Unidad 1:

Conceptos previos sobre ingeniería y proyectos de software

Unidad 1: Conceptos previos sobre ingeniería y proyectos de software

- Proceso de Resolución de Problemas.
- Proceso de la Construcción de Software.
- El proceso de Software: Definición, Beneficios, Madurez. Ciclos de Vida. Estándar IEEE sobre Proceso Software.
- Mapa de Actividades de un Proyecto. Nociones básicas sobre proyectos de software. Gestión y análisis de riesgos: identificación, enunciación, análisis, impacto y matriz de riesgos, mitigación y planes de contingencia.
- Planeamiento. Tareas críticas. Grafo de tareas. Herramientas: CPM y PERT.

Tiempo aproximado: 12 hs.

Objetivos

Que el alumno sea capaz de:

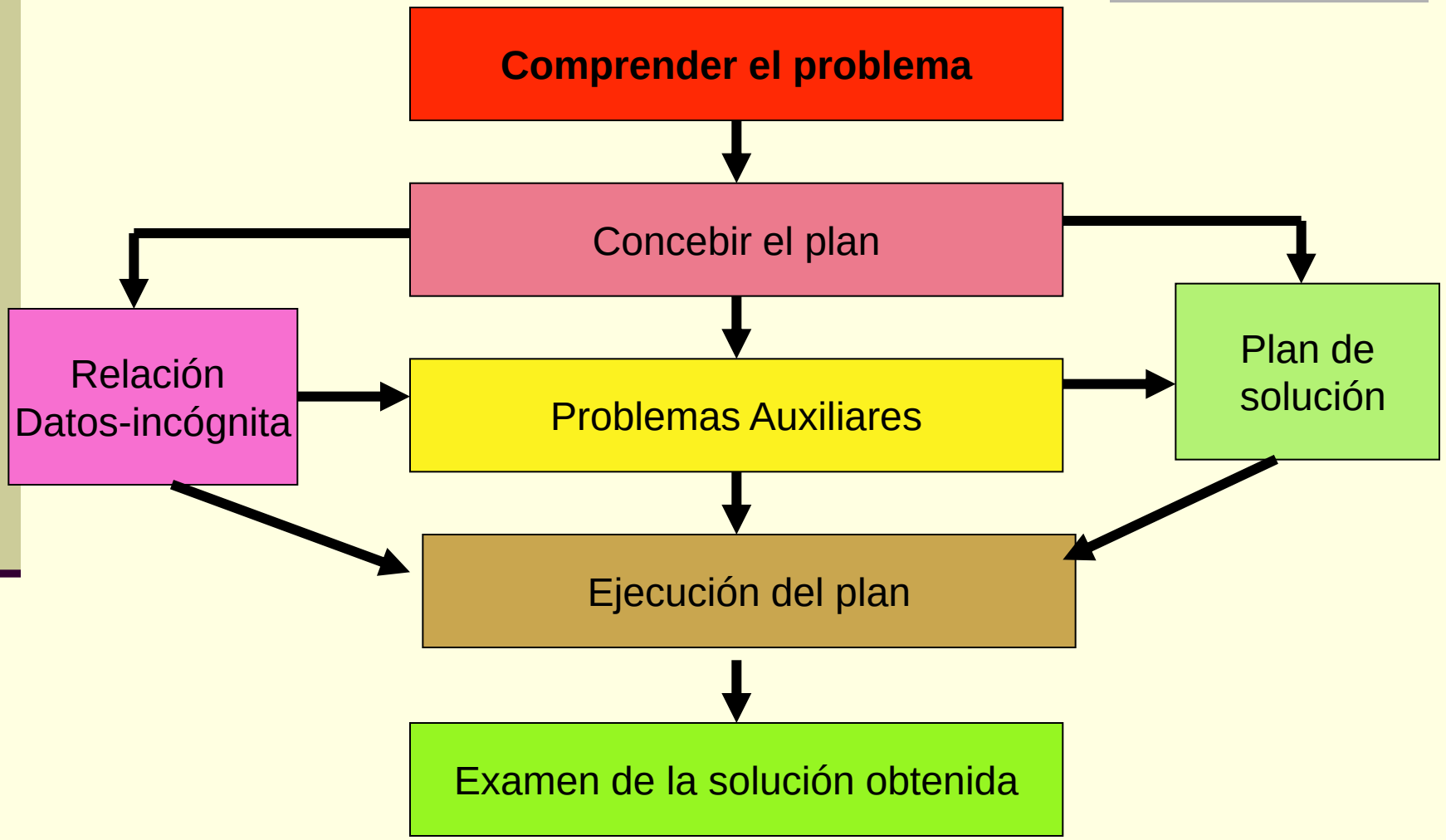
- Comprender que es la Ingeniería de Software y su aplicación en la formación de capacidades en el Licenciado en Sistemas de Información.
- Valorar la importancia de la construcción de los modelos de software como herramienta para gestionar proyectos en el contexto de las distintas organizaciones.
- Incorporar un marco conceptual que sea base para la comprensión y aplicación de modelos de ingeniería que eficienticen la construcción de proyectos de creación y/o modificación software.
- Comparar los distintos métodos en función de la eficiencia y la utilidad que brinda para el desarrollo de software.
- Investigar los distintos estándares para procesos de software disponibles en el mercado a través de búsquedas por internet y comparar el beneficio de cada uno.

Proceso de Resolución de Problemas

Poyla, considerado para muchos el padre de la heurística matemática, estableció cuatro fases en la resolución de problemas:

1. **Comprender el problema** : *¿Cuál es la incógnita?, ¿Cuáles son los datos?*
2. **Concebir un plan**: ¿Se ha encontrado con un problema semejante?,¿Conoce un problema relacionado con este?, ¿Podría enunciar el problema de otra forma?, ¿Ha empleado todos los datos?
3. **Ejecutar el plan**: ¿Son correctos los pasos dados?
4. **Examinar la solución obtenida**: ¿Puede verificar el resultado?, ¿Puede verificar el razonamiento?

Proceso de Resolución de Problemas



Procesos de Software

Objetivos

- Introducir modelos de procesos de software
- Describir tres modelos de procesos genéricos y cuándo pueden utilizarse
- Describir modelos de procesos de ingeniería de requerimientos, de desarrollo de software, pruebas y la evolución
- Explicar el modelo de Proceso Unificado de Rational
- Introducir la tecnología CASE para apoyar las actividades de proceso del software

Tópicos expuestos

- Modelos de procesos de software
- Iteración de procesos
- Actividades del proceso
- El Proceso Unificado de Rational
- Ingeniería de Software Asistida por Computadora

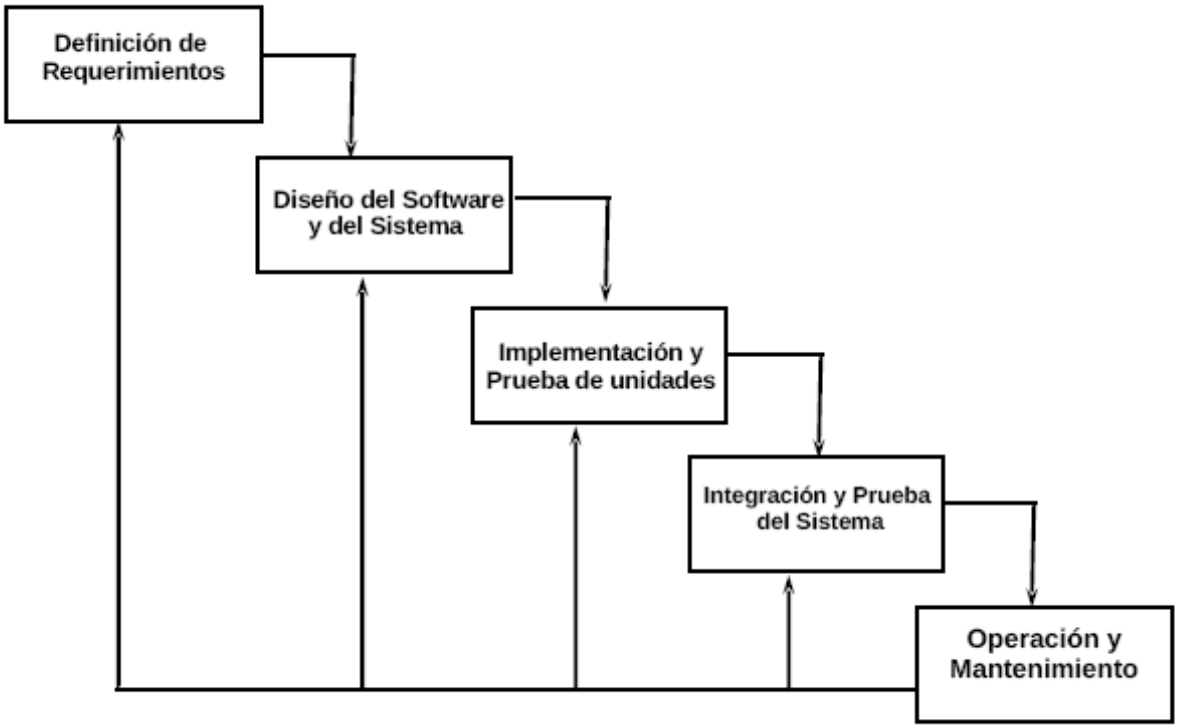
El proceso del software

- Un conjunto estructurado de actividades necesarias para desarrollar un Sistema de software
  - Específicación;
  - Diseño;
  - Validación;
  - Evolución.
- Un modelo de proceso del software es una representación abstracta de un proceso. Presenta una descripción de un proceso desde una perspectiva particular.

Modelos genéricos de proceso de software

- El modelo de cascada
  - Separadas y distintas fases de la especificación y el desarrollo.
- Desarrollo evolutivo
  - Especificación, desarrollo y validación están intercalados.
- Ingeniería de Software Basada en Componentes
  - El sistema está montado a partir de los componentes existentes.
- Hay muchas variantes de estos modelos, por ejemplo, el desarrollo formal donde se toma un proceso similar al de cascada, pero la especificación es una especificación formal que se perfecciona a través de varias etapas hacia un diseño implementable.

Modelo de cascada



Modelo de cascada - fases

- Análisis de requerimientos y definición
- diseño del sistema y del software
- Ejecución y unidad de pruebas
- Implementación y pruebas del sistema
- Operación y mantenimiento
- El principal inconveniente del modelo de cascada es la dificultad de acomodar el cambio después de que el proceso está en marcha. Una fase tiene que ser completada antes de pasar a la siguiente fase.

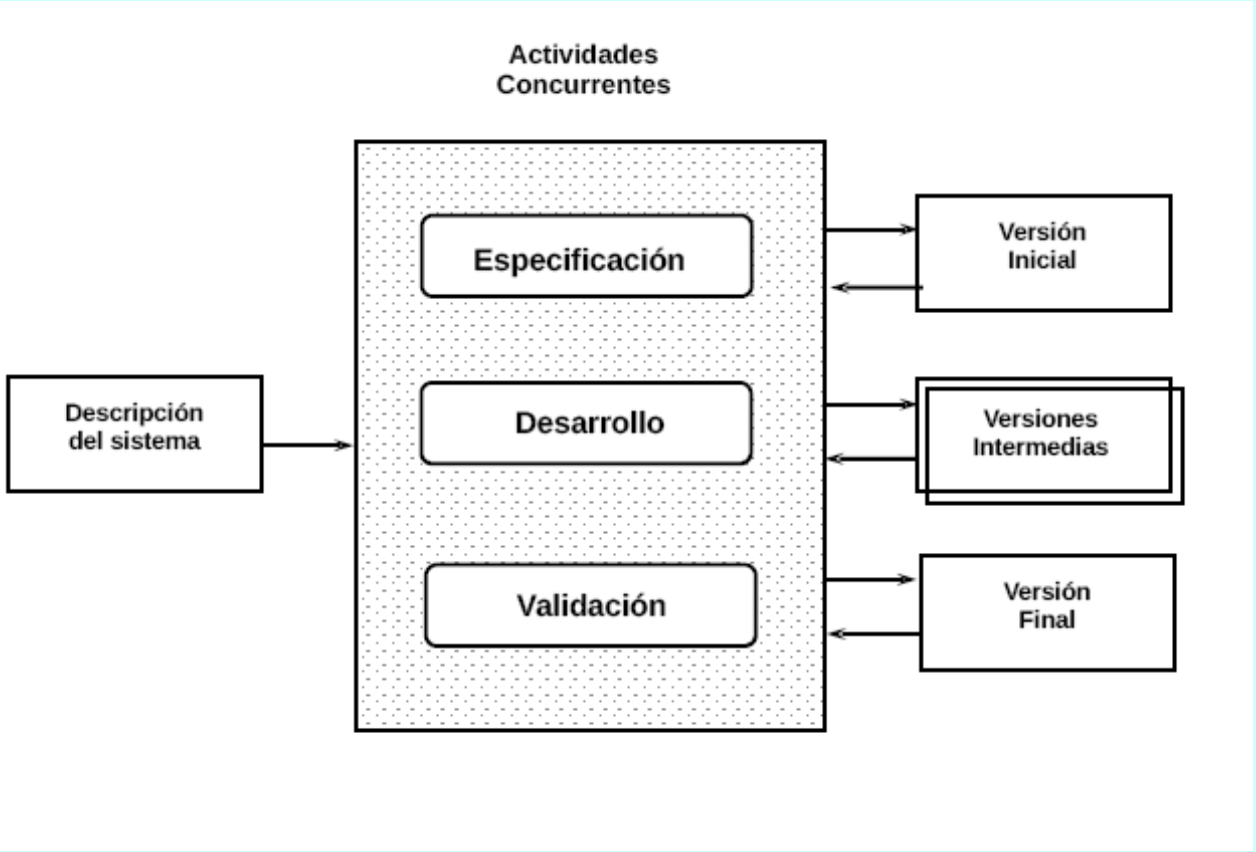
Modelo de cascada - problemas

- Rígida división del proyecto en fases distintas que hace difícil responder a la evolución de las necesidades del cliente.
- Por lo tanto, este modelo sólo es apropiado cuando los requisitos son bien entendidos y los cambios serán bastante limitados durante el proceso de diseño.
- Pocos sistemas de negocio tienen requerimientos estables.
- El modelo de cascada se utiliza para grandes proyectos de ingeniería de sistemas donde el desarrollo de un sistema se efectúe en varios sitios.

Desarrollo evolutivo

- Desarrollo exploratorio
  - El objetivo es trabajar con los clientes y evolucionar a un sistema final a partir de un esbozo inicial de especificación. Debería empezar con buen entendimiento de los requerimientos y añadir nuevas funciones en la forma propuesta por el cliente.
- Prototipado desechable
  - El objetivo es comprender los requisitos del sistema. Se puede empezar con especificaciones poco entendidas.

Desarrollo evolutivo



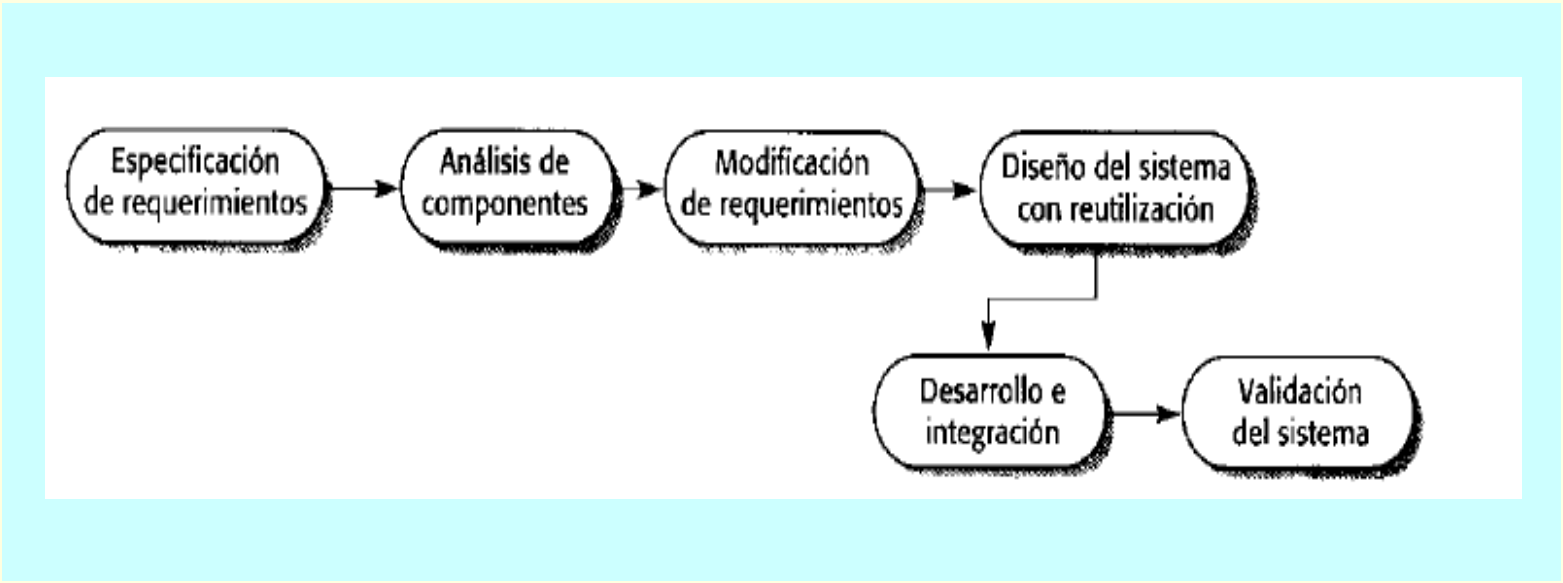
Desarrollo evolutivo

- Problemas
  - Poca visibilidad en el proceso;
  - Los sistemas son a menudo mal estructurado;
  - Habilidades especiales (por ejemplo, lenguajes para prototipado rápido) pueden ser requeridas.
- Aplicabilidad
  - Para sistemas interactivos pequeños o medianos;
  - Para partes de grandes sistemas (por ejemplo, la interfaz de usuario);
  - Para sistemas de corta vida.

Ingeniería de Software Basada en Componentes

- Sobre la base de la reutilización sistemática donde se integran los sistemas de los componentes existentes o COTS (Comercial-off-the-shelf) de sistemas.
- Las fases del proceso
  - Análisis de componentes;
  - Modificación de requerimientos;
  - Diseño del sistema con reutilización;
  - Desarrollo e integración.
- Este enfoque está siendo cada vez más utilizado a medida que los componentes estándar van surgiendo.

Desarrollo orientado a la reutilización



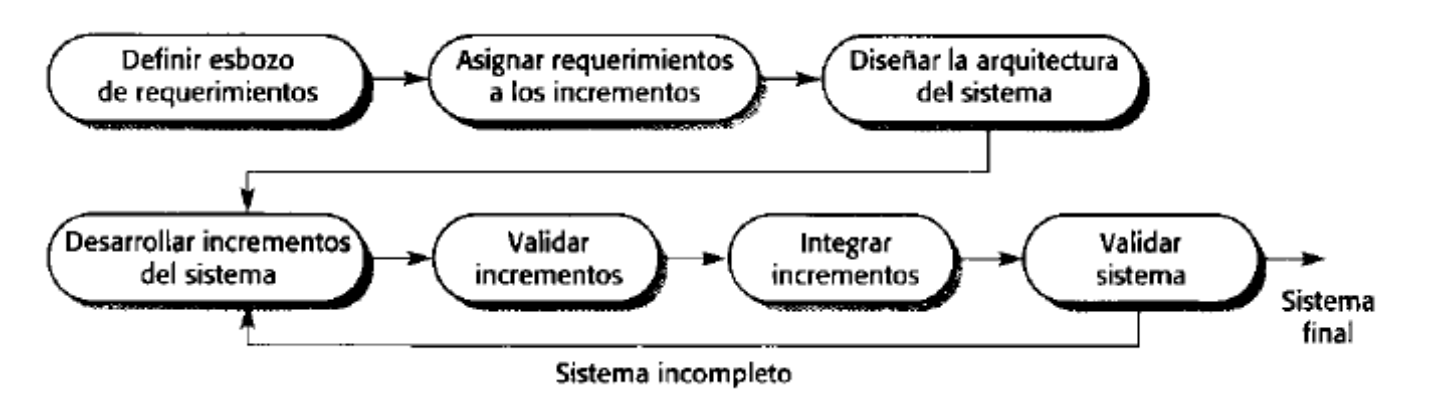
Iteración de Procesos

- Los requerimientos del sistema SIEMPRE evolucionan en el transcurso de un proyecto, así que el proceso de iteración donde las fases son revisadas, siempre son parte del proceso de grandes sistemas.
- Iteración se puede aplicar a cualquiera de los modelos de procesos genéricos.
- Dos enfoques (relacionados)
  - Entrega incremental;
  - Desarrollo en Espiral.

