Relaciones de aprendizaje significativo entre dos paradigmas de programación a partir de dos lenguajes de programación

Meaningful learning strategy between two programming paradigms using two programming languages

Omar Iván Trejos Buriticá
Ingeniero de Sistemas. Doctor en Ciencias de la Educación. Docente de planta, Universidad Tecnológica de Pereira. Pereira, Colombia.
Contacto: *omartrejos@utp.edu.co*

Fecha de recepción: 19 de marzo de 2013 **Clasificación del artículo: investigación**

Fecha de aceptación: 23 de noviembre de 2013 **Financiamiento: Universidad Tecnológica de Pereira**

Palabras clave: aprendizaje significativo, lenguajes de programación, paradigmas de programación.

Key words: Meaningful learning, programming languages, programming paradigms.

RESUMEN

Una de las dificultades que, con frecuencia, se presenta en los currículos de Ingenieria de Sistemas es el hecho de que se establecen pocas relaciones entre lo que los estudiantes han aprendido en una asignatura de programación y lo que van a aprender en otra. Este artículo plantea una propuesta para que se puedan establecer, desde la primera sesión, relaciones entre el paradigma de programación funcional (utilizando DrScheme como instancia tecnológica) y el paradigma de programación estructurada (utilizando DevC++ en su arista estructurada) como lenguaje de programación. Se plantean las experiencias recogidas al respecto de la aplicación de esta estrategia y, sobre los resultados se presentan algunas reflexiones que pueden enriquecer el desarrollo de la línea de programación en los programas de Ingeniería a partir de las relaciones que se pueden establecer tanto entre paradigmas de programación como entre los lenguajes de programación con miras a que el camino de aprendizaje de estos sea mucho más simple.

ABSTRACT

One of the difficulties you can find often in the Computer Engineering curricula is the poor relations between the knowledge coming from an specific subject in a semester and the knowledge is coming from the next semester in an specific area. This article presents a proposal in the establishment of the relations between the functional

paradigm (using DrScheme as a programming language) and the structured paradigm (using DevC++ in its structured profile). We present some experiences and, due the results, we comment the possible way to enrich the programming line in a Computer Engineer program from the relations between programming paradigms and programming languages.

* * *

INTRODUCCIÓN

El proceso de formación de ingenieros de sistemas y su currículo asociado incluye dentro de sus ejes temáticos una línea que pareciera ser infaltable dentro de su concepción epistemológica y que corresponde a la Línea de Programación de Computadores. Esta línea tendrá, por razones temáticas naturales, dos aristas que permanentemente se entrecruzan para dar origen a los respectivos contenidos: los paradigmas de programación y los lenguajes de programación.

Los paradigmas corresponden a los modelos matemáticos que subyacen a una determinada forma de resolver un problema y que, en la actualidad, involucra en gran medida la participación de tecnología informática, computadores y herramientas de desarrollo sin que estos elementos sean absolutamente imprescindibles. Los lenguajes de programación corresponden a conjuntos de instrucciones que permiten construir programas a la luz de determinados paradigmas de programación e, incluso, como combinación de algunos de ellos. Estos conjuntos de instrucciones llamados Lenguajes de Programación son aceptados por la comunidad tecnológica internacional y cuentan con recursos como compiladores, ejecutores, editores y ambientes integrados de desarrollo que, en algunos casos, están disponibles libremente y se puede acceder a ellos a través de la web y, en otros casos, corresponden a la línea del *software* privativo.

Este artículo es producto del proyecto de investigación “Análisis pedagógico, instrumental y conceptual de algunos paradigmas de programación como contenido de la asignatura Programación I del Programa Ingeniería de Sistemas y Computación de la UTP”, registrado bajo el código 6-12-14 de la Vicerrectoría de Investigaciones, Innovación y Extensión de la Universidad Tecnológica de Pereira y aprobado por el Consejo de Facultad de Ingeniería en octubre de 2012.

Como se ha planteado, una de las dificultades que manifiestan los mismos estudiantes respecto de los contenidos de las asignaturas es la desconexión que experimentan bien cuando se cambia de paradigma o bien cuando se cambia de lenguaje de programación; sabiendo que no son tan inconexos, los estudiantes se preguntan las razones por las cuales no se acude a lo que ya saben para desarrollar nuevos temas en el mundo de la programación de computadores que, por ser tan cambiante, sería de gran utilidad en el rendimiento del avance temático (Mackay, 2005).

Este artículo, por tanto, se justifica dada la gran importancia que tiene la línea de programación en el desarrollo de la base conceptual de la Ingeniería de Sistemas como programa de formación profesional de pregrado, tanto como las posibles relaciones que se pueden establecer entre los paradigmas y los lenguajes de programación a favor del proceso de aprendizaje que los involucra. Durante muchos años de experiencia, el autor ha interactuado con una gran cantidad de cursos que se han movido básicamente entre el paradigma funcional declarativo, el paradigma estructurado y el paradigma orientado a objetos, y en todos ellos ha encontrado la leve relación que los mismos docentes establecen entre unos y otros, reiniciando cada vez los cursos de programación desde el principio cuando se podrían “conectar” de forma que se pudiera avanzar más en las 16 semanas que constituyen un semestre.

