Proceso Software
Tema 2. Ciclos de Vida

Ciclo de Vida

- *"Una aproximación lógica a la adquisición, el suministro, el desarrollo, la explotación y el mantenimiento del software"* **IEEE 1074**

- *"Un marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de software, abarcando la vida del sistema desde la definición de los requisitos hasta la finalización de su uso"***ISO 12207-1**

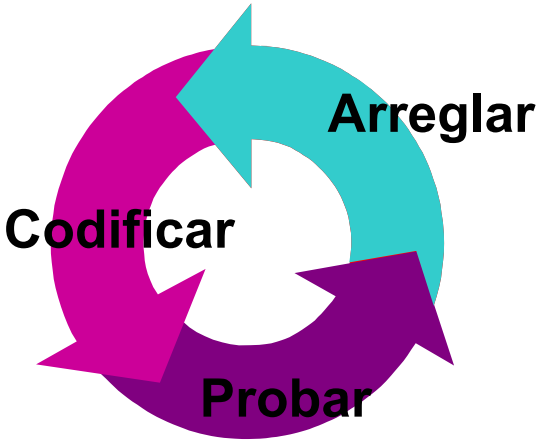
Ciclo de Vida

- Un ciclo de vida debe:
  - Determinar el **orden de las fases** del Proceso Software
  - Establecer los **criterios de transición** para pasar de una fase a la siguiente.

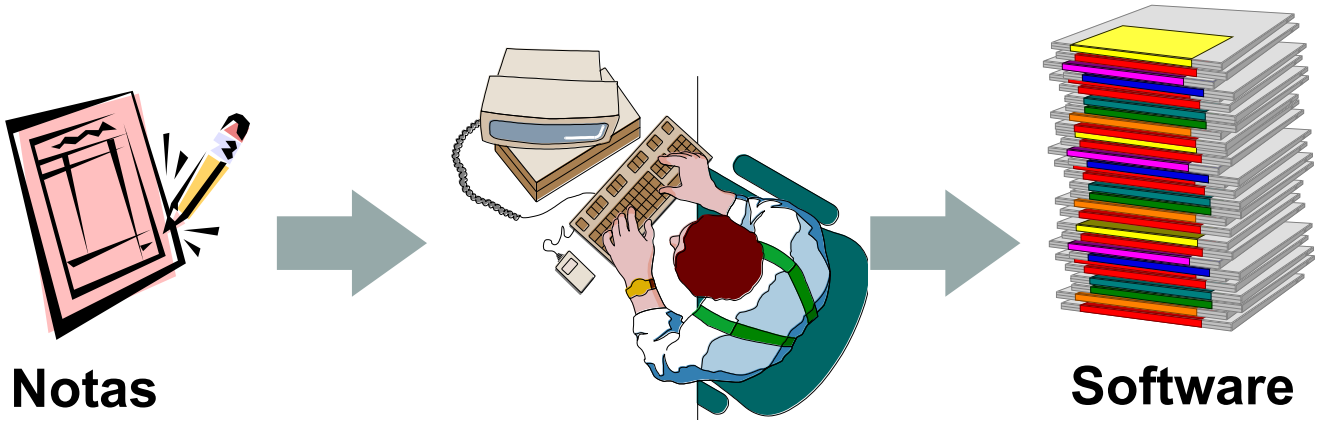
Ciclo de Vida

- Ciclo de vida ≠ Ciclo de vida de desarrollo
  - Ciclo de vida: Toda la vida del sistema, desde la concepción hasta el fin de uso.
  - Ciclo de vida de desarrollo: Desde el análisis hasta la entrega al usuario.