Entrega Incremental

- En lugar de entregar el sistema en una sola entrega, el desarrollo y la prestación se divide en incrementos, y con cada incremento se entrega parte de la funcionalidad requerida.
- Son priorizados los requerimientos de los usuarios y los requerimientos de más alta prioridad son incluidos en los primeros incrementos.
- Una vez que el desarrollo de un incremento se ha iniciado, los requisitos están congelados, a pesar de que los requerimientos de los incrementos posteriores pueden seguir evolucionando.

Entrega Incremental



Ventajas del Desarrollo incremental

- Entregas con valor para el cliente pueden ser hechas con cada incremento, de manera que la funcionalidad del sistema está disponible tempranamente.
- Los primeros incrementos actúan como un prototipo para ayudar a obtener requerimientos para incrementos posteriores.
- Menor riesgo de fracaso del proyecto general.
- La más alta prioridad del sistema de servicios tiende a recibir más pruebas.

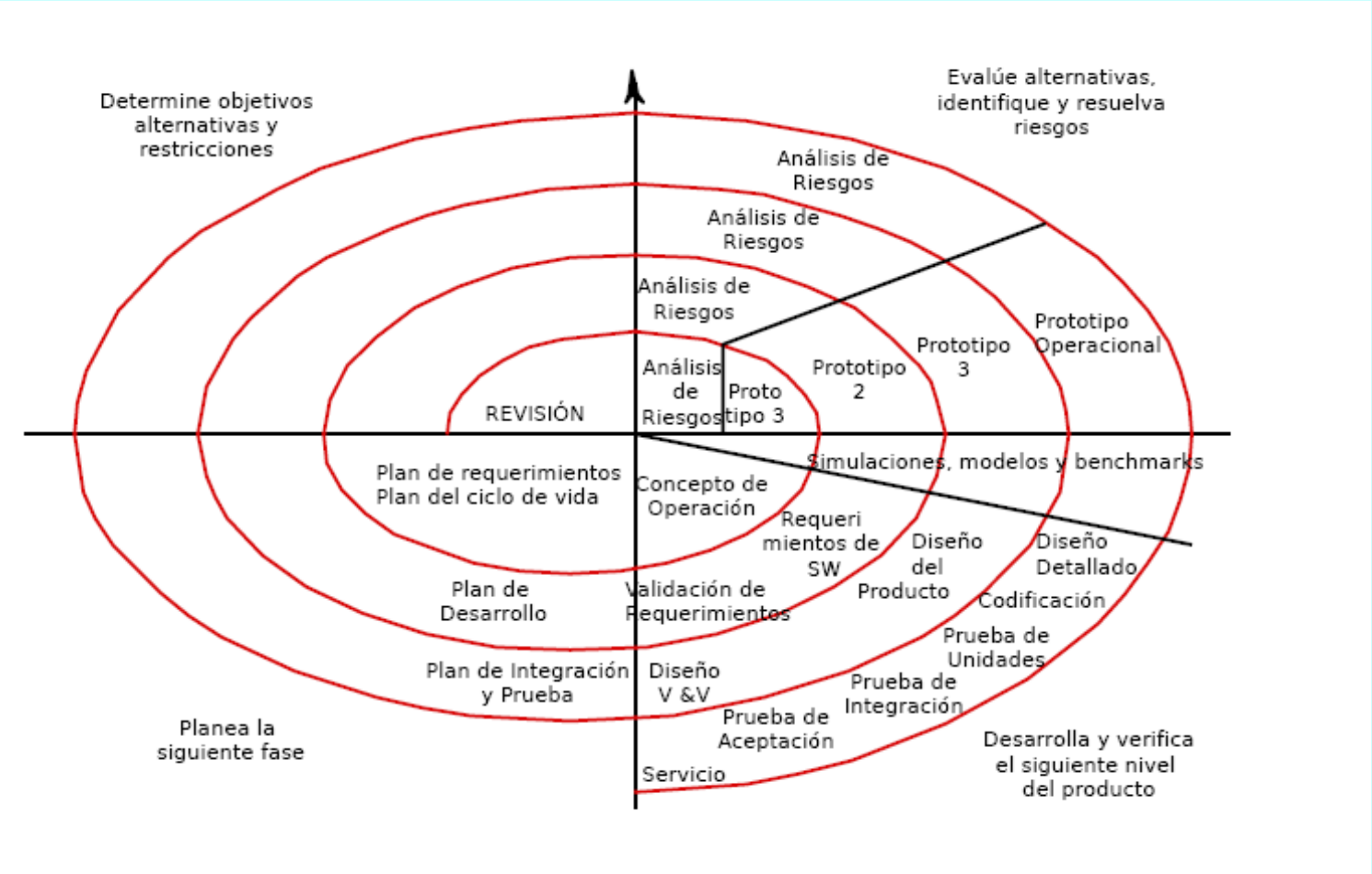
Programación extrema

- Un enfoque del desarrollo basado en el desarrollo y la entrega de los muy pequeños incrementos de funcionalidad.
- Código se basa en constante mejora, la participación de los usuarios en el equipo de desarrollo y programación en parejas.

Desarrollo en Espiral

- El proceso se representa como una espiral y no como una secuencia de actividades con marcha atrás.
- Cada bucle en la espiral representa una fase en el proceso.
- Fases no fijas, tales como las fases de diseño o especificación - bucles de la espiral se eligen en función de lo que se necesita.
- Riesgos se evalúan de forma explícita y se resuelven a través todo el proceso.

Espiral modelo el proceso de software



Sectores del modelo en espiral

- La fijación de objetivos
  - Los objetivos específicos de cada fase son identificados.
- Evaluación de riesgos y su reducción
  - Se evalúan los riesgos y las actividades puestas en marcha para reducir los principales riesgos.
- Desarrollo y validación
  - Un modelo de desarrollo para el sistema elegido es el que puede ser cualquiera de los modelos genéricos.
- Planificación
  - El proyecto es revisado y la próxima fase de la espiral es planeada.

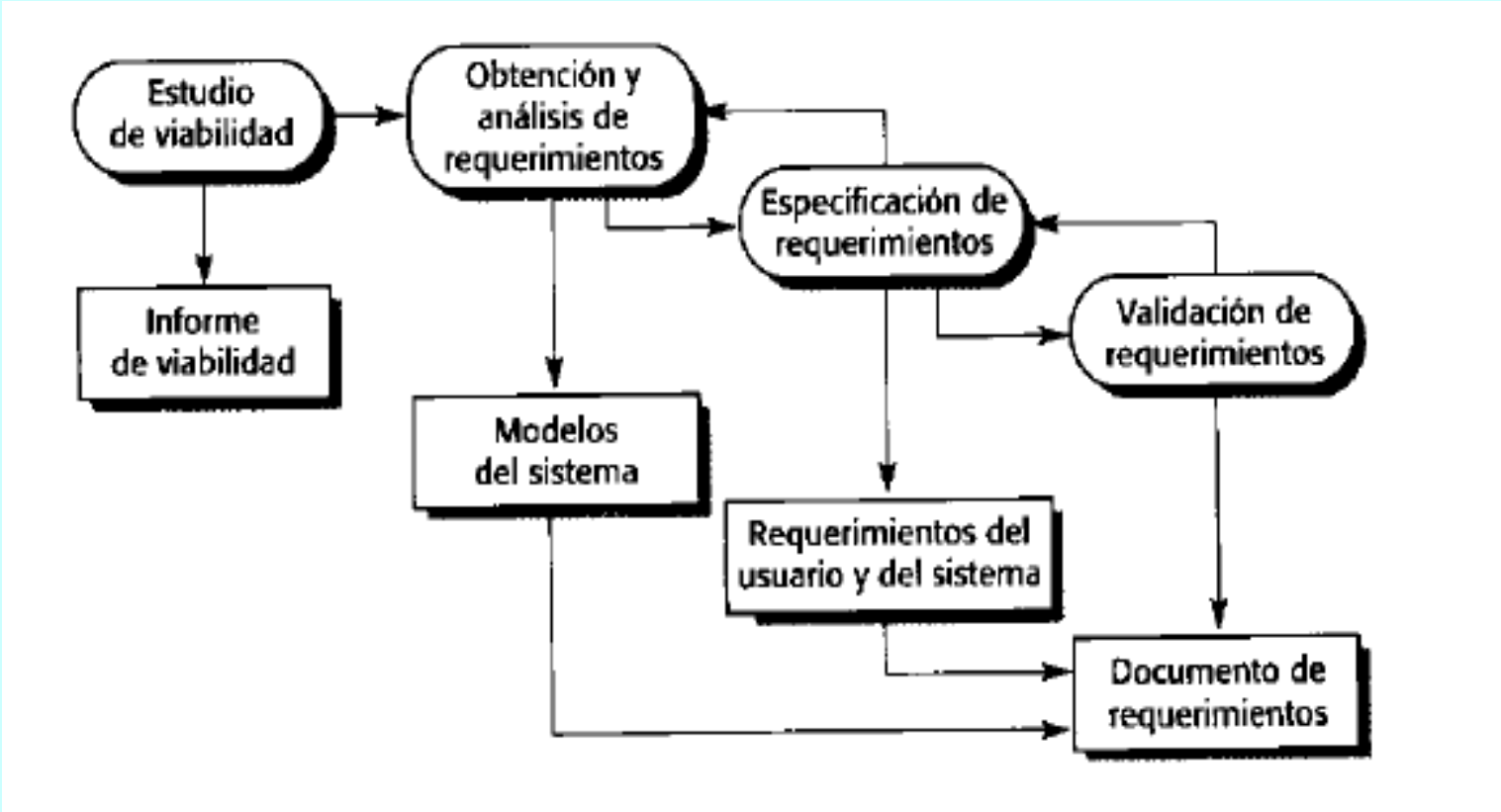
Actividades del proceso

- Especificación de software
- Diseño e implementación de software
- Validación de software
- Evolución del software

Especificación de software

- El proceso de establecer qué servicios son requeridos y las limitaciones en el funcionamiento del sistema y el desarrollo.
- Proceso de ingeniería de requerimientos
  - Estudio de viabilidad;
  - Obtención y análisis de requerimientos;
  - Especificación de los requerimientos;
  - Validación de requerimientos.

Proceso de ingeniería de requerimientos



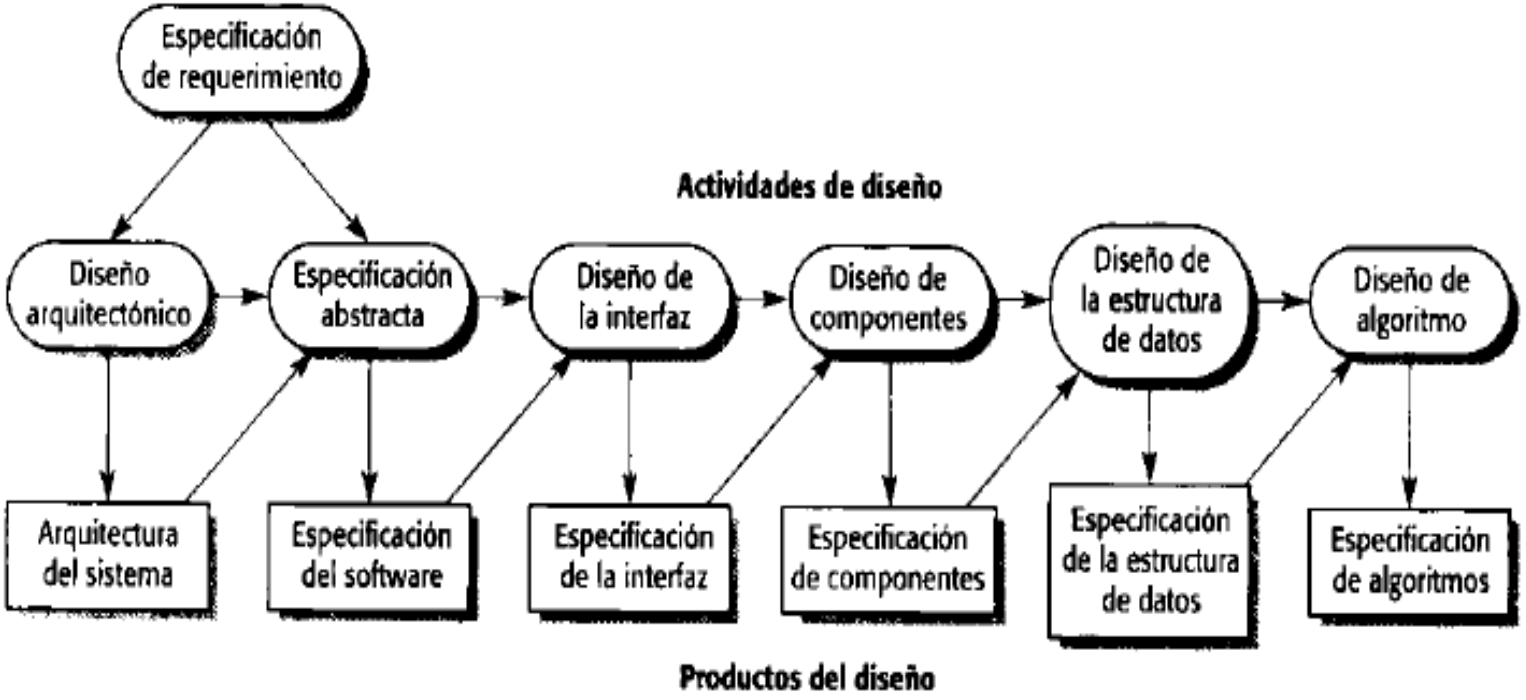
Diseño e implementación de software

- El proceso de convertir la especificación del sistema en un sistema ejecutable.
- Diseño de software
  - Diseñar una estructura de software que dé constancia de la especificación;
- Implementación
  - Traducir esta estructura en un programa ejecutable;
- Las actividades de diseño e implementación están estrechamente relacionadas y pueden ser interpoladas.

Actividades del proceso de Diseño

- Diseño arquitectónico
- Especificación abstracta
- Diseño de la interfaz
- Diseño de componentes
- Diseño de estructura de datos
- Diseño de algoritmos

El proceso de diseño de software



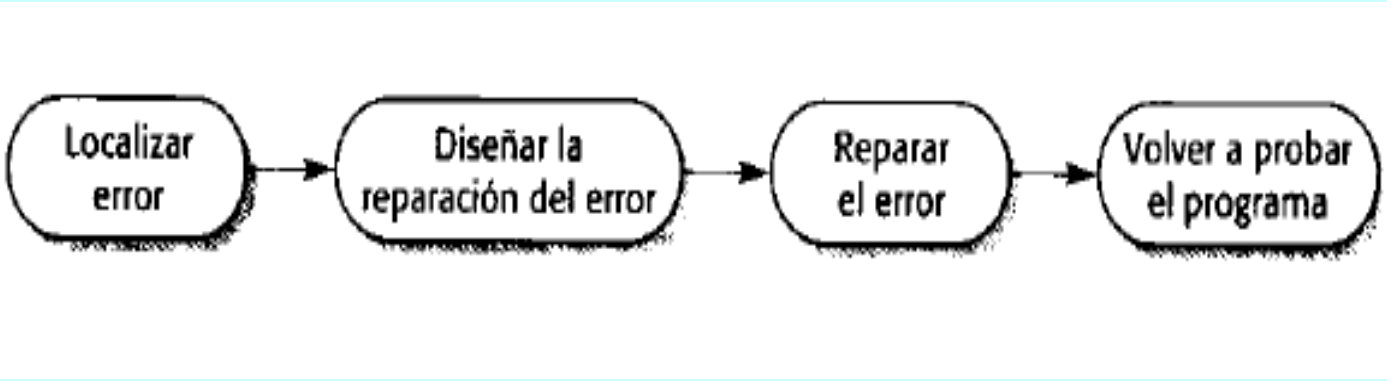
Métodos estructurados

- Enfoques sistemáticos para el desarrollo de un diseño de software.
- El diseño es por lo general documentado como un conjunto de modelos gráficos.
- Posibles modelos
  - Modelo de objetos;
  - Modelo de secuencia;
  - Modelo de transición de estados;
  - Modelo estructural;
  - Modelo de flujo de datos.

Programación y depuración

- Traducción de un diseño en un programa y la eliminación de errores de ese programa.
- La programación es una actividad personal - no hay ningún proceso de programación genérico.
- Los programadores deben acarrear alguna prueba del programa para descubrir defectos en el programa y eliminar esas fallas en el proceso de depuración.

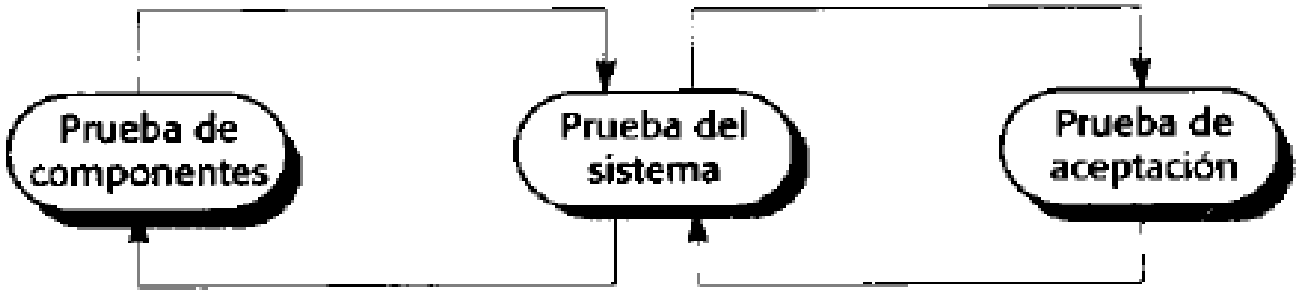
El proceso de depuración



Validación de software

- Verificación y validación (V & V) es para poner de manifiesto que un sistema cumple con su especificación y cumple los requerimientos del cliente.
- Implica el control y revisión de los procesos y pruebas del sistema.
- La prueba del sistema implica la ejecución del sistema con casos de prueba que se derivan de la especificación de los datos reales para ser procesados por el sistema.

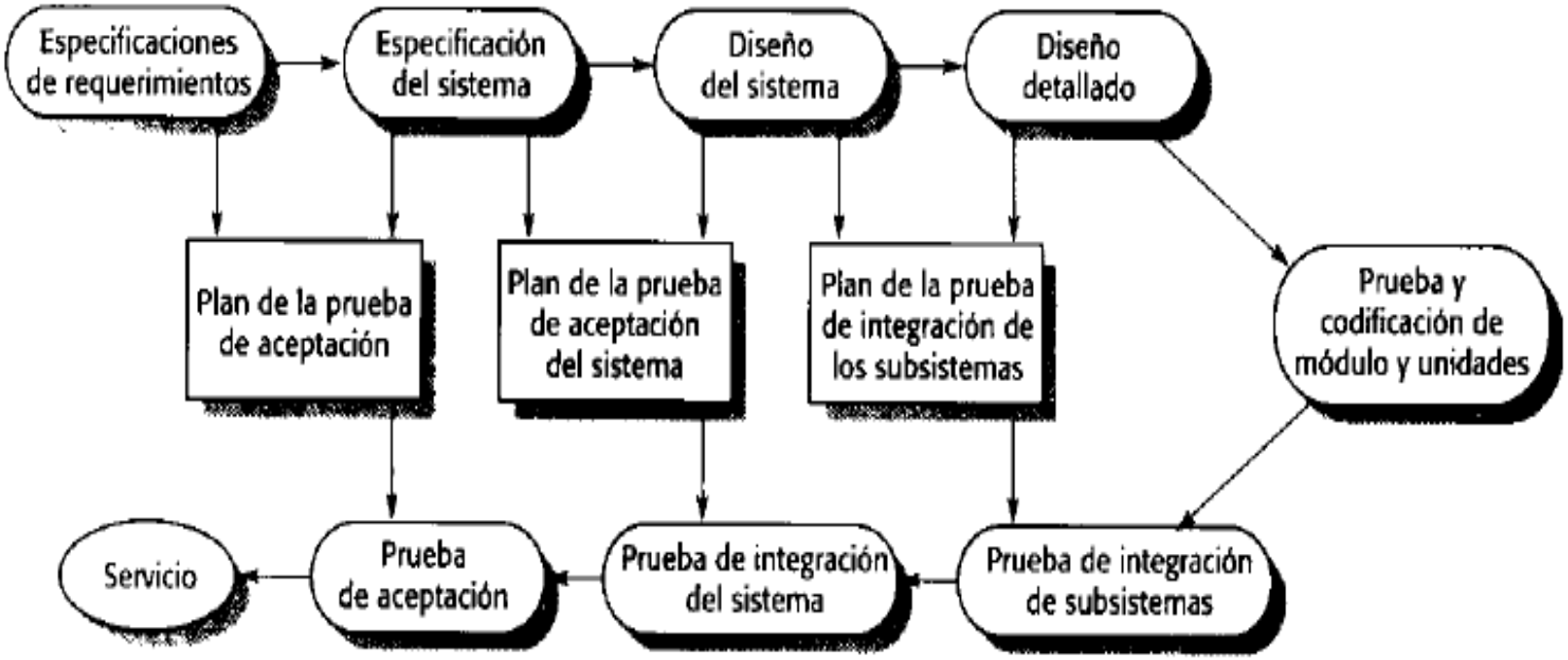
El proceso de pruebas



Etapas del proceso de pruebas

- Prueba de componentes o unidades
  - Cada uno de los componentes son probados por separado;
  - Componentes pueden ser funciones u objetos o agrupaciones coherentes de estas entidades.
- Pruebas del sistema
  - Prueba del sistema en su conjunto. Prueba de propiedades emergentes es particularmente importante.
- Pruebas de aceptación
  - Pruebas con los datos de los clientes para comprobar que el sistema cumple con los requerimientos del cliente.

