INGENIERÍA DE SOFTWARE I

UNTDF
Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur

Unidad I

Tema: El Proceso de Desarrollo de Software

UNTDF – 2020

Temario

***Proceso de desarrollo de software**:*

Marco de trabajo de las tareas que se requieren para construir software de alta calidad.

La Ingeniería de software abarca un conjunto de tres elementos clave que se utilizan en todo el proceso: *métodos*, *herramientas* y *procedimientos*

Proceso de Desarrollo de SW

- Proceso: Serie de operaciones usadas en la creación de un producto.

- Proceso de software: serie de actividades relacionadas que conduce a la elaboración de un producto de software.
  - ✓ Estas actividades pueden incluir el desarrollo de software desde cero en un lenguaje de programación estándar como Java o C.
  - ✓ El nuevo software empresarial con frecuencia ahora se desarrolla extendiendo y modificando los sistemas existentes, o configurando e integrando el software comercial o componentes del sistema.

Proceso de Desarrollo de SW

- Las descripciones de procesos deben incluir:

✓ **Productos**, que son los resultados de una actividad del proceso.

✓ **Roles**, que reflejan las responsabilidades de la gente que interviene en el proceso: gerente de proyecto, programador, otros.

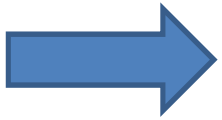
✓ **Precondiciones y postcondiciones**, que son declaraciones válidas antes y después de que se realice una actividad del proceso o se cree un producto. Por ejemplo, antes de comenzar el diseño arquitectónico, una precondición es que el cliente haya aprobado todos los requerimientos.

Actividades del PDSW

En cualquier proceso, existen cuatro grandes **grupos de actividades fundamentales:**

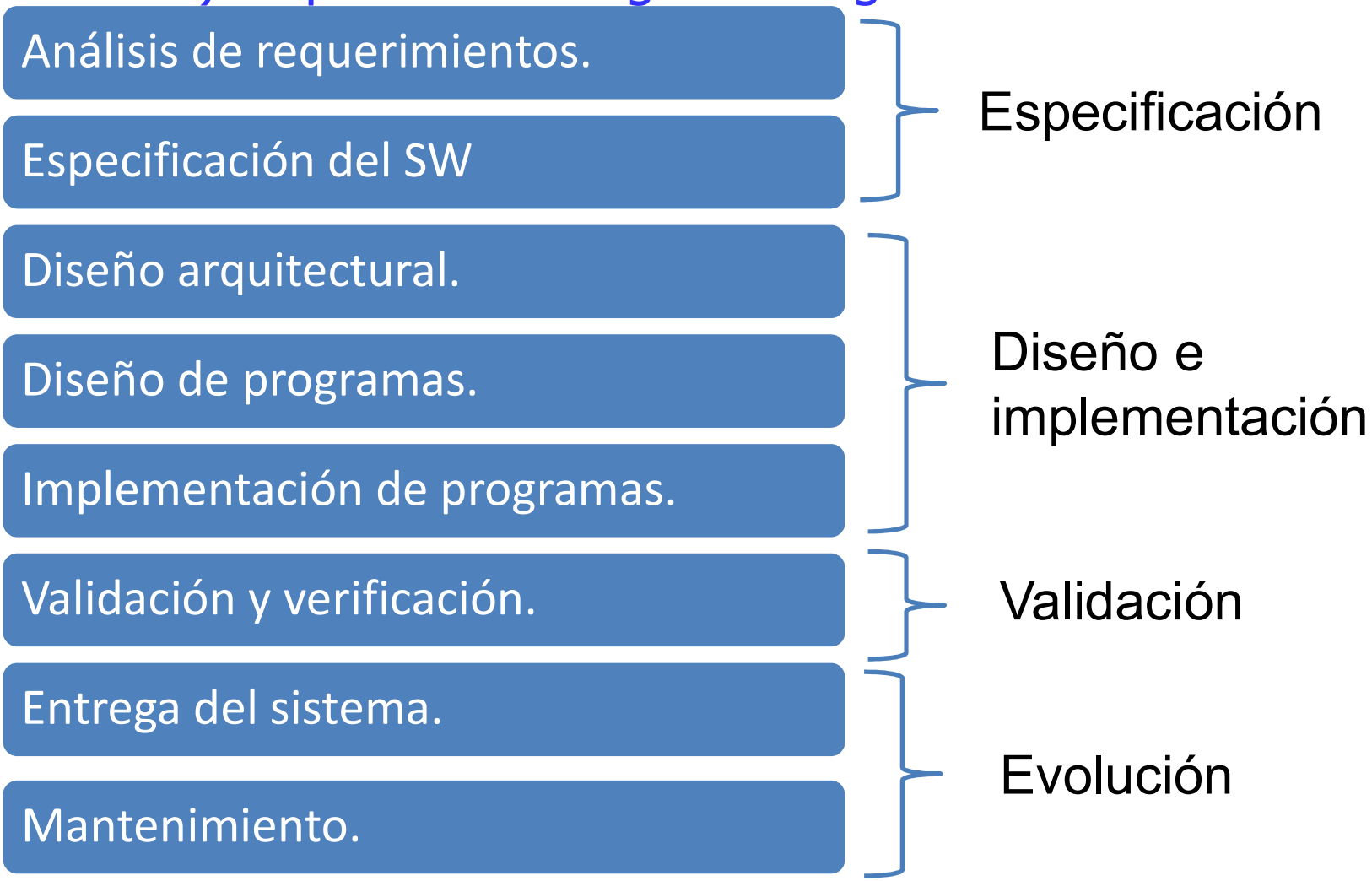
- Especificación del SW
- Diseño e implementación del SW
- Validación del SW
- Evolución del SW

Pueden dividirse en actividades de menor granularidad, como se muestra a continuación



Actividades del PDSW

En un Proceso genérico de SW (independientemente del modelo) se pueden distinguir las siguientes actividades:



Actividades del proceso de desarrollo de SW

- Se describen en forma independiente, indicando rol y resultados
- Utilizan y producen documentos
- Se relacionan e interactúan de diferentes maneras conformando distintos modelos de procesos de desarrollo de software
- De acuerdo al modelo, una actividad puede jugar un rol preponderante o incluso pudiera no existir