No debe descartarse tampoco que la visión que se presenta en este artículo es la visión del autor y que, si bien estas reflexiones se fundamentan no solo en la experiencia obtenida durante más de veinte años de ejercicio docente en el mundo de la ingeniería, sino en la interacción permanente tanto con estudiantes como con docentes del área de programación, corresponde solo a una propuesta tendiente a mejorar los procesos de aprendizaje asociados de forma que se hagan más efectivos los esfuerzos que, para ello, realizan los docentes.

Por esta razón es posible que algunas de estas reflexiones puedan ser compartidas con otros colegas, pero también que puedan ser controvertidas y, por ello, es interés del autor invitar a que, por algunos de los medios de divulgación científica, se manifiesten las opiniones y las reflexiones que enriquezcan una discusión que va en pro del mejoramiento de los procesos de formación en los programas de ingeniería. El objetivo de este artículo consiste en plantear una propuesta para establecer relaciones posibles entre el aprendizaje de la programación funcional (bien desde su paradigma o bien desde uno de sus lenguajes de aplicación) y la programación estructurada (Mackay, 2005) y someterlo a la mirada crítica de la comunidad que investiga procesos de aprendizaje en ingeniería.

Para el desarrollo de este artículo se acudió tanto a las bases tecnológicas, como corresponde a los paradigmas funcional y estructurado y a sus expresiones tecnológicas, lenguaje DrScheme y lenguaje c respectivamente, como a los fundamentos del aprendizaje significativo formulado por David Paul Ausubel y del aprendizaje por descubrimiento formulado por Jerome Seymour Bruner (1969). De la misma manera se monitorearon experiencias en el aula con cursos en paralelo, unos siguiendo la metodología tradicional de no establecer relaciones entre un paradigma y otro, y otros siguiendo la metodología que se propone en este artículo. Se hicieron las respectivas comparaciones y se plantearon algunas inferencias al respecto de dicha experiencia. Estas experiencias se hicieron con cursos paralelos de programación del programa Ingeniería de Sistemas y Computación de la Universidad Tecnológica de Pereira desde el primer semestre de 2011 hasta el primer semestre de 2013.

Sobre todo lo planteado anteriormente, el presente artículo comienza con una presentación de las teorías que lo fundamentan en lo conceptual (teoría del aprendizaje significativo y teoría del aprendizaje por descubrimiento) y continúa con lo puramente técnico que corresponde a los paradigmas de programación (funcional y estructurada) enunciadas desde la óptica de fundamentación matemática y asociada con un lenguaje de programación: DrScheme y C, respectivamente. Seguidamente se presenta la metodología que se utilizó dentro del proyecto de investigación tanto en su descripción como en su aplicación, se presentan los resultados, se abre el espacio para la discusión y finalmente se plantean unas conclusiones terminando el artículo con las respectivas referencias bibliográficas.

TEORÍA

Se presentan a continuación, y de manera sucinta, las teorías que se han tomado como base para el desarrollo del proyecto de investigación y que corresponden al ámbito de lo pedagógico, así como los paradigmas que se intentan relacionar en el contexto del presente artículo.

Aprendizaje significativo

Como "aprendizaje significativo" se conoce la teoría formulada por David Paul Ausubel quien planteó la gran importancia que tiene el significado de lo que se aprende dentro de un proceso de aprendizaje. Se llama aprendizaje significativo porque su fundamento es el concepto de significado, es decir, para qué sirve determinado conoci-

miento. La búsqueda de significado es una de las habilidades cognitivas de alto nivel innatas para el cerebro; es decir, cualquier evento, símbolo o situación que no sea claro para el cerebro, inmediatamente genera una reacción de este en la búsqueda de lo que significa, bien por medio de los patrones que tenga almacenados, bien mediante la información útil que haya recibido a través de los sentidos, o bien mediante la información simple (de uso poco frecuente) que haya guardado en la memoria a corto plazo.

El significado se convierte en el norte que permite que la información que llega al cerebro a través de los sentidos adopte una condición de patrón, información útil o información latente. Un patrón es un modelo informacional (o de conocimiento) que se usa con frecuencia y que llega a orientar ciertas decisiones del comportamiento humano basado en su significado. La información útil constituye ese conjunto de conocimientos que se usan con alta frecuencia y que, normalmente, tiende a ser un patrón sin serlo. La información latente es la información que ha llega por medio de los sentidos y que el cerebro aún no ha clasificado como información útil, ya porque no le haya encontrado significado o porque no es de uso frecuente.

