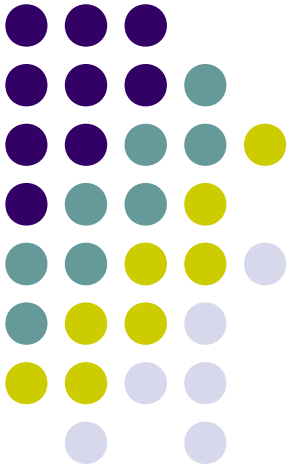
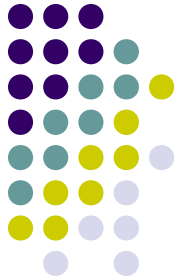
Ingeniería de software I

Tema II – La ingeniería de software

Universidad Nacional del Litoral
FACULTAD DE INGENIERÍA Y CIENCIAS HÍDRICAS



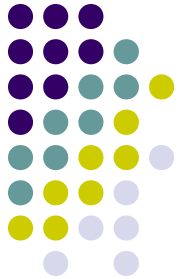
La complejidad del software



Deriva de:

- Complejidad del dominio del problema
- Dificultad de gestionar el desarrollo
- Flexibilidad del software
- Caracterización de problemas discretos

La complejidad del software

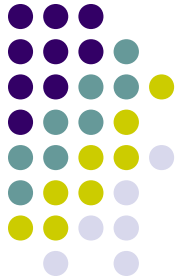


Lleva a:

- Proyectos retrasados
- Exceso en los presupuestos iniciales
- Deficiencia respecto a los requerimientos fijados

→ CRISIS del software

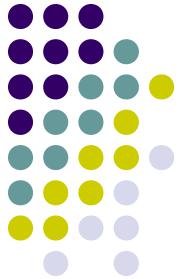
La complejidad del software



La ingeniería de software pretende:

- apoyar el desarrollo de software profesional o de dimensión industrial en lugar de la programación individual mediante técnicas, métodos y herramientas para dominar la complejidad;
- apoyar la especificación, el diseño y la evolución del sistema de software;
- considerar no sólo el software propiamente dicho sino también la documentación asociada y los datos de configuración requeridos para hacer que este software opere de manera correcta.

Tipos de productos de software



- Sistemas genéricos
  - Desarrollos estandarizados que se venden al mercado abierto. La organización que desarrolla también controla la especificación.
- Sistemas personalizados
  - Desarrollados por un contratista para un cliente en particular (a medida). El cliente participa en la especificación.
- Sistemas adaptados

Ingeniería de software
Evolución de la industria del software

Etapa1: Mayor importancia del hardware

- Software secundario con programación sin metodologías por empleados con conocimientos
- Poca movilidad laboral

Etapa 2: Evolución del hardware

- Necesidad de software más complejo
  - Aparición de las primeras "Casas de Software"
  - Movilidad laboral
  - Programación sin metodologías pero con desarrollos más grandes.
  - Mantenimiento difícil (mejor rehacer que modificar)

→ CRISIS del software

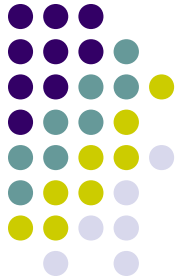
Ingeniería de software
Evolución de la industria del software

Etapa 3: Mayor importancia (y costos) del software…
pero

- ¿Por qué lleva tanto tiempo terminar los desarrollos?
- ¿Por qué es tan elevado el costo?
- ¿Por qué no es posible encontrar todos los errores antes de entregar el software al cliente?
- ¿Por qué resulta tan difícil constatar el progreso conforme se desarrolla el software?

Consecuencia → **La ingeniería de software**

Ingeniería de software
Características del software

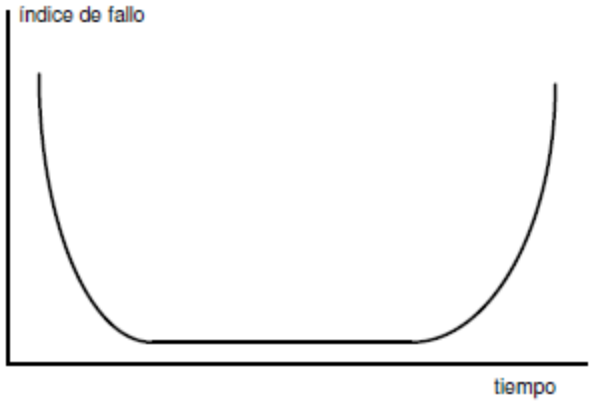


- SOFTWARE:
  - Instrucciones de computadora que cuando se ejecutan proporcionan la función y el comportamiento deseado,
  - Estructuras de datos que facilitan a los programas manipular adecuadamente la información, y
  - Documentos que describen la operación y el uso de los programas.
  - Naturaleza **LÓGICA**. Producto **INMATERIAL**

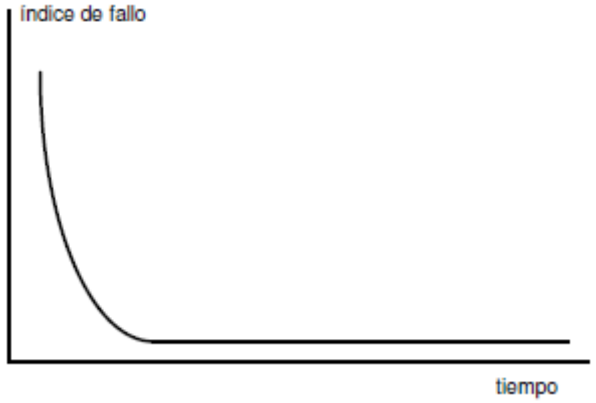
Ingeniería de software
Características del software

- No se fabrica, se desarrolla
  - El costo no está en los materiales
  - Costos en la ingeniería, no en la producción
  - Control de calidad de productos finales
- No se estropea ni se gasta

Índice de fallos del hardware



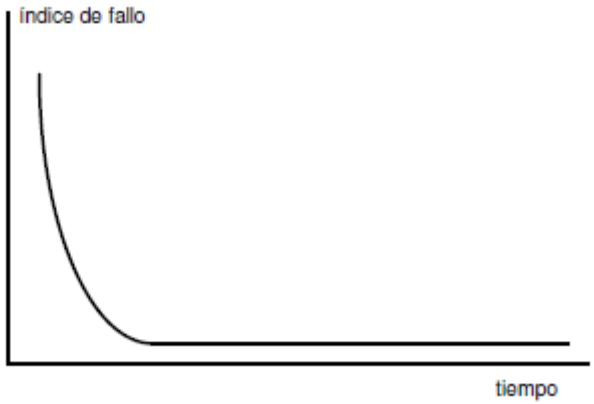
Índice de fallos del software



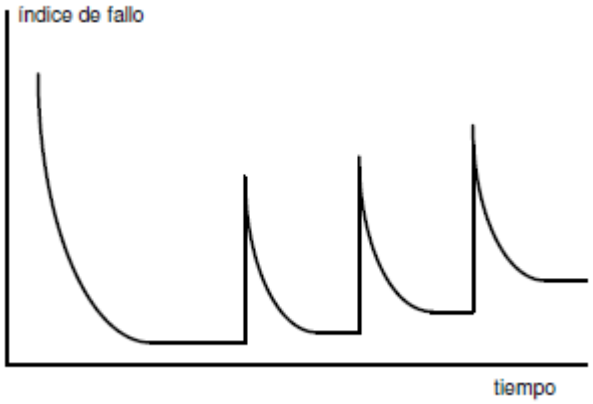
Ingeniería de software
Características del software

- No se estropea ni se gasta – Se deteriora

Índice de fallos teórico del software

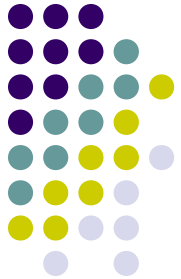


Índice de fallos real del software



- La mayoría del software es a medida
  - No existen catálogos de componentes
  - Baja reutilización para la mayoría de los desarrollos

Ingeniería de software
Problemas del software



- Imprecisa planificación y estimación de costos
  - Imprevistos
  - Planificación inexistente
  - Falta de acumulación de experiencia de proyectos anteriores
  - Administradores de proyectos que no saben de la producción de software
- La productividad es baja
  - Especificaciones ambiguas o incorrectas al inicio del proyecto
  - Inexistencia de evaluación del impacto de modificaciones
  - Inexistencia de documentación
- La calidad es mala
  - Consecuencia de lo anterior: software con muchos errores
  - Falta de planificación de pruebas
- Cliente insatisfecho
  - Análisis de requisitos inadecuado
  - Software con muchos errores

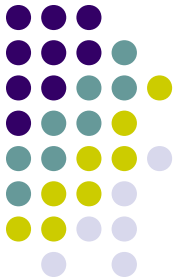
Ingeniería de software
Definición

Se encarga de:

- Definición de métodos aplicables a cada una de las fases del proceso de software;
- Construcción y uso de herramientas para automatizar estos métodos;
- Definición de procedimientos que establecen la secuencia de aplicación de los métodos, documentos y técnicas para garantizar la calidad de los productos desarrollados

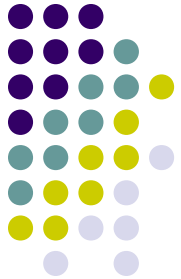
Se interesa por todos los aspectos de la producción de software, desde las primeras etapas de la especificación del sistema hasta el mantenimiento del sistema después que se pone en operación.

Ingeniería de software
Proceso de software



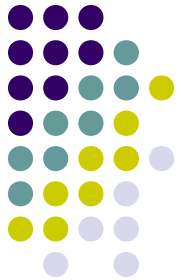
- El enfoque sistemático que usa la ingeniería de software se conoce como ***Proceso de software***, sus actividades son:
  - **Especificación del software**
    - Clientes e ingenieros definen las funcionalidades del software que se producirá y las restricciones de su operación.
  - **Diseño e implementación del software**
    - Diseño y programación cumpliendo con las especificaciones.
  - **Validación del software**
    - Verificación el software para asegurar que sea lo que el cliente requiere.
  - **Evolución del software**
    - Modificación del software para reflejar los requerimientos cambiantes del cliente y el mercado.
- Diferentes tipos de sistemas o aplicaciones necesitan distintos procesos de desarrollo.

