Polo del Conocimiento

Pol. Con. (Edición núm. 85) Vol. 8, No 12
Diciembre 2023, pp. 1497-1508
ISSN: 2550 - 682X
DOI: 10.23857/pc.v8i12.6358

Arquitectura de software con programación orientada a objeto

Software architecture with object-oriented programming

Arquitetura de software com programação orientada a objetos

José Belisario Vera-Vera [I]
belisariovera@espam.edu.ec
https://orcid.org/0000-0002-9101-3426

Correspondencia: belisariovera@espam.edu.ec

Ciencias de la Computación
Artículo de Investigación

*** Recibido:** 30 de octubre de 2023 ***Aceptado:** 29 de noviembre de 2023 *** Publicado:** 21 de diciembre de 2023

I. Escuela Superior Politécnica Agropecuaria de Manabí Manuel Félix López. Carrera de Ingeniería en Electrónica y Automatización Campus Politécnico Sitio El Limón, Calceta, Manabí, Ecuador.

Resumen

El objetivo de este artículo fue describir la Arquitectura de Software con Programación Orientada a Objeto a partir de la revisión de fuentes documentales actualizadas. Se ha encontrado que La Programación Orientada al Objeto (POO), agrupa un conjunto de técnicas que permiten desarrollar y mantener mucho más fácilmente programas de una gran complejidad. Es un paradigma de programación que define los programas en términos de "clases de objetos", Las características de orientación a objetos han sido agregadas a muchos lenguajes existentes como Python, Java, PHP, C++, Objective C, TypeScript, Smalltalk, entre otros, de este modo, C++ y Java son los dos lenguajes de programación orientada a objetos más usados Se puede concluir que, la evolución de la Arquitectura de Software escala cada vez más posiciones superiores, que permiten a los profesionales del dicho campo al desarrollo de software que permiten resolver diversos problemas de arquitectura con mejor calidad y en un menor tiempo para la satisfacción del usuario.

Palabras Clave: Arquitectura de sofware; Programación; Lenguaje de programación.

Abstract

The objective of this article was to describe the Software Architecture with Object-Oriented Programming based on the review of updated documentary sources. It has been found that Object Oriented Programming (OOP) brings together a set of techniques that allow highly complex programs to be developed and maintained much more easily. It is a programming paradigm that defines programs in terms of "object classes". Object-oriented features have been added to many existing languages such as Python, Java, PHP, C++, Objective C, TypeScript, Smalltalk, among others. In this way, C++ and Java are the two most used object-oriented programming languages. It can be concluded that the evolution of Software Architecture is increasingly climbing higher positions, which allow professionals in said field to develop software that They allow solving various architectural problems with better quality and in less time for user satisfaction.

Keywords: Software Architecture; Programming; Programming language.

Resumo

O objetivo deste artigo foi descrever a Arquitetura de Software com Programação Orientada a Objetos com base na revisão de fontes documentais atualizadas. Verificou-se que a Programação Orientada a Objetos (POO) reúne um conjunto de técnicas que permitem desenvolver e manter

programas altamente complexos com muito mais facilidade. É um paradigma de programação que define programas em termos de “classes de objetos”. Recursos orientados a objetos foram adicionados a muitas linguagens existentes, como Python, Java, PHP, C++, Objective C, TypeScript, Smalltalk, entre outras. desta forma, C++ e Java são as duas linguagens de programação orientada a objetos mais utilizadas.Pode-se concluir que a evolução da Arquitetura de Software está cada vez mais subindo posições mais altas, o que permite aos profissionais da referida área desenvolver software que Eles permitem resolver vários problemas de arquitetura com melhor qualidade e em menos tempo para satisfação do usuário.

Palavras-chave: Arquitetura de software; Programação; Linguagem de programação.

Introducción

La arquitectura de software da respuesta a cuáles son las soluciones necesarias para cumplir con los requisitos técnicos y comerciales de un programa informático. En contraste con la anterior, esta especialidad está centrada en los componentes, en lugar de todo el sistema, y se requiere un elevado nivel de conocimientos sobre software para aplicarla (Cristiá, 2021).