Actividad: Requerimientos del SW

Identificar el problema
Recolectar los requerimientos
Involucrar usuarios

Sintetizar y Documentar requerimientos funcionales y no funcionales

Análisis de requerimientos

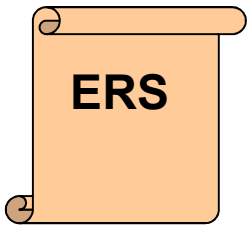
Documentos orientados al usuario y útiles para el analista:

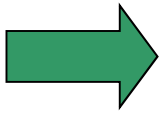
DRS
ERS

Definición de Requerimientos SW
Especificación de Requerimientos SW

Actividad: Especificación del SW

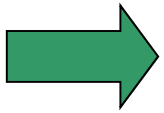
Requerimientos del software
Restricciones
Consideraciones técnicas





Modelado de Funcionalidad y Datos del SW

Especificación del sistema SW



Modelo del SW

Descripción orientada al desarrollador

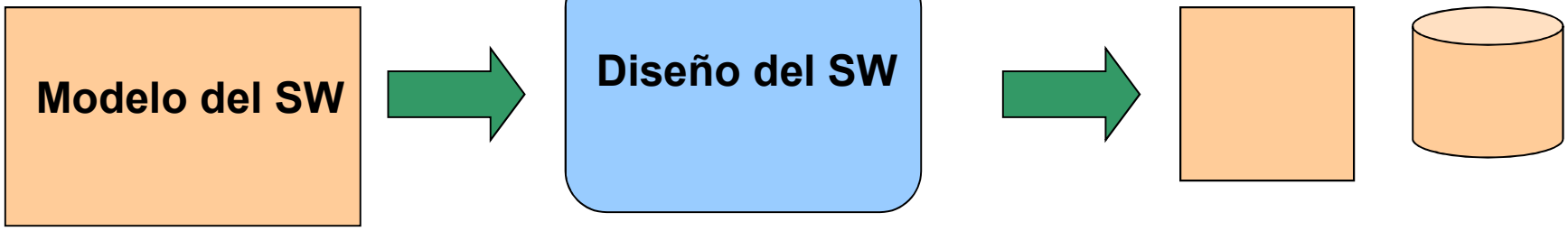
Describe el qué y no el cómo

Input al Diseño

Actividad: Diseño del SW

Especificación
del software
Restricciones
Consideraciones
técnicas

Diseño
arquitectural
Diseño detallado



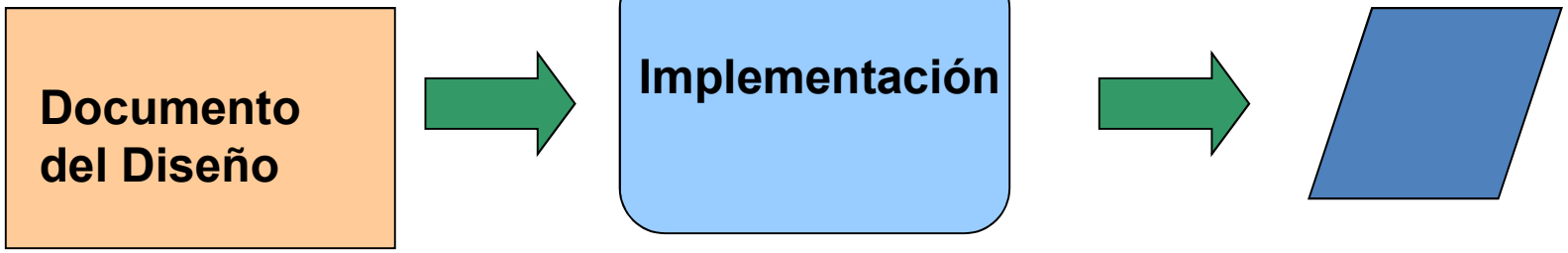
Descripción
detallada del
SW orientada al
implementador

Modelo de
diseño del SW,
BD, Intefaces, …

Actividad: Implementación del SW

Arquitectura y diseño detallado del software.
Restricciones
Consideraciones técnicas

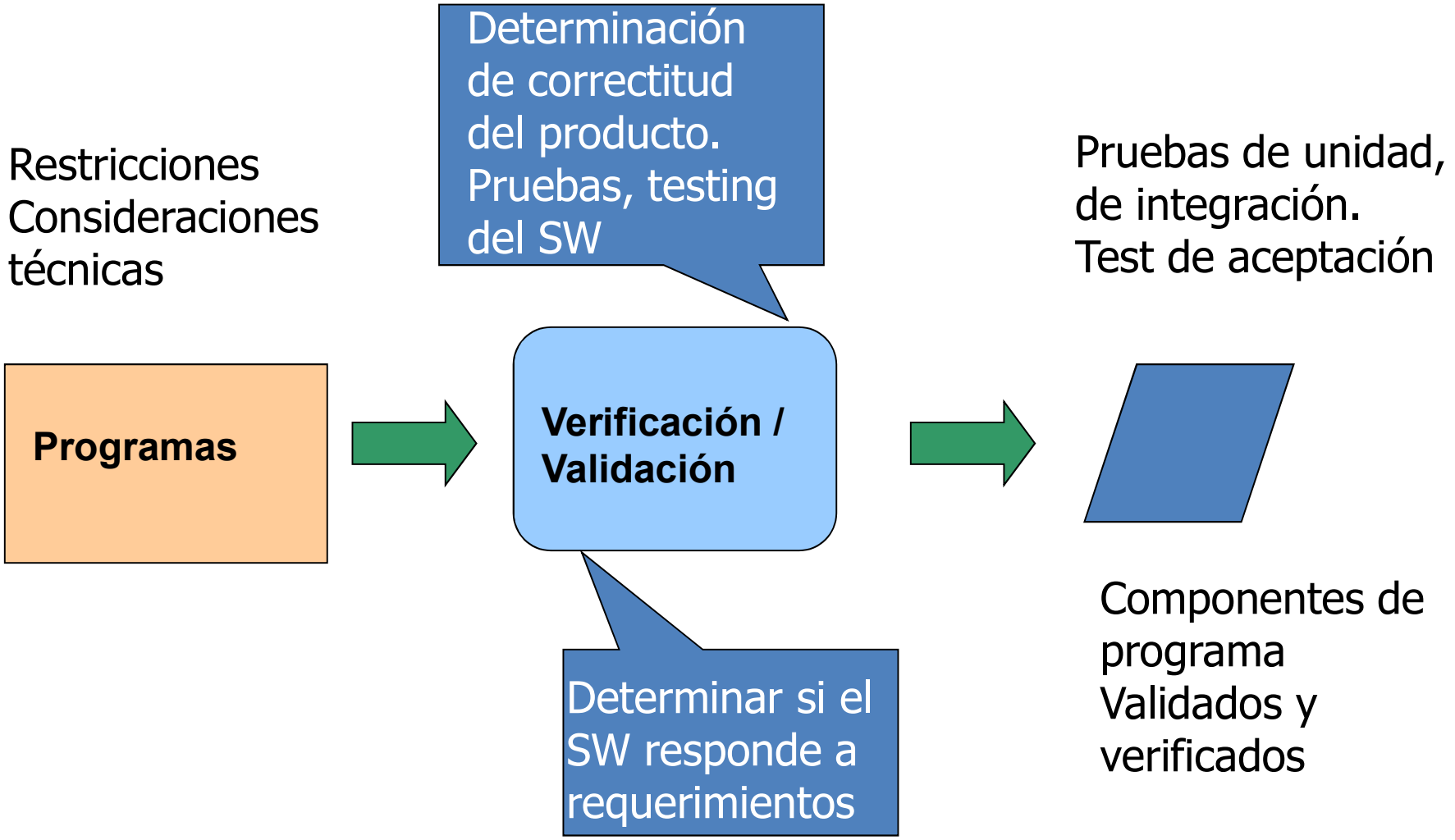
Implementación de programas en algún LP
Integración de componentes



