

diseño de sistema de informacion

Taller1°



26 de agosto de 2017

yeisson estiven castro

Contenido

[¿Que es un proceso de software? 2](#_Toc491540051)

[MODELO IDEAL 3](#_Toc491540052)

[MODELO EN CASCADA O LINEAL SECUENCIAL 4](#_Toc491540053)

[Mantenimiento: 5](#_Toc491540054)

[Características: 8](#_Toc491540055)

[MODELO EVOLUTIVO 8](#_Toc491540056)

[MODELOS DE PROCESOS: ESPECIALIZADO 9](#_Toc491540057)

[MODELO EVOLUTIVO VENTAJAS Y DESVENTAJAS 12](#_Toc491540058)

[VENTAJAS 12](#_Toc491540059)

[DESVENTAJAS 12](#_Toc491540060)

[MODELO INCREMENTAL VENTAJAS Y DESVENTAJAS 13](#_Toc491540061)

[VENTAJAS: 13](#_Toc491540062)

[DESVENTAJAS: 13](#_Toc491540063)

[MODELO EN CASCADA O LINEAL SECUENCIAL VENTAJAS Y DESVENTAJAS 13](#_Toc491540064)

[BIBLIOGRAFIA 15](#_Toc491540065)

# ¿Que es un proceso de software?

La meta de la ingeniería de software es construir productos de software, o mejorar los existentes; en ingeniería de procesos, la meta es desarrollar o mejorar procesos.

Un proceso de desarrollo de software es un conjunto de personas, estructuras de organización, reglas, políticas, actividades y sus procedimientos, componentes de software, metodologías, y herramientas utilizadas o creadas específicamente para definir, desarrollar, ofrecer un servicio, innovar y extender un producto de software.

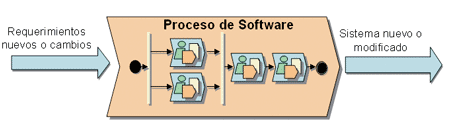
Un proceso de software efectivo habilita a la organización a incrementar su productividad al desarrollar software:

* Permite estandarizar esfuerzos, promover reusó, repetición y consistencia entre proyectos.
* Provee la oportunidad de introducir mejores prácticas de la industria.
* Permite entender que las herramientas deben ser utilizadas para soportar un proceso.
* Establece la base para una mayor consistencia y mejoras futuras.

Un proceso de software mejora los esfuerzos de mantenimiento y soporte:

* Define cómo manejar los cambios y liberaciones a sistemas de software existentes.
* Define cómo lograr la transición del software a la operación, y cómo ejecutar los esfuerzos de operación y soporte.

Necesitamos un proceso de software cuya funcionalidad esté probada en la práctica, y personalizado para que cumpla con nuestras necesidades específicas.



# MODELO IDEAL

**El modelo IDEAL**(Modelo para la Mejora Continua de Procesos) es un modelo elaborado por el SEI para guiar el inicio, planificación e implementación de iniciativas de mejora para el proceso de software en las organizaciones.  
  
El modelo IDEAL provee un enfoque disciplinado de ingeniería para la mejora del proceso de software, focaliza en el gerenciamiento del programa de mejoras y establece los fundamentos para una estrategia de largo plazo. El modelo consiste de cinco fases, compuestas por actividades:



**Iniciar:** su propósito es estableces los fundamentos básicos para garantizar la iniciativa de mejoramiento de procesos.  
se aclaran con la gerencia cuales son los objetivos de la empresa u organización que serán beneficiados por el esfuerzo.  
El apoyo de la alta dirección y de la gerencia en general es básica para el éxito del programa de mejoramiento. el compromiso de los gestionarios garantiza la disponibilidad de recursos, la infraestructura y la prioritizacion del proyecto de mejoramiento.  
Las actividades de esta fase determinan el éxito o el fracaso del programa.  
  
Diagnosticar: evaluar mediante un método formal las fortalezas y debilidades del proceso seguido por los proyectos.  
  
