Proyecto ICARO

Robótica pedagógica con software y hardware libre

Valentin Basel - Analista en sistemas informaticos

CIECS - UNC - CONICET

Resumen La robótica pedagógica busca incentivar el desarrollo cognitivo del alumno mediante la fabricación y programación de robots o mecanismos de automatización sencillos. El uso de tecnologías libres (software y hardware de especificaciones libres) permite adaptar y modificar los desarrollos a las necesidades concretas del aula, así como permitir abaratar costos posibilitando el uso de componentes reciclados. El proyecto ICARO, busca simplificar el complejo contenido técnico inherente al desarrollo de un robot , facilitando el acceso y la apropiación de la tecnología por parte de los colegios y los alumnos.

1. Introducción

La robótica pedagógica, dado su carácter polivalente y multidisciplinario, permite el abordaje de conocimientos variados como la electrónica, informática, física y matemática mediante la construcción de un juguete-objeto como puede ser un robot. El desarrollo de estos juguetes-objetos implica una experiencia que contribuye a expander la creatividad y el pensamiento reflexivo y científico de los alumnos (en relación a la formulación de hipótesis, la experimentación y la elaboración de conclusiones), los cuales al enfrentarse a un "problema" dado, aprenden a experimentar, diseñar y resolver situaciones de carácter constructivista. En el proceso de "pensar el robot" , se generan las condiciones de apropiación del conocimiento por parte del alumno. Se trata de otorgar a los alumnos un rol activo en sus aprendizajes, colocándolos como diseñadores de sus propios proyectos y constructores de conocimientos. El uso de software libre en el ámbito escolar, permite tener control sobre las características del mismo, permitiendo adaptarlo a las necesidades concretas del ámbito escolar y las realidades socio-económicas de la institución, es neutro frente a fabricantes (el alumno no es un "potencial cliente") y todo el material usado puede ponerse a disposición de otros docentes.

2. Hardware de especificaciones libres

Cuando nos referimos a Hardware de especificaciones libres, hablamos de todos los dispositivos electrónicos cuyos diagramas, especificaciones, planos y código fuente están abiertos y accesibles al público. Si bien el concepto de software libre no se puede aplicar linealmente al hardware por su naturaleza diferente, existe un consenso con respecto a que sus especificaciones de construcción estén bajo una licencia de tipo libre (creative commons, GPL) que permita su estudio y eventual copia. En ese sentido

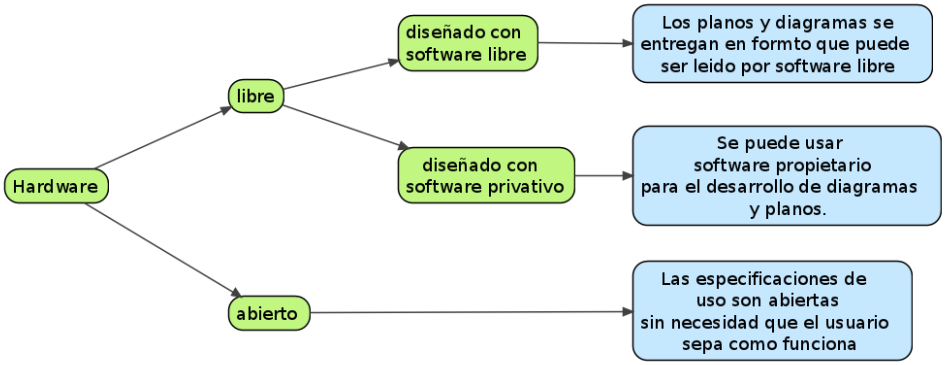
la diferencia más importante se da con respecto a lo que se considera hardware abierto, donde no importa poder replicar el hardware, si no tener la suficiente cantidad de información para poder interactuar con el dispositivo.

Ventajas

1. Protege y defiende la soberanía, permitiendo a las naciones no depender de ninguna otra que le provea los recursos necesarios para su desarrollo e independencia tecnológica.
2. Fomenta a que el hardware pueda ser de calidad, los estándares abiertos y que sean más económicos.
3. La re utilización y la adaptación de diseños (corés) permitiendo así innovar y mejorar los diseños de forma colaborativa a nivel mundial.
4. Ayudaría a las compañías a ahorrar costes y tiempos de diseño en sus trabajos.
5. Existen comunidades de diseño, programación, pruebas, y soporte que día a día crecen de forma dinámica y participativa.
6. Evita la alianza *trusted computing* y la gestión de derechos digitales (DRM), que imponen restricciones a los dispositivos electrónicos como por ejemplo electrodomésticos, computadoras, entre otras más.