Codificación Directa



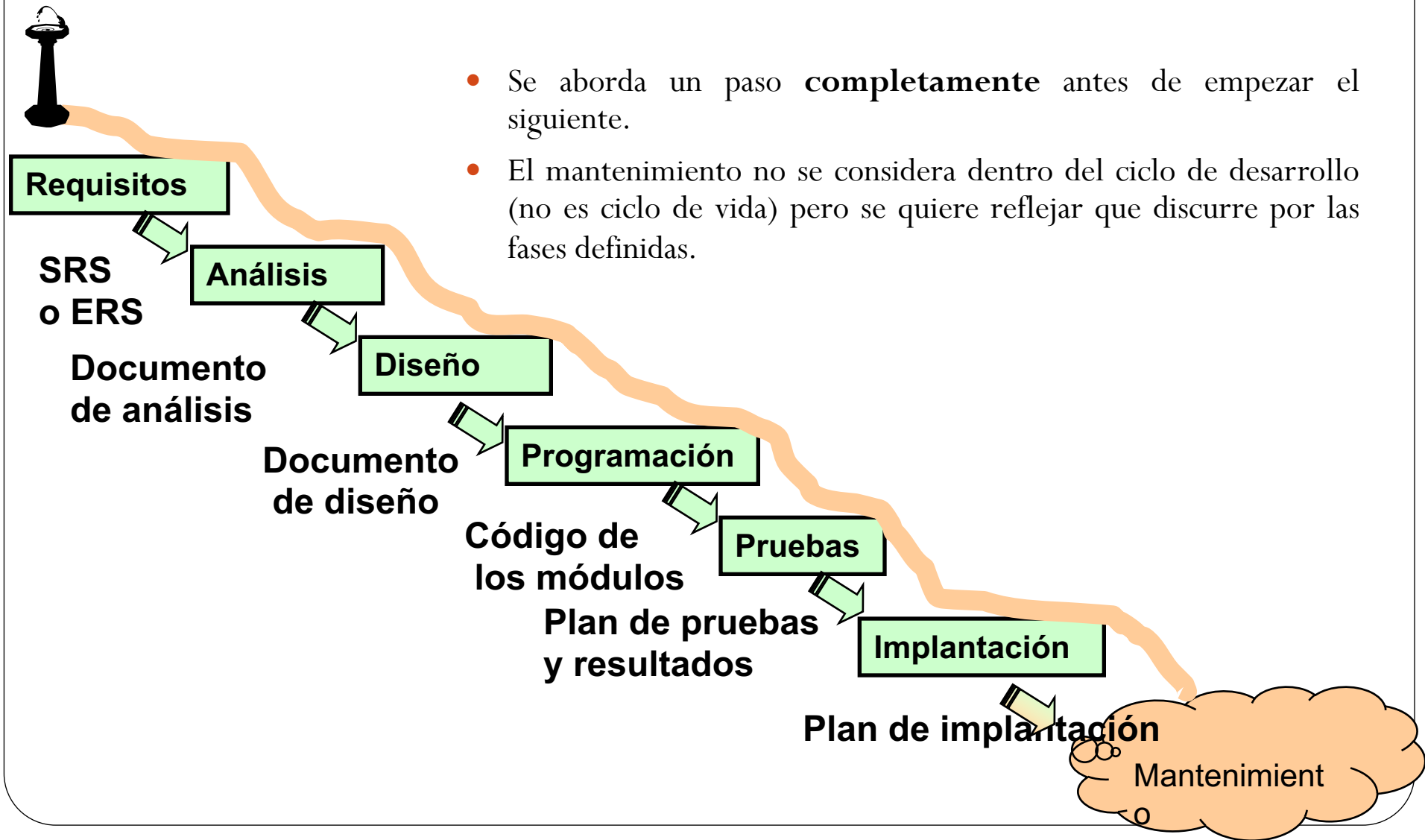
- También llamado **Code and Fix**
- Sinónimo de **arte** en el Desarrollo de Software
- Antónimo de ingeniería en el Desarrollo de Software
- **No es recomendable** para proyectos un poco más grande que "enanos"



Cascada

- También conocido como **Waterfall**.
- Nace en los años 70 como **alternativa al Code and Fix**, que provocó la crisis del software.
- Es el que sigue un gran número de metodologías y es **el más conocido, estudiado, difundido y empleado**.
- Pone especial énfasis en la **realización temprana de actividades** de definición de requisitos y de documentación de análisis y diseño como paso previo a la codificación.
- **Secuencia lineal** de las siguientes actividades generales: Analizar, Diseñar, Codificar, Probar e Implantar.
- Cada fase genera productos de salida que servirán a la siguiente fase para desarrollar la actividad de la misma como productos de entrada.

Cascada



- Se aborda un paso **completamente** antes de empezar el siguiente.
- El mantenimiento no se considera dentro del ciclo de desarrollo (no es ciclo de vida) pero se quiere reflejar que discurre por las fases definidas.

Cascada

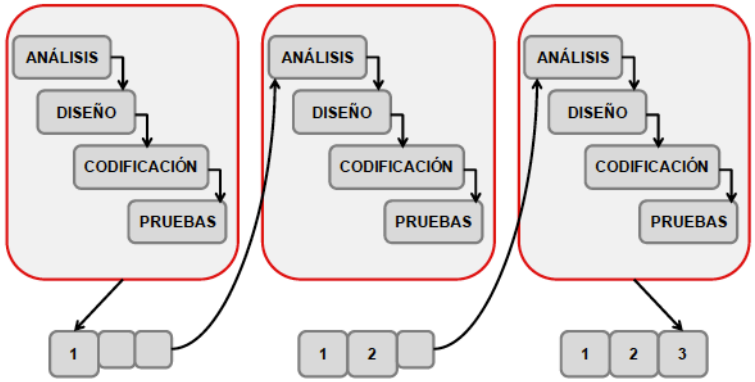
- Ventajas:
  - Conforma un **marco de referencia** para asignar todas las actividades de desarrollo software.
  - Es un método muy estructurado que establece pautas de trabajo muy claras.
  - Facilita mucho la **coordinación** de los recursos implicados.
  - Facilita la **disposición de hitos de seguimiento/control** en el desarrollo de proyectos.
  - Facilita la **estimación y el seguimiento/control** del progreso de las actividades.
  - Facilita la **detección de desviaciones** y la realización de acciones correctivas.
  - Proporciona productos entregables intermedios que conforman el producto final.

