Zorzal: Un ambiente colaborativo basado en bloques para competencias de programación

Stacco Joan Manuel[0000-0002-6450-6567], Kogan Pablo[0000-0003-1195-7556], Godoy Ingrid[0000-0002-2298-1378], Rodríguez Jorge[0000-0002-4697-6477], and Carina Fracchia[0000-0003-2437-7112]

Facultad de Informática - Universidad Nacional del Comahue
Buenos Aires 1400, 8300 Neuquén, Argentina
joan.stacco@est.fi.uncoma.edu.ar, pablo.kogan@fi.uncoma.edu.ar, ingrid.godoy@fi.uncoma.edu.ar, j.rodrig@fi.uncoma.edu.ar, carina.fracchia@fi.uncoma.edu.ar

Resumen Desde que las Ciencias de la Computación se incluyeron como contenido en la escolaridad obligatoria, es importante trabajar sobre modelos didácticos, herramientas y estrategias que ayuden en la enseñanza y el aprendizaje de esta ciencia para facilitar a los estudiantes la comprensión de estos conocimientos. En este documento presentaremos a la herramienta Hornero como gestor de torneos de programación en su nueva versión Zorzal, que posibilita el desarrollo de programas en bloques, de forma colaborativa, con una interfaz responsive que permite a los docentes crear sus propios torneos, problemas y hacer seguimiento de las resoluciones provistas por los estudiantes. Hornero es un sistema desarrollado por estudiantes, docentes y graduados de la Universidad Nacional del Comahue que implementa un modelo de enseñanza y aprendizaje lúdico y colectivo, a través de competencias de programación.

Keywords: Enseñanza de la Programación · Programación por Bloques · Programación Colaborativa · Plataforma de Programación Competitiva

1. Introducción

El interés creciente por incorporar la enseñanza de prácticas y conceptos fundamentales sobre Ciencias de la Computación en las propuestas curriculares para la educación obligatoria da lugar al desarrollo, a nivel global, de políticas e iniciativas en esta dirección[13,11].

Conocer sobre Ciencias de la Computación se ubica como un tipo de conocimiento fundamental para mejorar las posibilidades de comprender e intervenir un mundo donde la influencia de la informática se intensifica día a día, además de ser pieza clave para el ejercicio pleno de la ciudadanía.

Se observa que, si bien varios países realizan esfuerzos en esta dirección recorriendo trayectorias singulares en la construcción de propuestas curriculares para la informática en las escuelas, la integración de las Ciencias de la Computación de forma rigurosa y sostenida se encuentra en proceso de consolidación en la mayoría de los casos.

En este contexto la enseñanza de conceptos del área algoritmos y programación ocupa un lugar preponderante. Las tecnologías imponen nuevas realidades en la sociedad actual que requieren cambios dinámicos e innovadores, para poder mejorar los sistemas educativos [11].

Aprender sobre algoritmos y programación contribuye al desarrollo de habilidades para la resolución de problemas, ayuda a comprender como funciona el mundo y abre la posibilidad de establecer conexiones con conceptos fundamentales de la informática. Se trata de un cuerpo de conocimiento indispensable para la educación obligatoria, considerado equivalente a otros núcleos de aprendizajes prioritarios como lengua y matemática [4,6].

Los enfoques de programación basada en bloques y los torneos de programación ofrecen formas efectivas para proporcionar un primer encuentro entre estudiantes, de edades cada vez más tempranas, con conceptos sobre algoritmos y codificación.

La programación basada en bloques resulta ser un enfoque didáctico disciplinar especialmente adecuado para estudiantes sin formación previa en el área de conocimiento. Este ambiente proporciona formas simples de operar elementos sintácticos del lenguaje, mejora la legibilidad de los programas y evita errores de sintaxis básicos [15,18].

Los torneos de programación introducen la lógica del juego al proceso de aprendizaje, proponiendo desafíos breves que ponen en juego conocimientos sobre algoritmos con retroalimentación rápida. La secuenciación de desafíos breves con retroalimentación rápida contribuye al mejoramiento de los procesos formativos, por un lado ofrece al estudiante una oportunidad de poner a prueba los conocimientos elaborados, confirmar aprendizajes e identificar debilidades. Por otra, parte permite al docente recuperar información acerca de los procesos de aprendizaje en desarrollo con intención de ajustar las propuestas de enseñanza [5,3].