Otras definiciones hacen referencia Según el Software Engineering Institute (SEI), la arquitectura de software hace referencia a "las estructuras de un sistema, compuestas de elementos con propiedades visibles de forma externa y las relaciones que existen entre ellos”. La arquitectura de software se refiere a la estructura que ha de tener un software, las partes que vamos a construir y la forma en la que las vamos a combinar y juntar para poder trabajar con ellas (UNICESI, 2022).

La arquitectura de software, en un sentido estricto, se define como el conjunto de estructuras que componen el sistema, lo que incluye elementos de software, las relaciones entre los mismos, y las propiedades tanto de los elementos como de sus relaciones. En otras palabras, la arquitectura de software define el conjunto de componentes de un sistema, las interfaces de comunicación de los mismos, y la manera como estos componentes se comunican entre ellos usando estas interfaces (González, 2017)

La arquitectura de un sistema de software se diseña para satisfacer los requerimientos funcionales y no funcionales establecidos por los interesados en el sistema (ej.: usuarios, clientes, proveedores). Los funcionales se refieren a las tareas que se espera el sistema pueda realizar, mientras que los no-funcionales se refieren a la calidad con que se espera el sistema realice estas tareas. (Joyanes, 2014)

Actualmente producir software orientado a objetos exige del ser humano una gran capacidad de imaginación, abstracción y creatividad. Uno de los objetivos de los desarrolladores de software es resolver problemas informáticos de manera práctica. Esto conlleva la utilización en conjunto de las capacidades antes mencionadas, lo cual es una tarea difícil de realizar (Bolaños & Farinango, 2017).

La Ingeniería de Software ha permitido que junto a la disciplina de implementación se desarrolle el modelado de negocio, análisis y diseño, pruebas, entre otras disciplinas que por etapas dejan sus huellas en el proceso de desarrollo de software. En la incorporación y consolidación de estos elementos al proceso de desarrollo de software también se añaden los elementos arquitectónicos, abarcando la estructura de alto nivel de la organización del sistema, conocida como la arquitectura del software.

A partir de lo anterior el objetivo de este artículo fue describir la Arquitectura de Software con Programación Orientada a Objeto a partir de la revisión de fuentes documentatales actualizadas.

Desarrollo

Arquitectura de software

Es incuestionable que para alcanzar la calidad, excelencia y desarrollo en las sociedades modernas, la tecnología es la piedra angular en los diversos campos de desenvolvimiento del ser humano, este hecho se resalta más aún en el diseño de software, arquitectura de software o estructura de software. En atención a lo argumentado (Cervantes, Velasco, & Castro, 2016) destacan, en la actualidad, el software está presente en gran cantidad de objetos que nos rodean: desde los teléfonos y otros dispositivos que llevamos con nosotros de forma casi permanente, hasta los sistemas que controlan las operaciones de organizaciones de toda índole o los que operan las sondas robóticas que exploran otros planetas. Uno de los factores clave del éxito de los sistemas es su buen diseño; de manera particular, el diseño de lo que se conoce como arquitectura de software.

Según la norma del Institute of Electrical and Electronics Engineers (IEEE Std 1471-2000), la arquitectura de software es la organización fundamental de un sistema enmarcada en sus componentes, las relaciones entre ellos, y el ambiente, y los principios que orientan su diseño y evolución (IEEE, 2000). Al diseñar una arquitectura de software se crean y representan componentes que interactúan entre sí, con responsabilidades específicas y se organizan de forma tal que se logren los requerimientos establecidos. Se puede partir con patrones de soluciones

Pol. Con. (Edición núm. 85) Vol. 8, No 12, Diciembre 2023, pp. 1497-1508, ISSN: 2550 - 682X

probados que se conocen con el nombre de estilos arquitectónicos, patrones arquitectónicos y patrones de diseño (Rodríguez & Silva, 2016).