Ingeniería de software
Tipos de aplicaciones



- **Aplicaciones independientes.**
- **Aplicaciones interactivas basadas en transacción.**
- **Sistemas de control embebido.**
- **Sistema de procesamiento por lotes.**
- **Sistemas de entretenimiento.**
- **Sistemas para modelado y simulación.**
- **Sistemas de adquisición de datos.**

Modelos de proceso de software
Ciclo de vida



Es la sucesión de etapas por las que pasa el software desde que un nuevo proyecto es concebido hasta que se deja de usar.

Clasificación:

- **Dirigidos por un plan**

  Las actividades del proceso se planean por anticipado y el avance se mide contra dicho plan

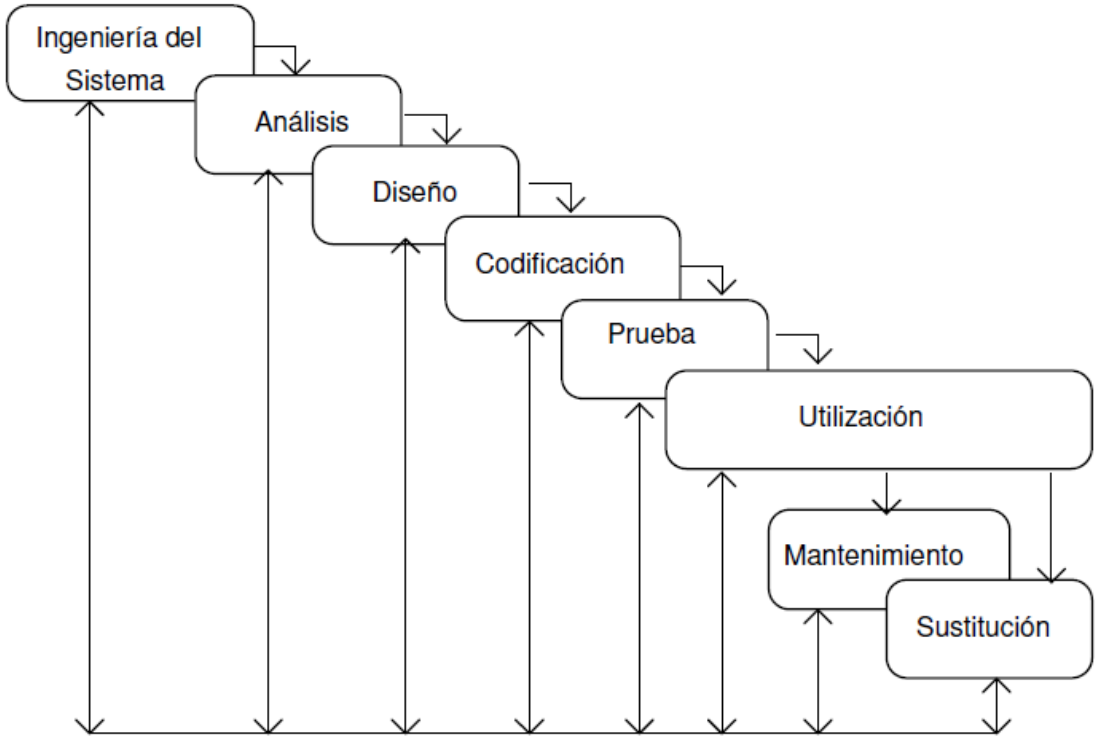
  - **Modelo en cascada (waterfall)**
  - **Desarrollo incremental o evolutivo**
  - **Ingeniería de software orientada a la reutilización.**

- **Procesos ágiles**

  La planeación es incremental y es más fácil de modificar el proceso para reflejar los requerimientos cambiantes del cliente

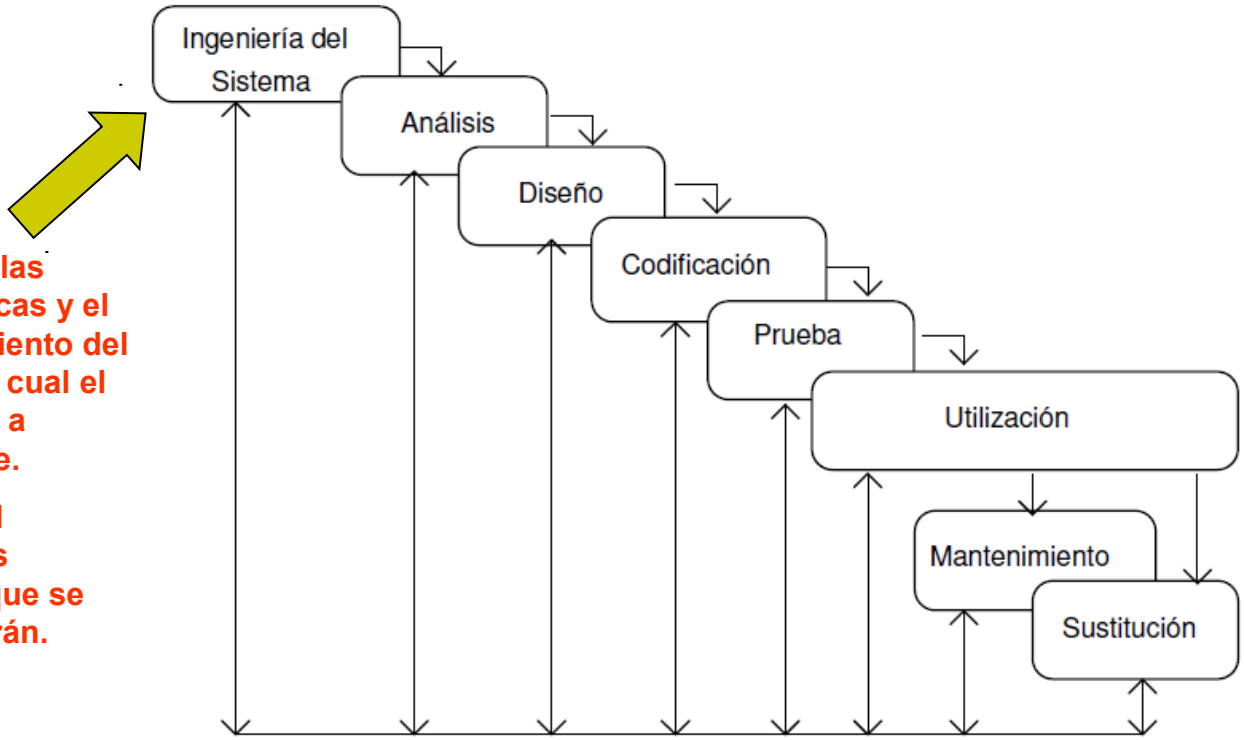
Modelos de proceso de software
Modelo en cascada

Es el más antiguo y se desarrolló a partir del ciclo convencional de una ingeniería



Modelos de proceso de software
Modelo en cascada

Es el más antiguo y se desarrolló a partir del ciclo convencional de una ingeniería

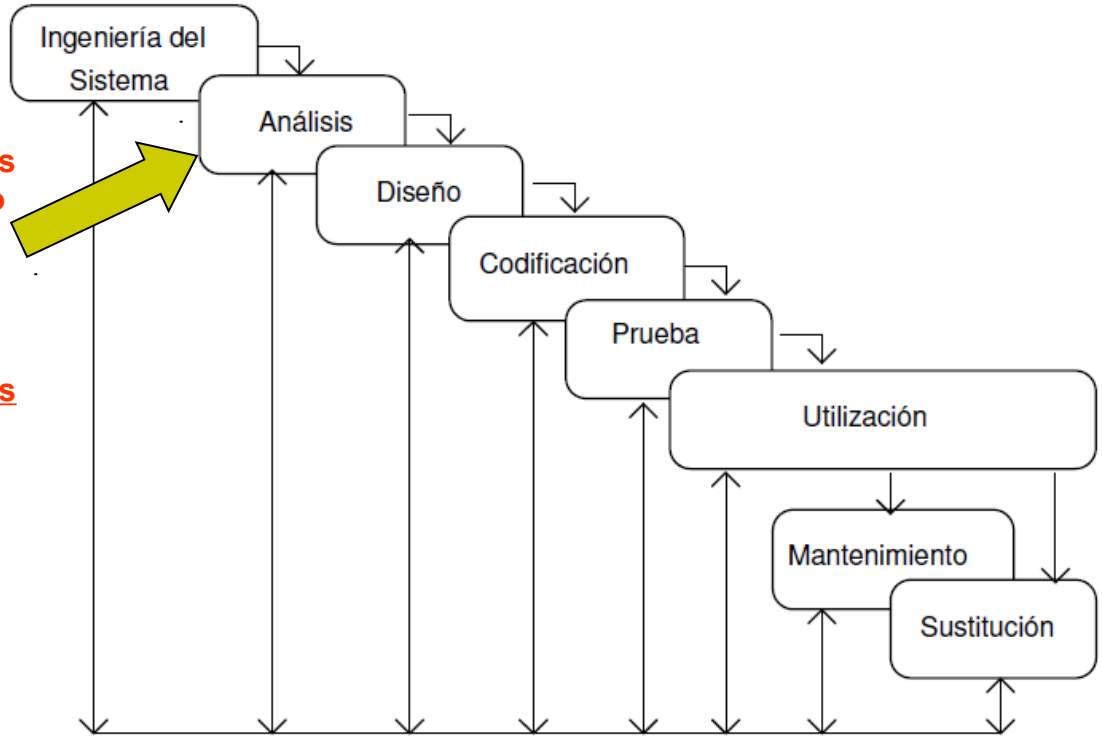


Análisis de las características y el comportamiento del sistema del cual el software va a formar parte.

Asignarle al software las funciones que se automatizarán.

