Modelos de Proceso de Desarrollo de Software

Modelos de Proceso de Desarrollo de Software

Contenidos

- Conceptos
- Ciclo de vida del desarrollo Software
- Modelos para el desarrollo de software

Introducción

Conceptos

Proceso:

- Define el marco de trabajo y permite un desarrollo racional y oportuno de la Ingeniería del Software

Método:

- Indica cómo construir técnicamente el software. Se incluyen técnicas de modelado y otras técnicas descriptivas

Conceptos

Herramientas:

- Proporcionan el soporte automático o semiautomático para el proceso y para los métodos

Notación:

- Conjunto de reglas gráficas o textuales para la representación de un modelo

Metodología:

- Colección de métodos para resolver un tipo de problemas
- Descomposición del proceso de desarrollo en actividades y los métodos adecuados para llevar a cabo dichas actividades

Conceptos

Ciclo de vida:

- El ciclo de vida del desarrollo Software (SDLC), es una secuencia estructurada y bien definida de las etapas en Ingeniería de software para desarrollar el producto software deseado.

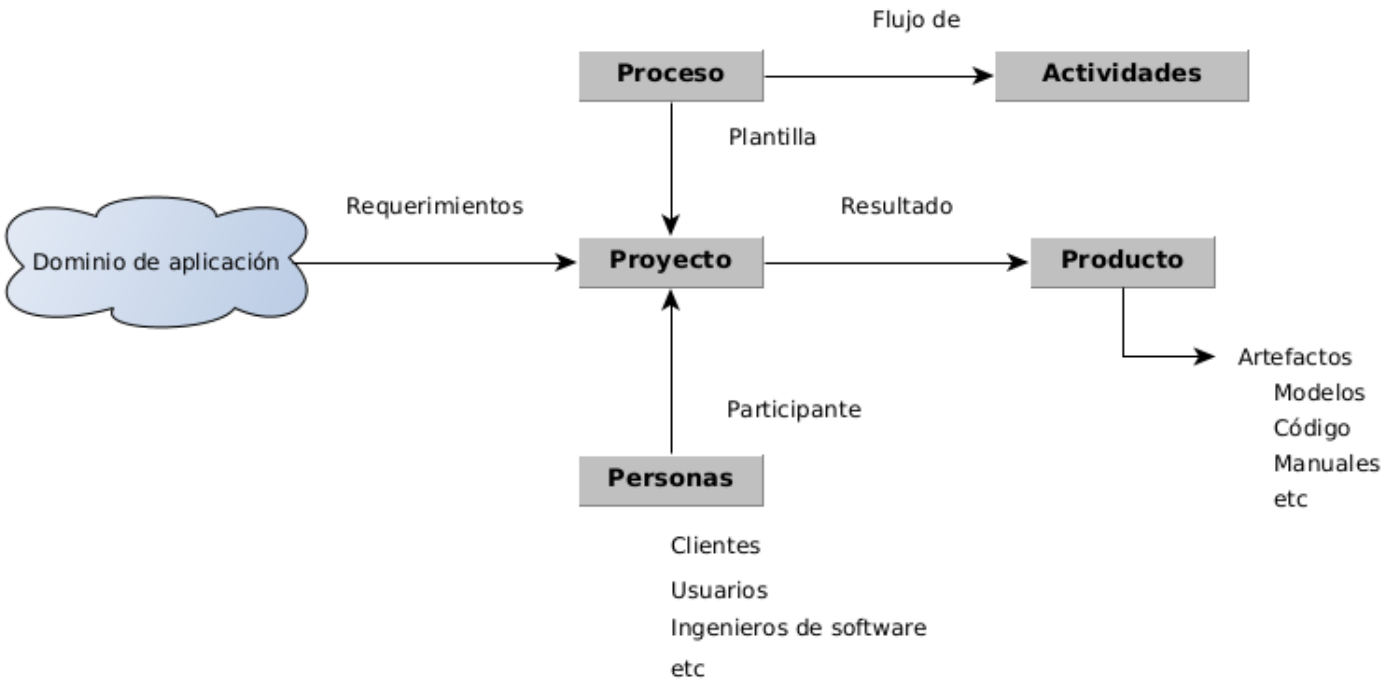
Modelo de desarrollo de software:

- Es una representación simplificada del proceso para el desarrollo de software, presentada desde una perspectiva específica.

Metodología de desarrollo de software:

- Es un enfoque estructurado para el desarrollo de software que incluye modelos de sistemas, notaciones, reglas, sugerencias de diseño y guías de procesos.

Conceptos


Flujo de
Proceso
Actividades
Plantilla
Requerimientos
Resultado
Dominio de aplicación
Proyecto
Producto
Artefactos
Modelos
Código
Manuales
etc
Participante
Personas
Clientes
Usuarios
Ingenieros de software
etc

Proceso de Software

Proceso:

Conjunto ordenado de actividades; una serie de pasos que involucran tareas, restricciones y recursos que producen una determinada salida esperada (Pfleeger, 2002)

Marco de trabajo de las tareas que se requieren para construir software de alta calidad (Pressman, 2010)

Proceso de Software

Un proceso de software debe especificar:

- La ***secuencia de actividades*** a realizar por el equipo de desarrollo (Flujo de actividades)
- Los ***productos*** que deben crearse
  - *resultados del trabajo (modelos, documentos, datos informes...)*
  - *qué y cuándo*
- La ***asignación de tareas*** a cada miembro del equipo y al equipo como un todo
- Los ***criterios*** para controlar el proceso
  - *se establece el control de gestión de los proyectos software*
  - *establecimiento de hitos*
- Las posibles ***heurísticas***

Proceso de Software

Importancia:

- Facilita la gestión del proyecto
- Establece una división del trabajo
- Facilita la comunicación de los miembros del equipo
- Permite la reasignación y la reutilización de personal especializado (transferencia entre proyectos)
- Mejora la productividad y el desarrollo (desarrollo reproducible)
- Establece el contexto en el que se aplican los métodos técnicos
- Gestiona el cambio adecuadamente
- Asegura la calidad

Ciclo de vida del desarrollo Software

Ciclo de vida:

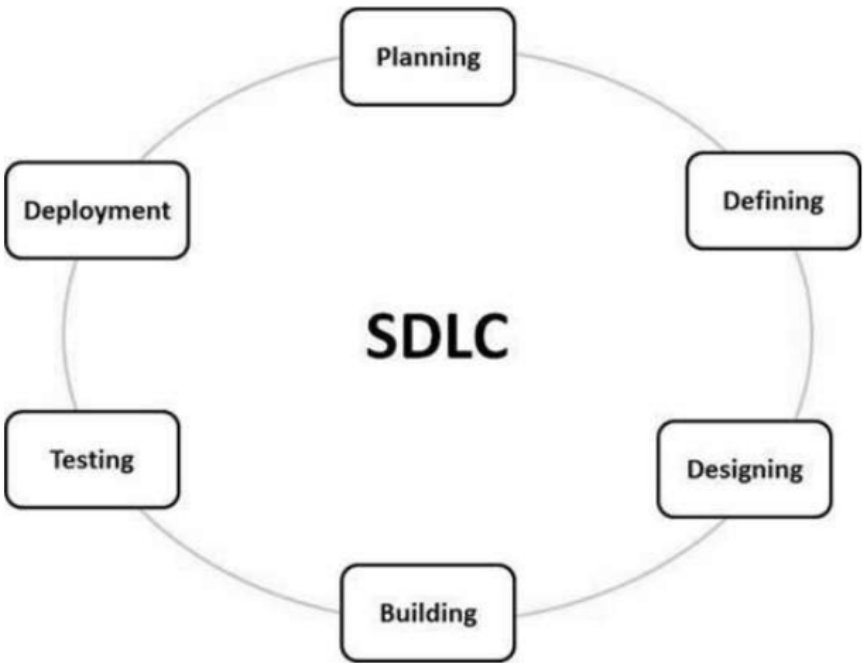
> *Las distintas fases por las que pasa el software desde que nace una necesidad de mecanizar un proceso hasta que deja de utilizarse el software que sirvió para ese objetivo, pasando por las fases de desarrollo y explotación (Frakes et al., 1991)*

Ciclo de desarrollo:

> *El período de tiempo que comienza con la decisión de desarrollar un producto software y finaliza cuando se ha entregado este. Este ciclo incluye, en general, una fase de requisitos, una fase de diseño, una fase de implantación, una fase de pruebas, y a veces, una fase de instalación y aceptación (AEC, 1986)*

Ciclo de vida del desarrollo Software

El ciclo de vida de desarrollo de software (SDLC) aporta una serie de pasos a seguir con la finalidad de diseñar y desarrollar un producto software de manera eficiente



Modelo de proceso

Un modelo de proceso software es una representación abstracta de un proceso software (Sommerville, 2005)

