La programación orientada a objetos facilidad para crear

- Object-oriented programming: easy to create

Mag. Eilen Lorena Pérez Montero (1)
Corporación Universitaria del Huila-CORHUILA, Colombia
eilen.perez@corhuila.edu.co

Mag. Flor de María Hernández Pérez (2)
Servicio Nacional de Aprendizaje- Tecnoparque Nodo
Neiva, Colombia
flhernandezp@sena.edu.co

Fecha de Recepción: 23 de Octubre de 2019 / Fecha de Aprobación: 3 de Noviembre de 2019

Resumen: En este artículo se presenta un estudio inicial relacionado con el concepto de programación y la relación directa con el entorno real. También da a conocer conceptos de herencia y como aplicarlo por medio de un ejemplo sencillo, así como también se explora una línea de tiempo de los lenguajes de programación hasta el año 1995 y el auge de la parte móvil en estos tiempos. El documento está estructurado de la siguiente manera, se inicia con una introducción, posteriormente se estudia el concepto de la programación como proceso, para luego hablar específicamente sobre el Paradigma Orientado a Objetos POO, el Concepto de Herencia en POO y los Lenguajes de Programación y se finaliza con unas conclusiones al respecto.
Palabras clave: Herencia, Lenguajes de programación, Programación orientada a objetos.

Abstract: This article presents an initial study related to the concept of programming and the direct relationship with the real environment. It also introduces concepts of inheritance and how to apply it through a simple example, as well as exploring a timeline of programming languages up to 1995 and the rise of the mobile part in these times. The document is structured in the following way, it begins with an introduction, then the concept of programming as a process is studied, and then specifically talks about the Object Oriented Paradigm POO, the Concept of Inheritance in POO and the Programming Languages and It ends with some conclusions in this regard.
Keywords: Inheritance, Object-oriented programming, Programming languages.

1. Introducción:

La programación expande la mente, ayuda a pensar en forma sistémica y ordenada, permitiendo que las personas mejoren y automaticen tareas que realizan en sus trabajos de la vida diaria. Es una actividad que implica un proceso mental, generalmente complejo y creativo que exige inteligencia, conocimiento, habilidad y disciplina [1]. Este acto de programar parte de un problema expresado en lenguaje natural, que enuncia límites y modela una solución en un lenguaje computacional.
La implementación de los lenguajes computacionales demanda un grado de rigurosidad en la concepción de los proyectos de desarrollo, debido a la importancia del diseño de software y a la ingeniería de requerimientos (funcionales y no funcionales) generando beneficios en la codificación y en la disminución de riesgos [3].

Es evidente que la innovación tecnológica conduce a evolucionar los lenguajes computacionales en el desarrollo de sistemas o software, con el objetivo de proporcionar al usuario el manejo de la información y un número variable de tareas ejecutadas con un ahorro significativo de tiempo, dando pronta respuesta a las exigencias que se originan en las más variadas esferas de la sociedad.

Dentro de la evolución de los lenguajes de programación, se tienen los de muy bajo nivel con dificultades de entendimiento y de migración de datos a otra máquina, hasta llegar a los de alto nivel de fácil integración de datos, comprensión y más parecido al lenguaje natural humano.

Intentando solventar estas deficiencias y la facilidad para leer y comprender el problema a la hora de programar, surge alrededor de los años 60 el modelo de la programación orientada a objetos por Kristen Nygaard y Ole-Johan Dahl, formada por una colección de objetos que interaccionan conjuntamente para representar y solucionar algún problema. Este paradigma difiere de la

(1) Ingeniera de Sistemas y Magíster en Tecnologías de la Información aplicadas a la Educación de la Universidad Pedagógica Nacional, Especialista en Edumática de la Universidad Autónoma. Actualmente, es Docente de la Corporación Universitaria del Huila CORHUILA y pertenece al Grupo de Investigación Procing de la misma Universidad.
(2) Ingeniera de Sistemas, Magister en Computación, Especialista en Soluciones y Desarrollo Informático de la Universidad del Cauca. Gestora de la línea de tecnologías virtuales del Tecnoparque nodo Popayán e integrante del Grupo de Investigación IDIS de la Universidad de Cauca.

programación imperativa o estructurada donde los datos y los métodos están separados y sin relación alguna, haciendo uso de los procedimientos y el mantenimiento de tantas líneas de código [12].