Si bien la programación basada en bloques demuestra ser un enfoque efectivo y es la forma más frecuente en la que los estudiantes establecen un primer contacto con la codificación, se espera que en el trayecto de la educación obligatoria, los mismos logren construir habilidades para desarrollar programas utilizando lenguajes de programación convencionales basados en texto. El pasaje a un entorno de programación tradicional suele resultar un proceso frustrante para la mayoría de los estudiantes, en este sentido el problema de la transición de bloques a texto emerge como un tópico de estudio para la enseñanza de la programación. Una posible dirección para el abordaje del problema, la expresan los entornos híbridos que combinan la representación basada en bloque con la basada en texto, permitiendo cambiar en forma bidireccional la manera en que se expresa un algoritmo[19,20].

Hornero es un gestor de torneos de programación, desarrollado por la Facultad de Informática, que está diseñado bajo las premisas de permitir competir empleando cualquier lenguaje de programación y ofrecer la posibilidad de jugar a todas las personas que quieran participar [7].

A partir del año 2014 la Facultad de Informática de la Universidad Nacional del Comahue comienza a desarrollar una Línea de Investigación y Desarrollo articulada a iniciativas en el ámbito de Extensión Universitaria que proponen la aproximación de estudiantes y escuelas secundarias a la programación a partir de la realización de torneos de programación gestionados por Hornero.

Esta iniciativa extensionista se viene realizando desde ese año en colaboración con varias escuelas secundarias de la región. En este espacio de colaboración de carácter interinstitucional se desarrollan actividades de formación docente en el campo de la enseñanza de la programación y de formación a estudiantes secundarios en conceptos fundamentales del área de conocimiento, promoviendo el aprendizaje basado en la resolución de problemas en un ambiente colaborativo [9].

En este trabajo se presenta el desarrollo de una herramienta que permite gestionar un ambiente colaborativo basado en bloques para competencias de programación y que, además, facilita el proceso de transición de bloques a texto.

El resto del documento se organiza de la siguiente manera. En la próxima sección, se presentan estudios previos que se ubican como marco conceptual para esta propuesta. A continuación, en la sección 3, se describen experiencias realizadas en el ámbito de la línea de investigación y desarrollo que conforman el espacio de trabajo experimental. En la sección 4 se incluye una descripción detallada de Hornero Versión Zorzal . El documento cierra con las secciones 5 y 6, en las que se formulan conclusiones elaboradas en el ámbito de este desarrollo y se postulan l´ıneas de trabajo futuro.

2. Marco Conceptual

En esta sección se describen los conceptos base que intentan responder el porqué es importante generar recursos para enseñar conceptos sobre algoritmos y codificación a partir del desarrollo de competencias de programación en un ambiente colaborativo basado en bloques.

Se han encontrado investigaciones que muestran el uso del aprendizaje competitivo para la adquisición de habilidades de programación de sistemas [17,3]. Los torneos de programación han sido promovidos desde el año 1970 por empresas e instituciones que buscan promocionar distintos lenguajes y entornos de programación, presentar innovar con tecnologías nuevas o simplemente como actividad de ocio.

Por lo general se tiende a que una persona o un grupo, desarrolle una aplicación, desde cero o una parte, como podría ser un módulo. Habitualmente se determina un tiempo fijo, y luego, un jurado formado por la comunidad o un grupo de personas vota y elige la solución ganadora.

Existen gestores de torneos online que facilitan la creación de torneos, en los cuales se puede especificar: el propósito del mismo, como diversión, entrenamiento, evaluación o la enseñanza de un tópico de programación; la duración del torneo que podría ser más flexible (abierto o sin límite) o más acotada (un día y horario prefijado); los de lenguajes de programación habilitados para competir;

los tópicos de programación y niveles de complejidad que se desean evaluar; y el idioma en el que se presentan los enunciados.

Algunas ventajas de estos sistemas online son la posibilidad de que se participe desde cualquier lugar del mundo, el almacenamiento de estad´ısticas que permitan analizar dificultades en cuando a comprensión de enunciados, cantidad de problemas resueltos y tiempos empleados en la resolución. Si el torneo está configurado como una tarea académica, los parámetros mencionados permitirán al docente tener un panorama de los temas que se han comprendido y en cuáles debe profundizar.