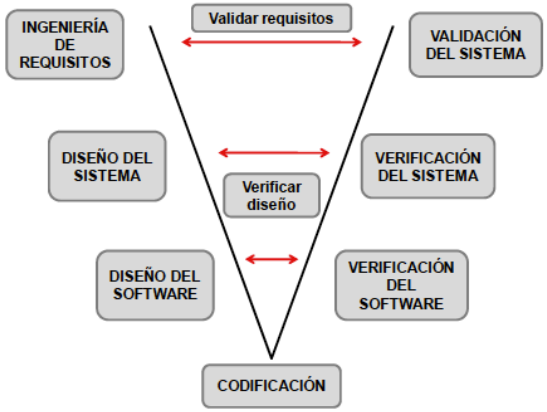
Cascada

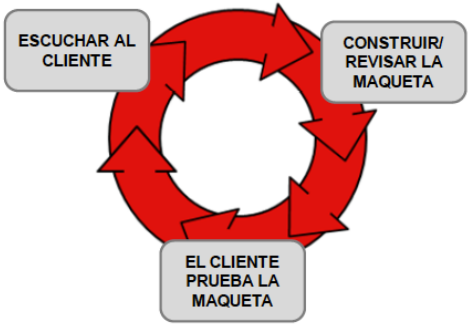
- Inconvenientes:
  - Comienza estableciendo **todos** los requisitos del sistema a desarrollar, aunque muchas veces esto no es viable.
  - **Gran rigidez**: cada actividad es prerrequisito de las que le siguen.
  - Los posibles **problemas se detectan tarde** aunque se incluyan actividades de verificación y validación.
  - Los únicos productos parciales aprovechables están en forma de documentos: **nada está hecho hasta que todo está hecho**.

Otros modelos de Ciclo de Vida

- Modelo Incremental.
- Modelo en V.
- Modelo en Espiral.
- Modelo de Prototipos.