Ausubel formula en su teoría que el aprendizaje se basa en tres fundamentos: el conocimiento previo, el nuevo conocimiento y la actitud del estudiante. El conocimiento previo es el conjunto de saberes que el estudiante tiene cuando se inicia un nuevo proceso formal de aprendizaje, como cuando se inicia un curso. La teoría del aprendizaje significativo establece que siempre que el ser humano se enfrenta a un proceso de aprendizaje, existe un conjunto de conocimientos que pueden considerarse como conocimiento previo, o sea, lo que el ser humano ya sabe antes de empezar a aprender. De acuerdo con esto, Ausubel establece que "si me preguntaran qué es lo más importante en el aprendizaje yo diría que es lo que el alumno ya sabe" (Ausubel, 1986) y con esto le está dando alta prioridad al conocimiento adquirido previo al inicio de un proceso de aprendizaje.

El nuevo conocimiento lo constituye el conjunto de saberes al que el alumno aún no había tenido oportunidad de acceder, que aún no se había formalizado desde alguna de las ciencias o que aún no se había difundido; lo que para el alumno pueda considerarse como nuevo e innovador, es decir, todo aquello que a un determinado alumno no haya llegado por ninguno de los medios o por ninguno de los sentidos. Es de aclarar que lo nuevo solo es nuevo para determinada persona, así no sea necesariamente nuevo.

La actitud del estudiante se define desde dos fronteras: la motivación y la capacidad que tenga el estudiante para relacionar conocimiento previo y nuevo conocimiento. En cuanto a la motivación, esta puede definirse como el ánimo y la voluntad que el estudiante pone para acceder a nuevos saberes y nuevos conocimientos y para incorporarse, de manera voluntaria, en un determinado proceso de aprendizaje. La motivación es la clave para que el cerebro busque todos los caminos posibles para establecer relaciones entre el conocimiento previo y el nuevo conocimiento. La capacidad para desarrollar estas relaciones se da cuando la motivación es lo suficientemente sólida como para que el mismo cerebro busque todos los caminos posibles que las posibiliten. Así, el modelo que estableció Ausubel para fundamentar su teoría del aprendizaje significativo determina que "el ser humano aprende mucho más fácil todo aquello que tiene significado para él" y parte del significado como esencia de dicha teoría basado en el conocimiento previo, el nuevo conocimiento y la actitud del estudiante.

Aprendizaje por descubrimiento

Otra teoría aceptada por el mundo, y que ha posibilitado caminos más expeditos para el aprendizaje, se conoce como la teoría del "aprendizaje por

descubrimiento", formulada por Jerome Seymour Bruner quien, a diferencia de Ausubel, propone que "el ser humano aprende mucho más fácil todo aquello que descubre" (Bruner, 1963). Con lo anterior propone un panorama que prioriza la fascinación y lo insólito como fundamento para que el proceso de aprendizaje ocupe el espacio de la memoria a largo plazo.

Según Bruner, actualmente profesor emérito de la Universidad de Nueva York, como descubrimiento se puede calificar a la "fascinación" que se traduce simplemente en la motivación que puede tener uno mismo para intentar explicarse lo insólito. En el proceso de aprendizaje, como se plantea en la teoría del aprendizaje por descubrimiento, se involucran tres procesos casi simultáneos: por una parte se tiene la adquisición de nueva información que se ha llamado como "adquisición". En segundo nivel, y casi al tiempo del primero, se presenta el proceso de manipular el conocimiento para hacerlo adecuado a nuevas tareas que el autor calificó como "transformación" y finalmente está el proceso que permite comprobar si la manera como hemos manipulado la información es adecuada a la tarea; etapa que el autor de la teoría calificó como "evaluación" (Bruner, 1991).

En relación con la teoría de aprendizaje significativo se puede decir que el concepto de "descubrimiento" corresponde a la chispa que abre el camino para encontrar el significado, ubicando el conocimiento en el nivel de un patrón que permite que la memoria a largo plazo asimile, apropie y, eventualmente, aplique cierto conjunto de conocimientos. Con esto se puede establecer una relación íntima entre el concepto de "significado" de la teoría de aprendizaje significativo y el concepto de "descubrimiento" de la teoría del aprendizaje por descubrimiento.

Finalmente, vale la pena incluir en esta fundamentación teórica sobre el aprendizaje que "todo conocimiento puede ser objeto de aprendizaje, es decir, todo se puede aprender" (Piaget, 1986), y que, por tanto, siempre existirán caminos expeditos para encontrar significado a lo nuevo y propiciar su descubrimiento por parte de las personas que se involucran en un proceso de aprendizaje (Piaget, 2001).

Programación funcional

La programación funcional se deriva del paradigma funcional, un modelo matemático basado en el cálculo *Lambda* que posibilita la construcción de soluciones simples basadas en funciones como núcleo básico de la programación. La función constituye el elemento principal a partir del cual se construye una solución que luego se revierte en un programa y que cuenta con características como paso de argumentos, nominación única, recursión, omisión de declaraciones y retornos automáticos (Schildt, 2000).