Modelos de proceso de software
Modelo en cascada

Es el más antiguo y se desarrolló a partir del ciclo convencional de una ingeniería

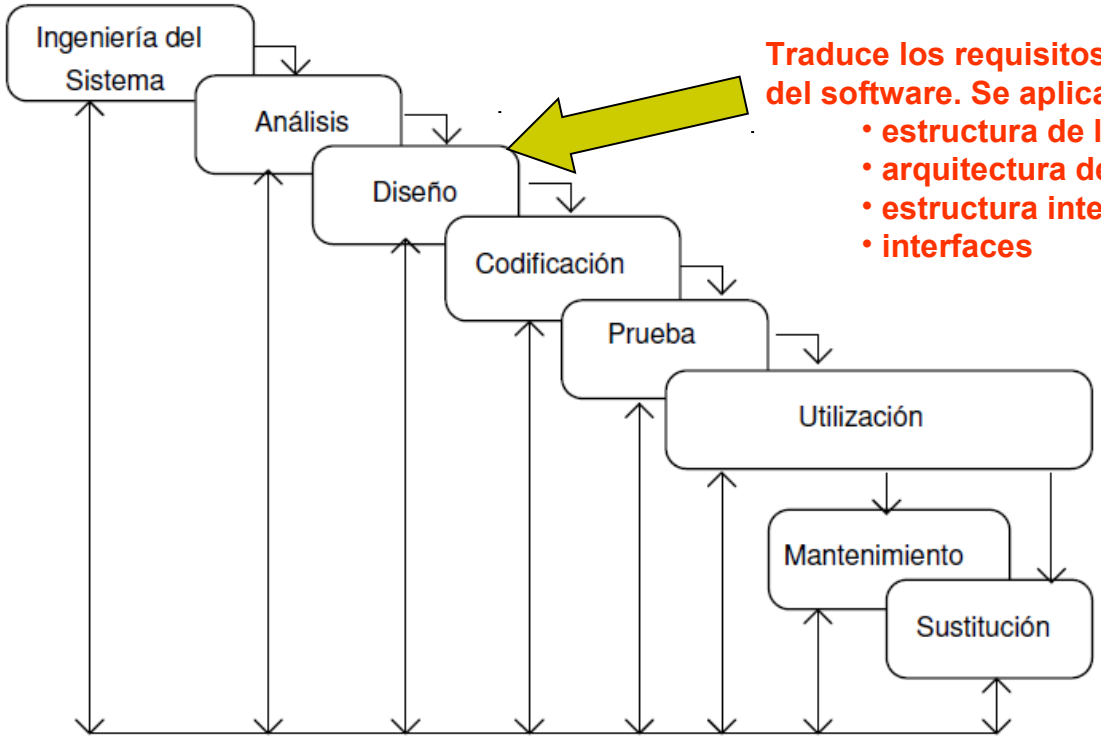


El análisis de requisitos debe ser más detallado para aquellos componentes del sistema que se van a implementar mediante software consensuados con el cliente

* **Datos**
* **Funcionalidades**
* **Interfaces**

Modelos de proceso de software
Modelo en cascada

Es el más antiguo y se desarrolló a partir del ciclo convencional de una ingeniería



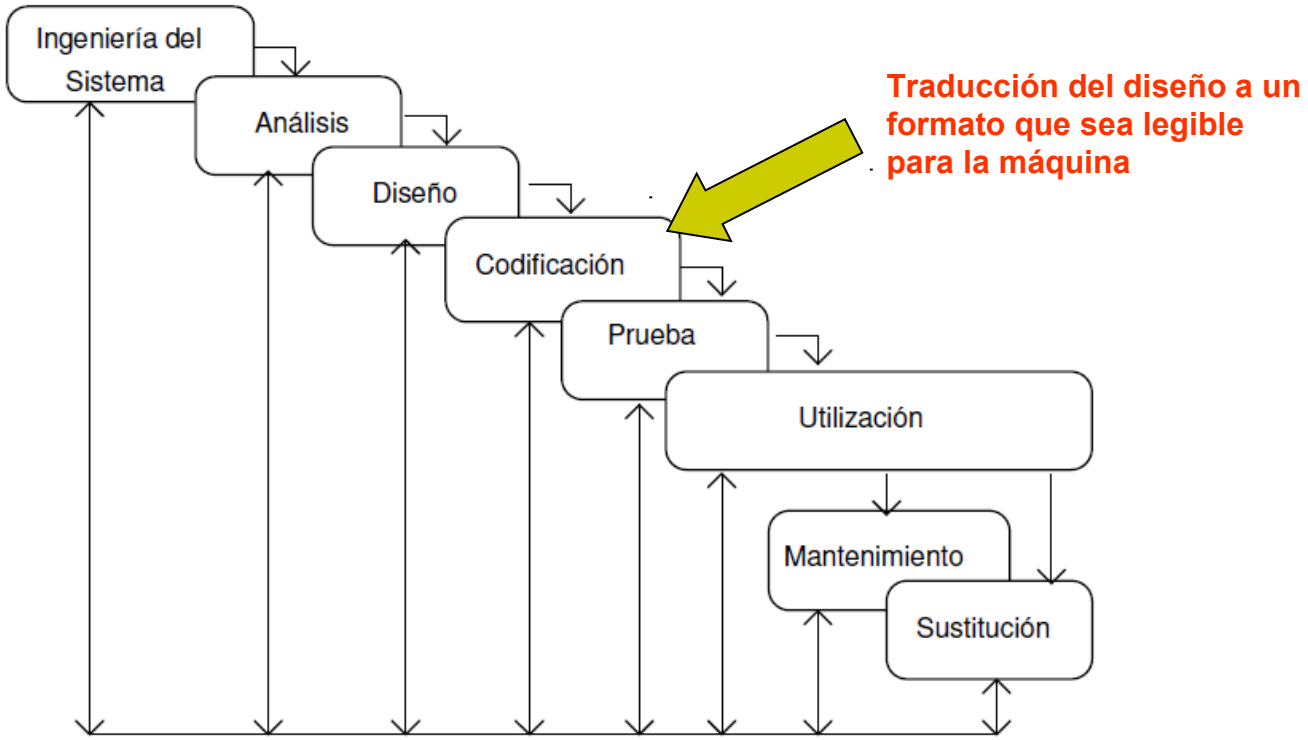
Traduce los requisitos en una representación del software. Se aplica a:

- **estructura de los datos**
- **arquitectura de las aplicaciones**
- **estructura interna de los programas**
- **interfaces**

Modelos de proceso de software

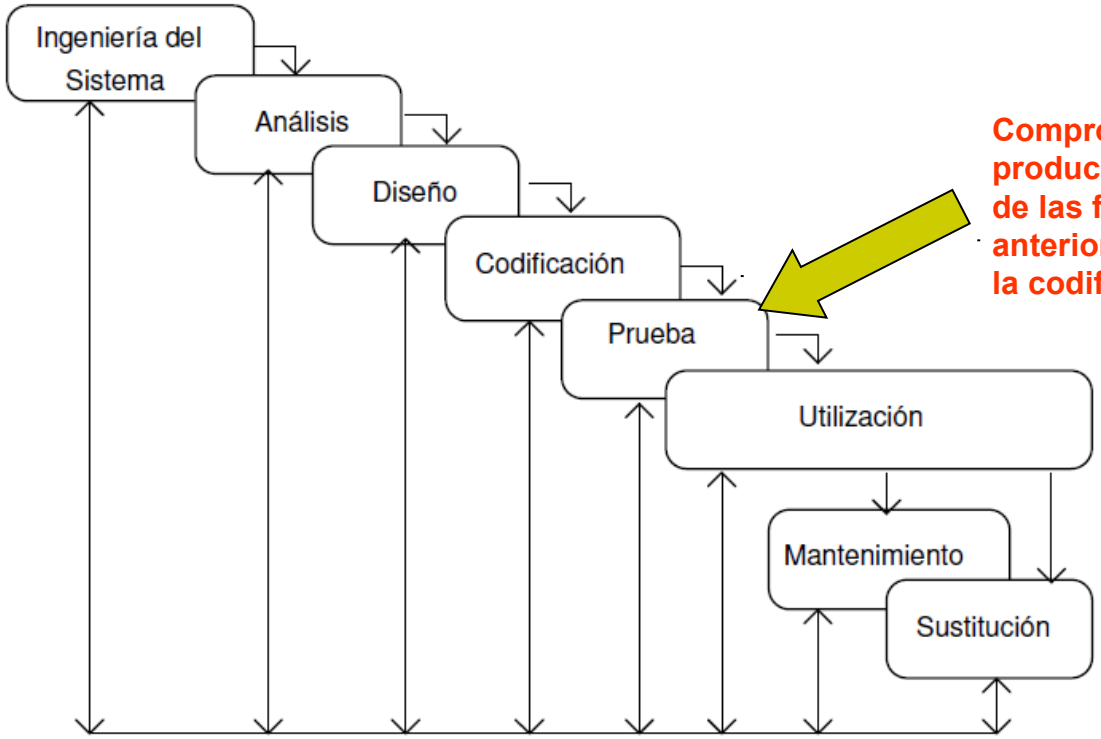
Modelo en cascada

Es el más antiguo y se desarrolló a partir del ciclo convencional de una ingeniería