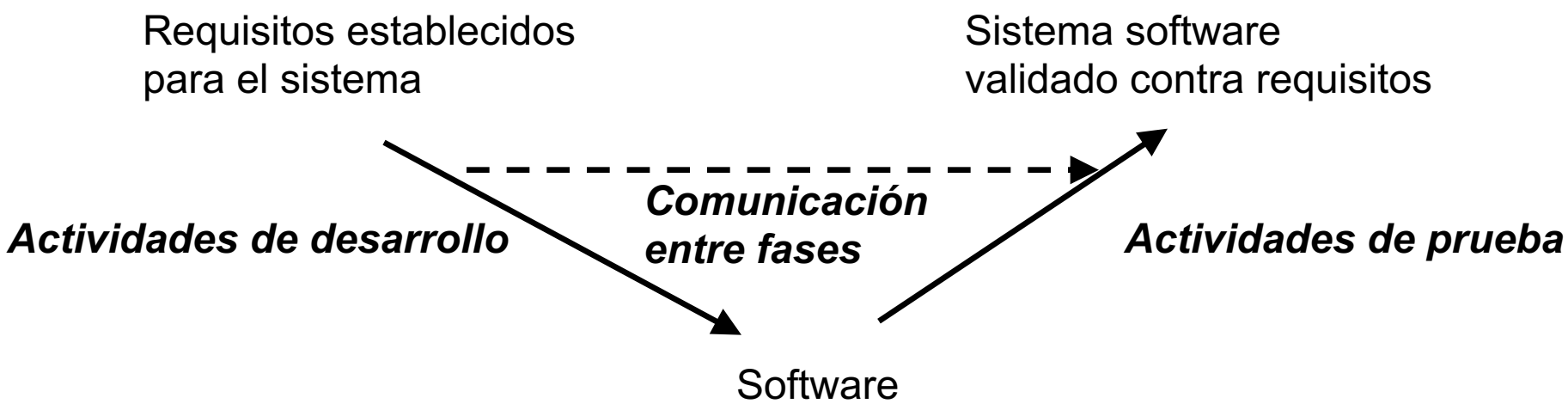






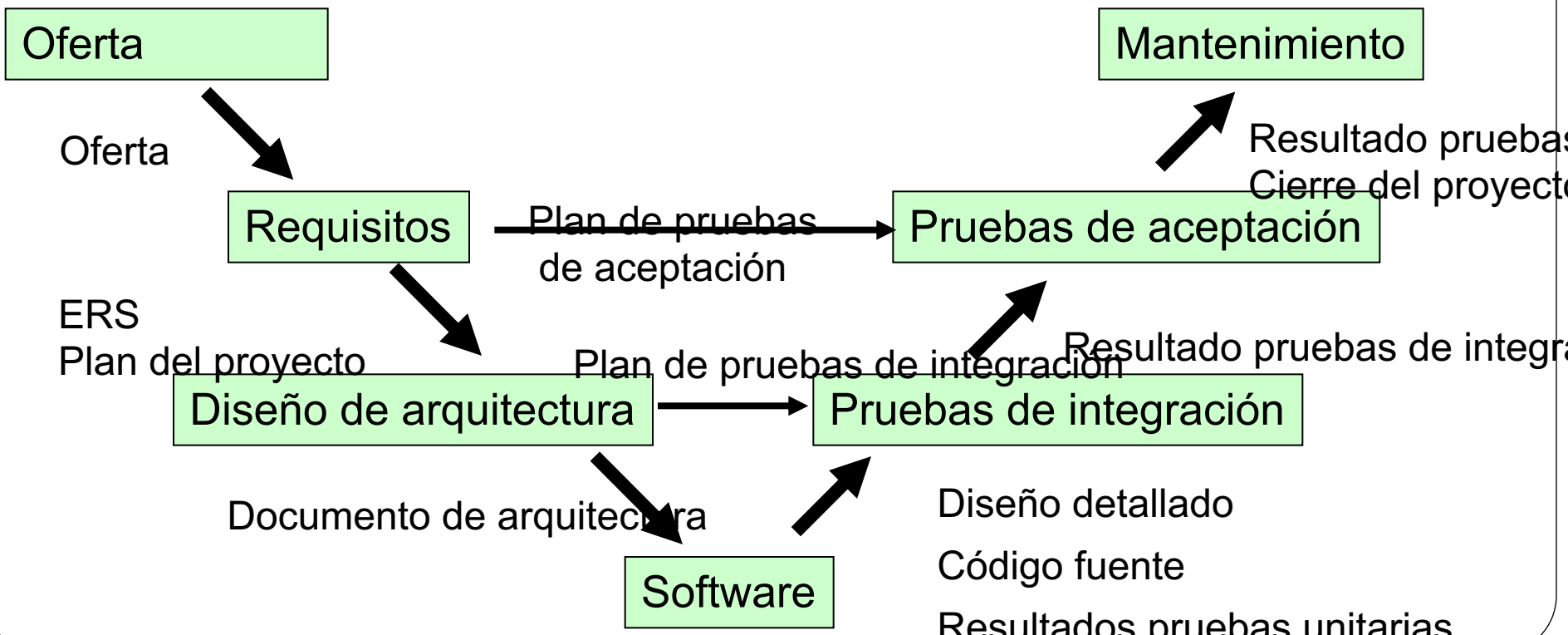
Ciclo de vida en V

- Es un modelo similar al de Cascada, aunque "mejorado".
- Tiene dos tiempos (tipos de actividades) claramente marcados:
  - Rama descendente: actividades de desarrollo.
  - Rama ascendente: actividades de prueba.
  - Ambas ramas se comunican en materia de pruebas:
    - Cada fase de la rama descendente tiene su homóloga en la rama ascendente para las actividades de prueba correspondientes.
- El esquema básico es el siguiente:



Ciclo de vida en V

- Un ejemplo muy simplificado podría ser el siguiente:



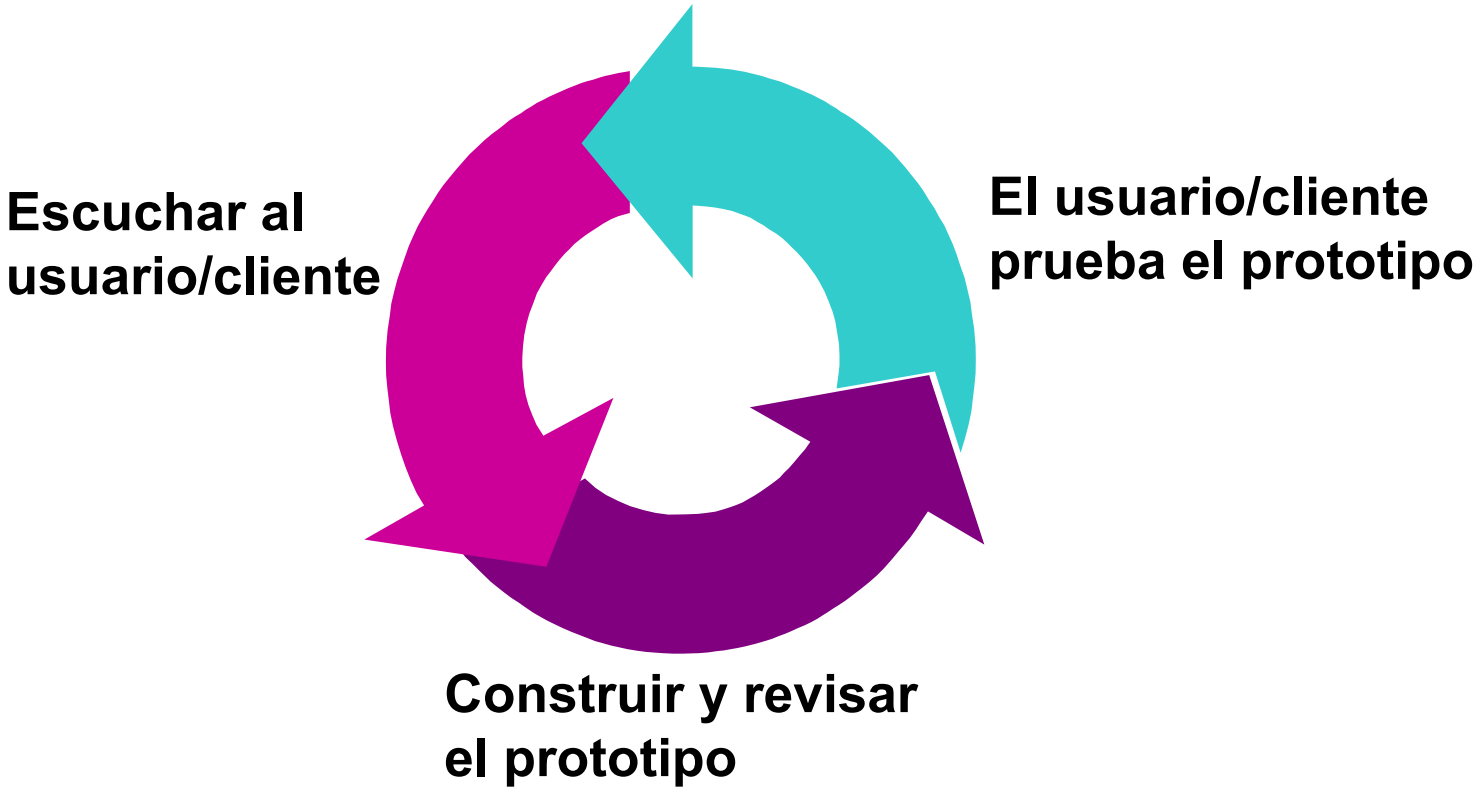
Ciclo de vida en V

- Ventajas:
  - Al ser una variante del de Cascada, hereda todas sus ventajas.
  - Conforma un marco de referencia para asignar todas las actividades de desarrollo software, incluyendo las actividades V&V de lo que se hace en las sucesivas etapas.
  - Favorece la consideración de las pruebas lo antes posible (comunicación horizontal entre las dos ramas de la V).