La arquitectura de software es considerada por (Cristiá, 2021), como el pilar de la ingeniería del software para el cambio. El diseño tiende a reducir los costos de mantenimiento porque reduce el costo de cambio, que es la principal fuente de costos, en consecuencia es una actividad esencial a la producción de software. La arquitectura de software debe ser el núcleo del diseño y desarrollo de un sistema de software, por lo cual, debe estar siempre en el primer plano y no solamente al principio del proyecto, sino durante todo el ciclo de vida, especialmente para sistemas con una larga vida útil (Estevez, 2019). Para aplicaciones complejas o grandes, la arquitectura debe ser considerada de antemano, dándole importancia a la arquitectura para lograr: Integridad conceptual; una adecuada y efectiva base para el reuso (conocimiento, experiencia, diseños y código) y un comunicación efectiva (Estevez, 2019).

Las Arquitecturas Software constituyen la piedra angular de toda fase de diseño software. Una Arquitectura Software (AS) representa la versión moderna del diseño de un sistema software. Representan el diseño de un sistema software complejo desde distintos puntos de vista, según los intereses de diversos usuarios ("stakeholders") (Capilla Sevilla, 2022). Las arquitecturas de software proveen las abstracciones críticas que permiten que las variaciones y características comunes dentro de una familia de productos sean simultáneamente manejadas (Estevez, 2019).

Diseño del software

El diseño se puntualiza como el proceso de definición de la arquitectura, componentes, interfaces y otras características de un sistema o componente y el resultado de ese proceso (Pizard, 2015). Así, el diseño como proceso, es una actividad del ciclo de vida en la cual se analizan los requisitos de software para producir una descripción interna del software que servirá como base para su construcción. Por su parte, el diseño, como resultado, describe la arquitectura del software (cómo se descompone y se organiza en componentes) y las interfaces entre esos componentes (Pizard, 2015).

La importancia del diseño del software se resume en una palabra: calidad. El diseño es el sitio en el que se introduce calidad en la ingeniería de software. Da representaciones del software que pueden evaluarse en su calidad (Pressman, 2010). Sin diseño se corre el riesgo de obtener un sistema inestable, que falle cuando se hagan cambios pequeños, o uno que sea difícil de someter a

prueba, o en el que no sea posible evaluar la calidad hasta que sea demasiado tarde en el proceso de software, cuando no queda mucho tiempo y ya se ha gastado mucho dinero (Pressman, 2010).

En el desarrollo del software, resulta necesario establecer un enfoque sistemático y disciplinado, es decir, una metodología para llevar a cabo dicho proceso, desde la óptica de la ingeniería del software, una metodología establece el orden en el que la mayoría de las actividades tienen que realizarse y los enlaces entre ellas (García & García, 2022). A partir de ello, se pueden distinguir una clasificación de las metodologías de software, entre las cuales se menciona, estructuradas (orientadas a procesos y orientadas a datos); orientadas a estados y transiciones; orientadas al diseño del conocimiento; orientadas a objetos; orientadas al desarrollo de sistemas hipermediales y basadas en métodos formales (García & García, 2022).

Arquitectura de Software con Programación Orientada a Objeto

La programación orientada a objetos (POO), tiene un enfoque hacia la ingeniería del software, comenzando por la identificación de los objetos involucrados en un problema y los mensajes que estos objetos deben responder. El programa resultante es una colección de objetos, cada uno con sus propios datos y su propio conjunto de responsabilidades. La interacción entre los objetos se realiza mediante el envío de mensajes entre sí (Galindo & Ramírez, 2015).

La Programación Orientada al Objeto (POO), agrupa un conjunto de técnicas que permiten desarrollar y mantener mucho más fácilmente programas de una gran complejidad. Es un paradigma de programación que define los programas en términos de “clases de objetos”, objetos que son entidades que combinan estado (propiedades o datos), comportamiento (procedimientos o métodos) e identidad (propiedad del objeto que lo diferencia del resto) (Rivas, 2021). La programación orientada a objetos, se trata de un estilo de programación que busca la sistematización de los mismos, modelando objetos para simular que se trata de la realidad (Joyanes, 2014). A través de esta metodología se puede facilitar el acceso de los usuarios a los programas de computación, incrementando la cobertura de internautas y promoviendo el uso masivo de estos componentes de las Tecnologías de Información y Comunicación (TIC) (Minaya, Mendoza, & Briones, 2019).