Componentes de programa.
Código en un LP

Sistema Ejecutable

Actividad: Verificación y validación



Ciclo de vida del SW

- **Ciclo de vida:** Periodo de tiempo que comienza al concebir la idea de un nuevo sistema de software, y termina cuando este se retira y deja de funcionar.
- **Proceso** es un conjunto de actividades y tareas relacionadas, que al ejecutarse de forma conjunta transforman una entrada en una salida.
- Un **Modelo de proceso**, es una plantilla, patrón o marco que define el proceso a través del cual se crea software

Se utilizan ambos términos (proceso y ciclo de vida) para definir los modelos disponibles.

Ciclo de vida del SW

Sirven para:

- Definir las actividades a llevarse a cabo en un proyecto de desarrollo de SW en distintas fases temporales.

- Lograr congruencia entre los distintos proyectos de desarrollo de SW en una misma organización.

- Proporcionar puntos de control y revisión de las decisiones sobre continuar con el proyecto o no.

Modelos de proceso de SW

- Definen la estructura de un proceso de desarrollo racional y controlable
- No existe un modelo universal
- Los modelos no son rígidos
- Son una guía respecto al orden en que deben realizarse las actividades
- Se basan en el reconocimiento que el software tiene un ciclo de vida

- ✓ **Modelos secuenciales**: Lineal, Cascada, RAD
- ✓ **Modelos evolutivos**: Incremental, Espiral, basado en reuso, Prototipos
- ✓ **Modelos formales**: Transformacional