- Hay varios modelos de procesos definidos en la bibliografía de Ingeniería del Software
- Cada modelo de proceso representa un proceso desde una perspectiva particular, por lo que sólo ofrece una información parcial sobre dicho proceso
- Los modelos de proceso genéricos, también llamados paradigmas de proceso, presentan un proceso desde una perspectiva arquitectónica, es decir, ofrecen un marco de definición para el proceso, pero no detallan las actividades específicas
- Dichos modelos no son descripciones definitivas de los procesos software, más bien son abstracciones útiles que se utilizan para explicar diferentes aproximaciones al desarrollo del software

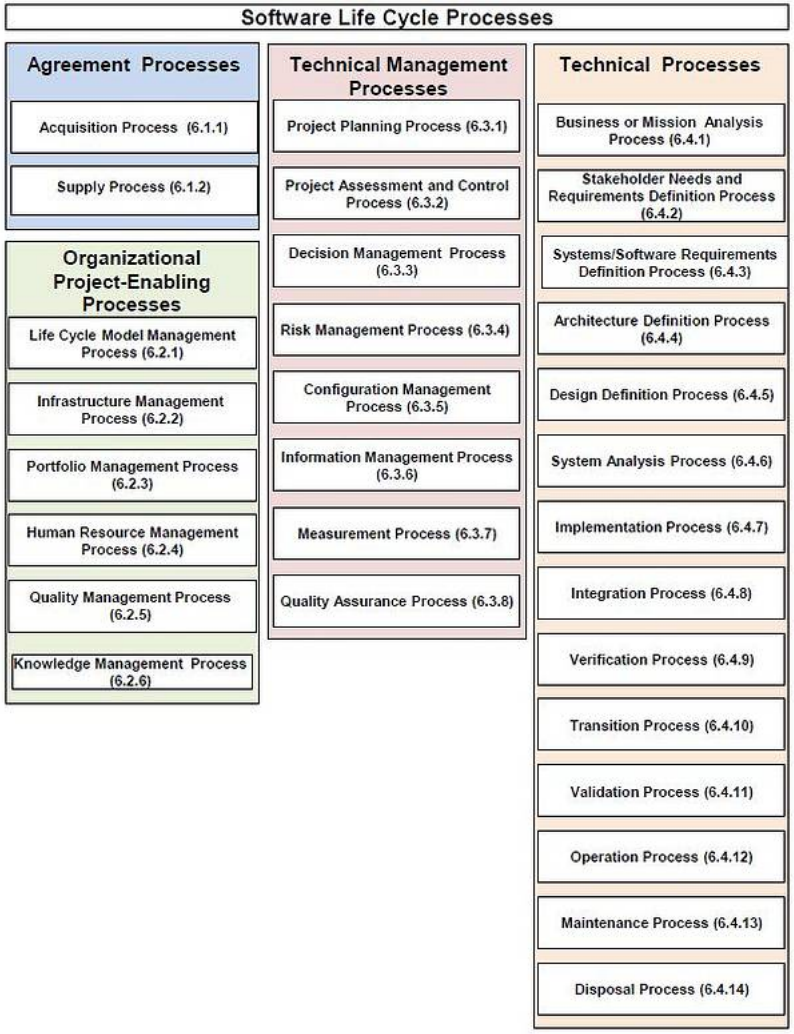
Modelo de proceso

- ***Especificación:*** Formulación de los requisitos y restricciones del sistema
- ***Diseño:*** Elaboración de un documento con el modelo del sistema
- ***Fabricación:*** Construcción del sistema
- ***Prueba:*** Comprobación de que el sistema cumple las especificaciones requeridas
- ***Instalación:*** Entrega del sistema al cliente y garantía de que es operativo
- ***Mantenimiento:*** Reparación de los fallos que aparecen en el sistema

Se puede separar en tres fases:

- ***La definición:*** Análisis de Sistemas; Análisis de Requisitos
- ***El desarrollo:*** Diseño; Codificación; Prueba
- ***El mantenimiento:*** Correctivo; Adaptativo; Perfectivo; Preventivo

Estándar ISO/IEC/IEEE 12207:2017



Clasificación de los modelos de proceso

Existen diferentes formas de clasificar los modelos de proceso en función de sus características y los tipos de sistemas a los que se aplican:

Modelos tradicionales:

Formados por un conjunto de fases o actividades en las que no tienen en cuenta la naturaleza evolutiva del software

- Clásico, lineal o en cascada
- Estructurado
- Basado en prototipos
- Desarrollo rápido de aplicaciones (RAD)

Clasificación de los modelos de proceso

Modelos evolutivos:

Son modelos que se adaptan a la evolución que sufren los requisitos del sistema en función del tiempo

- Incremental
- Iterativo
- En espiral

Modelos para sistemas orientados a objetos:

Modelos con un alto grado de iteratividad y solapamiento entre fases

- De agrupamiento
- Proceso Unificado

Clasificación de los modelos de proceso

Modelos basados en reutilización:

Tienen en cuenta la reutilización sistemática del software

- Basado en componentes
- Proceso Unificado

Procesos ágiles:

Enfatizan el desarrollo rápido, ponen el énfasis en la programación

- Programación extrema (XP)
- Scrum
- Desarrollo de software adaptativo
- Crystal

Clasificación de los modelos de proceso

Modelos para sistemas web:

Creados específicamente para el desarrollo de aplicaciones web

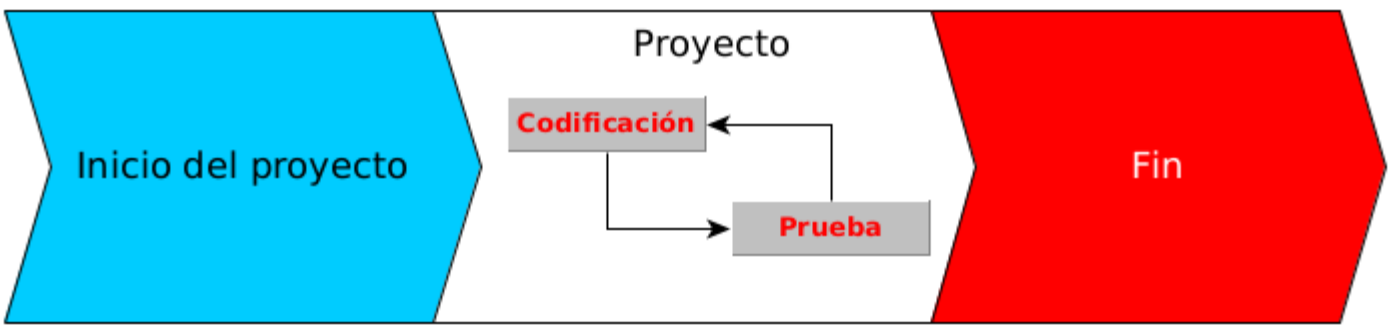
- Modelos de Pressman
- UML-based Web Engineering

Modelos tradicionales

Modelos para el desarrollo de software:

Modelo primitivo

- Se le conoce también con el nombre de Modelo Prueba y Error o Modelo Codifica y Mejora
- Proceso de desarrollo aplicado en las primeras experiencias de programación
- Supone una iteración de fases codificación-depuración sin ninguna planificación ni diseños previos



Modelos para el desarrollo de software:

Modelo primitivo

Inconvenientes:

- Código pobremente estructurado tras varias iteraciones
- Código espagueti
- Caro de desarrollar por las numerosas recodificaciones
- Posible rechazo del usuario al no existir análisis de requisitos
- Caro de depurar por la falta de planificación
- Caro de mantener por la falta de estructura y documentación

Modelos para el desarrollo de software:

Big - bang

Modelos para el desarrollo de software:

Big-bang

- El modelo Big Bang es un modelo donde no seguimos ningún proceso específico.
- El desarrollo sólo comienza con el dinero y los esfuerzos requeridos como entrada, y la salida es el software desarrollado que puede ser o no según los requisitos del cliente.
- Este modelo de Big Bang no sigue un proceso / procedimiento y se requiere muy poca planificación. Incluso el cliente no está seguro de lo que quiere exactamente y los requisitos se implementan sobre la marcha sin mucho análisis.
- Por lo general, este modelo se sigue para proyectos pequeños donde los equipos de desarrollo son muy pequeños.

Modelos para el desarrollo de software:

Big-bang

- El modelo Big Bang consiste en enfocar todos los recursos posibles en el desarrollo y codificación de software, con muy poca o ninguna planificación.
- Los requisitos se entienden e implementan a medida que se presentan.
- Cualquier cambio requerido puede o no necesitar renovar el software completo.
- Este modelo es ideal para proyectos pequeños con uno o dos desarrolladores trabajando juntos y también es útil para proyectos académicos o de práctica.
- Es un modelo ideal para el producto donde los requisitos no se entienden bien y no se da la fecha de lanzamiento final.
- La ventaja de este modelo Big Bang es que es muy simple y requiere muy poca o ninguna planificación.

Modelos para el desarrollo de software:

Big-bang

- Fácil de administrar y no se requieren procedimientos formales.
- Sin embargo, el modelo Big Bang es un modelo de muy alto riesgo y los cambios en los requisitos o los requisitos malentendidos pueden incluso conducir a una reversión completa o raspado del proyecto.
- Es ideal para proyectos repetitivos o pequeños con riesgos mínimos.