La arquitectura Orientadas a objetos, se fundamentan en la integración de los dos aspectos de los sistemas de información: datos y procesos. En este paradigma un sistema se concibe como un conjunto de objetos que se comunican entre sí mediante mensajes (García & García, 2022). El objeto encapsula datos y operaciones, es decir, este enfoque permite un modelado más natural del

mundo real y facilita enormemente la reutilización del software. Las metodologías orientadas a objetos acortan la distancia existente entre el espacio de conceptos (lo que los expertos o usuarios conocen) y el espacio de diseño e implementación (García & García, 2022).

En la POO, un objeto es una entidad virtual (o entidad de software), con datos y funciones que simulan las propiedades del objeto. Los objetos con los que se construyen los programas se ven como si fueran máquinas, las cuales están formadas por un conjunto de elementos autónomos. Las propiedades individuales de estos elementos y las relaciones entre sí definen el funcionamiento general de la máquina (Cervantes Ojeda, Gómez, Gónzalez, & García, 2016).

La Programación Orientada al Objeto (POO), es una técnica que intenta modelar el mundo real de mejor manera. Este enfoque de programación introduce conceptos como encapsulamiento y abstracción, para formar una nueva unidad llamada "clase". De esta manera mejora la reusabilidad y permite un mejor entendimiento de código (Bolaños & Farinango, 2017).

La Programación Orientada al Objeto (POO), busca resolver problemas con códigos ya funcionales, esto trae como ventaja el no crear códigos nuevos si ya existe uno que solucione el problema. Involucra en su desarrollo clases, objetos, propiedades, métodos para una mejor implementación al momento de programar (González, 2017).

Es importante resaltar que el proceso de resolver un problema mediante el uso de POO, sigue una serie de etapas (análisis, diseño e implementación) para poder construir un producto que satisfaga lo solicitado. En la etapa de análisis se determinan los requisitos funcionales, es decir, lo que se espera que el programa haga, el mundo del problema (contexto en el que se desarrolla el problema) y los requisitos no funcionales (restricciones tales como hardware, distribución geográfica y factores humanos). La fase de diseño considera las partes y diferentes relacionales que componen la solución. La fase de construcción, permite, partiendo del diseño, expresar la solución mediante algoritmos y construir el código fuente (Jaramillo & Cordona, 2018).

Las características de orientación a objetos han sido agregadas a muchos lenguajes existentes como Python, Java, PHP, C++, Objective C, TypeScript, Smalltalk, entre otros, de este modo, C++ y Java son los dos lenguajes de programación orientada a objetos más usados. La orientación a objetos se basa en la división del programa en pequeñas unidades lógicas de código; estas unidades es lo que se conoce como objetos. Éstos son unidades independientes que se comunican entre sí mediante mensajes para hacer que el programa funcione (Moncho Mas, 2001). Al dividir un programa en objetos, hace que el código sea más fácil de mantener y de reutilizar. El modelo de

programación orientada a objetos se basa en los siguientes conceptos fundamentales: clases, objetos, métodos y herencia. En un lenguaje orientado a objetos, se crean objetos utilizando un modelo conocido como clase. Una clase describe los atributos y el comportamiento que debe tener un objeto. Los comportamientos se denominan métodos.