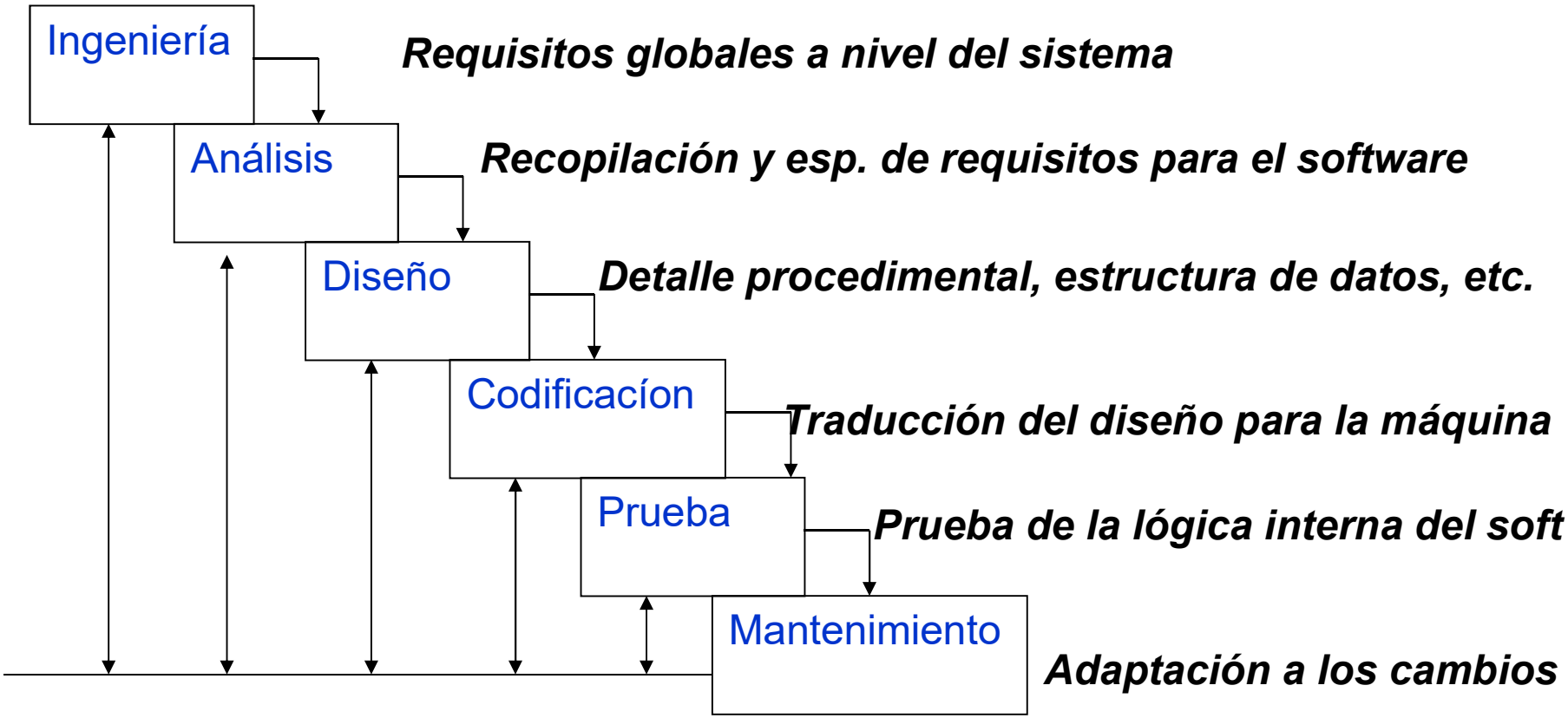
MPSW: Modelo en Cascada

- El primer modelo publicado sobre el PDSW se derivó a partir de procesos más generales de ingeniería de sistemas (Proceso Lineal secuencia delRoyce, 1970).

- El modelo en cascada es un ejemplo de un proceso dirigido por un plan; se deben planear y programar todas las actividades del proceso, antes de comenzar a trabajar con ellas

MPSW: Modelo en Cascada

Sigue un enfoque ascendente y secuencial y una insistencia en la progresión lineal del desarrollo del software que progresa a través de las siguientes etapas:



Dificultades del Modelo en cascada

Dificultades del enfoque ascendente:

✓ No hay nada para mostrar al usuario hasta que todo esté terminado.

✓ Las fallas más triviales se encuentran al comienzo del período de prueba y las más graves al final.

✓ La eliminación de fallas suele ser extremadamente difícil durante las últimas etapas de prueba del sistema.

✓ La necesidad de prueba con la computadora aumenta exponencialmente durante las etapas finales de prueba

Dificultades del Modelo en cascada

Dificultades del enfoque ascendente:

✓ El deseo de progreso ordenado de las etapas no es nada realista.

✓Durante el tiempo que dura el proyecto, el usuario podría cambiar de parecer respecto a lo que debe hacer el sistema.

✓Se apoya en técnicas anticuadas.

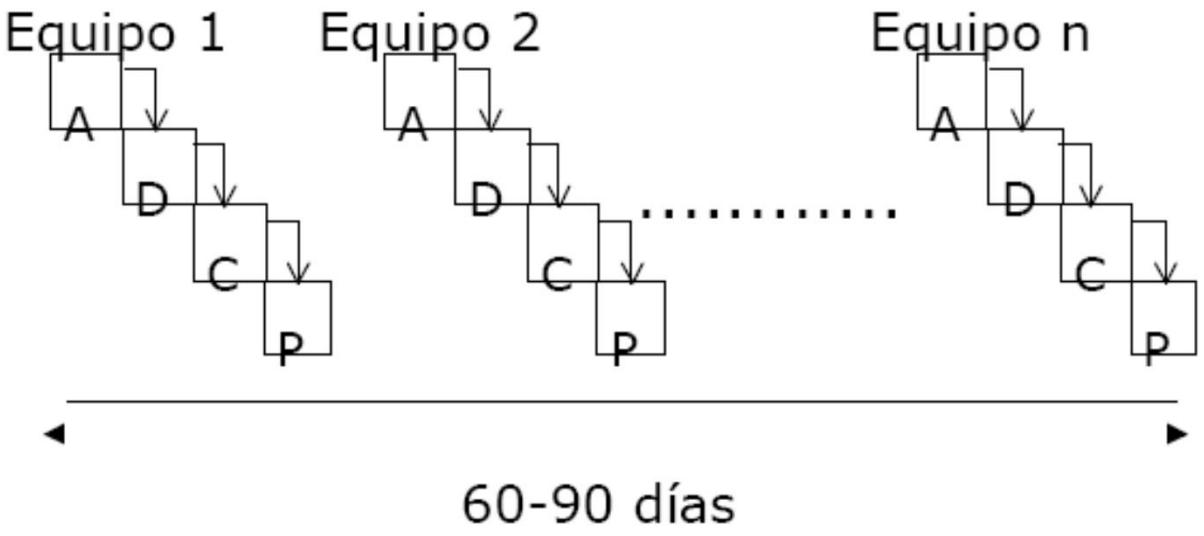
✓Resulta apropiado para sistemas pequeños

Modelo RAD (Desarrollo rápido de aplicaciones)

- Basado en el modelo lineal secuencial, con ciclos de desarrollo extremadamente cortos (90 días).
- Llevado a cabo por varios equipos de trabajo que siguen las etapas del proceso de manera simultánea. Cada equipo maneja una parte del sistema.
- Candidatos: sistemas que se pueden modularizar => equipos de desarrollo paralelos.
- Amplia participación del usuario en todas las etapas.
- Enfoque de construcción basado en el uso de componentes reutilizables y Herramientas de cuarta generación (generadores de código, herramientas de prueba automatizadas, etc.).

Modelo RAD

- Adaptación a “alta velocidad” del modelo lineal secuencial.
- Equipos trabajando en paralelo aplicando tecnología de componentes.



Modelo RAD. Restricciones

- Para proyectos grandes, es necesario contar con recursos suficientes para formar los equipos necesarios.
- Compromiso de colaboración entre desarrolladores y clientes para un tiempo de espera corto.
- No todas las aplicaciones son susceptibles de aplicar este modelo (no modularizables, bajo reuso de componentes).

