

## MODELOS DE DESARROLLO DE SOFTWARE

Imaginemos la construcción de un edificio. Lo normal es que se inicie con los cimientos y poco a poco se vayan incluyendo nuevas estructuras, desde luego ya definidas y pensadas probablemente por un ingeniero afín a esta área.



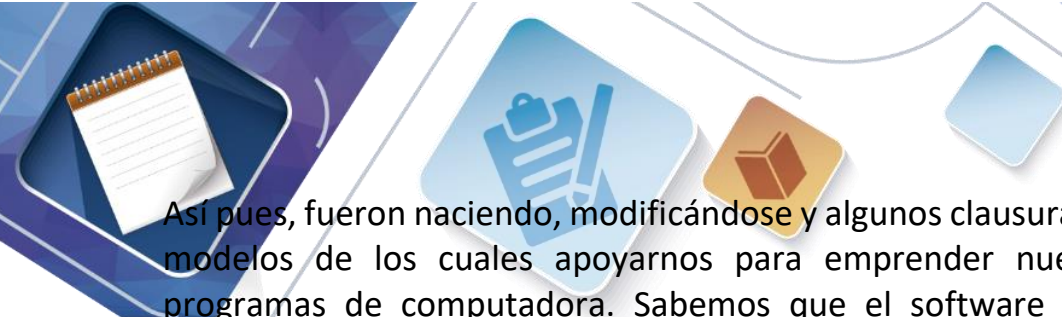
Recuperado de: <https://pixabay.com/es/azul-cubo-dise%C3%B1o-moderno-2137334/>

Autor: PIRO4D

Supongamos que inicien la obra sin planos previos y que utilicen una proporción de material determinado que soporte x cantidad de peso. Durante su elevación agregan muchos más materiales de los que puede soportar el edificio. Creo que todos podemos prever la catástrofe que esto representaría y si soporta hasta ser habitable pues aún más. Lo lógico es que se tengan planos previos y se utilicen fórmulas matemáticas para saber exactamente donde irá cada viga del edificio y cuantos ladrillos pondremos en él.

Esto mismo, guardando las proporciones, sucede con el planteamiento de un proyecto de desarrollo de software que no ha sido estructurado y no ha seguido un modelo que permita establecer cosas tan básicas pero determinantes como el tiempo de duración o los costes que este implica.

En la década de los 70's ocurrió lo que se denominó *La crisis del software*. Si ya has escuchado estas palabras probablemente sepas que significa sino, su solo nombre nos da una idea de que hubo problemas en el sector de la programación. No ahondaremos mucho en este momento histórico y nos bastará decir que esta llamada crisis, ocurrió básicamente por los planteamientos desordenados en los proyectos de desarrollo de software de la época. La mayoría de estas empresas fracasaron y surgió la necesidad de crear *La ingeniería del software*.




Así pues, fueron naciendo, modificándose y algunos clausurándose, diferentes modelos de los cuales apoyarnos para emprender nuestra creación de programas de computadora. Sabemos que el software tiene un objetivo sencillo y es satisfacer las necesidades (recreativas, de apoyo, desarrollo, etc.), de un público en particular o de uno en general. En este siglo ha ocurrido el mayor desarrollo de software jamás imaginado y existen programas que ni podemos imaginar sus alcances. Las exigencias del mercado son cada vez mayores y los consumidores de tecnología ya no se conforman con poco. Siempre hay quien pida más y no es posible permitirse errores que podrían llevar a la bancarrota a cualquier corporación, incluso a las grandes multinacionales.

Con esta pequeña introducción ahora si entramos a los principales modelos existentes para desarrollar proyectos de software. Cada uno de ellos cubre una necesidad específica. Fíjate bien en cuál es el adecuado para tu propósito e investiga más sobre el tema, eso sí, consulta las fuentes más confiables.

