

# RESUMEN

▼ Type	RESUMEN
👤 Author	⓵ Juan Pablo Frascino
☑ Reviewed	<input type="checkbox"/>

## UNIDAD 1: Introduccion a la ingenieria de software

### El software

Un SOFTWARE es un programa de computadora y tambien todos aquellos documentos asociados y configuraciones de datos necesarias para el funcionamiento correcto de dichos programas(datos parametricos). El software es un vehiculo para entregar un producto, la informacion.

Este puede ser desarrollado para un cliente en particular(producto personalizado) o para un mercado en general(producto generico).

Una diferencia importante entre estos tipos de software es que, en productos genéricos, la organización que desarrolla el software controla la especificación del mismo.

Para los productos personalizados, la organización que compra el software generalmente desarrolla y controla la especificación,

El software tiene diferentes ambitos de uso:

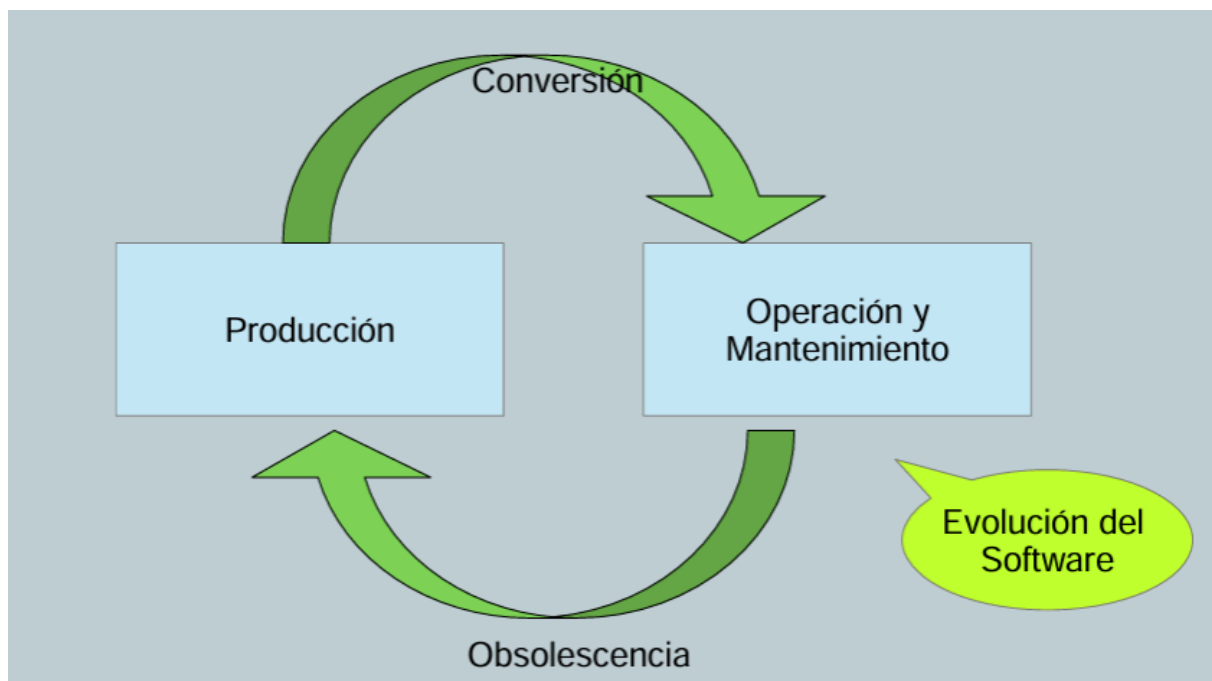
- Sistemas de tiempo real
- Sistemas empujados
- Sistemas de gestion
- Sistemas de ingenieria y cientificos
- Sistemas de inteligencia artificial

- Sistemas de PC

La calidad no tiene que ver sólo con lo que hace el software. En cambio, debe incluir el comportamiento del software mientras se ejecuta, y la estructura y organización de los programas del sistema y la documentación asociada. Esto se refleja en los llamados calidad o atributos no funcionales del software. Por lo tanto podemos decir que la calidad se puede lograr cumpliendo las PROPIEDADES EXIGIBLES de un software son:

- Cumplir con las funciones necesarias
- Facil de mantener
- Confiable
- Eficiente
- Facil de usar

## Ciclo de vida del software



## Crisis del software

En los años 50, la creación de software era de una manera mas artesanal, se hacia a prueba y error, esto no era lo mas optimo por lo que en los años 60 se comienza a notar grandes fracasos en el desarrollo de software dando lugar a una crisis del soft que deriva en la creación de la ingeniería del software en 1968.

