Corrección automatizada de programas como recurso pedagógico

Federico Aloi, Franco Bulgarelli, Nahuel Palumbo, Lucas Spigariol
UNQ - federico.aloi@gmail.com
UNQ / UTN (FRBA) - flbulgarelli@gmail.com
UNSAM / UTN (FRBA) - nahuel.palumbo@gmail.com
UNSAM / UTN (FRBA y FRD) - lspigariol@gmail.com

Resumen

La educación en programación tiene su propia complejidad tecnológica: requiere del dominio de diferentes herramientas informáticas tales como lenguajes de programación, bibliotecas, entornos de desarrollo, compiladores, etc, que se suman a lo más importante que es la comprensión y utilización de herramientas conceptuales, de conocimientos y de estrategias que conforman una manera de hacer y sobre todo una manera de pensar. El dominio del aspecto conceptual y el tecnológico requieren de una práctica intensa, por lo que es deseable que el docente despliegue en el aula caminos pedagógicos adecuados, con dificultad creciente pero sin saltos abruptos, evitando que teoría y práctica vayan por vías paralelas sino que se crucen y articulen permanentemente. A su vez, complementando lo anterior, es oportuno que brinde también propuestas para que el estudiante continúe con su proceso de aprendizaje por fuera del espacio áulico. Tratándose de procesos educativos que tienen como temática el desarrollo de software, consisten generalmente en resolver ejercicios construyendo una solución mediante un programa escrito en un determinado lenguaje de programación.
Un equipo de docentes universitarios desarrolló una plataforma educativa virtual, denominada *Mumuki*, que da soporte a una dinámica de enseñanza de programación que combina práctica y teoría, y que permite ser utilizada por los estudiantes tanto en el aula como fuera de ella. Se trata de un proyecto en pleno desarrollo, que cuenta con una versión funcionando con la que se tuvieron las primeras experiencias de utilización en universidades y otras instituciones educativas en el año 2015 y que, con mejoras incluidas, continua su uso durante el 2016. El presente trabajo analiza su utilización en instituciones educativas universitarias y de nivel medio a partir de la mirada reflexiva de los docentes y los estudiantes, detectando sus principales virtudes y debilidades, de manera de orientar acerca de su uso y retroalimentar al mismo proceso de desarrollo. Es una experiencia que se la puede enmarcar en la doble confluencia entre tecnología y educación: como herramienta tecnológica para facilitar el aprendizaje, y como un recurso pedagógico para desarrollar un conocimiento tecnológico.

Palabras clave: Programación, software educativo, Mumuki, corrección, on line

Introducción

Asumiendo la opción por enseñar a programar utilizando computadoras con las que los estudiantes puedan explorar, probar y jugar en un ambiente más realista de la disciplina, y de esta manera dejando de lado el camino tradicional de la hoja, papel y prueba de escritorio, a la vez que se logran mejores resultados surgen nuevas dificultades. En este contexto, se revaloriza todo esfuerzo por encontrar recursos, estrategias y mediaciones pedagógicas que faciliten en aprendizaje de la programación. Utilizar determinadas herramientas y no otras no es simplemente una elección sobre la mejor forma de transmisión de un contenido ya predefinido, como si se tratara de una mediación neutral e inocua. Es en el marco de este ejercicio de construcción curricular que el docente lleva adelante, que adquiere sentido analizar la forma de uso de la

plataforma para aprender a programar denominada *Mumuki*, que pretende ser una mediación entre el estudiante y el conocimiento sobre programación.

Planteo del problema

Es habitual que en las asignaturas donde se enseña a programar se les presente a los estudiantes una serie de ejercicios prácticos para que realicen. Por ejemplo, asumiendo que se trata de ejercicios donde hay que utilizar un lenguaje de programación para resolverlos, podría preguntarse:

¿Son ejercicios para hacer en papel o sobre la máquina? Si es en papel ¿cómo sabe el estudiante si es correcta su solución? Si es en computadora, ¿puede probar el ejercicio? ¿que el programa funcione es equivalente a que esté correctamente resuelto?

¿Son ejercicios que se hacen en el tiempo de clase, con el docente presente? ¿Se utiliza una dinámica de taller, donde cada uno avanza a su ritmo mientras el docente asiste? ¿Alcanza a responder todas las consultas individuales y hacer el seguimiento de cada estudiante?

¿Son ejercicios para hacer fuera del horario de clase? ¿El alumno realiza solo -o con otros estudiantes- los ejercicios, luego los entrega y tiempo después recibe una devolución? ¿Qué hace si tiene dificultades que no lo dejan avanzar?

