INVESTIGACIÓN e INNOVACIÓN en INGENIERÍAS
ISSN2344-8652

Ambiente visual integrado de desarrollo para el aprendizaje de programación en robótica

Comprehensive Visual Development Environment for Learning Robotics Programming

Carlos Alejandro Ruíz Ramírez **Jovani Alberto Jiménez-Builes**

Universidad Nacional de Colombia

Diana María Montoya Quintero
Instituto Tecnológico Metropolitano

Recibido: 28/07/2020
Aceptado: 27/11/2020
Publicado: 4/01/2021

Correspondencia de autores:
caaruizra@unal.edu.co

Resumen

Objetivo: Revisar algunos aspectos y características positivas del uso de la robótica en ambientes educativos y describir algunas de las características de los lenguajes de programación utilizados en su implementación. **Metodología:** se explica como un ambiente de desarrollo implementado mediante el uso de algunas características de computación en la nube, puede ser usado para aplicar robótica en escuelas, colegios y universidades. El entorno de programación combina la programación gráfica utilizando notación de procesos de trabajo (BPM) y programación textual. **Resultados:** se demostró que se puede hacer una transición gradual de lenguajes visuales a lenguajes de programación textuales, así como las ventajas de aprender estándares que permiten el potenciar el aprendizaje de habilidades en el modelado de procesos. **Conclusiones:** se comprobó la capacidad que tienen los entornos de programación diseñados con propósitos educativos para trabajar la robótica en un ambiente cloud computing, reduciendo la brecha que permite pasar de una programación gráfica a una programación textual usando múltiples dispositivos robóticos, en un único escenario de desarrollo.

Palabras clave: robótica educativa, programación visual, STEM, cuarta revolución industrial, inteligencia artificial.

Abstract

Objective: This work seeks to review some positive aspects and characteristics of using robotics in academic environments and to describe some features of programming languages used in robotics implementation. **Methodology:** This work explains how a development environment implemented through cloud computing features can be used to teach robotics at K-12 and college levels. The programming environment combines visual programming using Business Process Management (BPM) notation and text-based programming. **Results:** The obtained results reveal that a gradual transition from visual languages to text-based programming languages is feasible and present the advantages of teaching standards aimed at developing process modeling skills. **Conclusions:** This study demonstrates the potential exhibited by educational programming environments for teaching robotics in a cloud computing environment, thus reducing the gap between visual programming and text-based programming using multiple robotic devices within the same development scenario.

Keywords: Educational Robotics, Visual Programming, STEM, Fourth Industrial Revolution, Artificial Intelligence.

Como citar (IEEE): C. Ruíz-Ramírez., D. Montoya-Quintero., y J. Jiménez-Builes. "Ambiente visual integrado de desarrollo para el aprendizaje de programación en robótica". Investigación e Innovación en Ingenierías, vol. 9, n°1, 7-13, 2021. DOI: https://doi.org/10.17081/invinno.9.1.3957

Introducción

Este artículo se presenta como parte de los resultados de la tesis de maestría "Ambiente de desarrollo integrado de programación híbrida visual/textual integrado con artefactos robóticos para el aprendizaje la enseñanza de áreas STEM a través de programación de computadores", en el cual se desarrolló un entorno de desarrollo considerando las características particulares para Colombia [1].

La inteligencia artificial se desarrolla tempranamente en conjunto al desarrollo de la computación y ya desde 1956, J. McCarthy hacía uso de este término para referirse a la capacidad de las máquinas a tomar decisiones que normalmente estaban asociadas a las capacidades cognitivas humanas. A grandes rasgos, la inteligencia artificial puede describirse como el estudio de la simulación del comportamiento de la inteligencia humana aplicado a dispositivos electrónicos y máquinas [2]. Si bien la inteligencia artificial normalmente es entendida como una subárea de las ciencias de la computación, hay que notar que finalmente la IA es un área multidisciplinar que para su aplicación requiere de múltiples áreas del conocimiento humano, como la lógica, la biología, la electrónica, la física, el estudio de los lenguajes, entre otras.

La inteligencia artificial puede ser aplicada en múltiples contextos como el reconocimiento de imágenes, el procesamiento de datos para el pronóstico de tendencias, gestiones de decisiones de negocio, conducción de autos autónomos, la robótica, pero es de interés, las oportunidades que brinda su aplicación en la educación como son la adaptación de contenidos, flexibilizar las estrategias de aprendizaje, brindar autonomía a los colegios y universidades, entre otros. También ha sido ampliamente usada en la construcción y configuración de diferentes dispositivos robóticos para ser usados en la educación [3] con un sinnúmero de propósitos [4, 5], distinguiendo así dos enfoques distintos robótica para educación y robótica educativa.

Ya a principios del siglo 21, Prensky en sus investigaciones [6, 7] estudia las diversas perspectivas tecnológicas que tienen las nuevas generaciones que crecieron con la tecnología en contraposición con los adultos que se adaptaron al uso de herramientas digitales y a raíz de ello, da cuenta de que hay necesidad de incorporar las tecnologías dentro del ambiente de las aulas de clases. Diversos autores [8, 9] sostienen que las nuevas generaciones tienen una disposición natural y gran facilidad para el uso de las tecnologías de Información y las comunicaciones (TIC). Es conveniente que mediante las TIC se genere interés y fortalecimiento de las áreas STEM (ciencia, tecnología, ingeniería y matemáticas) en los estudiantes.

En las últimas décadas se han propuesto diversas maneras de incorporar las TIC en las aulas de clase. Por ejemplo, a través de cursos en línea (e-learning, b-learning, t-learning, u-learning) o mediante la aplicación de técnicas de la inteligencia artificial fortaleciendo el soporte a la enseñanza y al aprendizaje como los ambientes de tutoría inteligente y la robótica, entre otros.