- Inconvenientes:
  - Al igual que en Cascada, se comienza estableciendo **todos** los requisitos del sistema a desarrollar:
    - Muchas veces esto no es viable:
      - Puede ser difícil para el propio usuario.
      - Hay cambios de parecer de los usuarios, habiendo finalizado ya la fase de requisitos.
  - Posee una gran rigidez, aunque menos que en el de Cascada por haber comunicación horizontal.
  - Los únicos productos (parciales) aprovechables están en forma de documentos:
    - "Nada está hecho hasta que todo está hecho".
    - Un cambio debido a un error puede suponer un gran coste.

Prototipado

- El proceso básico del prototipado involucra las siguientes tres fases que se detallarán a continuación:



Prototipado

- Su sentido es análogo al que tiene en otras ingenierías:
  - El prototipado o construcción de un prototipo es un proceso que facilita al desarrollador la creación de un modelo del software a desarrollar.
- Tipos de prototipos:
  - Desechables: nos sirve para eliminar dudas sobre lo que realmente quiere el cliente además para desarrollar la interfaz que más le convenga al cliente.
  - Evolutivos: es un modelo parcialmente construido que puede pasar de ser prototipo a ser software, pero no tiene una buena documentación y calidad , hay que conseguir que tenga los criterios mínimos de calidad
- Es muy adecuado cuando no se sabe exactamente lo que se quiere construir. Algunas razones para no saberlo son:
  - El propio cliente/usuario no sabe exactamente lo que quiere.
  - El desarrollador no está seguro de la eficiencia de una aproximación.
  - Existen dudas en la interfaz hombre-máquina.
  - …

Prototipado

- Ventajas:
  - El prototipo es un mecanismo ideal para extraer requisitos cuando no están claros, incluso por parte del usuario.
    - "No sé lo que quiero hasta que veo lo que no quiero".

- Inconvenientes:
  - Existe una clara tendencia del usuario/cliente a creer que el trabajo ya está hecho y que en breve dispondrá de un sistema funcional.
    - En realidad, se está simplemente en la primera fase.
  - El desarrollador toma necesariamente decisiones de implementación simplemente porque le son conocidas y le permiten desarrollar rápidamente el prototipo.
    - No son decisiones adecuadas para el sistema real.
    - El problema radica en que estas decisiones a menudo se mantienen en el sistema real.

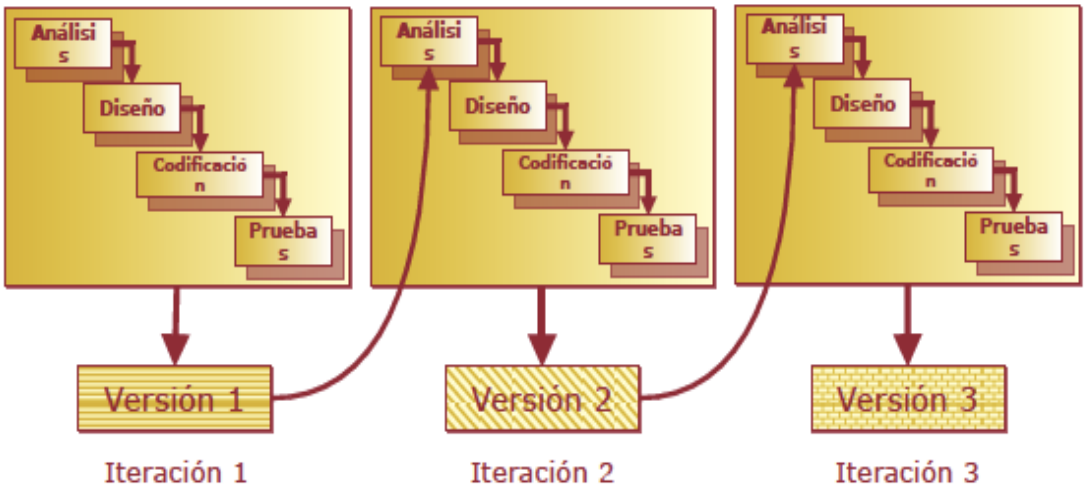
Modelos evolutivos

- El software, como todos los sistemas complejos, necesariamente evoluciona con el tiempo.
  - Los requisitos de gestión, del producto, el propio mercado, etc. a menudo hacen que sea imposible finalizar un producto completamente.
  - Esto conlleva a introducir una versión limitada para cumplir la presión competitiva y de gestión:
    - Se comprende perfectamente el conjunto de requisitos principales, pero todavía se tienen que definir los detalles de extensiones del producto.
- En éstas y en otras situaciones semejantes, se necesita un modelo de proceso que se haya diseñado explícitamente para acomodarse a un producto que evolucione con el tiempo.
- Sin embargo:
  - El enfoque en Cascada asume que al final de la secuencia marcada se entrega un sistema completo.
  - El enfoque de Prototipado desechable, ayuda a comprender los requisitos para luego acometer el desarrollo propiamente dicho.

