



 *CARRERA ING.DE SISTEMAS*

***INTEGRANTE:***

 ***DANIEL SORIA MURILLO***

 ***DOCENTE: ING. ERVIN FLORES***

 ***MATERIA: INGENIERIA DE SOFTWARE***

 ***UNIVERSIDAD BOLIVARIANA***

 ***GESTION 2009.***

# *Metodología de desarrollo de software. El Modelo en V o de Cuatro Niveles.*

***El Método-V define un procedimiento uniforme para el desarrollo de productos para las*** [***TIC***](http://wapedia.mobi/es/TIC)***. Es el estándar utilizado para los proyectos de la Administración Federal*** [***Alemán***](http://wapedia.mobi/es/Alemania) ***y de defensa. Como está disponible públicamente muchas compañías lo usan. Es un método de*** [***gestión de proyectos***](http://wapedia.mobi/es/Gesti%C3%B3n_de_proyectos) ***comparable a*** [***PRINCE2***](http://wapedia.mobi/es/PRINCE2) ***y describe tanto métodos para la gestión como para el desarrollo de sistemas.***

***La versión actual del Método-V es el Método-V XT que se terminó en*** [***Febrero del 2005***](http://wapedia.mobi/es/Febrero_del_2005)***. No es comparable con*** [***CMMI***](http://wapedia.mobi/es/CMMI)***. Mientras que CMMI solo describe "qué" se ha hecho, el Método-V describe el "cómo" y el "cuándo" y "quién" es el responsable de haberlo hecho.***

***El Método-V fue desarrollado para regular el proceso de desarrollo de software por la Administración Federal Alemana. Describe las actividades y los resultados que se producen durante el desarrollo del software.***

***El Método-V es una representación gráfica del ciclo de vida del desarrollo del sistema. Resume los pasos principales que hay que tomar en conjunción con las correspondientes entregas de los sistemas de validación.***

***La parte izquierda de la V representa la corriente donde se definen las especificaciones del sistema. La parte derecha de la V representa la corriente donde se comprueba el sistema (contra las especificaciones definidas en la parte izquierda). La parte de abajo, donde se encuentran ambas partes, representa la corriente de desarrollo.***

***La corriente de especificación consiste principalmente de:***

* ***Especificaciones de requerimiento de usuario***
* ***Especificaciones funcionales***
* ***Especificaciones de diseño***

***La corriente de pruebas, por su parte, suele consistir de:***

* ***Calificación de instalación***
* ***Calificación operacional***
* ***Calificación de rendimiento***

***los proyectos lleguen finalmente a ser exitosos desde los puntos de vista de objetivos de negocio, costos, funcionalidad, sencillez y capacidad de soporte. La corriente de desarrollo puede consistir (depende del tipo de sistema y del alcance del desarrollo) en personalización, configuración o codificación.***

***Pero la contra al chiste clásico de la implementación de soluciones de cómputo puede lograrse* utilizando metodologías de ingeniería y arquitectura de software*, logrando que***

***En general las metodologías llevan a cabo una serie de procesos comunes que son buenas prácticas para lograr los objetivos antes mencionados independientemente de cómo hayan sido diseñadas. Las fases que agrupan estos procesos son las siguientes:***

* **Análisis**
* **Especificación**
* **Diseño**
* **Programación**
* **Prueba**
* **Documentación**
* **Mantenimiento**
* **Reingeniería**

***Así mismo las diferentes metodologías tienen diversos ciclos de vida del desarrollo de software, los modelos más comúnmente utilizados son los siguientes:***

* **Modelo en cascada**
* **Modelo en espiral**
* **Modelo de prototipos**
* **Método en V**
* **Desarrollo por etapas**

***Escribiré más adelante acerca de cada una de las metodologías mencionadas a continuación de forma más extensa posteriormente, por lo pronto dejaré abierta la investigación a los lectores acerca de los diferentes marcos de trabajo y metodologías formales de desarrollo de software. Las metodologías a tratar desde el punto de vista de la arquitectura de software y la administración de proyectos serán las siguientes:***