¿Los ejercicios se plantean para practicar conceptos ya explicados previamente de manera teórica? ¿Son ejercicios para que cada estudiante practique opcionalmente o hay un compromiso de entrega y corrección?

La computadora como medio y obstáculo

Desde el punto de vista de quien ya sabe programar, la computadora no sólo es el soporte obvio donde el programa se va a ejecutar, sino que provee herramientas que facilitan la escritura del código y sobre todo ayudan a su puesta a punto mediante pruebas. Sin embargo, para quien está comenzando su aprendizaje se puede volver un obstáculo más. El hecho de enfrentarse de pronto a la computadora y deber no sólo entender sino también expresarse en ese lenguaje formal, con toda la complejidad de su sintaxis y lo abstracto de su simbología, manipular entornos de desarrollo con múltiples opciones, poder interpretar y depurar errores, realizar seguimientos, validar resultados y todo lo que implica lograr que un programa funcione y lo haga correctamente, puede tornarse frustrante, si no se utilizan las mediaciones pedagógicas apropiadas. Los lenguajes de programación que son de uso profesional en la industria del software están pensados para optimizar la productividad del profesional que sabe programar y no para quien está empezando a dar sus primeros pasos. Esto se nota de muchas maneras, pero en especial en el *feedback* que arroja frente a los errores: muchas veces, el mensaje de error es incomprensible para el estudiante ya que se explica a partir de conceptos de programación que no maneja u otros elementos del lenguaje qué aún no se le explicaron.

¿Cómo se si mi solución es correcta?

Un estudiante puede construir una solución y considerar que es correcta sin siquiera probarla o habiéndolo hecho con un juego de datos muy particular que no representa la variedad de situaciones a las que la solución debe dar respuesta adecuadamente. Cabe señalar que, sobre todo en los ejercicios más sencillos, cuando el estudiante se está iniciando en una nueva herramienta, el armado del caso de uso y todo el contexto en el cual la solución puede ser probada es tanto o más complejo que la solución en sí. Por otra parte, hay muchas veces donde el estudiante no logra que su solución funcione y por lo tanto no puede probarla, pero sin embargo, su planteo es correcto y con sólo ajustar cuestiones menores o detalles sintácticos se puede terminar de construir el programa correctamente. Esta dificultad para concluir un ejercicio suele

generar frustración en el estudiante e impedirle continuar avanzando con otros ejercicios.

Seguimiento del estudiante fuera del aula

Una situación frecuente de un alumno es no llegar a completar la práctica propuesta y no necesariamente por falta de dedicación o por factores externos determinantes, como podría sugerir una mirada docente simplista. Muchas veces tiene que ver con que el estudiante se encuentra solo frente al problema y no encuentra las mediaciones adecuadas, no sabe cómo empezar o se traba en algún punto y no puede continuar. Cuando logra realizar los ejercicios prácticos, no pueden probar si sus soluciones son correctas. Puede suceder que resuelva el ejercicio de manera equivocada creyendo que lo hizo correctamente y afiance una idea errónea.

Teoría y práctica

Realizar ejercicios aplicando un tema visto anteriormente en forma teórica es la forma clásica de articular teoría y práctica, pero ciertamente no es la única. Otra forma consiste en resolver ejercicios prácticos apelando a la experiencia, al sentido común o a conocimientos previos y luego conceptualizar recuperando lo realizado. También, un recurso válido es plantear un problema siendo consciente de que el estudiante no está en condiciones de completarlo, pero sí de avanzar hasta un punto donde se encuentra con un obstáculo que por sí mismo no puede superar, y recién en ese momento explicar el nuevo concepto que permite la resolución. Así como un día de lluvia es el mejor para vender paraguas, la necesidad práctica le permite al aprendiz interpretar el sentido y la utilidad de un concepto teórico.

La clase como taller

Quienes valoran pedagógicamente un esquema más participativo y que recupere el protagonismo del estudiante, suelen plantear una dinámica de taller o de clase invertida. En ella, el docente se descentra de su rol típico de dirigir la clase y asume una actitud de acompañamiento y seguimiento de los estudiantes mientras ellos trabajan con mayor autonomía en la solución de un problema, generalmente mediante el desarrollo de un programa. Las dificultades que se les presenta a los estudiantes probablemente son las mismas ya mencionadas, pero la diferencia es que cuentan con el docente a quien pueden recurrir. La cuestión es que las dificultades en los estudiantes surgen simultáneamente y teniendo en cuenta que -excepto alguna situación muy particular- la cantidad de estudiantes es ampliamente mayor que la de docentes, no es extraño que queden muchas consultas sin responder.

