Estrategia metodológica para aproximar los paradigmas funcional, estructurado y orientado a objetos en ingeniería de sistemas a partir de aprendizaje significativo

Methodological strategy to approach the Functional, Structured and Object Oriented paradigms in a Systems Engineering program using meaningful learning

Ómar Iván Trejos Buriticá[1]*

[1]PhD Ciencias de la Educación, Universidad Tecnologica de Pereira. *omartrejos@utp.edu.co

Fecha de recepción del artículo: 03/12/2012 Fecha de aceptación del artículo: 17/10/2013

Resumen

El presente artículo propone una estrategia metodológica con la intención de facilitar el aprendizaje de los paradigmas de programación identificados como: paradigma funcional, paradigma estructurado y paradigma orientado a objetos basados en los principios fundamentales del aprendizaje significativo, y acudiendo a la apropiación del conocimiento previo como base para el desarrollo de nuevos conocimientos. Este artículo es producto de un proyecto de investigación y se ha desarrollado acorde con las normas que rigen para la escritura de artículos científicos, realizando una exposición que incluye teoría, metodología, resultados, discusión y conclusiones. En su más simple esencia se presentan los tres paradigmas como un derivado del concepto que los une y que, al tiempo, es el que posibilita las diferencias sustanciales entre unos y otros.

Palabras clave

Aprendizaje, Aprendizaje significativo, Metodología, Programación, Programación funcional, Programación estructurada, Programación orientada a objetos.

Abstract

This paper presents a methodological strategy intended to facilitate the learning of programming paradigms identified as functional paradigm, structured paradigm and object-oriented paradigm based on the fundamental principles of meaningful learning and going to the appropriation of previous knowledge as a basis for development of new knowledge. This article is a product of a research project and has been developed in accordance with the rules for writing scientific papers that includes theory, methodology, results, discussion and conclusions. In its simplest essence you can find that the three paradigms are a derivative concept that unify and, at the time, establish differences between ones and the others.

Keywords

Learning, meaningful Learning, methodology, Programming, Functional Programming, structured Programming, object oriented programming.

1. Introducción

El presente artículo es un subproducto 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", recomendado por el consejo de facultad de ingenierías de la Universidad Tecnológica de Pereira y aprobado por la vicerrectoría de investigaciones, extensión e innovación bajo el código 6-12-14 de octubre de 2012.

Debido a la gran oferta académica del programa ingeniería de sistemas en Colombia y en América Latina (y de sus vertientes a nivel de tecnologías y otras ingenierías) y a la importancia que tiene la línea de programación de computadores, esencia y espina dorsal de la ingeniería de sistemas, tanto este artículo como el proyecto de investigación que lo inspira, encuentra una justificación en la búsqueda permanente de caminos pedagógicos que posibiliten el acceso al conocimiento especializado a través de estrategias que lo faciliten y que tengan en cuenta conceptos tan prácticos y aplicativos como el conocimiento previo, el nuevo conocimiento, la motivación y los conceptos fundantes del aprendizaje por descubrimiento.

De otra parte, el acceso a los fundamentos de la programación de computadores, las ingentes necesidades que tienen otras profesiones de acceder a este conocimiento y las posibilidades que brinda esta como camino para aprovechar los recursos y las facilidades tecnológicas en una máxima expresión, justifican esta propuesta tanto de relacionar los conceptos que el estudiante de programación (cualquiera que sea el programa al cual pertenece) necesita para encontrar aquellas relaciones que subyacen a los diferentes paradigmas de programación como los caminos que posibiliten un aprendizaje más expedito de los conceptos fundamentales y de sus posteriores aplicaciones.

El desarrollo de la línea de programación en un currículo formal de un programa de ingeniería de sistemas o de sus programas relacionados, implica realizar un análisis tanto de paradigmas como de lenguajes (como expresión de aquellos) de manera que se pueda hilar de forma coherente un conjunto de asignaturas que, a la postre, le proporcionen al estudiante los conceptos básicos y sus aplicaciones derivadas para capitalizar al máximo dichos paradigmas como modelos de solución de problemas computacionales.

Esta coherencia que los paradigmas y lenguajes de programación invitan a tener en cuenta en el diseño de los currículos de ingeniería de sistemas (y de sus programas relacionados o derivados) no siempre resultan efectivos y, con gran frecuencia, el estudiante de dichos programas se enfrenta a un conjunto de asignaturas en donde tiene contacto con paradigmas o lenguajes (enfoque conceptual o enfoque instrumental) débilmente relacionados y con un conjunto de temáticas distintas normalmente inconexas a pesar de que a todas ellas les subyacen conceptos y fundamentos comunes. Es allí en donde este artículo cobra su justificación, pues pretende aproximar tanto a los docentes como a los estudiantes y a las instituciones de ingeniería de sistemas a posibles relaciones entre tres paradigmas de programación (los más comunes) para que el gran beneficiado sea el proceso de aprendizaje en el estudiante y, con ello, pueda aprovechar al máximo las posibilidades que la tecnología, como eje temático académico y aplicativo, le proporciona.