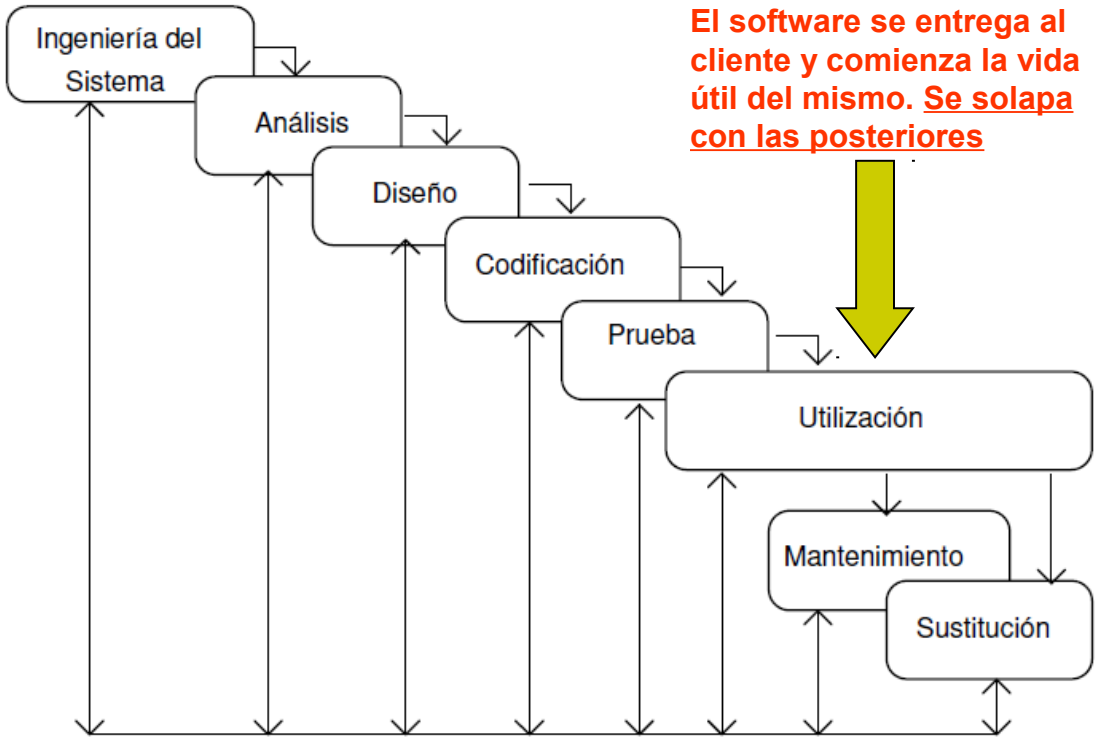
Modelos de proceso de software
Modelo en cascada

Es el más antiguo y se desarrolló a partir del ciclo convencional de una ingeniería



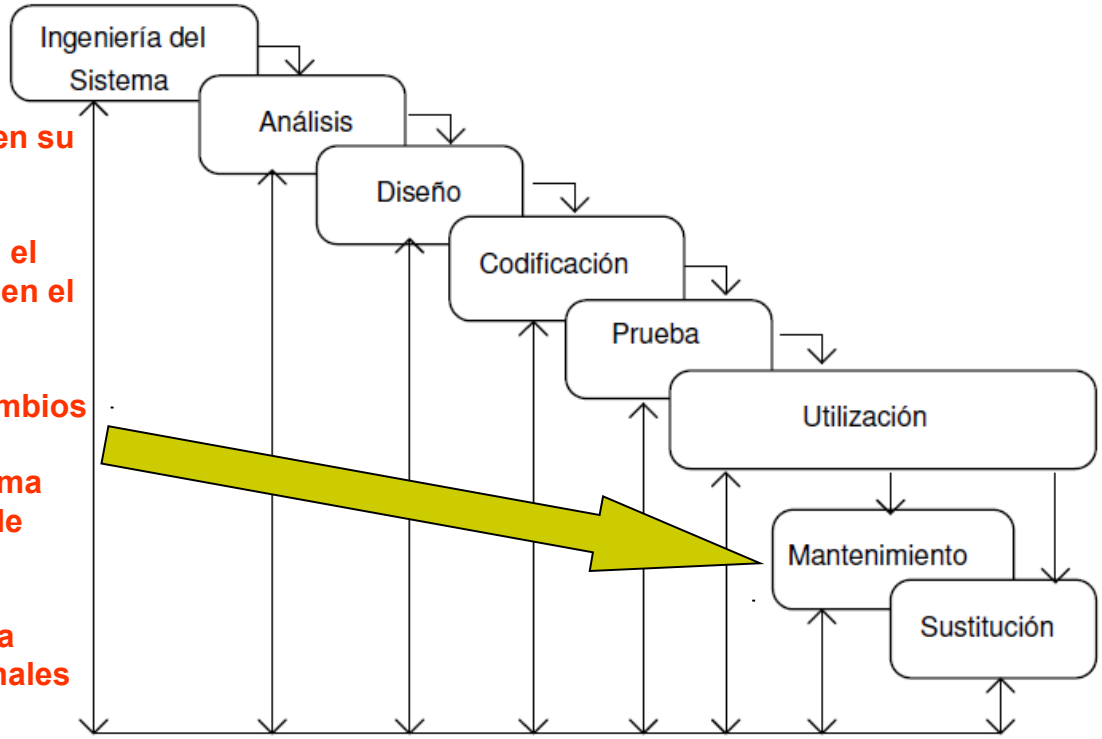
Modelos de proceso de software
Modelo en cascada

Es el más antiguo y se desarrolló a partir del ciclo convencional de una ingeniería



Modelos de proceso de software
Modelo en cascada

Es el más antiguo y se desarrolló a partir del ciclo convencional de una ingeniería

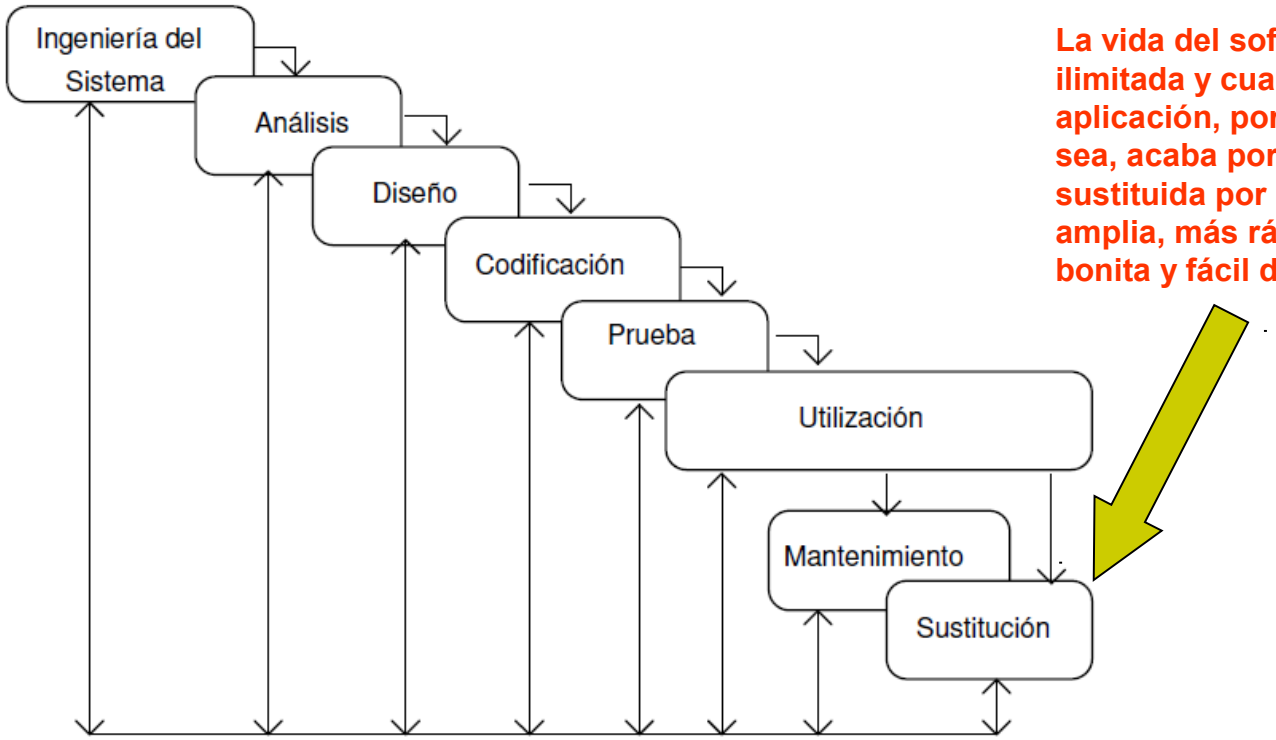


Cambios del software en su vida útil por:

- **Durante la utilización, el cliente detecte errores en el software.**
- **Que se produzcan cambios en alguno de los componentes del sistema informático (software de base, hardware).**
- **Que el cliente requiera modificaciones funcionales o ampliaciones no contempladas en el proyecto.**

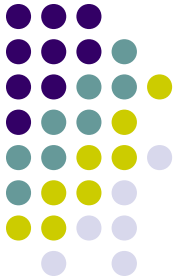
Modelos de proceso de software
Modelo en cascada

Es el más antiguo y se desarrolló a partir del ciclo convencional de una ingeniería



La vida del software no es ilimitada y cualquier aplicación, por buena que sea, acaba por ser sustituida por otra más amplia, más rápida o más bonita y fácil de usar

Modelos de proceso de software
Modelo en cascada



Problemas

- En la realidad los proyectos no siguen un ciclo de vida estrictamente secuencial. Hay iteraciones.
- Es difícil que se puedan establecer inicialmente todos los requisitos del sistema. El ciclo de vida clásico requiere la definición inicial de todos los requisitos y no es fácil acomodar en él las incertidumbres que suelen existir al comienzo de todos los proyectos.
- Hasta que se llega a la fase final del desarrollo (la codificación) no se dispone de una versión operativa de las aplicaciones.

Modelos de proceso de software
Modelo incremental o evolutivo

- Se basa en la idea de desarrollar una implementación inicial exponiéndola a los comentarios del usuario y refinándola a través de las diferentes versiones hasta que se desarrolla un sistema adecuado.

- Las actividades de especificación, desarrollo y validación se entrelazan en vez de separarse con una rápida retroalimentación entre éstas.

