Aplicación del aprendizaje basado en problemas y la tecnología informática a la enseñanza de programación en los primeros años de ingeniería

Ricardo Coppo[1], Javier Iparraguirre[1], Germán Feres[1], Gustavo Ursua[1], and Ana Cavallo[2]

[1] Universidad Tecnológica Nacional - Facultad Regional Bahía Blanca
[2] Instituto Superior Juan XXIII - Bahía Blanca
Mail de contacto: rcoppo@frbb.utn.edu.ar

Resumen La enseñanza de la programación de computadoras es una materia básica de las carreras relacionadas con las ciencias de la computación y de la ingenierías electrónica y de sistemas. Algunos estudios sugieren la necesidad de incrementar los factores motivacionales causados por la resolución exitosa de problemas relacionados con la carrera universitaria elegida, y que aumenten el entusiasmo, autoestima y perseverancia del alumno. Se evalúa la inserción en la dinámica del curso de programación tradicional un elemento didáctico, basado en una placa electrónica microcontroladora de licencia open source, a la que se le ha adicionado desarrollos de hardware propios realizados por la cátedra. Se trabajó con una metodología didáctica basada en la resolución de problemas (PBL) con el fin de observar su incidencia en la calidad del aprendizaje y la generación de motivación intrínseca por parte de los alumnos. Las primeras experiencias muestran un fuerte incremento en el desempeño de los alumnos y mejores resultados en los proyectos finales de cátedra.

Palabras clave: enseñanza de la programación, hardware didáctico, motivación.

1. Introducción

La mayoría de los educadores y docentes de programación coinciden en señalar que son pocos los estudiantes que afirman que aprender a programar una computadora es una tarea sencilla. El problema se agrava debido a que la mayoría de los cursos de programación se encuentran en los currículos de los primeros años de las carreras de

Ingeniería Electrónica, años que significan para la mayoría de los estudiantes transiciones importantes en sus hábitos de estudio, de vida, y comportamiento social [4,6].

La literatura especializada enfoca los avances de la enseñanza de la programación con el uso de tecnologías nóveles (proyecciones, multimedios, y otros) que pueden ser aplicadas a la didáctica del docente y en la organización de sus clases [7,8]. Menos atención ha sido dedicada a los factores cognitivos presentes en el proceso de aprendizaje. El estilo de aprendizaje y la motivación se destacan como los factores cognitivos que afectan en mayor medida el desempeño del alumno.

La motivación, que por definición es "la dirección y magnitud del comportamiento humano", tiene como factores indispensables la elección del curso de acción, la persistencia en el mismo y el esfuerzo que el alumno decidirá realizar [1,5].

Una vez decidido el curso de acción el factor emocional/afectivo tiene un rol preponderante. El alumno se relaciona no solo cognitivamente con su proyecto sino también en forma afectiva; característica que es fundamental para incrementar el tiempo y el esfuerzo que luego dedicará a la conclusión del mismo.

Tradicionalmente la motivación ha sido considerada bajo una perspectiva individualista. Sin embargo, como toda acción humana se realiza en un contexto, este ejerce una influencia indiscutible en todo el proceso de aprendizaje. Además, las exigencias de la sociedad moderna priorizan las actitudes colaborativas y la interacción entre pares [1].

En una primera evaluación, la mayoría de los docentes de informática, presienten que el aprendizaje de la programación en estos primeros años obedece más a una motivación extrínseca, como aprobar la materia con altas calificaciones o lucir como intelectual entre sus pares. Sin embargo, los alumnos que demuestran un verdadero interés personal en adquirir la capacidad de aprender a programar correctamente (motivación intrínseca) son los que luego logran los mejores resultados.

Mediante la aplicación de las metodologías de aprendizaje basado en problemas PBL *(Problem Based Learning)* [1,2] se intenta crear un contexto interactivo en el que al trabajar sobre la resolución de problemas individuales, se incrementa la autoestima y la coopera-

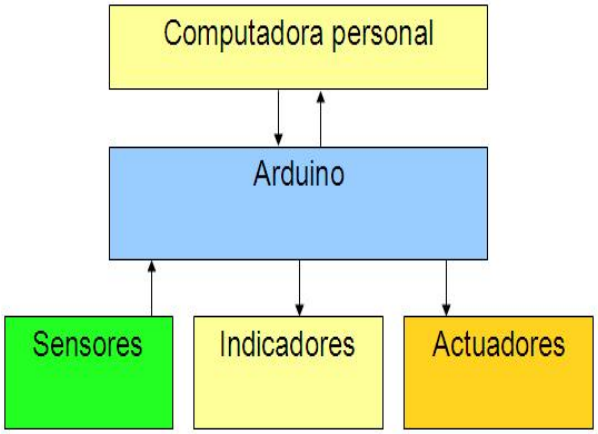

Figura 1. Diagrama en Bloques

ción entre los alumnos, integrando conocimientos adquiridos en otras áreas o materias.