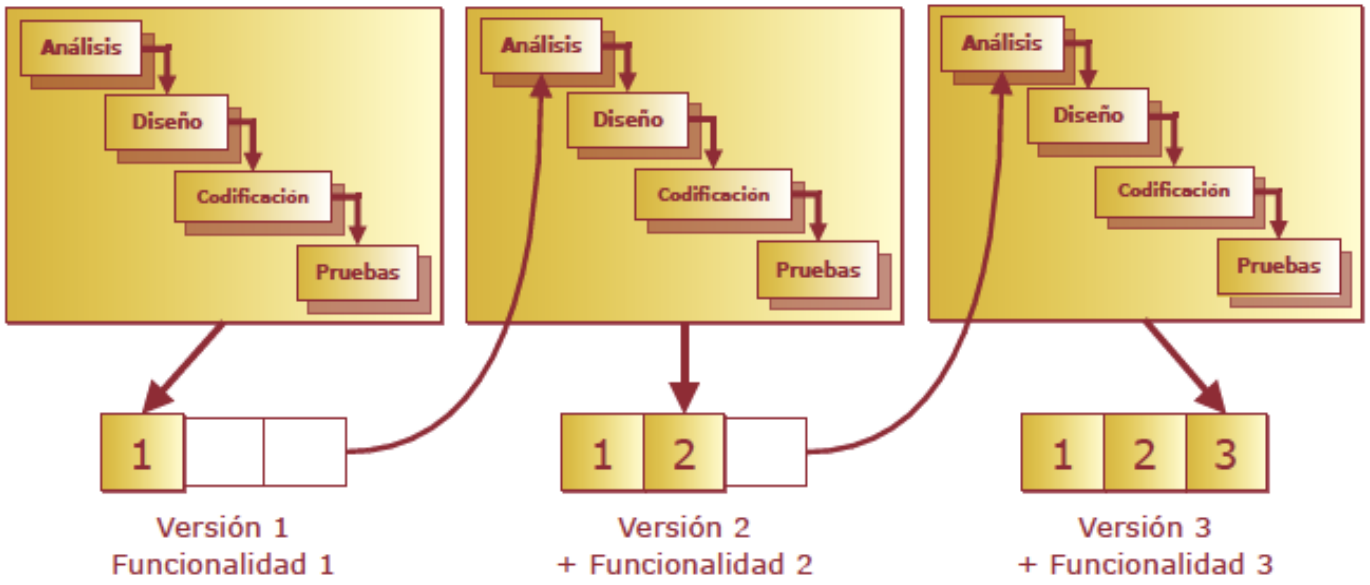
Modelos evolutivos

- En general, en la realidad, no se desarrolla para entregar un sistema listo para pasar a producción de forma completa.
- Sin embargo, los paradigmas vistos anteriormente, no tienen en cuenta la naturaleza evolutiva del software.
- Los modelos evolutivos son iterativos.
  - Se caracterizan por permitir desarrollar versiones cada vez más completas del software.
- Ejemplos de modelos evolutivos, que se verán a continuación:
  - Modelo Incremental.
  - Modelo en Espiral.

Iteración


Análisis
Diseño
Codificación
Pruebas
Versión 1
Iteración 1
Análisis
Diseño
Codificación
Pruebas
Versión 2
Iteración 2
Análisis
Diseño
Codificación
Pruebas
Versión 3
Iteración 3