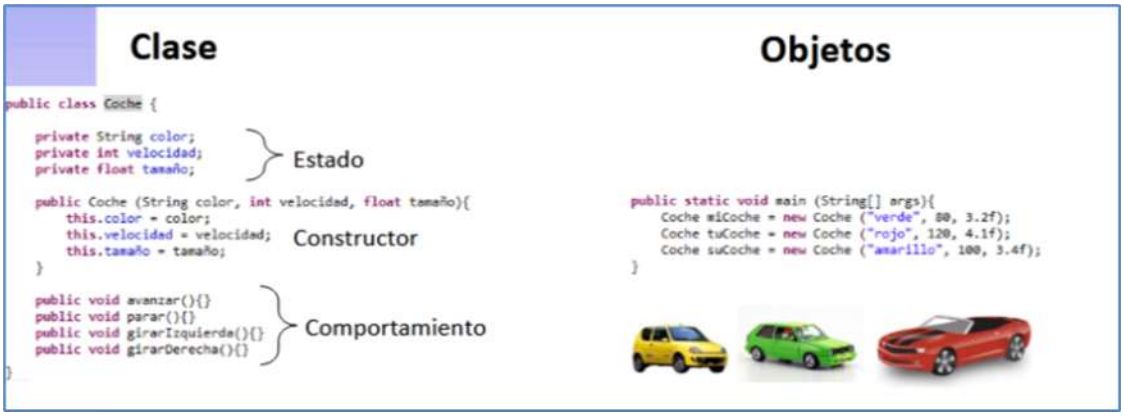
En la programación orientada a objetos, cada objeto tiene atributos y comportamientos. Cada atributo son datos o características que tiene el objeto, y cada comportamiento es algo que el objeto puede hacer. Los atributos se denominan propiedades y los comportamientos se denominan métodos. Los métodos se pueden usar para trabajar con sus propiedades del mismo objeto. En un lenguaje orientado a objetos, se crean objetos utilizando un modelo conocido como clase. Una clase describe los atributos y el comportamiento que debe tener un objeto.

El objeto, es un elemento de un programa que almacena cierta información (estado del objeto), realiza algunas acciones (comportamiento del objeto), según sus responsabilidades y posee una única identidad en el sistema. Un objeto es un ejemplar de una clase (Jaramillo & Cordona, 2018). Los objetos son representaciones (simples/complejas) (reales/imaginarias) de cosas: reloj, avión, coche. No todo puede ser considerado como un objeto, algunas cosas son simplemente características o atributos de los objetos: color, velocidad, nombre (Villena, 2019). A las propiedades o características se les da el nombre de atributos. Un atributo se declara dentro de una clase, Una clase puede contener muchos o ningún atributo. El nombre de un atributo, por convención inicia con letra minúscula. Cada atributo tiene un nombre y un tipo de dato asociado (Jaramillo & Cordona, 2018).

Una clase es una plantilla para fabricar objetos, por lo tanto, un objeto es una instancia de una clase. Las instancias de las clases (objetos) se comunican enviándose mensajes. El mundo real está lleno de objetos (concretos o abstractos), por ejemplo una persona, un carro, un triángulo. Los cuales se pueden agrupar y abstraer en una clase (Jaramillo & Cordona, 2018). La clase es un modelo o prototipo, que define las variables y métodos comunes a todos los objetos, o una plantilla genérica para un conjunto de objetos de similares características (Moncho Mas, 2001). La clase, es un Conjunto de objetos con estados y comportamientos similares. Se puede referir a la clase "automóvil" (cualquier instancia de la clasificación automóvil). La clasificación depende del problema a resolver (Villena, 2019).

Figura 1

Clases y objetos



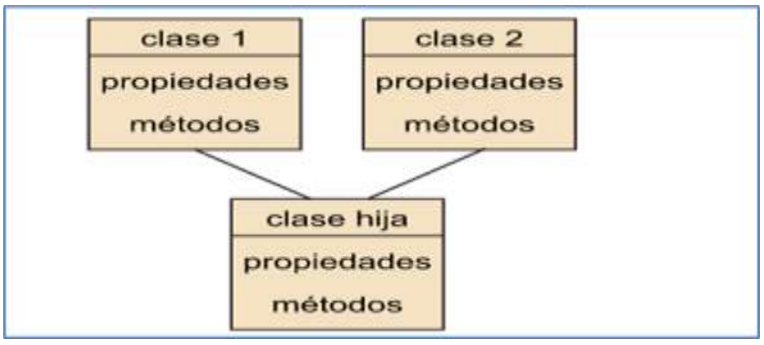
Nota. Fuente: (Villena, 2019)

