Desarrollo de Documentos Vivos y Experimentación con el Mundo Físico como Estrategias para la Retención y la Atracción de Estudiantes de Pregrado en los Cursos de Programación

Development of Living Documents and Experimentation with the Physical World as Strategies for the Retention and Attraction of Undergraduate Students in Programming Courses

Jairo Enrique Serrano Castañeda
Facultad de Ingeniería
Universidad Tecnológica de Bolívar
Cartagena de Indias, Colombia
jserrano@utb.edu.co

Juan Carlos Mantilla Gómez
Facultad de Ingeniería
Universidad Tecnológica de Bolívar
Cartagena de Indias, Colombia
jmantilla@utb.edu.co

Isaac Zuñiga Silgado
Facultad de Ingeniería
Universidad Tecnológica de Bolívar
Cartagena de Indias, Colombia
izuniga@utb.edu.co

Yuranis Henriquez Nuñez
Facultad de Ingeniería
Universidad Tecnológica de Bolívar
Cartagena de Indias, Colombia
yhenriquez@utb.edu.co

Juan Carlos Martinez-Santos
Facultad de Ingeniería
Universidad Tecnológica de Bolívar
Cartagena de Indias, Colombia
jcmartinezs@utb.edu.co

Gloria Isabel Bautista Lasprilla
Facultad de Ingeniería
Universidad Tecnológica de Bolívar
Cartagena de Indias, Colombia
gbautista@utb.edu.co

Resumen– ***Este artículo describe las estrategias pedagógicas que fueron planeadas al interior del programa de Ingeniería de Sistemas y Computación adjunto a la Facultad de Ingeniería de la Universidad --- antes de la pandemia, para captar la atención de nuevos estudiantes y mejorar la retención de estudiantes activos. En el caso de atracción de estudiantes nuevos, se crea una actividad promocional basado en las iniciativas relacionadas a Ciencia, Tecnología, Ingeniería y Matemáticas (STEM por sus siglas en inglés), que se presenta en colegios focalizados a partir de eventos académicos y concursos. En el caso de la retención, se rediseñaron aulas digitales a partir de la construcción personalizada de materiales didácticos usados en los cursos fundamentales de ciencias computacionales en los programas de ingeniería y a corto plazo sería un beneficio colateral para la continuidad de operaciones en la época de pandemia. Con estas medidas, el programa logró reducir 7% la deserción intersemestral y aumentar en un 77% en el número de estudiantes en una ventana de cinco años.***

Palabras Claves-- Enseñanza, Programación, Python, Ciencias computacionales, Ingeniería.

Abstract*: This article describes the pedagogical strategies planned within the Computer and Systems Engineering program attached to the Faculty of Engineering of the University --- before the COVID-19 pandemic, to attract the attention of new students and improve the retention of active students. In attracting new students, promotional activity is created based on Science, Technology, Engineering and Mathematics (STEM) initiatives, which target schools through academic events and competitions. In retention, we redesigned digital classrooms from the personalized construction of didactic materials used in the fundamental courses of computer science in engineering programs. In the short term, it would be a collateral benefit for the continuity of operations in the time of the pandemic. With these measures, the program reduced inter-monthly dropouts by 7% and increased the number of students by 77% in a five-year window.*

Keywords-- Teaching, Programming, Python, Computer Science, Engineering.

Digital Object Identifier (DOI):
http://dx.doi.org/10.18687/LACCEI2021.1.1.381
ISBN: 978-958-52071-8-9 ISSN: 2414-6390

I. Introducción

Teniendo en cuenta que las cifras de ingreso en estas carreras a nivel general han disminuido con el paso de los años [1], existen iniciativas de organismos como el Ministerio de Tecnologías de la Información y Comunicaciones de Colombia - MINTIC - que propician espacios para la implementación de proyectos en pro de la atracción de nuevos estudiantes para carreras en el área de

la informática y computación [2]. En algunos casos se busca incentivar a un sector especifico. Un ejemplo son las propuestas de W-STEM que invitan directamente a que las mujeres tomen estudios superiores en áreas relacionadas con la ingeniería y la tecnología, por ejemplo, el Global Women in STEM Leadership Summit [3], y las iniciativas de IEEE [4] y ACM [5], lideres a nivel mundial.

En el caso de la Universidad ---, se tiene un programa de Ingeniería en Sistemas y Computación donde el 90% de la población proviene de los estratos socioeconómicos bajos, lo cual implica que los estudiantes no tienen en general los recursos para adquirir elementos adicionales a los requeridos en las clases. Por lo tanto, se trabaja en iniciativas dentro de la universidad que propendan al aprovechamiento de recursos propios, generando espacios que incentiven las capacidades de los estudiantes y aporten al mejoramiento de su entorno. Además, en la búsqueda de motivar a nuevos jóvenes, se apuesta por estrategias académicas de tipo promocional que los incluya desde edades tempranas, antes de su ingreso a la educación superior.