Incremento

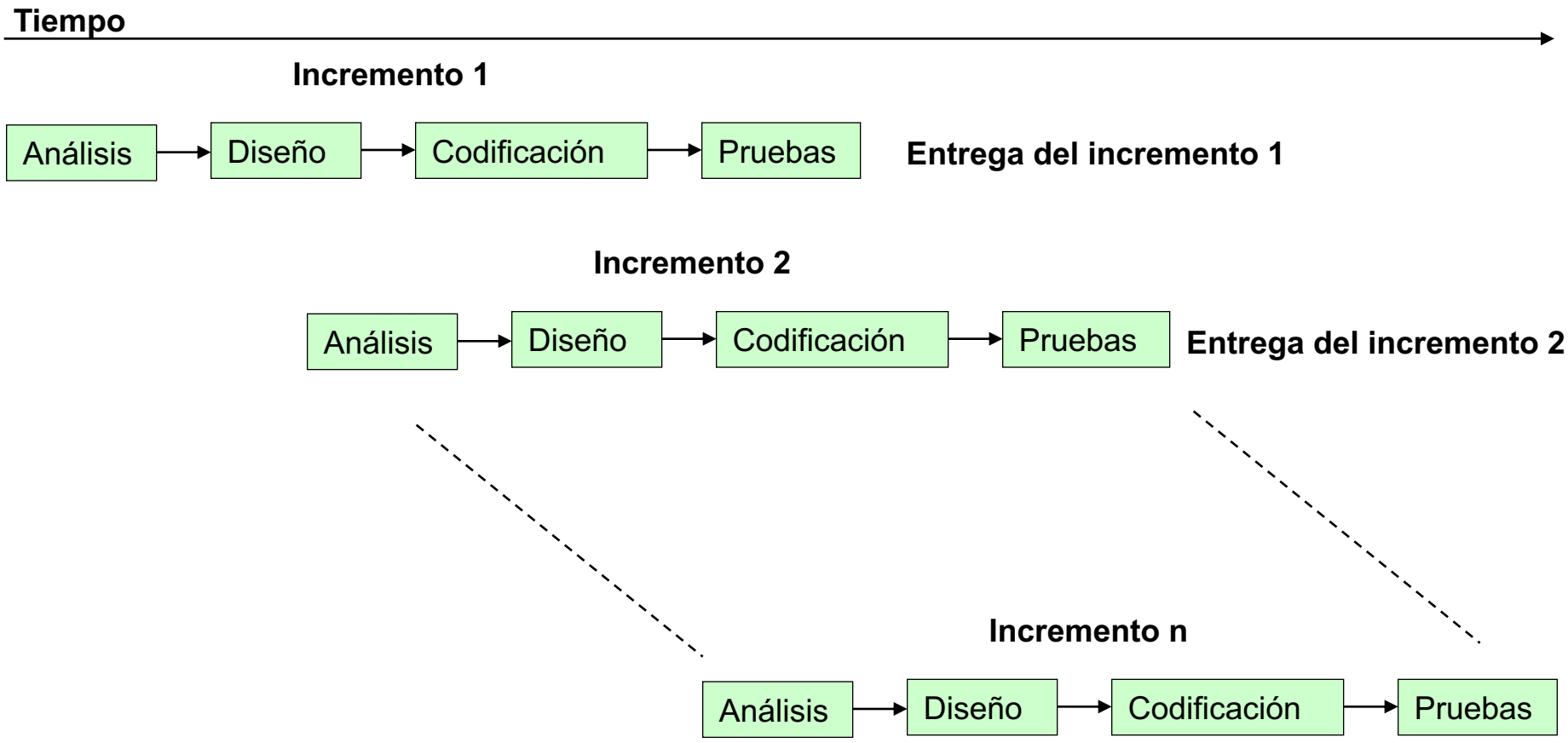

Análisis
Diseño
Codificación
Pruebas
Análisis
Diseño
Codificación
Pruebas
Análisis
Diseño
Codificación
Pruebas
1
1
2
1
2
3
Versión 1
Funcionalidad 1
Versión 2
+ Funcionalidad 2
Versión 3
+ Funcionalidad 3

Incremental

- Es el proceso de construir una implementación parcial del sistema global y posteriormente ir aumentando la funcionalidad del sistema.
- Es la aplicación reiterada de varias secuencias basadas en el modelo en Cascada.
  - Cada aplicación del ciclo constituye un incremento del software.
  - Cada incremento resulta en un producto operativo.
  - Cada incremento puede ser:
    - El refinamiento de un incremento anterior.
    - El desarrollo de una nueva funcionalidad no incorporada en incrementos anteriores.
  - En el primer incremento de un sistema se debe abordar el núcleo esencial del sistema (requisitos fundamentales).
  - En incrementos posteriores se abordarán funciones suplementarias (algunas ya conocidas y otras no).
- El usuario/cliente evalúa el resultado de un incremento y se elabora un plan para el incremento siguiente.
- El proceso se repite hasta que se elabore el producto completo.
- Al seguir un desarrollo incremental, el software debe construirse de tal forma que facilite la incorporación de nuevos requisitos.

Incremental

- Esquema básico:



Incremental

- Ventajas:
  - La dotación de personal no tiene que estar disponible para una implementación completa.
  - Permite la obtención de incrementos operativos a lo largo del proceso de desarrollo, frente a modelos tradicionales basados en el de Cascada.
    - Esto es, reduce el gasto que se tiene antes de alcanzar cierta capacidad inicial.
  - Reduce la posibilidad de que los requisitos del usuario/cliente cambien durante el desarrollo.
  - Mayor flexibilidad ante más funcionalidades.
  - La calidad y la estabilidad del software progresan con las iteraciones, y hay oportunidad para la mejora.
  - Ayuda en proyectos que utilizan tecnologías nuevas o en fase de aprendizaje por parte de la organización.