## Ingeniería de Software

La ingeniería de software es la disciplina de la ingeniería que comprende todos los aspectos de la producción, operación y mantenimiento del software (desde las primeras etapas de la especificación del sistema hasta el mantenimiento del sistema después de que se pone en operación).

Los ingenieros aplican METODOS y HERRAMIENTAS. Adoptan un enfoque SISTEMATICO y ORGANIZADO. Incluyen ACTIVIDADES TECNICAS de desarrollo del software y ACTIVIDADES DE GESTION. Ya que esto es lo que nos permite obtener un software de mayor calidad.

La ingeniería busca obtener resultados de la calidad requerida dentro de la fecha y del presupuesto. Es por eso que se deben analizar los posibles riesgos asignándole una probabilidad de ocurrencia y un impacto.

Diferentes tipos de sistemas necesitan distintos procesos de desarrollo. Por lo que no existen métodos y técnicas universales de ingeniería de software que sean adecuados para todos los sistemas y las compañías.

STAKEHOLDER: todos los que tengan un grado de interés en el proyecto.

Algunas definiciones importantes para ing en soft:

**Metodología:** Es la ruta o el camino que seguimos para lograr los objetivos y metas establecidos en un proyecto específico. Una metodología efectiva proporciona una guía paso a paso sobre cómo abordar un proyecto, definir sus alcances, asignar recursos, gestionar el tiempo y los riesgos, y medir el éxito.

**Modelo:** Es la representación abstracta y parcial de una realidad existente o imaginada.

ejemplo: el esquema de uso de una batidora, el plano de una casa a construir, etc.

**Sistema:** Es un conjunto ordenado de componentes relacionados entre sí, ya se trate de elementos materiales o conceptuales, dotado de una estructura, una composición y un entorno particulares.

**Paradigma:** Los paradigmas son marcos o modelos conceptuales que nos ayudan a comprender y abordar ciertas áreas del conocimiento o disciplinas. Representan un conjunto de creencias, suposiciones, valores y prácticas compartidas por una comunidad científica o profesional en un campo específico.

**Ingeniería de Software:** La Ingeniería de Software es una de las ramas de las ciencias de la computación que estudia la creación de software confiable y de calidad, basándose en métodos y técnicas de ingeniería, y brindando soporte operacional y de mantenimiento.

**Proceso:** Conjunto de actividades interrelacionadas que provocan una salida diferente a la entrada, es decir, la transforman.

**Método:** Conjunto de actividades con un propósito definido, en las cuales se pueden aplicar técnicas y herramientas específicas.


**Técnica:** Forma de llevar a cabo una actividad y que puede ser aplicada en diferentes métodos, pudiendo utilizar herramientas.

**Herramienta:** instrumento automatizado o semiautomatizado para realizar una acción de la mejor manera posible.

ejemplo: una buena cuchilla para lograr un picado exacto.

**Procedimiento:** Es un conjunto de acciones que se deben llevar a cabo siempre de la misma manera, para evitar variabilidad de los resultados en el marco de las mismas condiciones. En otras palabras, el procedimiento normaliza la forma de llevar a cabo una acción determinada a través del establecimiento de un método.

Fundamental el proceso ingenieril, son las actividades básicas que todo proceso de software debe tener:

1. ANALISIS(Que quiere el cliente)
2. DISEÑO(Como diseño la solución en cuestión a 3 factores , Costo, Tiempo y Calidad )

3. CONSTRUCCION/DESARROLLO

4. PRUEBAS/TESTING

5. MANTENIMIENTO

DIFERENTES TIPOS DE SOFTWARE NECESITAN DIFERENTES PROCESOS DE DESARROLLO

Las actividades basicas son actividades genericas complejos. pueden incluir muchas sub actividades y pueden organizarse de distintas maneras. Los resultados asociados a las actividades pueden ser productos intermedios y finales

Hay dos tipos de pensamientos a la hora de enfocarse en un problema:

- LINEAL: Pienso en el problema asilado del contexto
- SISTEMATICO: Pienso en todo el contexto que envuelve el problema

