Modelos de proceso o ciclo de vida

Para cada una de las fases o etapas listadas en el ítem anterior, existen sub-etapas (o tareas). El modelo de proceso o modelo de ciclo de vida utilizado para el desarrollo define el orden para las tareas o actividades involucradas también definen la coordinación entre ellas, enlace y realimentación entre las mencionadas etapas. Entre los más conocidos se puede mencionar: modelo en cascada o secuencial, modelo espiral, modelo iterativo incremental. De los antedichos hay a su vez algunas variantes o alternativas, más o menos atractivas según sea la aplicación requerida y sus requisitos.

Modelo Cascada: Este, aunque es más comúnmente conocido como modelo en cascada es también llamado modelo clásico, modelo tradicional o modelo lineal secuencial. El modelo en cascada puro difícilmente se utilice tal cual, pues esto implicaría un previo y absoluto conocimiento de los requisitos y etapas subsiguientes libres de errores; **ello sólo podría ser aplicable a escasos y pequeños desarrollos de sistemas. El paso de una etapa a otra de las mencionadas sería sin retorno, implicaría un diseño exacto y sin errores ni probable modificación o evolución.** Esto es utópico; ya que intrínsecamente el software es de carácter evolutivo, cambiante y difícilmente libre de errores, tanto durante su desarrollo como durante su vida operativa. Algún cambio durante la ejecución de una cualquiera de las etapas en este modelo secuencial implicaría reiniciar desde el principio todo el ciclo completo, lo cual redundaría en altos costos de tiempo y desarrollo. Sin embargo, el modelo cascada en algunas de sus variantes es uno de los actualmente más utilizados, por su eficacia y simplicidad, más que nada en software de pequeño y algunos de mediano porte; pero nunca (o muy rara vez) se lo usa en su forma pura, como se dijo anteriormente. En lugar de ello, siempre se produce alguna realimentación entre etapas, que no es completamente predecible ni rígida; esto da oportunidad al desarrollo de productos software en los cuales hay ciertas incertezas, cambios o evoluciones durante el ciclo de vida. Así por ejemplo, una vez capturados (elicitados) y especificados los requisitos (primera etapa) se puede pasar al diseño del sistema, pero durante esta última fase lo más probable es que se deban realizar ajustes en los requisitos (aunque sean mínimos), como fallas detectadas o propios requisitos han cambiado o evolucionado; Lo normal en el modelo cascada será entonces la aplicación del mismo con sus etapas realimentadas de alguna forma, permitiendo retroceder de una a la anterior (e incluso poder saltar a varias anteriores) si es requerido. De esta manera se obtiene un modelo cascada realimentado, Lo dicho es, a grandes rasgos, la forma y utilización de este modelo, uno de los más usados y populares.[] El modelo Cascada Realimentado resulta muy atractivo, hasta ideal, si el Proyecto presenta alta rigidez (pocos o ningún cambio, no evolutivo), los requisitos son muy claros y están correctamente especificados.

El modelo lineal o en cascada es el paradigma más antiguo y extensamente utilizado, sin embargo las críticas a él (ver desventajas) han puesto en duda su eficacia. Pese a todo, tiene un lugar muy importante en la Ingeniería de software y continúa siendo el más utilizado; y siempre es mejor que un enfoque al azar.[]

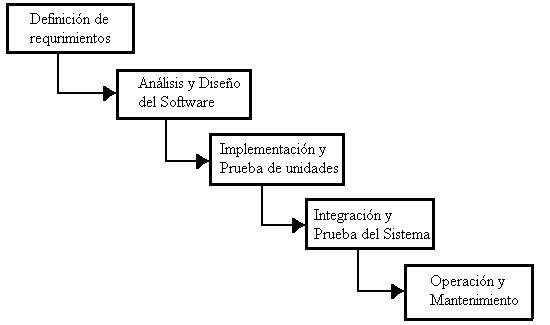
Desventajas del modelo cascada:

• Los cambios introducidos durante el desarrollo pueden confundir al equipo profesional en las etapas tempranas del proyecto. Si los cambios se producen en etapa madura (codificación o prueba) pueden ser catastróficos para un proyecto grande.