En el caso de la deserción estudiantil, el programa de Ingeniería de Sistemas y Computación de la Universidad --- para el año 2013 tenía una deserción del 19% semestral, cifra que contribuye a la problemática que se viene dando en los últimos años, por la falta de egresados en estas carreras, en la región y el país [6].

Este artículo cubre el trabajo realizado por el programa antes de la pandemia, insumo importante que ayudó al programa a estar preparado a una transición más suave en épocas de virtualidad completa.

Uno de los principales retos al interior de un programa es incentivar y atraer nuevos estudiantes. Además, se tiene el reto de disminuir la deserción utilizando estrategias pedagógicas que aseguren la calidad de nuestros egresados. A continuación, se mostrará como evolucionó la forma de impartir los contenidos en la Sección II. La forma como se evolucionó la idea original, desarrollo y evaluación en la Sección III. Como se llevan los contenidos al mundo real en la Sección IV. En la Sección V se presenta el impacto de las acciones tomadas en los cursos iniciales en los cursos de final de carrera. Finalmente, la Sección VI los resultados alcanzados y en la Sección VII se presentan las conclusiones más relevantes.

II. REDISEÑO CURRICULAR

La transformación del programa de Ingeniería de Sistemas y Computación inicia en el año 2013 con el rediseño de los cursos de Fundamentos de Programación y Programación I. Estos cursos pertenecen al área de Ciencias Computacionales y son tomados por todos los estudiantes de la Facultad de Ingeniería en su primer año. Estos cursos cuentan con una organización estandarizada, 16 semanas de clase, tres de ellas dedicadas a evaluaciones y 13 a presentación de contenidos y ejecución de actividades, como laboratorios y talleres.

Lo que motivo el cambio fue la reducción de las horas directas de cátedra, que paso de cinco a tres horas a la semana. El reto fue optimizar los recursos disponibles para la ejecución de los cursos. En términos de hora cátedra por parte de los profesores, esta reducción motivó al uso de nuevas estrategias pedagógicas que podrán mejorar el rendimiento sin deteriorar la relación efectividad - calidad.

El curso también se reestructuró para ser dictado con una hora de clase magistral y dos horas de laboratorio en la semana. En el proceso se logró construir un libreto estandarizado para ejecución de las actividades en clase, esto basado en que en el semestre pueden existir hasta 30 secciones diferentes, las cuales son evaluadas bajo el mismo examen impartido por la facultad.

Como resultados, se logró ajustar todos los contenidos programáticos en las tres horas directas de clase semanal. Se aumentó la eficiencia pedagógica del equipo de profesores en la asignación de tareas independientes y la capacidad de atención a estudiantes. Finalmente, las calificaciones se normalizaron disminuyendo la desviación estándar en las notas finales obtenidas por los estudiantes de la Facultad, que están cerca de mil estudiantes por semestre.

A. Diseño del Curso Semilla

Para el diseño del plan de curso se revisaron e incorporaron las recomendaciones dadas por la Association for Computing Machinery (ACM) en su currículo propuesto para ciencias computacionales [7]. Se integraron las tablas de saberes con los elementos necesarios para que la evaluación diera respuesta al logro de una competencia específica.

Desde el punto de vista de ACM se usó el área de conocimiento Software Development Fundamentals que incluye Algorithms and Design, Fundamentals Programming Concepts y Fundamental Data Structructures Fundamental. Estas tres grandes áreas integran competencias relacionadas con Familiarity (familiaridad), Usage (uso), y Assessment (eevaluación) de cada concepto dado en el plan de estudios.

Como resultado, se creó un curso semilla (o plantilla de curso), que permite estandarizar la forma como se desarrollan cada semana la temática, el material de apoyo, las actividades de refuerzo y evaluación, así como de contenidos adicionales. El curso semilla hace posible la actualización constante y fortalece el trabajo en equipo entre los profesores. En reuniones regulares, los profesores del área se reúnen para socializar experiencias y retro alimentar el proceso. Esto ha posibilitado el mejoramiento del aula virtual y del desarrollo de actividades presenciales.

En este momento, se han desarrollado más de 26 casos de estudio (guías integradas para la enseñanza de la

programación), originalmente escritas en texto plano y las soluciones descritas en el lenguaje de programación Python, como se describe más adelante. Los casos de estudio son compartidos a través de la plataforma institucional, la cual es desarrollada en Moodle.