Para favorecer la concreción de la competencia sana hay que tener en cuenta un reglamento simple y claro, sumado a iguales condiciones de participación en cuanto a conocimiento del lenguaje, buena conexión a Internet y disponibilidad equivalente de dispositivos. Para no depender de la conectividad, se trabaja con un puntaje asociado a los ejercicios resueltos, sin considerar el tiempo que ha llevado resolverlo o cuestiones relacionadas a la eficiencia del algoritmo.

Dentro de un contexto educativo comúnmente suelen presentarse situaciones de competencia (trabajos individuales, evaluaciones) y de cooperación. Generalmente en una situación de competencia los estudiantes sienten que pueden lograr sus objetivos, si y sólo si, los demás no consiguen alcanzar los propios, en cambio en un contexto de colaboración los estudiantes sienten que pueden lograr sus objetivos, si y sólo si, los estudiantes de su grupo también alcanzan los propios [10].

En un contexto académico, los torneos facilitan que los grupos de aprendizaje puedan durar entre una clase y varias semanas, asegurando que los estudiantes se involucren de manera activa e intelectualmente organizando el material, realizando resúmenes y explicaciones con el fin de integrarlo en las estructuras conceptuales existentes. Así mismo, se podrían usar grupos con poca duración, cuando los torneos tengan como fin concentrar la atención, establecer expectativas sobre el tema a trabajar o hacer un cierre de un tema específico.

En el contexto del aprendizaje colaborativo, es fundamental la gu´ıadocente en la toma de decisiones respecto a la forma de organizar los grupos y su tamaño, definición y asignación de roles a cada integrante, organización del aula y la selección de los materiales a usar durante la actividad propuesta. Se debe hacer hincapié en la explicación de la tarea y la estructura cooperativa comentando los criterios para el éxito, la responsabilidad individual y las conductas esperadas por parte de los estudiantes. Como en toda planificación de una actividad se deben especificar los objetivos académicos y de habilidades sociales, que expongan claramente las habilidades interpersonales que se quieren desarrollar en la actividad grupal.

En un trabajo grupal cada integrante que se suma a un grupo aporta aptitudes, destrezas y habilidades diferentes. Los grupos pequeños permiten aumentar la visibilidad y esfuerzo de los estudiantes, se evidencia fácilmente quién participa y trabaja de los que no lo hacen, lo que favorece el aumento de la responsabilidad de los mismos.

El trabajar con grupos de más de 5 estudiantes, si bien es mayor la cantidad de habilidades presentes, requiere de una mayor cantidad de interacciones entre sus miembros, lo cual puede resultar complejo al momento de construir consenso para la toma de decisiones. Suele suceder que la cantidad de recursos (materiales, dispositivos, etc.) o la naturaleza específica de la actividad es la que determina la cardinalidad del grupo.

Lo ideal sería trabajar con grupos heterogéneos dado que los integrantes aportan diferentes modos de encarar la resolución de problemas, se estimula la creatividad, y permite compartir puntos de vistas diferentes. Según la caracter´ısticas de los grupos de estudiantes y tipo de torneo, se puede trabajar con la conformación de grupos heterogéneos mediante procedimientos azarosos, evitando que los propios estudiantes propongan los miembros del grupo dado que de esta manera se tiende a grupos homogéneos.

El uso de roles facilita estructurar los esfuerzos cooperativos dando a conocer a los estudiantes lo qué se espera de ellos dentro del grupo que integran. Se recomienda la asignación de roles y alternarlos asegurando la rotación de los mismos.

El diseño del ambiente donde se realiza el torneo, también es un aspecto a tener en cuenta, dado que influye en el clima de aprendizaje que ayudará a los estudiantes a sentirse cómodos y seguros, además de contribuir a la participación de los estudiantes y docentes, en su comunicación y oportunidades para el contacto social de los mismos.

Al analizar el uso de competiciones en educación se pueden encontrar posturas en contra y otras a favor, respecto a estas últimas se mencionan la posibilidad de quitarle rigidez al plan de estudio cuando este no es lo suficientemente desafiante o de favorecer un incremento de la motivación de los estudiantes al proporcionarles una retroalimentación en base a su desempeño.