Establecer: realizar la planificación específica de los mejoramientos que se desea alcanzar.  
se desarrolla un plan detallado que sirve como plan de proyecto.  
se establece las estrategias y las prioridades.  
  
Actuar: implementar el mejoramiento de procesos llevando a cabo el plan de acción. Aquí se introducen o mejoran los procesos, se estrena a los respectivos niveles de personal, se miden los avances/beneficios logrados, se realiza proyectos pilotos, se implantan los procesos mejorados en los proyectos nuevos o existentes, se hacen mini evaluaciones para constatar la evolución del plan.  
  
Difundir: aprender de la experiencia del ciclo recién realizado y aumentar la habilidad de la empresa u organización para mejorar los procesos en forma continua.

### MODELO EN CASCADA O LINEAL SECUENCIAL

También llamado "Ciclo de vida básico" o "Modelo de cascada" tiene su origen en el "Modelo de cascada" ingeniado por Winston Royce, aunque omite los muchos bucles de este último. El Modelo Lineal Secuencial sugiere un enfoque sistemático o más bien secuencial del desarrollo de software que comienza en un nivel de sistemas y progresa con el análisis, diseño, codificación, pruebas y mantenimiento.

El Modelo Lineal Secuencial acompaña las siguientes actividades:

* Análisis de los requerimientos del software:

Es la fase en la cual se reúnen todos los requisitos que debe cumplir el software. En esta etapa es fundamental la presencia del cliente que documenta y repasa dichos requisitos.

* Diseño:

Es una etapa dirigida hacia la estructura de datos, la arquitectura del software, las representaciones de la interfaz y el detalle procedimental (algoritmo). En forma general se hace un esbozo de lo solicitado y se documenta haciéndose parte del software.

* Generación del código:

Es la etapa en la cual se traduce el diseño para que sea comprensible por la máquina. Esta etapa va a depender estrechamente de lo detallado del diseño

* Pruebas:

Esta etapa se centra en los procesos lógicos internos del software, asegurando que todas las sentencias se han comprobado, y en la detección de errores.

### Mantenimiento:

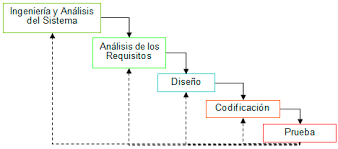
Debido a que el programa puede tener errores, puede no ser del completo agrado del cliente o puede necesitar, eventualmente acoplarse a los cambios en su entorno. Esto quiere decir que no se rehace el programa, sino que sobre la base de uno ya existente se realizan algunos cambios. El Modelo Lineal Secuencial es el paradigma de desarrollo de software más antiguo que existe, sin embargo esto no ha impedido que se haya creado una desconfianza alrededor de él basada en los siguientes errores reales:

Los proyectos raramente siguen el paradigma secuencial que propone el proyecto.

A menudo es difícil que el cliente exponga exactamente todos los requisitos.

El cliente debe tener paciencia.

Los responsables del desarrollo de software siempre se retrasan in necesariamente. Todo lo anteriormente expuesto es cierto pero este paradigma tiene un lugar bien definido e importante en el trabajo de la Ingeniería de Software aparte de proporcionar una plantilla en la que se encuentran métodos para análisis, diseño, codificación, pruebas y mantenimiento. Con todo y sus errores, sigue siendo el paradigma más utilizado en el desarrollo del software, siendo mucho mejor que un enfoque al azar.



**MODELO INCREMENTAL**

modelo incremental fue propuesto por Harlan Mills en el año 1980. Surgió el enfoque incremental de desarrollo como una forma de reducir la repetición del trabajo en el proceso de desarrollo y dar oportunidad de retrasar la toma de decisiones en los requisitos hasta adquirir experiencia con el sistema. Este modelo se conoce también bajo las siguientes denominaciones:

* Método de las comparaciones limitadas sucesivas.
* Ciencia de salir del paso.
* Método de atacar el problema por ramas.