El paradigma de programación funcional aborda la construcción de soluciones a partir de tres conceptos básicos: simplificar el objetivo por alcanzar, facilitar las pruebas de escritorios y reusar lo construido. Dado que el objetivo resulta ser lo más importante en la construcción de un programa, el paradigma de programación funcional posibilita no solo la clarificación del mismo, sino también la simplificación en los frecuentes casos en los cuales el objetivo resulta tener un cierto nivel de complejidad (Trejos, 2002).

Reusar lo construido es una tendencia que se ha fortalecido con la irrupción del paradigma funcional, dado que cuando se vuelve a utilizar lo que ya se tiene hecho se logra acudir a fragmentos de código (funciones) que no solo han funcionado apropiadamente, sino que también han sido probadas en tiempo de ejecución (Trejos, 2005). Por otra parte, la reutilización del código (a nivel de funciones) permite hacer un óptimo uso del tiempo y, por tanto, hace que la labor de programar se vuelva mucho más eficiente en el uso del único recurso que no tiene repuesto, como es el tiempo.

A partir de la aparición del concepto de librerías y bibliotecas (tanto estándares como de usuarios), la reutilización del código se convirtió en una estrategia usada con frecuencia, de forma que se pueda acudir a funciones eficientes y que, así mismo, pueda ser la programación basada en este paradigma.

Los tiempos modernos exigen que el desarrollo de *software* y la construcción de programas incluya un ingrediente de alta importancia como el buen uso del tiempo; para ello, la programación funcional con su filosofía de construcción de soluciones a partir de funciones proporciona el camino preciso y perfecto para que así se cumpla, sin desconocer que otros paradigmas brindan herramientas que también posibilitan caminos eficientes de solución. En la actualidad, la tendencia a construir soluciones basadas en funciones eficientes ha posibilitado no solo que se fortalezca la programación funcional, sino también la programación estructurada y la programación orientada a objetos, que son las tendencias modernas que marcan el avance tecnológico en la actualidad (Van Santen, 2010).

Programación estructurada

El primer paradigma formal de programación de computadores se conoció como la programación estructurada, dado que es un modelo de programación que se basa en la máquina de estados de Von Newmann y se fundamenta en tres estructuras básicas. Antes de la programación estructurada se acudía a una técnica conocida como "programación libre", en la cual cada programador hacía sus programas como a bien tuviera.

Sin embargo, el estudio exhaustivo de los programas realizados a partir de la llamada programación libre permitió ir encontrando que todos los programas se encontraban y hacían uso de tres estructuras específicas. A partir de allí, tomando los conceptos matemáticos de la máquina de estados de von Newmann y de la máquina de Turing, se configuró un paradigma que, luego de más de sesenta años de haber sido formulado, sigue teniendo alguna vigencia; dado el tiempo de refinamiento, no solo perdura en algunas expresiones tecnológicas, sino que ha abierto la puerta para que otros paradigmas irrumpan en el mundo de la programación de computadores solucionando apropiadamente lo que el paradigma estructurado no había podido solucionar, o para lo cual sus soluciones eran altamente ineficientes o sus conceptos profundamente escasos (Van Roy, 2003).

Como su nombre lo indica, la programación estructurada se basa en unas estructuras básicas que en cantidad son tres y en definición corresponden a la secuencia de instrucciones, los condicionales y los ciclos. Este tipo de programación también se conoce como programación imperativa, aunque algo de este concepto es compartido con otros paradigmas. La estructura de secuencia establece que una instrucción se ejecuta completamente luego de la anterior y antes de la siguiente, y con ello determina la precedencia de ejecución de las instrucciones, lo cual le hace merecedor, a este paradigma estructurado, de lo puramente imperativo. La determinación de esta estructura permitió que muchas tareas se pudieran hacer de manera específica utilizando completamente la capacidad del computador y sus sistemas de procesamiento electrónico, de manera que las tareas capitalizaran las altas velocidades que para tal fin se involucran. Con el avance de la tecnología y la aparición de técnicas de programación como los *threads* (hilos) se ha podido entender que esta estructura en un ambiente puramente imperativo puede llegar a tener utilidad; sin embargo, en otros ambientes (tal vez distribuidos o multiprocesados) posibilitan e impiden la realización de varias tareas al tiempo, como sucede modernamente con los procesos multihilos que son los que permiten, por ejemplo, la ejecución simultánea de varias ventanas en el sistema operativo Windows.