La plataforma permite a los profesores utilizar una amplia diversidad de recursos predefinidos propicios para las aulas virtuales como son la asignación de talleres, la realización de evaluaciones, desarrollo de foros, o la entrega de ensayos. De esta forma se presenta material académico, se establecen las actividades para los estudiantes y se generan los espacios de colaboración y comunicación que enriquecen la evaluación y la retroalimentación. Fig. 1 muestra el detalle de una la semana uno del curso de Fundamentos de Computación, al igual que las actividades asociadas.

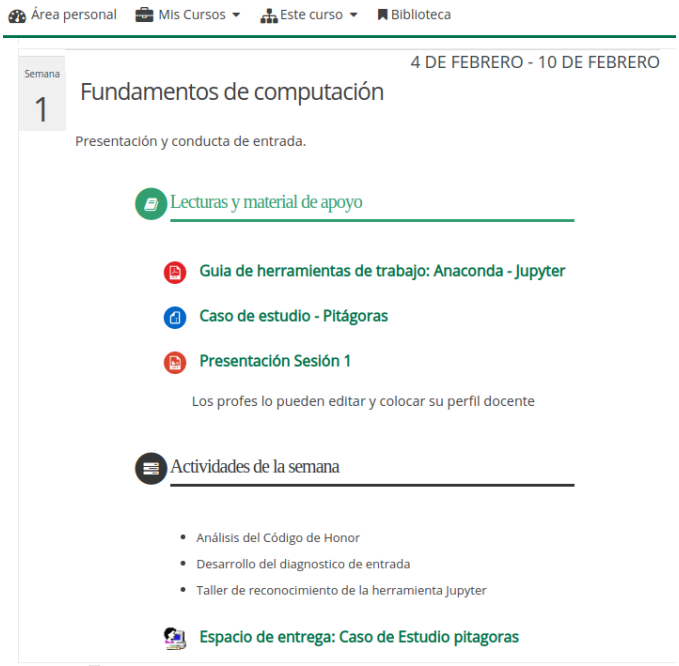

Fig 1: Captura de pantalla de la primera semana del curso semilla.

B. Diseño del Material del Curso

Para el desarrollo de los contenidos se enfocó en mejorar el acompañamiento realizado por el profesor en las clases magistrales y laboratorios. El material generado debe ser leído e interiorizado por los estudiantes antes de las clases magistrales. La intención de este material es generar preguntas fomentando la participación en torno al tema de la semana.

Para llevar a cabo este objetivo, se empleó la metodología de estudios de casos [8], la cual resultó ser muy efectiva a la hora de motivar al estudiante para participar de las actividades. Para la ambientación de estas cartillas se utilizaron temas relacionados con ciencia, tecnología, ingeniería y matemáticas (STEM) previamente conocidos por los estudiantes. Por ejemplo, se usaron ejercicios de física, cálculo, estadística, y ciencias en general donde se proponen situaciones para ser debatidas y ser analizadas en conjunto antes, durante, y después del encuentro presencial con el profesor.

C. Estructura de un Caso de Estudio

Inicialmente, los casos de estudio se desarrollaron como documentos para ser llevados impresos, estos incluyen historias de ambientación, explicaciones de contenidos, y ejemplos de programación. Cada recurso usado se desarrolló de la misma forma. Esto permitió homogeneizar no solo los contenidos, sino la forma de presentarlos a los estudiantes entre los diversos profesores y secciones de curso abiertas cada semestre. Cada estudio de caso tiene la siguiente estructura: contexto, requerimientos funcionales, requerimientos no funcionales, información, variables y estructuras de datos, estrategia de programación, casos de prueba, y envío de la actividad.

1) Contexto: El contexto es una historia que introduce el contexto donde se realizará el caso, trata un tema relacionado a un problema tipo STEM e induce a pensar una solución en términos computacionales.
2) Requerimientos Funcionales: En los requerimientos funcionales se declaran los objetivos de la solución que se desea alcanzar.
3) Requerimientos No Funcionales: En los requerimientos no funcionales se describe el comportamiento necesario para que el programa funcione correctamente.
4) Información, variables y estructuras de datos: Teniendo como base el contexto y los requerimientos, en esta sección se plantean una posible forma de organizar la información en el flujo del programa.
5) Estrategia de Programación: En la estrategia de programación, con la ayuda del lenguaje de programación Python se establecen posibles soluciones que dan respuesta a los requerimientos. En este punto es importante destacar que es posible que sea la primera vez que un estudiante se enfrenta a programar, por lo tanto, es necesario explicar detalladamente cada una de las opciones presentadas.
6) Casos de Prueba: Teniendo una posible solución planteada en el punto anterior, en los casos de prueba se brinda al estudiante datos de entrada y las posibles salidas del ejercicio, esto con el fin de comprobar que el trabajo se está realizando correctamente.
7) Envío de la Actividad: En el envío de la actividad se dan indicaciones del cómo se debe entregar el resultado usando la plataforma de educación virtual (Moodle).