Fundamentos teóricos

Dentro de las principales corrientes pedagógicas, un aspecto recuperado por la pedagogía crítica con suma importancia es el ida y vuelta vital entre práctica y teoría, en una dinámica mutuamente enriquecedora. Paulo Freire critica la dicotomía entre teoría y práctica y recomienda evitar todo tipo de propuesta "que menospreciase la teoría, negándole toda importancia y enfatizando exclusivamente la práctica como la única valedera o bien negase la práctica atendiendo exclusivamente a la teoría"[1]. A su vez, la teoría construccionista propuesta por Seymour Papert[2], creador del lenguaje de programación "Logo" -experiencia emblemática del uso de tecnología educativa-, entiende al espacio educativo como una instancia de creación en la que teoría y práctica confluyen de manera tal que es imposible –o pierde sentido- diferenciarlas. También en los ambientes universitarios, generalmente tendientes a focalizar en conceptos teóricos, tiene sentido articular teoría y práctica. En esta dirección, un trabajo realizado por la Universidad Tecnológica Nacional propone "una nueva relación donde la práctica deja de ser la mera aplicación de la

teoría para convertirse en fuente del conocimiento teórico" [3].

La clase invertida

Otro aporte fundamental sobre la dinámica de clase y el aprovechamiento de la presencia del docente en relación a otras mediaciones, es la que desarrollan Bergmann y Sams. Desde su experiencia docente personal, luego formalizada como *Flipped Classroom*, afirman: "El momento en que los alumnos necesitan que esté físicamente presente con ellos es cuando se atascan en un tema y necesitan mi ayuda personal. No me necesitan en el aula con ellos para darles contenidos; los contenidos los pueden recibir por su cuenta" [4].

La mediación

La experiencia relatada por Jacques Ranciere en el "Maestro Ignorante"[5] -donde la utilización de un libro dado por el docente a sus estudiantes les permite aprender más allá del ambiente escolar- puede ayudar a redescubrir la importancia y variedad de mediaciones dentro del complejo tejido de la práctica docente. En particular, ejemplifica la ruptura de la correlación lineal entre la explicación del profesor y la comprensión del alumno y el descubrimiento de nuevas mediaciones, que en el caso mencionado es un libro y en la presente investigación es un software, como facilitadoras del proceso de aprendizaje. Cuando desde su mirada constructivista del aprendizaje Vigotsky[6] plantea la *Zona de Desarrollo Próximo* como aquellos conceptos o habilidades que el estudiante no conoce pero que están a su alcance de ser aprendidos con las mediaciones adecuadas, está reconociendo el rol central del docente y a la vez está abriendo el juego a una variedad de recursos que facilitan el aprendizaje. En la actualidad, la posibilidad de utilizar herramientas tecnológicas con un fin de mediación dentro del rol docente va en clara sintonía con estas ideas.

El error dentro del proceso de aprendizaje

Asumiendo que el aprendizaje es un proceso de construcción, los errores no son vistos como fracaso sino como oportunidad de aprendizaje, por lo que es fundamental descubrirlos, asumirlos e interpretarlos dentro de una propuesta pedagógica.

En el campo de la programación, la detección e interpretación de los errores se vuelve un concepto fundamental. Poder validar que un programa funciona correctamente es un desafío permanente y presenta diferentes niveles de abordaje. Desde los errores que se producen en tiempo de desarrollo, impidiendo la ejecución del programa, y que se expresan en un lenguaje de difícil comprensión para un estudiante, hasta los que se producen en tiempo de ejecución y hacen que la aplicación se detenga inesperadamente sin manifestar detalles, poder detectarlos con precisión y entender sus causas es fundamental para resolverlos adecuadamente. A estos errores típicos se suman una serie de situaciones que los lenguajes de programación como tales no detectan, sino que son vistos como problemáticos desde la mirada docente y profesional y que tienen que ver con criterios de programación y diseño de sistemas, con buenas prácticas y recomendaciones para desarrollar software de calidad.

Mumuki, una plataforma virtual para aprender a programar

Frente al problema que representa la primera aproximación a un entorno de desarrollo de software, puede haber muchos enfoques posibles: hay docentes que optan por realizar prácticas sobre papel y otros prefieren utilizar directamente software de uso profesional.

Otro camino posible es el que transita el presente trabajo: la utilización de software especialmente diseñado para dar los primeros pasos en la programación, que introduzca desde un primer momento en el uso de la computadora y el lenguaje, pero con mayor sencillez y ayuda que una herramienta

profesional de desarrollo. Particularmente, se analiza el uso como recurso didáctico una plataforma virtual on-line denominada *Mumuki*, desarrollada por un grupo de docentes e investigadores universitarios. Es un software educativo para aprender a programar a partir de explicaciones y resolución de problemas; propone aprender conceptos de programación en un recorrido conformado por guías de ejercicios donde se combina teoría y práctica Esta herramienta se presenta al estudiante como una aplicación web interactiva, en la que se articulan explicaciones y ejemplos con el foco puesto en que cada uno realice su solución y sea probada y corregida instantáneamente, orientando acerca de los aciertos y errores.