Una de las primeras experiencias de la inteligencia artificial en educación fue el uso del lenguaje de programación de alto nivel y fácil aprendizaje Logo para la enseñanza de los conceptos básicos de programación a niños y jóvenes. Estas tempranas investigaciones, evidenciaron que la programación de computadores puede ser usada por los profesores para desarrollar nuevas maneras de entender, pensar y aprender [10]. Demostrando además que es importante y necesario el uso de computadores para el aprendizaje de nuevos conocimientos ya que facilitan los procesos de formación de los estudiantes.

En [11] se hace un análisis sobre los desafíos pedagógicos del siglo XXI. En él, se muestra que son necesarios cambios radicales respecto a la manera tradicional de la enseñanza, con el fin de incorporar el componente tecnológico, ya que, a raíz de la tecnología, se han originado cambios importantes en la manera en cómo las personas entienden el mundo, interactúan y aprenden [12].

En [11], también se desarrolla una discusión en torno a trabajos relacionados, en los que se destacan las características constructivistas de la robótica educativa y su acercamiento a la solución de problemas con actividades sociales [13]. De allí se hace un comparativo entre un grupo de robots usados en educación, teniendo en cuenta las características de la modularidad, reusabilidad, versatilidad y precio. De igual modo, se realiza la clasificación entre robots versátiles compuestos por pequeñas piezas que se pueden ensamblar según las necesidades y plataformas no versátiles que ya traen una estructura definida desde la fábrica. Por último, hacen una discusión sobre los mecanismos que permiten controlar las plataformas robóticas con el fin de que éstas puedan realizar tareas programadas previamente. Existen diferentes formas de controlar los robots como por ejemplo el envío de comandos de manera inalámbrica, programación a través de lenguajes visuales, programación a través de lenguajes textuales entre otros.

De acuerdo a las características con las que cuenta el dispositivo robótico, se usan diversos enfoques pedagógicos en las aulas de clase. Los dispositivos robóticos más básicos carecen de capacidad para ejecutar instrucciones programables por el usuario y su uso se limita a realizar tareas restringidas para los cuales fueron diseñados, (p. e. Bristlebot, carros solares, robots cuadrúpedos) [14]. Los dispositivos robóticos básicos regularmente se emplean en los niveles básicos de educación ya que permiten desarrollar a temprana edad, el interés en la robótica misma y en la electrónica. En contraste, en los niveles más avanzados los dispositivos robóticos son programables, modulares y permite ejecutar múltiples actividades programadas al tiempo, con un alto grado de precisión (por ejemplo: Ozobot, Lego EV3, Arduino, Raspberry PI, Bioloid Kit). Dentro de estos dispositivos robóticos avanzado es posible distinguir tres tipos: kits robóticos comerciales (Lego EV3, Bioloid), ordenadores de placa reducida (Arduino, Intel Galileo, Raspberry PI, Asus Tinker Board S, BeagleBoard) y dispositivos construidos a la medida por los investigadores mediante microcontroladores.

En [15] se muestra que se pueden emplear diferentes artefactos utilizados en la robótica de acuerdo a la complejidad deseada y el nivel educativo (por ejemplo: Lego NXT, Vernier, VMCU, HomeLab Kit, entre otros). Para aprender en ambientes de educación semipresencial se propone el uso de HomeLab y DistanceLab en el cual se diseña un contexto que facilita la programación directa y el control remoto de dispositivos empleando un ambiente de programación web. Es bueno contar que, la robótica educativa se ha implementado en diversos ambientes educativos de varios países. En [15] se muestra como ha sido el proceso de adopción en el sistema educativo de Estonia, comenzando desde la educación básica hasta culminar a nivel universitario, empleando metodologías de la educación semipresencial con el uso de microcontroladores y dispositivos electrónicos [16].

En este artículo se expone el diseño y la construcción de una plataforma integrada de desarrollo que mediante un ambiente de programación hibrido (gráfico – textual basado en la computación en nube) que sirve como herramienta de apoyo al aprendizaje de la programación secuencial de robots con aplicación en educación básica, media y superior. El manuscrito está compuesto las siguientes secciones: a continuación, se presentan los materiales y métodos haciendo énfasis en la revisión de los lenguajes de programación empleados en la robótica educativa. Después, en el siguiente capítulo se describen los pasos de la metodología utilizada para la construcción del entorno de programación. Luego, en el capítulo posterior se muestran los resultados obtenidos. Finalmente se exteriorizan las conclusiones y la bibliografía utilizada.

Materiales y métodos: lenguajes de programación utilizados en robótica educativa

Para programar los artefactos robóticos se requiere de lenguajes de programación, especialmente adaptados al hardware específico y por su posible complejidad se emplean en niveles educativos más avanzados [17]. Estos leguajes de programación se pueden clasificar según su método de entrada [18], en lenguajes de programación textuales que consiste en el ingreso de una serie de instrucciones siguiendo una sintaxis definida a través de texto y los leguajes de programación gráficos o visuales que permiten el ingreso de las instrucciones en modo de bloques visuales interconectados [19]. Algunos de los lenguajes gráficos o visuales más representativos son Scratch, LabView y Lego. En lenguajes de programación textuales puede encontrarse C#, JavaScript, Java, C/C++, Python, Pascal, Lisp entre otros.

Dentro de los lenguajes de programación visuales usado en la educación, uno de los más representativos es Scratch [20], el cual se encuentra destinado a niños y jóvenes de edades comprendidas entre los 8 y los 16 años. Este lenguaje, tiene el objetivo de facilitar el aprendizaje autónomo de los conceptos básicos de la programación. Similar a un rompecabezas, en Scratch las rutinas son construidas mediante la unión de bloques que representan instrucciones. El borde de los bloques representa la manera en cómo se pueden unir con otros bloques, facilitando que no existan errores de sintaxis. Scratch carece de funciones definidas por el usuario y no pueden crearse estructuras de datos adicionales que permitan el almacenamiento de información.