Modelos para el desarrollo de software:

Big-bang

Ventajas:

- Este es un modelo muy simple.
- Se requiere poca o ninguna planificación
- Fácil de manejar
- Muy pocos recursos requeridos
- Da flexibilidad a los desarrolladores
- Es una buena ayuda de aprendizaje para recién llegados o estudiantes.

Modelos para el desarrollo de software:

Big-bang

Desventajas:

- Muy alto riesgo e incertidumbre.
- No es un buen modelo para proyectos complejos y orientados a objetos.
- Modelo deficiente para proyectos largos y en curso.
- Puede resultar muy costoso si se malinterpretan los requisitos.

Modelos para el desarrollo de software:

Modelos lineales o secuenciales

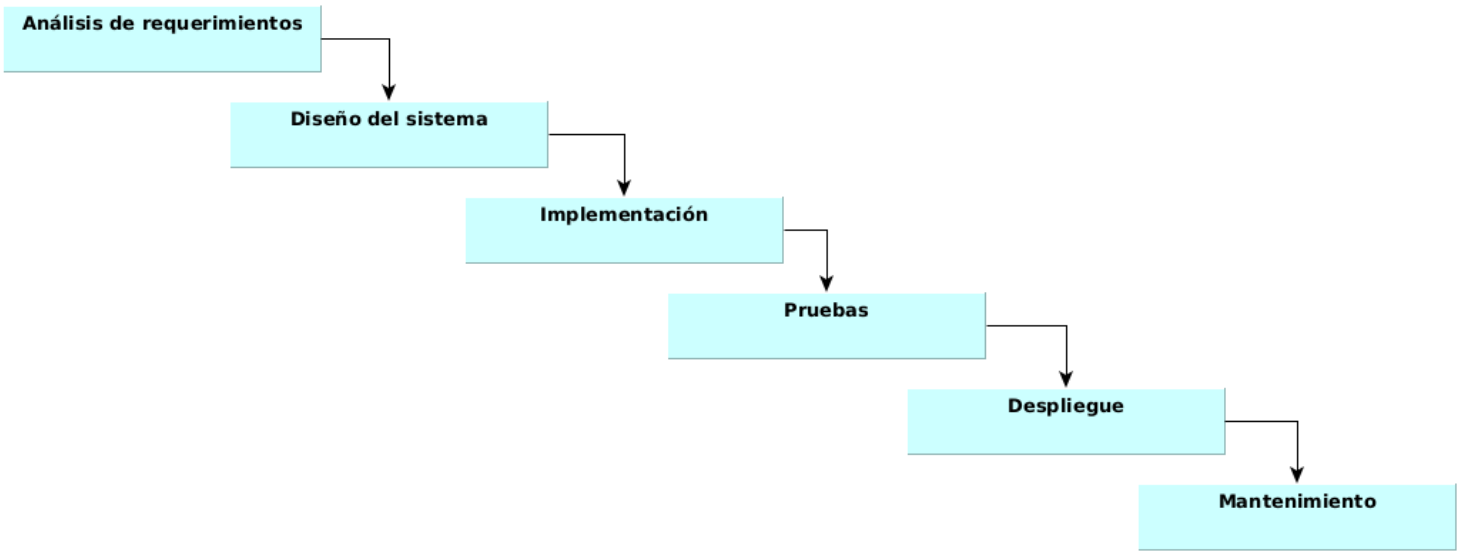
- Han sido ampliamente utilizados
- Ofrecen grandes facilidades a los gestores para controlar el progreso de los proyectos
- Proponen un enfoque sistemático, secuencial, para el desarrollo del software
- Comienza en un nivel de sistemas y progresa con el análisis, diseño, codificación, pruebas y mantenimiento
- Plantean fases separadas en la especificación y el desarrollo

Inconvenientes

- La filosofía de estos modelos de proceso no es realista
- No se ajusta al proceso de desarrollo software (raramente sigue un flujo secuencial sino que exige diversas iteraciones)
- No ofrece un soporte adecuado a las técnicas de desarrollo basadas en objetos y componentes

Modelos para el desarrollo de software:

El modelo en cascada.



Modelos para el desarrollo de software:

El modelo en cascada.

- Conocido también como modelo lineal o Clásico
- Versión original se debe a W. Royce [Royce, 1970]
- Se encuentra definido en la norma estándar 2167-A del DoD de EEUU

Características:

- Considera las actividades fundamentales del proceso especificación, desarrollo, validación y evolución.
- Los representa como fases separadas del proceso, tales como la especificación de requerimientos, el diseño del software, la implementación, las pruebas, etcétera.
- Paso de fase al conseguir los objetivos
- Obtención de documentos como criterio de finalización de fase
- El final de una fase puede suponer un punto de revisión

Modelos para el desarrollo de software:

El modelo en cascada.

Análisis y definición de requerimientos.

- Los servicios restricciones y metas del sistema se definen a partir de las consultas con los usuarios.
- Entonces, se definen en detalle y sirven de manera específica al sistema.

Diseño del sistema y del software.

- El proceso de diseño del sistema divide los requerimientos en sistemas.
- Establece una arquitectura completa del sistema.
- El diseño del software describe los elementos abstractos que son fundamentales para el software y sus relaciones.

Modelos para el desarrollo de software:

El modelo en cascada.

Implementaciones prueba de unitarias.

- Durante esta etapa el diseño del software se lleva a cabo como un conjunto de unidades de programas.
- La prueba unitaria implica verificar que cada una cumpla con su función específica.

Integración y prueba del sistema.

- Los programas o las unidades individuales de programas se integran y se prueban como un sistema completo para así asegurar que se cumplan los requerimientos del software.
- El software se entrega al cliente.

Modelos para el desarrollo de software:

El modelo en cascada.

Funcionamiento y mantenimiento.

- En esta fase el sistema se instala y se pone en funcionamiento práctico.
- El mantenimiento implica corregir errores no descubiertos en las etapas anteriores del ciclo de vida.
- Se pueden mejorar la implementación de las unidades del sistema y resaltar los servicios del sistema de acuerdo a nuevos requerimientos.

Modelos para el desarrollo de software:

El modelo en cascada.

Ventajas:

- Simple y fácil de entender y usar
- Fácil de manejar debido a la rigidez del modelo.
- Cada fase tiene entregables específicos y un proceso de revisión.
- Las fases se procesan y se completan una a la vez.
- Funciona bien para proyectos más pequeños donde los requisitos se entienden muy bien.
- Etapas claramente definidas.

Modelos para el desarrollo de software:

El modelo en cascada.

Ventajas:

- Hitos bien entendidos.
- Tareas fáciles de organizar.
- El proceso y los resultados están bien documentados.

Modelos para el desarrollo de software:

El modelo en cascada.

Desventajas

- No se produce ningún software que funcione hasta el final del ciclo de vida.
- Grandes cantidades de riesgo e incertidumbre.
- No es un buen modelo para proyectos complejos y orientados a objetos.
- Modelo deficiente para proyectos largos y en curso.
- No es adecuado para proyectos donde los requisitos tienen un riesgo de cambio moderado a alto. Entonces, el riesgo y la incertidumbre son altos con este modelo de proceso.
- Es difícil medir el progreso dentro de las etapas.

Modelos para el desarrollo de software:

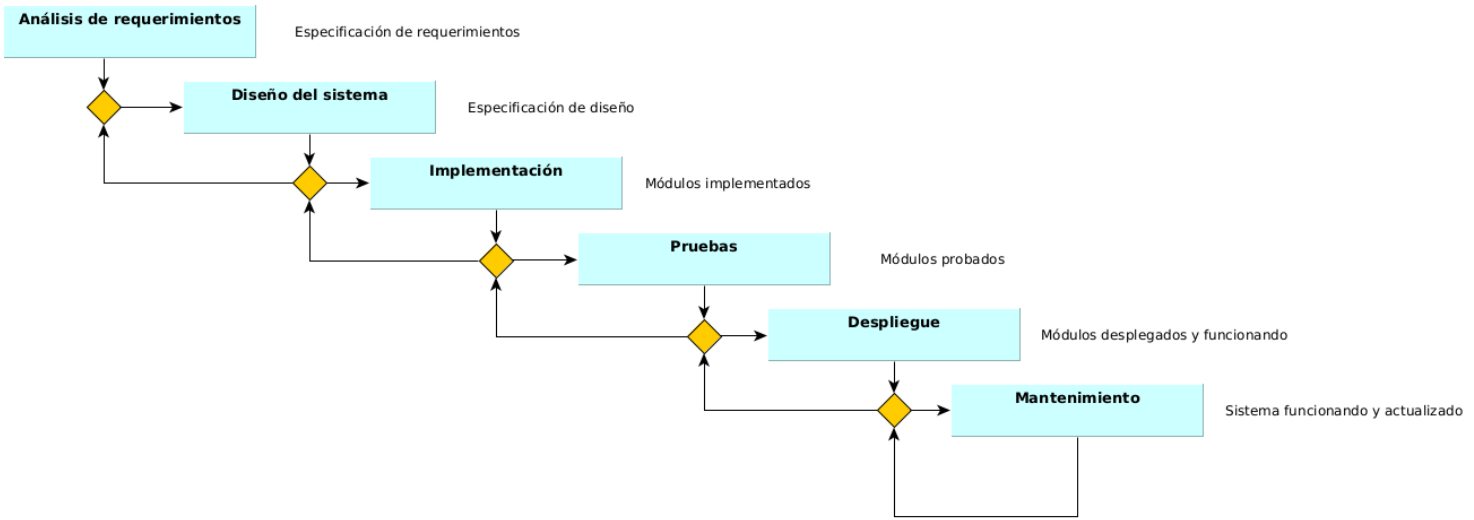
El modelo en cascada.