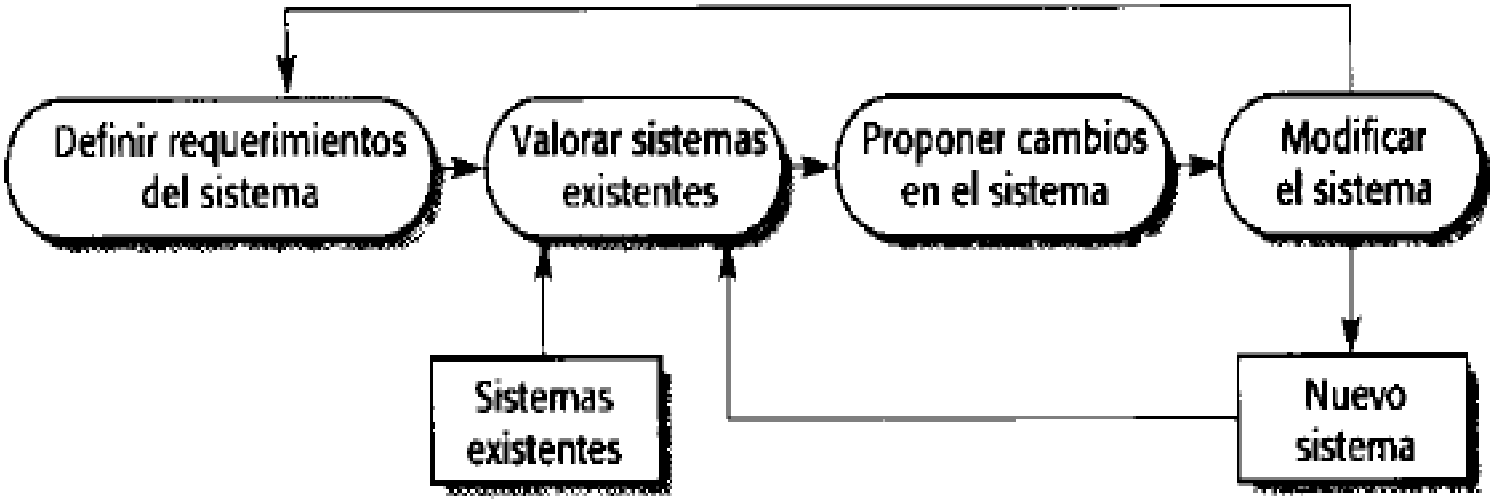
Fases de prueba



Evolución del software

- El software es inherentemente flexible y puede cambiar.
- Como los requerimientos cambian a través del cambio de las circustancias empresariales, el software que soporta la empresa también debe evolucionar y cambiar.
- Aunque ha habido una demarcación entre el desarrollo y evolución (mantenimiento) es cada vez más irrelevante como cada vez menos y menos sistemas son completamente nuevos.

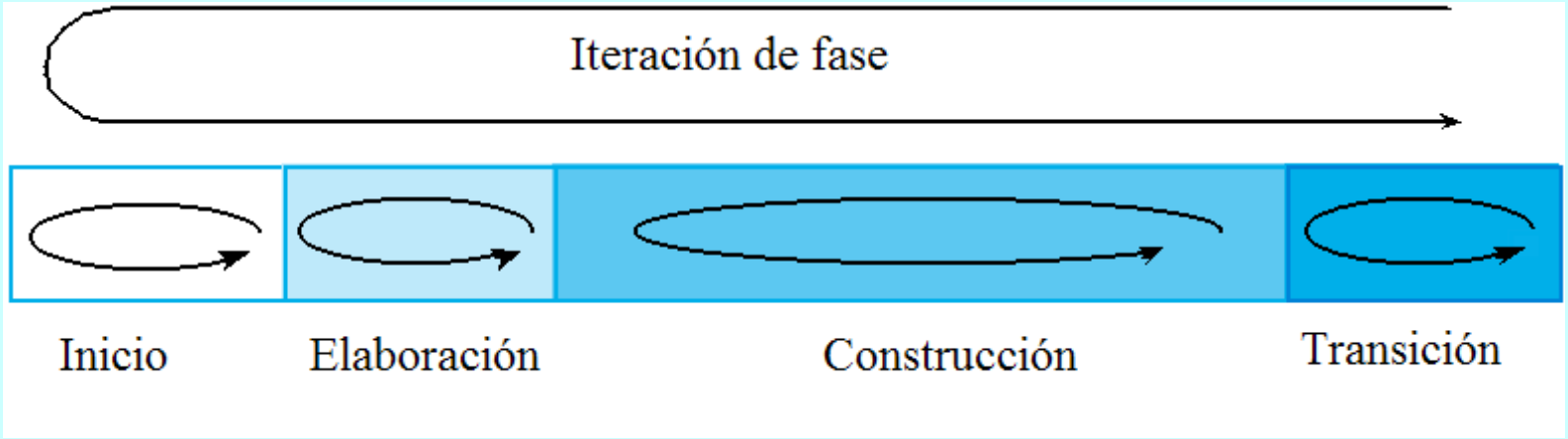
Evolución del sistema



El Proceso Unificado de Rational

- Un moderno modelo de proceso derivado de la labor sobre el UML y procesos asociados.
- Normalmente se describe a partir de 3 de perspectivas
  - Una perspectiva dinámica, que muestra las fases del modelo en el tiempo;
  - Una perspectiva estática que muestra las actividades de proceso;
  - Una perspectiva práctica que sugiere buenas prácticas a utilizar durante el proceso.

Fases del RUP



RUP - Fases

- Inicio
  - Establecer el modelo comercial para el sistema.
- Elaboración
  - Desarrollar una comprensión del dominio del problema y la arquitectura del sistema.
- Construcción
  - Diseño de sistemas, programación y pruebas.
- Transición
  - Desplegar el sistema en su entorno operativo.

Buenas prácticas en RUP

- Desarrollar software iterativamente
- Gestión de requerimientos
- Uso de arquitecturas basadas en componentes
- Modelos de software visuales
- Verificar la calidad del software
- Control de cambios al software

Los flujos de trabajo

Flujos de trabajo	Descripción
Modelado del negocio	Los procesos de negocio se modelan utilizando casos de uso del negocio
Requerimientos	Se definen los actores que interactúan con el sistema y se desarrollan casos de uso para modelar los requerimientos del sistema
Análisis y Diseño	Se crea y documenta un modelo del diseño utilizando modelos arquitectónicos, modelos de componentes, de objetos, y de secuencias
Implementación	Se implementan y estructuran en subsistemas los componentes del sistema. La generación automática de código de los modelos del diseño ayuda a acelerar este proceso
Prueba	Las pruebas son un proceso iterativo que se llevan a cabo conjuntamente con la implementación. A la finalización de la implementación tienen lugar las pruebas del sistema
Despliegue	Se crea una release del producto, se distribuye a los usuarios y se instala en su lugar de trabajo
Configuración y cambios de gestión	Este flujo de trabajo de soporte gestiona los cambios del sistema
Gestión del proyecto	Este flujo de trabajo de soporte gestiona el desarrollo del sistema
Entorno	Este flujo de trabajo se refiere a hacer herramientas software apropiadas disponibles para los equipos de desarrollo de software

Ingeniería de Software Asistida por Computadora

- Ingeniería de Software Asistida por Computadora (CASE) es un software para apoyar el desarrollo de software y procesos de evolución.
- Automatización de la actividad
  - Editores gráficos para el modelo de desarrollo del sistema;
  - Diccionario de datos para la gestión de las entidades de diseño;
  - Constructor GUI para la construcción de la interfaz de usuario gráfica;
  - Depuradores para ayudar a encontrar defectos en el programa;
  - Traductores automatizados para generar nuevas versiones de un programa.

Tecnología CASE

- La tecnología CASE ha dado lugar a mejoras significativas en el proceso del software. Sin embargo, estas no van acorde a la magnitud de las mejoras previstas alguna vez
  - Ingeniería de software requiere pensamiento creativo – esto no se automatiza fácilmente;
  - Ingeniería de software es una actividad de grupo y, para grandes proyectos, se dedica mucho tiempo a las interacciones del equipo. La tecnología CASE realmente no ayuda a ello.

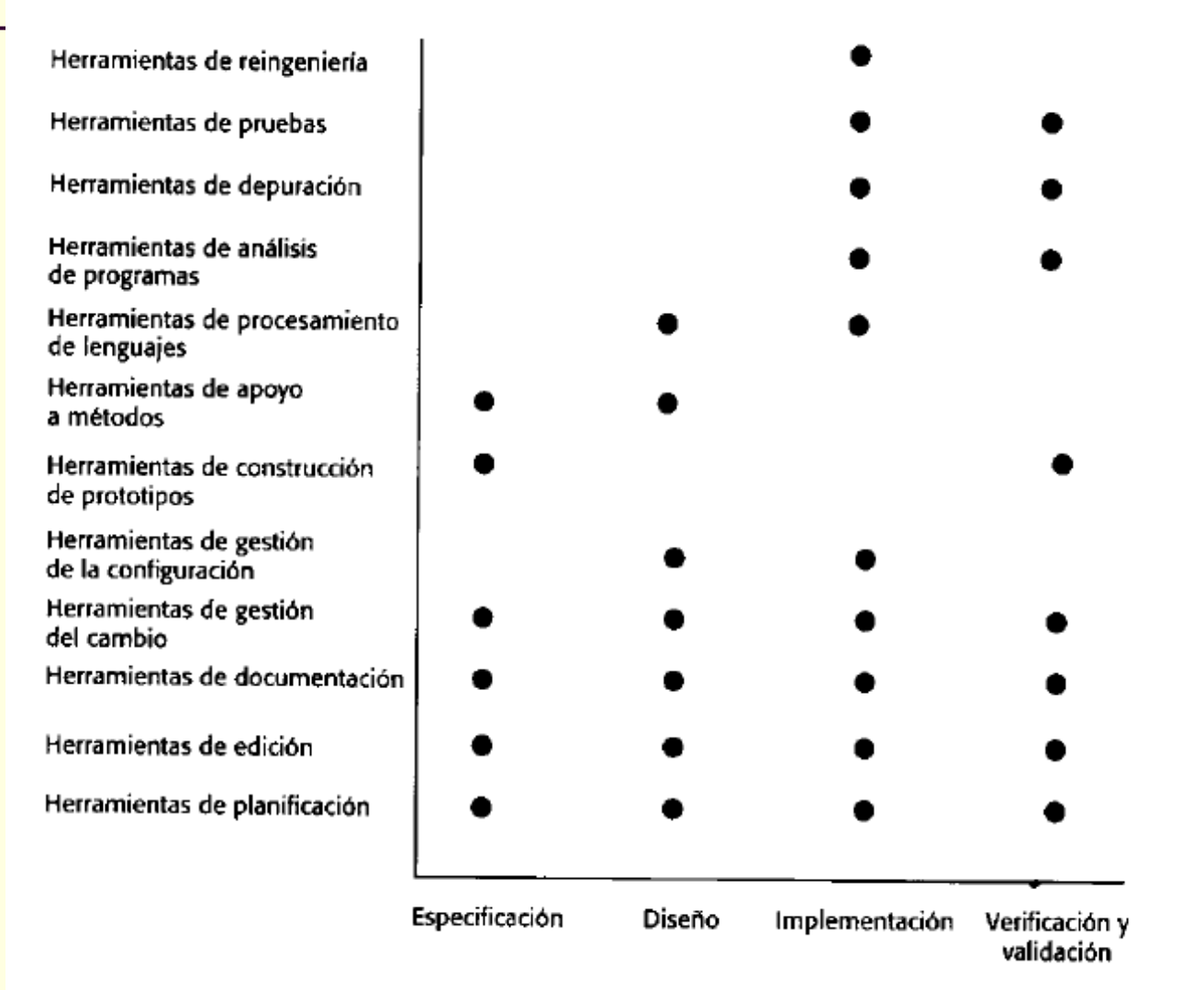
Clasificación CASE

- La clasificación nos ayuda a entender los diferentes tipos de herramientas CASE y su apoyo a actividades del proceso.
- Perspectiva funcional
  - Herramientas se clasifican de acuerdo a su función específica.
- Perspectiva de proceso
  - Herramientas se clasifican de acuerdo con las actividades de proceso que son compatibles.
- Perspectiva de integración
  - Herramientas se clasifican de acuerdo a su organización en unidades integradas.

Clasificación funcional de las herramientas CASE

Tipo herramienta	Ejemplos
Herramientas de planificación	Herramientas PERT, herramientas de estimación, hojas de cálculo.
Herramientas de edición	Editores de texto, editores de diagramas, procesadores de texto.
Herramientas de gestión del cambio	Herramientas de rastreo de requerimientos, sistemas de control de cambios.
Herramientas de gestión de la configuración	Sistema de gestión de las versiones, herramientas de construcción de sistemas.
Herramientas de construcción de prototipos	Lenguajes de muy alto nivel, generadores de interfaz de usuario.
Herramientas de apoyo a métodos	Editores de diseño. diccionarios de datos, generadores de código.
Herramientas de procesamiento de lenguajes	Compiladores, intérpretes.
Herramientas de análisis de programas	Generadores de referencias cruzadas, analizadores estáticos, analizadores dinámicos.
Herramientas de pruebas	Generadores de pruebas de datos, comparadores de archivos.
Herramientas de depuración	Sistemas de depuración interactiva.
Herramientas de documentación	Programas de diseño de páginas, editores de imágenes.
Herramientas de reingeniería	Sistemas de referencias cruzadas, sistemas reestructuración de programas.

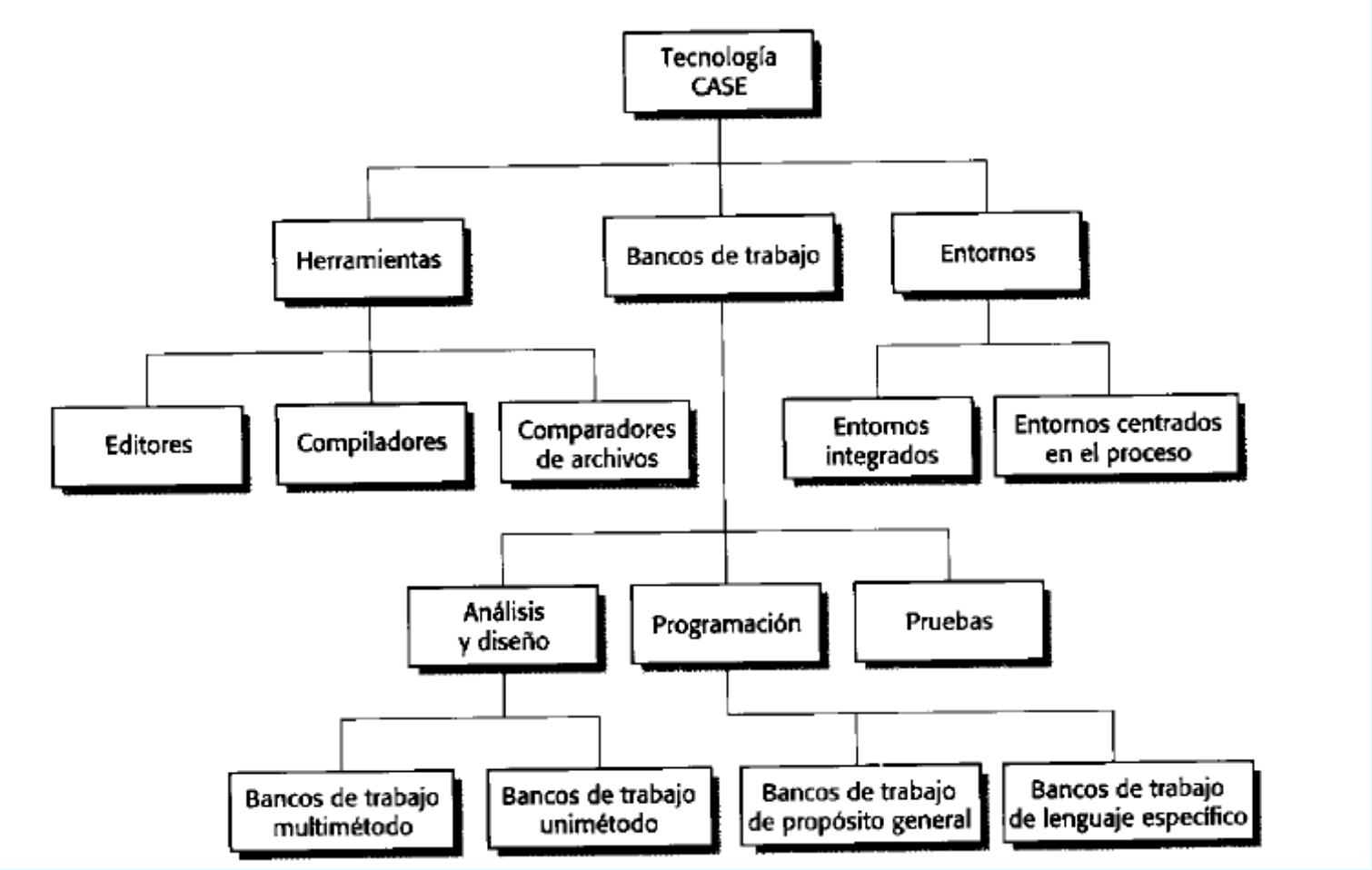
Clasificación basada en actividades de las herramientas CASE



Integración de herramientas CASE

- Herramientas
  - Tareas individuales de apoyo al proceso como control de coherencia de diseño, edición de texto, etc
- Bancos de trabajo
  - Apoyar una fase del proceso como la especificación o diseño, por lo general incluyen una serie de herramientas integradas.
- Entornos
  - Apoyo a la totalidad o una parte sustancial de todo un proceso del software. Normalmente incluyen varios bancos de trabajo integrados.

Herramientas, bancos de trabajo y entornos



Puntos clave

- Procesos de software son las actividades implicadas en la producción y la evolución de un sistema de software.
- Modelos de procesos software son representaciones abstractas de estos procesos.
- Actividades generales son la especificación, diseño e implementación, la validación y la evolución.
- Los modelos genéricos del proceso describen la organización de los procesos de software. Los ejemplos incluyen el modelo de cascada, desarrollo evolutivo e ingeniería de software basada en componentes.
- Los modelos de iteración de procesos describen el proceso del software como un ciclo de actividades.

Puntos clave

- Ingeniería de requerimientos es el proceso de desarrollo de una especificación de software.
- Diseño implementación comprenden la transformación de la especificación de requerimientos en un software ejecutable.
- La validación implica que el sistema cumple con las especificaciones y requerimientos del usuario.
- Evolución se refiere a la modificación del sistema después de que está en uso.
- El Proceso Unificado de Rational es un modelo de proceso genérico que separa las actividades de las fases que presenta.
- La tecnología CASE proporciona ayuda automatizada a los procesos de software.

TRABAJO PRÁCTICO N° 1

Unidad 1: Conceptos previos sobre ingeniería y proyectos de software

- **Objetivo:** Comparar las distintas herramientas CASE en función de la eficiencia y la utilidad que brinda para el desarrollo de software.
- **Tipo de Actividad:** búsqueda y análisis de información.
- **Modalidad de trabajo:** Individual o en pareja.
- **Evaluación:** presentación de archivo en el aula virtual según las consignas establecidas. Los criterios de evaluación serán:
  - Capacidad de análisis.
  - Coherencia en la justificación de la elección.
  - Cumplimiento de los aspectos formales del documento y utilización de lenguaje técnico apropiado.
- **Tiempo Asignado**: 1 hora de clase.