Por su lado, la estructura de decisión (o condicional) permite que se pueda escoger uno entre dos caminos lógicos dependiendo de una condición. Dicha condición se escribe en términos de operadores relacionales y booleanos, y se evalúa a partir de los valores "verdadero" o "falso" que se pueden originar como respuesta de su revisión. Aunque no es necesario que todo condicional tenga de manera explícita los dos caminos, la estructura de decisión posibilita que siempre se puedan tener dos posibles opciones frente a una misma situación que pueda ser escrita en términos de los operadores mencionados. La estructura de decisión es, posiblemente, la única que se ha mantenido vigente desde que fue planteada como estructura básica, y se ha extrapolado hacia otros paradigmas como el caso de la programación funcional y la programación orientada a objetos con las mismas características que la han hecho útil en la programación estructurada.

Finalmente, la estructura cíclica o iterativa permite que se pueda ejecutar un conjunto de varias instrucciones tantas veces como una condición lo permita, de manera que su evaluación consienta, tal como sucede en la estructura de condición, que se realicen las tareas iterativas que se hayan propuesto. La estructura cíclica es la que más ha evolucionado en cuanto a su concepción y características desde que fue formulada como una de las estructuras básicas. Actualmente se puede hablar de dos formas de procesos cíclicos: los ciclos formales de la programación estructurada (hacer hasta, repetir mientras, hacer para, etc.) y los ciclos llamados recursivos que, si bien no corresponden a las características de los ciclos estructurados, de todas formas mantienen el espíritu de ser formas de lograr que un conjunto de instrucciones se repitan de manera finita dependiendo de una condición. Otra forma de construir ciclos son los ciclos no estructurados, heredados de la programación libre, pero estos ciclos se salen del contexto del presente artículo.

METODOLOGÍA

Descripción

En lo cualitativo, para el desarrollo del proyecto de investigación, se acudió a la teoría de aprendizaje significativo y la teoría del aprendizaje por descubrimiento. Sobre la primera se tomó la necesidad de definir los conocimientos previos y el nuevo conocimiento de manera taxativa, de forma que se pudieran establecer relaciones entre ambos, esencia misma de este artículo y se pudiera posibilitar un tránsito simple entre el aprendizaje de uno y otro paradigma, como se ha especificado en párrafos anteriores. Adicionalmente se hizo hincapié en la definición del significado de la programación funcional y la programación estructurada para poder establecer posibles relaciones que las vincularan.

En cuanto a la teoría de aprendizaje por descubrimiento se acudió a la definición misma de esta teoría, según la cual se invita al mismo estudiante a que encuentre algunas de esas relaciones y, a partir de un paradigma, encuentre las caracteristicas que distinguen y diferencian el otro paradigma bajo el marco de la fascinación como alta motivación para encontrar explicación a lo que pareciera insólito, o mejor aún, lo que pareciera nuevo.

Se ha capitalizado la relación entre el concepto de descubrimiento de la teoría de aprendizaje por descubrimiento y su íntimo nexo con el conocimiento nuevo de la teoría del aprendizaje significativo, de manera que el mismo estudiante busque y encuentre conexiones lógicas entre uno y otro paradigma; de esta forma, que él mismo encuentre elementos que faciliten y hagan mucho más expedito su propio aprendizaje.

Durante todo el proceso de levantamiento de la información se llevaron cursos paralelos de manera que en uno de ellos se explicaran las características de los paradigmas de programación tal

como tradicionalmente se conciben y, en el otro, se plantearan todos los conceptos de la programación estructurada a partir de los fundamentos que podían relacionarse de la programación funcional. Las pruebas de evaluación que se hicieron fueron las mismas en ambos cursos paralelos y se confrontaron resultados de forma que se pudieran hacer unos francos intentos de establecer relaciones de rendimiento entre un curso y otro. En lo cuantitativo se levantó una información y se cuantificó de acuerdo con los resultados obtenidos y, a partir de ellos, se hicieron algunas inferencias que son las que inspiran este artículo.

Se trató de mantener gran rigor metodológico para la aplicación tanto de la metodología como de las pruebas que las evaluaban, pero no se descarta que se puedan realizar procesos que refinan dichos procesos investigativos y que permitan brindar aún mayor confiabilidad a las inferencias que de este tipo de proyectos se deriven.

Aplicación

Primeramente debe tenerse en cuenta que, en el currículo del programa Ingeniería de Sistemas y Computación de la Universidad Tecnológica de Pereira, durante los semestres de desarrollo del proyecto de investigación (I semestre de 2011 hasta el I semestre de 2013), la estructura de la línea de programación de computadores inicia con la asignatura Programación I de primer semestre, cuyo contenido corresponde al paradigma funcional y el lenguaje que lo hace efectivo se llama DrScheme. En el 2º semestre se sirve la asignatura Programación II cuyo contenido se basa en el paradigma estructurado y su lenguaje de programación es el lenguaje C.