Desventajas

- No se pueden adaptar los requisitos cambiantes.
- Ajustar el alcance durante el ciclo de vida puede finalizar un proyecto.
- La integración se realiza como un "big-bang" al final, que no permite identificar temprano ningún cuello de botella tecnológico o comercial o desafíos.
- La linealidad no se corresponde con la realidad del proceso de software

Modelos para el desarrollo de software:

El modelo en cascada.



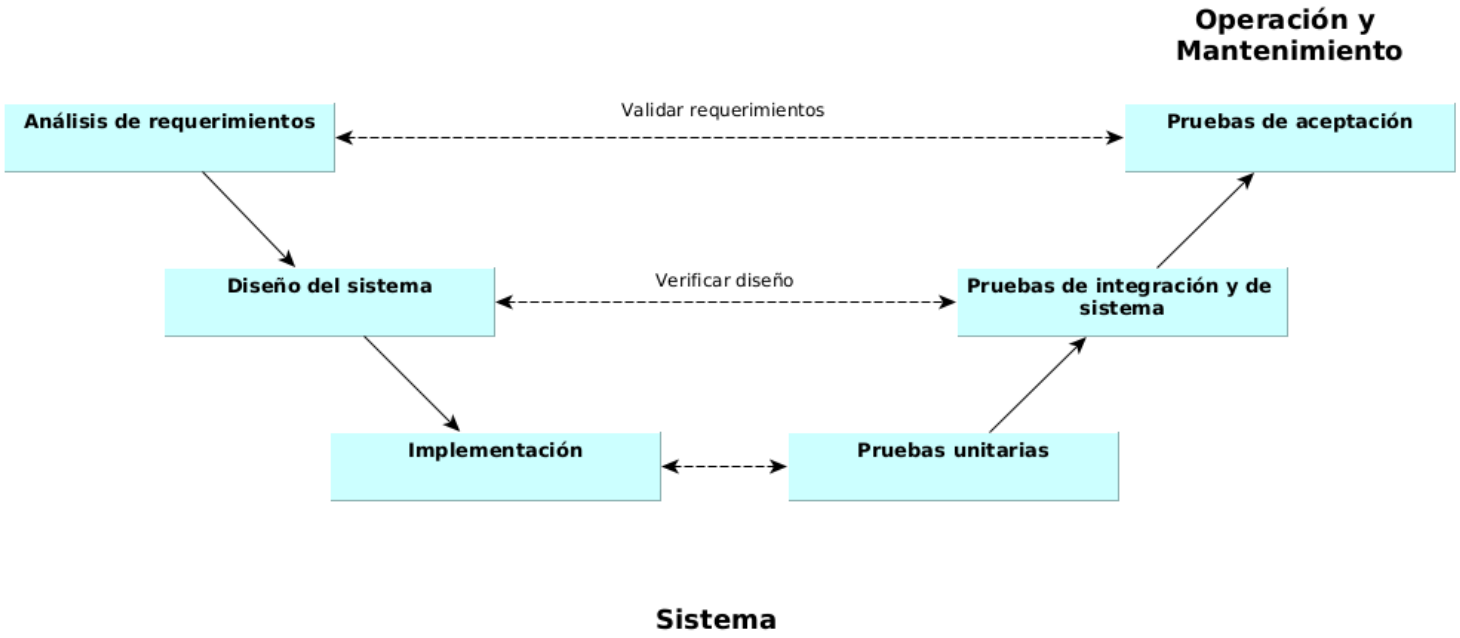
Modelos para el desarrollo de software:

El Modelo en V

- Es una variación del modelo en cascada que demuestra cómo se relacionan las actividades de prueba con las de análisis y desarrollo [GMD, 1992]
- Presenta una implantación ascendente
- Demuestra que el desarrollo de las pruebas se efectúa de manera síncrona con el desarrollo del programa
- Mientras que el modelo clásico centra su atención en los documentos y artefactos producidos, el modelo en V lo hace en la actividad y la exactitud

Modelos para el desarrollo de software:

El Modelo en V



Modelos para el desarrollo de software:

Modelos basados en prototipos

Un prototipo es un modelo experimental de un sistema o de un componente de un sistema que tiene los suficientes elementos que permiten su uso

- Los prototipos son un medio eficaz para aclarar los requisitos de los usuarios e identificar las características de un sistema que deben cambiarse o añadirse
- Mediante el prototipo se puede verificar la viabilidad del diseño de un sistema
- Es una aplicación que funciona
- Su finalidad es probar varias suposiciones con respecto a las características requeridas por el sistema
- Se crean con rapidez
- Evolucionan a través de un proceso iterativo
- Tienen un costo bajo de desarrollo

Modelos para el desarrollo de software:

Modelos basados en prototipos

Enfoques de desarrollo:

- ***Desechable:*** El prototipo es una versión rudimentaria del sistema que posteriormente es desechada
- ***Evolutivo:*** El prototipo debe convertirse, eventualmente, en el sistema final usado (alternativa al ciclo de vida) [Basili y Turner, 1975]
- ***Mixto:*** (prototipado operativo) [Davis, 1992]
  - *Se aplican técnicas convencionales para los requisitos bien conocidos*
  - *Combinación de prototipos desechables y evolutivos para los requisitos poco conocidos*

Modelos para el desarrollo de software:

Modelos basados en prototipos

Diferencias entre los prototipos desechables y evolutivos:

Desechable:

- Se enfoca el desarrollo en un proceso *rápido y sin rigor*
- Se enfoca en construir *sólo las partes problemáticas*
- El diseño se centra en *optimizar el tiempo de desarrollo*
- El objetivo último es *desecharlo*

Evolutivo:

- Se enfoca el desarrollo en un proceso *riguroso*
- Se enfoca en construir *primero las partes bien entendidas, sobre una base sólida*
- El diseño se centra en *optimizar la modificabilidad*
- El objetivo último es *incluirlo en el sistema*

Modelos para el desarrollo de software:

Modelos basados en prototipos

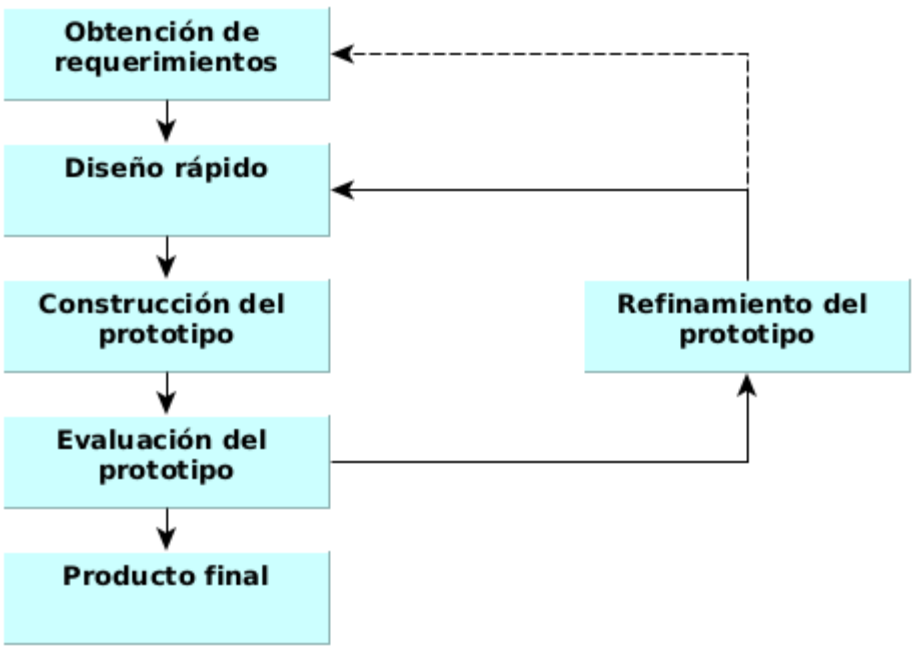
Prototipos desechables:

- Se desarrolla código para explorar factores críticos para el éxito del sistema
- La implementación usa lenguajes y/o métodos de desarrollo más rápidos que los definitivos
- Se usa como herramienta auxiliar de la especificación de requisitos y el diseño y permiten:
  - *Determinar la viabilidad de los requisitos*
  - *Validar la funcionalidad del sistema*
  - *Encontrar requisitos ocultos*
  - *Determinar la viabilidad de la interfaz de usuario*
  - *Examinar alternativas de diseño*
  - *Validar una arquitectura de diseño particular*
- Este enfoque suele derivar en un modelo lineal una vez que el prototipo ha cumplido su misión