En relación a las competencias de programación, se encuentran experiencias que muestran en sus resultados que los estudiantes que pertenecen a los equipos de entrenamiento demuestran mejores desempeños en las asignaturas relacionadas a la programación, pues cada uno de los desafíos implican dedicación y destreza [12].

Se pueden encontrar experiencias del uso de jueces online en cursos de programación, como la presentada en [5], estos varían de acuerdo al manejo de los problemas (temas, nivel de complejidad), lenguajes de programación, parámetros tenidos en cuenta para definir la correctitud del ejercicio (tiempo de ejecución, uso de recursos, buen estilo de programación). Estas experiencias sumado a los buenos resultados del aprendizaje grupal colaborativo en el lenguaje de Bloques son la bases para el desarrollo de hornero versión Zorzal.

3. Antecedentes

A partir del año 2014 como respuesta a las limitaciones del Torneo Argentino de Programación en cuanto a la complejidad de los ejercicios y los lenguajes disponibles para competir, ya que el torneo solo permitía programar en “Java”

o "C", se comenzó a desarrollar una Línea de Investigación y Desarrollo articulada a iniciativas en el ámbito de Extensión Universitaria que proponen la aproximación de estudiantes y escuelas secundarias a la programación a partir de la realización de torneos de programación gestionados por Hornero [7]. Esta iniciativa extensionista se viene realizando desde ese año en colaboración con varias escuelas secundarias de la región, donde se desarrollan actividades de formación docente en el campo de la enseñanza de la programación y de formación a estudiantes secundarios en conceptos fundamentales del área de conocimiento [9], incentivando el aprendizaje grupal y colaborativo.

Esta herramienta ha brindado el soporte tecnológico de todos los Proyectos de Extensión alineados bajo la categoría de Torneos de Programación y Resolución de Problemas, que se han llevado a cabo desde 2014 hasta la fecha, logrando intervenir en la formación de más de 1700 estudiantes secundarios a lo largo de estos años. Actualmente figuran 850 usuarios reales en la base de datos de Hornero. Considerando que esta herramienta ha sido utilizada principalmente en los encuentros organizados a través de los Proyectos de Extensión, siendo generosos al estimar una cota inferior al suponer que solo las dos terceras partes de los 850 usuarios totales fueron equipos de tres estudiantes (generalmente llegaban a cinco), y que el tercio restante fue de usuarios individuales, es matemáticamente evidente que el alcance de la herramienta abarca como m´ınimo a 1700 estudiantes secundarios.

Hasta el momento, ofrece dos modalidades de torneos, la primera de ellas consiste en resolver problemas con lápiz, papel y calculadora, y seleccionar una respuesta correcta entre tres opciones posibles (multiple choice). La segunda modalidad se trata de elaborar una solución para el problema propuesto mediante el desarrollo de programas escritos en algún lenguaje de programación. En ambas variantes, Hornero se encarga de mostrar el enunciado del problema, validar la solución enviada y responder si ésta es o no correcta. La modalidad utilizada para cada torneo depende del nivel de contenidos alcanzado por las diferentes escuelas que participan.

Durante estos años, los torneos realizados en el marco de los Proyectos de Extensión se desarrollaban tanto de manera presencial en encuentros organizados para tal fin, como también algunos torneos virtuales, en los cuales los grupos de estudiantes competían desde cada establecimiento educativo en una fecha y hora determinada previamente.

La versión Zorzal de Hornero propone incorporar el desarrollo de programación basada en bloques, de manera colaborativa y con una interfaz adaptable (responsive). Esto significa que los integrantes de cada equipo puedan compartir un mismo espacio de programación por bloques, estando en distintos lugares físicos y utilizando diferentes dispositivos electrónicos.