El ejercicio y su corrección

Lo que se le presenta al estudiante al ingresar a la plataforma -previa identificación personal que lo asocia a una cuenta para garantizar su privacidad y lo ubica en un curso vinculado a su docente- es básicamente una gran cantidad de problemas concretos, organizados y articulados entre sí, que el estudiante debe resolver mediante el desarrollo de un programa en un determinado lenguaje.

Al elegir un ejercicio, el alumno se encuentra con una explicación donde se enuncia el problema a resolver, con ejemplos y orientaciones iniciales, y se habilita un panel donde puede escribir el código de la solución. Según como se haya diseñado el problema, puede que disponga de ayuda adicional -que al solicitarla se muestra de inmediato-, que haya alguna parte del problema ya resuelta y deba concentrase en lo faltante u otras variantes. Cuando el estudiante termina de plantear su solución, con sólo presionar un botón la "envía" y la plataforma la ejecuta y evalúa, informando si es correcta o no. En caso que presente errores, se indica cuáles son y el lugar del código donde se produjeron, no sólo con la información que típicamente arroja un compilador -orientado a programadores experimentados- sino con precisiones y orientaciones adicionales propias de la forma de presentar la materia por parte de los docentes, y se da la posibilidad que el alumno corrija su solución y la vuelva a enviar. También se despliega una consola, donde el mismo alumno puede ir haciendo sus propias pruebas y viendo cómo va funcionando su programa.

A medida que se avanza en la resolución de los ejercicios se van poniendo en evidencia los elementos conceptuales relacionados que dan sustento a la práctica y a la vez permiten pasar a nuevos ejercicios de mayor complejidad.

Organización de los contenidos

Los ejercicios se organizan en guías, en función de los temas que se abordan. Mantienen un hilo conductor y están pensadas para una resolución progresiva. Los ejercicios que se van resolviendo dan pie a conclusiones y nuevos ejercicios. Dentro de las guías hay diferentes estilos de presentar el contenido, lo que apunta a que pueda ser usado en distintos momentos del proceso de aprendizaje del alumno o enmarcado de acuerdo a la propuesta pedagógica de cada docente. Entre ellos se destacan las guías de aprendizaje, orientadas a presentar conceptos mediante la resolución de problemas simples, y Guías de desafío, con problemas de complejidad creciente orientados a ganar agilidad.

Si bien hay una secuencia pensada previamente por el docente, es el estudiante quien decide la forma de uso. Con el fin de permitir recorridos diferentes y por otra parte minimizar la frustración de un ejercicio que no puede ser finalizado, la plataforma no requiere haber terminado correctamente un ejercicio para pasar al siguiente. Esta flexibilidad permite también que diferentes docentes planteen sus propios recorridos conceptuales teniendo como base el mismo libro de guías.

Desde el punto de vista del estudiante se percibe a la plataforma como una unidad, pero con una mirada analítica se distinguen claramente dos elementos. Por un lado está el software en sí, con las prestaciones concretas

que provee en cuanto a la lógica de funcionamiento, la usabilidad de su interfaz gráfica, las posibilidades de configuración y la forma en que organiza la información. Por otra parte, se encuentra el contenido que la plataforma administra, con los ejercicios organizados en guías y libros. Uniendo ambos aspectos, se suma un marco de lineamientos donde se recomienda cómo poder aprovechar las diferentes variantes que provee la plataforma a la hora de generar el contenido.

Características

Mumuki tiene elementos innovadores comparado con otras plataformas de enseñanza de la programación, ya sea de entidades comerciales o de instituciones educativas. En primer lugar, su contenido se encuentra íntegramente en español, no siendo una traducción, sino contenido original y local. Se hace énfasis en emplear una redacción informal, orientada a la pregunta, usando recursos gráficos como *emoticones* e imágenes, y basada en el *voseo* característico de Argentina. Esto permite establecer una cercanía entre el idioma de los problemas y el idioma del alumno y al mismo tiempo otorgarle un tinte nacional al contenido. En muchos casos, en vez de una mera traducción del error que arroja un compilador, se pasa a una explicación detallada del problema, no orientada a relatar el error evidente, sino al origen conceptual más probable del mismo. Por ejemplo, en lugar de indicar "uso de identificador no declarado" la plataforma reportará "Estás usando un predicado, pero no lo declaraste antes. Fijate si no te olvidaste de declararlo o si escribiste mal su nombre".