### **Modelo en cascada:**

Fue de los primeros modelos en aparecer, creado por W.W. Royce en 1970. El desarrollo de los proyectos de software se hace de forma lineal una fase a la vez. Esto trae ciertas implicaciones negativas. La primera de ellas es que no se tiene en cuenta las necesidades del cliente como parte primordial del proyecto, ya que todo su proceso se hace internamente olvidando el exterior. La segunda es que solo en la fase final puede verse el resultado, lo que conlleva a que muchos errores que en otros modelos pudieran ser detectados, aquí solo pueden descubrirse al final, con todo lo que ello implica (gastos adicionales, tiempo adicional, etc.).



### Modelo de desarrollo evolutivo (espiral):

Su creador es B. Bohem (1986 a 1988). Este modelo permite una mayor aproximación al usuario ya que identifica sus necesidades al exponer una primera versión del programa a su experiencia y a partir de allí de manera iterativa (repetición para lograr un objetivo), comienza a mejorar el software y a desarrollar nuevas herramientas.



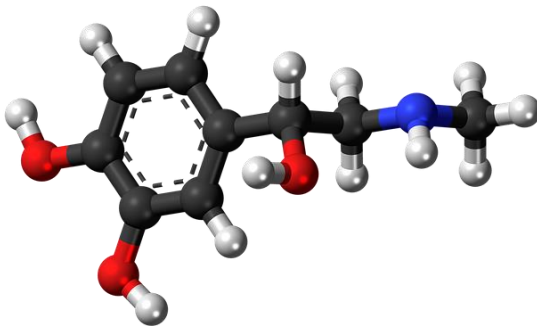
Recuperado de: <https://pixabay.com/es/escalera-espiral-arquitectura-600468/>

Autor: Stokpic

Su principal ventaja es que al estar en contacto con el cliente satisface mejor sus necesidades. Sin embargo, esto es bueno solo cuando el programa que vamos a desarrollar es pequeño o mediano ya que para proyectos de gran envergadura supone un problema a largo plazo cada modificación que se hace al código. Finalmente podría llegar a cambiarse tanto que no conserve su razón de ser inicial.

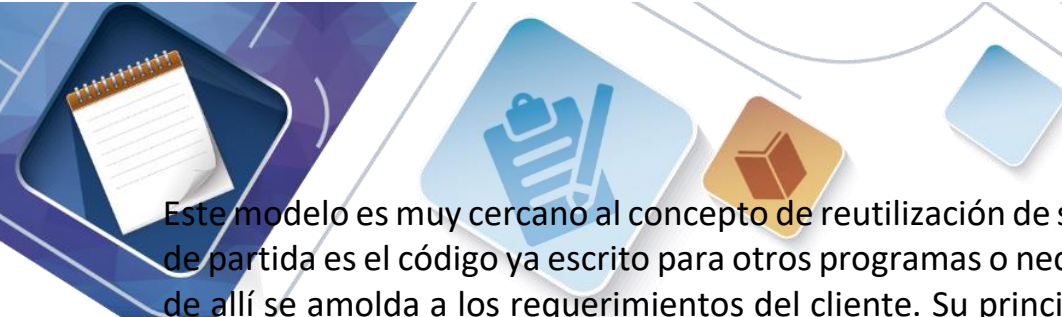
A menudo se recomienda para grandes proyectos usar una mezcla de lo mejor del modelo en cascada y lo mejor del modelo en espiral. Esto conlleva a que se elaboren proyectos que evolucionen a partir de un prototipo sin mucha especificación sin la necesidad de generar costos onerosos.

### Modelo de desarrollo basado en componentes:



Recuperado de: <https://pixabay.com/es/adrenalina-epinefrina-hormona-872345/>

Autor: Wikimedialimages



Este modelo es muy cercano al concepto de reutilización de software. Su punto de partida es el código ya escrito para otros programas o necesidades y a partir de allí se amolda a los requerimientos del cliente. Su principal ventaja es que se reducen considerablemente los costos y el tiempo estimado en un proyecto de desarrollo de software. También permite que se pueda aprender de los errores causados en otros programas. Lo realmente difícil de este modelo es que es altamente probable que no se cumplan las expectativas del cliente ya que se está amoldando a partir de un código establecido y lo que se espera es que se cree código a partir de las necesidades del cliente. Desarrollar programas a partir del cliente y no amoldar al cliente a nuestro programa.