Nosotros en la ingenieria de software nos debemos enfocar en el segundo

## **PRINCIPIOS DE LA INGENIERIA DE SOFTWARE:**

- 1) Agregar valor real al software
- 2) Diseñar tan simple como sea posible
- 3) Mantener una visión clara del software a construir
- 4) Desarrollar pensando que otros consumirán lo producido
- 5) Diseñar software pensando en el futuro
- 6) Planear para la reutilización
- 7) Pensar ... antes de hacer

## **DISCIPLINAS ASOCIADAS A LA ING SOFT:**

- INGENIERIA DE SOFTWARE: SOFTWARE, es la disciplina del desarrollo y mantenimiento de sistemas de soft confiables y eficientes que satisfagan los requisitos del cliente. relacionable con practicamente todas las disciplinas

- INGENIERA DE SISTEMAS: MAQUINAS/HERRAMIENTAS. tiene un enfoque interdisciplinario para realizar sistemas exitosos en el ambito de la ingenieria. requiere personas con distintos conocimientos(mecanica, electronica, civil)
- INGENIERA DE COMPUTACION: SOFTWARE, HARDWARE Y DISPOSITIVOS FISICOS, incorpora la ciencia y la tecnologia en el desarrollo y mantenimiento de componentes tanto de software como de hardware en los sistemas informaticos(especialmente empujados y distribuidos).
- SISTEMAS DE INFORMACION: PROCESOS ADMINISTRANDO INFORMACION en organizaciones. Integrando soluciones de tecnologia de la informacion y de procesos de negocios. Manejando toda la infraestructura de informacion(computadoras, comunicaciones, datos, sistemas) de una empresa.
- CIENCIAS DE LA COMPUTACION: Abarca desde fundamentos teoricos y algoritmicos hasta sistemas inteligentes, robotica y otras areas. Se aplica en promover avances en el campo, resolver desafios y problemas de maneras efectivas

# Diferencias

## Sommerville

## Pressman

El proceso de software tiene un enfoque mas estructurado para abordar el desarrollo y mantenimiento del proyecto, orientado a una evolución continua del software.	El proceso de software adapta un enfoque un poco más flexible que permita adaptar el desarrollo del software a las necesidades del usuario.
Hace énfasis en la ética	No menciona la ética
Cree que las fallas en el software son producto de las demandas crecientes y las expectativas bajas y no hace mención del software heredado	Cree que el software tiende a deteriorarse debido a la naturaleza de su entorno cambiante y menciona al software heredado como foco de atención y preocupación..
Las actividades de la ingeniería de software son: Especificación, construcción, validación y evolución.	Las actividades de la ingeniería de software son: Comunicación, Planeación, Modelado, Construcción, Despliegue.

# Semejanzas

-Ninguno de los 2 autores considera que haya un único proceso universal para desarrollar el software.

-Ambos autores coinciden en que la ingeniería de software busca desarrollar software de calidad a partir del uso de herramientas, métodos y procesos.

-Ambos autores reconocen que el software no se limita a los programas en sí, sino que también incluye documentación y datos de configuración.

-Ambos autores presentan un proceso estructurado para el desarrollo de software.  
Aunque usan metodologías diferentes.

-Ambos autores consideran las aplicaciones web como una de las grandes revoluciones del software