Por otro lado, *Mumuki Atheneum*, plataforma de ejercitación y corrección y pieza software principal, es en su totalidad código libre y gratuito, con el objetivo de facilitar la colaboración entre docentes de facultades y entusiastas en general. La libertad de las herramientas y materiales de estudio es fundamental para la democratización del conocimiento.

Herramientas para seguimiento

En forma complementaria, hay una serie de opciones que están pensadas para los docentes. Entre ellas, cabe destacar la herramienta para el seguimiento de los estudiantes de un grupo, *Mumuki Classroom*, con la que el docente puede ver el avance de cada alumno en la resolución de ejercicios propuestos. Permite ver al docente de manera organizada y clara la información sobre el progreso de sus propios estudiantes, llegando al detalle de no solo ver el código correcto de la solución que fue validado por *Mumuki*, sino también todos los intentos intermedios que fue probando

Gestión del contenido

Una herramienta clave que incluye *Mumuki* es el editor de contenidos, que le permite al docente redactar los enunciados de los ejercicios, presentar las ayudas típicas y explicitar las expectativas de aplicación de conceptos en la resolución del problema. Esta característica es fundamental para permitir la utilización de la plataforma en diferentes ámbitos educativos e independizar la tarea de creación de contenido de la que significa el mantenimiento de la plataforma como tal.

Experiencias de uso

Se analiza la forma de utilización de la plataforma en las diferentes instituciones educativas donde se está utilizando *Mumuki*, desde el año 2015 a la actualidad. Junto a varias que son de carácter universitario, se incluye también una experiencia en nivel medio, en un colegio técnico en programación.

En el ambiente universitario, se lo utiliza en la asignatura *Paradigmas de Programación*, de la carrera de Ingeniería en Sistemas de Información, en el ámbito de la Universidad Tecnológica Nacional, tanto en la Facultad Regional Buenos Aires como en la Facultad Regional Delta, como en la materia *Paradigmas de Programación* de la

Universidad Nacional de San Martín, dentro de la Tecnicatura Universitaria en Programación Informática. En todas ellas, se se abordan las temáticas de la programación funcional y lógica, utilizando respectivamente los lenguajes *Haskell* y *Prolog*.

La mirada de los docentes

Generalmente, *Mumuki* fue usado en el aula como complemento a las explicaciones de nuevos temas teóricos. A medida que los docentes introducían nuevos conceptos a la clase, estos eran acompañados con la resolución de ejercicios una guía de aprendizaje, para que cada paso de aprendizaje sea respaldado por una aplicación concreta de los problemas que soluciona. A su vez, como las guías de aprendizaje también presentan los conceptos de manera gradual, permitió que los docentes continúen con el desarrollo del tema dentro de una dinámica de ida y vuelta entre explicaciones teóricas y prácticas dentro del aula. Al final de las clases se recomendaban algunos ejercicios de las guías trabajadas en clase tanto para aquellos que no pudieron asistir a clase como los que terminaron con dudas. En muchos cursos, las clases tenían asociadas guías o ejercicios prácticos para que los estudiantes resolvieran solos en la casa. Estos sirvieron para afianzar los conceptos de la clase ante problemas de complejidad creciente, abarcando algún tema en profundidad y mostrando casos particulares que, debido al tiempo limitado de clase, no se pudieron abordar en el aula. En algunas ocasiones se dio como tarea ejercicios de aprendizaje sobre conceptos que se verían la clase siguiente. De esa manera, los estudiantes tuvieron un primer acercamiento al tema nuevo a través de una serie de ejercicios guiados. Algunos estudiantes los pudieron resolver y entender sin problemas, mientras muchos otras trajeron dudas a la clase, ya sea porque no pudieron resolver los problemas planteados o porque no terminaban de comprender lo que estaba pasando.

La refocalización del tiempo del equipo docente, sumado a la capacidad de monitoreo de estudiantes que ofrece la herramienta para docentes, permitió enfocarse en aquellos estudiantes que veían trabados en un mismo ejercicio o que habían podido resolver pocos o ninguno de ellos, ayudándolos a través de mensajes enviados por la misma herramienta. Uno de los docentes cuenta el caso emblemático de un estudiante. “Tenía un alumno que había avanzado bastante en la solución, pero no había recibido ninguna respuesta positiva por parte de *Mumuki* hasta entonces, ya que su solución tenía pequeños errores. Así que dicho alumno decidió empezar de nuevo tratando de ir avanzando a medida que *Mumuki* respondía positivamente. Pude darme cuenta a tiempo, gracias a disponer del historial de soluciones, y rescaté dicha solución indicándole dónde estaban los errores, sumado a los conceptos que había detrás. Un par de horas más tarde el alumno pudo pasar todas las pruebas de *Mumuki* para dicho ejercicio y me mandó dudas respecto al ejercicio que continuaba”.