- *No aplicable cuando*:
  - ✓ Sistemas con gran mantenimiento.
  - ✓ El grado de interoperatividad con programas ya existentes es alto

Modelos evolutivos

- Características:
  - ✓ Gestionan bien la naturaleza evolutiva del software
  - ✓ Se adaptan más fácilmente a los cambios introducidos a lo largo del desarrollo.
  - ✓ Son iterativos: construyen versiones de software cada vez más completas

- Se adaptan bien:
  - ✓ Los cambios de requisitos del producto
  - ✓ Fechas de entrega estrictas poco realistas
  - ✓ Especificaciones parciales del producto

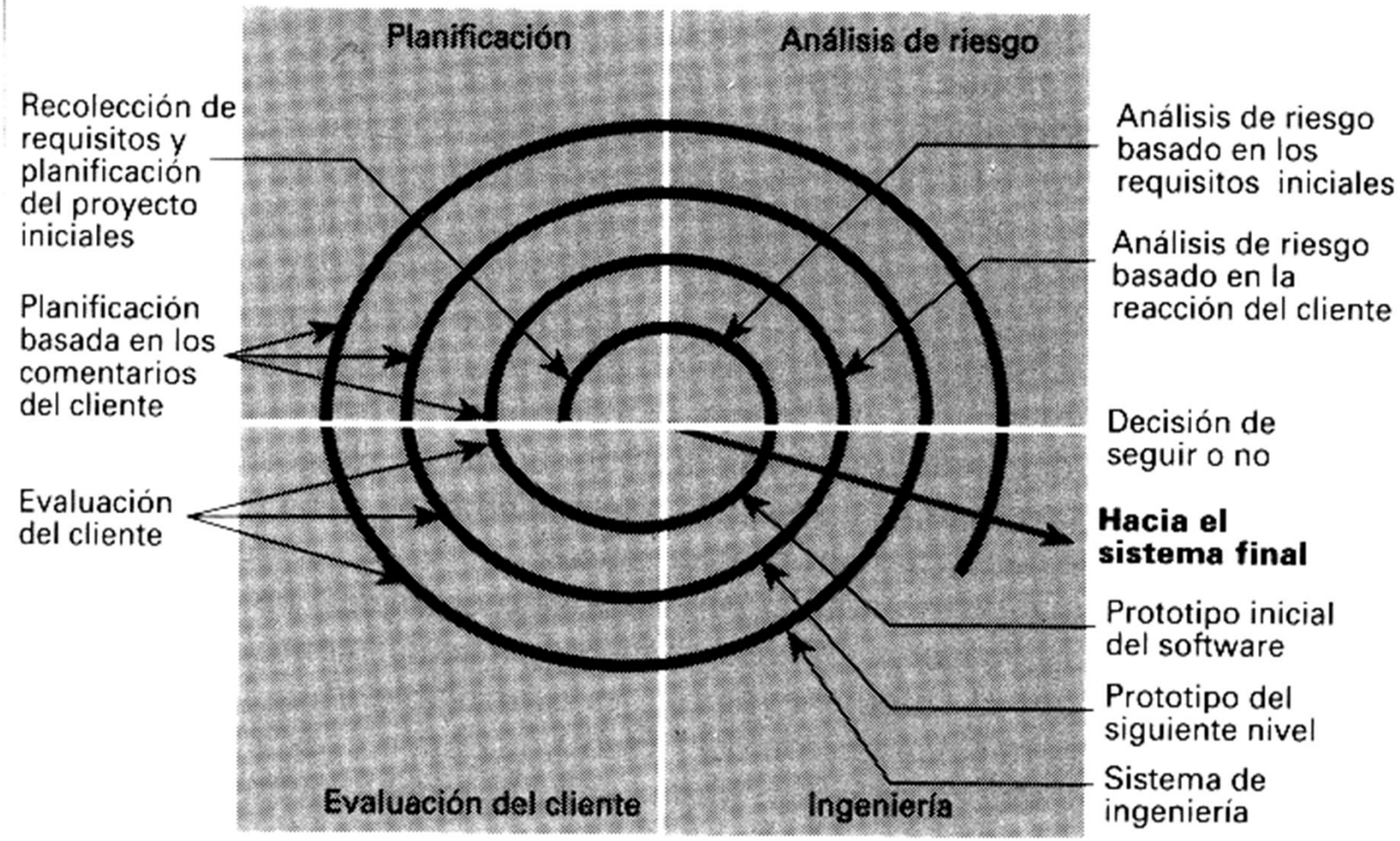
- Modelos evolutivos:
  - ✓ Modelo en Espiral
  - ✓ Modelo Incremental
  - ✓ Modelo de Desarrollo Basado en reuso de Componentes

Modelo en espiral

Actividades principales

1) **Planificación:** determinación de objetivos, alternativas y restricciones. Incluye:
   - ✓ Recolección de requisitos y planificación del proyecto iniciales (1a vuelta).
   - ✓ Planificación basada en los comentarios del cliente (vueltas siguientes).
2) **Análisis de riesgos:** análisis de alternativas e identificación /resolución de riesgos. Incluye:
   - ✓ Análisis de riesgos basado en los requisitos iniciales (1a. vuelta).
   - ✓ Análisis de riesgos basados en la reacción del cliente (2a. vuelta).
   - ✓ Se toma la decisión de seguir adelante o no.
3) **Ingeniería:** Desarrollo del producto de "siguiente nivel".
   - ✓ Se construye un prototipo cero (inicial del software) y se va refinando en los niveles sucesivos.
4) **Evaluación del cliente:** valoración de los resultados de la ingeniería en cada etapa.

Esquema del Modelo en espiral



Con cada iteración alrededor de una espiral (comenzando en el centro) se construyen sucesivas versiones del SW

Esquema del Modelo en espiral

Durante la primera vuelta:

- Se definen los objetivos, las alternativas y las restricciones,
- Se analizan e identifican los riesgos,
- El cliente evalúa el trabajo de ingeniería y sugiere modificaciones.

En base a los comentarios del clientes se produce la siguiente fase

- Planificación y análisis de riesgos. (en cada bucle de la espiral este análisis se traduce en una decisión de “seguir adelante o no”).
- En la mayoría de los casos, se sigue avanzando alrededor del espiral, con el prototipo del siguiente nivel (ingeniería) y la evaluación del cliente; ese camino lleva a un modelo completo del sistema.