El proceso realizado a partir de la aplicación de la metodología que se planteó en el numeral anterior involucró los siguientes elementos: una definición del significado de cada uno de los conceptos que se querían abordar, tanto para el paradigma funcional como para el paradigma estructurado, en el caso de aquellos conceptos que son plenamente compartidos en lo conceptual. Ahora bien, para aquellos conceptos que no se comparten conceptualmente en ambos paradigmas se estableció el significado acorde con su sentido dentro del paradigma al cual correspondiera, y se propusieron relaciones de analogía que permitiera derivar uno del otro.

También se realizó una definición clara y concreta de los conceptos que constituían el conocimiento previo en cada uno de los temas que se abordó con dicha metodología; el conocimiento previo se basó en el paradigma funcional y en su aplicación en lenguaje DrScheme, así como una definición clara y concreta de los conceptos que constituían el nuevo conocimiento en cada uno de los temas que se abordó con la metodología expuesta. Además, el nuevo conocimiento se basó en el paradigma estructurado y en su aplicación en Lenguaje C.

De la misma manera se precisó una definición de los criterios que permiten que se relacione el conocimiento previo definido (paradigma funcional) con los conceptos que constituyen el nuevo conocimiento (paradigma estructurado), y una definición clara de los conceptos nuevos que podrían constituirse en lo insólito o en lo innovador, acorde con la teoría del aprendizaje por descubrimiento.

Se hizo un desarrollo permanente de actividades de motivación frente al nuevo conocimiento, a la clarificación de los conocimientos previos mediante estrategias lúdicas, tecnológicas y conceptuales utilizando juegos como crucigramas, sopas de letras, sudokus y kakuros al inicio de cada sesión; se fijaron metas en cada sesión en las cuales se iban a abordar nuevos conceptos como parte del proceso de adquisición; se establecieron criterios para la aplicación del conocimiento nuevo en diferentes ámbitos de programación como parte del proceso de transformación y se

Tabla 1. Conceptos relacionados de programación

No	Tema	Conocimiento Previo (Paradigma Funcional)	Nuevo Conocimiento (Paradigma Estructurado)
1	Tipos de datos	Se refuerza el concepto de argumento y almacenamiento	Se explican los tipos de datos formales y el concepto de variable a partir del concepto de argumento
2	Concepto de función	Se retoman las características de una función en DrScheme	Se presenta la estructura de una función en Lenguaje C y se asocia con lo presentado en DrScheme
3	Estructura de un programa	Se refuerza el concepto de programa a partir de funciones	Se presenta la función principal y las funciones auxiliares como una derivación del concepto de funciones en programación funcional
4	Construcción de librerías	Se plantea el concepto de librerías a partir de los módulos gráficos, GUI y matemáticos en DrScheme	Se acude a las librerías estándar .h y se formula la manera de construirlas e incluirlas en los programas a partir del concepto funcional
5	Recursividad	Se retoman las tres partes de una función recursiva dentro del paradigma funcional	Se aplican las mismas partes a las funciones estructuradas
6	Funciones de biblioteca	Se acude a las librerías gráficas, GUI y matemáticas	Se acude a las librerías estándar (stdio.h, conio.h, dos.h, process.h, iostream.h y graphics,h)
7	Sintaxis del lenguaje	Se plantean las caracteristicas principales del lenguaje DrScheme	Se amplían las características del lenguaje DrScheme para llegar a la sintaxis del Lenguaje C
8	Pensamiento funcional	Se formulan los tres principios básicos del pensamiento funcional	Se formulan las estructuras y se aplican a partir de los tres principios básicos del pensamiento funcional
9	Pensamiento estructurado	Se plantean soluciones a nivel de programación funcional a partir de una aproximación a las estructuras básicas	Se retoman las soluciones con programación funcional y se reescriben con programación estructurada

Fuente: elaboración propia.

asociaron los conceptos con aplicaciones y situaciones de la vida cotidiana de los estudiantes; se diseñaron evaluaciones que servían de retroalimentación de los objetivos de aprendizaje presentados a los estudiantes y en consonancia con dichos objetivos previamente definidos; y se compartió con los estudiantes cada uno de los conceptos, significados y conocimientos que se plantearon como parte de la metodología, así como los fundamentos de la teoría de aprendizaje significativo y aprendizaje por descubrimiento para que en todo momento el mismo estudiante fuera consciente de sus propias metas y del avance que fuera teniendo frente a ellas.

A partir de estos principios, la estrategia metodológica aplicada permitió que diferentes temas fueran desarrollados tal como se plantea en la tabla 1 en la que se relacionan algunos de ellos.

RESULTADOS

En el desarrollo de los cursos paralelos de programación de computadores (uno con la metodología planteada y otro sin dicha metodología), a lo largo de los semestres de prueba se obtuvieron los resultados cuantitativos que se presentan en la tabla 2.

