

# Glosario General de Términos Arquitectura Modelo 4+1 Vistas

<<versión 1.0>>

Historial de Revisión.

|  |  |  |  |
| --- | --- | --- | --- |
| Fecha | Versión | Descripción | Autor |
| 29/11/2017 | 1.0 | Versión estable | Gabriel Huertas |

# Términos

A continuación se definen los términos resaltados en el documento “Arquitectura Modelo 4+1 Vistas Body Fitness Gym”

* ***Stakeholders:***significa ‘interesado’ o ‘parte interesada’, y que se refiere a todas aquellas personas u organizaciones afectadas por las actividades y las decisiones de una empresa [1].
* ***Developers:*** es un programador que se dedica a una o más facetas del proceso de desarrollo de software, un ámbito algo más amplio de la programación. Esta persona puede contribuir a la visión general del proyecto más a nivel de aplicación que a nivel de componentes o en las tareas de programación individuales [2].
* **Modelo de Vistas de Arquitectura 4+1:** es un modelo de vistas diseñado por el profesor Philippe Kruchten y que encaja con el estándar “IEEE 1471-2000” (Recommended Practice for Architecture Description of Software-Intensive Systems [5]) que se utiliza para describir la arquitectura de un sistema software intensivo basado en el uso de múltiples puntos de vista [3]
* **Requisitos Funcionales:** son aquellos que describen cualquier actividad que este deba realizar, en otras palabras, el comportamiento o función particular de un sistema o software cuando se cumplen ciertas condiciones [4].
* **Requisitos no funcionales:** representan características generales y restricciones de la aplicación o sistema que se esté desarrollando [5].
* **Usuario final:** es la persona que utilizará un software o un hardware luego de que este ha sido desarrollado, comercializado e instalado [6].
* **UML:** un lenguaje de modelado visual común y semántica y sintácticamente rico para la arquitectura, el diseño y la implementación de sistemas de software complejos, tanto en estructura como en comportamiento [7].
* **Diagrama de clases:** Un diagrama de clases sirve para visualizar las relaciones entre las clases que involucran el sistema, las cuales pueden ser asociativas, de herencia, de uso y de contenimiento [8].
* **Diagrama de datos:** Un modelo de base de datos muestra la estructura lógica de la base, incluidas las relaciones y limitaciones que determinan cómo se almacenan los datos y cómo se accede a ellos [9].
* **Diagrama de secuencia:** Un diagrama de secuencias muestra la interacción de un conjunto de objetos de una aplicación a través del tiempo, en el cual se indicaran los módulos o clases que formaran parte del programa y las llamadas que se hacen cada uno de ellos para realizar una tarea determinada, por esta razón permite observar la perspectiva cronológica de las interacciones [10].
* **Diagrama de comunicación:** Un diagrama de Comunicación modela las interacciones entre objetos o partes en términos de mensajes en secuencia. Los diagramas de Comunicación representan una combinación de información tomada desde el diagrama de Clases, Secuencia, y Diagrama de casos de uso describiendo tanto la estructura estática como el comportamiento dinámico de un sistema [11].
* **Diagrama de componentes:** muestra los elementos de un diseño de un sistema de software. Un diagrama de componentes permite visualizar la estructura de alto nivel del sistema y el comportamiento del servicio que estos componentes proporcionan y usan a través de interfaces [12].
* **Diagrama de paquetes:** El objetivo de estos diagramas es obtener una visión más clara del sistema de información orientado a objetos, organizándolo en subsistemas, agrupando los elementos del análisis, diseño o construcción y detallando las relaciones de dependencia entre ellos. El mecanismo de agrupación se denomina *Paquete* [13].
* **Diagrama de actividades:** se usan para describir las actividades de negocios y la funcionalidad de los sistemas de software [14].
* **Diagrama de estados:** Se especializa en mostrar transiciones entre diversos objetos [15].
* **Diagrama de despliegue:** se encuentra dentro de la familia de diagramas estructurales y describe un aspecto del sistema en sí. En este caso, el diagrama de despliegue describe el despliegue físico de información generada por el programa de software en los componentes de hardware [16].
* **Diagrama de casos de uso:** representa la forma en como un Cliente (Actor) opera con el sistema en desarrollo, además de la forma, tipo y orden en como los elementos interactúan (operaciones o casos de uso) [17].
* **Concurrencia:** es la tendencia de las cosas a producirse al mismo tiempo en un sistema. En lo que se refiere a temas de concurrencia en sistemas de software, hay dos aspectos importantes en general: poder detectar y responder a sucesos externos que se producen en un orden aleatorio y garantizar que estos sucesos se responden en un intervalo requerido mínimo [18].
* ***CRUD:*** Responde a las siglas en inglés (Create, Read, Update and Delete), Crear, Leer, Actualizar y Eliminar. Se refiere a las operaciones que se pueden realizar en un almacén de datos [19].
* **Arquitectura de Software:** representa las estructuras de un sistema, compuestas de elementos con propiedades visibles de forma externa y las relaciones que existen entre ellos [20]
* **Lógica de Negocio:** es la parte de un sistema que se encarga de codificar las reglas de negocio del mundo real que determinan cómo la información puede ser creada, mostrada y cambiada. En programación es una de las capas del modelo MVC Modelo–vista–controlador separando así la complejidad del desarrollo en capas independientes [21].