III. EVOLUCIÓN DEL CURSO

Una vez se pudo comprobar con los estudiantes la buena aceptación de los casos de estudio, se introdujo una mejora significativa al curso. Esto fue la implementación de los casos como documentos vivos, en los que convergen explicaciones y ejecución de código fuente y la incorporación de los

laboratorios virtuales de programación. Detalles de estas adiciones se presentan a continuación.

A. Documentos Vivos

Para este fin se usó Jupyter Notebooks [9], una evolución de iPython [10]. Jupyter Notebooks es ampliamente usada para materiales educativos interactivos y con grandes capacidades de escalabilidad a problemas y soluciones más especializadas [11].

Fig. 2 muestra el esquema de servicios dispuesto para los usuarios. Existen dos mecanismos de acceso de parte de los estudiantes para realizar las actividades, el primero es descargar directamente las cartillas desde la plataforma de aprendizaje, y ejecutarlas localmente en su computador previa instalación de Python y Jupyter directamente (en los laboratorios de la Universidad, esto esta´ instalado por defecto).

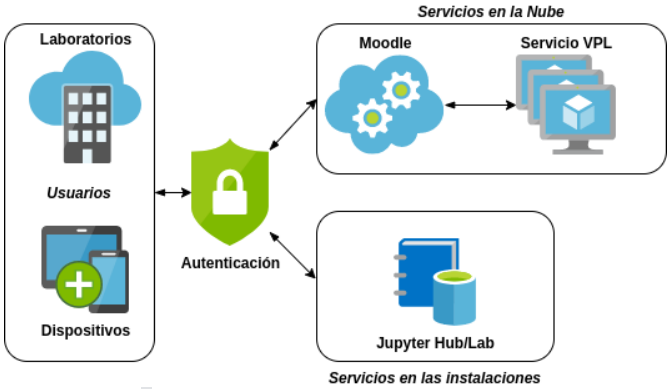

Fig 2: Esquema de servicios dispuestos para usuarios

El segundo mecanismo posible consiste en ingresar al portal del servidor de computación de alto desempeño, este tiene dispuesta una maquina con 32 cores y 96 GB de memoria al servicio, en el que se encuentra instalado Jupyter Hub y Jupyter Lab, esta opción es más especializada y se usa en cursos avanzados.

Cada usuario ingresa con sus credenciales de la institución a los servicios dispuestos para salvaguardar la integridad de los datos, pudiendo realizar las actividades y retos dispuestos en los cursos desde un navegador web y sin tener instalado el conjunto de aplicaciones necesarias en su computador personal o dispositivo móvil.

B. Laboratorios Virtuales de Programación

Además del uso de documentos vivos, se incorporó una nueva actividad, el uso de los laboratorios virtuales de programación o VPL [12]. Este es un proyecto de código abierto, originado en el Smith College, Massachusetts, Estados Unidos. Específicamente VPL es un módulo a través del cual el estudiante puede desarrollar experiencias interactivas de programación, en un ambiente controlado. Esto hace posible para el docente efectuar un seguimiento y retroalimentación mucho más detallado y específico.

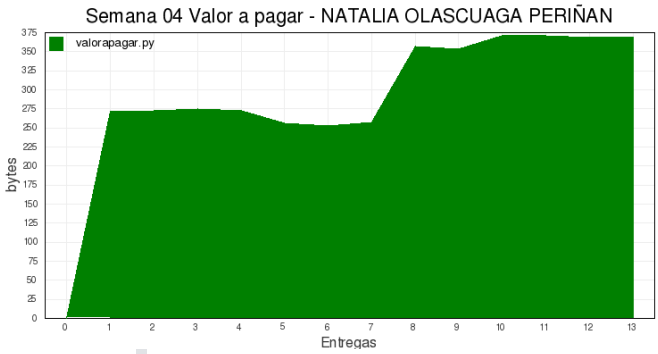

Fig 3: Tamaño de la solución construida por el alumno

Una de las ventajas del módulo es la facilidad para editar y ejecutar código Python (admite además otros lenguajes) desde el navegador, sin necesidad de salir de la plataforma de aprendizaje. El estudiante puede construir las soluciones en sesiones interactivas y de manera gradual. A medida que avanza en la construcción de su solución, el módulo VPL registra el progreso y almacena información, la queda a disposición del docente.