Hablamos de robótica pedagógica libre porque tanto el software como las especificaciones completas del hardware (como el software con el que fueron escritas y los estándares usados) tienen que tener una licencia que permita tener las cuatro libertades atribuidas al software libre, posibilitando su modificación, adaptación y estudio por parte de los alumnos y docentes. La ventaja de contar con sistemas abiertos para el estudio, es la posibilidad de ser escalable, es decir que los usuarios no están circunscriptos a las restricciones del fabricante y las modificaciones y experimentaciones pueden ser devueltas a la comunidad. Se logra de esta forma una interacción entre las universidades que investigan sobre las tecnologías y procesos pedagógicos, los colegios técnicos/secundarios que aplican y desarrollan robots educativos, y hasta instituciones de nivel primario que pueden usar los desarrollos para hacer practicas. La propuesta de usar hardware de especificaciones libres, ayuda a reducir la brecha tecnológica que se genera entre las instituciones privadas (con mayor capacidad adquisitiva) y los colegios públicos, los cuales generalmente tienen menor presupuesto para trabajar con robótica.

Figura 1. diagrama de definiciones de hardware libre



3. Aprendizaje basado en problemas

La corriente del Aprendizaje Basado en Problemas (ABP) presenta un método aplicable al trabajo de grupos de pocos integrantes. En estas actividades grupales los alumnos asumen responsabilidades y realizan acciones que son básicas en su proceso formativo; es usado principalmente en educación superior (Rhem, 1998; Jiménez, 2006). El ABP es una metodología centrada en el aprendizaje, en la investigación y reflexión que siguen los alumnos para llegar a una solución ante un problema planteado por el profesor.

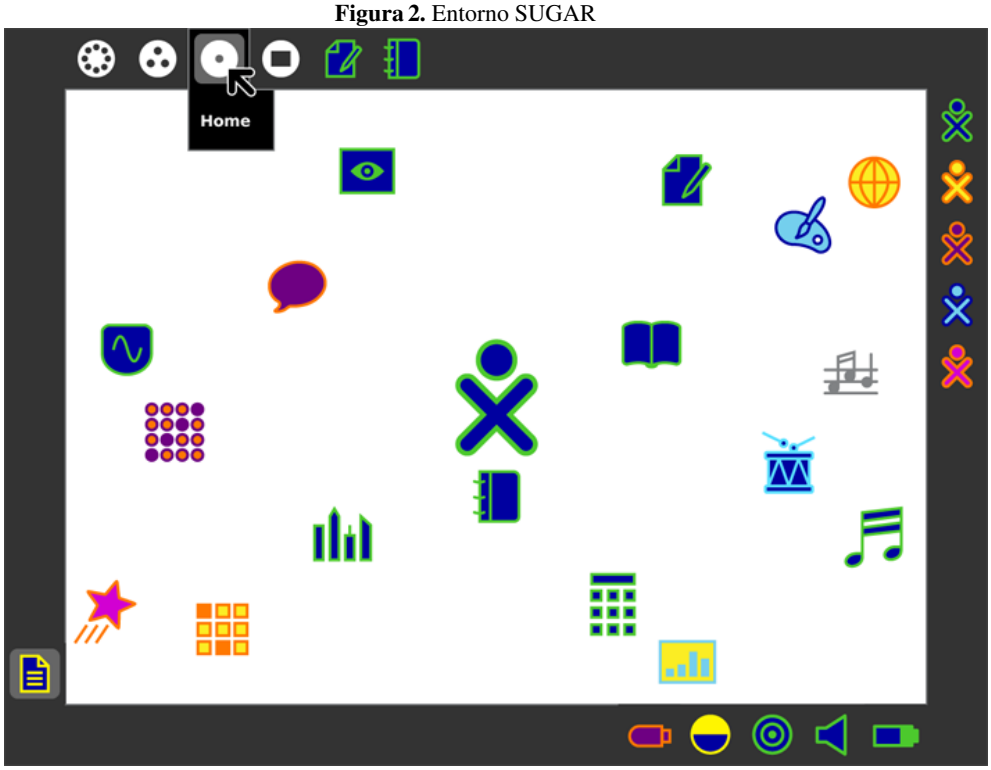
Generalmente, dentro del proceso educativo, el docente explica una parte de la materia y, seguidamente, propone a los alumnos una actividad de aplicación de dichos contenidos. Sin embargo, el ABP se plantea como medio para que los estudiantes adquieran esos conocimientos y los apliquen para solucionar un problema real o ficticio, sin que el docente utilice la lección magistral u otro método para transmitir ese temario. Barrows (1986) define al ABP como "un método de aprendizaje basado en el principio de usar problemas como punto de partida para la adquisición e integración de los nuevos conocimientos". En esta metodología los protagonistas del aprendizaje son los propios alumnos, que asumen la responsabilidad de ser parte activa en el proceso. En robótica pedagógica, se utiliza la idea del "problema a resolver" para plantear una situación ante la cual el alumno tenga que desarrollar una solución, investigando y construyendo un sistema de automatización; en ese sentido, la idea del "robot" es solo una excusa para que los grupos generen una situación constructivista que refuerce su aprendizaje y sus modelos de conocimiento.