* **Patrones de diseño:** es una solución repetible a problemas típicos y recurrentes en el diseño del software. Son soluciones basadas en la experiencia y que se ha demostrado que funcionan. No son un diseño terminado que puede traducirse directamente a código, sino más bien una descripción sobre cómo resolver el problema, la cual puede ser utilizada en diversas situaciones [22].
* ***Factory Method*:** Patrón de diseño creacional que define una interfaz para crear objetos pero deja que sean las subclases las que deciden qué clases instanciar [23].
* ***GUI:*** es un conjunto de formas y métodos que posibilitan a los usuarios la interacción con un sistema, empleando para esto gráficos e imágenes. Con formas gráficas se hace referencia a botones, íconos, ventanas, fuentes, etc. los cuales representan funciones, acciones e información en el contexto de ese sistema [24].
* ***Abstract Factory:*** es un patrón de diseño para el desarrollo de software. Provee una interfaz para crear familias de objetos relacionados o dependientes entre ellos sin especificar una clase en concreto [25].
* ***Decorator:*** patrón de diseño estructural que permite añadir responsabilidades a un objeto dinámicamente. Los decoradores proporcionan una alternativa flexible a la herencia para extender la funcionalidad [26].
* ***Template:*** define el esqueleto de un algoritmo en una operación, defiriendo algunos pasos hasta las subclases. Permite que éstas redefinan ciertos pasos del algoritmo sin cambiar la estructura del algoritmo en sí [27].
* **Escalabilidad:** es la propiedad deseable en un sistema, red o proceso que indica su habilidad para poder hacerse más grande sin perder calidad en sus servicios. La escalabilidad de un sistema requiere un pensamiento cuidadoso desde el principio de su desarrollo [28].
* **Reutilización:** Es el proceso de creación de sistemas de software a partir de un software existente, en lugar de tener que rediseñarlo desde el principio [29].
* ***Facade:***patrón de diseño creacional que Busca simplificar el sistema, desde el punto de vista del cliente, proporcionando una interfaz unificada para un conjunto de subsistemas, definiendo una interfaz de nivel más alto. Esto hace que el sistema sea más fácil de usar.  
  Este patrón busca reducir al mínimo la comunicación y dependencias entre subsistemas [30]
* **Arquitectura Cliente-Servidor:** El modelo cliente-servidor (client-server), describe el proceso de interacción entre la computadora local (el cliente) y la remota (el servidor). El cliente le hace peticiones (requests, solicitudes, requerimientos) al servidor, el cual procesa dicho requerimiento y retorna los resultados al cliente apropiado [31].
* **Arquitectura MVC:** es un patrón de arquitectura de software que, utilizando 3 componentes (Vistas, Modelos y Controladores) separa la lógica de la aplicación de la lógica de la vista en una aplicación. Es una arquitectura importante puesto que se utiliza tanto en componentes gráficos básicos hasta sistemas empresariales [32].
* ***HTTP:*** son las siglas de “Hypertext Transfer Protocol” es un protocolo de transferencia donde se utiliza un sistema mediante el cual se permite la transferencia de información entre diferentes servicios y los clientes que utilizan páginas web [33].
* **Servidor Web:** Es un programa que gestiona cualquier aplicación en el lado del servidor realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente generando una respuesta en cualquier lenguaje o aplicación en el lado del cliente [34].
* ***RESTful:*** es cualquier interfaz entre sistemas que use HTTP para obtener datos o generar operaciones sobre esos datos en todos los formatos posibles, como XML y JSON [35].
* ***DBMS:*** es un conjunto de programas que se encargan de manejar la creación y todos los accesos a las bases de datos, está compuesto por: DDL: Lenguaje de Definición de Datos, DML: Lenguaje de Manipulación de Datos, SQL: Lenguaje de Consulta [36].
* ***Framework:*** es un conjunto de herramientas, la cuales permiten trabajar de forma sencilla, y que la mayoría de las veces se pueden reutilizar en otros proyectos. En el diseño y desarrollo web, un Framework sirve para poder estructurar de una mejor forma nuestro código, escribir menos líneas [37].
* ***Hibernate:*** es una herramienta de Mapeo objeto-relacional (ORM) para la plataforma Java (y disponible también para .Net con el nombre de NHibernate) que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) o anotaciones en los beans de las entidades que permiten establecer estas relaciones [38].
* ***DAO:*** es un objeto que proporciona una interfaz abstracta a algún tipo de base de datos u otro mecanismo de persistencia. DAO se puede utilizar en un gran porcentaje de las aplicaciones, que requieran almacenamiento de datos. DAO oculta todos los detalles de almacenamiento del resto de la aplicación, actuando como intermediario entre la aplicación y la base de datos [39].
* ***SQL:*** es un lenguaje de programación estándar e interactivo para la obtención de información desde una base de datos y para actualizarla [40].
* ***TCP/IP:*** es un conjunto de protocolos que permiten la comunicación entre los ordenadores pertenecientes a una red. La sigla TCP/IP significa Protocolo de control de transmisión/Protocolo de Internet [41].