- Tipos de desarrollo evolutivo:
  - **Desarrollo exploratorio**
  - **Prototipos desechables**

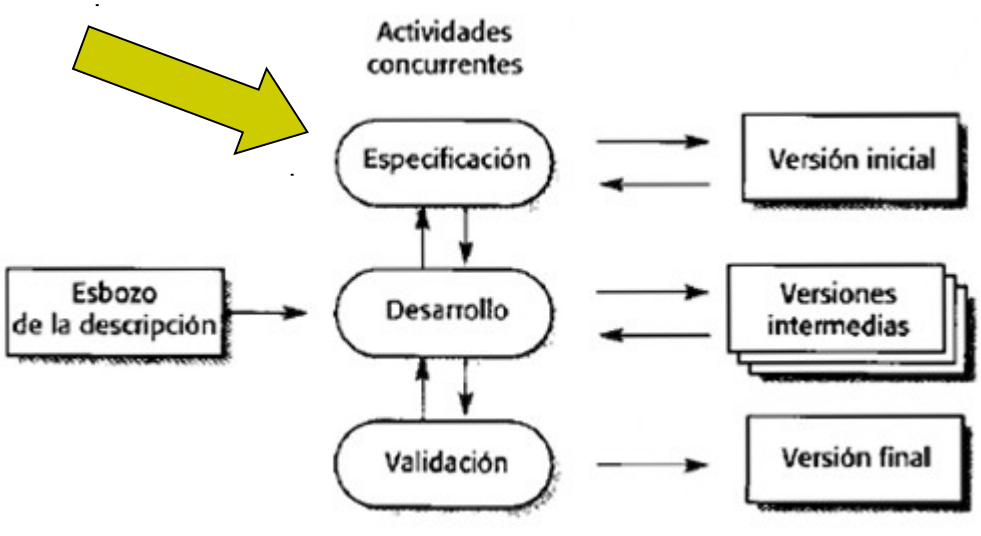
Modelos de proceso de software
Modelo incremental o evolutivo

- Desarrollo exploratorio:
  - Trabajar con el cliente para explorar los requerimientos.
  - Desarrollar primero **las partes más conocidas** y luego se evoluciona con las propuestas del cliente.

- Prototipos desechables:
  - Comprender los requerimientos del cliente y desarrollar una definición mejorada de los requerimientos del sistema. Se centra en los requerimientos que **no se comprenden del todo**.

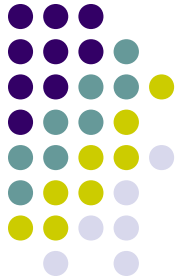
Modelos de proceso de software
Modelo incremental o evolutivo

Enfoque evolutivo. La especificación se puede desarrollar en forma creciente



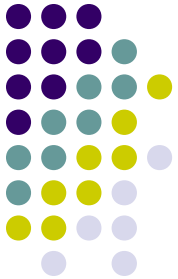
Cada incremento o versión del sistema incorpora algunas de las funciones que necesita el cliente

Modelos de proceso de software
Modelo en espiral de Boehm



- Combina las principales ventajas del ciclo de vida en cascada y el de construcción de prototipos.
- Proporciona un modelo evolutivo para el desarrollo de sistemas de software complejos más realista que el ciclo de vida en cascada.
- Permite la utilización de prototipos en cualquier etapa de la evolución del proyecto.
- Es un modelo de proceso dirigido por el riesgo.

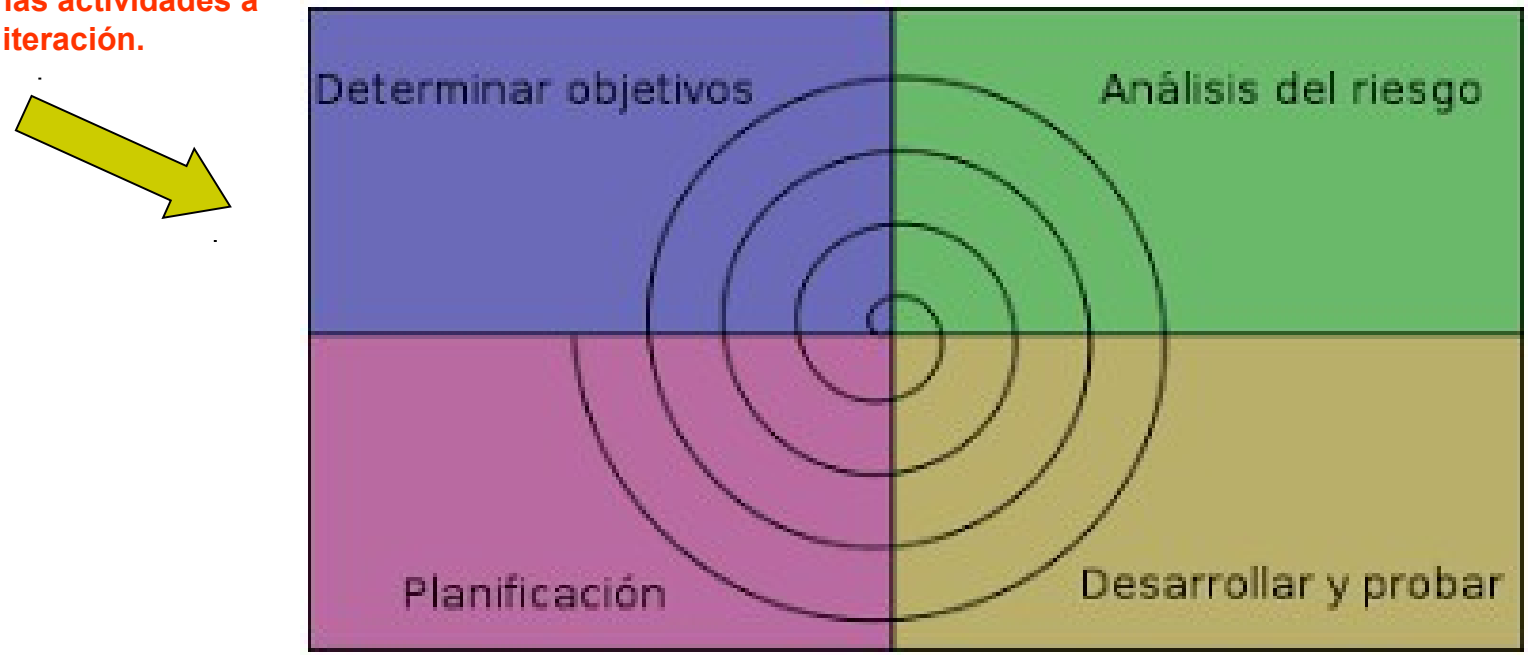
Modelos de proceso de software
Modelo en espiral de Boehm



- El proceso de software se presenta como una espiral y no como una secuencia de actividades con cierto retroceso de una actividad a otra. Cada ciclo en la espiral representa una fase del proceso de software, de esta manera, el ciclo más interno puede relacionarse con la factibilidad del sistema, el siguiente con la definición de requerimientos, el siguiente con el diseño del sistema, etc.

Modelos de proceso de software
Modelo en espiral de Boehm

Consiste en determinar las posibles alternativas y las restricciones. Esta fase equivale a la de recolección de requisitos del ciclo de vida clásico e incluye además la planificación de las actividades a realizar en cada iteración.



Modelos de proceso de software

Modelo en espiral de Boehm

En cada uno de los riesgos identificados del proyecto, se realiza un análisis minucioso. Se dan acciones para reducir el riesgo. Por ejemplo, si existe un riesgo que los requerimientos sean inadecuados, puede desarrollarse un sistema prototipo.

Modelos de proceso de software
Modelo en espiral de Boehm

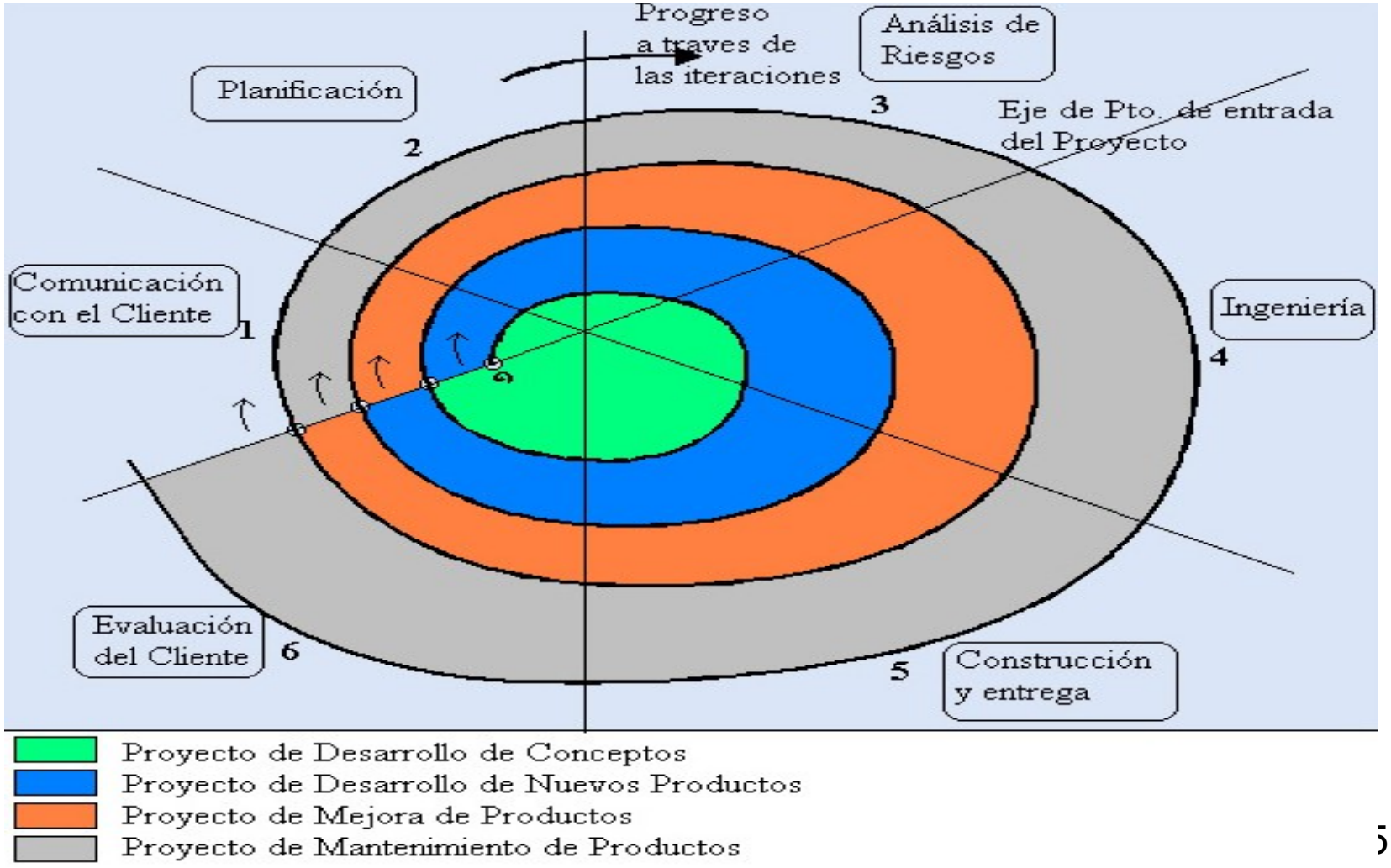
Después de una evaluación del riesgo, se elige un modelo de desarrollo para el sistema. Por ejemplo, la creación de prototipos desechables sería el mejor enfoque de desarrollo si predominan los riesgos en la interfaz del usuario. Si el principal riesgo identificado es la integración de subsistemas, el modelo en cascada sería el mejor.