Visto esto, este trabajo intenta explorar el uso de un sistema de hardware y software que introducen fuertes factores de motivación intrínseca en los alumnos de ingeniería electrónica, generándoles verdaderos deseos de poder desarrollar programas de software y palpar los resultados en equipos electrónicos reales.

Esto último además transfiere al estudiante un elemento de distinción entre sus pares (especialmente con los alumnos de otras carreras) que fortalece su autoestima y su decisión en la elección de la carrera elegida.

2. Materiales y metodología

2.1. Descripción de sistema

El sistema didáctico utilizado se divide en los siguientes subsistemas: un software de desarrollo que se ejecuta sobre una laptop o PC estándar y una placa de hardware Arduino interconectada con la computadora y diversos sensores o actuadores desarrollados por la cátedra que le permiten a la placa interactuar con el entorno o medio físico. (Figuras 1 y 2).

Figura 2. Sistema completo

2.2. Características del hardware

El elemento principal del hardware del sistema consta de una plataforma open-source denominada Arduino UNO [9], basada en un microcontrolador que dispone de 14 entradas/salidas digitales, 6 entradas analógicas, una interfase USB y, 32 KBytes de memoria para el programa.

La gran cantidad de entradas y salidas permite construir pequeños experimentos con sensores básicos de temperatura, LEDs, y displays de bajo consumo sin tener que recurrir a conexionados complejos. (Figura 3). La placa se monta en un soporte de acrílico para incentivar al alumno a preguntar sobre sus componentes y el funcionamiento del mismo. Las conexiones externas aceleran la configuración del sistema para diferentes ejercicios. (Figura 4).

El hardware open-source es aquel cuyo diseño y esquemas circuitales se publican, por ejemplo a través de la Internet, para que cualquier persona o empresa pueda fabricarlo o reproducirlo sin pagar licencias. Este modelo comercial se encuentra sustentado por el concepto del software open-source propulsado por los programadores del sistema operativo GNU/Linux.

La libre disponibilidad de la información de diseño permite a diversos proveedores de hardware la creación y venta de kits de compo-

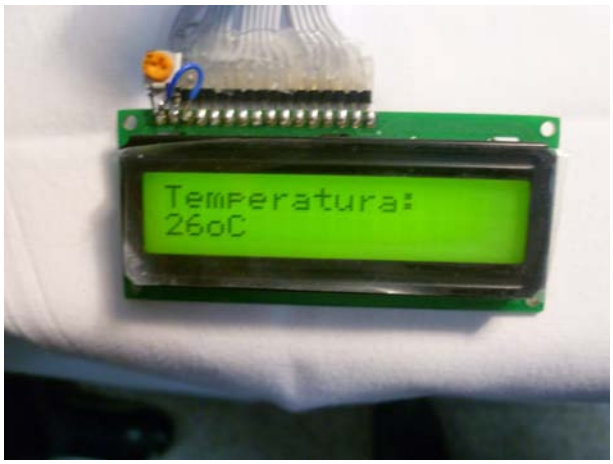

Figura 3. Control de temperatura

nentes para su armado, y posibilitan al alumno entusiasta construir su propia plataforma de experimentación a bajo costo.

Gracias a la característica open-source del hardware los docentes de la cátedra pudieron extender las facilidades básicas del sistema con desarrollos propios. La primera, una hilera de LEDS (1 sola dimensión espacial) permiten desarrollar aplicaciones como el "auto fantástico" o secuenciadores de luces como las que se instalan sobre patrullas policiales y de ambulancias. Una segunda extensión basado en una matriz de LEDS de dos dimensiones permite explorar problemas relacionados con ciclos y "recorridas" de matriz además de presentar conceptos iniciales sobre el funcionamiento de un display o monitor digital. (Figuras 5 y 6).

2.3. El ambiente de desarrollo

La programación de la placa se realiza por medio de un entorno de desarrollo integrado (IDE) de distribución gratuita que permite al usuario escribir aplicaciones en C que es uno de los objetivos principales del programa sintético de la materia. Una vez compiladas estas últimas, son transmitidas a la placa por medio del puerto USB a través del IDE. Es de destacar la importancia que el hardware reconoce esencialmente el mismo lenguaje de programación que los alumnos estudian en las clases teóricas.