**Referencias**

[1] "Significado de Stakeholder", *Significados.com*. [Online]. Available: https://www.significados.com/stakeholder/. [Accessed: 29- Nov- 2017].

[2] "Rol: Desarrollador", *Geoservice.igac.gov.co*. [Online]. Available: http://geoservice.igac.gov.co/mds/igac/ciaf/roles/desarrollador\_igac.html. [Accessed: 29- Nov- 2017].

[3] R. Moya, "Modelo “4+1” vistas de Kruchten (para Dummies)", *Jarroba*, 2017. [Online]. Available: https://jarroba.com/modelo-41-vistas-de-kruchten-para-dummies/. [Accessed: 29- Nov- 2017].

[4] "Requerimientos funcionales: Ejemplos", *Pmoinformatica.com*, 2017. [Online]. Available: http://www.pmoinformatica.com/2017/02/requerimientos-funcionales-ejemplos.html. [Accessed: 29- Nov- 2017].

[5] "Requerimientos no funcionales: Ejemplos", *Pmoinformatica.com*, 2017. [Online]. Available: http://www.pmoinformatica.com/2017/02/requerimientos-no-funcionales-ejemplos.html. [Accessed: 29- Nov- 2017].

[6] L. Alegsa, "Definición de usuario final (end user)", *Alegsa.com.ar*, 2015. [Online]. Available: http://www.alegsa.com.ar/Dic/usuario\_final.php. [Accessed: 29- Nov- 2017].