Es de anotar que los resultados presentados en la tabla 2 corresponden al promedio de las notas de cada prueba en cada uno de los cursos donde se realizaron las pruebas de este proyecto de investi-

Tabla 2. Resultados cuantitativos

Prueba	%	Promedio cursos aplicando metodología					Promedio cursos sin aplicar metodología				
		I 2011	II 2011	I 2012	II 2012	I 2013	I 2011	II 2011	I 2012	II 2012	I 2013
I Parcial	30	4,2	4,3	4,3	4,4	4,3	3,8	3,6	3,9	4,3	3,9
II Parcial	30	3,4	3,4	3,6	3,6	3,7	3,2	3,2	3,1	2,9	3,5
Ex. Final	40	4,0	4,1	4,1	4,2	4,2	4,0	4,1	3,7	3,6	4,0
Definitiva	100	4,9	4,0	4,0	4,1	4,1	3,7	3,7	3,6	3,6	3,8

Fuente: elaboración propia.

gación. Se puede destacar que el avance numérico en el promedio de las notas no es muy notorio dado que, en un mismo semestre, en algunos cursos el promedio fue aumentando mientras que en otro no tuvo un comportamiento distinguible. Sin embargo, se nota que la aplicación de esta metodología, con el paso del tiempo, va generando un refinamiento que posibilita que las notas equivalentes (por ejemplo, promedio de notas del I Parcial) vaya mejorando en el caso de los cursos donde se aplicó la metodología. No se nota un comportamiento caracterizable en los cursos en donde esta no se aplicó.

Si se compara uno a uno el promedio de las notas en cada semestre, se encuentra que en algunos fueron mejor las notas en los cursos donde la metodología se aplicó y en otros fue mejor el promedio de las notas de los otros cursos; sin embargo, la tendencia a mejorar las notas con el paso del tiempo es un hecho notorio en los cursos del primer grupo (o sea donde se aplicó la metodología). Las notas definitivas fueron mejores cuantitativamente en los cursos del primer grupo, en comparación con los cursos del segundo grupo. Aunque no se nota un comportamiento incremental con el paso del tiempo, se puede notar que si se compara uno a uno dichos promedios, son mejores los que se obtuvieron con la metodología propuesta.

En cuanto al análisis cualitativo se debe acotar que este análisis se hizo teniendo en cuenta la claridad de los conceptos por parte de los alumnos y el rendimiento académico independiente de las notas que obtenían en sus pruebas. Este rendimiento académico se basó en la manera como ellos avanzaban en el desarrollo de los temas, en la calidad de las preguntas que hacían, en la forma como apropiaban y aplicaban los nuevos conceptos, y en las relaciones que establecían, *motu proprio*, entre el paradigma funcional y el paradigma estructurado. La tabla 3 nos muestra los resultados que se obtuvieron, teniendo en cuenta que este promedio se logró considerando como significativo el concepto que se derivara de la mayoría de los estudiantes. Si bien los análisis cualitativos fueron mucho más amplios, debe anotarse que por razones propias de la estructura de este artículo solo se citan los resultados resúmenes que compilan toda la experiencia en su arista puramente cualitativa.

El concepto definitivo cualitativo se obtiene observando la valoración cualitativa mayoritaria entre los conceptos obtenidos a lo largo del semestre. De esta manera, si de los tres conceptos dos de ellos son iguales, entonces este pasa a ser el concepto definitivo. En caso de que se encuentren tres conceptos diferentes se acude a la escala y se busca cuál se ubica en el centro de los tres conceptos, y ese pasa a ser el concepto definitivo. En caso de que los tres conceptos sean iguales, entonces ese mismo pasa a ser el concepto definitivo.

Vale la pena destacar que los conceptos cualitativos de rendimiento académico de todos los cursos son notoriamente mejores en los cursos donde se aplicó la metodología que en los cursos en

Tabla 3. Resultados cualitativos

Prueba	%	Promedio cursos aplicando metodología					Promedio cursos sin aplicar metodología				
		I 2011	II 2011	I 2012	II 2012	I 2013	I 2011	II 2011	I 2012	II 2012	I 2013
I Parcial	30	B	B	E	B	E	R	R	B	R	B
II Parcial	30	B	MB	MB	E	B	B	B	B	B	R
Ex. Final	40	MB	B	B	MB	B	R	B	B	R	R
Definitiva	100	B	B	MB	MB	B	R	B	B	R	R

E=Excelente; MB=Muy bueno; B=Bueno; R=Regular; M=Malo

Fuente: elaboración propia.

los que no se aplicó. Por la rigurosidad de los análisis realizados desde el punto de vista cualitativo, puede decirse que los cursos en los que se aplicó la metodología establecieron relaciones más sólidas entre el conocimiento previo (paradigma de programación funcional) y el nuevo conocimiento (paradigma de programación estructurada); esto se notó de manera evidente en el desarrollo de los talleres, actividades y dinámicas tendientes a la asimilación, la apropiación, la aplicación y el cuestionamiento de los conceptos fundamentales de cada paradigma.