Usar componentes reciclados, a diferencia de los KITs de fabricantes, fomenta la investigación, por parte de los alumnos, de aspectos físicos y mecánicos que en otros sistemas ya vienen resueltos. En ese sentido, las soluciones planteadas por los alumnos pueden ser menos "espectaculares" a nivel visual que un sistema pre armado por un fabricante, pero su carácter pedagógico se ve reforzado por el hecho de tener que resolver todos los aspectos de fabricación y no sólo dedicarse a seguir instrucciones diseñadas.

4. Sugar

Sugar es un entorno gráfico diseñado para trabajar con niños de edad escolar dentro del proyecto OLPC (*One Laptop per Child*), se trata de un sistema con kernel (núcleo) GNU/linux FEDORA y un entorno gráfico propio (SUGAR) diseñado especificamente para niños. En SUGAR se replantea el paradigma del "oficinista" como forma de trabajo gráfico; los alumnos no usan ficheros, escritorio, etc. En cambio, el sistema está pensado como un "vecindario" donde los chicos pueden hacer trabajos colaborativos, tener sus aplicaciones ordenadas y a la vista, y llevar un "diario" (*Journals*) donde se graban sus "actividades". Una característica técnica de SUGAR, es que su entorno gráfico esta íntegramente programado en el lenguaje Python, permitiendo tener de forma fácil el código fuente para hacer modificaciones o adaptaciones especificas. SUGAR tiene su propia constelación de software de carácter pedagógico, pensado como "actividades". Aunque al ser un sistema GNU/linux FEDORA, puede ejecutar cualquier software compilado para esa "distribución", pero la integración con las actividades especificas para SUGAR permite tener más orden dentro del entorno gráfico.