Por otra parte, tenemos el entorno de programación que se incluye con los dispositivos Lego EV3 y desarrollado en asocio de Lego y LabVIEW. Es un un lenguaje de rápido aprendizaje, sin embargo, su uso se limita a los estudiantes de niveles más básicos de educación [15]. El entorno EV3, se compone de una serie de cajas interconectables que realizan una tarea específica. Dentro de estos bloques encontramos acciones (bloques verdes), acciones de control de flujo (bloques naranjas), acciones de censado (bloques amarillos), operaciones de datos (bloques rojos), acciones avanzadas (bloques azules) o funciones definidas por el usuario (cian). Cada caja tiene una imagen única que identifica la acción realizada, así como parámetros de configuración.

Finalmente, en los entornos de programación gráficos se encuentra LabVIEW de la empresa National Instruments, el cual es un entorno de desarrollo visual, donde las acciones se realizan mediante la interconexión de bloques parametrizables. Las aplicaciones construidas en LabVIEW pueden poseer una interfaz gráfica de usuario para generar tableros de comandos y con ellos gestionar y monitorizar eventos. Por sus características de permitir la construcción de programas de alta complejidad y versatilidad su uso requiere de experticia técnica [21], sin embargo, su costo limita su adopción fuera de ambientes industriales.

Por otra parta, en la robótica se han usado diversos lenguajes de programación de propósito general, que han sido adaptadas al hardware específico como Java, C, C++, Python [22]. Estos lenguajes de programación generalmente no representan costos ya que son implementaciones de software libre en los diferentes Hardware (Ver Tabla 1).

Tabla 1. Lista de algunos lenguajes de programación empleados en entornos educativos para la enseñanza de la robótica

Lenguaje	Tipo de entrada	Paradigma de programación	Dispositivos robóticos compatibles e Implementación
C	Textual	Imperativa	Lego EV3 (ev3dev)
			Lego NXT (RobotC)
			Arduino
			PIC (PICC)
			Raspberry Pi (GCC)
C++	Textual	Orientada a Objetos	Lego EV3 y NXT (C4Ev3)
			Raspberry Pi (G++)
JAVA	Textual	Orientada a Objetos	Lego EV3 y NXT (Lejos)
			Raspberry Pi (OpenJDK)
Python	Textual	Multiparadigma	Lego EV3 (Ev3dev)
			LegoNXT (PyNXC)
Scratch	Visual	Imperativa / Dirigida por eventos	Lego NXT (Enchanting)
			Raspberry Pi
LabView	Gráfico	Flujo de datos	Lego EV3
			Raspberry Pi
Lego Mindstorn	Gráfico	Flujo de datos	Lego EV3 y NXT
Open Roberta	Gráfico	Imperativa / dirigida por eventos	Lego EV3

Fuente: [1, 14, 20].

A pesar de los múltiples lenguajes de programación que existen en el mercado, aún existen brechas en la transición de un lenguaje visual a un lenguaje de programación visual/grafico que permitan a un estudiante realizar un proceso gradual.

Metodología

Para permitir un entorno que permita la transición gradual entre lenguajes de programación visual a textual, fue propuesta la construcción de un nuevo ambiente de programación visual y sus respectivos componentes. Realizar su construcción siguiendo un modelo en cascada [23]. Las fases de construcción del entorno de desarrollo fueron:

Definición del comportamiento funcional de la plataforma

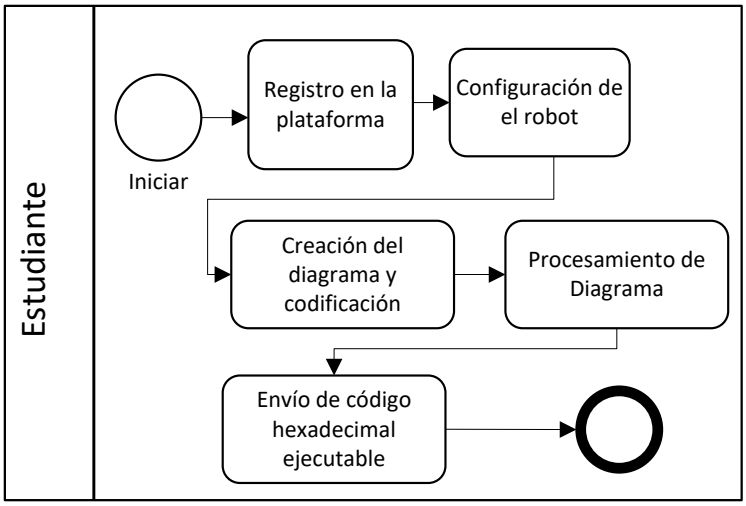
Los estudiantes podrán realizar las siguientes operaciones a través de internet mediante el uso de un navegador web:

- Registrarse con un usuario y contraseña en la plataforma.
- Gestionar los dispositivos robóticos en la plataforma donde se asigna a cada robot un código único y una clave para el acceso del robot.

- Realizar la sincronización de los dispositivos robóticos y el entorno de programación.
- Ejecutar tareas de manera remota en el robot.
- Construir rutinas de programación
- Compilar las rutinas y generar el código máquina ejecutable
- Envío de código ejecutable a los dispositivos robóticos.

El flujo de estas actividades se encuentra en la Figura 1.

Figura. 1. Flujo de trabajo de un estudiante en la plataforma robótica.