[7] "Qué es el lenguaje unificado de modelado (UML)", *Lucidchart*. [Online]. Available: https://www.lucidchart.com/pages/es/qu%C3%A9-es-el-lenguaje-unificado-de-modelado-uml. [Accessed: 29- Nov- 2017].

[8] P. Salinas, "Modelo de Clases", 2017. [Online]. Available: https://users.dcc.uchile.cl/~psalinas/uml/modelo.htm. [Accessed: 29- Nov- 2017].

[9] "Qué es un modelo de base de datos", *Lucidchart*. [Online]. Available: https://www.lucidchart.com/pages/es/qu%C3%A9-es-un-modelo-de-base-de-datos. [Accessed: 29- Nov- 2017].

[10] K. Cevallos, "UML: Diagrama de Secuencia", *INGENIERÍA DEL SOFTWARE*, 2015. [Online]. Available: https://ingsotfwarekarlacevallos.wordpress.com/2015/07/07/uml-diagrama-de-secuencia/. [Accessed: 29- Nov- 2017].

[11] "Diagrama de comunicación", *Es.wikipedia.org*. [Online]. Available: https://es.wikipedia.org/wiki/Diagrama\_de\_comunicaci%C3%B3n. [Accessed: 29- Nov- 2017].

[12] Diagramas de componentes de UML: Referencia", *Msdn.microsoft.com*. [Online]. Available: https://msdn.microsoft.com/es-es/library/dd409390.aspx. [Accessed: 29- Nov- 2017].

[13] M. Cillero, "Diagrama de Paquetes - manuel.cillero.es", *manuel.cillero.es*. [Online]. Available: https://manuel.cillero.es/doc/metrica-3/tecnicas/diagrama-de-paquetes/. [Accessed: 29- Nov- 2017].

[14] "Diagrama de actividades - UML", *Lucidchart*. [Online]. Available: https://www.lucidchart.com/pages/es/diagrama-de-actividades-uml. [Accessed: 29- Nov- 2017].

[15] "Diagrama de máquina de estados - UML", *Lucidchart*. [Online]. Available: https://www.lucidchart.com/pages/es/diagrama-de-m%C3%A1quina-de-estados-uml. [Accessed: 29- Nov- 2017].

[16] "Tutorial de diagramas de implementación", *Lucidchart*. [Online]. Available: https://www.lucidchart.com/pages/es/diagrama-de-despliegue. [Accessed: 29- Nov- 2017].

[17] P. Salinas, "Tutorial de UML - Casos de Uso", *Users.dcc.uchile.cl*. [Online]. Available: https://users.dcc.uchile.cl/~psalinas/uml/casosuso.html. [Accessed: 29- Nov- 2017].

[18] "Concepto: Concurrencia", *Cgrw01.cgr.go.cr*. [Online]. Available: http://cgrw01.cgr.go.cr/rup/RUP.es/LargeProjects/core.base\_rup/guidances/concepts/concurrency\_EE2E011A.html. [Accessed: 29- Nov- 2017].

[19] "Glosario informático - Definición de términos informáticos", *Glosarioit.com*. [Online]. Available: http://www.glosarioit.com/#!CRUD. [Accessed: 29- Nov- 2017].

[20] L. Bass, P. Clements, R. Kazman, Software Architecture in Practice, 2nd Edition, Addison Wesley, 2003

[21] "Lógica de negocio", *Es.wikipedia.org*. [Online]. Available: https://es.wikipedia.org/wiki/L%C3%B3gica\_de\_negocio. [Accessed: 29- Nov- 2017].

[22] Codecriticon. (2014). *Introducción a los Patrones de Diseño*. [online] Available at: http://codecriticon.com/introduccion-patrones-diseno/ [Accessed 29 Nov. 2017].

[23] Msdn.microsoft.com. (n.d.). *Patrones de Fabricación: Fábricas de Objetos*. [online] Available at: https://msdn.microsoft.com/es-co/library/bb972258.aspx [Accessed 29 Nov. 2017].