Figura 2. Entorno SUGAR

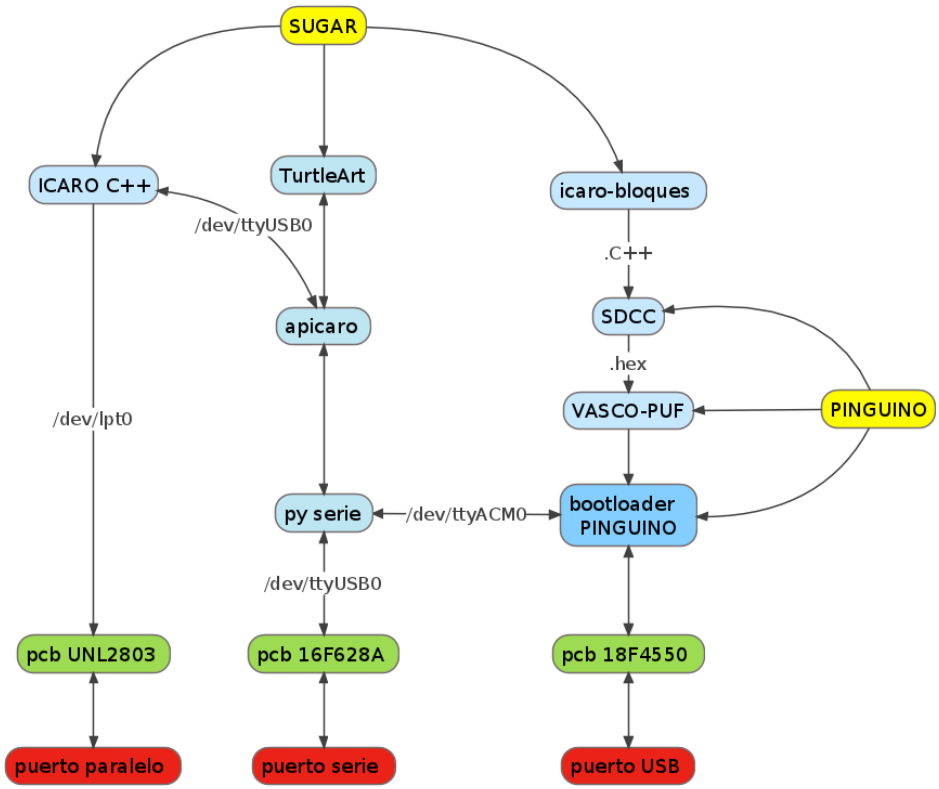


5. Proyecto Icaro

ICARO es un proyecto de desarrollo de software y hardware libre para la enseñanza de robótica en colegios primarios y secundarios. Se trata de acercar de forma transparente y sencilla los fundamentos de la robótica tratando de simplificar conceptos técnicos complejos para que los docentes necesiten conocimientos mínimos para poder trabajar en el aula.

Se compone de una serie de paquetes de software que trabajan con placas de hardware de bajo costo y fabricación, permitiendo investigar y diseñar pequeños robot pedagógicos de forma sencilla, reciclando componentes de electrónica y aprovechando las características de los distintos laboratorios de informática que se pueden encontrar en los colegios. La idea principal es lograr software de muy bajos requerimientos que pueda aprovechar cualquier tipo de computadora existente en un colegio o las netbooks del plan OLPC, CONECTAR IGUALDAD o cualquier otro plan provincial vigente.

Figura 3. Mapa de los distintos paquetes del proyecto ICARO y su relación con el hardware



6. ICARO C++

Icaro empezó como un lenguaje de pseudo código escrito en C++, pensado para trabajar con computadoras antiguas con puerto paralelo y bajos recursos. La idea era poder trabajar en la terminal de los sistemas GNU/linux, programando directamente sobre el puerto. El circuito de hardware para trabajo con ICARO C++ es simplemente un integrado UNL 2803 que eleva la tensión de salida de los 8 pines I/O del puerto paralelo, permitiendo tener 8 salidas y 6 entradas digitales. ICARO C++ es un software que trata de ser lo mas genérico posible para poder ser portado a cualquier arquitectura que soporte compilación con GCC.

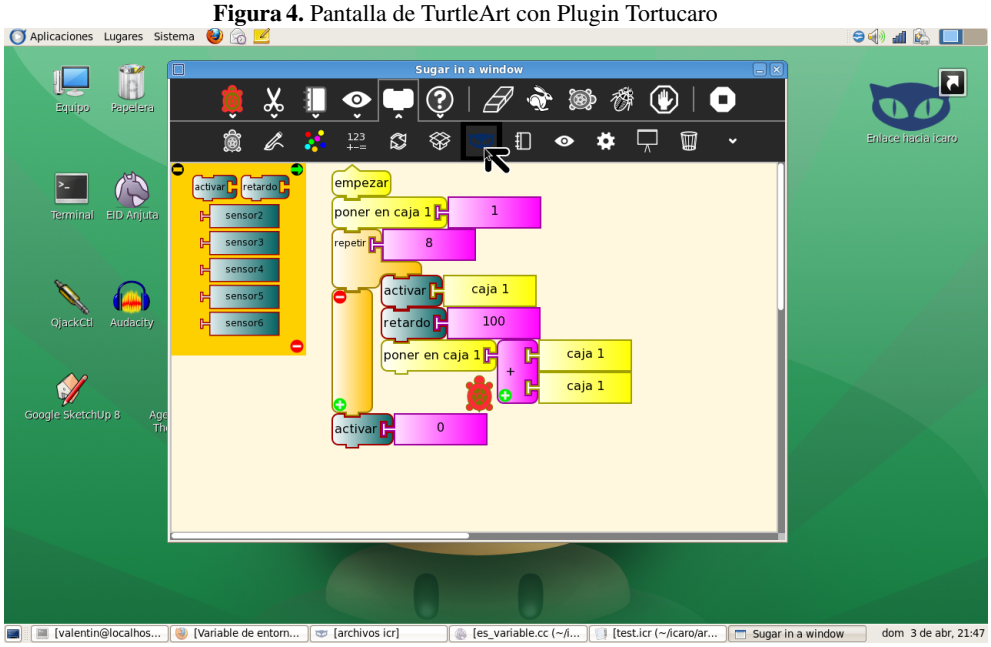
Con el desarrollo de una placa capaz de trabajar con puerto serie (basada en los pics 16f628A), el lenguaje se adaptó para leer y escribir en dichos puertos. Actualmente el desarrollo de ICARO C++ está detenido y soló es recomendable para computadoras muy antiguas, dado que los sistemas modernos ya no vienen los conectores de puerto paralelo, ni puerto serie, los cuales fueron remplazados íntegramente por el estándar USB.