CONSIGNAS

- Fecha de realización: ú**ltima hora de clase**
- Forma de entrega: **subir archivo con nombre "TP1-Apellido/s" al aula virtual**
- Formato:
  - Papel A4
  - Márgenes: Superior = 3 cm, Inferior = 3cm, Derecho = 3 cm, Izquierdo = 3cm.
  - Sangría = no Justificado = si
  - Letra arial. Títulos 14 subrayado. Texto: 12. Interlineado: 1,5

Tarea:

- Buscar en internet métodos de diseño de software.
- Analizar los diferentes métodos y realizar una comparación entre ellos.
- Presentar en el informe el detalle de las características técnicas, una planilla de comparación, y elegir el que considere más conveniente justificando objetivamente su decisión.

Sistemas socio-técnicos

Objetivos

- Explicar lo que es un sistema socio-técnico y la distinción entre este y un sistema técnico informático
- Introducir el concepto de propiedades emergentes del sistema, tales como la fiabilidad y la seguridad
- Explicar las actividades implicadas en el proceso de la ingeniería de sistemas
- Explicar por qué el contexto organizacional de un sistema afecta a su diseño y uso
- Examinar los “sistemas legados” y el por qué estos son críticos para muchas empresas

Tópicos Expuestos

- Propiedades emergentes del sistema
- Ingeniería de sistemas
- Organizaciones, personas y sistemas informáticos
- Sistemas heredados

¿Qué es un sistema?

- Una colección intencionada de componentes interrelacionados que trabajan juntos para lograr objetivos comunes.
- Un sistema puede incluir el software, hardware mecánico, eléctrico y electrónico y ser manejado por personas.
- Los componentes del sistema dependen de otros Componentes del sistema
- Las propiedades y el comportamiento de los componentes del sistema están inextricablemente entremezclados

Categorías de Sistemas

- Sistemas técnico - informáticos
  - Sistemas que incluyen hardware y software, pero donde los operadores y los procesos operativos normalmente no son considerados como parte del sistema. El sistema no es auto-consciente.
- Sistemas socio-técnicos
  - Sistemas que incluyen sistemas técnicos, y también procesos operativos y personas que usan e interactúan con el sistema técnico. Los sistemas socio-técnicos se rigen por las políticas y normas organizacionales.

Características del Sistema socio-técnico

- Propiedades emergentes
  - Propiedades del sistema de un todo que dependen de los componentes del sistema y sus relaciones.
- No-determinista
  - No siempre producen el mismo resultado cuando se presenta la misma entrada, porque el comportamiento de los sistemas es parcialmente dependiente de los operadores humanos.
- Complejas relaciones con los objetivos organizacionales
  - La medida en que el sistema organizacional respalda los objetivos no sólo depende del propio sistema.

Propiedades emergentes

- Propiedades del sistema en su conjunto y no las propiedades que se pueden derivar de las propiedades de los componentes de un sistema
- Las propiedades emergentes son una consecuencia de las relaciones entre los componentes del sistema
- Por lo tanto, sólo pueden ser evaluados y medidos una vez que los componentes se han integrado al sistema

Ejemplos de propiedades emergentes

Propiedad	Descripción
Volumen	El volumen de un sistema (el espacio total ocupado) varía dependiendo de cómo estén ordenados y conectados los montajes de los componentes.
Fiabilidad	La fiabilidad del sistema depende de la fiabilidad de los componentes, pero interacciones inesperadas pueden causar nuevos tipos de fallos y, por lo tanto, afectar a la fiabilidad del sistema
Protección	La protección del sistema (su capacidad para resistir ataques) es una propiedad compleja que no se puede medir fácilmente, los ataques pueden ser ideados de forma que no fueron predichos y así vencer las protecciones incorporadas
Reparabilidad	Esta propiedad refleja hasta qué punto resulta fácil arreglar un problema con el sistema una vez que ha sido descubierto. Depende de la posibilidad de diagnosticar el problema, acceder a los componentes defectuosos y modificarlos o reemplazarlos
Usabilidad	Esta propiedad refleja la facilidad de usar el sistema. Depende de los componentes técnicos del sistema, sus operarios y su entorno de operaciones.

Tipos de propiedades emergentes

- Propiedades funcionales
  - Estas aparecen cuando todas las partes de un sistema trabajan juntas para lograr algún objetivo. Por ejemplo, una bicicleta tiene la propiedad funcional de ser un dispositivo de transporte una vez que se ha montado a partir de sus componentes.
- Propiedades emergentes no funcionales
  - Ejemplos de ellas son la fiabilidad, el rendimiento, la protección y la seguridad. Estos se relacionan con el comportamiento del sistema en su entorno operativo. A menudo son críticos para sistemas informáticos pues la falta de alcanzar un cierto nivel definido mínimo en estas características puede

Requerimientos del software

Objetivos

- Introducir los conceptos de requerimientos del usuario y sistema
- Describir los requerimientos funcionales y no funcionales
- Explicar la forma en que los requerimientos de software pueden ser organizados en un documento de requerimientos de software

Tópicos expuestos

- Requerimientos funcionales y no funcionales
- Requerimientos del usuario
- Requerimientos del sistema
- Especificación de la interfaz
- El documento de requerimientos de software

Ingeniería de requerimientos

- El proceso de establecimiento de los servicios que el cliente requiere de un sistema y las limitaciones con las que opera y se desarrolla.
- Los requerimientos son la descripción de los servicios del sistema y las limitaciones que se generan durante el proceso de ingeniería de requerimientos.

Qué es un requerimiento?

- Puede ir desde una declaración de un servicio con un alto nivel de abstracción o de una limitación del sistema a una detallada especificación funcional formal.
- Esto es inevitable, ya que los requerimientos pueden servir una doble función
  - Puede ser la base para una oferta para un contrato - por lo tanto debe estar abierto a la interpretación;
  - Puede ser la base del contrato en sí - por lo tanto, debe definirse en detalle;
  - Ambas declaraciones pueden llamarse requerimientos.

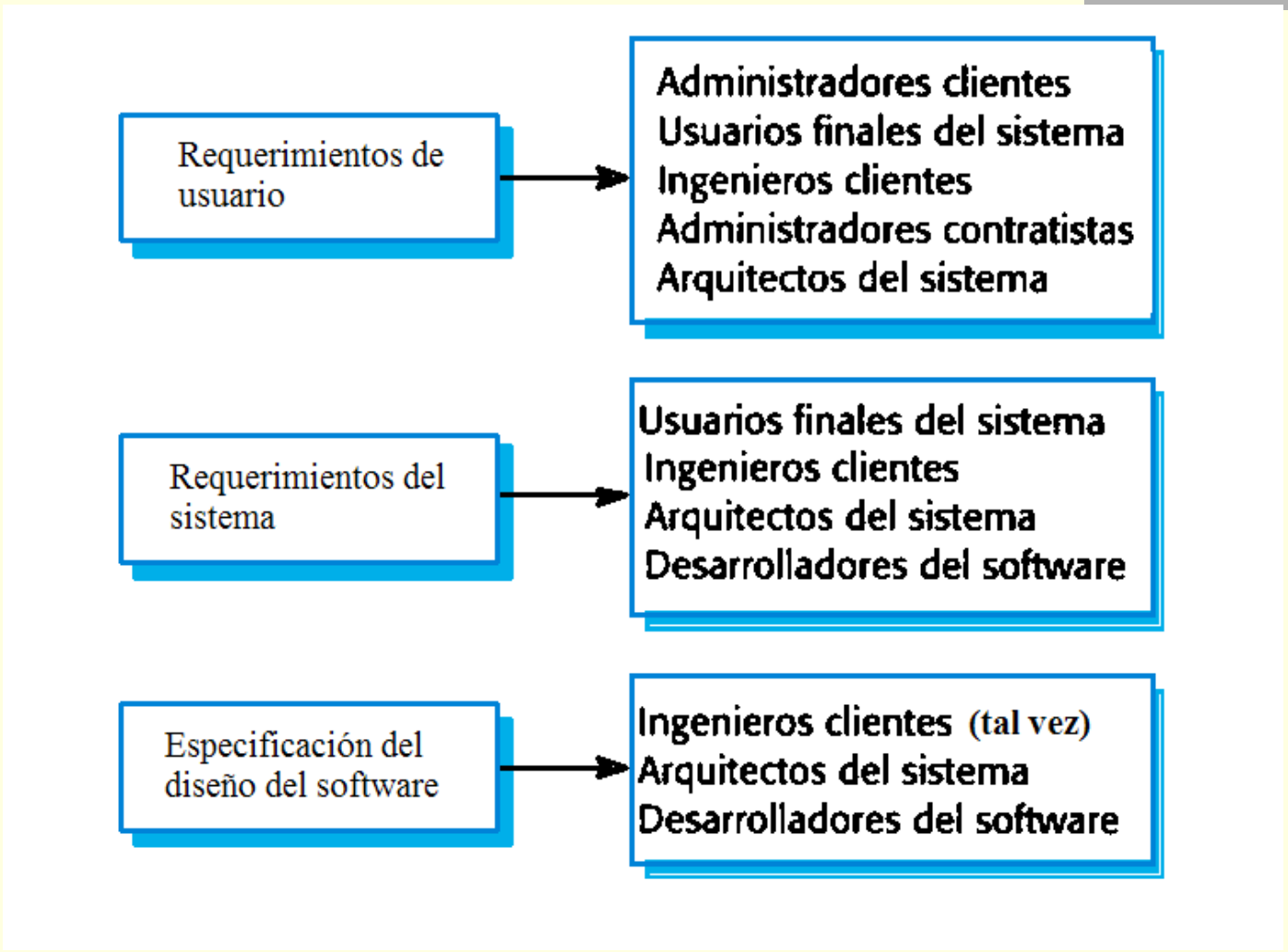
Abstracción de requerimientos (Davis)

Si una compañía desea establecer un contrato para un proyecto de desarrollo de software grande, debe definir sus necesidades de una forma suficientemente abstracta para establecer a partir de ella una solución. Los requerimientos deben redactarse de tal forma que varios contratistas pueden licitar el contrato, ofreciendo, quizás, formas diferentes de cumplir las necesidades de los clientes en la organización. Una vez que el contrato se asigna, el contratista debe redactar una definición del sistema para el cliente más detalladamente de forma que éste comprenda y pueda validar lo que hará el software. Ambos documentos se pueden denominar documento de requerimientos para el sistema.

Tipos de requerimientos

- Requerimientos de usuario
  - Declaraciones en lenguaje natural y los esquemas de los servicios que proporciona el sistema y sus limitaciones operacionales. Escrito para los clientes.
- Requerimientos del sistema
  - Un documento estructurado que establece la descripción detallada de las funciones del sistema, los servicios y las limitaciones operacionales. Define lo que debe aplicarse, de manera que puede ser parte de un contrato entre el cliente y el

Requisitos de los lectores



Requerimientos funcionales y no funcionales

- Los requerimientos funcionales
  - Declaraciones de los servicios que debe proporcionar el sistema, la forma en que el sistema debe reaccionar a las entradas y la forma en que el sistema debe comportarse en situaciones particulares.
- Requerimientos no funcionales
  - limitaciones en los servicios o funciones ofrecidas por el sistema como de tiempo, limitaciones en el proceso de desarrollo, normas, etc
- Requerimientos del dominio
  - Requerimientos que se derivan del dominio de aplicación del sistema y que reflejan las características de ese dominio.

Requisitos estándar IEEE

- Define una estructura genérica para un documento de requerimientos que debe ser instanciada para cada sistema específico.
  - Introducción.
  - Descripción general.
  - Requerimientos específicos.
  - Apéndices.
  - Índice.

Estructura de un documento de requerimientos

- Prefacio
- Introducción
- Glosario
- Definición de requerimientos del usuario
- Arquitectura del sistema
- Especificación de requerimientos del sistema
- Modelos del sistema
- Evolución del sistema
- Apéndices
- Indice

Puntos clave

- Los requerimientos determinan lo que debe hacer el sistema y definen las restricciones en su funcionamiento e implementación.
- Los requerimientos funcionales establecen los servicios que el sistema debe proporcionar.
- Los requerimientos no funcionales restringen el sistema en desarrollo y el proceso de desarrollo que se debe utilizar.
- Los requerimientos de usuario son declaraciones de alto nivel de lo que el sistema debe hacer. Los requerimientos de usuario deben ser escritos utilizando el lenguaje natural, tablas y diagramas.

Puntos clave

- Los requerimientos del sistema tienen por objeto comunicar las funciones o servicios que el sistema debe proporcionar.
- Un documento de requerimientos de software es una declaración de los requerimientos del sistema.
- La estándar de la IEEE es un punto de partida útil para la definición de estándares de requerimientos más detallados.

Diseño Arquitectónico

Objetivos

- Introducir el diseño arquitectónico y discutir su importancia
- Explicar el diseño arquitectónico y las decisiones que tienen que hacerse
- Introducir tres estilos arquitectónicos, que abarcan la organización, la descomposición y el control
- Debatir arquitecturas de referencia que se utilizan para comunicar y comparar arquitecturas

Tópicos expuestos

- Decisiones de diseño arquitectónico
- Sistema de organización
- Descomposición estilos
- Control de estilos
- Arquitecturas de referencia

Arquitectura de software

- El proceso de diseño para la identificación de los sub-sistemas que componen un sistema y el marco para el sub-sistema de control y comunicación es el diseño arquitectónico.
- El resultado de este proceso de diseño es una descripción de la arquitectura de software.

Diseño arquitectónico

- Una fase temprana del proceso de diseño del sistema.
- Representa el vínculo entre los procesos de especificación y diseño.
- Suelen llevarse a cabo en paralelo con las actividades de algunas especificaciones.
- Se trata de identificar los principales componentes del sistema y sus comunicaciones.

Ventajas de la arquitectura explícita

- Comunicación entre los stakeholders
  - La arquitectura puede ser utilizada como un foco de discusión del sistema por los stakeholders.
- Análisis del sistema
  - Significa que el análisis de si el sistema puede hacer frente a sus requerimientos no funcionales es posible.
- Reutilización a gran escala
  - La arquitectura puede ser reutilizable a través de una variedad de sistemas.

La arquitectura y características del sistema

- Rendimiento
  - Localizar y reducir al mínimo las operaciones críticas de comunicaciones. Gran uso en lugar de componentes de grano fino.
- Seguridad
  - Use una arquitectura con activos críticos en las capas interiores.
- Protección
  - La localización de las características esenciales para la seguridad en un pequeño número de sub-sistemas.
- Disponibilidad
  - Incluir componentes redundantes y los mecanismos de tolerancia a fallos.
- Mantenibilidad
  - Uso de grano fino, los componentes reemplazables.

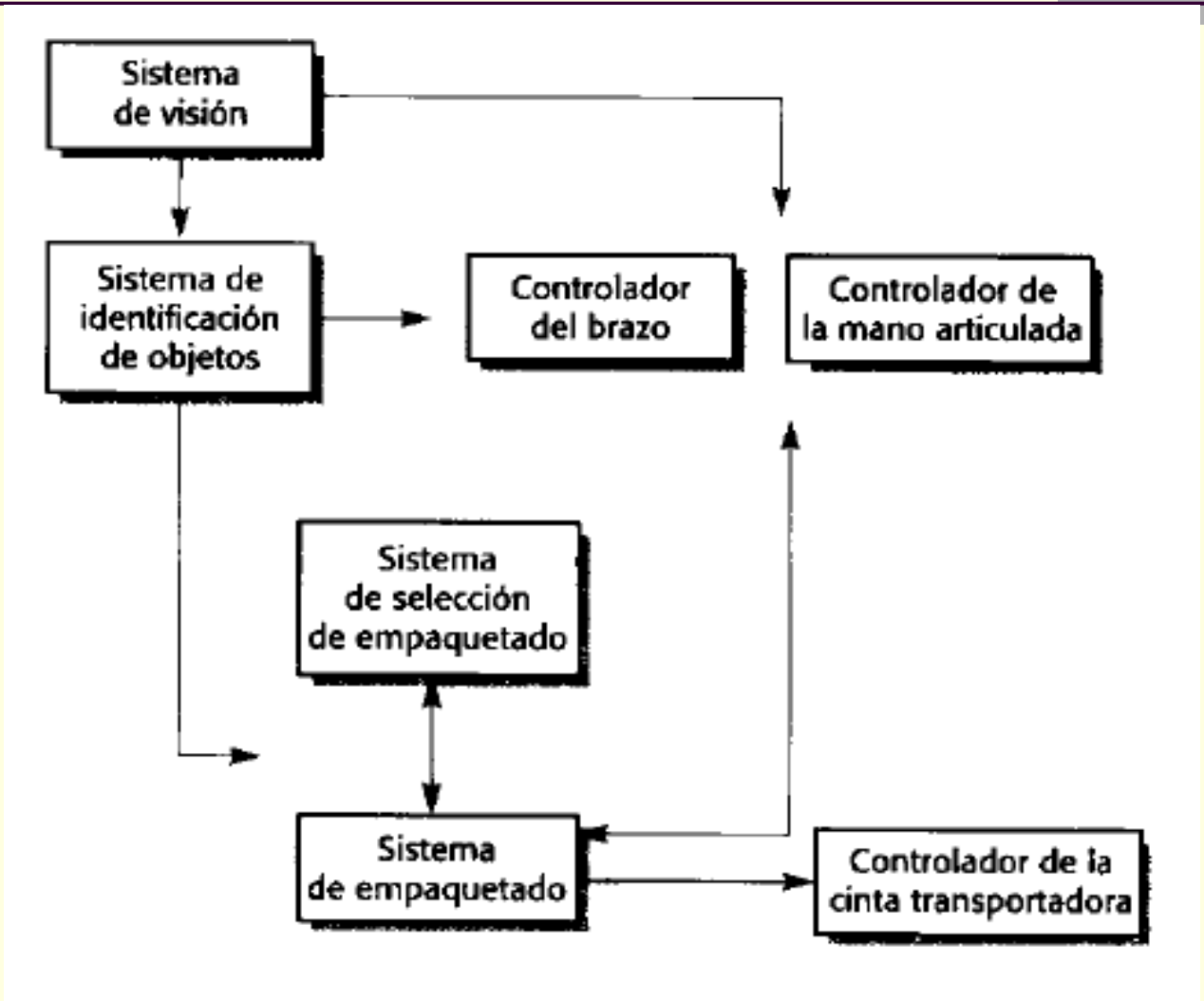
Conflictos Arquitectónicos

- Uso de componentes mejora de gran manera el rendimiento de grano, pero hace más difícil el mantenimiento.
- Hace más difícil la introducción de los datos redundantes, pero mejora la disponibilidad de seguridad.
- La localización de las características relacionadas con la seguridad más medios de comunicación por lo general degradan el rendimiento.

Estructuración del sistema

- Concerniente a la descomposición del sistema en sub-sistemas.
- El diseño arquitectónico se expresa normalmente como un diagrama de bloques que presentan un panorama general de la estructura del sistema.
- Modelos más específicos muestran cómo los sub-sistemas comparten los datos, se distribuyen y la interfaz con los demás también pueden ser desarrollados.

Sistema de control de robot de embalaje



Box y diagramas

- Muy abstracto - que no muestran la naturaleza de los componentes ni las relaciones de las propiedades visibles externamente de los sub-sistemas.
- Sin embargo, útil para la comunicación con las partes interesadas y para la planificación de proyectos.

Decisiones de diseño arquitectónico

- Diseño arquitectónico es un proceso creativo, por lo que el proceso es diferente dependiendo del tipo de sistema que se está desarrollado.
- Sin embargo, es común una serie de decisiones, en todos los procesos de diseño.

Decisiones de diseño arquitectónico

- Existe una arquitectura de aplicaciones genéricas que se pueden utilizar?
- Cómo se distribuirá el sistema?
- Qué estilos arquitectónicos son apropiados?
- Qué enfoque se utilizará para la estructura del sistema?
- Cómo el sistema se descompone en módulos?
- Qué estrategia de control se debe utilizar?
- Cómo el diseño arquitectónico se evaluará?
- Cómo debe ser documentada la arquitectura?

Reutilización de la arquitectura

- Sistemas en el mismo dominio a menudo tienen arquitecturas similares que reflejan conceptos del dominio.
- La aplicación de líneas de producción se construye en torno a un núcleo con arquitectura particular, con variantes que satisfagan las necesidades del cliente.
- Arquitecturas de aplicación se tratan en el capítulo 13 y las líneas de producción en el capítulo 18.