Modelos evolutivos: Incremental

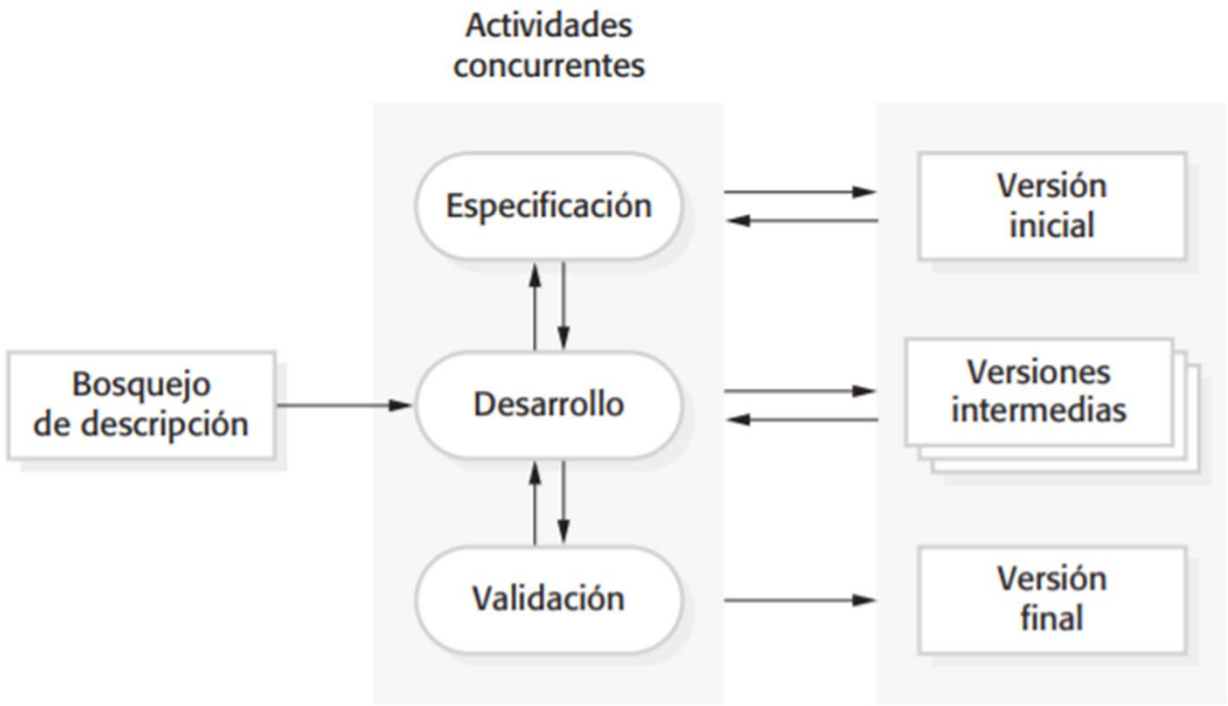
- El desarrollo incremental se basa en la idea de diseñar una implementación inicial, exponer ésta al comentario del usuario, y luego desarrollarla en sus diversas versiones hasta producir un sistema adecuado.

- El desarrollo de software incremental, que es una parte fundamental de los enfoques ágiles, es mejor que un enfoque en cascada para la mayoría de los sistemas empresariales, de comercio electrónico y personales.

- El desarrollo incremental refleja la forma en que se resuelven problemas. Al desarrollar el software de manera incremental, resulta más barato y fácil realizar cambios en el software conforme éste se diseña.

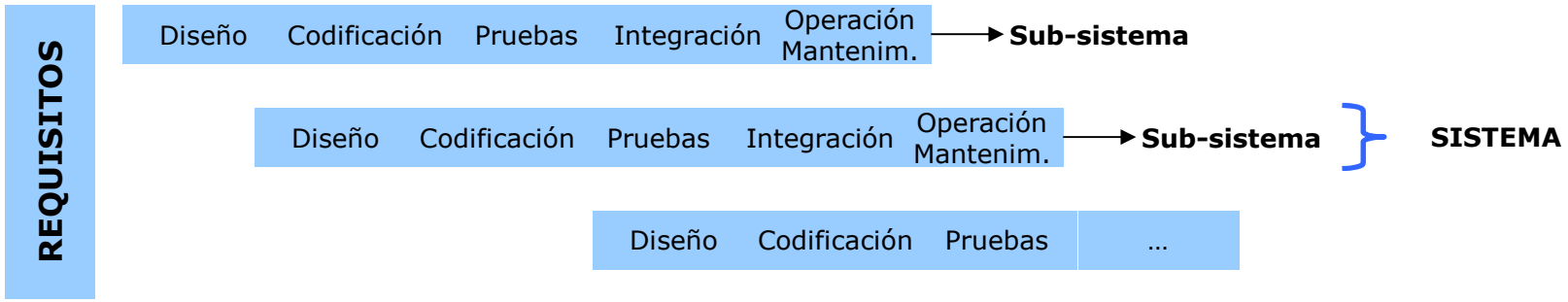
Modelos evolutivos: Incremental

- Las actividades de especificación, desarrollo y validación están entrelazadas en vez de separadas, con rápida retroalimentación a través de las actividades.



Modelos evolutivos: incremental

Incremental



El modelo incremental mitiga la rigidez del modelo en cascada, descomponiendo el desarrollo de un sistema en partes; para cada una de las cuales se aplica un ciclo de desarrollo.

Las ventajas que ofrece son:

- ✓ El usuario dispone de pequeños subsistemas operativos que ayudan a perfilar mejor las necesidades reales del conjunto.
- ✓ El modelo produce entregas parciales en periodos cortos de tiempo y permite la incorporación de nuevos requisitos que pueden no estar disponibles o no ser conocidos al iniciar el desarrollo.

Modelos evolutivos: Incremental

- Es un modelo cuyas etapas consisten de *incrementos expandidos de un producto de software operativo*.

- Cada iteración devuelve un "**Incremento**" del SW
- Un incremento es un **producto operacional del SW.**
- El primer incremento suele ser un núcleo básico desarrollado en base a los *requerimientos centrales*.

- Muchas funciones suplementarias se dejan para después
- Se evalúa (p.ej., por el cliente) el producto entregado
- Como resultado se desarrolla un plan para el incremento siguiente
- Se itera hasta elaborar el producto completo

Modelos evolutivos: Incremental

Ventajas

- Es interactivo
  Con cada incremento se entrega al cliente un producto operacional al cliente, que puede evaluarlo
- Personal
  Permite variar el personal asignado a cada iteración
- Gestión riesgos técnicos
  Por ejemplo, disponibilidad de hardware específico

Inconvenientes

La primera iteración puede plantear los mismos problemas que en un modelo lineal secuencial

Modelos evolutivos: Incremental

- El desarrollo incremental ahora es en cierta forma el enfoque más común para el desarrollo de sistemas de aplicación.