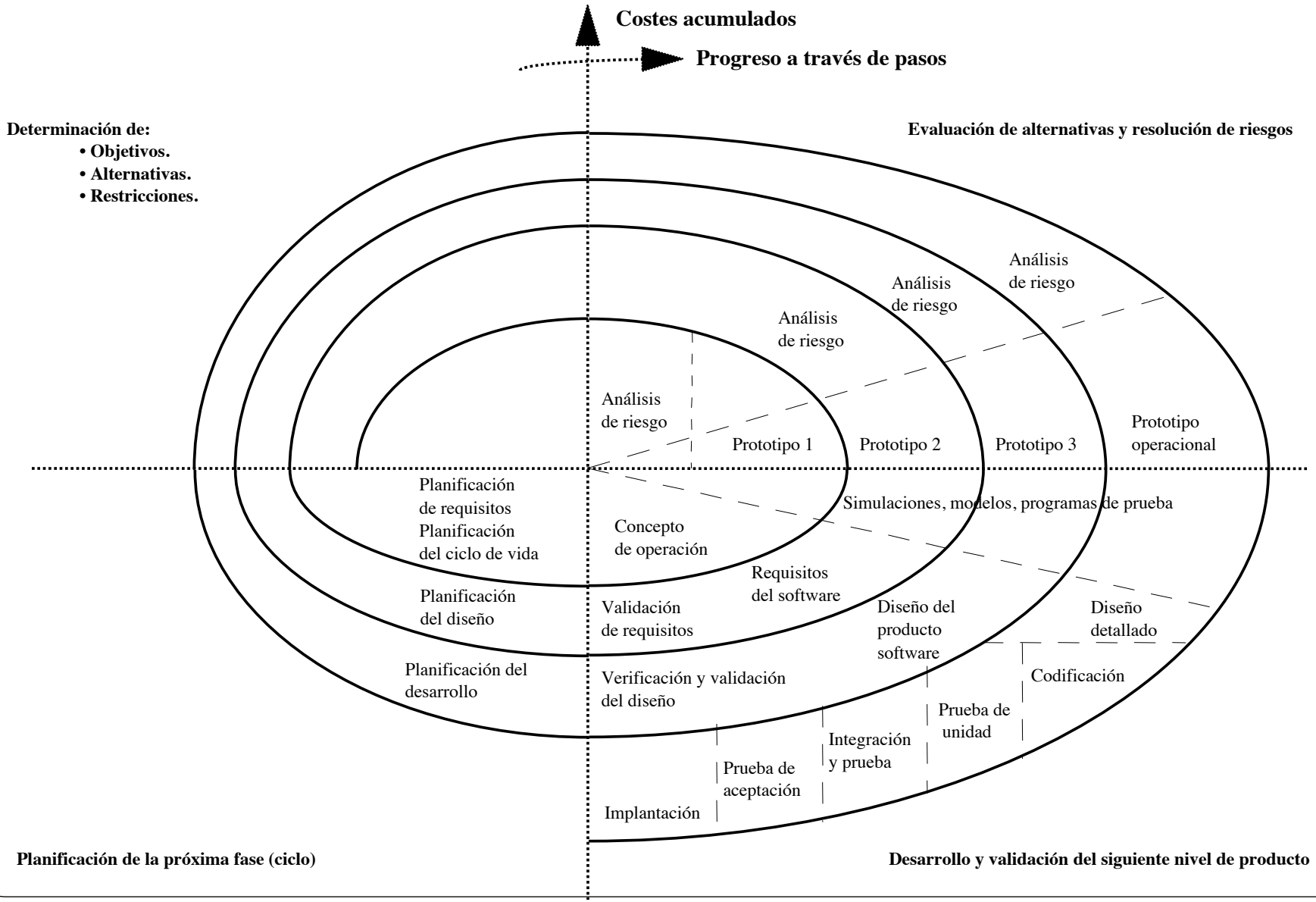
Incremental

- Inconvenientes:
  - Puede ser muy complejo definir el núcleo operativo para lograr el primer incremento:
    - Un software con capacidades suficientes como para ser operativo y que facilite la incorporación de nuevos requisitos.
  - Puede resultar complicado establecer y priorizar las funcionalidades que se mejoran o incorporan a los sucesivos incrementos.
  - Las soluciones de incrementos anteriores pueden no ser válidas para incrementos posteriores.
  - No es útil en proyectos cortos en duración o ámbito ni en los que la funcionalidad a implementar este perfectamente definida, ya que el coste extra de planificación por incremento no merece la pena

Espiral

- Fue propuesto por Barry W. Boehm (1988)
- Es un modelo de proceso evolutivo que:
  - Permite combinar la naturaleza interactiva de construcción de prototipos con los aspectos controlados y sistemáticos del modelo en Cascada.
  - Además, pone especial énfasis en el análisis de los riesgos para reducir su impacto.
- El software se desarrolla en una serie de versiones incrementales.
  - Durante las primeras iteraciones, la versión incremental será un modelo en papel o un prototipo.
  - Durante las últimas iteraciones, se producen versiones cada vez más completas del sistema.
- La idea básica es que cada fase (vuelta de la espiral) establece los siguientes pasos:
  - Determinación de objetivos, alternativas y restricciones.
  - Evaluación de alternativas (considerando análisis de riesgos).
  - Desarrollo del siguiente nivel de producto.
  - Planificación de la siguiente fase (ciclo en la espiral).

Espiral


Costes acumulados
Progreso a través de pasos
Determinación de:
• Objetivos.
• Alternativas.
• Restricciones.
Evaluación de alternativas y resolución de riesgos
Análisis de riesgo
Análisis de riesgo
Análisis de riesgo
Análisis de riesgo
Prototipo 1
Prototipo 2
Prototipo 3
Prototipo operacional
Planificación de requisitos Planificación del ciclo de vida
Concepto de operación
Simulaciones, modelos, programas de prueba
Planificación del diseño
Validación de requisitos
Requisitos del software
Diseño del producto software
Diseño detallado
Planificación del desarrollo
Verificación y validación del diseño
Codificación
Prueba de unidad
Integración y prueba
Prueba de aceptación
Implantación
Planificación de la próxima fase (ciclo)
Desarrollo y validación del siguiente nivel de producto

Espiral

- Ventajas:
  - Permite adaptar el proceso de desarrollo a las necesidades cambiantes del proyecto y al conocimiento que se va adquiriendo.
  - Permite el manejo de prototipos, enlazándolo con el análisis de riesgos.
  - Gestiona explícitamente los riesgos.
    - Permitirá reducirlos antes de que se conviertan en problemas.

- Inconvenientes:
  - Requiere de una considerable habilidad para la consideración del riesgo.

Ciclos de vida OO. Características

- Desarrollo iterativo e incremental.
- Eliminación de fronteras entre fases.
- Incorporan bibliotecas de clases y otros componentes reutilizables

Procesos del ciclo de vida del software ISO 12207