Estilos arquitectónicos

- El modelo arquitectónico de un sistema puede ajustarse a un modelo genérico o estilo arquitectónico.
- La conciencia de estos estilos puede simplificar el problema de la definición de arquitecturas de sistemas.
- Sin embargo, la mayoría de los grandes sistemas son heterogéneos y no siguen un mismo estilo arquitectónico.

Modelos arquitectónicos

- Utilizarse para documentar un diseño arquitectónico.
- Modelo estructural estático, que muestra los principales componentes del sistema.
- Modelo de proceso dinámico que muestra el modelo de proceso de la estructura del sistema.
- Modelo de interfaz que define las interfaces de sub-sistemas.
- Modelo de relaciones, como un modelo de flujo de datos que muestra las relaciones de sub-sistemas.
- Modelo de distribución que muestra cómo los sub-sistemas se distribuyen a través de computadoras.

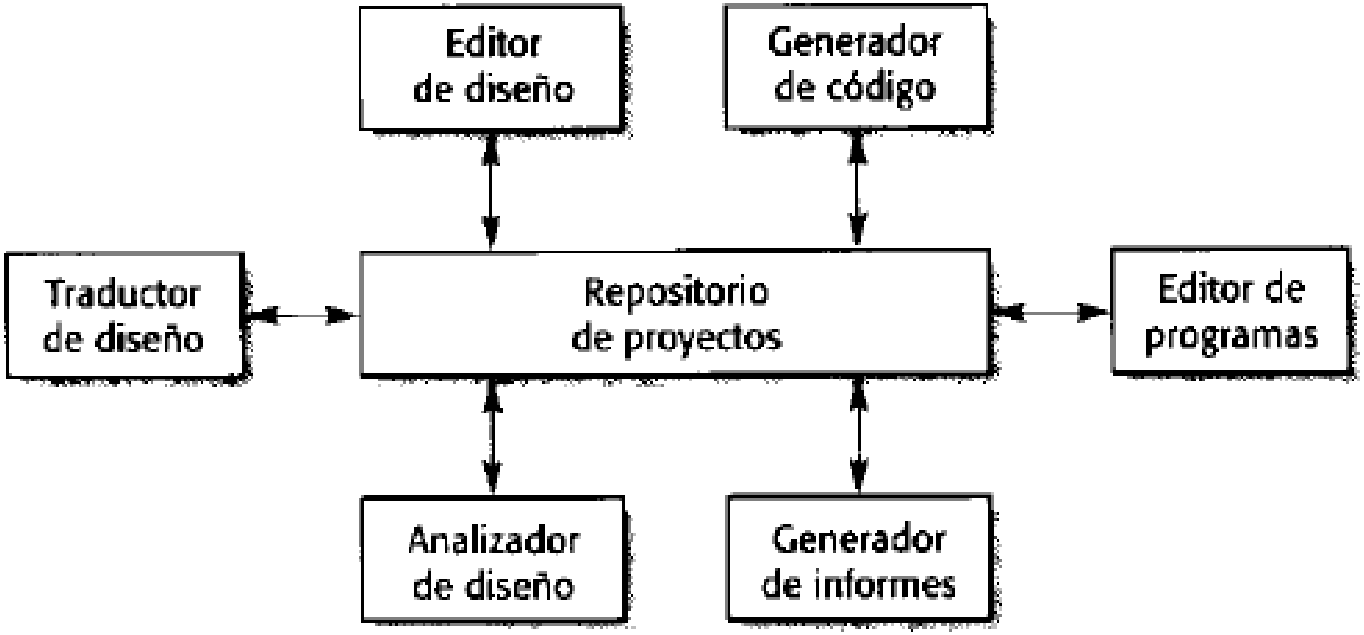
Sistema de organización

- Refleja la estrategia básica que se utiliza para estructurar un sistema.
- Tres estilos de organización son ampliamente utilizados:
  - Una reposición de datos compartida;
  - Servicios compartidos y servidores de estilo;
  - Una máquina abstracta o estilo de capas.

El modelo repositorio

- Sub-sistemas de intercambio de datos. Esto puede hacerse de dos maneras:
  - Datos compartidos se lleva a cabo en un repositorio o base de datos central y puede ser visitada por todos los sub-sistemas;
  - Cada sub-sistema mantiene su propia base de datos y pasa datos explícitamente a otros subsistemas.
- Cuando grandes cantidades de datos sean compartidos, el modelo de repositorio compartido es más comúnmente utilizado.

Arquitectura de herramientas CASE



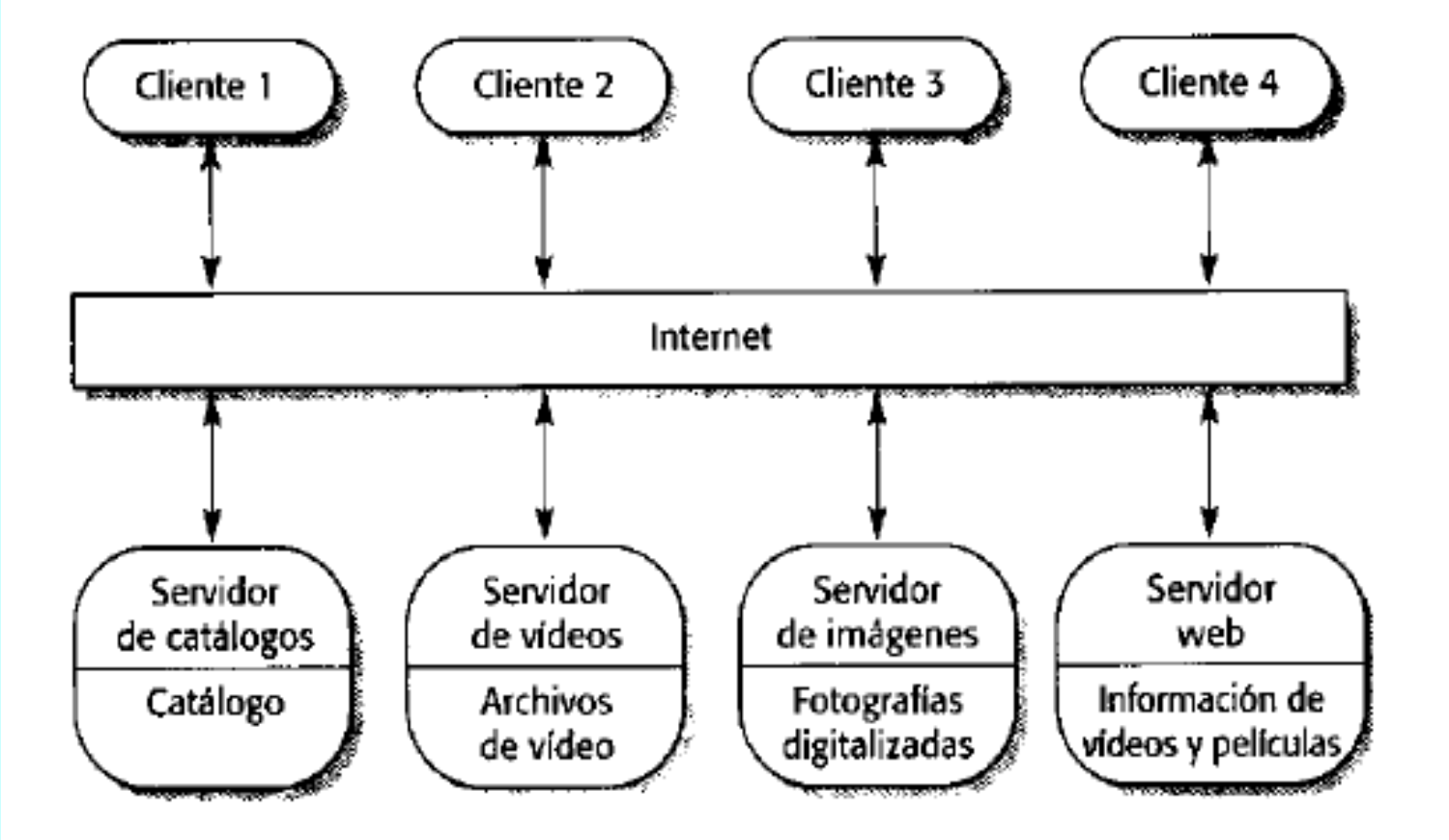
Modelo Repositorio Características

- Ventajas
  - Manera eficaz de compartir grandes cantidades de datos;
  - Sub-sistemas no tienen por qué preocuparse de cómo los datos se producen, por ejemplo, la gestión centralizada copia de seguridad, seguridad, etc
  - Un modelo a compartir se publica como el esquema del repositorio.
- Desventajas
  - Sub-sistemas deben ponerse de acuerdo sobre un modelo repositorio de datos. Inevitablemente, un compromiso;
  - La evolución de datos es difícil y costosa;
  - No hay lugar para las políticas de gestión específicas;

Modelo cliente-servidor

- Sistema distribuido que muestra cómo el modelo de datos y procesamiento se distribuye a través de una gama de componentes.
- Conjunto de servidores independientes que ofrecen servicios específicos, tales como la impresión, gestión de datos, etc
- Conjunto de clientes que piden a éstos los servicios.
- Red que permite a los clientes acceder a los servidores.

biblioteca de imágenes y películas



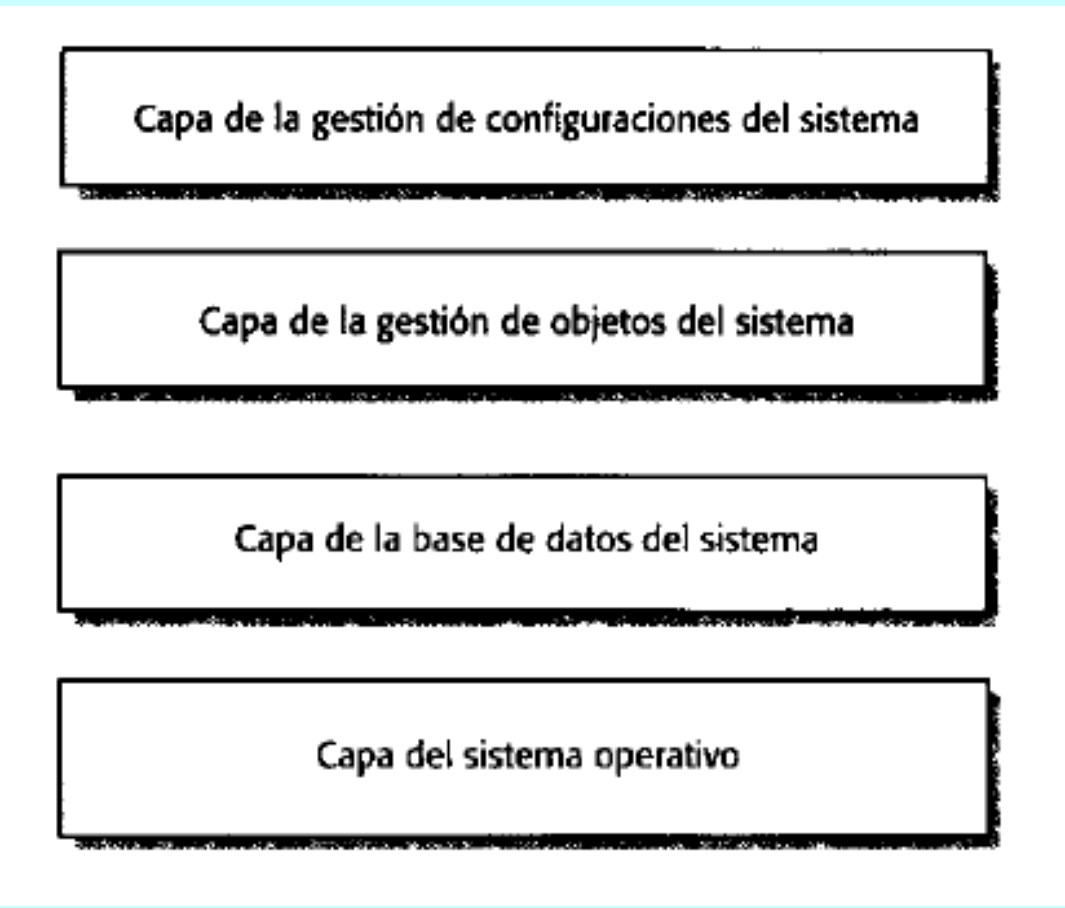
Cliente-servidor características

- Ventajas
  - Distribución de datos es sencilla;
  - Hace un uso eficaz de los sistemas en red. Puede requerir hardware más barato;
  - Fácil añadir nuevos servidores o actualizar los servidores existentes.
- Desventajas
  - No hay un modelo de datos compartidos, así que los sub-sistemas utilizan diferentes datos de la organización. Intercambio de datos puede ser ineficaz;
  - Redundantes en la gestión de cada servidor;
  - No hay registro central de nombres y servicios - que puede ser difícil de averiguar qué servidores y servicios están disponibles.

Modelo de máquina abstracta (capas)

- Se utiliza para modelar la interacción de sub-sistemas.
- Organiza el sistema en un conjunto de capas (o máquinas abstractas) cada uno de los cuales provee un conjunto de servicios.
- Apoya el desarrollo gradual de sub-sistemas en diferentes capas. Cuando una capa cambia, sólo la capa adyacente se ve afectada.

Versión del sistema de gestión



Estilos de descomposición modular

- Los estilos de la descomposición de sub-sistemas en módulos.
- No hay distinción rígida entre la organización y la descomposición modular del sistema.

Sub-sistemas y módulos

- Un sub-sistema es un sistema en su propio derecho cuyo funcionamiento es independiente de los servicios prestados por otros subsistemas.
- Un módulo es un componente del sistema que proporciona servicios a otros componentes, pero normalmente no se considera como un sistema separado.

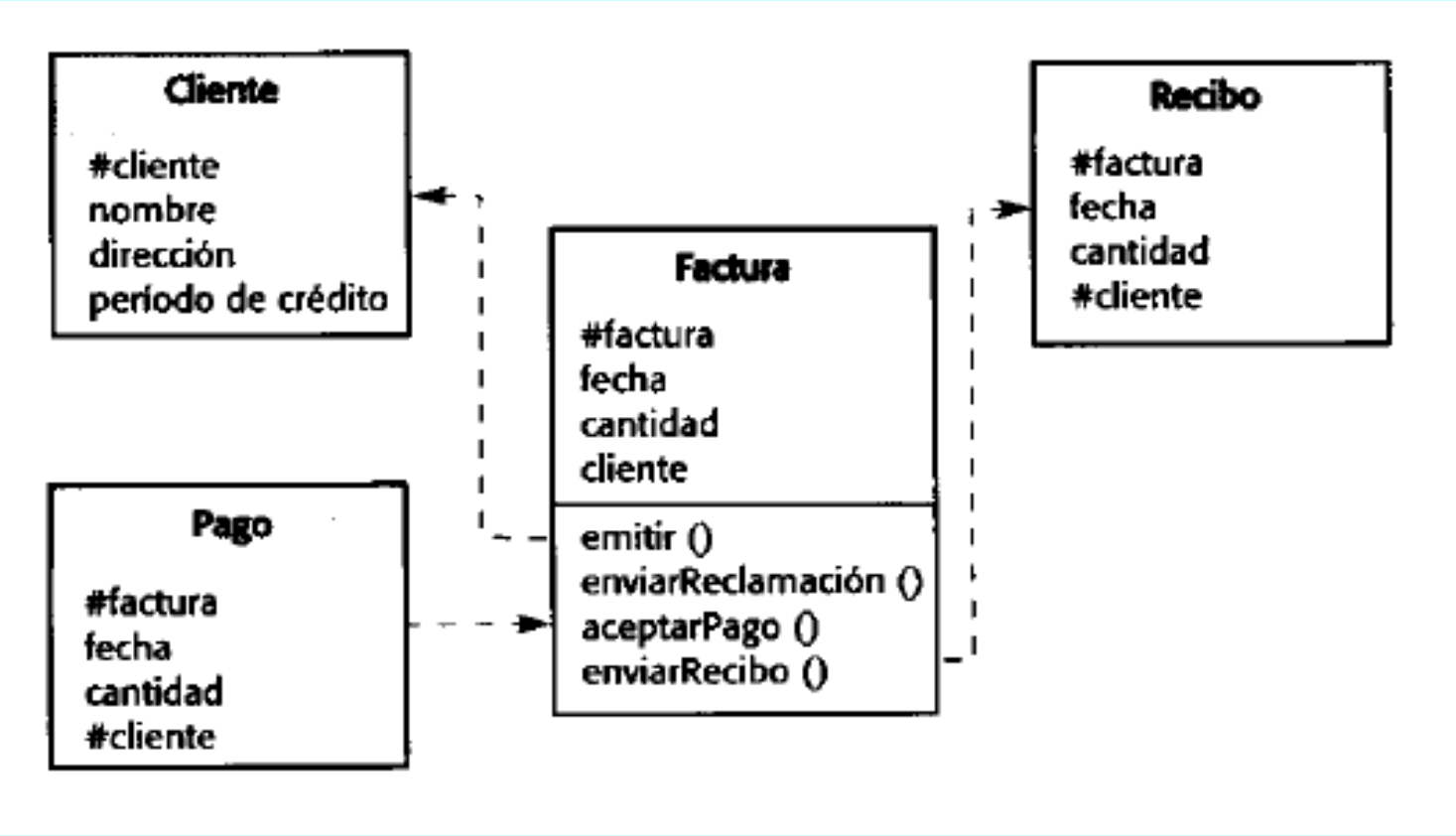
Descomposición modular

- Otro nivel estructural en que sub-sistemas se descomponen en módulos.
- Dos modelos de descomposición modular
  - Un modelo de objetos cuando el sistema se descompone en la interacción objeto;
  - Una tubería o modelo de flujo de datos cuando el sistema se descompone en módulos funcionales que transforman las entradas en salidas.
- Si es posible, las decisiones acerca de la concurrencia deben retrasarse hasta que los módulos se apliquen.

Modelo de Objetos

- Estructura del sistema en un conjunto de objetos débilmente acoplados con interfaces bien definidas.
- La descomposición orientada a objetos se refiere a la identificación de clases de objetos, sus atributos y operaciones.
- Cuando se aplica, los objetos se crean a partir de estas clases y algunos modelos de control para coordinar las operaciones del objeto.

Sistema de procesamiento de la factura



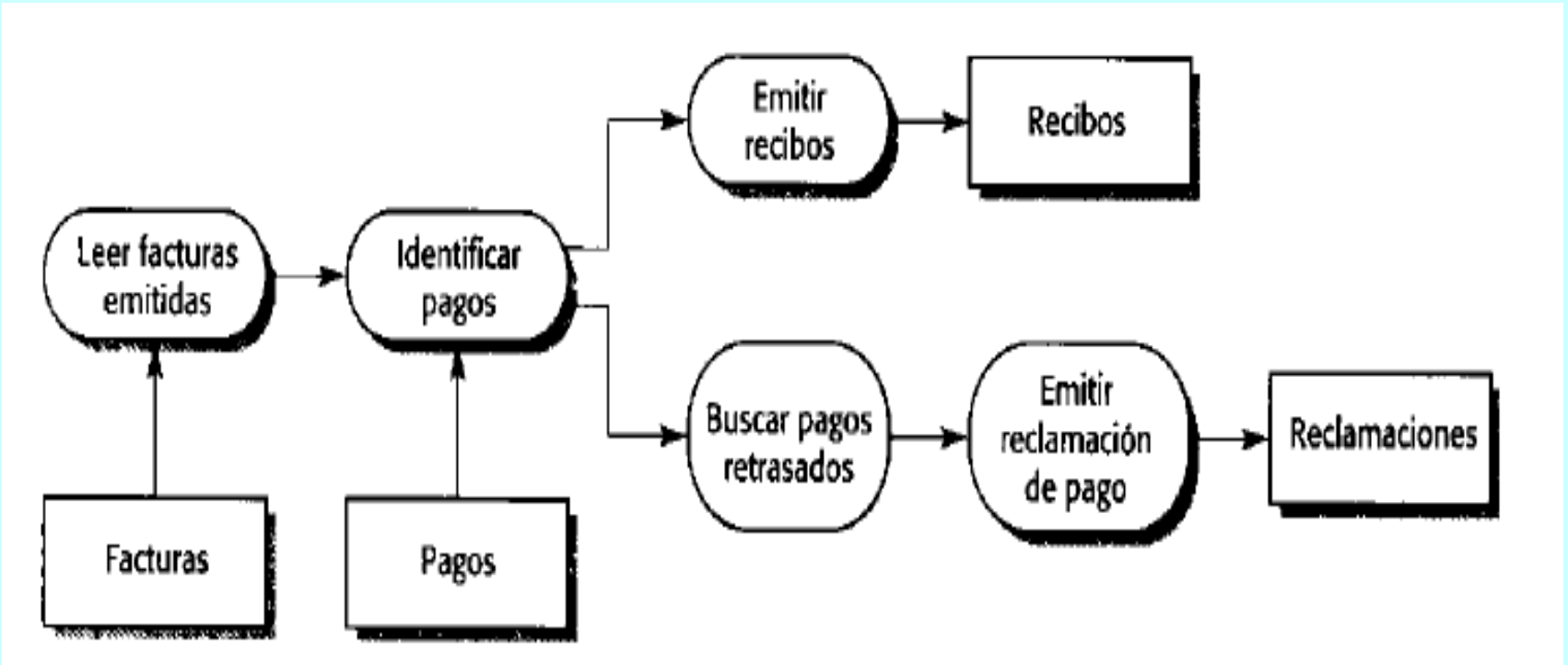
Ventajas del modelo de objetos

- Los objetos son débilmente acoplados por lo que su aplicación puede ser modificada sin afectar a otros objetos.
- Los objetos pueden reflejar entidades del mundo real.
- Implementación de lenguajes orientados a objetos son extensamente usados.
- Sin embargo, los cambios de interfaz de objetos puede causar problemas y entidades complejas pueden ser difíciles de representar como objetos.