De no ser porque el área de programación de computadores es la espina dorsal de la ingeniería de sistemas como programa de formación profesional de la cual se derivan todas las otras líneas de profundización, y que dicho programa es el que más se ofrece tanto en Colombia como en los países latinoamericanos, bien en su presentación como ingeniería de sistemas o bien como programa anexo, conexo o derivado de este, la relación entre paradigmas y lenguajes de programación podría ser un problema menor; pero teniendo en cuenta eso debe reconocerse que bien vale la pena abordarlo. De una parte, para que el estudiante encuentre un camino de relación entre paradigmas y lenguajes de programación que, con gran frecuencia, es esqui-

vo; por otro lado, para que los mismos docentes puedan tener mucho más claros los conceptos que subyacen a cada paradigma y, de esta forma, puedan compartir sus conocimientos por caminos más didácticos y simples; por último, para que las instituciones puedan capitalizar de una mejor forma todos los esfuerzos que hacen en pro del aprendizaje de temáticas que, como esta, constituyen el centro de atención y derivación de la ingeniería de sistemas y de sus programas derivados.

En el área de programación de computadores, específicamente orientada a la ingeniería de sistemas, se han desarrollado muchos estudios tendientes a fortalecer los conceptos que subyacen a un determinado paradigma. En este sentido, el volumen de producción bibliográfica es alto, al punto que todo un mundo de opciones comerciales se abre cuando un estudiante quiere acceder a los conceptos de un determinado lenguaje de programación, y un mundo más reducido cuando su intención va orientada más hacia los paradigmas de programación.

Este mundo bibliográfico, y su acceso a él, se han disparado a raíz de la penetración masiva de la internet y de sus servicios asociados, puesto que en tiempos modernos, el acceso a la información y a las fuentes de conocimiento no solo compete a las bibliotecas y librerías sino también a los sitios virtuales que posibilitan, desde diferentes horizontes legales, la descarga de libros alrededor del tema tratado.

Poca información formal se encuentra alrededor de las posibles relaciones que se puedan establecer entre los paradigmas de programación y, también, entre los lenguajes de programación, ya que un altísimo porcentaje de los esfuerzos bibliográficos se han centrado en la presentación de libros que expliquen, con diferentes estilos y formas, los principios de un determinado paradigma (en pocos casos), de un determinado lenguaje de programación (en muchos casos) y de la posible relación entre ellos (en mínimos casos).

De todas formas no se puede desconocer los esfuerzos realizados por autores como Odell (1991), Schildt (1998), Joyannes (2006) y Van Roy (2008) alrededor de la temática que inspira el presente artículo. En cuanto a la producción de artículos de investigación científica que aborden esta temática, debe admitirse, que si bien el tema ha sido tocado con gran timidez por los autores, no ha constituido la temática central de los artículos que se han revisado en las bases de datos Science Direct, IEEExplore y ProQuest, sin dejar de aceptar que, en algunos de ellos, se han presentado destellos que dejan entrever la importancia de la temática dentro del contexto de la programación de computadores y específicamente en su aplicación hacia la ingeniería de sistemas.

Lo innovador en este artículo es que las posibles relaciones que se plantean entre el paradigma de programación, programación estructurada, y programación orientada a objetos, están basadas en dos teorías de aprendizaje: la teoría del aprendizaje significativo, formulada por el Dr. David Paul Ausubel y la teoría del aprendizaje por descubrimiento, formulada por el Dr. Jerome Seymour Bruner, que, a raíz de la tendencia masiva a la formación por competencias promovida en las instituciones desde los organismos de control del Ministerio de Educación de Colombia, cobra mucha más importancia, toda vez que estas teorías no solo le imprimen el concepto de significado al conocimiento tecnológico, como es el tema de este artículo, sino que extrapolan dicho significado para encontrar posibles relaciones con otros modelos formales que se mueven dentro de la misma esfera académica de los paradigmas y los lenguajes de programación.

Este artículo propone un núcleo conceptual a partir del cual pueden desarrollarse los tres paradigmas de programación mencionados, y que sirve a su vez como puente de relación y aplicación de los tres paradigmas y de sus expresiones llevadas al nivel de un lenguaje de programación. Esta propuesta es susceptible de mejora, de discusión y de crítica, pero es un buen punto de partida para que los docentes, los estudiantes y las instituciones relacionadas con programas de ingeniería de sistemas, o que incluyan la programación de computadores

entre sus objetivos de aprendizaje, comiencen a tener conciencia de que cuando se plantean posibles relaciones entre teorías o modelos cercanos o que pertenecen a una misma esfera del conocimiento, su aprendizaje no solo es mucho más expedito sino, al tiempo, mucho más efectivo y sólido.

Por todo lo dicho hasta ahora, el objetivo general de este artículo consiste en formular y explicar un núcleo temático que sirva de base para establecer relaciones entre los paradigmas de programación funcional, programación estructurada y programación orientada a objetos de manera que posibilite un camino de apropiación, asimilación y aplicación de estos paradigmas, y de sus relaciones, que solidifique los procesos de aprendizaje que giren en torno a cualquiera de estos temas y permita el alcance de sus respectivos logros.