***Metodologías tradicionales***

* **Capability Maturity Model (SW-CMM)**
* **Capability Maturity Model Integration for Development (CMMI-DEV)**
* **Big Design Up Front (BDUF)**
* **Cleanroom Software Engineering**
* **Rational Unified Process (RUP)**
* **Essential Unified Process for Software Development (EssUP)**
* **Fusebox Lifecycle Process (FLiP)**
* **Software Process Improvement and Capability dEtermination (SPICE)**
* **Métrica**
* **Jackson System Development (JSD)**
* **Joint Application Development (JAD)**
* **Open Unified Process (OpenUP)**

***Metodologías ágiles***

* **Extreme Programming (XP)**
* **Scrum**
* **Agile Modeling Adaptive Software Development (ASD)**
* **Crystal Clear**
* **Dynamic Systems Development Method (DSDM)**
* **Feature Driven Development (FDD)**
* **Lean Software Development (LSD)**
* **Agile Unified Process (AUP)**
* **Software Development Rhythms**
* **Agile Documentation**
* **ICONIX Process**
* **Microsoft Solutions Framework (MSF)**
* **Agile Data Method**
* **Database Refactoring**
* **LeanCMMI**

**Debido a la coyuntura económica internacional*, que diría mi primo) que quiero construirme una casa.***

***¿Qué pensaríais de mí si empezara a poner ladrillos sin antes haber hecho un estudio...
...del suelo, materiales, recursos, y sin haber hecho un diseño previo? Pensaríais de mí, lo mismo que yo pienso de la gente que se pone a programar sin seguir una metodología de programación.***

***La metodología que os presento es el Modelo en V, o Modelo de Cuatro Niveles.***

***¿En qué casos la utilizo?
Se trata de un proceso ideal, por su robustez, para proyectos pequeños, con equipos de una a cinco personas.***

***También es ideal, por su claridad, para toda esa gente que nunca ha programado siguiendo una metodología. Para el proyecto final de carrera o para ese cliente que te ha conseguido un amigo de un amigo que te lo pide a ti y no se dirige a una empresa por esto de la* desaceleración*.***

***¿En que consiste exactamente?
La figura que aparece a continuación presenta el Modelo en V, o Modelo de Cuatro Niveles, del ciclo de vida de un proyecto de desarrollo de software. El modelo representa, en forma de V, las relaciones temporales entre las distintas fases del ciclo de desarrollo de un proyecto.***

******

***En los niveles lógicos del 1 al 4, para cada fase del desarrollo, existe una fase correspondiente o paralela de verificación o validación. Esta estructura obedece al principio de que para cada fase del desarrollo debe existir un resultado verificable.***

***En la misma estructura se advierte también que la proximidad entre una fase del desarrollo y su fase de verificación correspondiente va decreciendo a medida que aumenta el nivel dentro de la V. La longitud de esta separación intenta ser proporcional a la distancia en el tiempo entre una fase y su homóloga de verificación.***

* ***El nivel 1 está orientado al “cliente”. El inicio del proyecto y el fin del proyecto constituyen los dos extremos del ciclo. Se compone del análisis de requisitos y especificaciones, se traduce en un documento de requisitos y especificaciones.***
* ***El nivel 2 se dedica a las características funcionales del sistema propuesto. Puede considerarse el sistema como una caja negra, y caracterizarla únicamente con aquellas funciones que son directa o indirectamente visibles por el usuario final, se traduce en un documento de análisis funcional.***
* ***El nivel 3 define los componentes hardware y software del sistema final, a cuyo conjunto se denomina arquitectura del sistema.***
* ***El nivel 4 es la fase de implementación, en la que se desarrollan los elementos unitarios o módulos del programa.***

***¿Tengo que hacer documentación de todo?
Por supuesto. Cada fase tiene que estar respaldada por su documento correspondiente y test.***

***¿Por qué utilizar una metodología?
Porque es lo más rápido y barato. Volviendo al ejemplo de la casa, imaginad la cantidad de veces que debería volver atrás y tirar paredes ya hechas porque de pronto descubro que el suelo es inestable, la bañera no cabe, la instalación eléctrica no la había tenido en cuenta…
Pues, con el código pasa exactamente lo mismo.***