4. Hornero - Versión Zorzal

Hornero ha sido creado teniendo como meta dos premisas, primera *que pueda competir cualquier persona, para todos*, y segunda *sea posible participar pro-*

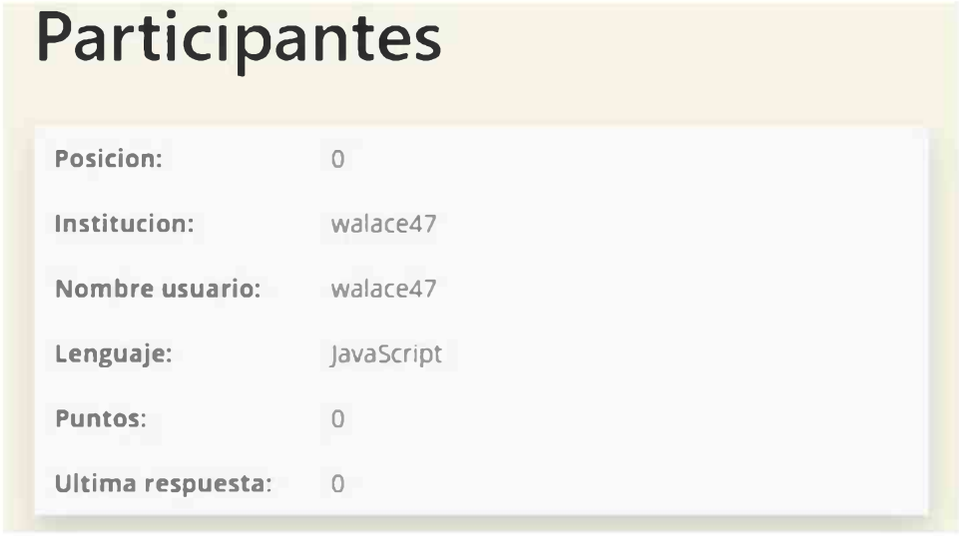
gramando en cualquier lenguaje. Hasta el año 2019, en los torneos con estudiantes de nivel secundario, se ha trabajado sobre una propuesta didáctica basada en la resolución de ejercicios en papel durante un encuentro para luego pasar a resolución de problemas utilizando pseudocodigo en PSeInt http://pseint.sourceforge.net/, con muy buenos resultados en relación a la asimilación de conceptos sobre programación.

La nueva versión *Zorzal*, que se presenta en este trabajo, pretende mejorar el cumplimiento de las dos premisas originales bajando el piso, en esta dirección se incluye un ambiente de programación basado en bloques [15] [18]. Además agrega una tercer premisa, *que se pueda jugar de forma colaborativa*.

Así mismo, permite a docentes o administradores crear competencias incorporando nuevos problemas o seleccionando algunos del pool de ejercicios precargados, como también hacer corrección y seguimiento sobre los trabajos elaborados por los estudiantes. A su vez, cada ejercicio tiene asignado un conjunto de etiquetas sobre características y complejidades que permiten la catalogación de los mismos y facilitar la búsqueda a la hora de armar los torneos.

La dinámica de los torneos comienza conformando grupos de entre tres a cinco estudiantes, que se registran en la herramienta hornero2.fi.uncoma.edu.ar. Los grupos se distribuyen ocupando todo el espacio disponible del aula, si se trata de un torneo presencial o bien cada uno desde su escuela o casa para el caso virtual. La pantalla principal de la competencia permite visibilizar en todo momento una tabla de posiciones y la cuenta regresiva del torneo. Dichos elementos se van actualizando periódicamente y agrega una cuota de adrenalina a la competencia.

Figura 1: Menú principal desplegado en un dispositivo móvil



(a) Tabla participantes vista móvil

(b) Tabla participantes vista web

Figura 2: Tabla de participantes

Arquitectura Responsive

Zorzal tiene un arquitectura orientada a servicios. Esta arquitectura tiene varios beneficios, entre ellos, desacoplar el cliente del servidor y generar aplicaciones reutilizables y adaptables, además de facilitar la integración de un rango mayor de tecnolog´ıas.

Como se muestra en la Figura 1 la nueva versión adapta la presentación del contenido a las dimensiones del dispositivo que se está utilizando, permitiendo que la visualización se acomode satisfactoriamente a monitores, proyectores y diferentes dispositivos móviles como celulares o tabletas. Para lograr esto se usan librerías de css, primefaces [1] y boostrap 4 [16], que permiten diseñar alternativas para el despliegue de datos en diferentes pantallas. En la Figura 2b se observa una tabla desplegada en un monitor, mientras que la figura 2a muestra como se presenta esa misma tabla en un dispositivo móvil. Estas librerías integran interfaces para que sea más fácil el manejo de de componentes en diferentes pantallas, ya sea tablas o formularios que cambian totalmente la forma de estructurarse.