¿Qué es software?	Programas de cómputo y documentación asociada. Los productos de software se desarrollan para un cliente en particular o para un mercado en general.
¿Cuáles son los atributos del buen software?	El buen software debe entregar al usuario la funcionalidad y el desempeño requeridos, y debe ser sustentable, confiable y utilizable.
¿Qué es ingeniería de software?	La ingeniería de software es una disciplina de la ingeniería que se interesa por todos los aspectos de la producción de software.
¿Cuáles son las actividades fundamentales de la ingeniería de software?	Especificación, desarrollo, validación y evolución del software.
¿Cuál es la diferencia entre ingeniería de software y ciencias de la computación?	Las ciencias de la computación se enfocan en teoría y fundamentos; mientras la ingeniería de software se enfoca en el sentido práctico del desarrollo y en la distribución de software.
¿Cuál es la diferencia entre ingeniería de software e ingeniería de sistemas?	La ingeniería de sistemas se interesa por todos los aspectos del desarrollo de sistemas basados en computadoras, incluidos hardware, software e ingeniería de procesos. La ingeniería de software es parte de este proceso más general.
¿Cuáles son los principales retos que enfrenta la ingeniería de software?	Se enfrentan con una diversidad creciente, demandas por tiempos de distribución limitados y desarrollo de software confiable.
¿Cuáles son los costos de la ingeniería de software?	Aproximadamente 60% de los costos del software son de desarrollo, y 40% de prueba. Para el software elaborado específicamente, los costos de evolución superan con frecuencia los costos de desarrollo.
¿Cuáles son los mejores métodos y técnicas de la ingeniería de software?	Aun cuando todos los proyectos de software deben gestionarse y desarrollarse de manera profesional, existen diferentes técnicas que son adecuadas para distintos tipos de sistema. Por ejemplo, los juegos siempre deben diseñarse usando una serie de prototipos, mientras que los sistemas críticos de control de seguridad requieren de una especificación completa y analizable para su desarrollo. Por lo tanto, no puede decirse que un método sea mejor que otro.
¿Qué diferencias ha marcado la Web a la ingeniería de software?	La Web ha llevado a la disponibilidad de servicios de software y a la posibilidad de desarrollar sistemas basados en servicios distribuidos ampliamente. El desarrollo de sistemas basados en Web ha conducido a importantes avances en lenguajes de programación y reutilización de software.

## Etica en la ing de software

Los ingenieros de software tienen responsabilidades con la profesión de ingeniería y la sociedad. No deben preocuparse únicamente por temas técnicos. Debe comportarse de forma ética y moralmente responsable. Un profesional etico debe atender los siguientes puntos:

1. **Confidencialidad** Por lo general, debe respetar la confidencialidad de sus empleadores o clientes sin importar si se firmó o no un acuerdo formal sobre la misma.
2. **Competencia** No debe desvirtuar su nivel de competencia. Es decir, no hay que aceptar de manera intencional trabajo que esté fuera de su competencia.
3. **Derechos de propiedad intelectual** Tiene que conocer las leyes locales que rigen el uso de la propiedad intelectual, como las patentes y el copyright. Debe ser cuidadoso para garantizar que se protege la propiedad intelectual de empleadores y clientes.
4. **Mal uso de computadoras** No debe emplear sus habilidades técnicas para usar incorrectamente las computadoras de otros individuos. El mal uso de computadoras varía desde lo relativamente trivial (esto es, distraerse con los juegos de la PC del compañero) hasta lo extremadamente serio (diseminación de virus u otro malware).

## Unidad 2: Modelos de proceso de software

El proceso de software es el conjunto de actividades y resultados asociados para generar un producto de software. Diferentes software necesitan diferentes procesos, por lo que no hay un único e ideal proceso de software.

Recordemos las actividades básicas que todo proceso de software debe tener y veámoslas con más profundidad:

### ANÁLISIS/ESPECIFICACIÓN

Comprender y definir los servicios que brindará el software, las restricciones bajo las que operará y las características de calidad exigidas. Se conoce también como PROCESO DE INGENIERÍA DE REQUISITOS. Se realiza un estudio de factibilidad, elicitación y análisis de los requisitos, especificación de estos requisitos y por último se validan.

### DISEÑO y CONSTRUCCIÓN

Es transformar la especificación de requisitos en un producto ejecutable. Se divide en 2 partes, el diseño y el desarrollo. Esta parte es muy dependiente del método de desarrollo de software elegido (tradicional o ágil)

Que se diseña?

- La arquitectura
- La interfaz
- Los componentes
- La estructura de datos

Que se desarrolla?

- El código fuente y el código ejecutable/interpretable
- Se prueba el código generado
- Se debuggean los errores

## **VERIFICACION/VALIDACION/TESTING**

Se comprueba que el software este de acuerdo a las especificaciones y cumpla las expectativas de los clientes y usuarios. Las Verificaciones y las Validaciones pueden hacerse antes de tener el software funcionando.

Probar programas ejecutándolos (testing) es la principal técnica de validación.

Planificar la pruebas, ejecutar la prueba y corregir los defectos.

Tipos de Pruebas:

- Pruebas de Componentes: pruebas unitarias
- Pruebas del Sistema: pruebas de integración
- Pruebas de Aceptación: con datos reales y con usuarios

## **MANTENIMIENTO**

Es el Proceso de mantenimiento del software para satisfacer las necesidades cambiantes de los clientes. Hoy en día el proceso de desarrollo y el proceso de evolución no son dos procesos separados. El software se va modificando continuamente incluso antes de ponerse en operación. Muy poco software es un sistema completamente nuevo.