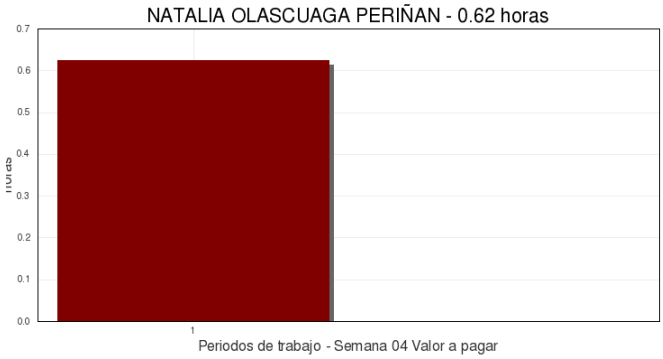

Fig 4: Periodos de trabajo del Alumno

El docente puede verificar cómo ha avanzado cada alumno desde distintos puntos de vista, así como revisar el código desarrollado por el estudiante en diferentes momentos de tiempo durante la experiencia. Esto sirve de insumo de información que VPL pone a su disposición relacionados con aspectos como el tamaño de la solución entregada (en bytes), como se muestra en la Fig. 3, o el tiempo consumido por el estudiante en las sesiones de trabajo (en tiempo), como se puede ver en la Fig. 4, y la a solución a la actividad en sus entregas parciales sucesivas (código de la solución como estaba en cada momento que el estudiante decide “guardar”), como se muestra en la Fig. 5.

#	Entregado el	Descripción
13	viernes, 28 de febrero de 2020, 14:57	valorapagar.py 369b 13l
12	viernes, 28 de febrero de 2020, 14:56	valorapagar.py 369b 13l
11	viernes, 28 de febrero de 2020, 14:55	valorapagar.py 371b 13l
10	viernes, 28 de febrero de 2020, 14:54	valorapagar.py 372b 13l
9	viernes, 28 de febrero de 2020, 14:50	valorapagar.py 354b 13l
8	viernes, 28 de febrero de 2020, 14:45	valorapagar.py 357b 13l
7	viernes, 28 de febrero de 2020, 14:43	valorapagar.py 258b 9l
6	viernes, 28 de febrero de 2020, 14:42	valorapagar.py 253b 9l
5	viernes, 28 de febrero de 2020, 14:41	valorapagar.py 257b 9l
4	viernes, 28 de febrero de 2020, 14:36	valorapagar.py 273b 9l
3	viernes, 28 de febrero de 2020, 14:35	valorapagar.py 275b 9l
2	viernes, 28 de febrero de 2020, 14:35	valorapagar.py 273b 9l
1	viernes, 28 de febrero de 2020, 14:22	valorapagar.py 272b 9l

Menos detalle

Fig 5: Registro de entregas sucesivas

Con el módulo VPL, el profesor puede obtener fácilmente un análisis detallado sobre la similitud existente entre las soluciones desarrolladas por los estudiantes y a partir de esa información detectar grupos, tendencias y diversas situaciones de interés y ejercer un control de autoría más preciso.

Para la ejecución de las actividades se pueden configurar restricciones de edición que permiten inhibir acciones como la subida de archivos provenientes de fuentes externas, lo cual asegura que el estudiante deba construir su propia solución. También se pueden configurar opciones de control tales como periodos de disponibilidad, cantidad máxima de archivos, tamaños límite, acceso mediante contraseñas, el uso exclusivo de navegadores seguros, entre otros.

1) Evaluación usando VPL: El esquema de evaluación utilizado en los cursos de Fundamentos de Programación implica que semanalmente el alumno deba desarrollar varios tipos de actividad. Las actividades se describen a continuación.

- Prácticas de programación interactivas en las sesiones de clase. Estas están basadas en los casos de estudio publicados previamente en el curso. Se desarrollada usando el módulo VPL con la orientación permanente del docente *in situ*. Para estas actividades, el estudiante debe formular la solución a partir de la descripción del caso y sus requerimientos. Seguidamente y utilizando el VPL, el estudiante hace la implementación y las pruebas de esta; finalmente, recibe la retroalimentación por parte del docente.
- Prácticas de programación independientes. El estudiante también debe atender actividades prácticas de programación como trabajo independiente, para resolver casos seleccionados. El estudiante puede escoger el ambiente de programación de su preferencia, pero que debe entregar su solución en el aula virtual del curso usando la herramienta VPL.
- Actividades de trabajo colaborativo. Aquí el docente selecciona y publica un conjunto de casos o problemas que constituyen material de estudio que serán libremente abordados por los alumnos como actividad para el estudio y la ejercitación. Paralelamente, se abre un foro de discusión en el aula virtual en el cual los alumnos, a medida que van interactuando con los casos, publican libremente sus preguntas, dudas, o situaciones de error encontradas. Esto permite que sus entradas al foro sean comentadas y respondidas por otros estudiantes, y retroalimentadas también por el docente a cargo. Con estas actividades se busca fortalecer la interacción y el trabajo colaborativo.