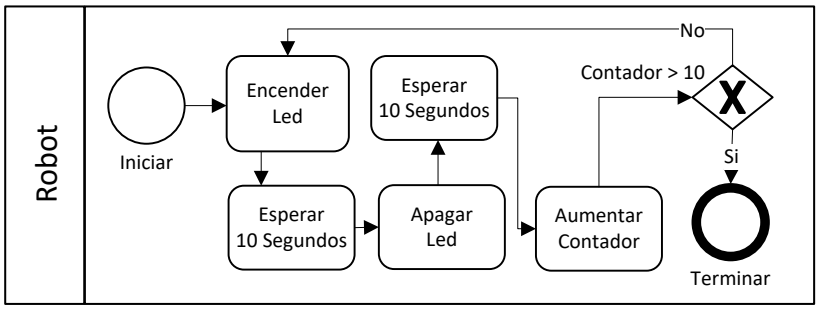


Fuente: Elaboración propia

Para la definición de rutinas, se usó una versión reducida de la notación propuesta para la definición de Modelo y Notación de Procesos de Negocio BPMn [24] que permite representar el flujo de actividades a ejecutar en el robot. Una rutina está compuesta por una serie de cajas redondeadas que representan actividades, piscinas, carriles y estructuras de control como compuertas lógicas, eventos de inicio y finalización interconectadas mediante flechas que representan el orden de ejecución (Ver Figura 2).

Ya que la notación no cuenta con una manera de representar ciclos For o While de la programación imperativa, en caso de requerirse, debe usarse compuertas lógicas del tipo "dada una condición retornan a la tarea de inicio deseada". El uso de esta una notación estandarizada frente a la elaboración de una notación propia, facilita al usuario la familiarización con una notación que en un futuro le permita representar y dar soluciones a procesos en la industria, además de desarrollar el pensamiento procedural y sistemático.

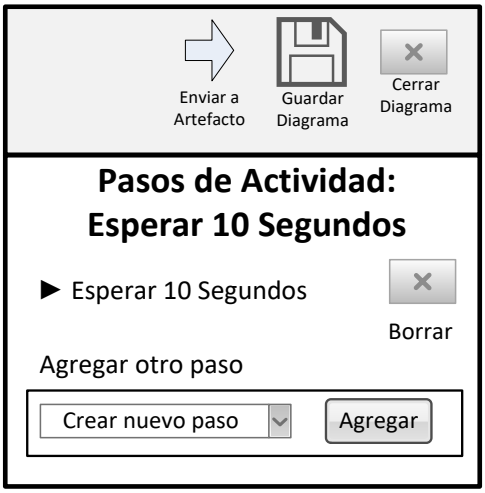
Figura 2. Diagrama para el parpadeo de un LED



Fuente: Elaboración propia

Cada actividad, representada mediante cajas redondeados, es integrada por una serie ordenada de pasos (Ver Figura 3). Un paso representa un fragmento de código textual ejecutable que puede ser interpretado como una acción por el robot. Los pasos pueden ser funcionalidades precargadas o fragmentos de código que el usuario hubiera desarrollado con anterioridad.

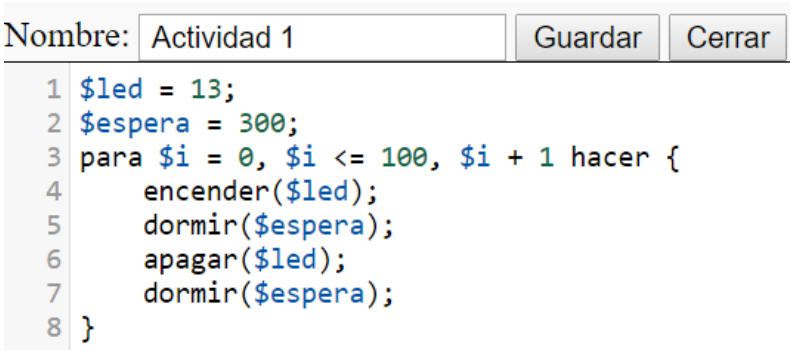
Figura 3. Pasos de una actividad



Fuente: Elaboración propia

Para el código de los pasos, se optó por la elaboración de un lenguaje de programación textual que cuenta con una sintaxis similar al lenguaje PHP. El lenguaje está provisto con instrucciones de control para bucles (while, for) y condicionales (if elseif else), soporta definición de funciones de usuario y declaración de variables para el almacenamiento de datos. (Ver Figura 4).

Figura 4. Ejemplo de código fuente en un paso.



Fuente: Elaboración propia

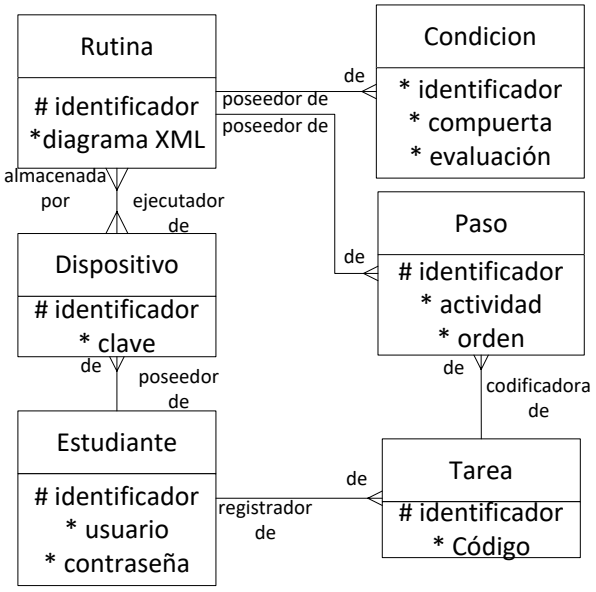
Posterior a la construcción de la rutina y la declaración de código de los pasos, el estudiante puede enviar las rutinas a los robots para ejecutarlas y verificar si se cumple con el objetivo propuesto. En el proceso de envío se hace una compilación de la rutina para convertirla en código máquina.