2. La programación como proceso:

Teniendo en cuenta que la programación permite encontrar diversas maneras de abordar problemas y de plantear soluciones, que necesitan un proceso de diseño y de requerimientos, existen lenguajes computacionales de alto nivel que implementan el modelo de programación orientado a objetos como una propuesta adecuada surge la siguiente pregunta ¿Por qué no enseñar a que estas nuevas generaciones construyan sus propias herramientas con el modelo de programación orientado a objetos para dar solución a los problemas? Este contexto es por lo tanto, un importante desafío donde se hace necesario implementar la programación de las computadoras y la combinación de metodologías en el desarrollo de software; herramientas fundamentales en las diferentes ramas del conocimiento, y particularmente en la ingeniería.

3. Programación orientada a objetos:

El modelo de programación orientado a objetos denominado POO expresa un programa como un conjunto de objetos que colaboran entre sí para realizar tareas [4]. Un objeto es una entidad, sujeto o cosa que se encuentra en situaciones o problemas de nuestro mundo real, formados por datos que representan la estructura del objeto y los métodos que implementan las operaciones que se debe realizar sobre los datos [5].
Un conjunto de objetos se representa abstractamente por una Clase, que tienen en común una misma estructura y un mismo comportamiento, las clases se parecen a moldes donde se definen la forma de los objetos. Lo que significa que un objeto es un ejemplar de una clase. Así mismo el modelo de programación orientada a objetos presenta características de encapsulamiento, herencia y polimorfismo [9].

Entre los lenguajes que soportan el modelo de programación orientada a objetos están Smalltalk, C++, Delphi (Object Pascal), Java y C#, lenguajes que han incorporado al mercado y han ido creciendo y mejorando poco a poco utilizando y desechando características de otros lenguajes que perecieron durante el proceso.

Este nuevo escenario de programación simplifica la escritura, mantenimiento y la reutilización, razón por la cual este escrito expone las posibilidades de la Herencia en la POO. Para ello es importante señalar que a la hora de resolver un problema con el modelo de la programación orientada a objetos se debe inicialmente hacer un análisis para tener claridad de los datos de entrada, proceso y salida; seguidamente se identifica los objetos presentes y sus relaciones en el dominio del problema. La idea fundamental es jerarquizar clases, equiparándose con las utilidades que debe proporcionar cada objeto que lo compone, en especial aquellos que el usuario debe usar más; así la funcionalidad logra tener elementos comunes; para especializar o extender la funcionalidad de una clase, derivando de ellas nuevas clases.
De esta manera, la jerarquización hace parte de un todo, los objetos derivados son objetos descendientes con características y procedimientos en común [11]. Este principio se denomina Herencia y su beneficio se centra en la transmisión de código entre unas clases y otras.
En otras palabras, es la relación entre dos o más clases donde una es la principal superclase y otras son secundarias llamadas clases, donde al mismo tiempo los objetos adquieren las propiedades de los otros.
Para argumentar el concepto de herencia, se tiene un ejemplo concreto de una empresa compuesta por dos grupos de empleados que devengan su salario de dos formas distintas: por hora laborada y de forma integral o asalariado. Al diseñar la estructura de clases (diagrama 1) se tiene dos objetos: uno denominado EmpleadoporHora con los datos nombre del empleado, departamento en el que labora, cargo que desempeña, número de horas laboradas, valor de la hora trabajada, salario mensual horas (nombreEmp, deptoEmp, cargoEmp, nhoraslab, vhoraslab,salmensualH) y otro objeto llamado EmpleadoAsalariado con los datos nombre del empleado, departamento en el que labora, cargo que desempeña, salario mensual asalariado (nombreEmp, deptoEmp, cargoEmp,salamensualS).

Fig. 1.
Estructura de clases EmpleadoporHora y EmpleadoAsalariado

Es notorio visualizar en el diagrama 1 que los dos objetos tienen en común los datos nombreEmp, deptoEmp, cargoEmp, conduciendo a dos clases relacionadas de la misma manera, lo que permite construir el diagrama de jerarquía de objetos y crear una tercera clase llamada Empleado. Esta clase se le conoce como superclase y se derivan dos clases, denominadas subclases que son EmpleadoporHora y EmpreladoAsalariado (Diagrama 2).