Para desarrollar este artículo se hizo necesario acudir a los fundamentos conceptuales de los paradigmas de programación funcional, programación estructurada y programación orientada a objetos, así como conocer y apropiar las bases de la teoría del aprendizaje significativo y la teoría del aprendizaje por descubrimiento. De la misma manera, acudiendo a las teorías de aprendizaje, se establecieron relaciones entre los paradigmas de programación y se llegó a la formulación de un núcleo temático que no solo fortalece los tres paradigmas sino que además los explica y los relaciona, que corresponde al espíritu que inspira este artículo.

Al mismo tiempo se hizo necesario acudir a lenguajes de programación que cristalizaran lo planteado en este artículo y por ello se utilizó DrScheme como lenguaje de programación del paradigma funcional, Lenguaje C como lenguaje de programación del paradigma estructurado y Java como lenguaje de programación del paradigma orientado a objetos. De la misma manera, para el desarrollo de este artículo, se socializó con los estudiantes la propuesta planteada y se hicieron pruebas que permitieran monitorear el efecto de las relaciones establecidas, del núcleo temático definido y de las posibilidades que se abren cuando se pueden aplicar dichas relaciones independiente del paradigma con el cual se esté trabajando.

Conceptualmente, para poder llegar a un núcleo que posibilitara relaciones entre los tres paradigmas mencionados, se acudió a la profundización en el tema tanto desde lo puramente conceptual como desde el modelo matemático y su expresión tecnológica a través de un lenguaje de programación. De la misma forma se evidenciaron y evaluaron las potencialidades del núcleo definido como base de relación para los paradigmas y se socializó con los estudiantes de diferentes niveles de programación, cada uno de ellos dedicado al estudio y aplicación de un determinado paradigma. Se desarrollaron pruebas y se obtuvieron resultados cuantitativos y asimismo se realizaron observaciones y entrevistas que posibilitaron un análisis cualitativo del tema estudiado. Se hicieron los análisis correspondientes y recopiló toda la experiencia en un documento que es parte constitutiva del proyecto de investigación y, de allí, se derivó este artículo.

Toda esta experiencia fue desarrollada con los estudiantes de las asignaturas Programación I, Programación II y Programación III de Ingeniería de Sistemas y Computación de la Universidad Tecnológica de Pereira desde el primer semestre de 2010 hasta el segundo semestre de 2012. Todo el proceso se observó, se documentó y se evaluó en cada uno de los pasos de investigación. Es de anotar que en el programa Ingeniería de Sistemas y Computación, la asignatura Programación I tiene por contenido el paradigma de programación funcional con DrScheme como lenguaje de programación; la asignatura Programación II tiene por contenido el paradigma de programación estructurada con Lenguaje C como lenguaje de programación; y la asignatura Programación III tiene por contenido el paradigma de programación orientada a objetos con Java como lenguaje de programación.

Tanto el presente artículo, como subproducto 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", como las reflexiones asociadas se basan en la hipótesis de que siempre es posible siempre encontrar caminos que faciliten los objetivos de aprendizaje en cualquier temática, independiente de ella, y que estos tienen que ver con las posibles relaciones que se establezcan entre un área de conocimiento específico y los conocimientos previos y nuevos conocimientos asociados con dicha área.

Para explicar todo el proceso investigativo que inspira este artículo se acude a una introducción del tema, se presentan las bases teóricas que fundamentan la investigación, se explica la metodología utilizada tanto en su concepción como en su aplicación, se presentan los resultados cuantitativos y cualitativos, se plantean los elementos de juicio que se tuvieron en cuenta para la discusión y finalmente se formulan unas conclusiones terminando con las referencias bibliográficas que sirvieron de base para el desarrollo del artículo.

2. Teoría

A continuación se presenta una aproximación tanto a 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 un esbozo general de los paradigmas que se intentan relacionar en el contexto del presente artículo.

2.1 Aprendizaje significativo

Con el nombre de aprendizaje significativo se conoce la teoría formulada por el Dr. 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 conocimiento. 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 clara, el cerebro inmediatamente genera una reacción de este en la búsqueda de lo que significa, bien a través de los patrones que tenga almacenados, bien a través de la información útil que haya recibido a través de los sentidos o bien a través de la información simple (de uso poco frecuente) que haya guardado en la memoria a corto plazo.

Por lo dicho, 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 (Ausubel, 1986). Un patrón es un modelo informacional (o de conocimiento) que se usa con frecuenta 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 llega a través de los sentidos y que el cerebro aún no ha clasificado como información útil, bien porque no le haya encontrado significado o bien porque no es de uso frecuente.

El autor de la teoría, Dr. 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 a esto, el Dr. 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), y con esto le está dando alta prioridad al conocimiento adquirido previo al inicio de un proceso de aprendizaje.

De acuerdo a esto, el nuevo conocimiento lo constituye el conjunto de saberes que el alumno aún no había tenido oportunidad de acceder a ellos, 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 (Piaget, 1986). Es de aclarar que lo nuevo solo es nuevo para determinada persona, así no sea necesariamente nuevo.

Acorde con la teoría, 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.

Por lo tanto, el modelo que estableció el Dr. Ausubel para fundamentar su teoría del aprendizaje significativo determina que "…el ser humano aprende mucho más fácil todo aquello que tiene significado para él" (Ausubel, 1986), y parte del significado como esencia de dicha teoría basado en el conocimiento previo, el nuevo conocimiento y la actitud del estudiante.