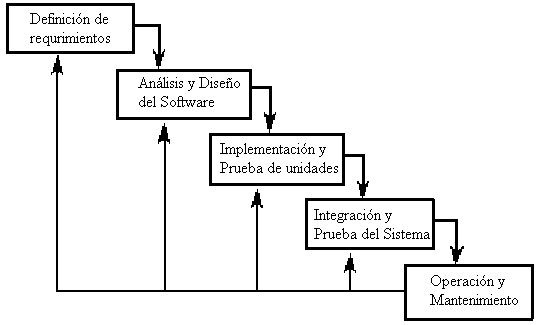
• No es frecuente que el cliente o usuario final explicite clara y completamente los requisitos (etapa de inicio); y el modelo lineal lo requiere. La incertidumbre natural en los comienzos es luego difícil de acomodar.

• El cliente debe tener paciencia ya que el software no estará disponible hasta muy avanzado el proyecto. Un error detectado por el cliente (en fase de operación) puede ser desastroso, implicando reinicio del proyecto, con altos costos.

Modelo cascada Puro o Secuencial :



Modelo cascada Realimentado:

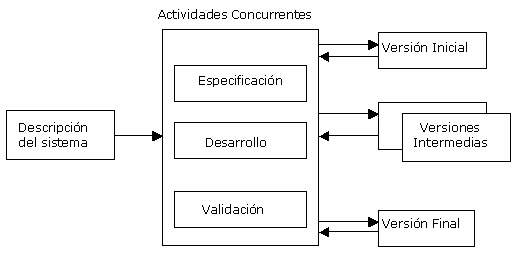


Modelos evolutivos

El software evoluciona con el tiempo. Los requisitos del usuario y del producto suelen cambiar conforme se desarrolla el mismo. Las fechas de mercado y la competencia hacen que no sea posible esperar a poner en el mercado un producto absolutamente completo, por lo que se debe introducir una versión funcional limitada (algo como una versión beta) de alguna forma para aliviar las presiones competitivas. En esas u otras situaciones similares los desarrolladores necesitan modelos de progreso que estén diseñados para acomodarse a una evolución temporal o progresiva, donde los requisitos centrales son conocidos de antemano, aunque no estén bien definidos a nivel detalle. En el modelo Cascada y Cascada Realimentado no se tiene en cuenta la naturaleza evolutiva del software, se plantea como estático con requisitos bien conocidos y definidos desde el inicio. 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 iterativos incrementales y espirales (entre otros) son dos de los más conocidos y utilizados del tipo evolutivo.

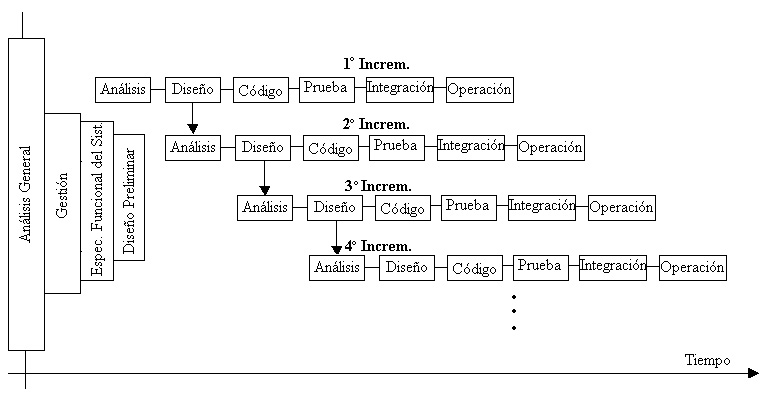
Modelo iterativo incremental:

- Diagrama genérico del desarrollo evolutivo incremental.



