FastApi (REST). *Grafana* es el componente encargado de la visualización de los datos. Soporta varios tipos distintos de orígenes de datos. En este proyecto se utilizan las direcciones http de consulta de FastApi (REST) para obtener los datos a mostrar.

Swagger provee una interfaz web para probar el correcto funcionamiento de las direcciones http de FastApi (REST). *MongoExpress* provee una interfaz web para ver y manipular los datos almacenados por MongoDB.

Todos los componentes están unificados bajo Docker, un entorno de virtualización que permite automatizar las configuraciones, instalación de software necesario para el funcionamiento de cada componente y el funcionamiento en varios sistemas operativos distintos (Windows, Mac, Linux).

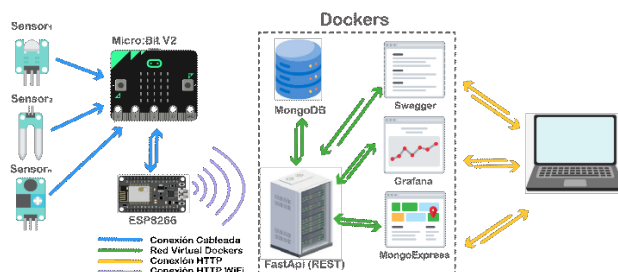


Fig. 1: Detalle del Servidor

Si bien el foco de la Fig.1 es mostrar el esquema del servidor, también puede apreciarse la manera en la que los sensores se conectan a la `placa MicroBit y ésta a la ESP8266 quien envía los datos via Wifi.

2.2 Protocolo de Comunicación

En esta sección se muestra el protocolo de comunicación entre la placa MicroBit y la placa ESP8266 y entre la placa ESP8266 y el FastApi (REST) o Rest Server.

La Fig. 2 muestra como es el flujo de comunicación entre los componentes mencionados.

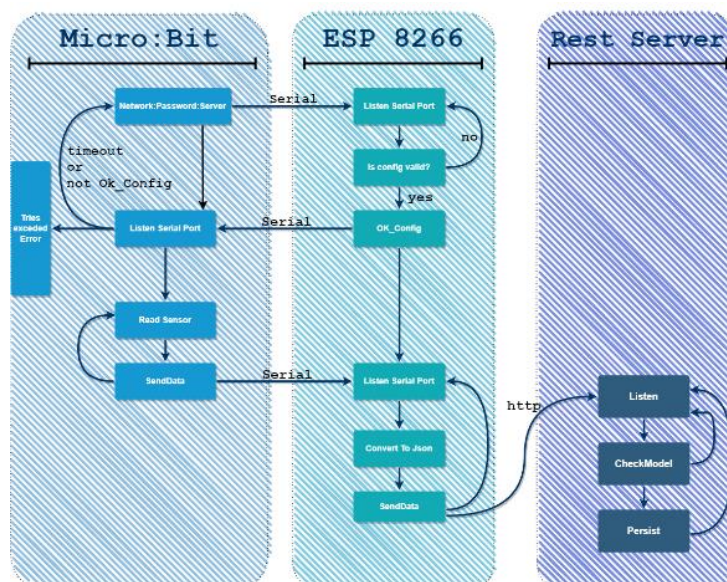


Fig. 2:Protocolo de comunicación

En un principio la MicroBit envía los parámetros de configuración de red a la ESP8266. La ESP8266 los recibe, si los parámetros son correctos le envía OK a la MicroBit y empieza a esperar que la MicroBit le envíe datos de los sensores. Si los parámetros no son correctos vuelve a esperar a que la MicroBit le envíe los parámetros de red. Una vez que la MicroBit le envía los parámetros de red a la ESP, la MicroBit se queda esperando que la ESP8266 le diga si los parámetros son válidos, si los parámetros están bien la MicroBit empieza la tarea de leer los sensores y enviar los datos a la ESP8266, seguirá haciendo esto hasta que se apague. Si los parámetros no son correctos y no agotó las chances de configurar la ESP8266 le enviará nuevamente los parámetros, si agotó las chances mostrará un mensaje de error. Una vez que se chequearon los parámetros y la MicroBit está en etapa de lectura de sensores y envío de datos, la ESP8266 se queda en un bucle de esperar los datos de la MicroBit, convertirlos a formato json y enviarlos a FastApi (REST) o Rest Server (Fig. 2). Por último, FastApi (REST) está esperando conexiones desde que empieza a funcionar, cuando recibe una conexión verifica que los datos cumplan con ciertos parámetros y de ser así los guarda en la base

MongoDB. En caso de no cumplir los parámetros no guarda los datos en la base y vuelve a quedarse esperando nuevas conexiones.

Notar que están fuera del alcance de este trabajo los aspectos de seguridad y de validación de arribo correcto de los datos.