Arquitectura del ambiente visual de desarrollo integrado

El ambiente de desarrollo fue construido por cuatro capas mediante protocolo HTTP, las capas están compuestas por:

- Frontend: Construida mediante HTML, Javascript y CSS, se encarga de la interacción del estudiante con el entorno de desarrollo. Comprende las interfaces gráficas de la aplicación.
- Backend: Gestiona las peticiones de los servicios web y la ejecución de las tareas. Fue construido siguiendo un patrón modelo, vista, controlador (MVC) [25].
- ORM: Enlace que permite mapear entre objeto y relaciones existentes en una base de datos.
- Base de datos: Almacenamiento y consulta de los datos de manera relacional, se gestiona la información de los estudiantes, dispositivos robóticos y rutinas. (Ver Figura 5).

Figura 5. Modelo de datos del entorno de desarrollo



Fuente: Elaboración propia

Selección de dispositivos para la integración con el ambiente visual de desarrollo

De acuerdo a la disponibilidad del mercado local se seleccionan tres plataformas robóticas (diseño propio basado en microcontrolador, tarjeta de desarrollo y kit comercial de robótica educativa) que se conectarán y harán uso del ambiente visual de desarrollo. Las características necesarias que debe poseer los dispositivos robóticos para la implementación del proyecto son:

Código. Se requiere que el dispositivo sea programable, además que esta programación se pueda realizar de manera inalámbrica, a través de conexión serial mediante USB o mediante algún otro mecanismo.

Almacenamiento y procesamiento. El dispositivo debe tener la capacidad para almacenar en memoria y ejecutar una máquina virtual que sea capaz de interpretar el código que envía el estudiante. Se evalúa el procesador integrado por el dispositivo robótico, memoria RAM, ROM o FLASH y EEPROM

Integración de sensores. Posibilidad de adaptar sensores mediante entradas analógicas y entradas digitales, con el objetivo de conectar dispositivos que puedan enriquecer el proceso de formación como dispositivos para la detección de fuentes de luz, detección de obstáculos mediante infrarrojo o ultrasonido. Se evalúa el número de entradas análogas y digitales.

Nuevos dispositivos. Posibilidad de conexión a otros dispositivos mediante buses de comunicación como I2C o SPI.

Costos. Disponibilidad en el mercado local colombiano y precio comercial por debajo de los $500 USD.

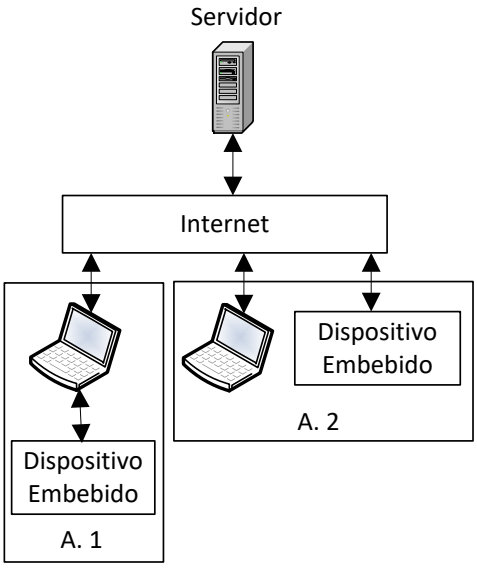
Comunicación Robot – Entorno de desarrollo

Con el objetivo de permitir la trasmisión de datos entré el robot y el entorno de desarrollo, deberá ser tenida en cuenta las capacidades de comunicación de los dispositivos robóticos, con este objetivo, a través de peticiones a servicios REST se definieron los siguientes mecanismos (Ver Figura 6):

Conexión mediante dispositivo intermedio. El dispositivo robótico carece de mecanismos para realizar una conexión directa al servidor donde se encuentra desplegado el entorno de desarrollo, por lo cual se requiere de un dispositivo intermedio con conexión a internet para realizar la transferencia de los datos.

Conexión directa. El dispositivo robótico cuenta con mecanismos para realizar una conexión directa al servidor mediante una interfaz de red propia.

Figura 6. Diagrama de comunicación robot – servidor remoto



Fuente: Elaboración propia

Preprocesamiento y compilación de código

Con el fin de realizar el proceso de compilación de la rutina, se requiere un proceso de preprocesamiento que realice la unión del archivo XML del diagrama de la rutina, la información de los pasos y las compuertas. El proceso de preprocesamiento sigue las siguientes fases (Ver Figura 7):

Creación de arreglo asociativo. Lectura de base de datos y representación del diagrama XML como un arreglo asociativo.

Representación de arreglos. Se generan dos arreglos asociativos en los que se incluye la información de las actividades y las compuertas lógicas existentes en el diagrama de la rutina. Se usa el identificador único del elemento como índice del arreglo asociativo.

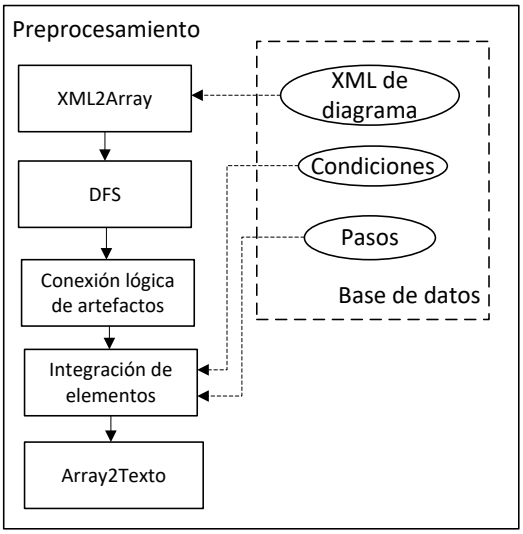
Recorrido del grafo. Se emplea el algoritmo DFS (Depth First Search) para realizar un recorrido de la totalidad del diagrama BPM. La exploración determinará el orden del código en el archivo de texto plano.