La evaluación de estas actividades se desarrolla aplicando rúbricas diseñadas para verificar los niveles de logro de cada estudiante. Se miden diversos aspectos de la solución como son la funcionalidad, su robustez lógica, la calidad del código, la creatividad, entre otras. Actualmente, el equipo docente está desarrollando acciones para adicionar al modelo la evaluación automatizada de las soluciones con base en *scripts* y casos de prueba debidamente acotados.

2) Ventajas de los VPL: El uso de esta herramienta representa sin duda ventajas significativas dentro del proceso.

- Se alcanzan niveles de participación más altos y se despiertan mayores niveles de motivación que se reflejan en porcentajes más altos de entrega, y mejores resultados promedio
- Se estimula el trabajo colaborativo mediante la libre interacción, característica que es altamente valorada por los alumnos.
- El docente dispone de más y mejor información para retroalimentar tempranamente al estudiante.

IV. EXPERIMENTANDO CON EL MUNDO FÍSICO.

Con la experiencia ganada en el diseño, desarrollo, y ejecución de los casos de estudio nació una oportunidad de mejora, inspirada en el trabajo realizado con legos en el aula [13] [14].

A. YarpTP [15] y YarpTp Notebooks [16]

Se propuso buscar un robot que pudiese ser construido por los mismos estudiantes, a bajo costo, y brindara una experiencia interactiva durante todo el proceso, similar a lo vivido con los

documentos vivos ya creados. El hallazgo fue YarpTP [15], el cual es un móvil que puede ser fabricado con una Raspberry Py, algunos elementos electrónicos, y una impresora 3D.

El móvil viene con un API para su manejo, los YarpTp Notebooks [16], lo cual facilitó su integración con el modelo actual del curso.

La inclusión del móvil mejoró la dinámica de las actividades de programación. Esta nueva forma de realizar las actividades se desarrolla en conjunto con estudiantes nuevos (primer semestre 2020). Estos son los primeros beneficiarios del rediseño micro curricular, el cual incluye casos de estudio que repercuten en un mundo físico a través del móvil controlado por un documento vivo.

Para el desarrollo de estos documentos vivos, llamados cartillas, se hicieron una serie de ajustes en los contenidos, motivando el trabajo en equipo en el que se usan metodologías ágiles para llegar a una meta [17]. La estructura propuesta de estas nuevas cartillas se explica a continuación.

1) Contexto: Es el entorno donde se desarrolla el caso de estudio, describiendo una situación en particular, cada cartilla aumenta la complejidad tratada y busca integrar las soluciones.
2) Resultado de Aprendizaje: Es la descripción de la meta que se debe cumplir. Esta se debe plantear en términos de cumplimiento respecto a las temáticas descritas en ACM.

3) Historia de Usuario: En esta parte se realiza la especificación completa de las metas alcanzables respecto a la funcionalidad de la solución. Se describen de forma ágil, usando el formalismo: Como (*How*), Quiero (*What*) y Para (*Why*). Por ejemplo, Como el robot quiero moverme 1 metro adelante para alcanzar la meta.

4) Información, variables y estructuras de datos: Es el planteamiento del cómo se organizan los datos y convertirlos en variables organizadas, siguiendo las reglas y estructuras del lenguaje de programación.

5) Estrategia de Solución: Es la integración completa entre la solución propuesta y la explicación para llegar a esa solución. En la práctica serán acciones del cómo se mueve el robot en el tablero dispuesto para la actividad.
Es importante resaltar que, para la correcta ejecución de la actividad, cada equipo de trabajo debe tener disponible un prototipo del móvil, el cual cuesta alrededor de 90 dólares americanos. En la Fig. 6 se muestra el escenario de pruebas.

Fig 6: Equipo de estudiantes de pregrado y secundaria trabajando en el prototipo robótico.

B. Estrategias de Atracción Nuevos Estudiantes
La incorporación del móvil al laboratorio de Programación ha permitido su utilización en campañas de fomento de la Ingeniería de Sistemas entre niños y jóvenes. Esta estrategia ha permitido ampliar el contacto con los estudiantes de secundaria de colegios de la ciudad. Los colegios se invitan a las instalaciones de la Universidad a participar en una feria de programas, donde los jóvenes se les presentan la oferta de programas. Para el caso de Sistemas y Computación, cuando el grupo de estudiantes llega, se les propone un reto, el cual debe ser cumplido por el robot. Al estudiante se le orienta el trabajo en equipo para completar el reto.
La estrategia usada con los colegios se describe a continuación.