El Modelo Incremental combina elementos del Modelo Lineal Secuencial con la filosofía interactiva de Construcción de Prototipos. Como se muestra en la Figura 1, el modelo incremental aplica secuencias lineales de forma escalonada mientras progresa el tiempo en el calendario. Cada secuencia lineal produce un incremento del software. El primer incremento generalmente es un producto esencial denominado núcleo.  
  
En una visión genérica, el proceso se divide en 4 partes:

* Análisis
* Diseño
* Código
* Prueba



Sin embargo, para la producción del Software, se usa el principio de trabajo en cadena o Pipeline. Con esto se mantiene al cliente en constante contacto con los resultados obtenidos en cada incremento. Es el mismo cliente el que incluye o desecha elementos al final de cada incremento a fin de que el software se adapte mejor a sus necesidades reales. El proceso se repite hasta que se elabora el producto completo. De esta forma el tiempo de entrega se reduce considerablemente.

El Modelo Incremental es de naturaleza interactiva brindando al final de cada incremento la entrega de un producto completamente operacional. Este modelo es particularmente útil cuando no se cuenta con una dotación de personal suficiente. Los primeros pasos los pueden realizar un grupo reducido de personas y en cada incremento se añadirá personal, de ser necesario. Por otro lado, los incrementos se pueden planear para gestionar riesgos técnicos.

Durante el proceso se trata de llevar a cabo al proyecto en diferentes partes que al final terminará siendo la solución completa requerida por el cliente, pero éstas partes no se pueden realizar en cualquier orden, sino que dependen de lo que el cliente este necesitando con más urgencia, de los puntos más importantes del proyecto, los requerimientos más básicos, difíciles y con mayor grado de riesgo, ya que estos se deben hacer al comienzo, de manera que se disminuya la dificultad y el riesgo en cada versión.

De este modo podemos terminar una aplicación ejecutable (primera versión) que podrá ser entregada al cliente para que éste pueda trabajar en ella y el programador pueda considerar las recomendaciones que el cliente efectúe para hacer mejoras en el producto. Estas nuevas mejoras deberán esperar a ser integradas en la siguiente versión junto con los demás requerimientos que no fueron tomados en cuenta en la versión anterior.

El modelo incremental consiste en un desarrollo inicial de la arquitectura completa del sistema, seguido de sucesivos incrementos funcionales. Cada incremento tiene su propio ciclo de vida y se basa en el anterior, sin cambiar su funcionalidad ni sus interfaces. Una vez entregado un incremento, no se realizan cambios sobre el mismo, sino únicamente corrección de errores. Dado que la arquitectura completa se desarrolla en la etapa inicial, es necesario conocer los requerimientos completos al comienzo del desarrollo.

l iniciar del desarrollo, los clientes o los usuarios, identifican a grandes rasgos, las funcionalidades que proporcionará el sistema. Se confecciona un bosquejo de requisitos funcionales y será el cliente quien se encarga de priorizar que funcionalidades son mas importantes. Con las funcionalidades priorizadas, se puede confeccionar un plan de incrementos, donde en cada incremento se indica un subconjunto de funcionalidades que el sistema entregará. La asignación de funcionalidades a los incrementos depende de la prioridad dada a los requisitos. Finalizado el plan de incrementos, se puede comenzar con el primer incremento.

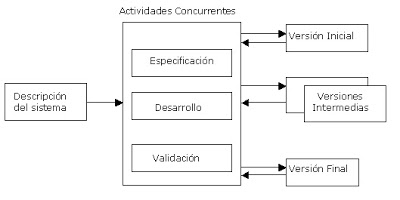
### Características:

* Se evitan proyectos largos y se entrega "algo de valor" a los usuarios con cierta frecuencia.
* El usuario se involucra más.
* Difícil de evaluar el costo total.
* Difícil de aplicar a los sistemas transaccionales que tienden a ser integrados y a operar como un todo.
* Requiere gestores experimentados.
* Los errores en los requisitos se detectan tarde.
* El resultado puede ser positivo.