Función orientada a pipelining

- Transformaciones funcionales transforman sus entradas en salidas.
- Puede ser denominado un modelo de filtro y tubo (como en el shell de UNIX).
- Variantes de este enfoque son muy comunes. Cuando las transformaciones son secuenciales, se trata de un modelo secuencial que se utiliza ampliamente en sistemas de procesamiento de datos.
- En realidad, no adecuado para sistemas interactivos.

Sistema de procesamiento de la factura



Ventajas Modelo de tubos

- Apoya la reutilización de transformación.
- Para la organización intuitiva de comunicación interesados.
- Fácil de añadir nuevas transformaciones.
- Relativamente fácil de aplicar, ya sea como un sistema concurrente o secuencial.
- Sin embargo, requiere de un formato común para la transferencia de datos a lo largo de la tubería y difícil de apoyo basada en eventos de interacción.

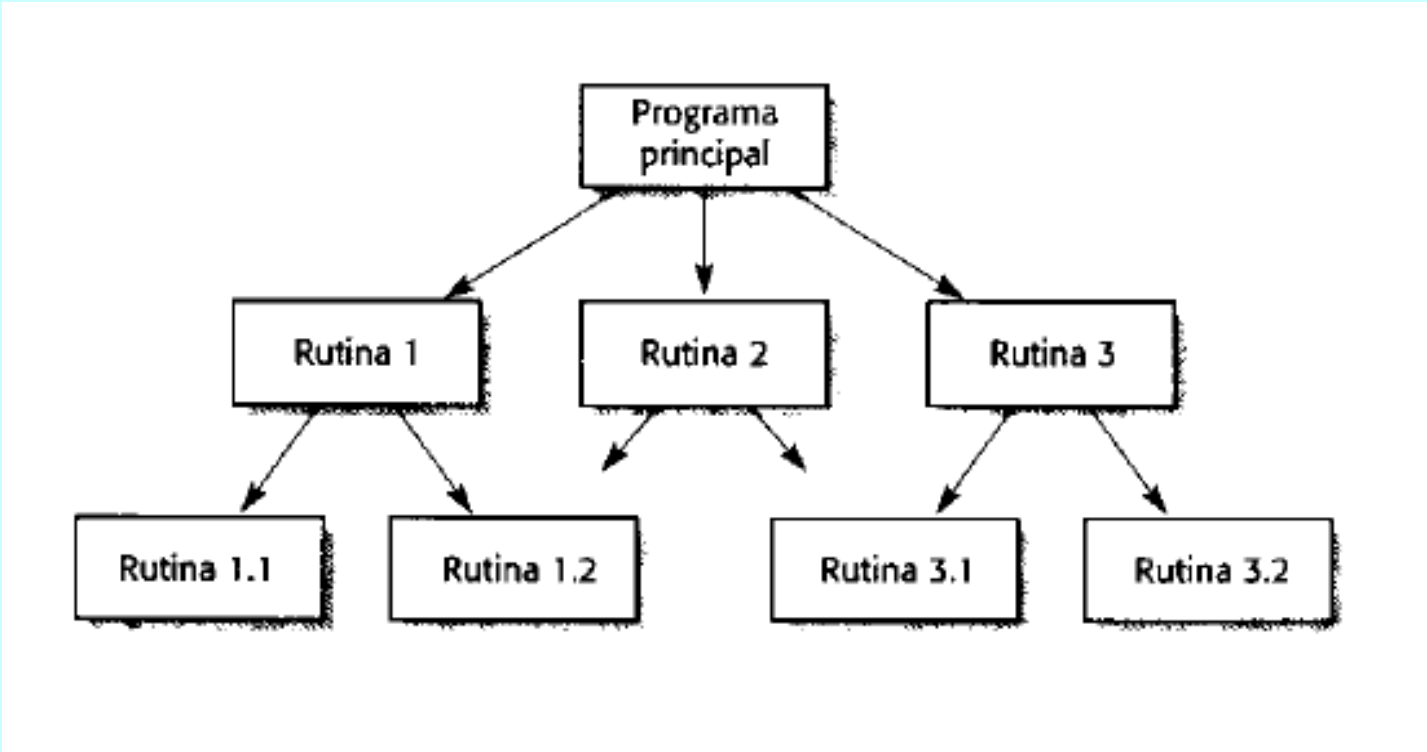
Control de estilos

- Están relacionadas con el control de flujo entre los sub-sistemas. Distinta de la descomposición del sistema modelo.
- Control centralizado
  - Un sub-sistema tiene la responsabilidad general de control de otros subsistemas.
- Control basado en eventos
  - Cada sub-sistema puede responder a eventos generados externamente de los demás sub-sistemas o del entorno del sistema.

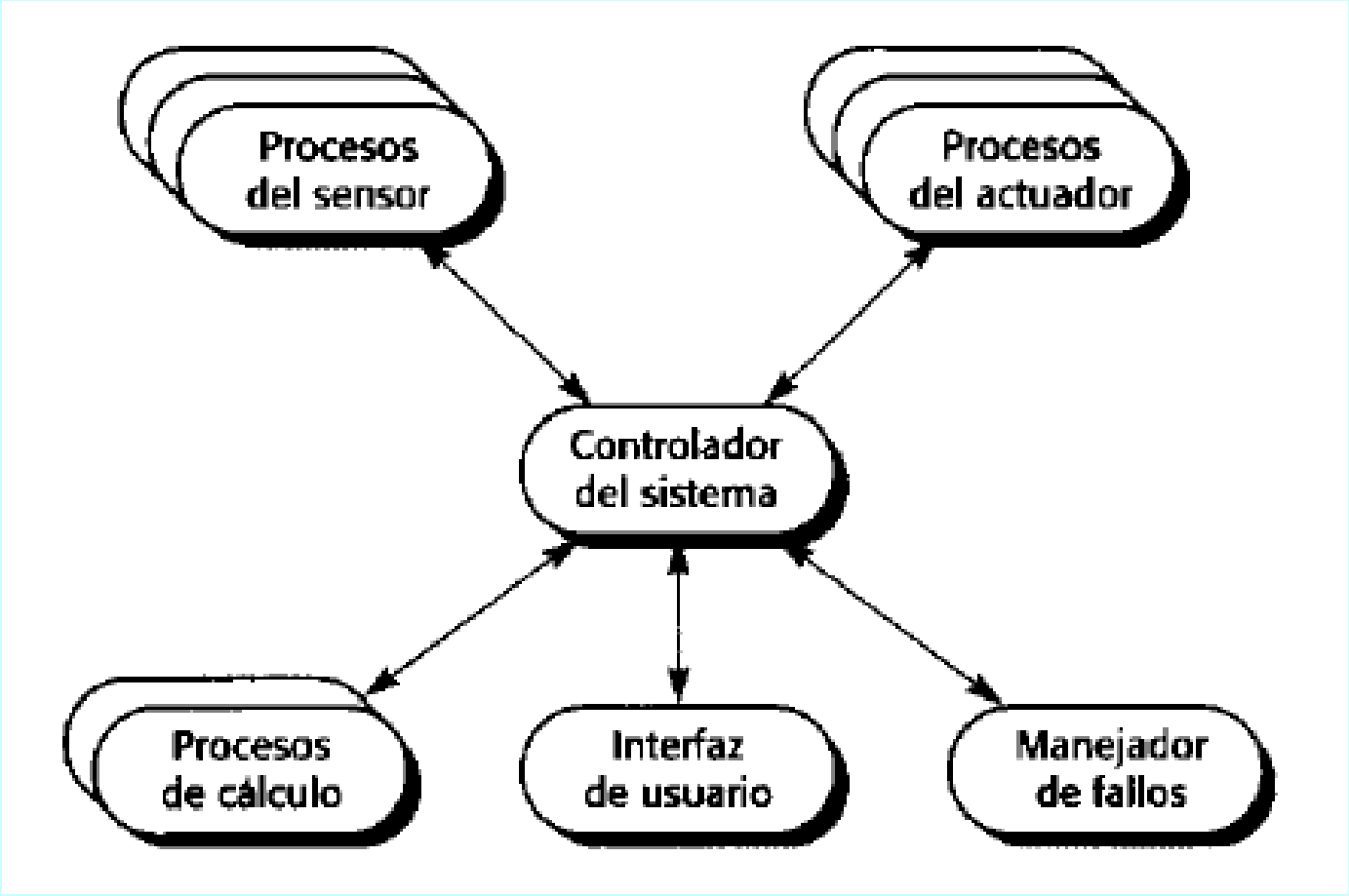
Control centralizado

- Un sub-sistema de control tiene la responsabilidad de gestionar la ejecución de otros subsistemas.
- Modelo de retorno
  - Modelo de subrutinas de arriba hacia abajo donde el control se inicia en la cima de una jerarquía de subrutinas y se mueve hacia abajo. Aplicables a los sistemas secuenciales.
- Modelo administrador
  - Aplicable a sistemas concurrentes. Un componente del sistema controla la interrupción, la puesta en marcha y coordinación de otros procesos del sistema. Puede ser implementado en sistemas secuenciales como una declaración.

Modelo de control llamada/retorno



Modelo de control centralizado para sistema de tiempo real



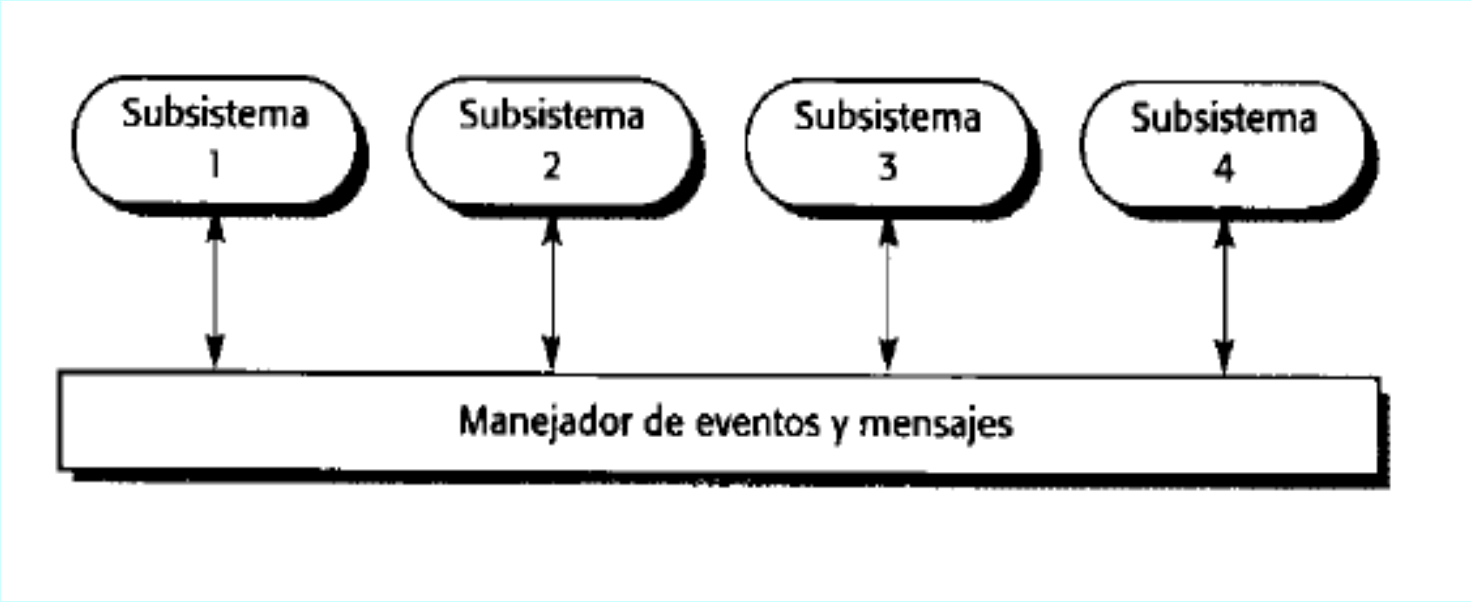
Sistema impulsado por eventos

- Impulsada por eventos generados externamente en el que la coordinación del evento está fuera del control de los sub-sistemas que procesan el evento.
- Dos principales modelos
  - Modelos de difusión. Un acontecimiento es transmitido a todos los sub-sistemas. Cualquier sub-sistema que puede manejar el evento pueden hacerlo;
  - Modelos de manejo de interrupciones. Utilizados en sistemas de tiempo real donde las interrupciones son detectadas por un manejador de interrupción y pasan a algún otro componente para la transformación.
- Otros modelos de sistemas impulsados por eventos incluyen a las hojas de cálculo y sistemas de producción.

Modelo de difusión

- Eficaz en la integración de sub-sistemas en diferentes ordenadores en una red.
- Sub-sistemas de registro de un interés específico en los acontecimientos. Cuando se producen, el control es transferido a la sub-sistema que puede manejar el evento.
- Las políticas de control no están incrustadas en el evento y manejador de mensaje. Sub-sistemas deciden sobre los acontecimientos de interés para ellos.
- Sin embargo, los sub-sistemas no saben cuándo o si un evento se manejará.

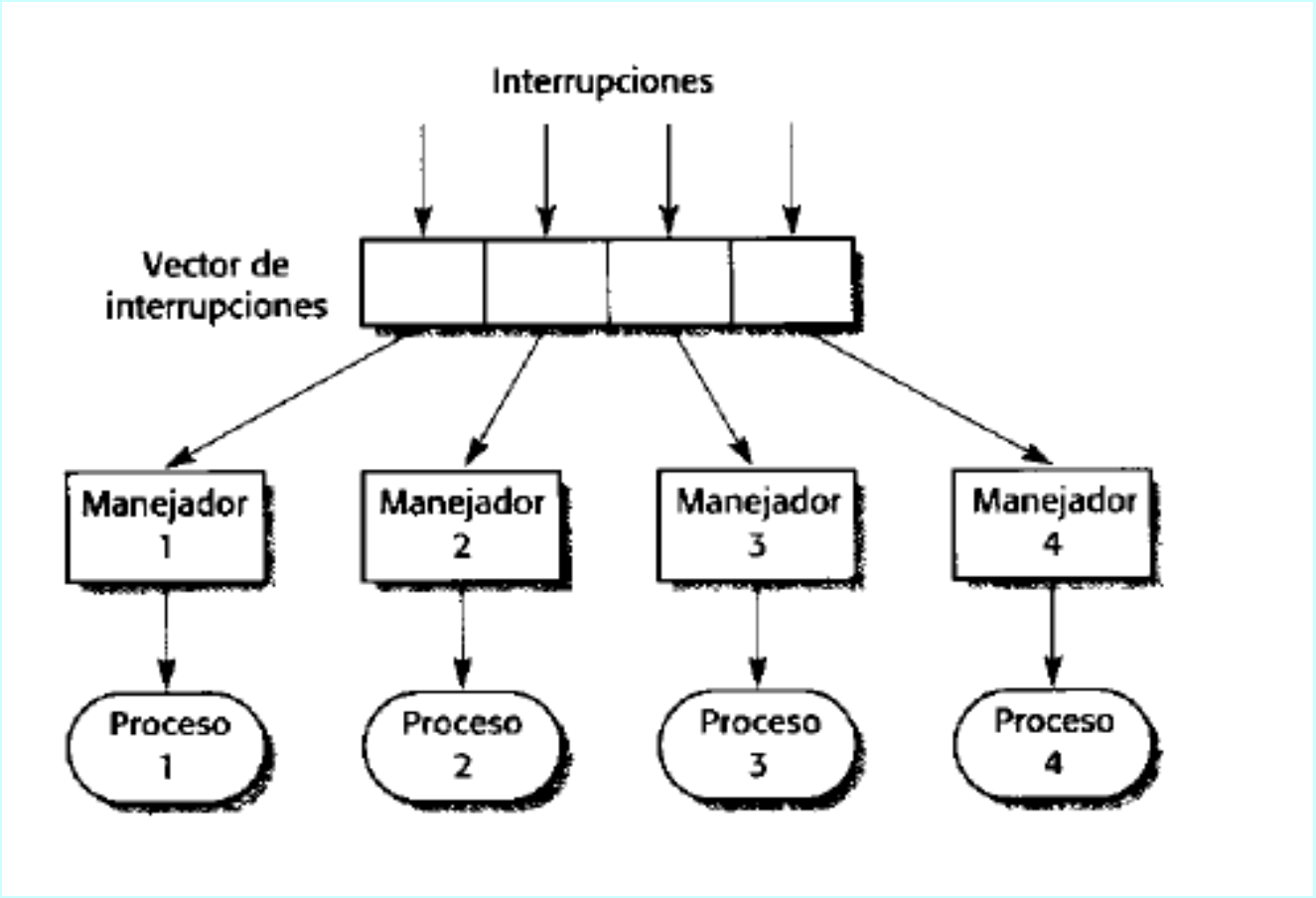
Difusión o transmisión selectiva



Sistemas impulsado por interrupciones

- Utilizados en sistemas de tiempo real, donde una respuesta rápida a un evento es fundamental.
- Se conocen los tipos de interrupción con un manejador definido para cada tipo.
- Cada tipo se asocia con una ubicación de memoria y un interruptor de hardware causas de transferencia a su manejador.
- Permite una respuesta rápida, pero compleja y difícil de programar para validar.

Modelo de control conducido por interrupciones



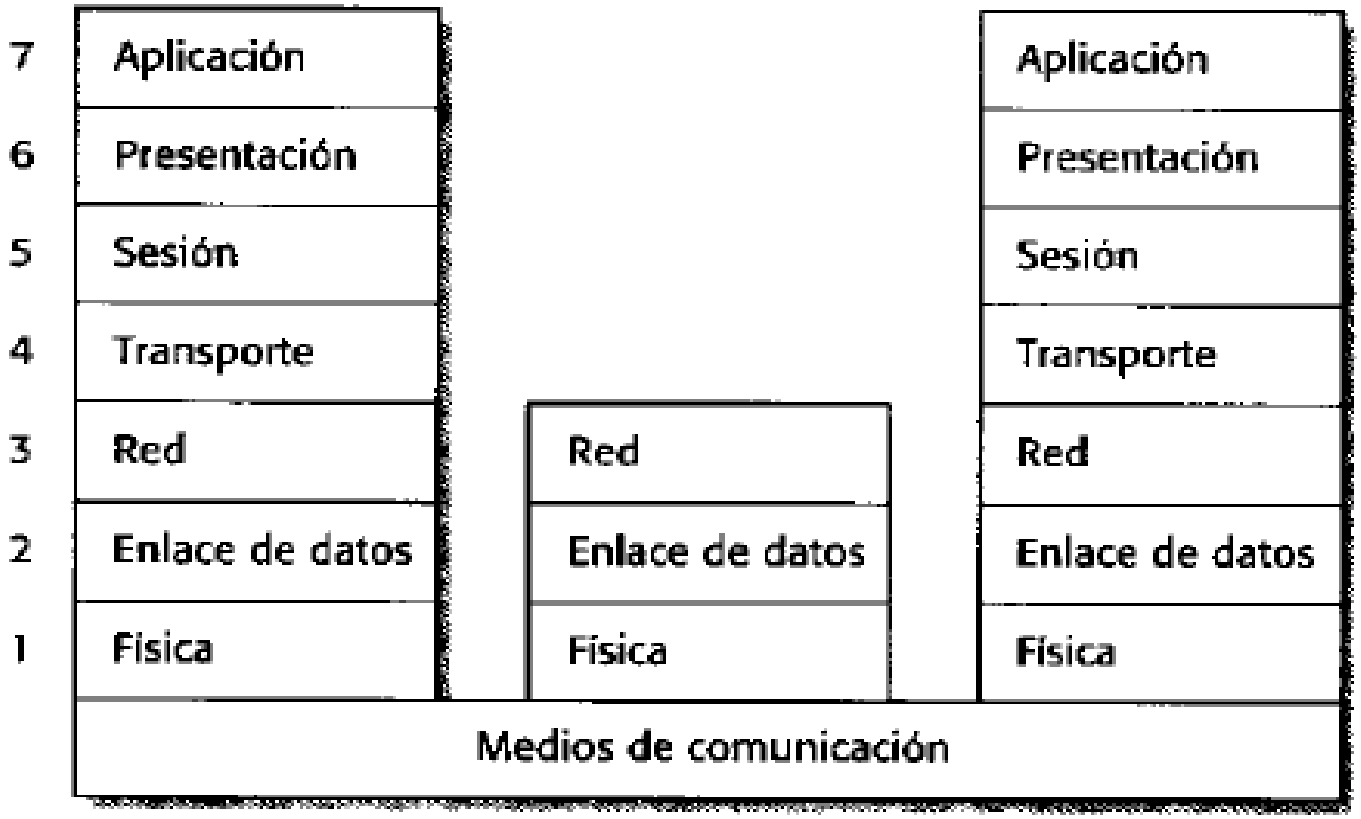
Arquitecturas de referencia

- Modelos arquitectónicos pueden ser específicos para algunas aplicaciones de dominio.
- Dos tipos de dominio específico de modelo
  - Modelos genéricos que son abstracciones de una serie de sistemas reales, y que encierran las principales características de estos sistemas. Contempladas en el Capítulo 13.
  - Modelos de referencia que son más abstractos, modelo idealizado. Proporcionar un medio de información acerca de que clase de sistema y de la comparación de diferentes arquitecturas.
- Modelos genéricos son por lo general los modelos de abajo hacia arriba, los modelos de referencia son de arriba abajo.