2.2 Aprendizaje por descubrimiento

Tomamos 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 el Dr. 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 cual 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.

De acuerdo a su autor, el Dr. 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 (Bruner, 1969). En el proceso de aprendizaje, como se plantea en la teoría del aprendizaje por descubrimiento, se involucran tres procesos casi simultáneos: de 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"; finalmente está el proceso que permite comprobar si la manera con que hemos manipulado la información es adecuada a la tarea, etapa que el autor de la teoría calificó como "evaluación" (Bruner, 1991).

Por lo tanto, en relación con la teoría de aprendizaje significativo, puede definirse 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 (Piaget, 2001); 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.

Por último, 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), 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.

2.3 Programación funcional

Según lo estudiado, 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 (Guzmán, 2005). 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.

Acorde con la teoría, el paradigma de programación funcional aborda la construcción de soluciones a partir de tres conceptos básicos: simplificar el objetivo a 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 la simplificación en los frecuentes casos en los cuales el objetivo resulta tener un cierto nivel de complejidad.

Por lo tanto, la realización de las pruebas de escritorio sigue siendo el mecanismo excelso para la comprobación y detección de errores en la construcción de un programa. Las pruebas de escritorio exigen el seguimiento lineal de las instrucciones que se han incorporado dentro de un programa y la verificación de sus resultados. En tiempos modernos, las pruebas de escritorio se han facilitado a partir de la incorporación de los botones Debug en los IDE (Integrated Development Environment) que son los ambientes de desarrollo que se han construido para facilitar la tarea de transcripción, digitación, edición, compilación y ejecución: todo en uno (Van Santen, 2010).

Según esto, el método tradicional de la realización de las pruebas de escritorio utiliza papel y lápiz y, cuando un programa se basa en funciones, realizarlo posibilita que las pruebas sean mucho más confiables, que estas sean menos agotadoras y que, por lo tanto, sean mucho más ágiles. Factores de gran impacto e importancia en el proceso de construcción de soluciones que pueden ser implementadas a través de un computador.

Por tanto, 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 ya han funcionado apropiadamente sino que también ya han sido probadas en tiempo de ejecución. De 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 asimismo pueda ser la programación basada en este paradigma.

Los tiempos modernos exigen que el desarrollo de software y, con él, la construcción de programas incluya un ingrediente de alta importancia como es el buen uso del tiempo y, 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.

2.4 Programación estructurada

Corresponde al primer paradigma formal de programación de computadores que se conoció como la programación estructurada dado que es un mo-

delo 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.

No obstante, el estudio exhaustivo de los programas realizados a partir de la llamada programación libre permitió ir hallando que todos los programas se encontraban y hacía uso de tres estructuras específicas y, 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 y, 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.

Tal 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 (MacKay, 2005). Este tipo de programación también se conoce como programación imperativa aunque algo este concepto es compartido con otros paradigmas.

Según esto, 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 (Schildt, 2000). 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 pero en otros ambientes (tal vez distribuidos o multiprocesados) posibilitan o 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 (Trejos, 2005).

De otra parte, la estructura de decisión (o condicional) permite que se pueda escoger uno de 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 es 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.

Por último, la estructura "cíclico" o "iterativa" permite que se pueda ejecutar un conjunto de varias instrucciones tantas veces como una condición lo permita de manera que su evaluación permita, 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 caracte-

rísticas 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 (Trejos, 2002). 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.

A partir de la consolidación de la programación estructurada como principal ejemplo del paradigma imperativo surgieron en el mercado una gran cantidad de lenguajes que, incluso hoy, a más de cincuenta años de su aparición, siguen vigentes. Tal es el caso de Fortran, Cobol, Pascal, Logo y C que han logrado mantener en el mercado la esencia de la programación estructurada bien ampliándola o bien mejorándola o, mejor aún, evolucionando hacia otro paradigma pero partiendo de los elementos que caracterizan el paradigma imperativo.

2.5 Paradigma orientado a objetos

Corresponde a uno de los paradigmas más aplicativos y más modernos de la programación. Se fundamenta en la interpretación del mundo a partir del concepto de clases que, conceptualmente, son una declaración descriptiva de todo lo tangible e intangible a partir de sus características y de sus usos (Deitel, 2010). En este paradigma, las características se conocen como atributos y los usos se conocen como métodos. La instanciación de una clase se conoce como objeto de donde se deriva el nombre que se ha universalizado.

El paradigma de programación orientada a objetos aparece como la solución a aquellos problemas que la programación estructurada no había podido resolver, así como la necesidad de aproximarse un poco más al mundo real concibiéndolo en sus características y no haciendo reinterpretaciones que no siempre se ajustaban a la realidad, como sucedía en otros paradigmas (Lemay, 2008). Bajo la lente de la programación orientada a objetos todo lo que nos rodea puede ser descrito por sus características o atributos y por sus usos o métodos.

De esta manera, a diferencia de la programación estructurada, en el paradigma orientado a objetos, dos entes informáticos son iguales si el conjunto completo de sus atributos y el conjunto completo de sus métodos son exactamente iguales (Van Roy 2003). Esto abrió un camino para la interpretación del mundo y para la expansión de dicha interpretación debido a que se fueron encontrando nuevos usos a objetos que ya se habían concebido en la realidad y que aún no se habían pensado desde la óptica que proporciona este paradigma.