DISCUSIÓN

Una de las ventajas de establecer criterios tanto cualitativos como cuantitativos es que se posibilita la evaluación de este tipo de propuestas metodológicas desde dos aristas: una que revisa los resultados cuantitativos que no se deben descuidar, dado que el sistema de valoración de la evaluación en la educación superior así lo exige y teniendo en cuenta que, si se aplica apropiadamente, puede arrojar resultados valiosos y confiables. Por otra parte, la revisión de los resultados cualitativos basados mucho más en el comportamiento de los estudiantes, en la manera como apropian los conceptos, en el nivel de las inquietudes que formulan, en la manera como aplican y cuestionan lo visto y en las relaciones que establecen entre el paradigma de programación funcional con el paradigma de programación estructurada, lo cual permite tener un panorama mucho más objetivo de cada curso en comparación con lo que arrojan las notas.

Si el objetivo es que los estudiantes alcancen un nivel de suficiencia tanto en la comprensión del paradigma de programación funcional como del paradigma de programación estructurada y de la relaciones que entre los dos pueden establecerse, entonces puede decirse que los cursos en los cuales la metodología se aplicó los estudiantes aprendieron más que aquellos en donde no se aplicó dicha metodología. No se puede cerrar la posibilidad de refinar los procesos que se adoptaron en cuanto a estrategias, actividades, talleres, evaluaciones, quices y pruebas en general, pero debe destacarse que se tuvo particular cuidado en aplicarlos de manera idéntica en ambos cursos paralelos, de hacerlo en los mismos horarios y, prácticamente, en las mismas condiciones tanto ambientales como circunstanciales, así como en los mismos días, de manera que se pudieran obtener resultados de alta confiabilidad.

Puede decirse, con los resultados cuantitativos y cualitativos de este proyecto, que se confirmó que el estudiante aprende mucho más cuando se relaciona el nuevo conocimiento que se le quiere compartir con el conocimiento previo que ya trae, que para el caso de este proyecto de investigación corresponde al paradigma de programación funcional y al paradigma de programación estructurada respectivamente. Puede pensarse en

diferentes variantes que se podrían adoptar con el ánimo de verificar la validez de los datos que se han recogido y las valoraciones cualitativas que se han asignado; sin embargo, debe destacarse que la rigurosidad investigativa realizada y la experiencia directa con los mismos estudiantes ha permitido confirmar, *in situ* y en tiempo real, las conclusiones que se obtuvieron desde la óptica cualitativa y cuantitativa.

CONCLUSIONES

Según los hallazgos encontrados se puede asegurar que es posible establecer relaciones entre el paradigma de programación funcional y el paradigma de programación estructurada, y que mediante dichas relaciones es mucho más asimilable el paradigma de programación estructurada y su metodología es mucho más digerible si se parte de conceptos fundamentales, como el concepto de función. Toda vez que se establecen relaciones entre el nuevo conocimiento y el conocimiento previo se develan a los estudiantes y se utilizan en la apropiación de conocimiento, los resultados van en el sentido de los objetivos del aprendizaje y el concepto de significado se fortalece cada vez que podemos establecer relaciones entre el conocimiento previo y el nuevo conocimiento.

Siempre es posible establecer relaciones entre conocimientos previos y nuevos conocimientos y entre paradigmas de programación, cualquiera que sea el orden en que estos se vean, con el ánimo de potencializar el proceso de aprendizaje. Todo proceso investigativo debe validarse desde lo cuantitativo para analizar los formalismos que se involucran, y desde lo cualitativo, para analizar las relaciones que se derivan de dichos formalismos. Si el docente lo quiere, siempre podrá encontrar caminos por donde los objetivos de aprendizaje se pueden lograr de manera más expedita, más comprensible y más digerible por parte de los estudiantes.

REFERENCIAS

Ausubel P. (1986). *Sicología Educativa: un punto de vista cognoscitivo*. México: Editorial Trillas.

Bruner J. (1991). *Actos de significado*. Madrid: Alianza Editorial.

Bruner J. (1963). *El proceso de la Educación*. México: Editorial Hispanoamericana.

Bruner J. (1969). *Hacia una teoría de la instrucción*. México: Editorial Hispanoamericana.

Mackay D. (2005). *Information Theory, Inference and Learning Algorithms.* Cambridge: University Press.

Piaget J. (1986). *Inteligencia y afectividad*. Buenos Aires: Aique.

Piaget J. (2001). Psicología y Pedagogía. México: Crítica.

Schildt H. (2000). *The Complete Reference C*. USA: McGraw Hill.

Trejos O. (2005). *Fundamentos de Programación*. Pereira, Editorial Papiro:

Trejos O. (2002). *La esencia de la lógica de programación*. Manizales: Centro Editorial Universidad de Caldas.

Van Roy P. (2003). *Concepts, Techniques and Models of Computer Programming.* Estocolmo: Swedish Institute of Computer Science.

Van Santen R. (2010). *2030 Technology that Will Change the World*. Oxford: Oxford University Press.