Otra realidad que se percibe teniendo en cuenta las observaciones de los docentes de los diferentes cursos, es que la mayoría de los estudiantes en algún momento de la cursada de la materia se sienten lo suficientemente seguros como para dejar de usar *Mumuki* y pasar a utilizar el entorno profesional del lenguaje de programación, o por cierto tiempo los utilizan simultáneamente.

Estadísticas de uso

Analizando estadísticas de la base de datos de soluciones subidas por los estudiantes, surge que fue usado de maneras variadas. Al momento, hay registradas más de unas 120000 soluciones y más de 500 usuarios, lo cual arroja un promedio de 200 soluciones por cada uno, de las cuales aproximadamente el 30% fue correcto, lo que sugiere que la solución correcta corresponde en promedio al tercer intento. Ciertamente, estos promedios son indicadores relativos, ya que hay ejercicios

que la amplia mayoría lo resuelve correctamente al primer intento, y hay algunos ejercicios donde las soluciones correctas son muy pocas.

La experiencia de los estudiantes

Para complementar la mirada de los docentes respecto del uso de la plataforma se recurrió a la percepción de los mismos estudiantes sobre su proceso de aprendizaje, lo que metodológicamente se implementó mediante una encuesta.

Antes de abordar específicamente el uso de *Mumuki*, se preguntó en forma general sobre las características de un software educativo que serían las más importantes a su criterio. Eligiendo entre opciones preestablecidas, lo más ponderado por los estudiantes -con amplia distancia frente a las siguientes- fue "que permita comprender claramente los conceptos teóricos". Esta primera apreciación es coincidente con una de las ideas básicas que fundamentan este trabajo en relación a la necesaria articulación entre teoría y práctica.

El principal aspecto favorable que encuentran los estudiantes en *Mumuki* es la respuesta que da frente a los errores, destacado por más del 60%. Sin ser una pregunta con opciones, hay coincidencias en este aspecto, con apreciaciones que destacan "que indique errores con precisión", "la corrección al momento", "que se pueda autocorregir y que haya ayudas y explicaciones para encarar los ejercicios", por citar algunas.

Otros aspectos valorados como lo mejor de la herramienta son la gradualidad y variedad de ejercicios. "que sean ejercicios cortos que iban aumentando la dificultad", "sirve como primer paso antes de encontrarse con la programación pura que es más complicada".

También hay un grupo de estudiantes que destacan la articulación con la teoría. Por ejemplo, "lo mejor es la forma de explicar", como también "permitió comprender claramente los conceptos teóricos". Un estudiante explica que "es muy útil poder ver como lo que se ve en clase es aplicable a una herramienta y no quede sólo como teoría".

Otro grupo rescata la interfaz gráfica y la interactividad, señalando "amigabilidad de la interfaz" y "tener la consola a mano y que se guarde el progreso". Con mayor énfasis, un alumno agrega "me parece una excelente iniciativa ya que la plataforma es muy amigable, te incita a seguir resolviendo cada ejercicio porque sus enunciados claros y correlativos". En particular, es interesante la opinión de un estudiante, que afirma "es mucho mejor que hacer ejercicios y 'creer' que lo que se hizo está bien".

Por otra parte, los estudiantes plantean dificultades. Lo hacen en pequeña proporción en la apreciación general de la herramienta, pero ante la pregunta concreta por problemas, se encuentran más respuestas. También se consultó a los estudiantes por sugerencias para mejorar, y allí hubo mayor cantidad de comentarios interesantes para recuperar.

Hay un grupo que dice haber utilizado poco o nada la herramienta y que no da detalles acerca de los motivos. Algunos aclaran que pese a eso ven favorablemente su uso, como uno de ellos que afirma "me parece bien la utilización de esta herramienta aunque a mí no me ayudó en nada". Algunas pocas opiniones son claras en cuanto plantean directamente que no lo ven como una ayuda. Por ejemplo, se afirma que "me parece mejor utilizar directamente software de uso comercial". Otros plantean problemas puntuales que muestran que hubo una utilización intensa del software. Entre ellos, hay coincidencias en señalar como poco entendibles ciertos mensajes de error y varias sugerencias de mejorar la claridad de las explicaciones y consignas. Un comentario crítico da cuenta que "*Mumuki* espera respuestas muy específicas, le falta flexibilidad". Y en tono de sugerencia, otra persona agrega "se podría mejorar detección de errores, contemplando soluciones alternativas".