Los atributos también expandieron la interpretación del mundo permitiendo que muchas de las características de diferentes objetos que aún no habían sido consideradas por su utilidad, comenzaran a ser consideradas, pues se llegó a la conclusión de que aportaban lo suficiente como para tenerlas en cuenta. De esta forma, un objeto tan simple como un lapicero comenzó a tener una interpretación en sus atributos y en sus métodos.

Un lapicero puede tener atributos como peso, color, material, textura, sabor, etc., y de la misma forma, puede servir para diferentes usos como escribir, marcar, señalar, llamar la atención, despertar a alguien, etc. Como puede verse, el paradigma de programación orientada a objetos permitió una visión mucho más aproximada al mundo y una interpretación mucho más exacta de dicho mundo dentro de los sistemas computacionales. La irrupción de este paradigma generó todo un movimiento nuevo en la programación a partir del cual se fueron popularizando, incluso, nuevos conceptos en programación tales como la herencia, la expansión, el polimorfismo y el encapsulamiento, por nombrar apenas unas de las más representativas.

La popularización de este paradigma, producto de su gran aplicabilidad y de la manera como permitía realizar aplicaciones que simplificaban los objetivos, reusaban el código y facilitaban las pruebas, llevó a que la tecnología se volcara hacia un lenguaje de programación que aprovechó al máximo las características de dicho paradigma como es el lenguaje Java. Sin desconocer la presencia de otros lenguajes

de gran fortaleza en el mundo de la programación orientada a objetos como C++, Delphi y Visual Basic, el lenguaje de programación Java tomó la delantera posiblemente por su alta portabilidad y, particularmente, por las relaciones que se pudieron establecer entre los diferentes sistemas operativos para lo cual Java, con la idea de su máquina virtual, resolvió el problema de las diferencias y, en cambio, aprovechó las similitudes en pro de un rendimiento mayor y de una portabilidad superior.

Ha sido tal la penetración del lenguaje de programación Java y del paradigma orientado a objetos que la programación de los dispositivos móviles está tendiendo a aprovechar todo el potencial del lenguaje Java en el sistema operativo que mas tiende a popularizarse como es el sistema operativo Android. Aunque no se puede desconocer ni las bondades de otras expresiones tecnológicas ni de otros paradigmas, la programación orientada a objetos ha ido ganando muchísimo terreno al punto que, en la actualidad, son muchas las instituciones educativas que han tomado este paradigma como contenido del primer contacto que tienen los estudiantes en sus programas de ingeniería.

3. Metodología

3.1 Descripción

Tal como se ha expuesto, hasta el momento, el contenido de este artículo relata la experiencia pedagógica alrededor de un concepto núcleo común entre los tres paradigmas, su manera de presentarla a los estudiantes y sus efectos dentro del contexto del avance a nivel de contenido y a nivel de aprendizaje de la programación. Desde lo puramente pedagógico, dos elementos cobran importancia en esta experiencia: la relevancia que tiene para los estudiantes el saber que conocimientos previos sirven de fundamento para el desarrollo de nuevos conocimiento y el significado que los conocimientos previos le dan al nuevo conocimiento (teoría del aprendizaje significativo – Dr. David Paul Ausubel) y la motivación que imprime la fascinación y lo nuevo cuando el mismo alumno es conducido por caminos por los cuales descubre nuevos conocimientos que tienen íntima relación con lo que ya saben (aprendizaje significativo – Dr. Jerome Seymour Bruner).

En este sentido, el concepto que se promueve en los tres paradigmas expuestos hace referencia a que la programación de computadores consiste en la construcción de soluciones que resuelven problemas de la sociedad, intentando interpretar la realidad, a partir de las posibilidades que brindan los avances tecnológicos y específicamente los paradigmas de programación, los lenguajes de programación y los dispositivos electrónicos modernos. Bajo esta concepción, los tres paradigmas encuentran un primer punto de encuentro en cuanto a su propósito y a los alcances que, a partir de ellos, pueden existir.

En la sección anterior se proporcionó una aproximación a los tres paradigmas (funcional, estructurado y orientado a objetos) en donde se develan sus principales características. De ellas se podría destacar en el paradigma funcional el concepto de función como piedra angular, en el paradigma estructurado la existencia de las tres estructuras básicas (secuencias, decisiones y ciclos) y en el paradigma orientado a objetos la definición de atributos y métodos para construir clases y, con ello, poder definir objetos.

A esto se le suma que en el paradigma funcional la interpretación del mundo acude a los métodos que se involucran en su funcionamiento, en el paradigma estructurado se destaca la interpretación de dichos procesos, pero a la luz de las estructuras y en el paradigma orientado a objetos, se resalta la interpretación del mundo a partir de objetos (instancias que se distinguen por sus atributos –características- y sus métodos –usos-). Tal como se muestra en la Tabla 1.