Programación en Bloques

La implementación del ambiente de programación basado en bloques de *Zorzal* se basa en librer´ıa open source Blockly desarrollada por Google. Blockly provee una interfaz web con entorno de programación que permite arrastrar, soltar y conectar bloques que representan elementos sintácticos del lenguaje sobre un lienzo. Una colección de bloques conectados conforman un programa que se puede ejecutar directamente en el navegador [8,14].

Al entorno ofrecido por defecto por la librer´ıa, se agregan dos nuevos bloques que permiten conectar con una competencia Hornero o ejecutar de manera local. La Figura 3, muestra el espacio de trabajo desarrollado para Hornero Zorzal, Por otro lado en la Figura 4 se puede observar los bloques agregados a Blockly.

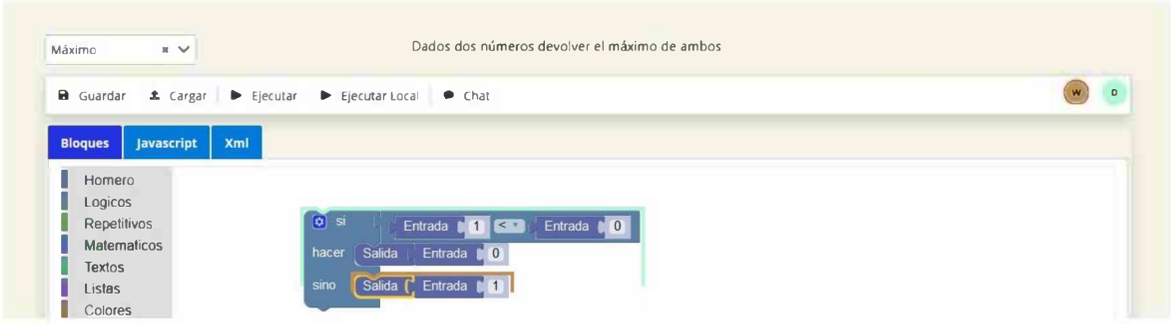


Figura 3: Espacio de trabajo colaborativo

1. Bloque Entrada, este bloque se utiliza para tomar las entradas, parámetros, que envía el torneo Hornero. En la ejecución local, solicita una entrada por teclado. Figura 4b.
2. Bloque Salida, este bloque tiene una conexión a la derecha que permite insertar el valor a enviar como respuesta del problema ofrecido por el torneo Hornero. En la ejecución local, se usa ese bloque para desplegar alguna respuesta directamente en el navegador. Figura 4a.

Ejecuciones. El aplicativo tiene dos modos de ejecución, el *Modo Local* que permite, mediante una interfaz de usuario, probar entradas para el algoritmo y retornar por pantalla la respuesta, sin interactuar con el torneo. Por otro lado, el *Modo Competencia* posibilita interactuar con el servidor Hornero y jugar con otros equipos. Cuando se ejecuta en este último modo, al contestar tres veces consecutivas de forma correcta, se muestra un mensaje de felicitaciones y se informa que ha ganado un punto. Esta situación se refleja automáticamente el las tabla de posiciones del torneo.

(a) Bloque Salida

(b) Bloque Entrada

Figura 4: Bloques Entrada y Salida

En conjunto, los dos modos de ejecución, permiten a los estudiantes probar y depurar el código mientras lo consideren necesario utilizando el modo local, cuando están conformes con su solución la pueden ejecutar contra el servidor.

Compilación bloques a texto. La librería implementada, permite por defecto generar código en los siguiente lenguajes Javascript, Python, Dart, Lua, PHP y XML. Además se brinda una interfaz para generar código en otros lenguajes y producir código en bloques a través del lenguaje XML. En la versión Zorzal el código producido en bloques se compila automáticamente en XML y JavaScript, lo que posibilita que los estudiantes puedan comprender una forma de representación textual equivalente de la solución desarrollada.