Además de reiterados pedidos de mayor cantidad y variedad de ejercicios, se

mencionan variadas ideas, como por ejemplo la inclusión de ejercicios resueltos, donde se pueda ver cómo está construida la solución.

Analizando las respuestas como un conjunto se percibe entre líneas que las sugerencia y problemas muestran que hay interés, que vale la pena seguir mejorando la herramienta porque resulta útil. Uno de ellos los expresa como dando aliento para continuar con tarea "¡A seguir trabajando, que es una plataforma muy interesante!"

En la escuela secundaria

Una mención aparte amerita la experiencia de utilización de *Mumuki* en el nivel medio. Se trata de la materia *Laboratorio de Programación 2*, dictada en el marco de la *Tecnicatura en Programación con Orientación en Desarrollo de Software Libre* del Instituto Secundario Sagrado Corazón Al Cal, ubicado en Villa Jardín, Lanús Oeste, Provincia de Buenos Aires. Se trata de una materia donde el objetivo es trabajar con programación con bloques, utilizando el lenguaje *Gobstones* [7].

Durante el 2015, primer año de la utilización de la herramienta, los estudiantes contaban con cierta experiencia en programación, pero tenían prácticamente nula capacidad de abstracción y difícilmente podían extrapolar los conceptos aprendidos para utilizarlos en distintas situaciones. Sumado a esto, no estaban acostumbrados a escribir correos electrónicos, de modo que las horas de clase constituían el único espacio de consulta. Al ser una materia nueva, tampoco existía material de consulta adecuado para la edad y los conocimientos. Por otra parte, las ausencias a clase fueron bastante comunes lo cual volvía difícil el seguimiento de la materia.

Para quienes venían siguiendo el curso, la resolución de las guías sirvió de repaso y práctica, mientras que para el resto se transformó en una herramienta fundamental en la incorporación de nuevos conceptos. La introducción de *Mumuki* logró que todos los estudiantes practicaran fuera del horario de la clase, dato no menor teniendo en cuenta el bajo nivel de involucramiento con las materias que suele darse en la escuela media. Todos los estudiantes pudieron acceder a la plataforma y muchos de ellos se mostraron mucho más entusiasmados que con la herramienta de escritorio.

Desde la percepción de los mismos estudiantes surgieron tres cosas fundamentales: la importancia que se le da a la explicación de los errores, el lenguaje coloquial utilizado en las explicaciones de los ejercicios y el hecho de no tener que instalar ningún software adicional para poder practicar.

En lo que va del 2016, la herramienta se viene utilizando como complemento para afianzar conceptos teóricos y formalizar cuestiones que no se llegan a discutir en clase. El nivel de aceptación por parte de los estudiantes es incluso superior al del pasado año.

Hay un aspecto más a destacar que está relacionado con el su potencial inclusivo. Aproximadamente un 25% de los estudiantes del curso no cuentan con una computadora en su casa con la cual practicar y una proporción similar no cuenta con acceso fijo a Internet. Sin embargo, si se tiene en cuenta a dispositivos inteligentes *smartphones* o *tablets* y acceso a Internet por redes móviles, estos porcentajes crecen considerablemente, alcanzando casi a la totalidad de los estudiantes.

Es en este contexto que la herramienta *Mumuki* se convierte en posibilitadora: permite que aquellos que no hubieran podido practicar, por carecer de una computadora, puedan hacerlo desde su celular.

Conclusiones

Poniendo foco en el desafío inicial de abordar el carácter mediador de *Mumuki* como herramienta pedagógica para facilitar el proceso de aprendizaje, y teniendo en cuenta la descripción de la plataforma y el análisis de sus primeras experiencias de utilización, se puede afirmar que, aún con limitaciones

propias de su etapa de desarrollo, ya presenta resultados favorables y alentadores.

Del entrecruzamiento de las percepciones de docentes y estudiantes -los protagonistas del proceso educativo- sumado a un análisis de la herramienta como tal, surgen pistas que permiten ratificar algunas características, orientan para reformular otras y de esta manera retroalimentar el trabajo de desarrollo del software y la generación de contenidos. El enfoque metodológico del presente trabajo no busca cuantificar la magnitud de este aporte, sino describir y ayudar a descubrir las variables que constituyen el proceso en el que adquiere sentido su utilización como recurso pedagógico.