La herencia, es uno de los conceptos más importantes en la programación orientada a objetos y consiste en la posibilidad de creación de clases a partir de otras existentes. Lo que hace tan potente la herencia es que la nueva clase puede heredar de la primera sus propiedades y sus métodos (aparecen así los conceptos de clase padre o superclase y clase hija o subclase) (Moncho Mas, 2001). Es el mecanismo que le permite a un objeto heredar propiedades de otra clase de objetos. Según el Servicio de Difusión de la Creación Intelectual (Sedici), la herencia permite a un objeto contener sus propios procedimientos o funciones y heredar los mismos de otros objetos (Sedici, 2013).

La herencia múltiple se produce cuando una clase puede ser derivada de más de una clase. El siguiente ejemplo muestra cómo a partir de dos clases que definen objetos de sonido e imagen se puede crear una nueva clase que herede de las dos anteriores (Moncho Mas, 2001).

Figura 2

Herencia múltiple



Nota. Fuente: (Moncho Mas, 2001).

Es común que un objeto pertenezca a la misma familia o clasificación de otro objeto, lo que conlleva a decir que los objetos que poseen el mismo tipo de datos y responden a los mensajes de la misma forma, pertenecen a la misma clase, esta clase describe un grupo de objetos en particular que tienen idénticas características (datos) y comportamientos (funcionalidad), aplicando buenas prácticas de programación, para definir el nombre de una clase (Galindo & Ramírez, 2015).

Conclusiones

Este documento muestra algunos rasgos de la arquitectura de software con programación orientada a objeto, la cual está orientada al desarrollo de software en piezas simples y reutilizables de planos de código (clases) para crear objetos diversos que puedan interactuar entre sí para y crear programas de cómputo más complejos y más funcionales que permita resolver problemas de programación en diversas entidades. Las características de orientación a objetos han sido agregadas a muchos lenguajes existentes como Python, Java, PHP, C++, Objective C, TypeScript, Smalltalk, entre otros, de este modo, C++ y Java son los dos lenguajes de programación orientada a objetos más usados.

En tal sentido, la evolución de la Arquitectura de Software escala cada vez más posiciones superiores, que permiten a los profesionales del dicho campo al desarrollo de software que permiten resolver diversos problemas de arquitectura con mejor calidad y en un menor tiempo para la satisfacción del usuario.

Referencias

Bolaños, M., & Farinango, G. (2017). Análisis de Asimilación de la Programación Orientada a Aspectos (POA), en los Estudiantes de la Materia Aplicaciones en Ambientes Propietarios de la Facultad de Sistemas en la Escuela Politécnica Nacional. Escuela Politécnica Nacional. Quito. Ecuador. Trabajo de titulación. https://bibdigital.epn.edu.ec/bitstream/15000/17322/1/CD-7817.pdf, pp.93.

Capilla Sevilla, R. (2022). Tema 1. Concepto de Arquitectura Software y Principios de Diseño. Universidad Rey Juan Carlos. España. https://burjcdigital.urjc.es › DAS-Temas, pp.241.

Cervantes Ojeda, J., Gómez, M., Gónzalez, P., & García, A. (2016). Introducción a la programación orientada a objetos. México: Universidad Autónoma Metropolitana. Primera edición. Pág.200.

Cervantes, H., Velasco, P., & Castro, L. (2016). Arquitectura de software. Conceptos y ciclo de desarrollo. México, D.F: Cengage Learning Editores, S.A. de C.V. Pág.66.

Cristiá, M. (2021). Una Teoría para el Diseño de Software. Universidad Nacional de Rosario. Argentina. Ingeniería de Software 2. https://www.fceia.unr.edu.ar/ingsoft/intro-diseno.pdf, pp.1-13.