Programación colaborativa

Hornero Zorzal, permite compartir un espacio de trabajo a grupos, con cardinalidad de entre 2 y 5 estudiantes, en las competencias de programación. Para lograr esto se realiza el siguiente procedimiento: primero, uno de los integrantes del equipo debe inscribirse en el torneo y crear el espacio de trabajo de programación en bloques; segundo, el dueño del espacio debe compartirlo con el resto de los integrantes a través de un enlace que referencia al espacio compartido; y por último los compañeros de equipo acceden al espacio de trabajo compartido utilizando el enlace.

Todos los integrantes del equipo pueden cooperar, agregando, borrando, moviendo o modificando bloques, para construir programas de forma colaborativa. Cada estudiante conectado está distinguido por un color, como muestra la Figura 5b. Cuando un miembro del equipo selecciona un nuevo bloque a agregar, el resto visualiza que bloque se está agregando y en que lugar del espacio de trabajo es agregado. Cada integrante participa desde su computadora rompiendo las barreras físicas.

La Figura 5a muestra como dos estudiantes comparten el proceso de elaboración de código, con marco celeste se destaca el bloque seleccionado por uno de ellos y en marrón el bloque manipulado por el compañero.

El espacio compartido le brinda a los docentes una herramienta de evaluación, seguimiento e intervención de los procesos de enseñanza y de aprendizajes. En este ambiente dispone de la posibilidad de administrar la ayuda y analizar

las secuencias de bloques desarrolladas por cada equipo, en el momento que está ocurriendo la competencia de manera sincrónica o una vez terminada la competencia de manera asincrónica.

(a) Usuarios conectados

(a) Bloques seleccionados

Figura 5: Progrmación colaborativa basada en bloques en Zorzal

La interacción paralela puede suceder dentro del espacio de trabajo a través de un chat embebido en aplicación o agregando un bloque de comentario que permite, además, documentar el código. Por otra parte, en forma complementaria al canal de chat embebido, es posible realizar una comunicación simultanea a través de un servicio de vídeo conferencia como Google-Meet, Jitsi Meet, Zoom o whatsapp, lo que permite una dinámica muy fluida del trabajo.

La implementación del chat y del espacio común se realizó con web-sockets en el servidor usando la librería socket.io [2] que mantienen de manera sincrónica los cambios que realiza cada miembro del equipo.

5. Conclusiones

Los conceptos de programación son incorporados de manera natural. La dinámica de los torneos partiendo de resolver cada problema varias veces, en papel para luego pasar al ambiente de programación en bloques, incita a los estudiantes a reutilizar el planteo resuelto en papel y transformarlo en bloques.

Por otra parte, Zorzal busca favorecer los procesos de transición desde la programación basada en bloques a la programación basada en texto y propone mecanismos amigables que ayudan a comprender la equivalencia entre las representaciones que se utilizan para expresar un algoritmo.

Asimismo, la posibilidad de trabajar de forma colaborativa permite a cada equipo llegar a soluciones construidas con el aporte de todos los integrantes. De esta manera se busca que los estudiantes aprendan a programar a través de la competencia y en un entorno distendido, introduciéndolos en una trama de aprendizaje compartida y de trabajo colaborativo.

6. Trabajos Futuros

La experiencia de uso de la herramienta nos imponen algunas l´ıneasde trabajo que se pretenden abordar.

Para simplificar la interpretación de los enunciados, cada problema podrá tener asociada una resolución parcial pre-elaborada que se ubique como dispositivo de modelado cognitivo y de andamiaje de los aprendizajes en desarrollo.

Para amortiguar la transición de bloques a texto se va a embeber dentro de la herramienta un compilador de bloques a los lenguajes de programación más utilizados en la enseñanza. Con el mismo objetivo se trabajará en la implementación de un compilador inverso, de lenguajes basados en texto a bloques.

Para mejorar la selección de la base de datos de ejercicios se trabajará sobre métricas de valoración, versionado y catalogación de problemas. Así también, para facilitar la confección de los torneos se trabajará en el concepto de Meta-torneos, que aporta beneficios en dos direcciones, por una parte permite enfocarse más en el diseño y discutir la estructura del torneo, mientras por otro lado, ofrece un modelo conceptual. Es decir el meta torneo lleva embebida la decisiones de diseño.