En términos generales, podemos distinguir, en la figura 4, los pasos generales que sigue el proceso de desarrollo de un producto software. En el modelo de ciclo de vida seleccionado, se identifican claramente dichos pasos. La Descripción del Sistema es esencial para especificar y confeccionar los distintos incrementos hasta llegar al Producto global y final. Las actividades concurrentes (Especificación, Desarrollo y Validación) sintetizan el desarrollo Pormenorizado de los incrementos, que se hará posteriormente. El diagrama nos muestra en forma muy esquemática, el funcionamiento de un ciclo iterativo incremental, el cual permite la entrega de versiones parciales a medida que se va construyendo el producto final. Es decir, a medida que cada incremento definido llega a su etapa de operación y mantenimiento. Cada versión emitida incorpora a los anteriores incrementos las funcionalidades y requisitos que fueron analizados como necesarios. El incremental es un modelo de tipo evolutivo que está basado en varios ciclos Cascada realimentados aplicados repetidamente, con una filosofía iterativa. En la figura se muestra un refino del diagrama previo, bajo un esquema temporal, para obtener finalmente el esquema del Modelo de ciclo de vida Iterativo Incremental, con sus actividades genéricas asociadas.

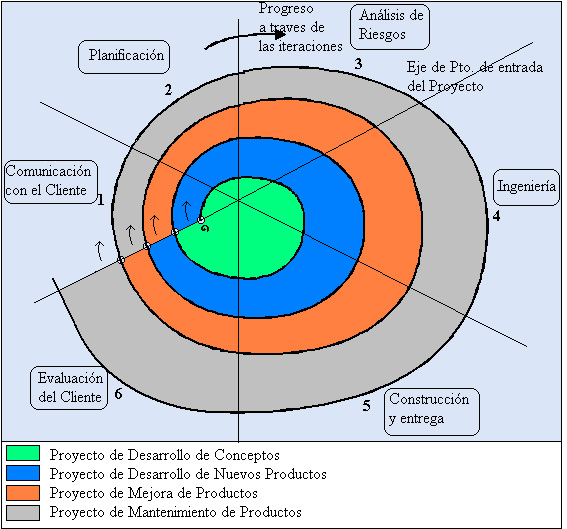
-Modelo iterativo incremental para el ciclo de vida del software.



En la figura 5 se muestra un refino del diagrama previo, bajo un esquema temporal, para obtener finalmente el esquema del Modelo de ciclo de vida Iterativo Incremental, con sus actividades genéricas asociadas. Aquí se observa claramente cada ciclo cascada que es aplicado para la obtención de un incremento; estos últimos se van integrando para obtener el producto final completo. Cada incremento es un ciclo Cascada Realimentado, aunque, por simplicidad, en la figura 5 se muestra como secuencial puro. La selección de este modelo permite realizar entregas funcionales tempranas al cliente (lo cual es beneficioso tanto para él como para el grupo de desarrollo). Se priorizan las entregas de aquellos módulos o incrementos en que surja la necesidad operativa de hacerlo, por ejemplo para cargas previas de información, indispensable para los incrementos siguientes. El modelo iterativo incremental no obliga a especificar con precisión y detalle absolutamente todo lo que el sistema debe hacer, (y cómo), antes de ser construido (como el caso del cascada, con requisitos congelados). Sólo se hace en el incremento en desarrollo. Esto torna más manejable el proceso y reduce el impacto en los costos. Esto es así, porque en caso de alterar o rehacer los requisitos, solo afecta una parte del sistema. Aunque, lógicamente, esta situación se agrava si se presenta en estado avanzado, es decir en los últimos incrementos. En definitiva, el modelo facilita la incorporación de nuevos requisitos durante el desarrollo.

Modelo espiral

El modelo espiral fue propuesto inicialmente por Barry Boehm. Es un modelo evolutivo que conjuga la naturaleza iterativa del modelo MCP con los aspectos controlados y sistemáticos del Modelo Cascada. Proporciona potencial para desarrollo rápido de versiones incrementales. En el modelo Espiral el software se construye en una serie de versiones incrementales. En las primeras iteraciones la versión incremental podría ser un modelo en papel o bien un prototipo. En las últimas iteraciones se producen versiones cada vez más completas del sistema diseñado. El modelo se divide en un número de Actividades de marco de trabajo, llamadas regiones de tareas. En general existen entre tres y seis regiones de tareas (hay variantes del modelo). En la figura se muestra el esquema de un Modelo Espiral con 6 regiones.