7. Turtleart

Uno de los programas instalados por defecto dentro del sistema SUGAR es Turtleart[1], este es una tortuga gráfica inspirada en LOGO, que permite el dibujo artístico mediante el ensamblaje de funciones de programación. Se trata de una actualización del famoso lenguaje LOGO diseñado por Seymour Papert y adaptado para trabajar dentro del proyecto OLPC.

[1] http://wiki.sugarlabs.org/go/Activities/TurtleArt

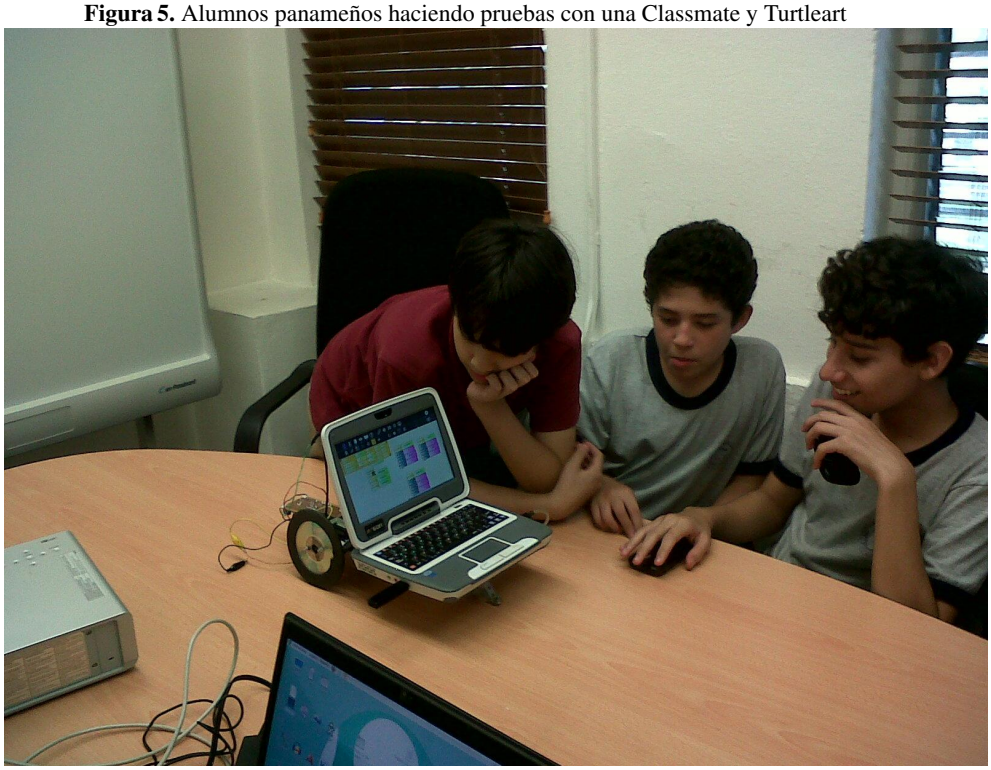
Figura 4. Pantalla de TurtleArt con Plugin Tortucaro



8. Tortucaro

Tortucaro es un Plugin para Turtleart que implementa una serie de primitivas (bloques básicos dentro de LOGO) que permiten generar una capa de abstracción para leer y mandar señales a las placas de control. Aprovechando las primitivas de TurtleArt y agregándole el plugin Tortucaro, se puede desarrollar un robot que responda a las señales del puerto Serie (o mediante un conversor serie/usb como el integrado ftl232) a traves de una API programada en python y Pyserial.

Figura 5. Alumnos panameños haciendo pruebas con una Classmate y Turtleart



El uso de Turtleart permite aprovechar la documentación y cuadernos del aula existentes para dar una visión inicial del software sin necesidad de trabajar desde el principio con hardware.

9. Icaro-bloques

Con Tortucaro se puede trabajar conectado a la netbook y usando placas basadas en los integrados de Microchip PIC 16F628A[2] (la gama más básica de PIC), aunque no permite el desarrollo de robots independientes de la netbook. Para poder trabajar con micro controladores con capacidad para tener un *Bootloader* (como los PICs 18F4550[3], 18F2550[4]) se desarrolló ICARO-BLOQUES, un entorno gráfico muy similar a TurtleArt, pero específico para generar código ANSI C para el compilador SDCC. ICARO-BLOQUES está basado en todo el trabajo hecho para el proyecto PINGUINO, un clon de ARDUINO pero hecho con arquitectura de PIC (18F2550).