Arquitecturas de referencia

- Modelos de referencia se derivan de un estudio de la solicitud de dominio en lugar de los sistemas existentes.
- Puede utilizarse como base para la aplicación del sistema o para comparar diferentes sistemas. Actúa como un estándar contra el que los sistemas pueden ser evaluados.
- Modelo OSI es un modelo de capas para los sistemas de comunicación.

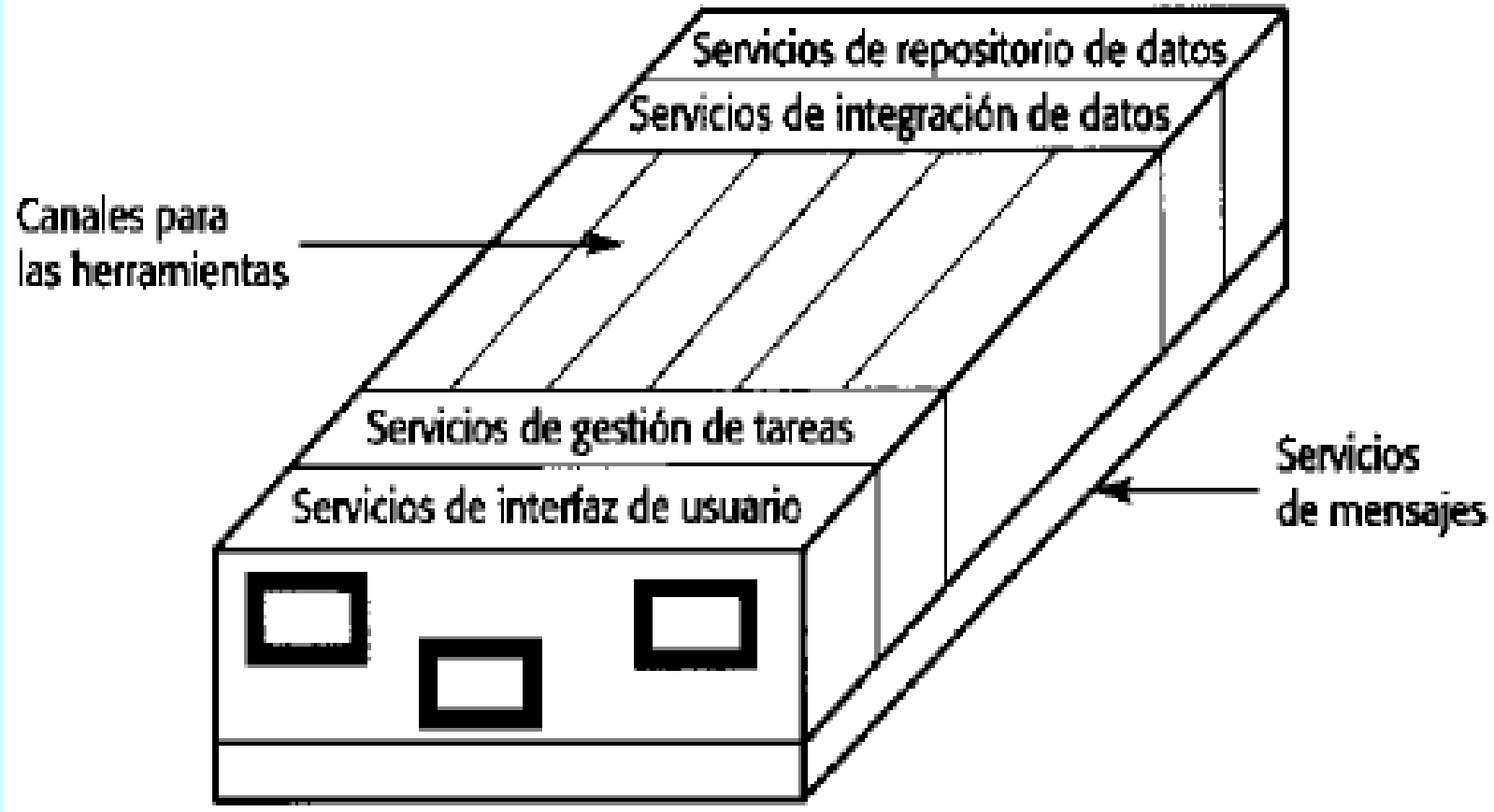
Modelo de referencia OSI



Caso modelo de referencia

- Servicios de datos
  - Almacenamiento y gestión de datos.
- Servicios de integración de datos
  - Gestión de grupos de entidades.
- Servicios de gestión de tareas
  - Definición y establecimiento de modelos de procesos.
- Los servicios de mensajería
  - Herramienta-herramienta y herramientas de comunicación y el medio ambiente.
- Servicios de interfaz de usuario
  - Interfaz de usuario de desarrollo.

El modelo de referencia ECMA para entornos CASE



Puntos clave

- La arquitectura de software es el marco fundamental para la estructuración del sistema.
- Decisiones de diseño arquitectónico incluye decisiones sobre la arquitectura de aplicaciones, la distribución y los estilos arquitectónicos que se utilizará.
- Diferentes modelos arquitectónicos, como un modelo estructural, un modelo de control y un modelo de descomposición puede ser desarrollado.
- Sistema de repositorio de modelos organizativos son modelos cliente-servidor y modelos de máquinas abstractas.

Puntos clave

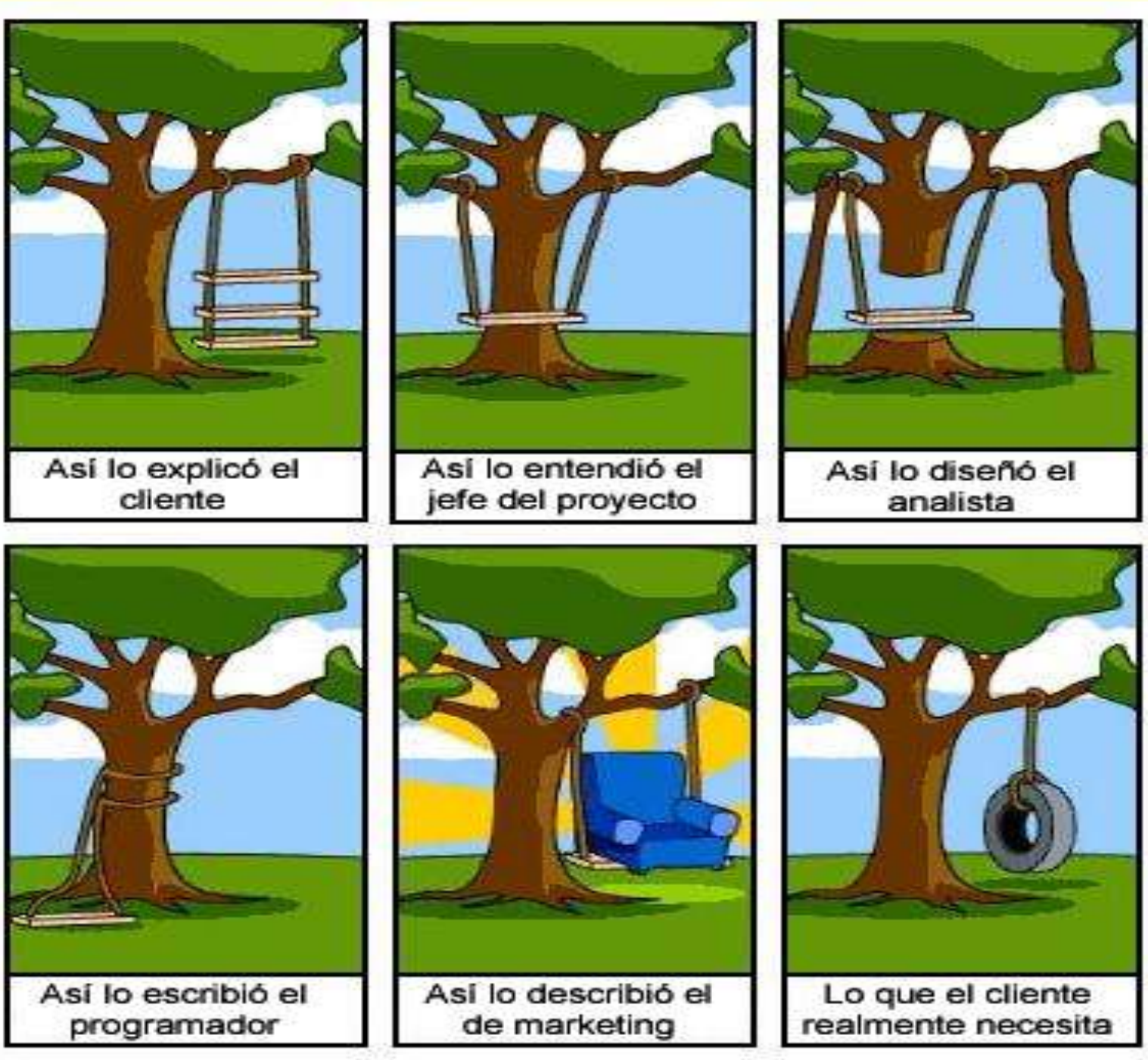
- Modelos de contron incluyen control centralizado y modelos impulsados por eventos.
- Arquitecturas de referencia pueden ser utilizadas para comunicarse de dominio específico de las arquitecturas y para evaluar y comparar los diseños arquitectónicos.

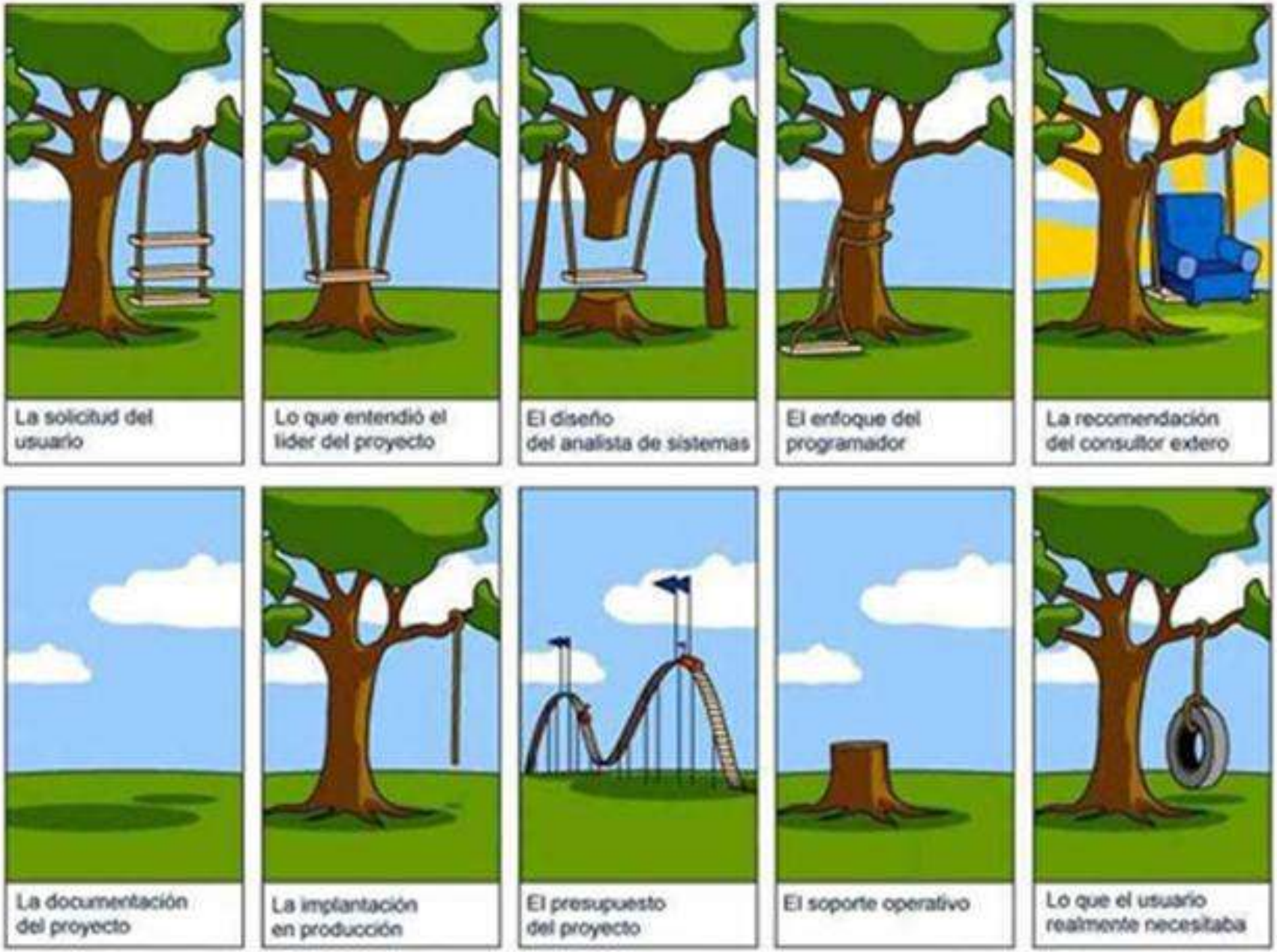
Modelos arquitectónicos

- Diferentes modelos arquitectónicos pueden producirse durante el proceso de diseño
- Cada modelo presenta diferentes perspectivas sobre la arquitectura

Atributos de la arquitectura

- Rendimiento
  - La localización de las operaciones para reducir al mínimo sub-sistema de comunicación
- Seguridad
  - Use una arquitectura con activos críticos en capas internas
- Protección
  - Aislar los componentes esenciales para la seguridad física
- Disponibilidad
  - Incluir componentes redundantes en la arquitectura
- Mantenibilidad
  - Uso de grano fino, autónomo de los componentes


Así lo explicó el cliente
Así lo entendió el jefe del proyecto
Así lo diseñó el analista
Así lo escribió el programador
Así lo describió el de marketing
Lo que el cliente realmente necesita


La solicitud del usuario
Lo que entendió el líder del proyecto
El diseño del analista de sistemas
El enfoque del programador
La recomendación del consultor extero
La documentación del proyecto
La implantación en producción
El presupuesto del proyecto
El soporte operativo
Lo que el usuario realmente necesitaba

Gestión de proyectos

Objetivos

- Explicar las principales tareas emprendidas por los gestores de proyectos de software
- Introducir la gestión de proyecto de software y describir sus características distintivas
- Discutir la planificación de proyectos y el proceso de planificación
- Mostrar cómo las representaciones gráficas son usadas por la gestión de proyectos
- Discutir la noción de los riesgos y el proceso de gestión de riesgos

Tópicos Expuestos

- Actividades de gestión
- Planificación de proyectos
- Calendarización del proyecto
- Gestión de riesgos

Gestión de proyectos software

- Concerniente con actividades implicadas en asegurar que el software es entregado a tiempo y acorde a los requerimientos de la organización, al desarrollar y procurar el software.
- La gestión de proyectos es necesaria porque el desarrollo de software está siempre sujeto a limitaciones de presupuesto y calendario fijadas por la organización desarrolladora del software.

Distinciones de la gestión de software

- El producto es intangible.
- El producto es especialmente flexible.
- Ingeniería de software no es reconocida como una disciplina de ingeniería con la misma condición de una mecánica, ingeniería eléctrica, etc
- El proceso de desarrollo de software no está estandarizado.
- Muchos proyectos de software son proyectos únicos.

Actividades de gestión

- Redacción de la propuesta.
- Planificación y calendarización del proyecto.
- Estimación de costes del proyecto.
- Supervisión y revisión del proyecto.
- Selección y evaluación del personal.
- Redacción y presentación de informes.

Gestión – Aspectos comunes

- Estas actividades no son propias de gestión de software.
- Muchas de las técnicas de la gestión de proyectos de ingeniería son igualmente aplicables a la gestión de proyectos de software.
- Técnicamente complejos sistemas de ingeniería tienden a sufrir los mismos problemas que sistemas de software.

Dotación de personal del proyecto

- Puede que no sea posible nombrar al hombre ideal para trabajar en un proyecto
  - Proyecto de presupuesto no puede permitir la utilización de personal altamente remunerado;
  - Personal con la experiencia adecuada puede no estar disponible;
  - La organización desea desarrollar las habilidades de sus empleados.
- Administradores tienen que trabajar dentro de estas limitaciones, especialmente cuando hay escasez de personal capacitado.

La planificación de proyectos

- Probablemente, la mayor parte del tiempo que consume la actividad de gestión de proyectos.
- Actividad continua desde la idea hasta la entrega del sistema. Los planes deben ser revisados regularmente así se disponga de nueva información.
- Distintos tipos de plan pueden ser desarrollados para apoyar el plan principal de proyecto de software con presupuesto y calendarización.

Tipos de plan de proyecto

Plan	Descripción
Plan de calidad	Describe los procedimientos y los estándares de calidad que se utilizarán en un proyecto. Véase el Capítulo 24.
Plan de validación	Describe el enfoque, los recursos y la programación utilizados para la validación del sistema.
Plan de gestión de configuraciones	Describe los procedimientos para la gestión de configuraciones y las estructuras a utilizar. Véase el Capítulo 29.
Plan de mantenimiento	Predice los requerimientos del mantenimiento del sistema, los costes del mantenimiento y el esfuerzo requerido. Véase el Capítulo 27.
Plan de desarrollo del personal	Describe cómo se desarrollan las habilidades y experiencia de los miembros del equipo del proyecto.

Planificación del proyecto

Establecer las limitaciones del proyecto
Hacer las evaluaciones iniciales de los parámetros del proyecto
Definir los hitos del proyecto y los resultados
Mientras que los proyectos no se ha completado o cancelado **repetir**
    Elaborar cronograma
    Iniciar las actividades según el calendario previsto
    Esperar (por un rato)
    Examen de la marcha del proyecto
    Revisar las estimaciones de los parámetros del proyecto
    Actualizar el calendario del proyecto
    Volver a negociar las limitaciones del proyecto y los resultados
    Si (surgen problemas) **entonces**
        Iniciar la revisión técnica y la posible revisión
    Fin Si
fin de **repetir**

El plan de proyecto

- El plan del proyecto establece:
  - Los recursos disponibles para el proyecto;
  - Una división del trabajo;
  - Un plan de trabajo.