Región 1 - Tareas requeridas para establecer la comunicación entre el cliente y el desarrollador.

• Región 2 - Tareas inherentes a la definición de los recursos, tiempo y otra información relacionada con el proyecto.

• Región 3 - Tareas necesarias para evaluar los riesgos técnicos y de gestión del proyecto.

• Región 4 - Tareas para construir una o más representaciones de la aplicación software.

• Región 5 - Tareas para construir la aplicación, instalarla, probarla y proporcionar soporte al usuario o cliente (Ej. documentación y práctica).

•Región 6 - Tareas para obtener la reacción del cliente, según la evaluación de lo creado e instalado en los ciclos anteriores.

Las actividades enunciadas para el marco de trabajo son generales y se aplican a cualquier proyecto, grande, mediano o pequeño, complejo o no. Las regiones que definen esas actividades comprenden un conjunto de tareas del trabajo: ese conjunto sí se debe adaptar a las características del proyecto en particular a emprender.

Al inicio del ciclo, o proceso evolutivo, el equipo de ingeniería gira alrededor del espiral (metafóricamente hablando) comenzando por el centro (marcado con ๑ en la imagen ) y en el sentido indicado; el primer circuito de la espiral puede producir el desarrollo de una especificación del producto; los pasos siguientes podrían generar un prototipo y progresivamente versiones más sofisticadas del software. Cada paso por la región de planificación provoca ajustes en el plan del proyecto; el coste y planificación se realimentan en función de la evaluación del cliente.

Cada uno

de los circulitos (๏) fijados a lo largo del eje representan puntos de arranque de los distintos proyectos

(relacionados); a saber:

• Un proyecto de «desarrollo de conceptos» comienza al inicio de la espiral, hace múltiples iteraciones hasta que se

completa, es la zona marcada con verde.

Software 10

• Si lo anterior se va a desarrollar como producto real, se inicia otro proyecto: «Desarrollo de nuevo Producto». Que

evolucionará con iteraciones hasta culminar; es la zona marcada en color azul.

• Eventual y análogamente se generarán proyectos de «mejoras de productos» y de «mantenimiento de productos»,

con las iteraciones necesarias en cada área (zonas roja y gris, respectivamente).

Cuando la espiral se caracteriza de esta forma, está operativa hasta que el software se retira, eventualmente puede estar inactiva (el proceso), pero cuando se produce un cambio el proceso arranca nuevamente en el punto de entrada apropiado (por ejemplo, en «mejora del producto»).

El modelo espiral da un enfoque realista, que evoluciona igual que el software; se adapta muy bien para desarrollos a gran escala.

El Espiral utiliza el MCP para reducir riesgos y permite aplicarlo en cualquier etapa de la evolución. Mantiene el enfoque clásico (cascada) pero incorpora un marco de trabajo iterativo que refleja mejor la realidad. Este modelo requiere considerar riesgos técnicos en todas las etapas del proyecto; aplicado adecuadamente debe reducirlos antes de que sean un verdadero problema. El Modelo evolutivo como el Espiral es particularmente apto para el desarrollo de Sistemas Operativos (complejos); también en sistemas de altos riesgos o críticos (Ej. navegadores y controladores aeronáuticos) y en todos aquellos en que sea necesaria una fuerte gestión del proyecto y sus riesgos, técnicos o de gestión.

Desventajas importantes:

• Requiere mucha experiencia y habilidad para la evaluación de los riesgos, lo cual es requisito para el éxito del proyecto.

• Es difícil convencer a los grandes clientes que se podrá controlar este enfoque evolutivo.

Este modelo no se ha usado tanto, como el Cascada (Incremental) o MCP, por lo que no se tiene bien medida su eficacia, es un paradigma relativamente nuevo y difícil de implementar y controlar.