Modelos para el desarrollo de software:

Modelos basados en prototipos

Prototipos desechables:



Modelos para el desarrollo de software:

Modelos basados en prototipos

Aplicaciones de los prototipos desechables:

- Interfaz de usuario
- Formatos de informes
- Formatos de gráficos
- Organización de bases de datos
- Rendimiento de bases de datos
- Precisión e implementación de cálculos complejos
- Partes con respuesta crítica en el tiempo en sistemas de tiempo real
- Rendimiento de sistemas interactivos
- Viabilidad de partes del sistema en las que no se tiene experiencia

Modelos para el desarrollo de software:

Modelos basados en prototipos

Prototipado evolutivo:

- Enfoque de desarrollo que se utiliza cuando no se conoce con seguridad lo que se quiere construir
- Se comienza diseñando e implementando las partes más destacadas del sistema
- La evaluación del prototipo proporciona la realimentación necesaria para aumentar y refinar el prototipo
- El prototipo evoluciona y se transforma en el sistema final
- Permite solventar objeciones del usuario
- Sirve para formalizar la aceptación previa
- Introduce flexibilidad en la captura de requisitos

Modelos para el desarrollo de software:

Modelos basados en prototipos

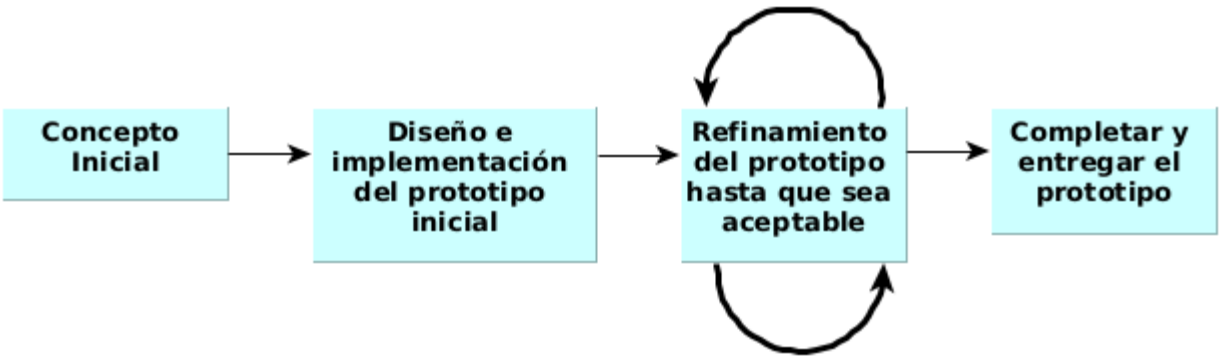
Prototipado evolutivo:

- Es útil cuando el área de aplicación no está definida, cuando el riesgo de rechazo el alto, o como forma de evaluar el impacto de una aplicación
- El prototipado es un subproceso que puede incluirse como parte de otros modelos de proceso, por ejemplo, puede combinarse con un ciclo en cascada para intentar solventar ciertas carencias de este.
- El sistema se puede llegar a deteriorar tendiendo hacia el modelo primitivo
- Se suele refinar el prototipo hacia el sistema final en lugar de desecharlo y empezar desde el principio
- El cliente puede encontrar atractivo el prototipo y quedarse con el prototipo como sistema final
- Relajación de los desarrolladores
- No disminuye el tiempo entre la definición de los requisitos y la entrega del producto

Modelos para el desarrollo de software:

Modelos basados en prototipos

Prototipado evolutivo:



Modelos para el desarrollo de software:

Desarrollo rápido de aplicaciones

- El modelo de desarrollo rápido de aplicaciones, (***RAD – Rapid Application Development***) o modelo de la caja de tiempo surgió como respuesta al modelo formal y al ciclo en espiral
- Enfatiza un ciclo de desarrollo extremadamente corto (60 o 90 días)
- No es un modelo bien definido:
  - *Secuencia de integraciones de un sistema evolutivo o de prototipos que se revisan con el cliente è descubrimiento de los requisitos*
  - *Cada integración se restringe a un período de tiempo bien definido (caja de tiempo)*
- Es un modelo secuencial (Separación en fases de cada caja de tiempo)
- Presenta integraciones constantes y está centrado en el código más que en la documentación
- Usa el desarrollo basado en componentes y se basa en el uso efectivo de herramientas y frameworks

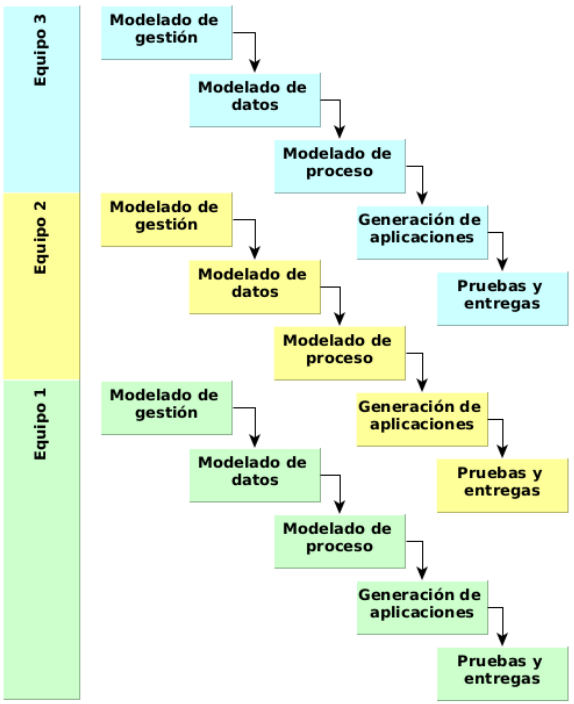
Modelos para el desarrollo de software:

Desarrollo rápido de aplicaciones

- Requiere la participación activa del usuario
- Cuando se utiliza en S.I. Automatizados, comprende las fases de:
  - *Modelado de gestión*
  - *Modelado de datos*
  - *Modelado del proceso*
  - *Generación de aplicaciones*
  - *Pruebas y entrega*
- Las limitaciones de tiempo demandan un ámbito de escalas
- Si una aplicación de gestión puede modularse de forma que pueda completarse cada una de las funciones principales en menos de tres meses, es un candidato del DRA.
- Cada una de estas funciones puede ser afrontadas por un equipo DRA diferente y ser integradas en una sola aplicación

Modelos para el desarrollo de software:

Desarrollo rápido de aplicaciones



Modelos evolutivos

Modelos para el desarrollo de software:

Modelos evolutivos

- El software , al igual que todos los sistemas complejos, evolucionan con el tiempo [Gilb, 1988]
- Se caracterizan porque permiten desarrollar versiones cada vez más completas del software , teniendo en cuenta la naturaleza evolutiva del software
- Presentan la filosofía de poner un producto en explotación cuanto antes (prototipado evolutivo)
- Existen muchos modelos de proceso evolutivos y tienen la característica de ser iterativos
- Los desarrollos orientados a objetos se ajustan a un modelo de proceso iterativo e incremental y lo mismo se puede afirmar para los desarrollos basados en componentes
- Esto es así porque las tareas de cada fase se llevan a cabo de una forma iterativa y se desarrollan en un ciclo de desarrollo ***análisis-diseño-implementación-análisis*** que permite hacer evolucionar al sistema

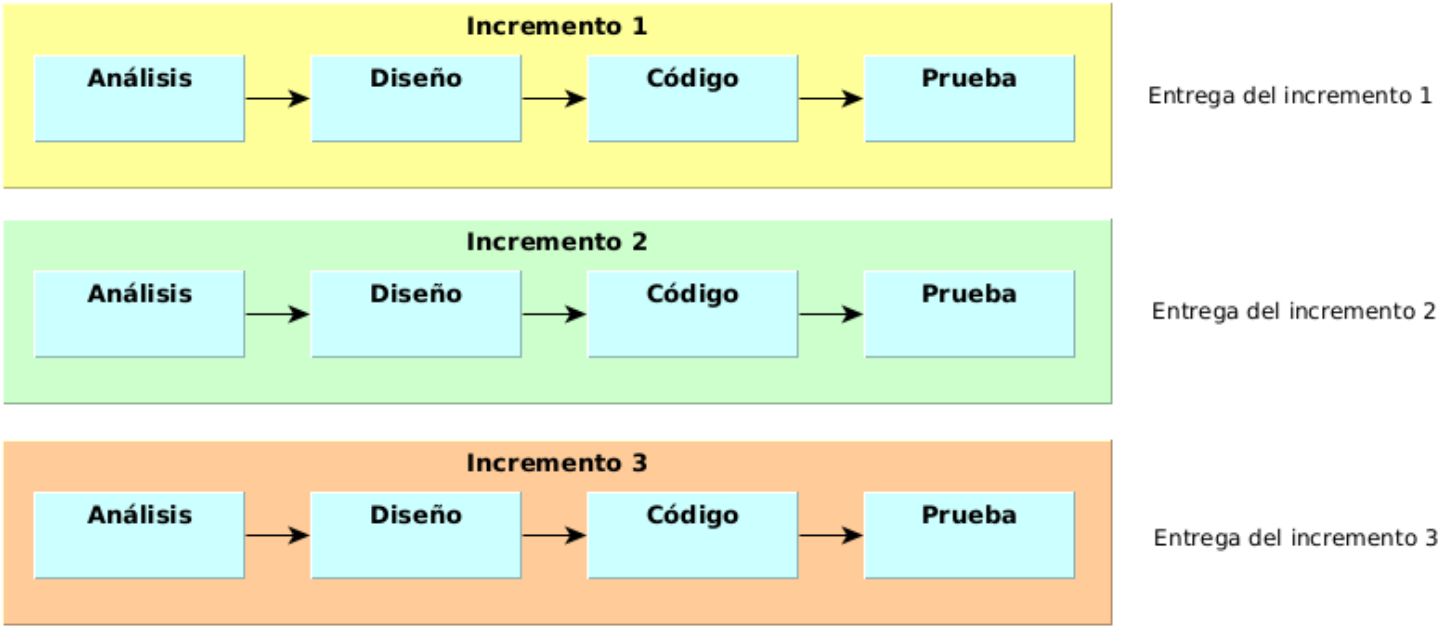
Modelos para el desarrollo de software:

Modelo Incremental

- En este modelo, el sistema, tal y como está especificado en la especificación de requisitos del software , se divide en subsistemas de acuerdo a su funcionalidad
- Las versiones se definen comenzando con un subsistema funcional pequeño y agregando funcionalidad con cada nueva versión
- Cada nueva parte entregada se denomina incremento
- Combina elementos del modelo en cascada con la filosofía interactiva de construcción de prototipos
- Aplica secuencias lineales de forma escalonada mientras progresa el calendario del proyecto
- Cada secuencia lineal supone un incremento

Modelos para el desarrollo de software:

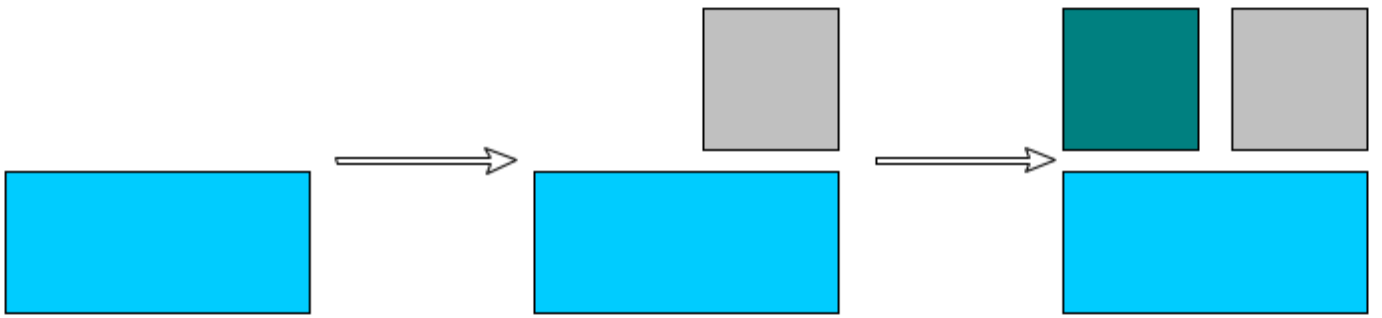
Modelo Incremental



Modelos para el desarrollo de software:

Modelo Incremental

Desarrollo incremental: sistema parcial, funcionalidad completa



Modelos para el desarrollo de software:

El modelo Iterativo

- Entrega un sistema completo desde el principio, para posteriormente cambiar la funcionalidad de cada subsistema con cada versión
- Mejora iterativamente las versiones en evolución hasta que el sistema completo se implemente y esté listo para desplegarse.
- No intenta comenzar con una especificación completa de los requisitos.
- En cada iteración, se realizan modificaciones de diseño y se agregan nuevas capacidades funcionales.
- Se basa en la evolución de prototipos ejecutables, mensurables y evaluables
- Se van incorporando cambios en cada iteración
- Exige más atención e implicación de todos los actores del proyecto

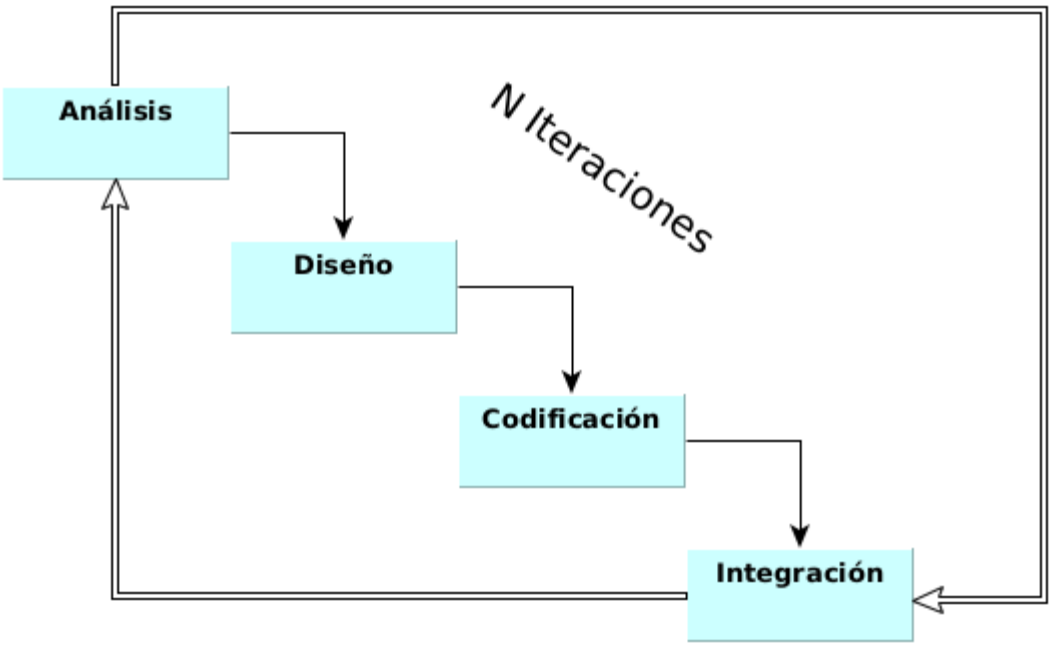
Modelos para el desarrollo de software:

El modelo Iterativo

- Cada iteración reproduce el ciclo de vida en cascada, pero a una escala menor
- Los objetivos de cada iteración se establecen en función de la evaluación de las iteraciones precedentes
- Las fases tradicionales se cubre gradualmente en las diversas iteraciones
- Las actividades internas se solapan porque dentro de una iteración no necesitan terminarse de golpe, siendo la transición entre dos actividades progresiva
- Para la evaluación de las iteraciones:
  - *Deben definirse criterios de evaluación de las iteraciones*
  - *Una iteración se marca por etapas intermedias que permitan medir los progresos.*
  - *Debe haber al menos dos etapas de revisión una inicial y una de evaluación*
  - ***Revisión inicial:*** *fija los objetivos y criterios de la iteración*
  - ***Revisión de evaluación:*** *valida los resultados*

Modelos para el desarrollo de software:

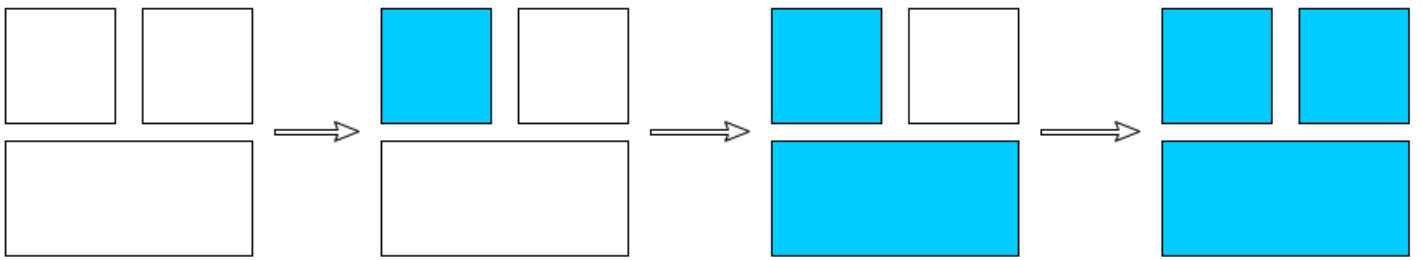
El modelo Iterativo.



Modelos para el desarrollo de software:

El modelo Iterativo.