2.3 Extensión al entorno MaKeCode

Con el fin de encapsular la complejidad relacionada a configurar la placa ESP y al envío de datos, se creó una extensión con una familia de bloques para el entorno MakeCode. El bloque “*setup_esp*” permite setear el nombre de la red, la contraseña de acceso, la url del servidor y un número entero que representa la cantidad de intentos a realizar antes de avisar que hay un error. El bloque de “*send data*” envía los datos avisando desde donde se toma, que tipo de sensor y el dato recolectado por el sensor. Por último, el bloque “*ping*” sirve para validar el estado del servidor. Adicionalmente, se sumó una ayuda contextual que se accede al pasar por sobre cada bloque. Lo anterior puede visualizarse en la Fig.3.

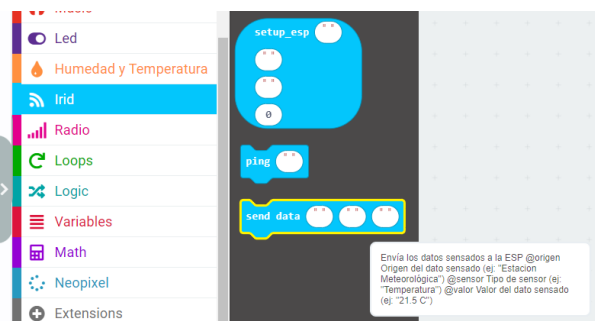


Fig. 3: Familia de bloques de la extensión para MakeCode

La creación de estos tipos de bloques especiales se hace mediante la edición de un archivo JavaScript titulado “*custom.ts*” dentro del entorno MakeCode (Fig. 4). Este archivo se crea por defecto al momento de la creación de los proyectos y se accede mediante la pestaña de modo JavaScript.

2.4 Guías de preconfiguración de la ESP y despliegue del servidor

Se desarrollaron dos *guías*, una relacionada al servidor y otra para preconfigurar la placa ESP. Una primera guía que explica paso a paso como poner en marcha el servidor para la recepción, almacenamiento y visualización de datos (llamada “*IRID Instructivo para el despliegue de ambiente*”³). La segunda guía, dada que la forma de comunicar la placa MicroBit con la placa ESP8266 es mediante el puerto serie de estas, es necesario realizar preconfiguraciones en la placa ESP para usarlos, por tal motivo y por

³ “*IRID Instructivo para el despliegue de ambiente*” está disponible en <https://zenodo.org/records/13234008>

única vez es necesario cargarle esa preconfiguración a la placa. Para este fin se creó una guía con el paso a paso (llamada “*IRID Instructivo de programación de la ESP⁴*”).

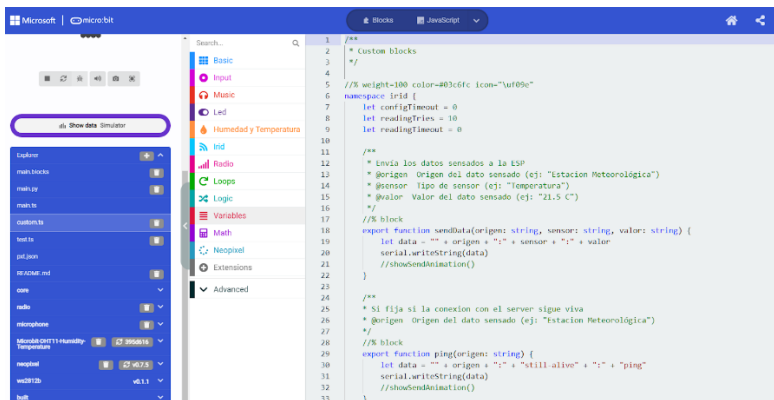


Fig. 4: Archivo custom.ts

3 Líneas de Investigación Futura

Se proponen como futuras líneas de investigación:

- Analizar otro tipo de protocolos para el envío de datos como alternativa a REST, como por ejemplo MQTT.
- Ampliar el protocolo de comunicación para que exista una validación bidireccional respecto al arribo de los datos y reenviar en caso de falla.
- Ampliar a más cantidad de gráficos la visualización de datos
- Mejorar la usabilidad de las guías destinadas a usuarios finales
- Conformar un kit de solución al estilo “Hágalo usted Mismo”

4 Bibliografía Básica

- Fernández Quiñonez, S. L. (2022). Implementación de sistemas iot utilizando técnicas de programación visual.
- Pech, J., & Novák, M. (2020). Use Arduino and micro: bit as teaching platform for the education programming and electronics on the stem basis. V International Conference on Information Technologies in Engineering Education (pp. 1-4). IEEE.
- Pradeep, A. (2023). Enabling IoTs with ESP32 for Affordable Education. 5th International Conference on Inventive Research in Computing Applications (pp. 1368-1373). IEEE.
- <https://www.alldatasheet.es/datasheet-pdf/pdf/1132995/ESPRESSIF/ESP8266.html>
- <https://support.microbit.org/support/solutions/articles/19000119052-details-of-microbit-v2>

⁴ “*IRID Instructivo de programación de la ESP*” está disponible en <https://zenodo.org/records/13240796>