Estevez, E. (2019). Arquitectura y diseños de sistemas. Arquitectura de Sofware. Parte I . Universidad Nacional del Sur. https://cs.uns.edu.ar/~ece/ads/downloads/Clases/2019%2004%20AyDS%20-%20Arquitectura%20de%20Software%20Parte%201.pdf, pp.62.

Galindo , O., & Ramírez, P. (2015). Enseñanza de la programación orientada a objetos usando Alice 3D y el patrón MVC. Universidad Autónoma de Bucaramanga, Colombia. https://repository.unab.edu.co/bitstream/handle/20.500.12749/3532/2015_Articulo_Galindo_Parra_Omaira_Isabel.pdf?sequence=2&isAllowed=y, pp.1-11.

García, F., & García, A. (2022). Tema 1: Introducción a la Ingeniería del Software. Universidad de Salamanca. España. https://repositorio.grial.eu/bitstream/grial/2742/1/IS_I%20Tema%201%20-%20Introduccio%CC%81n%20a%20la%20IS.pdf, pp.113.

González, A. (2017). Conceptos de programación orientada a objetos (POO). Universidad Nacional de la Plata. Argentina. http://sedici.unlp.edu.ar/bitstream/handle/10915/119243/Documento_completo.pdf-PDFA.pdf?sequence=1&isAllowed=y, pp.1-7.

IEEE. (2000). ISO/IEC/IEEE 42010: Defining architecture. Institute of Electrical and Electronics Engineers (IEEE). http://www.iso-architecture.org/ieee-1471/defining-architecture.html .

Jaramillo, S., & Cordona, S. (2018). Principios de programación orientada a objetos. Armenia, Quindío – Colombia: ELIZCOM SAS. ISBN: 978-958-8801-76-6. Pág.319.

Joyanes, L. (2014). Fundamentos de programación: algortimos y estructura de datos. Madrid. España: Google Books.

Minaya, C., Mendoza, J., & Briones, J. (2019). Los pilares de la programación orientada a la web: un enfoque teórico. Universidad, Ciencia y Tecnología; Volumen Especial Nº 01. https://uctunexpo.autanabooks.com/index.php/uct/article/view/42, pp. 4-12.

Moncho Mas, V. (2001). Introducción a la programación orientada a objetos. Madrid: McGraw Hill Interamericana.

Pizard, S. (2015). Ingeniería de Software. Universidad de la República. https://eva.fing.edu.uy/pluginfile.php/242570/mod_resource/content/2/Dise%C3%B1o%20-%20Clase%201.pdf, pp.1-20.

Pressman, R. (2010). Ingeniería del software. Un Enfoque Práctico. México, D. F: McGraw-Hill Companies, Inc. Séptima edición. Pág.805.

Rivas, M. (2021). Modulo: Programación Orientada a Objetos. Conalep - Xalapa. México. https://www.conalepveracruz.edu.mx/iniciobackup/wp-content/uploads/2021/03/Programaci%C3%B3n-orientada-a-objetos-M%C3%93DULO-PROFESIONAL.pdf, pp.68.

Rodríguez, A., & Silva, L. (2016). Arquitectura de software para el sistema de visualización médica Vismedic. RCIM; Vol.8, No.1 Ciudad de la Habana, Cuba. http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1684-18592016000100006.

Sedici. (2013). Introducción a la Programación Orientada a Objetos (POO). Servicio de Difusión de la Creación Intelectual (Sedici). Universidad Nacional de la Plata. Argentina. http://sedici.unlp.edu.ar/bitstream/handle/10915/29797/Clase_2013.pdf?sequence=3&isAllowed=y, pp.55.

Villena, J. (2019). Programación Orientada a Objetos. Universidad Carlos III de Madrid. España. https://www.it.uc3m.es/java/git-gisc/units/oo-herencia/slides/ProgramacionOrientadaAObjetos.pdf, pp.100.