Para automatizar la retroalimentación, se almacenarán estadísticas que mejoren las posibilidades de comprender los procesos de enseñanza y de aprendizaje de la programación con propósitos ligados a la investigación y al mejoramiento de la práctica.

Para favorecer el vínculo y la comunicación, se trabajará en articular una Comunidad de Docentes Hornero.

Para asegurar la co-labor, entendida como la participación equilibrada de todos los miembros del equipo, se almacenarán en un repositorio versiones del código de bloques que muestren la participación individual.

Para aproximar las posibilidades de aprender y reducir el nivel de frustración de los estudiantes, se trabajará en la gestión de torneos dinámicos, que asignen problemas de acuerdo al progreso de la competencia.

Referencias

1. A. Bailey and S. Jonna. *Primefaces Theme Development*. Packt Publishing Ltd, 2015.
2. T. Cadenhead. *Socket. IO Cookbook*. Packt Publishing Ltd, 2015.
3. T. Cerny and B. Mannova. Competitive and collaborative approach towards a more effectiveeducationincomputerscience. *Contemporaryeducationaltechnology*,2(2), 2011.
4. Consejo Federal de Educación. Res 343/18. *Resoluciones CFE*, 2018.
5. D. Coore and D. Fokum. Facilitating course assessment with a competitive programming platform. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pages 449–455,2019.
6. Department for Education - GOV.UK. National curriculum in England: computing programmes of study - Gov.UK. Último acceso junio de 2020, website https: //www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study.

7. C. C. Fracchia, P. Kogan, and S. Amaro. Competir+ motivar+ hornero= aprender programación. *Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología*, (18):19–29, 2016.
8. N. Fraser. Ten things we've learned from blockly. In *2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)*, pages 49–50. IEEE, 2015.
9. I. Godoy and P. Kogan. *Desafíos para Escuela Media: Resolución de Problemas y Programación*. 2019, avalado por Resolución FaI 086/18.
10. D. JONHSON and R. JONHSON. Aprender juntos y solos. aprendizaje cooperativo, competitivo e individualista. *Aique*, 1999.
11. O. McGarr and K. Johnston. Curricular responses to computer science provision in schools: current provision and alternative possibilities. *The Curriculum Journal*, 2020.
12. Y. E. Molina-Hurtatiz, D. M. Espinosa-Sarmiento, E. F. Sánchez-Trujillo, et al. Uso de la programación competitiva como aporte a la enseñanza de la lógica algorítmica. experiencia universidad de la amazonia. *Revista científica*, 3(26):109–116, 2016.
13. D. Passey. Computer science (cs) in the compulsory education curriculum: Implications for future research. *Education and Information Technologies*, 22(2):421–443, 2017.
14. E. Pasternak, R. Fenichel, and A. N. Marshall. Tips for creating a block language with blockly. In *2017 IEEE Blocks and Beyond Workshop (B&B)*, pages 21–24. IEEE, 2017.
15. M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, and Y. Kafai. Scratch: Programming for all. *Communications of the ACM*, 52(11):60–67, Nov. 2009.
16. J. Spurlock. *Bootstrap: Responsive Web Development*. .º'Reilly Media, Inc.", 2013.
17. C. C. Trujillo, E. L. Gomez, and M. L. B. Cerdá. Eficacia del aprendizaje cooperativo en comparacion con situaciones competitivas o individuales. su aplicación en la tecnología: Una revisión sistemática/effectiveness of cooperative learning compared to competitive or individual situations and its application to technology: a systematic review/efficacité de l'apprentissage coopératif face aux situations compétitives ou individuelles. utilisation des technologies: une révision systématique. *Enseñanza & Teaching*, 30(2):81, 2012.
18. D. Weintrop. Block-based programming in computer science education. *Communications of the ACM*, 62(8):22–25, 2019.
19. D. Weintrop and U. Wilensky. Comparing block-based and text-based programming in high school computer science classrooms. *ACM Transactions on Computing Education (TOCE)*, 18(1):1–25, 2017.
20. D. Weintrop and U. Wilensky. Transitioning from introductory block-based and text-based environments to professional programming languages in high school computer science classrooms. *Computers & Education*, 142:103646, 2019.