Modelos de proceso de software
Modelo en espiral de Boehm

El proyecto se revisa y se toma una decisión sobre si hay que continuar con otro ciclo de la espiral. Si se opta por continuar, se trazan los planes para la siguiente fase del proyecto.

Modelos de proceso de software

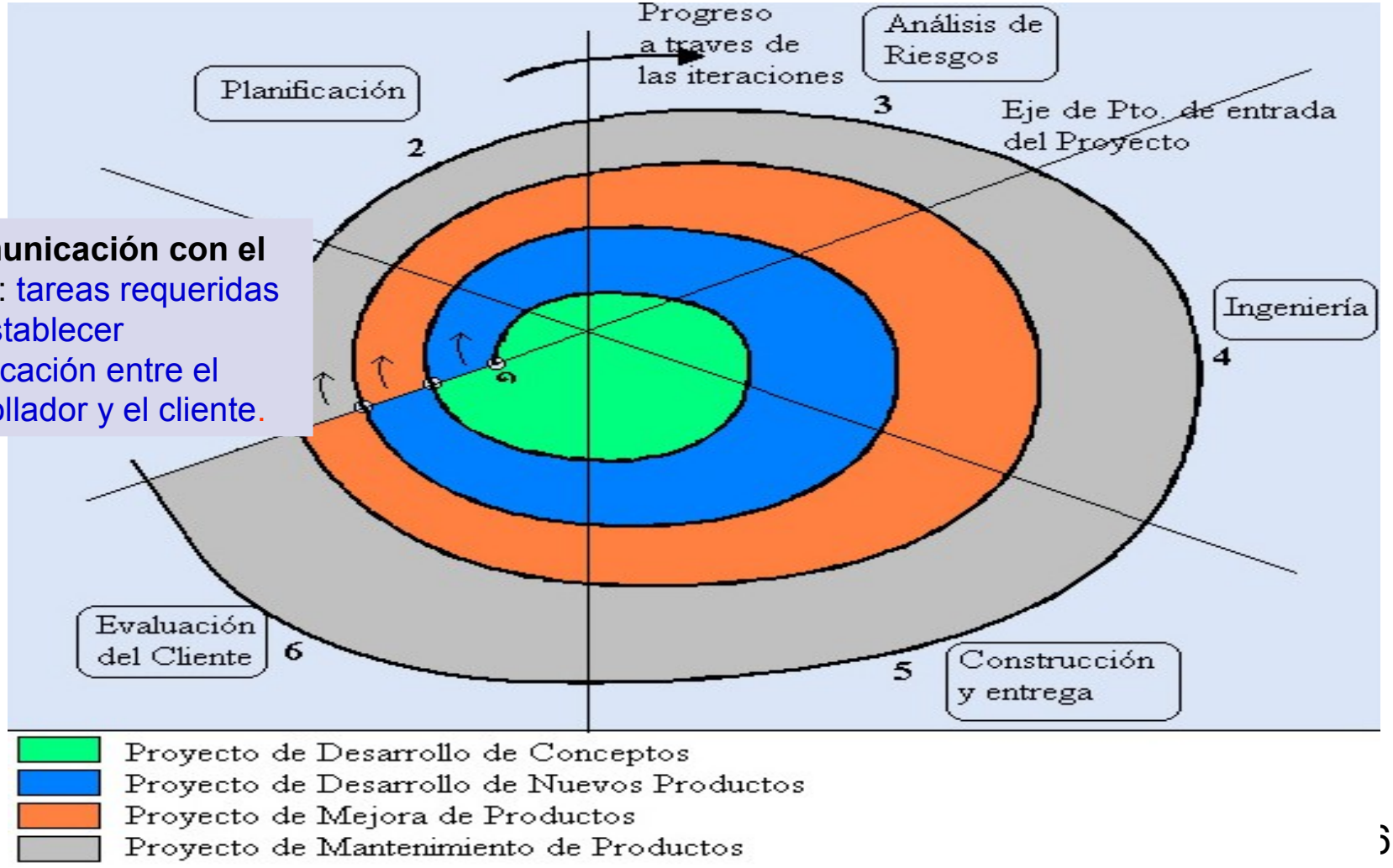
Modelo típico de seis regiones



Modelos de proceso de software

Modelo típico de seis regiones

1. **Comunicación con el cliente**: tareas requeridas para establecer comunicación entre el desarrollador y el cliente.



Modelos de proceso de software

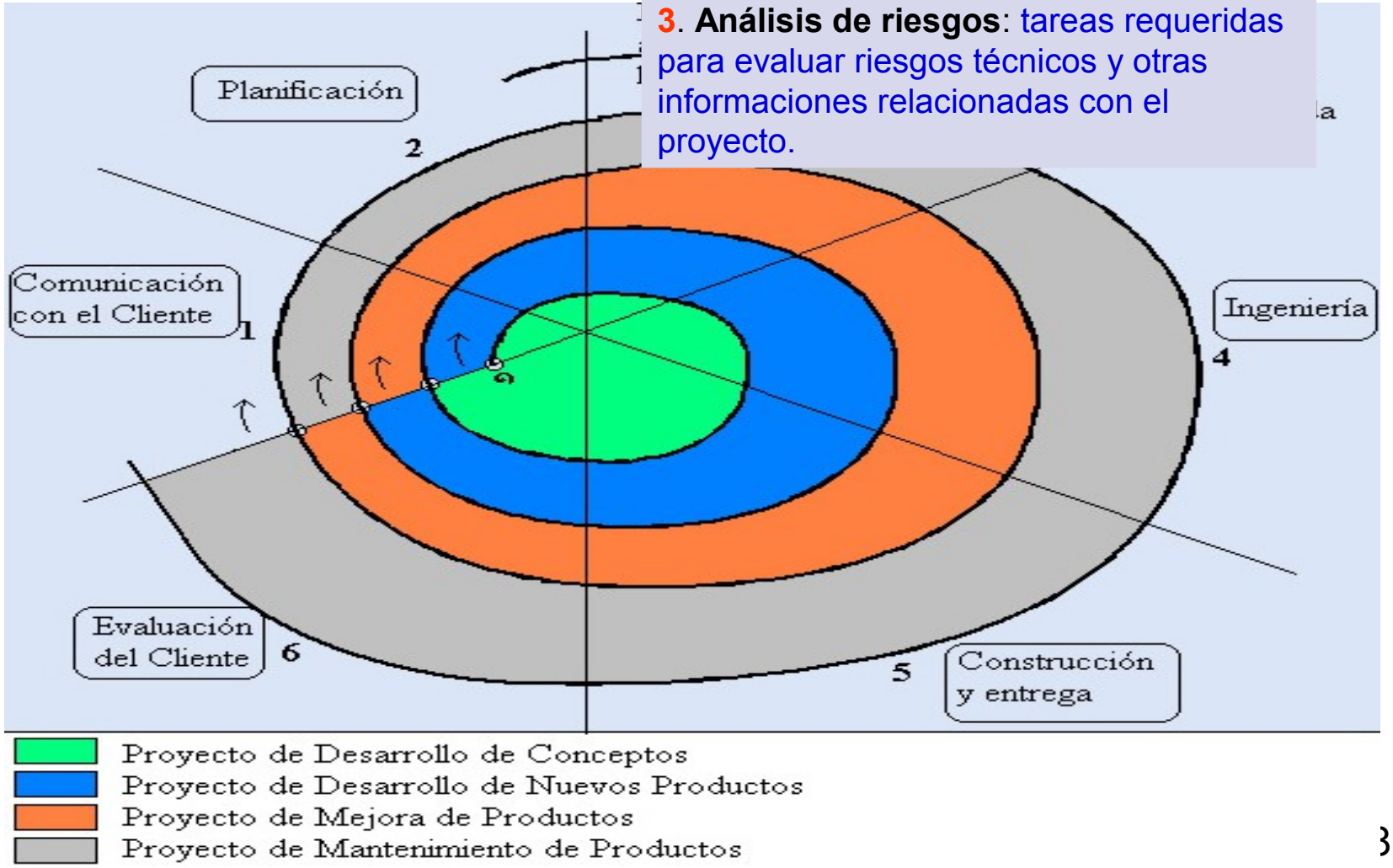
Modelo típico de seis regiones

2. **Planificación**: tareas requeridas para definir recursos, el tiempo y otras informaciones relacionadas con el proyecto.
Son todos los requerimientos.