ACTIVIDAD	RESULTADO
ESPECIFICACIÓN	•Doc. de Especificación de Requisitos
DISEÑO E IMPLEMENTACIÓN	•Modelos de Diseño •Código
VERIFICACIÓN Y VALIDACIÓN	•Modelos corregidos •Software corregido
EVOLUCIÓN	•Software actualizado

En el proceso de software tambien interfieren ACTIVIDADES DE GESTION como:

- Planificar y controlar el proyecto(costos-tiempo-recursos)
- Administrar el personal
- Administrar riesgos
- Administrar calidad
- Administrar la configuracion del software
- Mejora del proceso de software

## Software heredado

Son programas antiguos, modifciados de manera continua, costosos de mantener y riesgosos de hacer evolucionar. Muchas veces es considerado de "mala calidad". La decision generalmente es no hacer nada ya que satisfacen las necesidades del usuario y corren de manera confiable.

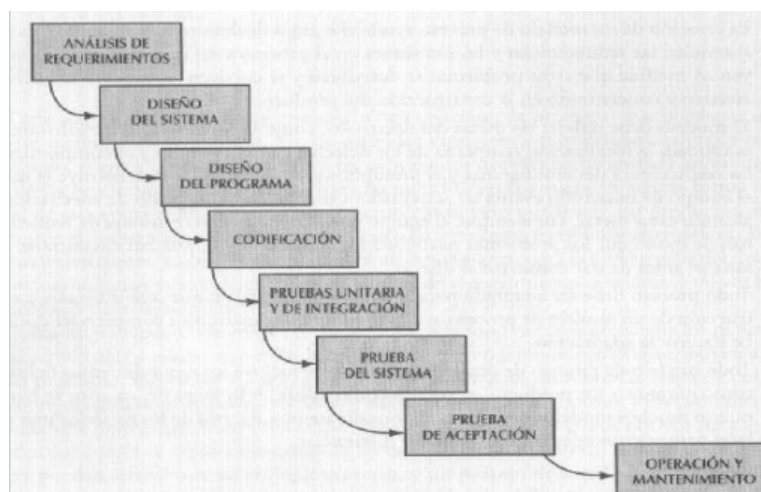
Se modifican cuando ya son necesarios o indispensables nuevos ambientes de computos y tecnologias, nuevos requisitos, necesidad de operacion con otros sistemas, cambios en redes, etc.

## Capas de la ingeniera de software



## MODELOS

### MODELO CASCADA



Fue el primero en salir. Cada fase es una fase autónoma y no se comienza hasta que se haya terminado la anterior. Es por esto mismo que es muy difícil volver atrás un proceso una vez terminado.

#### VENTAJAS:

- Buena visibilidad del proceso
- fácil de administrar
- etapas bien definidas
- fácil de entender para el cliente

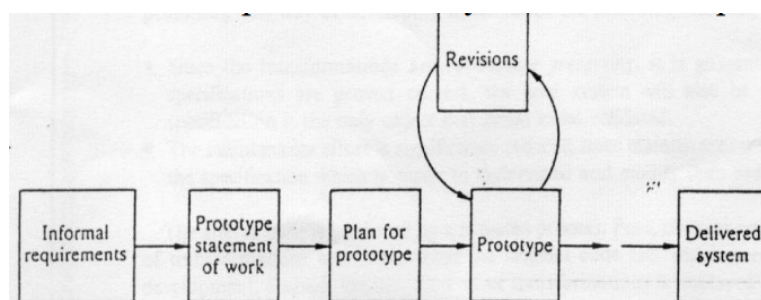
- conduce a sistemas robustos al segmentar las etapas

#### DESVENTAJAS:

- Inflexible
- perspectiva estatica de los requisitos
- mucha documentacion en cada etapa
- trata al proceso de soft como si fuera una linea de montaje sin tener en cuenta la naturaleza evolutiva del mismo
- se pueden presentar errores originales del principio en el final y es muy dificil(costoso) de volver atras

Se puede aplicar en proyectos grandes, donde el cliente tiene bien definido lo que quiere en totalidad, no se recomienda para sistemas novedosos o de alta complejidad.