### MODELO EVOLUTIVO

Los evolutivos son modelos iterativos, permiten desarrollar versiones cada vez más completas y complejas, hasta llegar al objetivo final deseado; incluso evolucionar más allá, durante la fase de operación. Los modelos “Iterativo Incremental” y “Espiral” (entre otros) son dos de los más conocidos y utilizados del tipo evolutivo.

La idea detrás de este modelo es el desarrollo de una implantación del sistema inicial, exponerla a los comentarios del usuario, refinarla en N versiones hasta que se desarrolle el sistema adecuado. Una ventaja de este modelo es que se obtiene una rápida realimentación del usuario, ya que las actividades de especificación, desarrollo y pruebas se ejecutan en cada iteración.



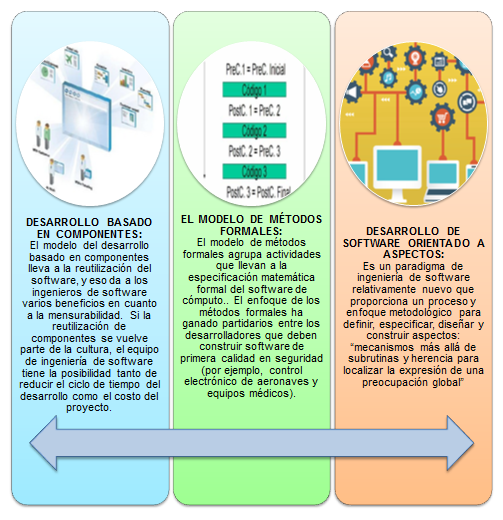
Existen dos tipos de desarrollo evolutivo:

* Desarrollo Exploratorio: El objetivo de este enfoque es explorar con el usuario los requisitos hasta llegar a un sistema final. El desarrollo comienza con las partes que se tiene más claras. El sistema evoluciona conforme se añaden nuevas características propuestas por el usuario.
* Enfoque utilizando prototipos: El objetivo es entender los requisitos del usuario y trabajar para mejorar la calidad de los requisitos. A diferencia del desarrollo exploratorio, se comienza por definir los requisitos que no están claros para el usuario y se utiliza un prototipo para experimentar con ellos. El prototipo ayuda a terminar de definir estos requisitos.

### MODELOS DE PROCESOS: ESPECIALIZADO

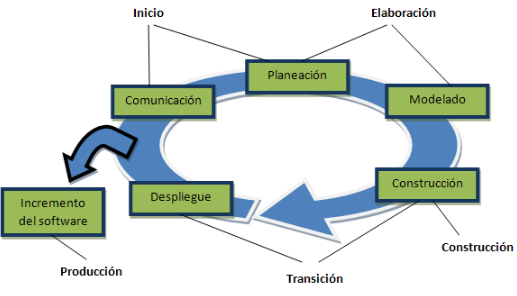
Los modelos de proceso especializado tienen muchas de las características de uno o más de los modelos tradicionales que se presentaron en las secciones anteriores. Sin embargo, dichos modelos tienden a aplicarse cuando se elige un enfoque de ingeniería de software especializado o definido muy específicamente.

Los modelos de proceso especializados son los siguientes:

[](https://ingsotfwarekarlacevallos.files.wordpress.com/2015/04/11.png)

##### **EL PROCESO UNIFICADO**

El proceso unificado es un intento por obtener los mejores rasgos y características de los modelos tradicionales del proceso del software, pero en forma que implemente muchos de los mejores principios del desarrollo ágil de software. El proceso unificado reconoce la importancia de la comunicación con el cliente y los métodos directos para describir su punto de vista respecto de un sistema. Sugiere un flujo del proceso iterativo e incremental, lo que da la sensación evolutiva que resulta esencial en el desarrollo moderno del software.

[](https://ingsotfwarekarlacevallos.files.wordpress.com/2015/04/21.png)

La fase de concepción del PU agrupa actividades tanto de comunicación con el cliente como de planeación. Al colaborar con los participantes, se identifican los requerimientos del negocio, se propone una arquitectura aproximada para el sistema y se desarrolla un plan para la naturaleza iterativa e incremental del proyecto en cuestión.