Modelos de proceso de software

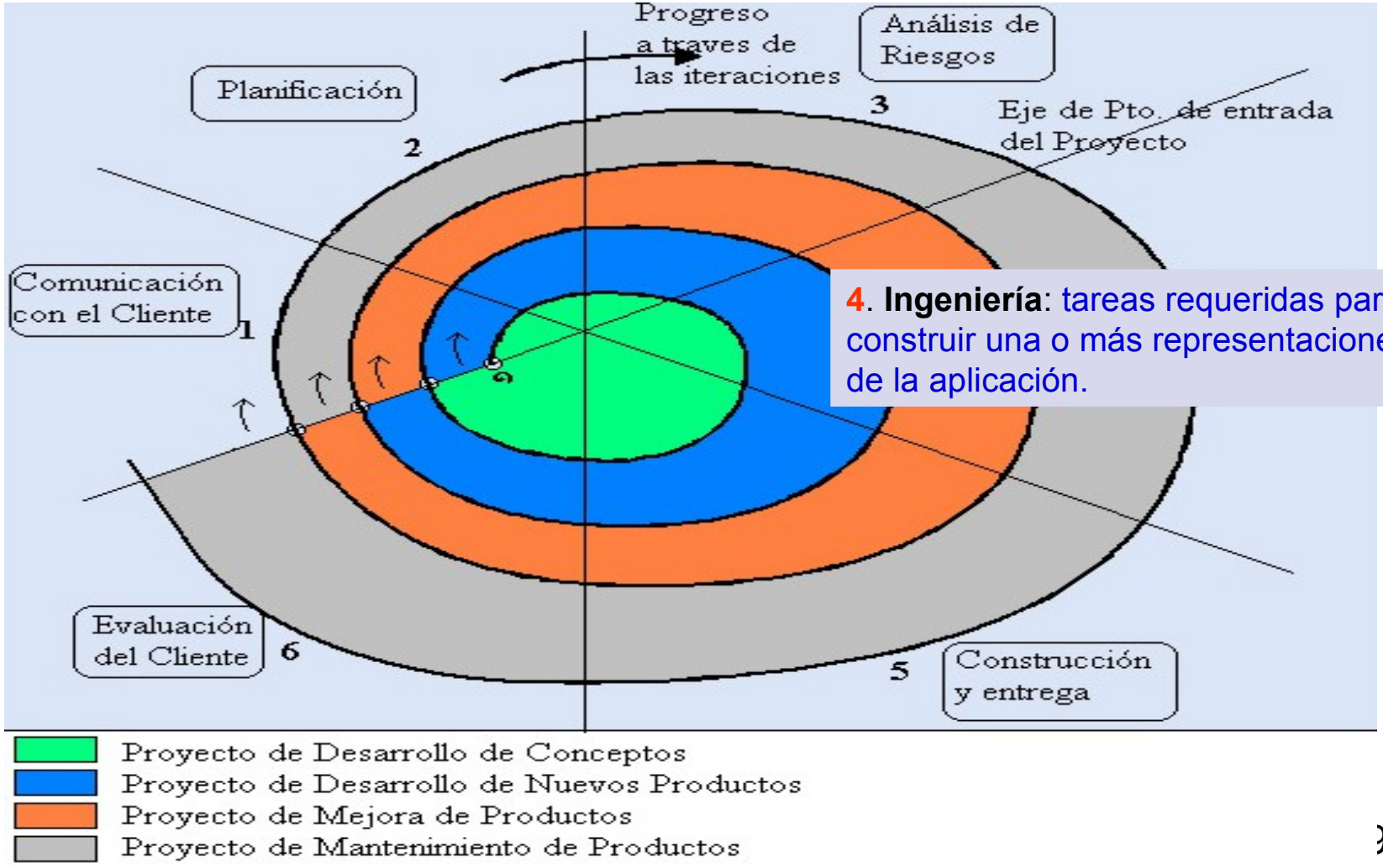
Modelo típico de seis regiones

3. **Análisis de riesgos**: tareas requeridas para evaluar riesgos técnicos y otras informaciones relacionadas con el proyecto.



Modelos de proceso de software

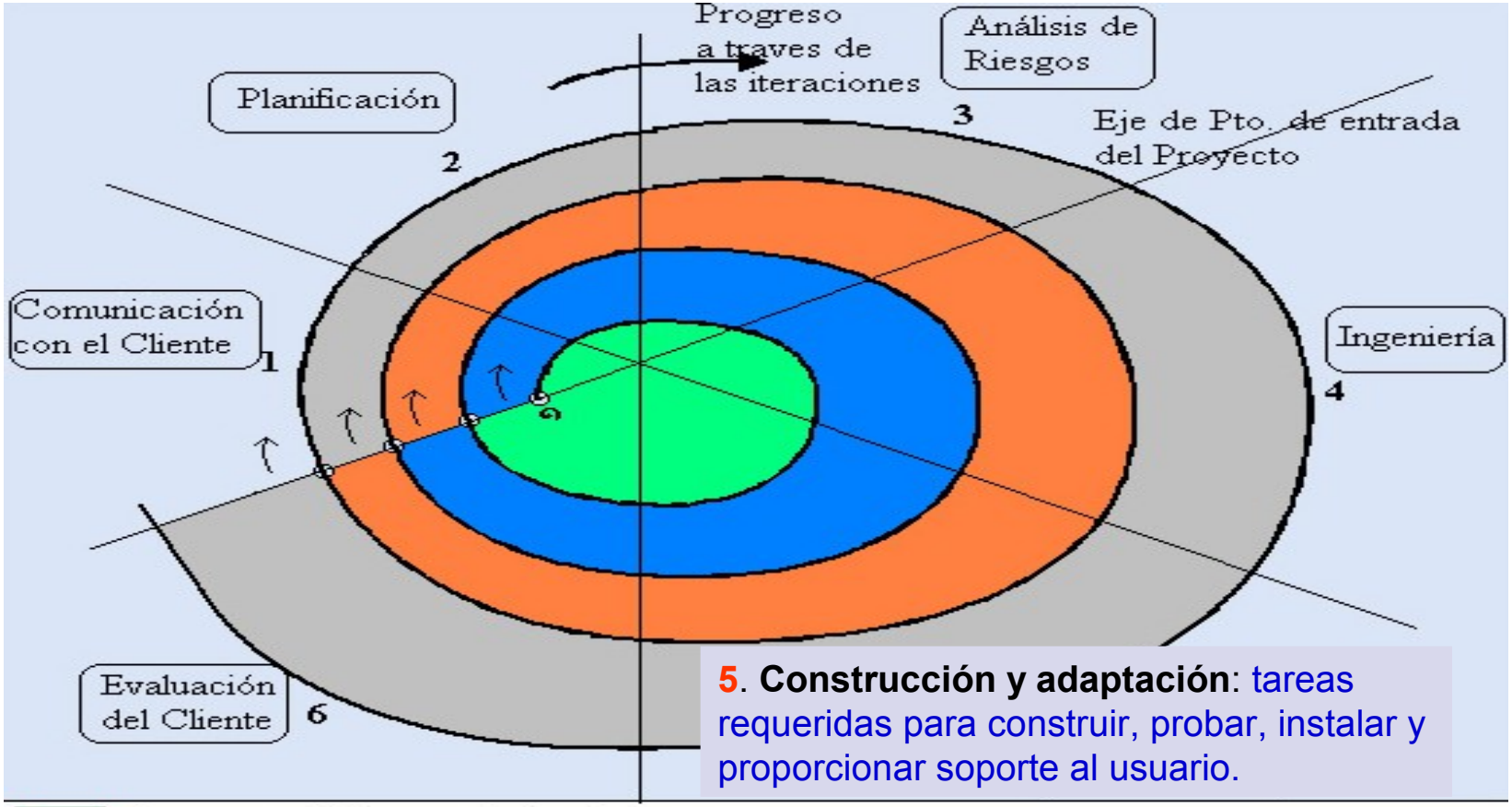
Modelo típico de seis regiones



4. **Ingeniería**: tareas requeridas para construir una o más representaciones de la aplicación.

Modelos de proceso de software

Modelo típico de seis regiones



5. **Construcción y adaptación**: tareas requeridas para construir, probar, instalar y proporcionar soporte al usuario.

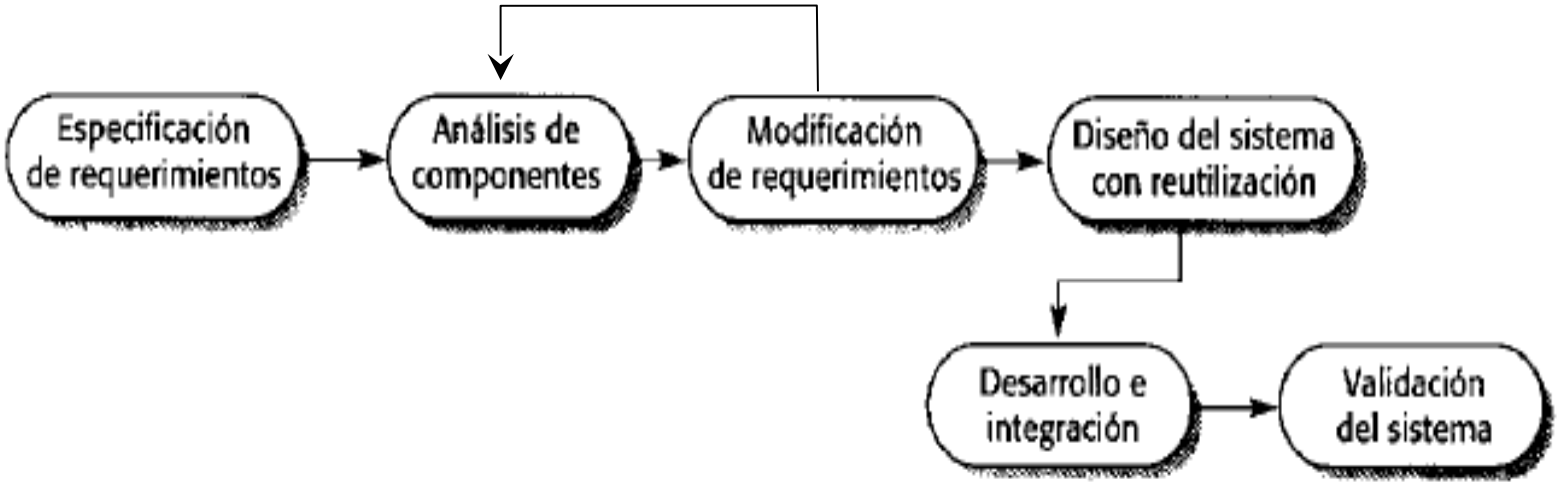
Modelos de proceso de software

Modelo típico de seis regiones

6. **Evaluación del cliente**: tareas requeridas para obtener la reacción del cliente según la evaluación de las representaciones del software creadas durante la etapa de ingeniería e implementación durante la etapa de instalación.