[2] ww1.microchip.com/downloads/en/devicedoc/40044f.pdf
[3] ww1.microchip.com/downloads/en/devicedoc/39632c.pdf
[4] http://ww1.microchip.com/downloads/en/DeviceDoc/39632e.pdf

PINGUINO fue escrito por Jean Mandom con la idea de poder tener un hardware basado en PIC con compatibilidad con las placas ARDUINO (basadas en ATMEL) y escrito integramente en PYTHON. PINGUINO a su vez está basado en el proyecto VASCO-PUF que diseñó un *boatloader* y un software para carga del mismo. El integrado PIC 18F4550 utilizado en PINGUINO tiene varias características interesantes para su elección dentro del proyecto ICARO: es más barato que su equivalente en ATMEL, tiene integración USB por hardware, viene disponible en formato DIP (que facilita la fabricación de PCBs al no usar soldadura superficial), posee 40 pines de entrada/salida, PWM, 8 entradas analógicas (PORT-A), conexión Serie, etc.

La experiencia lograda con el PCB basado en PIC 16F628A, hizo que en el desarrollo del PCB nuevo se tomaran algunas consideraciones de diseño:

1. Se mantuvo la compatibilidad con los puertos de entrada digital del PCB NP03 / NP04 (16f628A).

2. Se separaron 4 puertos del PIC para poder controlar un *driver* de potencia (L293D) para motores de corriente continua.

3. Cinco puertos para servos motores de aero modelismo.

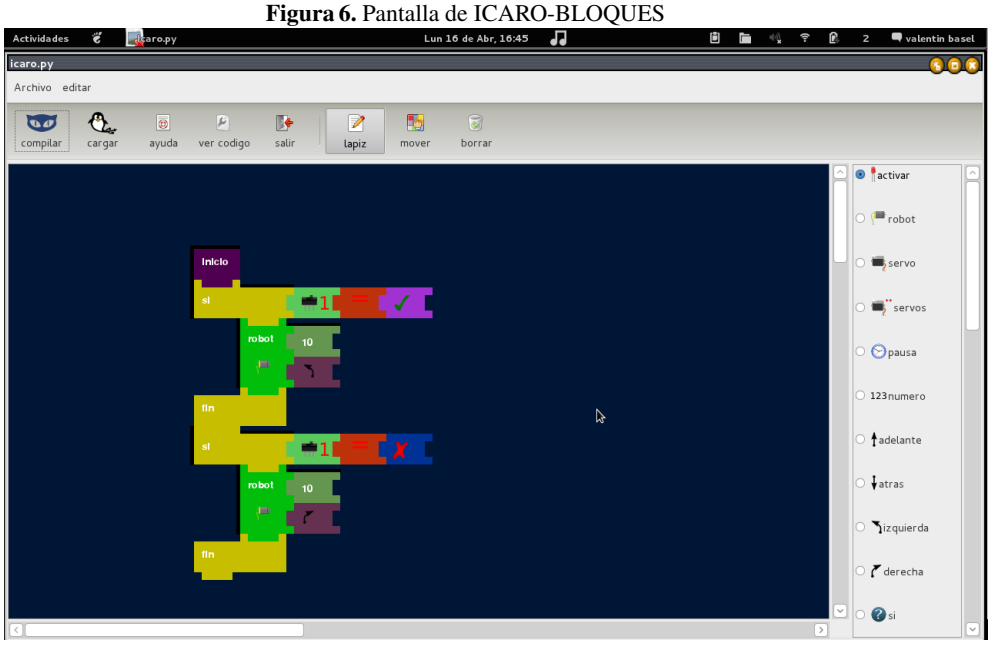
4. Ocho puertos de entrada analógica.

5. Ocho puertos de Expansión para mantener compatibilidad con Tortucaro y los pcb NP03 / NP04.

De esta forma el hardware queda más “cerrado”, pero mucho más funcional para trabajar en el aula con materiales mínimos: un par de motores CC, servos y sensores digitales -como un botón o un swicht “final de carrera”-.