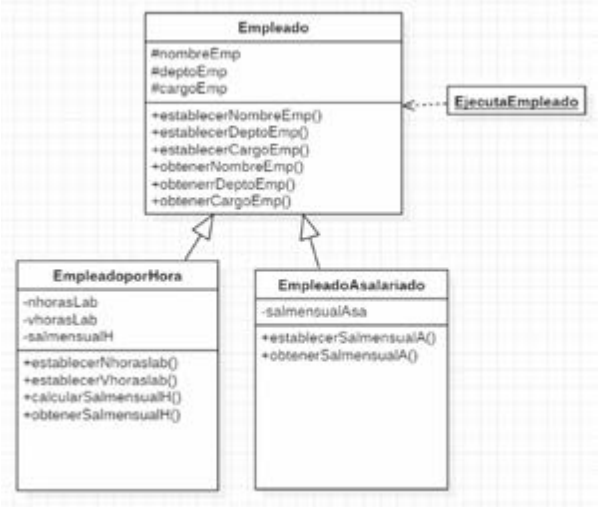

Fig. 2.
Diseño del diagrama de clases con herencia

En este ejemplo se especifica que las clases EmpleadoporHora y EmpreladoAsalariado heredan de la clase Empleado, es decir, EmpleadoporHora y EmpreladoAsalariado podrán acceder a las características y métodos de Empleado.

4. Concepto de herencia en Poo:

La herencia constituye un valioso concepto en el modelamiento de la programación orientada a objetos debido a la organización de las acciones en diferentes clases, permitiendo a los objetos ser definidos y creados como tipos especializados de objetos preexistentes. El nuevo objeto puede definir nuevos atributos, nuevas operaciones, y redefinir las operaciones heredadas si se considera necesario.

Se hace entonces necesario que el desarrollador incorpore el modelo orientado a objetos por su flexibilidad para expresar y dar soluciones a innumerables situaciones que de otra forma se habrían convertido en un reto. Como valor agregado para el programador se incluye la reducción de los tiempos de desarrollo ya que al integrar la herencia, unifica características propias que tienen ciertos objetos, permitiendo que el código no se aplique en un solo caso, sino que sea reutilizable, y al ser una manera de programar permite su implementación en las distintas tecnologías de programación.
Por consiguiente, se concluye que la tecnología orientada a objetos ofrece múltiples ventajas al desarrollo del software resaltando el concepto de herencia que permite mitigar dificultades relacionadas con la gestión en el proceso de desarrollo cuando se realiza una correcta jerarquía de clases coherente en las etapas de análisis y diseño de un sistema.

5. Lenguajes de programación:

Al observar e interactuar con un lenguaje de programación se puede deducir que simplemente es un conjunto que contiene elementos gramaticales y semánticos que sirven para definir estructuras validas para dar solución a un problema específico. La tecnología avanza, lo que indica que los lenguajes de programación han ido evolucionando, lo que es positivo por la mejoras en arquitectura, facilidad en procesos de conexión con gestores de base de datos, autoayuda para procesos, generación de código automático, etc. El avance ha sido significativo, porque si se analiza y haciendo una trazabilidad de los primeros lenguajes comprados con los de los últimos tiempos, las diferencias son abismales, pero la lógica para programarlos no cambia, los conceptos de la programación son los mismos, a continuación el diagrama 3 muestra la línea de tiempo de los lenguajes de programación hasta el año de 1995. [13]

Fig. 3.
Línea de tiempo de los lenguajes de programación

Son años de evolución, y los cambios significativos en cada lapso recorrido como se detallará a continuación [13]:

A. Autocode (1952): "Sistemas de codificación simplificados" creado en la década de 1950 exclusivo para computadoras digitales en Manchester, Cambriedge y Londres.
A. Fortran (1957): un lenguaje de programación diseñado para computación numérica y informática científica. IBM lo diseño en 1957 para el desarrollo de soluciones científicas y de ingeniería.
B. Angol 68 (1968): Algorithmic Languaje, un lenguaje sucesor de Algol 60, que demostró características mas amplias de aplicación y con sintaxis mas rigurosa, este es uno de los lenguajes que primero fue definido antes de ser implementado.
C. Pascal (1970): desarrollado por Niklaus Wirth, un lenguaje que permitió definir tipos de datos y facilitando la creación de estructuras de datos dinámicas y recursivas (listas, árboles y gráficos). Su nombre se debe al matemático Blaise Pascal.
D. C (1972): Un lenguaje diseñado para programación estructurada, que incluye el terminador de código (punto y coma), ha sido base para muchos otros lenguajes e incluye llaves para el agrupamiento de bloques de código.
E. C++ (1980): creado con propósitos de programación del sistema, pero su funcionalidad se extendió para ser utilizado en el desarrollo de aplicaciones de escritorio. Su sintaxis se heredo de C, e incluye características de programación imperativas y orientadas a objetos.
F. Perl (1987): un lenguaje de alto nivel y propósitos generales, hereda características de otros lenguajes como C, AWK , poco sofisticado porque constaba de una pagina manual que sufrió varios cambios y lanzó varias versiones.
G. Phyton (1991): centrado en la legibilidad, una versión para reemplazar al lenguaje ABC, utiliza un sintaxis mas corta, omite el punto y coma y las llaves para bloques de código.
H. Java (1995): un lenguaje de programación diseñado para solucionar el tema de dependencia al momento de implementar, un lenguaje para desarrollo web y de escritorio.
I. PHP (1995): un lenguaje del lado del servidor (scripting) para desarrollo web y desarrollos de propósitos generales, logra combinarse con HTML, motores de plantillas y marcos web.