1) Invitación del colegio. El profesor enlace del colegio selecciona un grupo 25 estudiantes interesados en estudiar carreras STEM.
2) Visita a la Universidad. Una vez en los laboratorios se trabajan las guías de YarpTp.
3) Diseño del robot. Las guías van desde modelado 3D hasta programar el robot para cumplir unos retos.
4) Construcción del robot. Se realiza un concurso y se imprimen en 3D las tres mejores propuestas de robot móvil.
5) Solución del reto. Los robots se construyen usando los modelos creados por los estudiantes y se compite resolviendo los retos. Un ejemplo de reto se muestra en la Fig. 7.

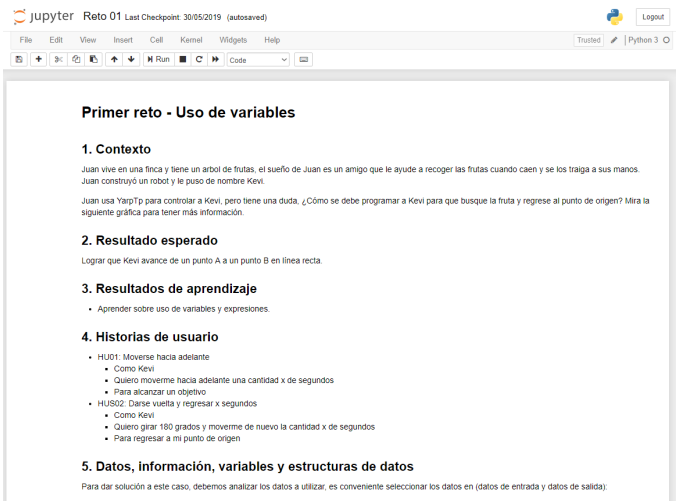

Fig 7: Introducción al primer reto.

V. IMPACTO

Con el rediseño del 2012, se redujo el número total de créditos, lo que implicó reorganizar los contenidos de algunos cursos o la eliminación de algunos cursos. Este fue el caso de Modelos Cuantitativos, el cual tenía dos cursos previos llamados Investigación de Operaciones I y II, los cuales fueron removidos, y sus contenidos integrados al nuevo curso de Sistemas y Modelos Matemáticos.

En la malla curricular del programa Ingeniería de Sistemas y Computación, el curso de Sistemas y Modelos Matemáticos requiere algunas competencias esenciales previas que debe adquirir un estudiante para poder asimilar mejor las competencias de este curso del área de la ingeniería aplicada en su campo laboral. Uno de los propósitos generales es "permitir a los futuros profesionales del área de las ciencias computacionales desarrollar habilidad para plantear y utilizar en forma eficiente y científica los modelos cuantitativos como herramienta para la solución de problemas relativos a la utilización de recursos en cualquier tipo de sistema u organización donde les corresponda actuar".

Con este curso el estudiante identifica diferentes algoritmos y métodos de resolución de problemas que normalmente se presentaban en los cursos de Investigación de Operaciones. También aprende a emplear algoritmos basado en la teoría de redes para facilitar la planificación de proyectos y reducir los riesgos en tiempo de ejecución de estos. Por estas razones, revisar los resultados en el curso de Sistemas y Modelos Matemáticos es un buen indicador de entrada para los estudiantes y de predicción al respecto de cómo se van a desempeñar en su futuro profesional. Es por ello, que se tomó como referente para medir el impacto del nuevo modelo curricular.

La experiencia con base en los promedios generales en cada semestre se muestra en la Fig. 8. La parte superior corresponde a cursos con estudiantes que tomaron Programación antes del rediseño. La parte inferior corresponde a cursos con estudiantes que han sido parte del nuevo modelo curricular.

Año	Promedio (sobre 5.0)
2010	3.5
2011	3.7
2012	3.4
2013	3.8
2014	3.9
2015	4.1
2016	3.6
2017	3.6
2018	3.6
2019	3.6

Fig 8: Promedio global del curso de Modelos Cuantitativos.

El promedio antes de la implementación del nuevo modelo curricular fue de 3.6 y después es de 3.7. Esto solo representa dos puntos porcentuales. Sin embargo, se debe recalcar que el número de estudiantes sobre el cual se hace el promedio ha venido creciendo en los últimos años, lo que indica mejoras en la retención y un crecimiento en el número de estudiantes que culminan satisfactoriamente el curso.

VI. RESULTADOS ALCANZADOS

El rediseño curricular de los cursos de programación ha pasado por diferentes etapas. El primer gran cambio fue la reducción de las horas de clase directa de cinco a tres horas, con el propósito de homogenizar los cursos en el programa de Ingeniería de Sistemas y Computación. Ese cambio obligó a la institución a pensar en formas de optimizar los recursos con los que se disponían. El primero de los cambios fue la incorporación de la metodología de estudio de casos STEM. El siguiente paso fue la utilización de documentos vivos y la integración completa del currículo de ACM, donde el caso (el problema a resolver) y la solución por parte de los estudiantes están en un Python Notebook.