Reconocimiento de elementos. Se realiza un recorrido de los elementos extraídos del diagrama y se hallan sus enlaces empleando los índices del arreglo asociativo.

Extracción de condiciones. Se recupera de la base de datos la información sobre los condicionales de las compuertas lógicas y las actividades.

Finalmente, se construye un único archivo de código intermedio que sirve de insumo al compilador.

Figura 7. Proceso de preprocesamiento de rutina



Fuente: Elaboración propia

En el proceso de compilación se traducen las sentencias y las instrucciones al código de operación de la máquina virtual. El compilador cuenta con:

Tabla de definición de constantes. Usado en la traducción de instrucciones a un código numérico.

Analizador léxico. Tokeniza el código fuente recibido como argumento caracterizado por los elementos del lenguaje.

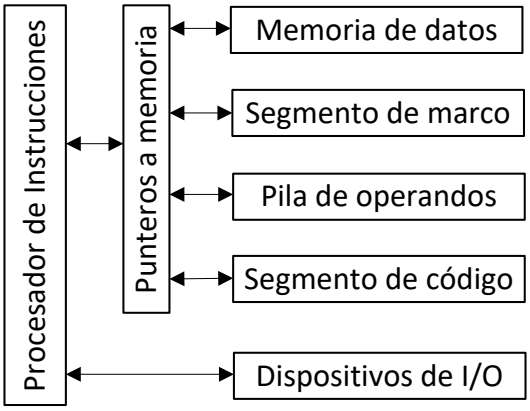
Analizador sintáctico. Comprueba y semantiza la salida del analizador léxico. Definición de las reglas sintácticas del lenguaje y construcción del código máquina.

Tabla de símbolos. Almacena el nombre de las variables, el nombre de las funciones y el ámbito de su declaración.

El código resultante, es descargado por el robot mediante REST. El robot procesa y ejecuta el código máquina mediante una máquina virtual.

Máquina virtual

Figura 8. Arquitectura de la máquina virtual



Fuente: Elaboración propia

La máquina virtual se ejecuta en el dispositivo robótico y está basada en arquitectura Harvard, es decir, los datos y las instrucciones se encuentran en segmentos separados de memoria. Está compuesta por un segmento de código, memoria de datos, una pila de operadores, dispositivos de entrada y salida, segmento de marco, procesador de instrucciones y una serie de punteros. Se encarga de la ejecución del código hexadecimal ejecutable generado en la etapa de compilación. El uso de una máquina virtual permite ejecutar el mismo código ejecutable en los diferentes dispositivos robóticos.

Validación y verificación

Se realiza la validación del tipo caja negra sobre todos los elementos desarrollados con el fin de corregir y determinar posibles errores que posea el programa. Se realizaron pruebas de usabilidad en la cual a cinco usuarios se les definirán un conjunto de tareas a realizar mientras sus acciones con el mouse son registradas y cronometradas. Se concluirá con una encuesta para determinar el grado de usabilidad desde la perspectiva del usuario.

Resultados y discusión

Selección de alternativas de plataformas robóticas

Para la realización del proyecto se optó por el usó de un dispositivo robótico por cada una de las categorías definidas: kit para robótica educativa, computador de placa reducida y placa de diseño propio (Ver Tabla 2).

Tabla 2 Comparación de características de computador de placa reducida

	Raspberry PI 2	Arduino Uno R3	BeagleBone Black
Velocidad de reloj	Quad-core ARM a 900MHz	20 MHz	1GHz ARM Crotes A8
Flash	32GB Tarjeta SD	32KB	4GB on-board flash storage
EEPROM	0 B	1024 B	0 B
Ram	1024 MB	2048 B	512 MB
Puertos de I/O	26	19	63
Canales PWM	1	6	8
Canales Análogos	0	6	7
Resolución puertos análogos	No aplica	10-bits	12-bits
Comunicación	Ethernet, USB, I2C, SPI	SPI, Serial USART, I2C	UART, PWM, LCD, GPMC, MMC1, SPI, I2C, CAN BUS
Precio	35 USD	23 USD	50 USD
Modulo USB	Integrado	Externo	Integrado

Fuente: [1]

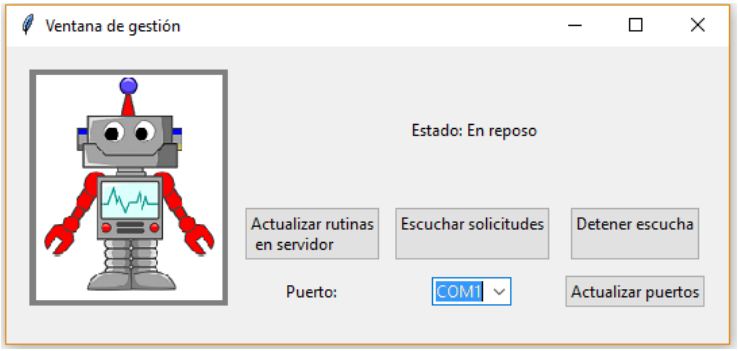
Para la elección del computador de placa reducida, se revisaron las características del Arduino UNO R3, Raspberry PI 2 y BeagleBone Black. Luego de evaluar las alternativas, se opta por el uso del Raspberry PI por las capacidades de cómputo suficientes, amplio número de puertos de entrada y salida, posibilidad de conexión directa a internet, almacenamiento interno, su amplia comunidad y su disponibilidad a nivel mundial. Para la tarjeta de desarrollo se usó un microcontrolador PIC 18F4550 que se había construido con anterioridad al proyecto.