Desarrollo iterativo: sistema completo; funcionalidad parcial



Modelos para el desarrollo de software:

Modelo en espiral

- Propuesto inicialmente por B. Boehm
- Es un modelo de proceso evolutivo, que proporciona el potencial para el desarrollo rápido de versiones incrementales del software
- Puede considerarse como un metamodelo de proceso
- Reúne características del modelo clásico y de prototipos
- Aparece el análisis de riesgo
- Se divide en un número de actividades estructurales, también denominadas regiones de tareas.
- En el modelo original de Boehm aparecen cuatro regiones de tareas:
  - *Planificación*
  - *Análisis de riesgos*
  - *Ingeniería*
  - *Evaluación del cliente*

Modelos para el desarrollo de software:

Modelo en espiral

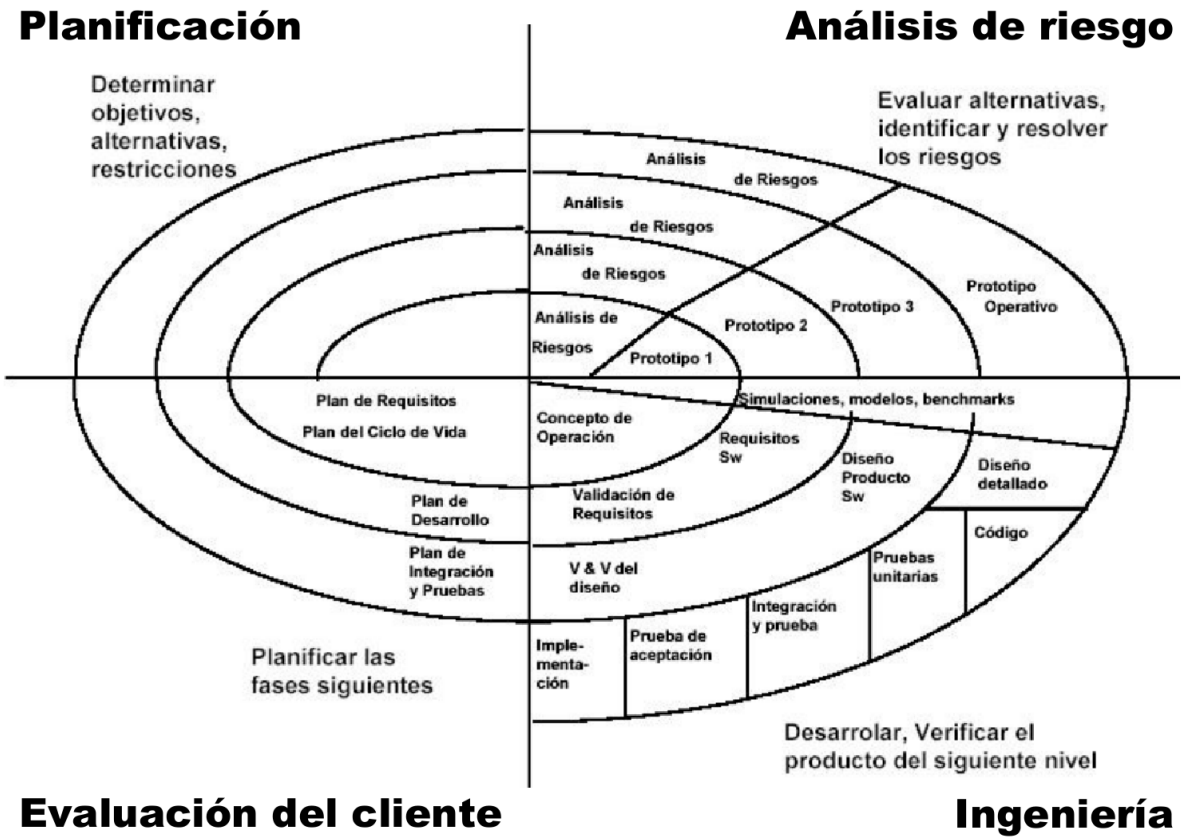
- El avance se realiza desde el centro de la espiral hacia el exterior
- Existe un reconocimiento explícito de las diferentes alternativas para alcanzar los objetivos del proyecto
- El modelo se centra en identificar los riesgos de cada alternativa, así como las formas de solventarlos
- La división de los proyectos en ciclos, cada uno con un acuerdo al final, implica que existe un acuerdo para los cambios a realizar o para la finalización del mismo, en función de lo aprendido a lo largo del proyecto
- Es un método que se adapta a cualquier tipo de actividad, alguna de las cuales no existen en otros paradigmas, como puede ser la consulta a asesores externos

Variantes:

- Modelo en espiral de Pressman
- Modelo win-win

Modelos para el desarrollo de software:

Modelo en espiral



Modelos para el desarrollo de software:

Modelo en espiral

Ventajas:

- Refleja de forma más realista la idiosincrasia del desarrollo de software
- Toma lo mejor y evita lo peor de los demás modelos, según la situación en cada momento
- Las opciones de reutilización se tienen en cuenta desde el primer momento
- Proporciona una preparación para la evolución, crecimiento y cambio
- Proporciona un mecanismo para incorporar objetivos de calidad en el desarrollo
- Se centra en la eliminación de errores y opciones no atractivas desde el principio
- Determina el nivel de esfuerzo de cada fase en cada proyecto

Modelos para el desarrollo de software:

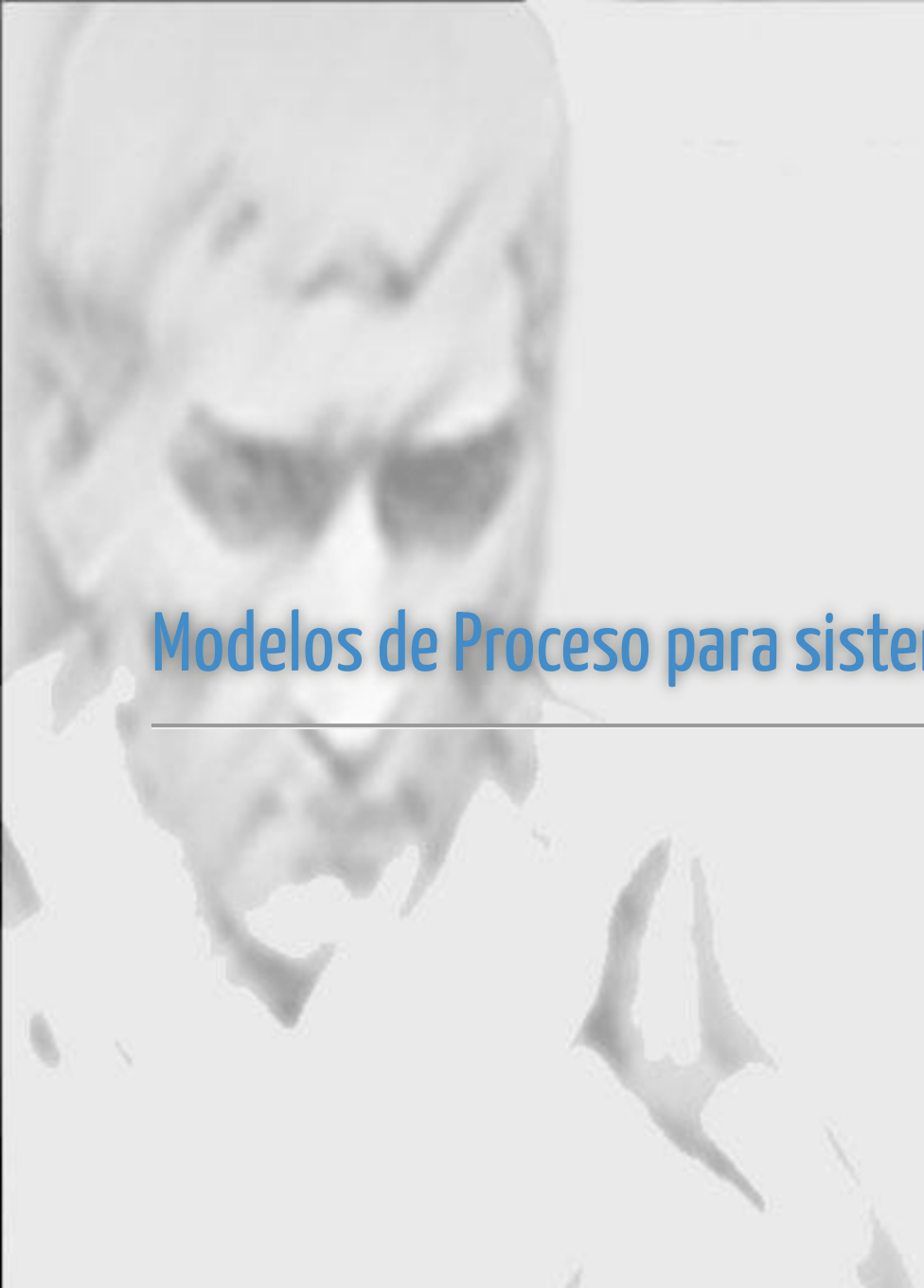
Modelo en espiral

Ventajas:

- Se sigue el mismo procedimiento para el desarrollo que para el mantenimiento, con lo que se evitan los problemas de las “mejoras rutinarias” de alto riesgo
- Permite una gran flexibilidad
- Se adapta bien al diseño y programación orientado a objetos