## MODELO DE PROTOTIPOS



Se construye y experimenta con una imitacion/maqueta del sistema de soft, esto nos ayuda a entender y mostrar las funcionalidades requeridas al cliente. Esta maqueta no es el software final, sino una pequeña creacion para presentar y debatir sobre decisiones en el software final.

Tiene 2 enfoques:

**DESECHABLE:** Es una maqueta basica hecha solo para presentar al cliente. Con ella se genera documentacion con la info obtenida de su presentacion, y luego se desecha y se continua con el soft final.

**EVOLUTIVO:** El prototipo es presentado, y luego con la info obtenida de la presentacion se pule/refina para continuar utilizandolo e integrarlo/convertirlo en el soft final.

#### VENTAJAS:

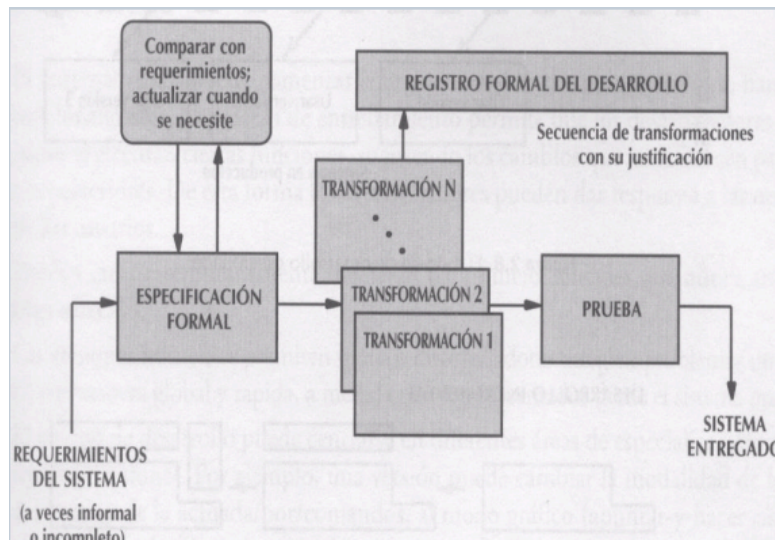
- Alta participacion de los clietnes y usuarios(lo que reduce riesgos de compresion)
- Provee un modelo del sistema de manera temprana
- En el evolutivo provee al cliente con un soft q cumple sus necesidades inmediatas
- la especificacion se puede realizar de forma creciente(bueno para cuando no estamos seguro de lo q queremos)
- En el desechable reduce el costo total del desarrollo al minimizar los cambios por defectos en los requisitos

#### DESVENTAJAS:

- poca visibilidad del avance del proceso al haber poca documentacion
- en el desechable los usuarios tienden a creer que el prototipo ya es el soft final, dando una falsa expectativa del final del desarrollo
- requiere tecnologia avanzada y desarrolladores con conocimiento en el manejo de ellas para trabajar eficientemente
- en el evolutivo tiende a una estructura deficiente debido a los cambios continuos sobre el prototipo, haciendo dificil establecer un nivel de calidad adecuado
- la especificacion es pobre ya que se construye sin una especificacion inicial.

Se puede aplicar en sistemas con especificaciones incompletas o con incertidumbres de lo q quieren, tambien en sistemas novedosos o con especificaciones de alto riesgo. O en sistemas pequeños como de corta vida o partes de sistemas mas grandes.

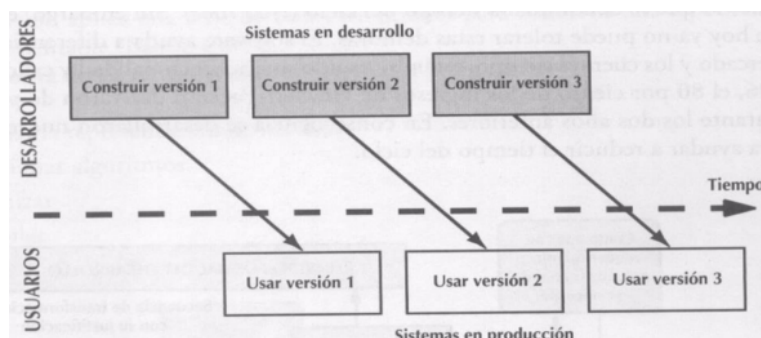
## MODELO DE TRANSFORMACION FORMAL



Utiliza un modelo formal matemático para transformar la especificación del cliente en un software entregable. NUNCA FUE IMPLEMENTADO EN LA INDUSTRIA. debido principalmente a que no era eficiente(mucho tiempo y esfuerzo en lograr la especificación) y debías tener todo un equipo especializado en esta técnica de modelado, lo cual era costoso de contratar o formar.