Para el kit robótico educativo se evaluó el uso del Lego Minstorm EV3 y Bioloid Premium Kit. Se seleccionó Lego Minstorm EV3 ya que el Bioloid Premium Kit se encuentra por fuera del rango del precio límite fijado.

Implementación de la plataforma

El entorno de desarrollo fue construido mediante el uso de un stack LAMP (Linux, Apache, MySQL, PHP), para el cual se diseño un framework basado en la arquitectura modelo vista controlador y filosofía don›t repeat yourself (DRY). Con el fin de facilitar el reúso del preprocesador y el compilador, se desarrollaron como librerías externas al framework. Para cada una de las plataformas robóticas se construyeron máquinas virtuales que se adaptaran a las interfaces de comunicación y las capacidades. Para la tarjeta desarrollo propio basada en microcontrolador se emplea el software CCS C Compiler y se construyó una interfaz de comunicación en MS-Windows usando Python (Ver Figura 9) que sirva de puente de comunicación entre el microcontrolador y el sistema remoto. Para el dispositivo Lego EV3 y Raspberry PI se construyó la máquina virtual y la interfaz de comunicación sobre Python.

Figura 9. Interfaz de comunicación en Windows para comunicación con microcontrolador



Fuente: Elaboración propia

Experiencia de uso sobre la interfaz de programación

Con el objetivo de evaluar posibles debilidades existentes, sobre una población de ciento cinco personas conformada por profesionales del área de informática, se les solicitó la realización de cuatro tareas sobre la plataforma, que incluía el registro de usuario (1), adición de un dispositivo robótico (2), creación de una rutina (3) y la adición de dos pasos en una actividad (4). Se cronometró el tiempo para la realización de las actividades y se realizaron tres preguntas posteriores a la ejecución. En promedio, en la realización de cada tarea se tardaron (1) 02 minutos 57 segundos, (2) 01 minuto 31 segundos, (3) 36 segundos y (4) 3 minutos 11 segundos respectivamente (Ver Tabla 3).

Tabla 3. Resultado de la encuesta

Pregunta	Si	No	Tal vez
¿Presenta el sistema facilidad de uso?	97	2	6
¿Presenta el sistema facilidad de aprendizaje?	94	5	6
¿Recomendaría el aplicativo a otro usuario?	95	2	8

Fuente: Elaboración propia

Conclusiones

Se evidencia desde décadas atrás que, dada las características de los estudiantes actuales, es necesario y conveniente buscar mecanismos que permitan integrar la tecnología en los ambientes educativos con el fin de flexibilizar y facilitar los procesos de aprendizaje. La aplicación de la robótica educativa, enmarcada en prácticas constructivistas y como una didáctica en las aulas de clase presenta permite al estudiante resolver un problema mediante la creación de su conocimiento a través de un proceso dinámico, participativo e interactivo.

Existen en el mercado diversos kits robóticos que incluyen los ambientes de programación. Estos kits son diseñados en países con un alto desarrollo económico. Sin embargo, sus costos limitan la adopción en modelo educativo colombiano. Adicionalmente sus prácticas no están completamente acordes con las necesidades de los estudiantes colombianos.

En este artículo se recapitularon algunas de las ventajas sobre la aplicación de robótica educativa. También se muestran algunas características de los lenguajes de programación usados en la robótica educativa. Se presentó adicionalmente un entorno de programación basado en cloud computing diseñado para la aplicación de robótica en ambientes educativos que busca reducir las brechas al permitir a los estudiantes pasar fácilmente de una programación visual a una programación textual usando múltiples dispositivos robóticos en un único entorno de desarrollo. Después de analizar los paradigmas predominantes en la construcción de representaciones computacionales del entorno, se optó por emplear la técnica de mapeo por celdas de ocupación. modelos matemáticos del robot, validándose de esta forma el potencial de los métodos intuitivos en el diseño de controladores.

El sistema fue puesto a prueba en diferentes tipos de entornos interiores como fueron pasillos, oficinas, laboratorios y aulas de clase. Los modelos obtenidos se ajustaron tanto a las características morfológicas de los entornos de prueba, como a sus características métricas.

Referencias bibliográficas

1. C. A. Ruiz, “Ambiente de desarrollo integrado de programación híbrida visual/textual integrado con artefactos robóticos para el aprendizaje la enseñanza de áreas STEM a través de programación de computadores”. Tesis de Maestría en Ingeniería, Universidad Nacional de Colombia, 2019.
2. G. A. Wiggins y A. Smaill, Musical Knowledge: what can Artificial Intelligence bring to the musicians? 2000.
3. J. Prentzas, “Artificial intelligence methods in early childhood education”, en Artificial Intelligence, Evolutionary Computing and Metaheuristics, *Springer, 2013*, pp. 169–199. DOI: https://doi.org/10.1007/978-3-642-29694-9_8
4. Y.-W. Cheng, P.-C. Sun, y N.-S. Chen, “The essential applications of educational robot: Requirement analysis from the perspectives of experts, researchers and instructors”, *Comput. Educ.*, vol. 126, pp. 399-416, nov. 2018. DOI: https://doi.org/10.1016/j.compedu.2018.07.020.
5. C. Angeli y N. Valanides, “Developing young children’s computational thinking with educational robotics: An interaction effect between gender and scaffolding strategy”, *Comput. Hum. Behav.*, vol. 105, p. 105954, abr. 2020. DOI: https://doi.org/10.1016/j.chb.2019.03.018.
6. M. Prensky, “Digital natives, digital immigrants part 1”, *Horiz.*, vol. 9, no 5, pp. 1–6, 2001. DOI: https://doi.org/10.1108/10748120110424816.
7. M. Prensky y B. D. Berry, “Do they really think differently”, *Horiz.*, vol. 9, no 6, pp. 1–9, 2001. DOI: https://doi.org/10.1108/10748120110424843
8. B. De Wever, P. Mechant, P. Veevaete, y L. Hauttekeete, “E-Learning 2.0: social software for educational use”, *Multimedia Workshops 2007 ISMW’07. Ninth IEEE International Symposium on*, 2007, pp. 511–516. DOI: https://doi.org/10.1109/ISM.Workshops.2007.91
9. F. Fovet, “Impact of the use of Facebook amongst students of high school age with Social, Emotional and Behavioural Difficulties (SEBD)”, *2009 39th IEEE Frontiers in Education Conference, 2009*, pp. 1–6. DOI: https://doi.org/10.1109/FIE.2009.5350786
10. Papert, Mindstorms: Children, Computers, and Powerful Ideas, Edición: New Ed. New York: The Perseus Books Group, 1993. DOI: https://doi.org/10.1145/1045071.1045074