Después de 1995 han surgido nuevas tecnologías de programación, el auge en este tiempo esta en las aplicaciones móviles para sistemas operativos Andorid y los , pero que al igual utilizan conceptos básicos de programación como son los tipos de datos, herencia polimorfismo, recursividad, funciones y procedimientos en fin. Lo mimos se aplica para desarrollos de aplicaciones web y todo esto se enmarca en la utilización de framework, lenguaje híbridos o multiplataforma que han logrado minimizar el trabajo en ciertos procesos y combina arquitecturas adecuadas para obtener productos de calidad.

6. Conclusiones:

Se analiza la importancia de la programación durante la formación del ingeniero de sistemas y profesiones a fines, teniendo que en la programación se debe incluir el manejo de conceptos fundamentales que permiten o facilitan el desarrollo.

Se define la evolución de los lenguajes de programación, y que esa evolución refleja la adaptación a los diferentes tipos de necesidades en cada desarrollador.

El programador debe poseer habilidades de lógica de programación, habilidades que se deben fortalecer por medio de desarrollo de pensamiento en la resolución de problemáticas del entorno real.

El lenguaje de programación es muy importante. El lenguaje es quien entrega al programador los elementos para plasmar sus ideas y hace que el computador las entienda la tarea que se le ha asignado.

Que las técnicas de enseñanza deben ir evolucionando, porque la evolución de las tecnologías esta es constante crecimiento.

Es un trabajo inicial que busca dar respuesta a malas prácticas en programación, explorando cada unos de los factores que influyen en el proceso inicial de aprendizaje al que se enfrente los estudiantes de ingeniería de sistemas y afines.

REFERENCIAS / REFERENCES:

[1] Eslava Muñoz, V. J. (2012). Aprendiendo a programar paso a paso con C. ISBN eBook en PDF: 978-84-686-1062-7. Obtenido de http://www.bubok.es/libros/215408/Aprendiendo-a-programar-paso-a-paso-con-C.

[2] Alarcón, M., María, C., Rotger, V., Herrera, M., & Olivera, J. (2011). Aplicación Web para el Área de Ingeniería Clínica. XVIII Congreso Argentino de Bioingeniería SABI 2011, (págs. 1-10). Mar del Plata.

[3] Arias Chaves, M. (2007). La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de softwar. InterSedes, 1-13.

[4] Booch, G. (1996). Análisis y diseño orientado a objetos con aplicaciones. Pearson.

[5] Durán, F., Gutiérrez, F., & Pimentel, E. (2007). Programación orientada a objetos con Java. Madrid: Thomson.

[6] Fernández Romero, Y., & Díaz González, Y. (2012). Patrón Modelo-Vista-Controlador. Revista Telem@tica, 11(1), 47-57.

[7] Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (2002). Patrones de diseño: elementos de software orientado a objetos reutilizable. Pearson Educación.

[8] García Molina, J., Menárguez Tortosa, M., & Moros Valle, B. (2015). Una propuesta para organizar la enseñanza de la. Depto. de Informática y Sistemas Universidad de Murcia. Murcia.

[9] López Román, L. (2013). Metodologia de La Programacion Orientada a Objetos. México: Alfaomega.

[10] Merlino, H. (2006). Patrón de Diseño de Vistas Adaptables. Buenos Aires: Laboratorio de Sistemas Inteligentes. Facultad de Ingeniería. Universidad de Buenos Aires.

[11] Trueba Espinosa, A., Camarena Sagredo, J., Martínez Reyes, M., & López García, M. (2012). Automation of the Codification of the Model-View-Controller Pattern (MVC Pattern) in Projects Oriented to the Web. CIENCIA ergo-sum, 19(3), 239-250.

[12] Vélez Serrano, J., Peña Abri, A., Gortázar Bellas, F., & Sánchez Calle, Á. (2011). Diseñar y programar, todo es empezar.: Una introducción a la programación orientada a objetos usando UML y JAVA. Madrid: Dykinson.

[13] Deitel. H.M, Deitel P.J. C, How to program. Prentice Hall. 2010.