- Este enfoque puede estar basado en un plan, ser ágil o, más usualmente, una mezcla de dichos enfoques.
  - ✓En un enfoque basado en un plan se identifican por adelantado los incrementos del sistema;
  - ✓Si se adopta un enfoque ágil, se detectan los primeros incrementos, aunque el desarrollo de incrementos posteriores depende del avance y las prioridades del cliente.

Modelos basados en reuso: Ensamblaje de componentes

- Los enfoques orientados a la reutilización se apoyan en una gran base de componentes de software reutilizable y en la integración de marcos para la composición de dichos componentes.
- En ocasiones, tales componentes son sistemas por derecho propio (sistemas comerciales, off-the-shelf o COTS) que pueden mejorar la funcionalidad específica, como el procesador de textos o la hoja de cálculo
- La ingeniería de software orientada a la reutilización tiene la clara ventaja de reducir la cantidad de software a desarrollar y, por lo tanto, la de disminuir costos y riesgos; por lo general, también conduce a entregas más rápidas del software

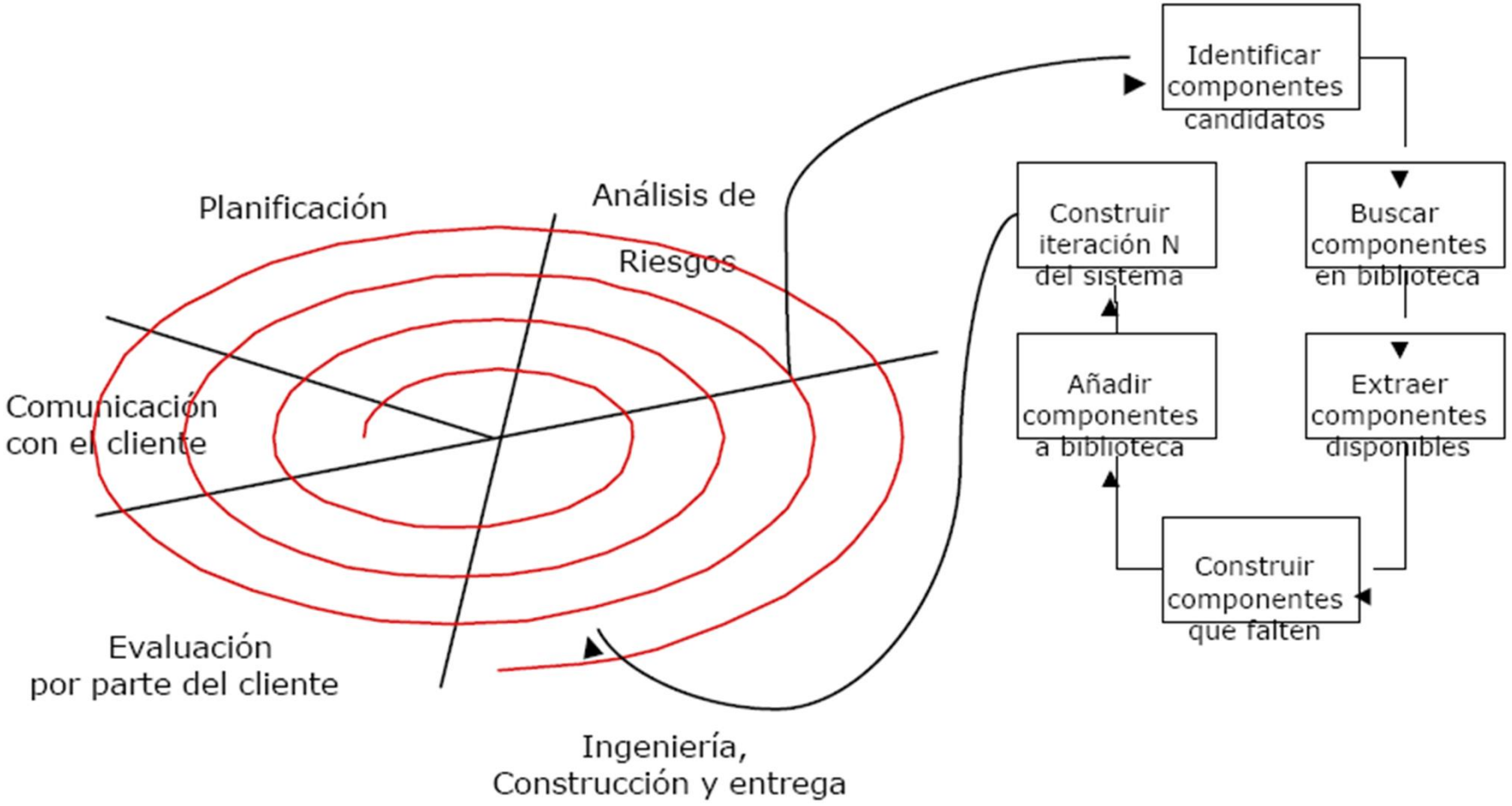
Etapas de un proceso de DBC

- Análisis de componentes: A partir de la especificación de requerimientos, se realiza una búsqueda de componentes para implementarla.
- Modificación de requerimientos Durante esta etapa se analizan los requerimientos usando información de los componentes descubiertos. Luego se modifican para reflejar los componentes disponibles
- Diseño de sistema con reutilización Durante esta fase se diseña el marco conceptual del sistema o se reutiliza un marco conceptual existente.
  - Es posible que deba diseñarse algo de software nuevo, si no están disponibles los componentes reutilizables.

Etapas de un proceso DBC

- Desarrollo e integración: Se diseña el software que no puede procurarse de manera externa, y se integran los componentes y los sistemas COTS para crear el nuevo sistema.
- La integración del sistema, en este modelo, puede ser parte del proceso de desarrollo, en vez de una actividad independiente.