##### **PROCESO PERSONAL DEL SOFTWARE**

El proceso personal del software (PPS) pone el énfasis en la medición personal tanto del producto del trabajo que se genera como de su calidad. Además, el PPS responsabiliza al profesional acerca de la planeación del proyecto (por ejemplo, estimación y programación de actividades) y delega en el practicante el poder de controlar la calidad de todos los productos del trabajo de software que se desarrollen. El modelo del PPS define cinco actividades estructurales:



### MODELO EVOLUTIVO VENTAJAS Y DESVENTAJAS

### VENTAJAS

* La especificación puede desarrollarse de forma creciente.
* Los usuarios y desarrolladores logran un mejor entendimiento del sistema. Esto se refleja en una mejora de la calidad del software.
* Es más efectivo que el modelo de cascada, ya que cumple con las necesidades inmediatas del cliente.

### DESVENTAJAS

* Proceso no Visible: Los administradores necesitan entregas para medir el progreso. Si el sistema se necesita desarrollar rápido, no es efectivo producir documentos que reflejen cada versión del sistema.
* Sistemas pobremente estructurados: Los cambios continuos pueden ser perjudiciales para la estructura del software haciendo costoso el mantenimiento.
* Se requieren técnicas y herramientas: Para el rápido desarrollo se necesitan herramientas que pueden ser incompatibles con otras o que poca gente sabe utilizar.

# MODELO INCREMENTAL VENTAJAS Y DESVENTAJAS

### VENTAJAS:

* Con un paradigma incremental se reduce el tiempo de desarrollo inicial, ya que se implementa la funcionalidad parcial.
* También provee un impacto ventajoso frente al cliente, que es la entrega temprana de partes operativas del software.
* El modelo proporciona todas las ventajas del modelo en Cascada realimentado, reduciendo sus desventajas sólo al ámbito de cada incremento.
* Resulta más sencillo acomodar cambios al acotar el tamaño de los incrementos.

### DESVENTAJAS:

* El modelo incremental no es recomendable para casos de sistemas de tiempo real, de alto nivel de seguridad, de procesamiento distribuido y/o de alto índice de riesgos.
* Requiere de mucha planeación, tanto administrativa como técnica.
* Requiere de metas claras para conocer el estado del proyecto.

### MODELO EN CASCADA O LINEAL SECUENCIAL VENTAJAS Y DESVENTAJAS

**VENTAJAS**

* Se debe tener en cuenta que fue el primer modelo empleado, y por lo tanto es mejor que ninguno.
* Facilita la gestión del desarrollo.

**DESVENTAJAS**

* En general, establecer todos los requisitos al principio del procesode desarrolloes un mito inalcanzable, Los usuarios no pueden imaginarse lo que quieren hasta que no ven un sistema funcionando.
* Los requisitos no se pueden congelar mientras dura el desarrollo. El mercado cambia, todo cambia.
* El usuario debe esperar mucho tiempo hasta ver los resultados
* Los errores de análisis y diseño son costosos de eliminar, y se propagan a las fases siguientes con un efecto conocido como bola de nieve.
* Se genera mucho mantenimiento inicial debido al período de congelación de requisitos y éste recae, en su mayor parte.

### BIBLIOGRAFIA

<https://sg.com.mx/revista/1/procesos-software#.WZyl8yiLQdU>

<http://calidadsw2010.blogspot.com.co/2010/02/modelo-ideal.html>

<http://modelosprocesosdesoftware.blogspot.com.co/p/modelo-en-cascada-o-lineal-secuencial.html>

<https://procesosoftware.wikispaces.com/Modelo+Incremental>

<http://jorgetrejos.blogspot.com.co/2010/08/modelo-evolutivo.html>

<https://ingsotfwarekarlacevallos.wordpress.com/2015/04/29/modelos-de-procesos-especializado/>