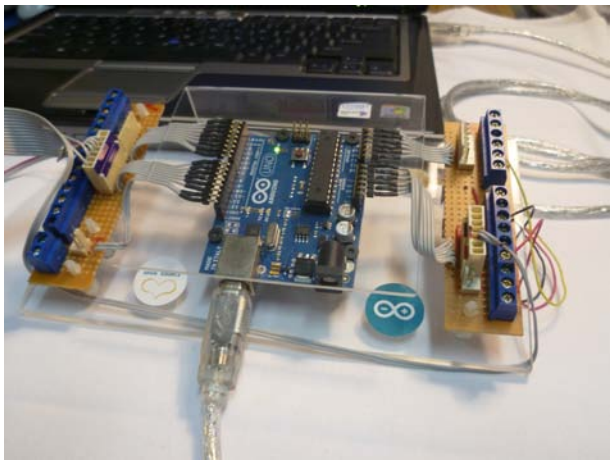

Figura 4. Placa y conexion a la PC

3. Aplicación educativa

En las primeras clases de la materia el alumno aprende sobre la arquitectura de las computadoras y los pasos necesarios para producir un programa ejecutable. Los primeros ejercicios se desarrollan con entrada/salida en modo terminal, es decir sobre la antigua pantalla del DOS histórico. Conceptos básicos como el sistema binario, la noción de algoritmo, y las primeras instrucciones de bifurcación y ciclo en el lenguaje C resultan abstractos y difíciles de entender para la mayoría de los estudiantes no motivados.

Un ejercicio clásico, realizado en la mayoría de las cátedras de programación para fijar estos conceptos, es la realización de un programa en el que se hace la conversión de números del sistema decimal al sistema binario (base 2) con salida a pantalla de línea de comando en modo terminal. El ejercicio, de nivel conceptual interesante para aprender a programar, no resulta motivador para el estudiante porque el alumno percibe su salida como algo antiguo o de poca utilidad.

Para realizar este mismo ejercicio sobre la placa de hardware solo bastan conectar algunos diodos emisores de luz (LED) a las salidas correspondientes y modificar ligeramente el programa originalmente destinado a la pantalla DOS para que opere con los LEDS.

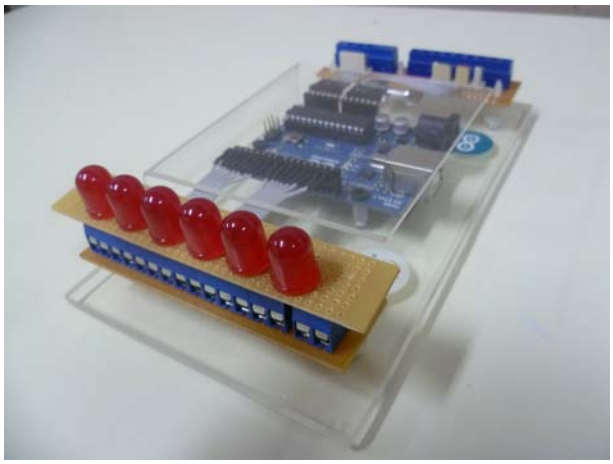

Figura 5. Hilera de LEDS - Problemas 1D

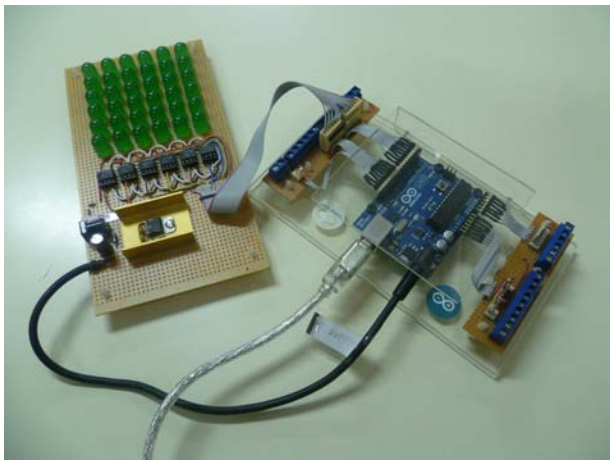

Figura 6. Matriz de LEDS - Problemas 2D

Frecuentemente el primer intento del programa no funciona correctamente (luces fuera de secuencia, lógica errónea, entre otras fallas), pero a diferencia del ejercicio equivalente en pantalla, el alumno se motiva más, quiere ver su código funcionando, muestra mayor tezón en la búsqueda de la solución, y se retira del laboratorio con mayor satisfacción personal por haber cumplido el objetivo.

Cabe señalar que la configuración presentada se adapta a ejercicios más complicados y a metodologías didácticas diferentes. Por

ejemplo, fácilmente la configuración presentada se puede adaptar a la enseñanza de autómatas regulares simulando sistemas de alarmas, control de semáforos, electrodomésticos inteligentes, por mencionar solo algunas ideas. Aunque estos temas se estudian en mayor profundidad en materias más avanzadas en la carrera, percibir como es su funcionamiento despierta mucho interés en el alumno que recién inicia y además le permite visualizar el tipo de sistema que podrá diseñar en un futuro cercano.