Ensamblaje de componentes en un modelo evolutivo



Modelo de prototipos

- ***Facilita al programador la creación de un modelo de SW.***
- Puede considerarse un modelo en sí mismo o un aporte a los distintos modelos de procesos de desarrollo
- El prototipado consiste en la construcción de modelos de prueba, que simulen el funcionamiento que se pretende conseguir en el sistema.
- Esta forma de trabajo previo suele tener como principal objetivo la **experimentación con un entorno similar al pretendido**, para obtener retro-información del usuario o cliente que ayuda a los desarrolladores en la concreción de los requisitos

Modelo de prototipos

- Los prototipos pueden ser:

  - **Ligeros:** dibujos de pantallas de interfaz con simulación de funcionamiento por enlaces a otros dibujos…

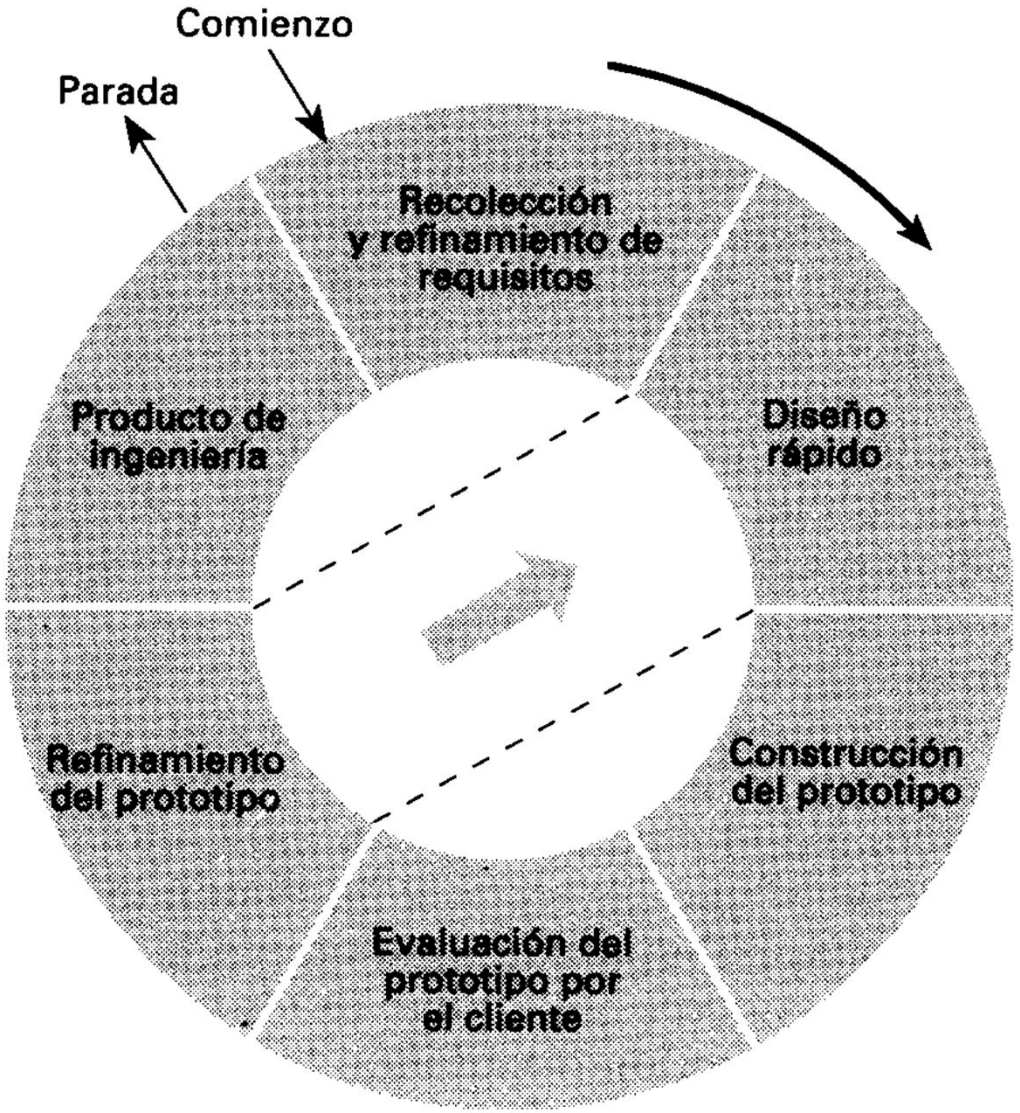
  - **Operativos:** Módulos de software con funcionamiento propio que se desarrollan sin cubrir las funcionalidades completas del sistema, normalmente en entornos RAD (rapid application development”).

Modelo de prototipos

- La construcción de prototipos comienza con la recolección de requisitos.
  - ✓ El analista y el cliente se reúnen, definen los objetivos globales para el software, identifican los requisitos.

- Luego se produce un "diseño rápido" que conduce a un prototipo.
- El prototipo es evaluado por el cliente / usuario y se utiliza para refinar los requisitos del software a desarrollar.
- El prototipo es refinado sucesivamente en un proceso interactivo para que satisfaga las necesidades del cliente.
- Generalmente este primer prototipo llamado "prototipo cero" se tira, porque es lento o grande o difícil de usar

Modelo de prototipos



Modelo de prototipos

Desventajas

- ✓ El cliente ve *el prototipo y lo confunde con el sistema real*. No entiende la necesidad de volver a hacerlo.

- ✓ El equipo de desarrollo toma *decisiones rápidas* para poder construir el prototipo, que más tarde *son difíciles de revertir* (por ejemplo el lenguaje de programación).

Modelo Transformacional

- Basado en *especificaciones formales*: el desarrollo de software como una secuencia de pasos que gradualmente *transforma* una especificación en implementación hasta un programa ejecutable
- Se deben transformar requerimientos informales en especificaciones formales*: métodos de especificación formal.*
- Pretende reducir errores automatizando pasos del desarrollo. Las transformaciones preservan la corrección.

Modelo Transformacional

Evolución de la Ingeniería de SW

- Problemas
  - ✓ Hace falta una formación especializada para aplicar la técnica
  - ✓ Muchos aspectos de los sistemas reales son difíciles de especificar formalmente

- Aplicabilidad
  - Sistemas críticos en los que la seguridad y fiabilidad debe poder asegurarse antes de poner el sistema en operación

Bibliografía

- ✓ "Fundamentals of Software Engineering". Carlo Guezzi.
- ✓ "Ingeniería del Software" – Un enfoque Práctico" R.Pressman.
- ✓ "Ingeniería de Software" . Ian Sommerville.