11. M. Ruzzenente, M. Koo, K. Nielsen, L. Grespan, y P. Fiorini, "A review of robotics kits for tertiary education", *Proceedings of International Workshop Teaching Robotics Teaching with Robotics: Integrating Robotics in School Curriculum*, 2012, pp. 153–162. DOI: https://doi.org/10.1007/s10798-012-9210-z

12. A. Larcher., F. Turri, J. Collins, I. Derweesh, A. Volpe, J. Kaouk, R. Koon, "Definition of a structured training curriculum for robot-assisted partial nephrectomy: A Delphi-consensus study from the ERUS Educational Board", *Eur. Urol*. Suppl., vol. 17, n.o 2, pp. e678-e682, mar. 2018. DOI: https://doi.org/10.1016/S1569-9056(18)31310-1.

13. N. Reich-Stiebert, F. Eyssel, y C. Hohnemann, "Involve the user! Changing attitudes toward robots by user participation in a robot prototyping process", Comput. Hum. Behav., vol. 91, pp. 290-296, feb. 2019. DOI: https://doi.org/10.1016/j.chb.2018.09.041.

14. B. S. Blais, "Using Python to Program LEGO Mindstorms Robots: The PyNXC Project", *Python Pap*., 2010.

15. S. Seiler, R. Sell, y D. Ptasik, "Embedded System and Robotic Education in a Blended Learning Environment Utilizing Remote and Virtual Labs in the Cloud, Accompanied by 'Robotic HomeLab Kit'", *Int. J. Emerg. Technol. Learn. IJET,* vol. 7, no 4, dic. 2012. DOI: https://doi.org/10.3991/ijet.v7i4.2245

16. X. Xie, C.-C. Huang, Y. Chen, y F. Hao, "Intelligent robots and rural children", *Child. Youth Serv*. Rev., vol. 100, pp. 283-290, may 2019. DOI: https://doi.org/10.1016/j.childyouth.2019.03.004.

17. C. Fernández-Llamas, M. A. Conde, F. J. Rodríguez-Lera, F. J. Rodríguez-Sedano, y F. García, "May I teach you? Students' behavior when lectured by robotic vs. human teachers", *Comput. Hum. Behav.*, vol. 80, pp. 460-469, mar. 2018. DOI: https://doi.org/10.1016/j.chb.2017.09.028.

18. M. Campusano, J. Fabry, y A. Bergel, "Live programming in practice: A controlled experiment on state machines for robotic behaviors", *Inf. Softw. Technol*., vol. 108, pp. 99-114, abr. 2019. DOI: https://doi.org/10.1016/j.infsof.2018.12.008.

19. B. Jost, M. Ketterl, R. Budde, y T. Leimbach, "Graphical Programming Environments for Educational Robots: Open Roberta - Yet Another One?", *2014 IEEE International Symposium on Multimedia*, 2014, pp. 381–386. DOI: https://doi.org/10.1109/ISM.2014.24

20. J. Maloney, M. Resnick, N. Rusk, B. Silverman, y E. Eastmond, "The Scratch Programming Language and Environment", *ACM Trans. Comput. Educ*., vol. 10, no 4, pp. 1–15, nov. 2010. DOI: https://doi.org/10.1145/1868358.1868363.

21. J. M. Gomez-de-Gabriel, A. Mandow, J. Fernandez-Lozano, y A. J. Garcia-Cerezo, "Using LEGO NXT Mobile Robots With LabVIEW for Undergraduate Courses on Mechatronics", *IEEE Trans. Educ*., vol. 54, no 1, pp. 41–47, feb. 2011. DOI: https://doi.org/10.1109/TE.2010.2043359

22. R. U. Pedersen, J. Nørbjerg, y M. P. Scholz, "Embedded Programming Education with Lego Mindstorms NXT Using Java (leJOS), Eclipse (XPairtise), and Python (PyMite)", *Proceedings of the 2009 Workshop on Embedded Systems Education, New York, NY, USA*, 2009, pp. 50–55. DOI: https://doi.org/10.1145/1719010.1719019

23. W. W. Royce y others, "Managing the development of large software systems", en proceedings of IEEE WESCON, 1970, vol. 26, pp. 1–9. DOI: https://doi.org/10.4236/cus.2018.62015 1,006

24. S. A. White, "Introduction to BPMN", *IBM Coop*., vol. 2, no 0, p. 0, 2004.

25. G. E. Krasner y S. T. Pope, "A Cookbook for Using the Model-view Controller User Interface Paradigm in Smalltalk-80", *J Object Oriented Program*, vol. 1, no 3, pp. 26–49, ago. 1988.