4. Resultados

El sistema propuesto se puso en marcha en forma experimental en la cátedra Informática I de la Universidad Tecnológica Nacional - Facultad Regional Bahía Blanca durante los ciclos lectivos 2011 y 2012. En el primer año se disponía de un solo equipo al que se le aplicó la técnica "open shop batch" en que cada par de alumnos disponía de 20 minutos para resolver un problema específico. La respuesta obtenida alentó la publicación de los primeros resultados y a iniciar acciones para la adquisición de mayor cantidad de equipos.

En el año 2012 se efectuaron más laboratorios con el nuevo hardware de soporte y con mayor cantidad de sensores y LEDS que permitieron plantear problemas de mayor complejidad. La aparición comercial a finales de dicho año de equipos similares de aún menor costo que el original produjo dos resultados sorprendentes. La primera es la donación por parte de los alumnos de 10 equipos para la cátedra, indicando que aunque no eran para su provecho directo, advertían su importancia para los futuros alumnos de la materia en años posteriores. El segundo fue la adquisición de mas de 30 equipos adicionales para uso personal de los alumnos a quienes se les despertó el interés por desarrollar sus propios proyectos.

También se observó una sensible mejora en los trabajos de fin de materia en los años 2011, año en que uno de los trabajos recibió el importante segundo premio en el concurso estudiantil de las jornadas JAIIO 40 - Jornadas Argentinas de Informática e Investigación Operativa desarrollado en la ciudad de La Plata [10]. La asistencia al congreso y su participación fue un factor emotivo y motivante para el asistente. Otros proyectos presentaron ese año incluyeron juegos

electrónicos combinando desarrollos incipientes en hardware y software y manejo de bases de datos simples entre otros.

En el grupo de los alumnos del año 2012 los proyectos finales de software presentan una complejidad que triplica, evaluada en cantidad de líneas de código, la producción de años anteriores.

5. Conclusiones

Las primeras experiencias con la metodología presentada han sido muy positivas. La participación y motivación de los alumnos al ver sus proyectos convertidos en componentes y diseños reales incentiva la creatividad y búsqueda de proyectos de mayor envergadura. El uso de un hardware comercialmente económico y con gran difusión en Internet permite al alumno despertar hábitos de investigación y búsqueda de problemas que (aunque resueltos) pueden ser implementados con bajo costo. Un proyecto exitoso deriva en otro y así sucesivamente; crece la autoestima y brinda elementos de valoración entre sus pares.

En nuestro diseño prevaleció la idea de experimentar con diversos sensores y medios de salida, en un esfuerzo para determinar experimentos motivadores y significativos para la enseñanza de la programación. En una versión final, más apta para cursos numerosos, se debería definir una configuración básica de sensores y construir todo el sistema físicamente en una sola unidad. Eventualmente se podría extender el sistema básico a través de módulos independientes para periféricos más complejos o de mayor potencia. El sistema propuesto debería ser acompañado por dos documentos con diferentes tipos de ejercicios propuestos, uno para el alumno y otro para el profesor.

Referencias

1. Dörnyei, Z., *Teaching and Researching Motivation*, Pearson Education, Malaysia, 2001.
2. O'Kelly J., Gibson J.P., *Robocode & Problem Based Learning: A non-prescriptive approach to teaching programming*, In ITICSE '06: Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education, 2006.
3. Lin H.T., Kuo T.H., *Teaching programming technique with edutainment robot construction*, IEEE 2nd International Conference on Education Technology and Computer (ICETC), Shanghai, June 2010.

4. Huet I., Pacheco O.R., Tavares J., Weir G., *New Challenges in Teaching Introductory Programming Courses: a Case Study*, 34th ASEE/IEEE Frontiers in Education Conference, Savannah, Georgia, October, 2004.
5. Jenkins T., *Teaching Programming - A Journey from Teacher to Motivator*, 2nd Annual LTSN-ICS Conference, London, 2001.
6. Jenkins T., *On the Difficulty of Learning to Program*, 3rd Annual LTSN-ICS Conference, Loughborough University, 2002.
7. Matsumura K., Daisuke S., Aiguo HE, *A C Language Programming Education Support System base on Software Visualization*, IEEE Joint Conferences on Pervasive Computing (JCPC), Tamsui, Taipei, December 2009.
8. Shyu Y.H., Chen P.W., *PLL: A Programming Languages Lab System*, 21st International Conference on Distributed Computing Systems Workshops (ICDCSW01), Mesa, Arizona, USA, 2001.
9. Arduino Team, *Arduino Home Page*, http://www.arduino.cc, visitado: Marzo 2013.
10. Diaz, A., Coppo R.J., *Calculadora geoespacial*, Jornadas Argentinas de Informática e Investigación Operativa, JAIIO 40, Segundo Premio Concurso Estudiantil, La Plata, Argentina, 2012.