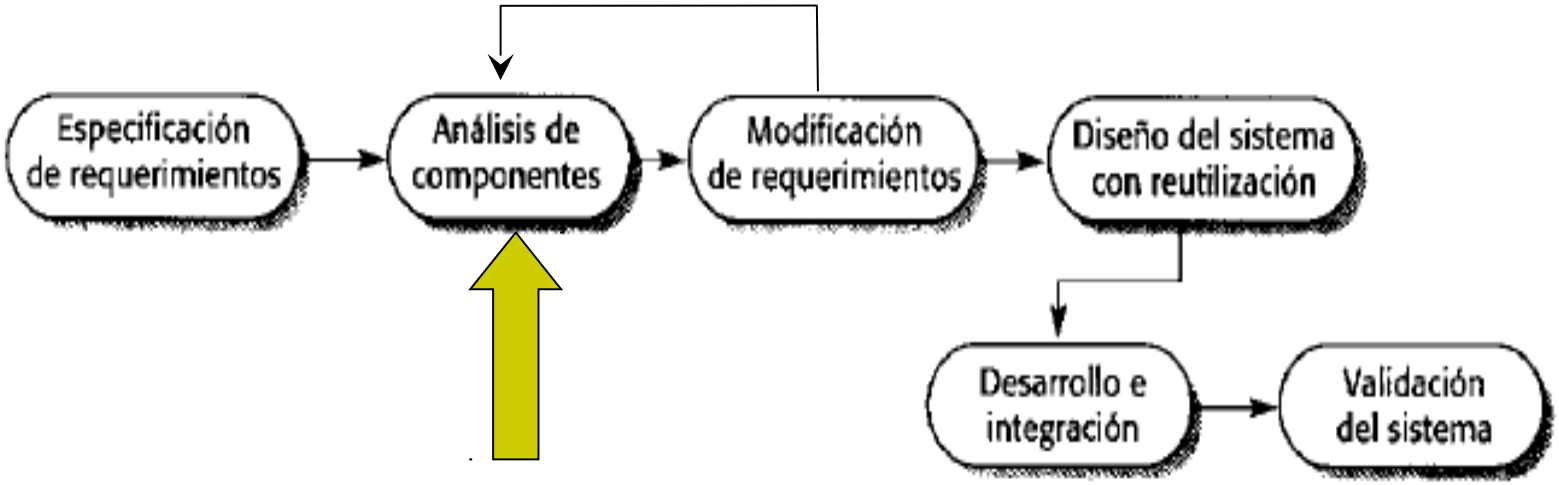
Modelos de proceso de software
IS orientada a la reutilización

- La reutilización ocurre independientemente del proceso de desarrollo empleado.



Modelos de proceso de software
IS orientada a la reutilización

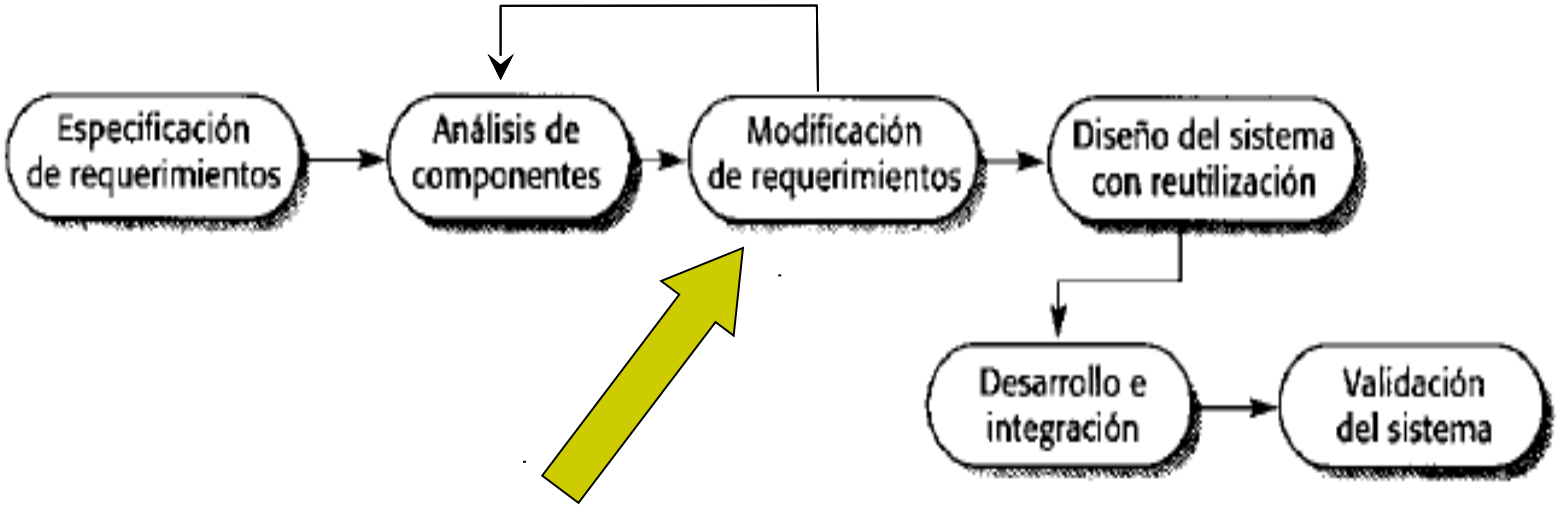
- La reutilización ocurre independientemente del proceso de desarrollo empleado.



Dada la especificación de requerimientos, se realiza una búsqueda de componentes para implementar dicha especificación. Por lo general, no hay coincidencia exacta y los componentes que se usa proporcionan sólo parte de la funcionalidad requerida.

Modelos de proceso de software
IS orientada a la reutilización

- La reutilización ocurre independientemente del proceso de desarrollo empleado.



En esta etapa se analizan los requerimientos usando información de los componentes descubiertos. Luego se modifican para reflejar los componentes disponibles. Donde las modificaciones son imposibles, puede regresarse a la actividad de análisis de componentes para buscar soluciones alternativas.

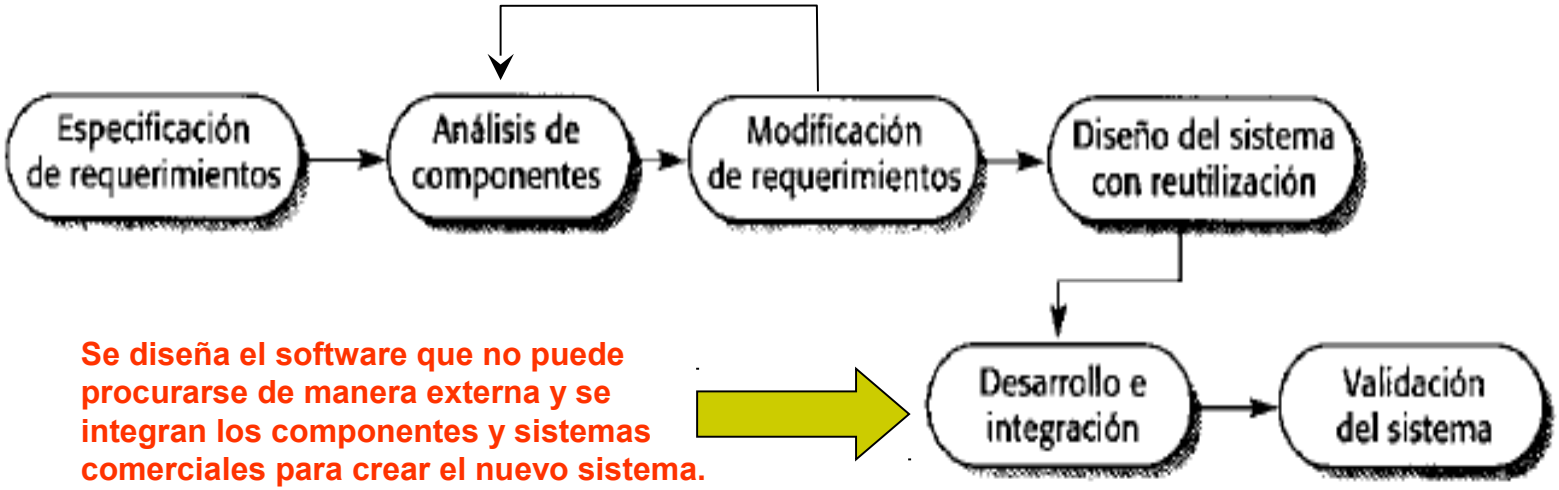
Modelos de proceso de software
IS orientada a la reutilización

- La reutilización ocurre independientemente del proceso de desarrollo empleado.

En esta etapa se diseña el marco conceptual del sistema o se reutiliza el existente. Los creadores toman en cuenta los componentes que se reutilizan y organizan el marco de referencia para atenderlo. Es posible que deba diseñarse algo de software nuevo si no están disponibles los componentes reutilizables.

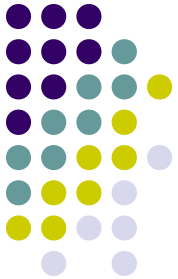
Modelos de proceso de software
IS orientada a la reutilización

- La reutilización ocurre independientemente del proceso de desarrollo empleado.



Se diseña el software que no puede procurarse de manera externa y se integran los componentes y sistemas comerciales para crear el nuevo sistema.

Modelos de proceso de software
IS orientada a la reutilización



Componentes que se pueden utilizar:

- Servicios Web que se desarrollan en concordancia para atender servicios estándares y que están disponibles para la invocación remota.

- Colecciones de objetos que se desarrollan como un paquete para su integración con el marco de componentes como J2EE o .NET.

- Sistemas de software independientes que se configuran para usar en un entorno particular.