[24] Alegsa, L. (2016). *Definición de GUI (Interfaz Gráfica de Usuario)*. [online] Alegsa.com.ar. Available at: http://www.alegsa.com.ar/Dic/gui.php [Accessed 29 Nov. 2017].

[25] Ecured.cu. (n.d.). *Abstract Factory - EcuRed*. [online] Available at: https://www.ecured.cu/Abstract\_Factory [Accessed 29 Nov. 2017].

[26] Gala, F. (2015). *Qué es el patron de diseño Decorator*. [online] Rootear. Available at: https://rootear.com/desarrollo/patron-decorator [Accessed 29 Nov. 2017].

[27]

[28] Alegsa, L. (2016). *Definicion de Escalabilidad*. [online] Alegsa.com.ar. Available at: http://www.alegsa.com.ar/Dic/escalabilidad.php [Accessed 29 Nov. 2017].

[29] García, V. (2011). *Reutilización de Software Ventajas y Desventajas*. [online] Programación en JAVA. Available at: https://programarjava.wordpress.com/2011/12/13/reutilizacion-de-software-ventajas-y-desventajas/ [Accessed 29 Nov. 2017].

[30] Juarez, M. (2011). *Facade*. [online] Migranitodejava.blogspot.com.co. Available at: http://migranitodejava.blogspot.com.co/2011/06/facade.html [Accessed 29 Nov. 2017].

[31] Alegsa, L. (2016). *Definición de cliente/servidor (computación)*. [online] Alegsa.com.ar. Available at: http://www.alegsa.com.ar/Dic/cliente\_servidor.php [Accessed 29 Nov. 2017].

[32] Hernández, U. (n.d.). *MVC (Model, View, Controller) explicado.*. [online] CódigoFacilito. Available at: https://codigofacilito.com/articulos/mvc-model-view-controller-explicado [Accessed 29 Nov. 2017].

[33] Definista (2014). *¿Qué es Http? - Su Definición, Concepto y Significado*. [online] Conceptodefinicion.de. Available at: http://conceptodefinicion.de/http/ [Accessed 29 Nov. 2017].

[34] Ecured.cu. (n.d.). *Servidor Web - EcuRed*. [online] Available at: http://www.ecured.cu/Servidor\_Web [Accessed 29 Nov. 2017].

[35] BBVAOpen4U. (n.d.). *API REST: qué es y cuáles son sus ventajas en el desarrollo de proyectos*. [online] Available at: https://bbvaopen4u.com/es/actualidad/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos [Accessed 29 Nov. 2017].

[36] Plataforma Sistemas - Informática y Programación. (n.d.). *Ques es un DBMS*. [online] Available at: https://plataformasistemas.wordpress.com/bases-de-datos/ques-es-un-dbms/ [Accessed 29 Nov. 2017].

[37] Cacharreros de la Web. (n.d.). *¿Que es un Framework y para que sirve?*. [online] Available at: https://cacharrerosdelaweb.com/2016/08/que-es-un-framework-para-que-sirve.html [Accessed 29 Nov. 2017].

[38] Blog.educacionit.com. (2013). *¿Qué es Java Hibernate? – Educacion IT*. [online] Available at: http://blog.educacionit.com/2013/02/07/que-es-java-hibernate/ [Accessed 29 Nov. 2017].

[39] Foro Tecnico. (n.d.). *¿Qué es DAO (Objetos de Acceso a Datos)? Definición*. [online] Available at: http://www.cavsi.com/preguntasrespuestas/dao-objetos-de-acceso-a-datos/ [Accessed 29 Nov. 2017].

[40] Rouse, M. (n.d.). *¿Qué es SQL o lenguaje de consultas estructuradas? - Definición en WhatIs.com*. [online] SearchDataCenter en Español. Available at: http://searchdatacenter.techtarget.com/es/definicion/SQL-o-lenguaje-de-consultas-estructuradas [Accessed 29 Nov. 2017].

[41] CCM. (n.d.). *TCP/IP*. [online] Available at: http://es.ccm.net/contents/282-tcp-ip [Accessed 29 Nov. 2017].