Ahora bien, un punto de encuentro de los tres paradigmas es el hecho de que todos, como paradigmas de programación, intentan interpretar y replicar el funcionamiento del mundo a partir de las posibilidades que brinda la tecnología. Esto nos lleva a pensar en que se puede expandir la tabla expuesta si se tiene en cuenta que en el paradigma funcional

Tabla 1. Características principales de los paradigmas

Paradig	Funcional	Estruct	Orient a Objetos
Caract.	Función	Estructuras	Atributos y métodos
Interpretación del mundo	Métodos que se involucran en el funcionamiento. del mundo	Visión del mundo a partir de estructuras	El mundo se interpreta como una colección de objetos

los métodos que se involucran en el funcionamiento del mundo corresponden a funciones (si se lleva a la minucia propia de los lenguajes de programación); en el paradigma estructurado, el concepto de función también es posible y, de hecho, la misma programación imperativa se hace mucho más sencilla cuando se acude a dicho concepto capitalizando al máximo las tres estructuras.

En el paradigma orientado a objetos, así como los atributos pueden asociarse con el concepto de variable propio del paradigma estructurado y que, en el fondo, continúa siendo el mismo con algunas pequeñas variantes, los métodos se implementan a partir de funciones cuyas características no son tan distantes de las que se usan en los otros paradigmas y, dejando en claro eso sí, que adoptan nuevas formas y conceptos para que el paradigma se haga efectivo. Bajo este marco la Tabla 2 muestra el nuevo concepto incorporado.

De esta manera se encuentra un núcleo común a los tres paradigmas, el cual es el concepto de función, en sus tres formas:

a) Función del paradigma funcional como base para la construcción de soluciones que interpreten el comportamiento del mundo.
b) Función del paradigma estructurado como base para la construcción de soluciones que aprovechen al máximo las tres estructuras básicas.
c) Función del paradigma orientado a objetos como concepto y mecanismo para la construcción de métodos que, asociados con los atributos, permitan el aprovechamiento al máximo de la definición de clases y orientación a objetos.

Esto pone el concepto de función, en cualquiera de sus tres sabores, como el núcleo en el cual los tres paradigmas pueden converger y a partir del cual se puede asimilar más fácilmente las bondades y ventajas de cada uno de ellos.

A todo esto se le puede adicionar que, en los tres paradigmas, la estructura general de una función es altamente similar, pues una función consta (en su más simple esencia) de un nombre de función, seguido de argumentos o parámetros para que la función logre su objetivo, luego un indicador de inicio seguido por el cuerpo de la función y que termina con el finalizador de la función, tal como se muestra en la Figura 1.

Tabla 2. Expansión de la Tabla 1

Parad.	Funcional	Estruct	Orientado a Objetos
Caract.	Función	Estructuras	Atributos y métodos
Concepto Común	F u n c i ó n		
Interpretación del mundo	El mundo se concibe a través de funciones que describen su comportamiento	Se construyen funciones que capitalizan al máximo las tres estructuras	Los métodos se plantean como funciones aprovechando las bondades de la orientación a objetos

Nombre de la función (argumentos o parámetros).
Inicio de la función.
Cuerpo de la función.
Fin de la función.

Figura 1. Estructura general de una función.

Aunque debe aceptarse que dentro de cada paradigma este esquema de función tiene unas pequeñas variantes (como los tipos de datos, la declaración de variables, los polimorfismos, los retornos, etc.) no se puede negar que este esquema resulta ser muy útil en los tres paradigmas y que la aproximación conceptual que existe tanto en su utilidad como su formulación es un camino que facilita significativamente el aprendizaje de la aplicación de los tres paradigmas en el desarrollo de programas.

Es de anotar que el cuerpo de la función está constituido por el conjunto de instrucciones que permiten que la función logre su objetivo, teniendo en cuenta que una función es un conjunto de instrucciones, distinguible con nombre propio y argumentos o parámetros para su funcionamiento, que logra un objetivo específico.

El concepto de función, al margen de las características de cada paradigma, permite que se enfrenten los tres problemas grandes de la programación como son la simplificación del objetivo de una aplicación, la realización fácil y confiable de las pruebas de escritorio y la reutilización del código existente.

Es de anotar que encontrar este concepto común tiene poca relevancia si no se lleva apropiadamente al contenido de las asignaturas en las cuales se estudia la programación de computadores, sus paradigmas y sus expresiones tecnológicas a través de los lenguajes de programación, dentro de un proceso formativo como sería el caso de la ingeniería de sistemas.

Si bien el orden en que se presenten los paradigmas a los estudiantes podría no ser importante (pues corresponde a una discusión presentada en otro artículo), se presenta en la Tabla 3 una propuesta basada en la secuencia de tres asignaturas.

Puede notarse que son más las similitudes que se presentan en los contenidos de los cursos de programación de las que podrían pensarse y que la esencia de cada contenido está reflejada en dos conceptos: el paradigma de programación que subyace a cada curso y el concepto de función, como núcleo central presente en los tres paradigmas y gran herramienta que simplifica la programación y facilita el aprendizaje no solo de los paradigmas sino de sus características y de sus aplicaciones a la luz de los lenguajes de programación.

Es de anotar que otro concepto común a los tres contenidos debería ser la metodología de programación, es decir, la manera como se aborda un problema, cómo se analiza y cómo se resuelve, todo ello a la luz de un paradigma.

Tabla 3. Propuesta de incorporación del concepto de función.