## MODELO DE DESARROLLO POR FASES

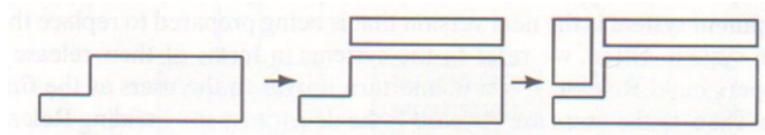
El sistema se diseña de modo que pueda ser entregado por piezas/partes. Esto permite que el usuario disponga de una cierta funcionalidad mientras se desarrolla el resto del sistema. 2 sistemas funcionan en paralelo el sistema en operación(usuarios) y el sistema en desarrollo(desarrolladores)- los sistemas se van manejando por versiones



Tenemos 2 enfoques:

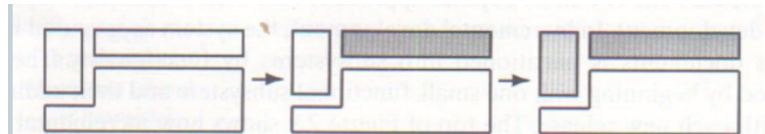
INCREMENTAL





Hago el 100% de una fase/funcionalidad a la vez hasta que el sistema se encuentra completo/terminado

#### EVOLUTIVO/ITERATIVO



Hago las funciones basicas de todas las funcionalidades, y luego en las siguientes fases voy completando un poco de cada parte hasta que el sistema se encuentra completo/terminado.

#### VENTAJAS:

- los usuarios pueden empezar a utilizar el sistema rapidamente(las funcionalidades mas importantes se entregan al principio)
- los usuarios experimentan con el sistema en cada entrega dando un mayor feedback a los desarrolladores para continuar con el sistema(bajo riesgo de fallar el sistema final)
- los requisitos pueden ir evolucionando en cada version
- se reparten mejor los equipos de desarrollo en areas y tiempos

#### DESVENTAJAS

- los incrementos deben ser relativamente pequeños ya q pueden ser dificiles de adaptar
- es dificil indentificar los elementos comunes que requieren todas las versiones al no tener una esepficiacion inicial robusta
- en las sucesivas entregas se puede ir desmejorando la arquitectura del sistema
- dificil saber la cantidad de iteraciones totales requeridas
- constantes negociaciones requeridas sobre el alcance del sistema

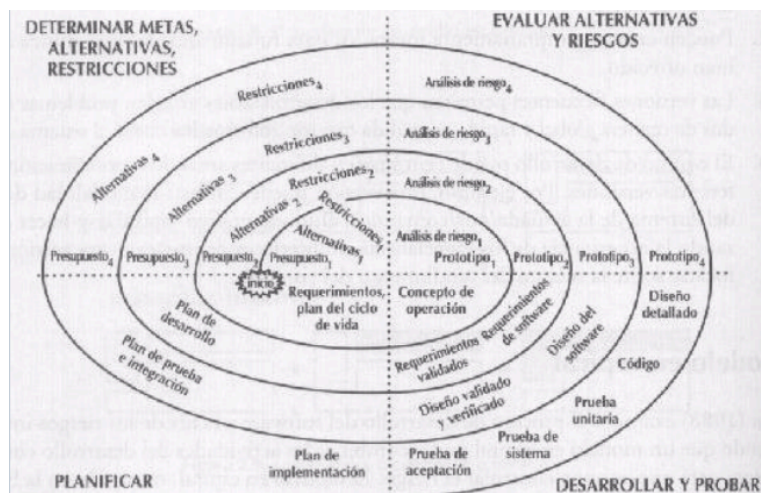
- costos de generar documentacion si los lapsos de entrega entre versiones es muy corto

Se puede aplicar cuando hay una necesidad del cliente de tener una funcion funcional(aunque limitada) lo antes posible debido a cuestiones del mercado o negocio.