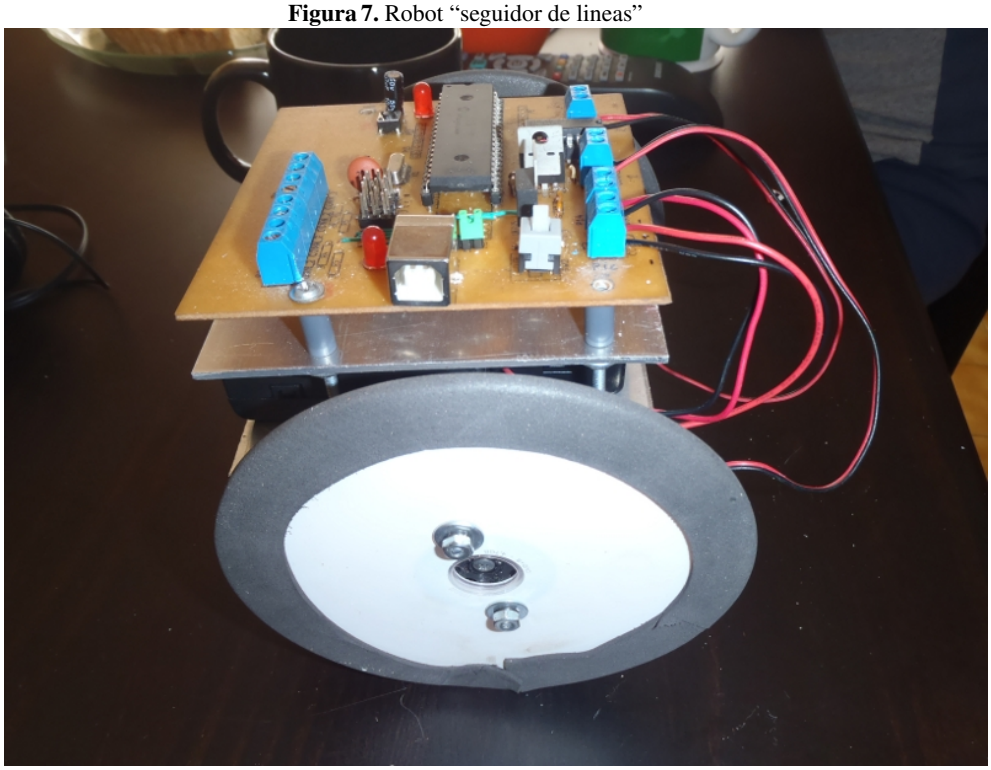
El software se condicionó a las especificaciones de hardware para simplificar el desarrollo de la lógica de un robot sencillo pero con la potencia de un lenguaje de programación como ANSI C.

Figura 6. Pantalla de ICARO-BLOQUES



En el gráfico anterior se puede observar el código para generar la lógica de un robot seguidor de línea de dos motores y un sensor cny70. Cuando el sensor marca 1 (blanco) el robot gira a la izquierda, cuando marca 0 (negro) gira a la derecha.

Figura 7. Robot “seguidor de lineas”



Este robot fabricado con componentes reciclados, puede seguir una línea negra sobre fondo blanco, toda la circuiteria de control está alojada en la placa NP05 y sólo es necesario un destornillador para conectar los cables a las borneras respectivas. La placa fue fabricada de forma casera, usando una impresora laser para transferir el diseño CAD del circuito a una plaqueta de cobre y cloruro férrico.

10. Experiencia con Icaro en las escuelas

En el año 2011, la Escuela de Oficio San Cayetano[5] (Bs. As) decidió usar ICARO C++ para dar un curso de robótica para los alumnos de sus talleres. Consiguieron un fabricante de PCBs y fueron los mismos alumnos quienes soldaron los componentes de sus placas. Como en las instalaciones de esta escuela tienen PCs bastantes antiguas, pudieron trabajar con el puerto paralelo y usar el software ICARO C++. Los chicos diseñaron robots usando juguetes rotos y placas Icaro conectadas al puerto paralelo.

[5] http://roboticaeducativa2011.wordpress.com/2011/03/29/curso-de-robotica-en-san-cayetano/