	PROG. I	PROG. II	PROG. III
Paradigma	Funcional	Estructurado	Orientado a Objetos
Temas Centrales	El concepto de función recursividad	El concepto de función librerías apuntadores	El concepto de función atributos métodos
Contenido General	Presentación de paradigma argumentos metodología de program. El concepto de función condicionales recursividad listas vectores recursos gráficos almacenam. herramientas adicionales.	Presentación de paradigma variables metodología de program. El concepto de función y librerías condicionales recursividad apuntadores arreglos archivos recursos gráficos herramientas adicionales.	Presentación de paradigma atributos metodología de program. funciones y métodos condicionales polimorfismo y sobrecarga herencia arreglos recursividad almacenamiento herramientas adicionales.

3.2 Aplicación

Esta estrategia se ha utilizado en los cursos Programación I, Programación II y Programación orientada a objetos del ingeniería de sistemas y computación de la Universidad Tecnológica de Pereira desde el primer semestre de 2011 hasta el segundo semestre de 2012, lo cual significó que se tuvieron dos cohortes de análisis completas dado que los estudiantes que iniciaron en este proceso en la asignatura Programación I (en el primer semestre de 2011) fueron los que llegaron, dentro del proceso, a la asignatura Programación orientada a objetos en el primer semestre de 2012 y los que iniciaron este proceso (en el segundo semestre de 2011) en la asignatura Programación I fueron los que llegaron, dentro del proceso, a la asignatura Programación orientada a objetos en el segundo semestre de 2012.

La relación entre los conceptos previos adquiridos en cada una de las asignaturas y su conexión con los nuevos conceptos presentados en la asignatura siguiente, fue un motor de alto significado de los estudiantes puesto que aquellos que habían asimilado los conceptos de un paradigma pudieron afianzar dichos conceptos en el paradigma siguiente y, de paso, asimilar los conceptos propios de dicho paradigma.

Los estudiantes que tenían conceptos débiles en un paradigma, y que aún así habían logrado la cualificación para avanzar en las asignaturas, pudieron repasar y reforzar los conceptos débiles pudiendo encontrar un camino para clarificar tanto el paradigma anterior que hubieran visto como el paradigma actual que estuvieren viendo. Aquellos estudiantes que tenían muy débiles los conceptos y que no alcanzaron la cualificación para continuar en un curso, encontraron la gran importancia académica de tener claros los conceptos básicos puesto que ese núcleo temático (el concepto de función) era de vital importancia para los paradigmas y las asignaturas subsiguientes dentro de su proceso formativo. Es de anotar que esta propuesta, ya en escena, parte del principio de que los estudiantes sean conscientes de la metodología que se está usando, de sus beneficios y de que ellos participen activamente en el desarrollo de actividades que propendan por su aprendizaje acorde con lo planteado en esta estrategia.

4. Resultados

En lo cuantitativo los resultados fueron bastante prometedores. Para ello se realizaron las mismas pruebas, tanto en los cursos donde se estaba siguiendo esta metodología, como en los cursos en donde no se siguió la metodología, es decir, cursos en donde se vieron los contenidos de las asignaturas sin que se presentara un tema específico que sirviera como nexo conceptual entre los tres paradigmas.

En los cursos en donde el concepto de función se utilizó como nexo temático entre los paradigmas, los resultados cuantitativos fueron mejores que en los otros cursos y, a decir, por los mismos alumnos, lo que facilitó la apropiación de los conceptos de un determinado paradigma fue la relación que se estableció entre lo que ya sabían y lo nuevo que estaban viendo, es decir, en palabras de Ausubel, la relación entre el conocimiento previo y el nuevo conocimiento. Una tabla resumen de los promedios de los cursos y su comparativo puede verse en las Tablas 4 y 5. Es de aclarar que las notas presentadas corresponden al promedio general de cada curso.

Tabla 4. Resultados Cuantitativos Comparativos (1)

Periodo de investigación	I Sem 2011	II Sem 2011	I Sem 2012	II Sem 2012
Asignaturas	Prog I	Prog II	Prog OO	
Con metodología propuesta	3.9	4.2	4.4	
Sin metodología propuesta	3.4	3.8	3.9	

Tabla 5. Resultados cuantitativos comparativos (2).

Periodo de investigación	I Sem 2011	II Sem 2011	I Sem 2012	II Sem 2012
Asignaturas		Prog I	Prog II	Prog OO
Con metodología propuesta		4.0	4.2	4.3
Sin metodología propuesta		3.5	3.5	4.1

Sin desconocer que este tipo de estrategias pueden ser refinadas con una aplicación más amplia en el tiempo, lo cual permitiría consolidar algunas conclusiones, no deja de llamar la atención el hecho de verificar que los resultados cuantitativos (junto con la opinión de los estudiantes y el análisis cualitativo de su desempeño académico) parecieran indicar una tendencia que posibilita pensar en que la estrategia utilizada es la apropiada y que esta facilita un camino para la asimilación, apropiación y aplicación de los conceptos propios de cada paradigma pero partiendo del tema que las une a todas: el concepto de función.

5. Discusión

En general, en lo cualitativo, se ha percibido una gran motivación en los estudiantes especialmente en aquellos que van avanzando a través de las asignaturas. Según sus propias palabras, los estudiantes encuentran muy importante el hecho de que se les comuniquen los objetivos de aprendizaje, las estrategias, los logros propuestos y la metodología que subyace a todo ello. De esta manera ellos mismos se comprometen con lo que deben aprender y, según los resultados, así pareciera.