El modelo incremental lo podemos utilizar cuando tenemos unos requisitos medianamente estables

El modelo iterativo cuando no se saben bien todos los requisitos y sabemos que van a ir cambiando por que el usuario no sabe definir bien lo q quiere

## MODELO EN ESPIRAL



Tiene 4 fases/ciclos, al comienzo de cada ciclo se realiza un estudio y documentacion de factibilidad, si se resuelve que ciclo de desarrollo es factible entonces se comienza/continua con el desarrollo, si se resuelve que no, se cancela y desecha. Cada fase es un submodelo de proceso, es decir puedo utilizar los demas modelos(cascada, prototipos, fases, etc) y tiene un subciclo de vida(analisis, diseño, desarrollo, pruebas, etc).

### VENTAJAS

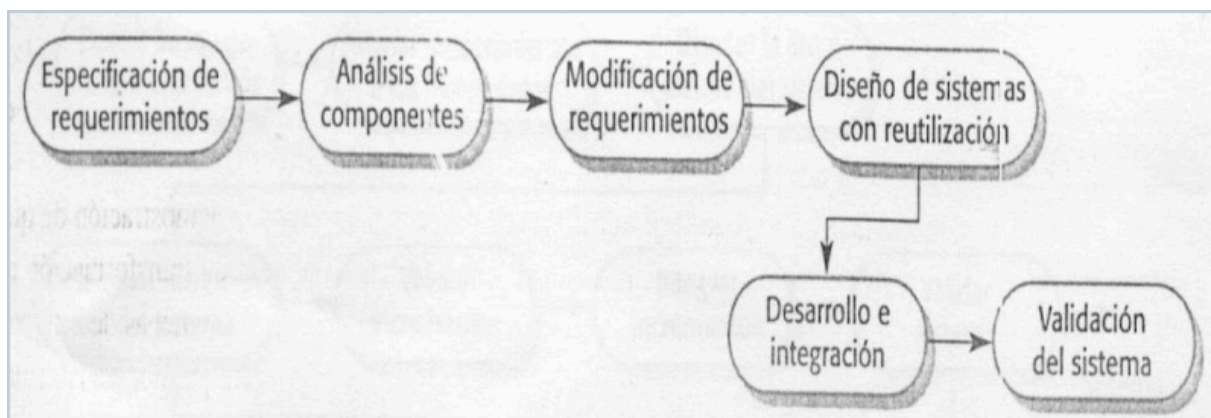
- considera el analisis de riesgos
- buena visibilidad ya que cada ciclo debe producir un documento
- reduce la posibilidad de cambios al tener participacion usuario cliente en todos los ciclos

## DESVENTAJAS

- requiere un equipo experimentado en la evaluación de riesgos
- No contempla el caso de cambios no previstos durante alguna etapa, entonces se tendrán que repetir etapas anteriores del proceso.

Se puede aplicar en sistemas muy grandes, complejos y ambiciosos o en proyectos innovadores. También cuando el cliente no quiere clavarse con un sistema muy costoso y largo de hacer, por lo que si en alguna etapa la factibilidad da que no conviene continuar, se puede ahorrar tiempo y dinero.

## MODELO BASADO EN REUTILIZACION



Se desarrolla una cantidad de componentes de software comunes que puedan ser utilizados en desarrollo de varios sistemas distintos. El reuso de componentes no está limitado exclusivamente a componentes ejecutables, sino también al reuso de componentes de diseño e incluso reuso de requisitos. Es importante contar con documentación de los componentes reutilizables.

## VENTAJAS:

- Se reducen los costos y riesgos al reducirse la cantidad de soft a desarrollar
- se tiene una entrega mas rapida del soft
- se pueden implementar mejoras a bajo costo en la funcionalidad de los componentes

- se incrementa la confiabilidad el soft ya que los compentes ya han sido validados anteriormente, por lo q la validacion se reduce a la integracion de los mismos.

#### DESVENTAJAS:

- debido al rehuso de componentes puede q no se cumpla con las necesidades reales del cliente
- se puede llegar a perder el control del mantenimiento del sistema si se reusa mucho un componente que es actualizado constantemente
- la evolucion de los requisitos no es contemplada
- cambios en los requisitos pueden requerir cambios en los componentes, lo cual muchas veces no es factible
- es dificil indetificar un potencial componente reutilizable

Se puede aplicar en sistemas que no varian mucho el uno de otro, como por ejemplos sistemas de stock o de facturacion donde los requisitos de los clientes son siempre parecidos.