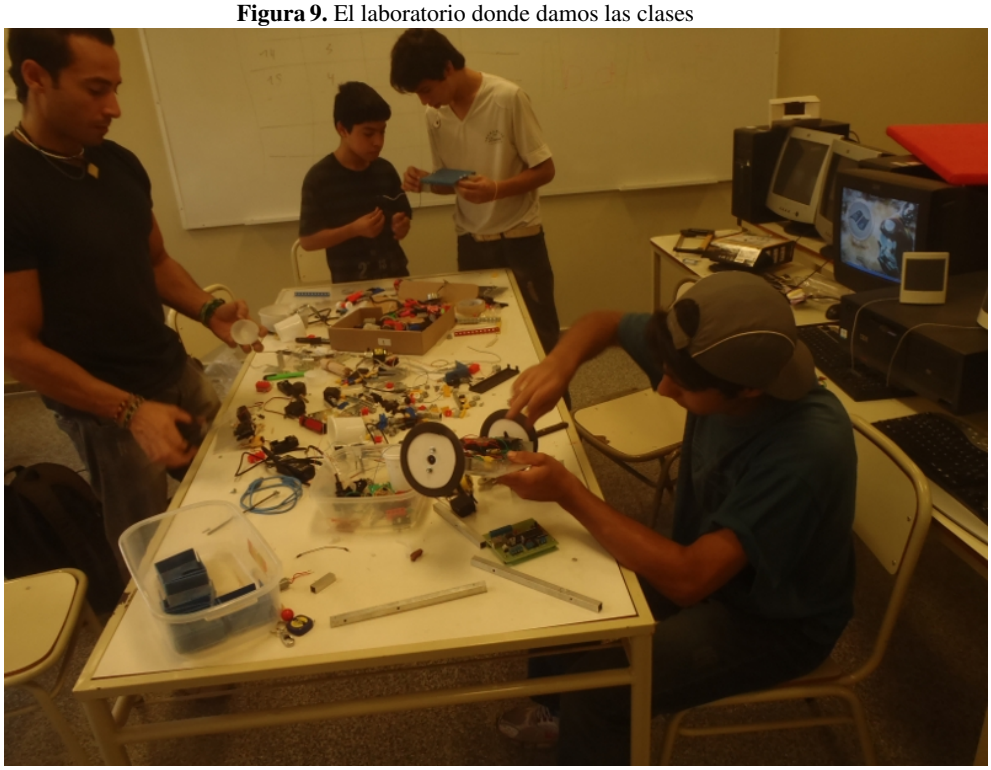
Figura 8. Curso de robótica en la Escuela de Oficios San Cayetano

Actualmente estamos dando un taller en el marco del Centro de Actividades Juveniles [6] (CAJ) en la escuela CBU rural "La Carbonada" anexo del IPEM 320 "Jorge Cafrune", con chicos de quinto año, donde desarrollamos la actividad de fabricar pequeños robots con componentes reciclados de impresoras y juguetes RASTI[7].

Los chicos tienen que diseñar un robot para SUMOBOT con el cual compiten contra un robot diseñado por los docentes. Nos separamos en grupos de a dos y comenzamos a ver distintos robots de "pelea" diseñados por universidades así como las mismas competencias. Luego, cada grupo comienza a recolectar componentes que podrían necesitar para el desarrollo de su robot: motores, piezas de plástico, cartón, etc.

[6] http://www.cba.gov.ar/vercanal.jsp?idCanal=3267
[7] http://www.rasti.com.ar/

Figura 9. El laboratorio donde damos las clases



Organizamos el trabajo en bloques para poder aprovechar el tiempo de clases. Las dos primeras clases son para desarrollar la mecánica del robot, sobre todo investigar comó es un robot de SUMO. Los chicos tienen que tener en cuenta características de los motores y peso del robot, así como su resistencia. En las dos clases siguientes usarán ICARO-BLOQUES para poder programar una estrategia de pelea contra el robot de los docentes y diseñarán un chasis de cartulina.

11. Conclusiones

La robótica pedagógica se presenta como una alternativa interesante para el desarrollo de actividades en el ámbito escolar, permite abordar un gran número de situaciones constructivistas donde los alumnos pueden diseñar y aprender mediante el estudio de un sistema de automatismo (un robot) y generar situaciones de apropiación del conocimiento.

ICARO se plantea como una solución para poder implementar la robótica pedagógica aprovechando las ventajas que ofrecen el software y hardware libre, simplificando el contenido técnico (a nivel de programación y electrónica) y permitiendo concentrar la atención de los alumnos en el desarrollo de un problema concreto. En este contex-

to, el robot (o cualquier sistema de automatización) sólo será una excusa para trabajar conceptos de las ciencias, matemáticas etc, y no una finalidad en si misma.

Las ventajas de contar con el código fuente de todo el proyecto (tanto del software como del diseño de hardware) permite investigar y mejorar el mismo, para adaptarlo a necesidades concretas, o la posibilidad de "escalar" en complejidad, trabajando a muy alto nivel de programación o con código ANSI C directamente sobre el micro controlador.