Vale la pena recordar que, en todo momento en la aplicación de esta estrategia de aprendizaje dentro del marco de la metodología planteada, se ha sobrepuesto lo metodológico a lo técnico dado que es el paradigma de programación el que subyace al lenguaje de programación y no lo contrario. Esto lleva a pensar entonces que se prioriza lo humano sobre lo tecnológico, en concordancia con lo dicho.

Para un estudiante de programación, y tal como lo establecen las teorías de aprendizaje citadas en este artículo, resulta de gran importancia encontrar conexiones entre el nuevo conocimiento y el conocimiento previo pues eso les imprime significado al saber así como "descubrir" que un tema de la programación subyace a los paradigmas facilitando su aprovechamiento y su aplicación de una manera más sencilla.

En los resultados cuantitativos se percibe una tendencia a mejorar por parte de los estudiantes en la medida en que avanzan en los paradigmas y, de paso, en las asignaturas puesto que, bajo esta técnica, cada uno refuerza los conceptos anteriores y prepara el camino para los nuevos conceptos. Es de anotar que todo esto está supeditado a los estilos de enseñanza de los docentes y que, si bien los comparativos en los cursos no puede hacerse de una manera lineal, si es posible interpretar lo que dicen los datos en el sentido de pensar que la metodología presentada surte unos efectos positivos en pro del proceso de aprendizaje.

Una de las ventajas que tiene esta metodología es que le muestra a los alumnos un camino e conexión entre los paradigmas de programación, algo tan simple y tan escaso en los cursos de programación en las universidades dado que normalmente cada curso se presenta de manera independiente y completamente inconexa con las otras vertientes paradigmáticas y con los otros lenguajes de programación. La relación entre el conocimiento previo y el nuevo conocimiento definitivamente afianza los conceptos y, con ello, posibilita mucho más que se alcancen los logros de aprendizaje.

Durante el periodo de prueba se desarrollaron permanentemente ejercicios, acudían al concepto de función como eje temático central pero que aterrizaba sus posibilidades en el horizonte propio de cada paradigma. Esta propuesta queda, sin embargo, supeditada al estilo de enseñanza de cada docente, al nivel con que el docente tenga claro los conceptos fundamentales de cada paradigma, a la experiencia que cada docente haya tenido con los lenguajes de programación, a la empatía que se dé entre un docente y los estudiantes de su curso y a las estrategias a las que el docente acuda para que los alumnos apropien esta metodología propuesta.

6. Conclusiones

- Es posible encontrar un camino más expedito que facilite el aprendizaje tanto de los paradig-

mas de programación de computadores como la apropiación de los lenguajes de programación.

- Existe un tema que une a los tres paradigmas seleccionados (paradigma funcional, paradigma estructurado, paradigma orientado a objetos) que permite asimilar fácilmente sus conceptos asociados, y es el tema de funciones.
- Es posible encontrar en las funciones el camino para que se pueda avanzar fácilmente en el conocimiento profundo de los lenguajes de programación, de su sintaxis, de sus características, de sus bondades, de sus recursos y de sus "caprichos" sintácticos.
- Todo tema, por complejo que sea, se simplifica significativamente si se acude al concepto de función.
- El significado de la programación va íntimamente ligado al concepto de función si las condiciones profesionales, humanas y personales del docente así lo permiten y si él así lo quiere para su estudiantes
- Los tres problemas fundamentales de la programación quedan claramente resueltos a partir del concepto de función

Referencias

1. Ausubel, P. (1986). Sicología Educativa: Un punto de vista cognoscitivo. México, Trillas.
2. Bruner, J. (1963). El proceso de la Educación. México, Hispanoamericana.
3. Bruner, J. (1969). Hacia una teoría de la instrucción. Editorial Hispanoamericana. México.
4. Bruner, J. (1991). Actos de significado. Alianza Editorial. Madrid.
5. Deitel. P. (2010). Java How to program. Deitel Inc. Prentice Hall. USA
6. Guzman, H. (2005). Introducción a la programación con Scheme. Editorial Tecnológica de Costa Rica.
7. Lemay, L. (2008). Teach yourself Java in 21 days. Sams Net Publishing. USA.
8. Mackay, D. (2005). Information Theory, Inference and Learning Algorithms. Cambridge University Press. Londres, UK.
9. Piaget, J. (1986). Inteligencia y afectividad. Buenos Aires, Aique.
10. Piaget, J. (2001). Psicología y Pedagogía. México, Crítica.
11. Schildt. H. (2000). The Complete Reference C. Osborne McGraw Hill. USA.
12. Trejos B, O. (2005). Fundamentos de Programación. Editorial Papiro. Pereira. Colombia.
13. Trejos B., O. (2002). La esencia de la Lógica de Programación. Centro Editorial Universidad de Caldas. Manizales. Colombia.
14. Van Roy, P. (2003). Concepts, Techniques and Models of Computer Programming. Swedish Institute of Computer Science. Estocolmo. Suecia.
15. Van Santen, R. (2010). 2030 Technology that will change the world. Oxford University Press.