Incentivación de la ejercitación

Se constató un gran involucramiento de los estudiantes en la práctica, lo cual es reconocido por los mismos alumnos y confirmado por el volumen de soluciones subidas. La valoración de la posibilidad de una respuesta entendible e inmediata y la valoración positiva que se hace sobre la interfaz y su usabilidad, van en el mismo sentido. Incluso algunos problemas planteados y las sugerencias acerca de mayor claridad y más cantidad de ejercicios, ratifican la importancia del rumbo emprendido. En palabras de un alumno, “me ayudaron a abrir mi mente y pensar las soluciones de otra manera”.

Articulación entre teoría y práctica

La expectativa de que *Mumuki* permita una mayor articulación entre teoría y práctica, se puede ver en diferentes planos. El planteo de poder practicar los conceptos teóricos explicados en clase, generalmente mediante guías de desafío, es el más consolidado: Los alumnos destacan claramente que su mayor lugar de utilización es en sus casas y destacan más la posibilidad de ejercitar y probar sus soluciones que de utilizar la plataforma para aprender nuevos conceptos. La utilización en tiempo de clase como recurso para descubrir los conceptos a partir de la práctica es una apuesta de los desarrolladores que va calando en los equipos docentes, pero aún necesita maduración.

Creación de contenido

La diferenciación entre la plataforma en sí y el contenido de ejercitación -que para el estudiante es transparente ya que se presenta como un todo- permite la identificación de roles que confirman el carácter dinámico del proyecto y posibilitan su crecimiento. Los desarrolladores del software de la plataforma, más allá que también sean docentes, pueden seguir trabajando en la mejora continua del producto de manera autónoma. A su vez, todo docente que quiera utilizar la plataforma para sus clases no necesita involucrarse con el desarrollo sino concentrase en la generación de contenido acorde a sus opciones.

Seguimiento

Disponer de una herramienta que puede ser accedida en cualquier momento logró que el alumno pueda sentirse más contenido. La inclusión de herramientas de seguimiento docente y la posibilidad de dar *feedback* desde la misma herramienta, también ayudó a fortalecer el acompañamiento del docente.

Investigación, desarrollo y docencia

En otro plano de análisis, otro aspecto que la experiencia deja ver es lo fecundo de la confluencia entre la vocación docente, la iniciativa para el desarrollo de software y la actitud investigadora. A modo de cierre -del presente artículo-, pero con la intención de seguir andando y abriendo caminos, mejor que lo que podría decir cualquier docente es la afirmación entusiasmada de un estudiante: “Es motivador saber que se hacen cosas así en la facultad, dan ganas de seguir con la carrera y de arrancar proyectos”.

Trabajo futuro

Focalizando en la plataforma, recogiendo las inquietudes de docentes y alumnos surgen diversas líneas de acción para continuar modificando y enriqueciendo *Mumuki*, como por ejemplo seguir introduciendo herramientas de seguimiento y variantes en la formulación de guías y ejercicios. En relación al contenido, es permanente el desafío de seguir generando nuevos ejercicios y revisando los actuales para proponer recorridos más significativos. Otra línea de trabajo pendiente es sistematizar y formalizar en una secuencia didáctica la forma de uso sugerida de la herramienta, para que sirva de referencia a otros docentes.

Aplicación en otras áreas

Teniendo en cuenta que la arquitectura de *Mumuki* diferencia lo que son las opciones y facilidades de la plataforma como tal de los contenidos propiamente dichos, resulta una herramienta lo suficientemente genérica y versátil para permitir la confección de apuntes interactivos, guías de ejercitación, prácticas, desafíos, juegos o incluso exámenes sobre otras temáticas: el requerimiento es que la información se estructure en torno a problemas con soluciones verificables. Más allá de que la plataforma fue concebida en primer lugar para ayudar en la enseñanza de la programación, el tipo de problema tampoco está prefijado: ejercicios de matemática, física, ajedrez, son otros distintos escenarios posibles de aplicación de la herramienta.

Referencias

[1] Freire, Paulo, *Carta a quien pretende enseñar,* Siglo Veintiuno, Buenos Aires, 1994.

[2] Papert, Seymour, *Logo Philosophy and Implementation,* LCSI, 1999.

[3] Secretaría Académica y de Planeamiento (UTN), *Didáctica en la Universidad,* Universidad Tecnológica Nacional, Buenos Aires, 2007.

[4] Bergmann, Jonathan; Sams, Aaron, *Dale la vuelta a tu clase*, Innovación educativa, España, 2014

[5] Ranciere, Jacques, *El maestro ignorante*, Laertes, Barcelona, 2002.

[6] Vigotsky, Lev. *Pensamiento y lenguaje,* Visor, Madrid, 1993

[7] Martínez López, Pablo E., *Las bases conceptuales de la Programación: Una nueva forma de aprender a programar.* La Plata, Buenos Aires, Argentina, 2013.