Los cambios no se detuvieron ahí. Lo siguiente fue incorporar el módulo de VPL, lo cual permitió una mejor interacción entre el estudiante y el docente para el proceso de realimentación. El último cambio, pero no por ello, el menos importante, fue la introducción del robot y la definición de retos. Se paso del estudio de casos a la solución de retos con resultados físicos que nos solo ayuda a incentivar al estudiante a resolver retos cada vez más complejos, sino que nos permitió usarlo con

estudiantes de colegio como una actividad de captura de nuevos estudiantes.

VI. Conclusiones

El proceso nos ha permitido no solo mejorar los promedios finales de los cursos, sino reducir la deserción intersemestral debido a bajo rendimiento en un 7%. Eso ha permitido que el programa haya crecido 77% en número de estudiantes. Además, la metodología ha servido para motivar a nuevos estudiantes a ingresar a la Facultad de Ingeniería a estudiar programa en la línea STEM.

Reconocimientos

Loa autores agradecen a todos los profesores y estudiantes colaboradores de la Facultad de Ingeniería de la Universidad Tecnológica de Bolívar, y también al Grupo de Investigación GRITAS.

Bibliografía

[1] Ministerio de Educación Gobierno de Colombia. SPADIES 3.0 - Sistemas información.

[2] Héctor Alcides Peña Gil, Katherine Andrea Cuartas Castro, and Giovanny Mauricio Tarazona Bermúdez. La brecha digital en Colombia: Un análisis de las políticas gubernamentales para su disminución. Redes de Ingeniería, pages 59-71, September 2017.

[3] Takoi Hamrita. The Global Women in STEM Leadership Summit. Library Catalog: www.stemwomensummit.org.

[4] IEEE Women. IEEE Women in Engineering. Library Catalog: wie.ieee.org.

[5] Wendy Powley. ACM-W supporting, celebrating and advocating for Women in Computing. Library Catalog: women.acm.org.

[6] Observatorio TI FEDESOFT. Brecha talento humano de la industria TI actualizada a 2017.

[7] Association for Computing Machinery (ACM) Joint Task Force on Computing Curricula and IEEE Computer Society. Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science. ACM, New York, NY, USA, 2013.

[8] Marcia C. Linn and Michael J. Clancy. The Case for Case Studies of Programming Problems. Commun. ACM, 35(3):121-132, March 1992.

[9] Bernadette M. Randles, Irene V. Pasquetto, Milena S. Golshan, and Christine L. Borgman. Using the Jupyter Notebook as a Tool for Open Science: An Empirical Study. In 2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL), pages 1-2. IEEE, June 2017.

[10] F. Perez and B. E. Granger. IPython: A System for Interactive Scientific Computing. Computing in Science Engineering, 9(3):21-29, May 2007.

[11] Michael B. Milligan and Michael B. Jupyter as Common Technology Platform for Interactive HPC Services. In Proceedings of the Practice and Experience on Advanced Research Computing - PEARC '18, pages 1-6, New York, New York, USA, 2018. ACM Press.

[12] Juan Carlos Rodríguez del Pino, Enrique Rubio Royo, and Zenún Hernández Figueroa. A Virtual Programming Lab for Moodle with automatic assessment and anti-plagiarism features. Proceedings of The 2012 Internacional Conference on e-Learning, e-Business, Entreprise Information Systems, & e-Government. ISBN: 1-60132-209-7, 2012.

[13] Sabiha A Wadoo and Rahul Jain. A LEGO based undergraduate control systems laboratory. In 2012 IEEE Long Island Systems, Applications and Technology Conference (LISAT), pages 1-6. IEEE, May 2012.

[14] Valentin Haak, Joerg Abke, and Kai Borgeest. Conception of a Lego Mindstorms EV3 simulation for teaching C in computer science courses. In 2018 IEEE Global Engineering Education Conference (EDUCON), pages 478-483. IEEE, April 2018.

[15] Yuranis Henriquez, Kevin Pedroza, Kevin Vega, and Jairo E. Serrano C. Yet another robot platform for teaching programming: YarpTp. In 2018 IEEE 2nd Colombian Conference on Robotics and Automation (CCRA), pages 1-5, November 2018. ISSN: null.

[16] N Yuranis Henriquez, Jairo E. Serrano C, and Juan Carlos Martinez-Santos. YarpTp Notebooks a Tool For Teaching Programming. In 2019 Congreso Internacional de Innovación y Tendencias en Ingenieria (CONIITI), pages 1-6, October 2019.

[17] S. M. Saleh, S. M. Huq, and M. A. Rahman. Comparative Study within Scrum, Kanban, XP Focused on Their Practices. In 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE), pages 1-6, February 2019.