Desventajas:

- No ha sido desarrollado en el mundo de la contratación comercial sino en el de desarrollo interno
- Puede resultar difícil convencer a grandes clientes de que el enfoque evolutivo es controlable
- Necesita experiencia en la evaluación de riesgos
- Necesita una elaboración adicional de los pasos del modelo



Modelos de Proceso para sistemas orientados a objetos

Modelos para el desarrollo de software:

Modelos de Proceso para sistemas orientados a objetos

Surgieron por la dificultad de adaptar los modelos tradicionales al desarrollo de sistemas orientados a objetos

Características principales:

- Eliminación de las fronteras entre fases
- Uso de componentes reutilizables
- Alto grado de iteratividad y solapamiento
- Desarrollo incremental

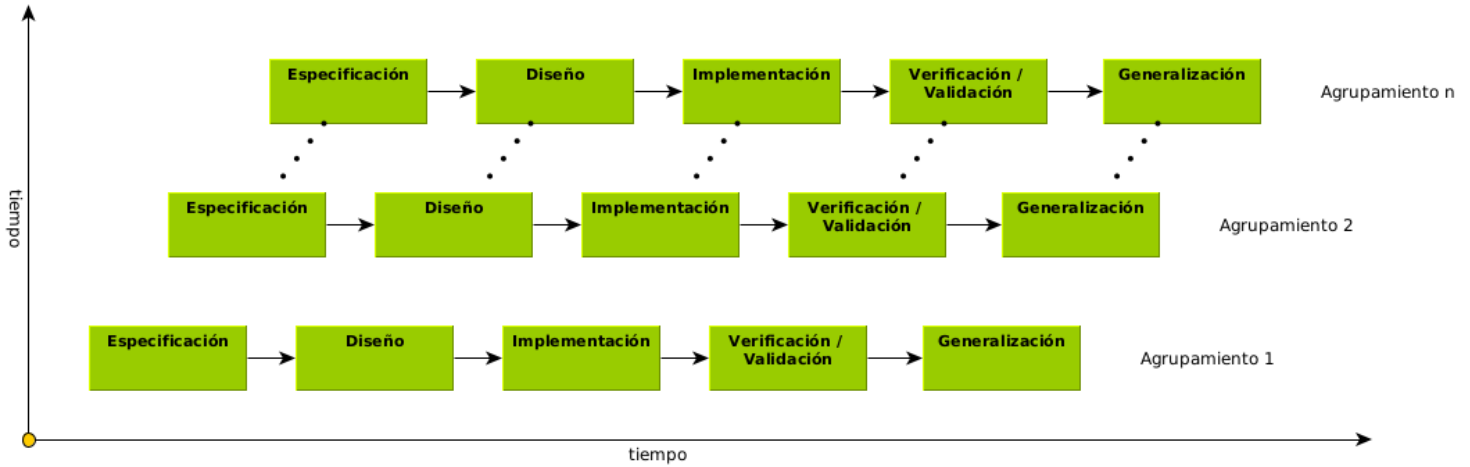
Modelos para el desarrollo de software:

Modelo de agrupamiento

- Propuesto por Bertrand Meyer
- Concepto clave: AGRUPAMIENTO (cluster)
  - *Unidad organizativa básica*
  - *Grupo de clases relacionadas o, recursivamente, clusters relacionados*
  - *Unidad natural para el desarrollo por parte de un único desarrollador*
  - *Evita el efecto todo-nada propio del modelo en cascada*
- Tiene un componente secuencial y un componente concurrente
  - *Existencia de diferentes subciclos de vida (uno para cada cluster) que pueden solaparse en el tiempo*
  - *Cada subciclo de vida que gobierna el desarrollo de un cluster está formado por:* ***especificación***, ***diseño***, ***implementación***, ***verificación/validación*** *y* ***generalización***
- Tiene un enfoque ascendente
- La ocultación de la información posibilita la forma del modelo de clusters de ingeniería concurrente

Modelos para el desarrollo de software:

Modelo de agrupamiento



Modelos para el desarrollo de software:

Proceso Unificado

- Definido por ***Rational Software Corporation***
- Representa la evolución del proceso ***Objectory*** de Rational
- Utilización de ***UML*** como lenguaje de modelado
- Basado en componentes
- **Conducido por casos de uso:**
  - *Los casos de uso se implementan para asegurar que toda la funcionalidad se realiza en el sistema y verificar y probar el mismo.*
  - *Como los casos de uso contienen las descripciones de las funciones, afectan a todas las fases y vistas.*

Modelos para el desarrollo de software:

Proceso Unificado

- **Centrado en la arquitectura:**
  - *La arquitectura se describe mediante diferentes vistas del sistema.*
  - *Es importante establecer una arquitectura básica pronto, realizar prototipos, evaluarla y finalmente refinarla durante el curso del proyecto.*
- **Iterativo e incremental:**
  - *Resulta práctico dividir los grandes proyectos en mini proyectos, cada uno de los cuales es una iteración que resulta en un incremento.*
- El Proceso Unificado se repite a lo largo de una ***serie de ciclos*** y cada ciclo consta de ***cuatro fases***.
- Cada ciclo concluye con una ***versión del producto*** para los clientes.

Modelos para el desarrollo de software:

Proceso Unificado

Fases del proceso unificado:

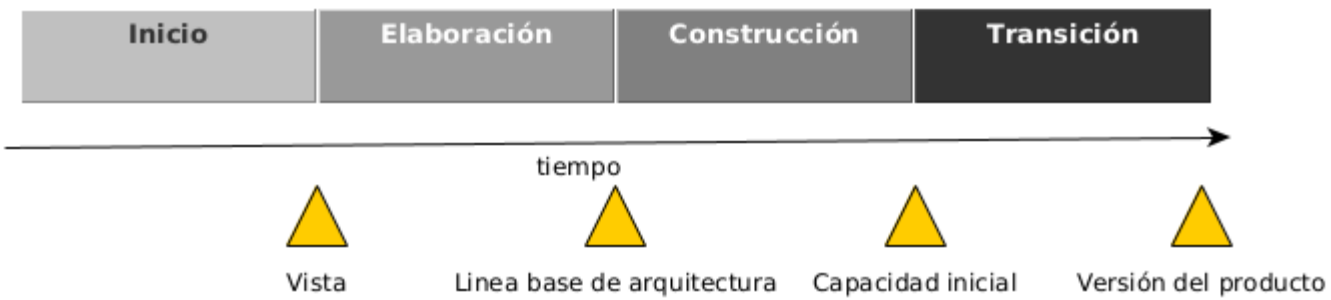
- ***Inicio:*** se define el alcance del proyecto y se desarrollan los casos de negocio.
- ***Elaboración:*** se planifica el proyecto, se especifican en detalle la mayoría de los casos de uso y se diseña la arquitectura del sistema.
- ***Construcción:*** se construye el producto.
- ***Transición:*** el producto se convierte en versión beta. Se corrigen problemas y se incorporan mejoras sugeridas en la revisión.
- Dentro de cada fase se puede, a su vez, descomponer el trabajo en ***iteraciones*** con sus incrementos resultantes.
- Cada fase termina con un ***hito***, cada uno de los cuales se caracteriza por la disponibilidad de un conjunto de componentes de software.

Modelos para el desarrollo de software:

Proceso Unificado

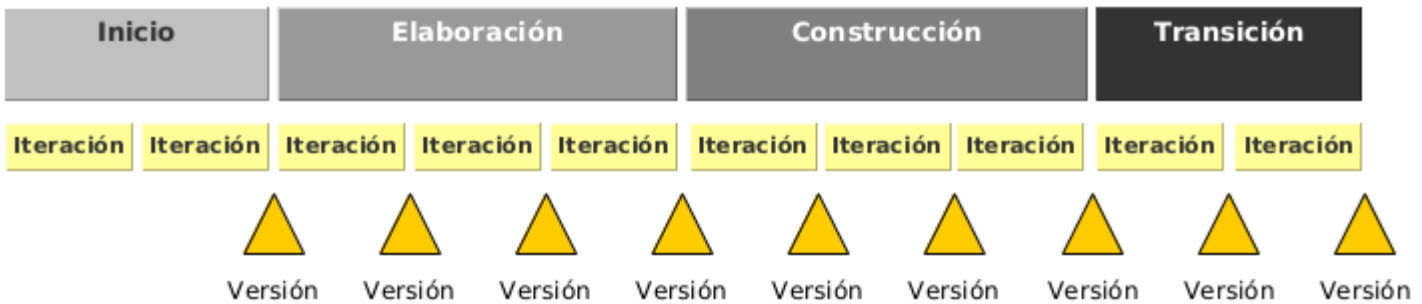
Objetivos de los hitos:

- Toma de decisiones para continuar con la siguiente fase.
- Controlar el progreso del proyecto.
- Proporcionar información para la estimación de tiempo y recursos de proyectos sucesivos.



Modelos para el desarrollo de software:

Proceso Unificado



Modelos para el desarrollo de software:

Proceso Unificado

- Las iteraciones discurren a lo largo de los flujos de trabajo