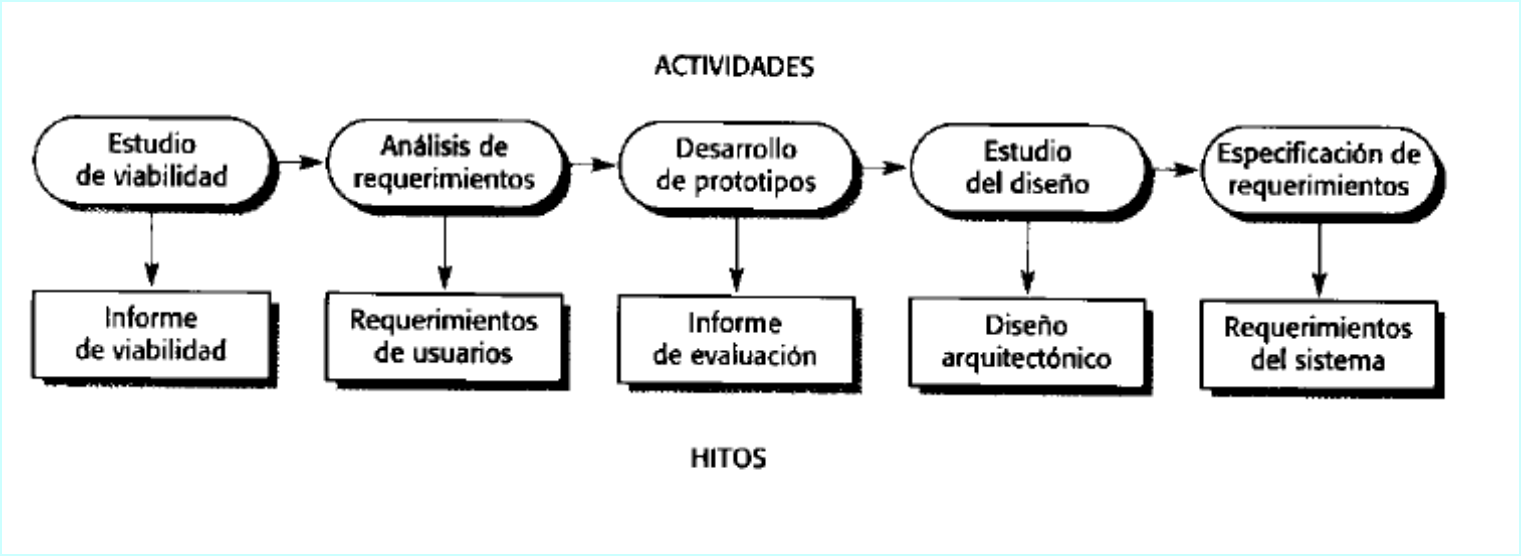
Estructura del plan del proyecto

- Introducción.
- Organización del proyecto.
- Análisis de riesgo.
- Requerimiento de los recursos de hardware y software.
- División del trabajo.
- Programa del proyecto.
- Mecanismos de supervisión y presentación de informes.

Organización de las actividades

- Las actividades en un proyecto deben organizarse para producir resultados tangibles de la gestión para evaluar los progresos realizados.
- *Hitos son el punto final de una actividad del proceso de software.*
- *Una entrega es el resultado del proyecto que se entrega al cliente.*
- El proceso de cascada permite la definición directa de hitos progresivos.

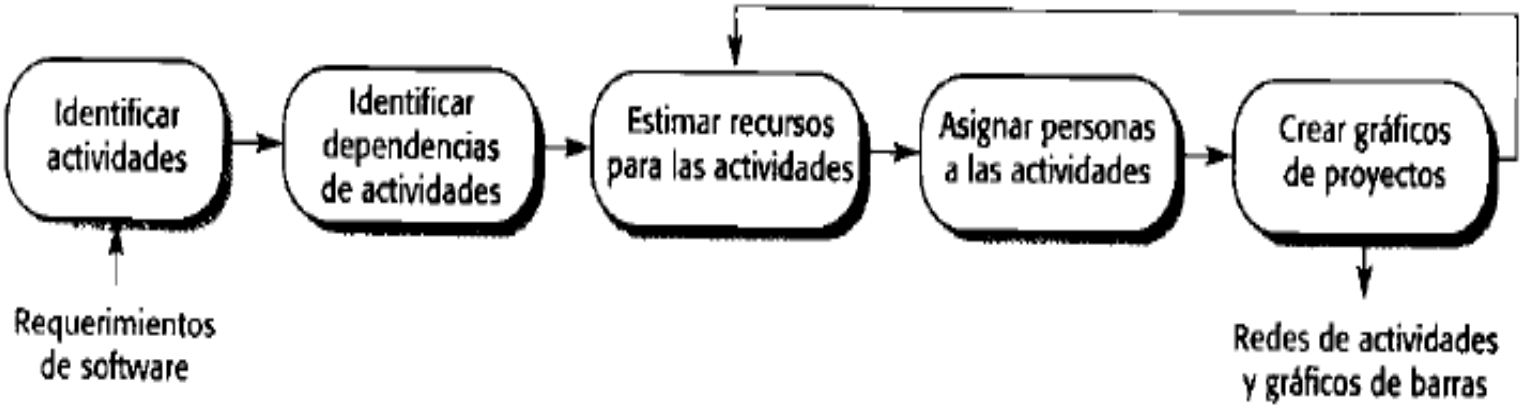
Hitos en el proceso de especificación de requerimientos



Calendarización del proyecto

- Dividir el proyecto en tareas y estimar el tiempo y los recursos necesarios para completar cada tarea.
- Organizar tareas simultáneamente para aprovechar al máximo el uso de la fuerza de trabajo.
- Minimizar la dependencia de las tareas a fin de evitar retrasos causados por una tarea en espera para completar otra.
- Depende de la intuición y experiencia de los gerentes del proyecto.

Proceso de calendarización del proyecto



Problemas de calendarización

- La estimación de la dificultad de los problemas y por lo tanto, el coste de desarrollo de una solución es difícil.
- La productividad no es proporcional al número de personas que trabajan en una tarea.
- Adición de personas al finalizar un proyecto hace que se prolongue más debido a los gastos generales de comunicación.
- Lo inesperado siempre ocurre. Permitir siempre en la planificación, la contingencia.

Gráficos de barras y redes de actividades

- Notaciones gráficas para ilustrar el calendario del proyecto.
- Mostrar división del proyecto en tareas. Las tareas no deben ser demasiado pequeñas. Deberían tomar una semana o dos.
- Los gráficos de actividad muestran la dependencia de las tareas y la ruta crítica.
- Gráficos de barras muestran la calendarización propia contra el tiempo.

Duración y dependencias de las tareas

Actividad	Duración (dias)	Dependencias
T1	8	
T2	15	
T3	15	T1 (M1)
T4	10	
T5	10	T2, T4 (M2)
T6	5	T1, T2 (M3)
T7	20	T1 (M1)
T8	25	T4 (M5)
T9	15	T3, T6 (M4)
T10	15	T5, T7 (M7)
T11	7	T9 (M6)
T12	10	T11 (M8)

Red de actividades

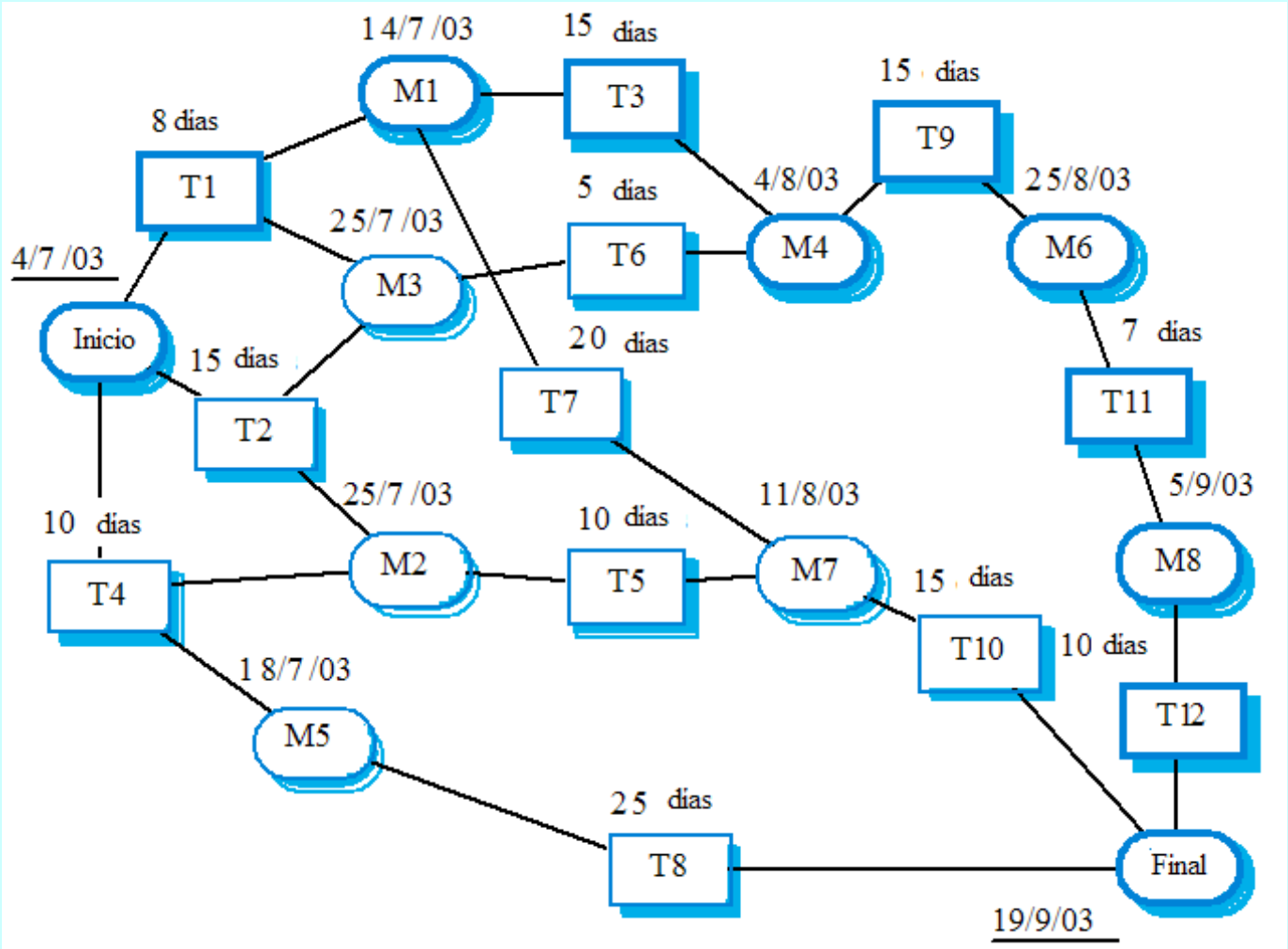
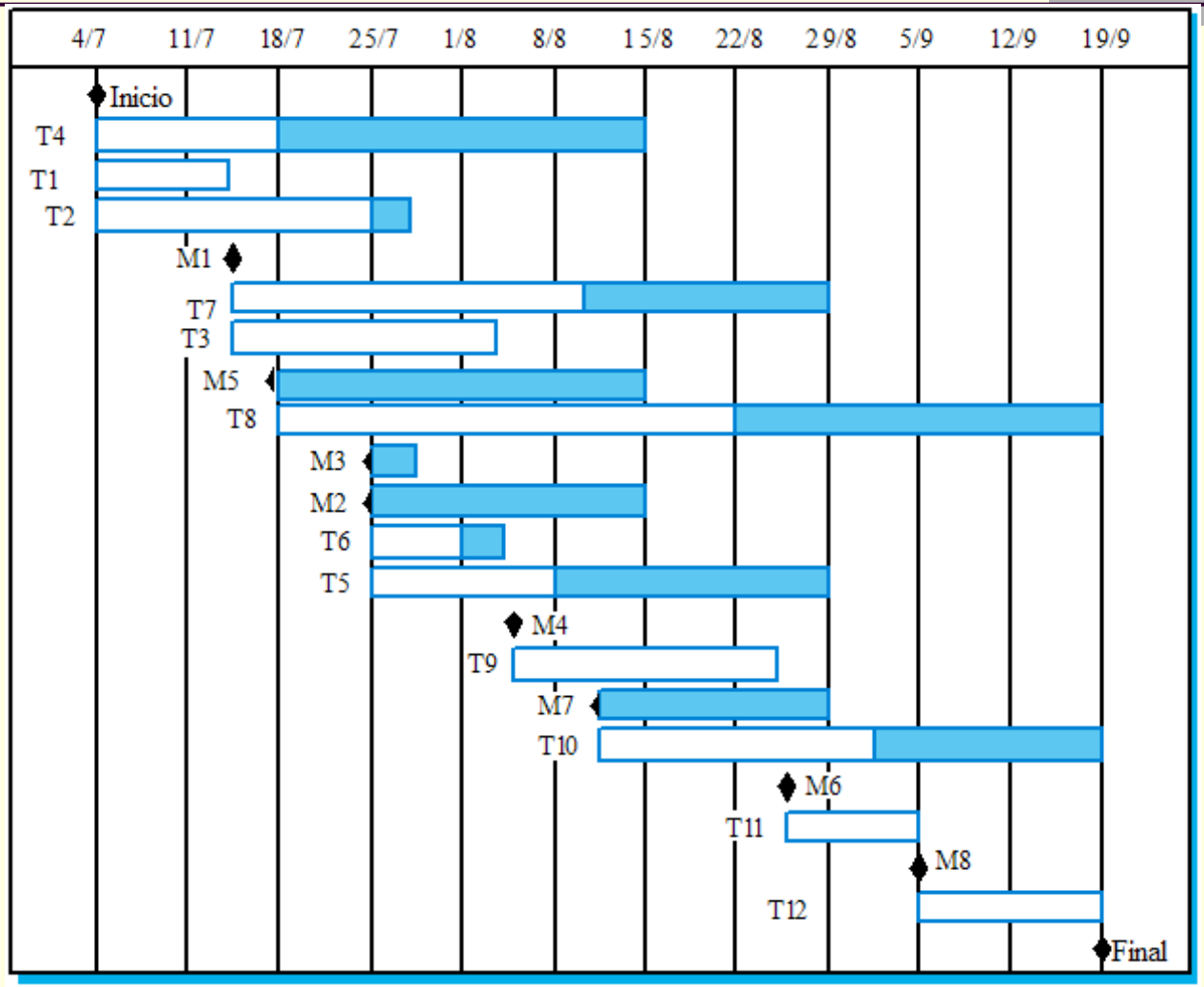
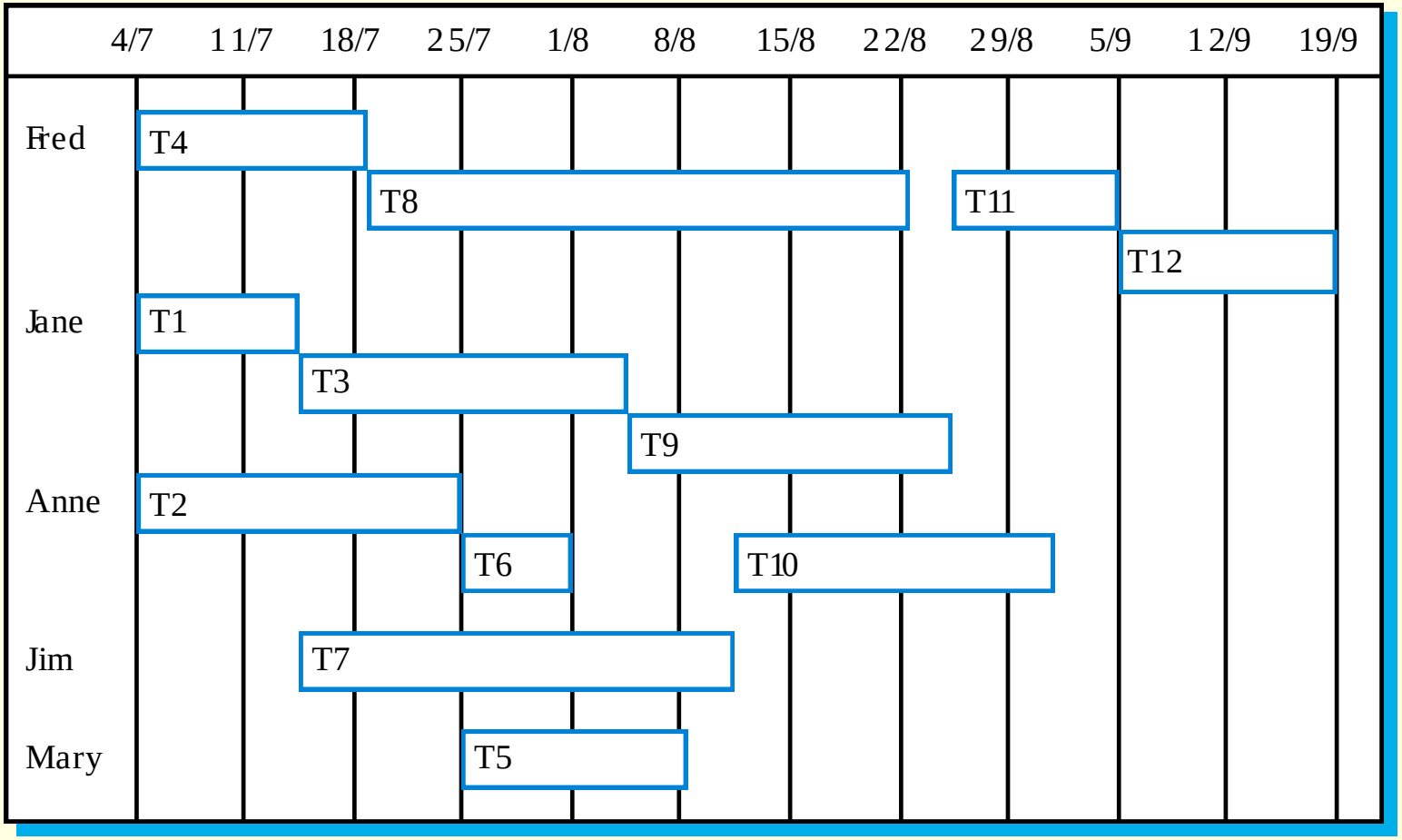


Gráfico de barras de actividades



Asignación de personal/tiempo



La gestión del riesgo

- La gestión del riesgo se refiere a la identificación de riesgos y la elaboración de planes para reducir al mínimo su efecto sobre un proyecto.
- Un riesgo es una probabilidad de que algunas circunstancias adversas se produzcan
  - Afectan a la calendarización del proyecto o a los recursos;
  - Los riesgos del producto afectan a la calidad o al funcionamiento del software que se está desarrollando;
  - Los riesgos de negocio afectan a la

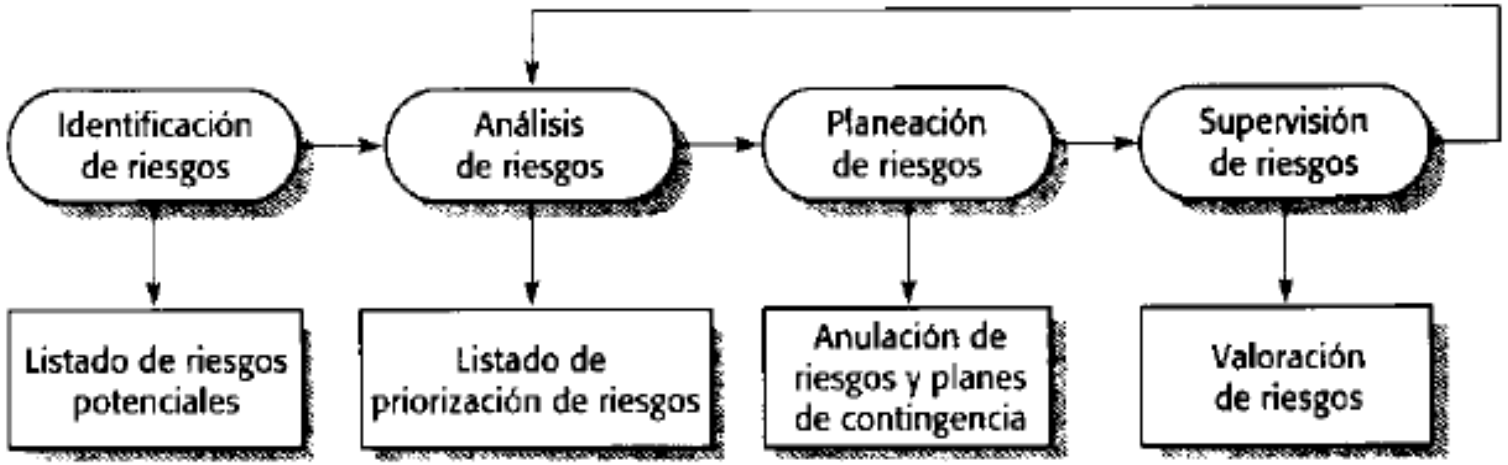
Riesgos posibles del software

Riesgo	Afecta a	Descripción
Rotación de personal	Proyecto	Personal con experiencia abandona el proyecto antes de que finalice.
Cambio de gestión	Proyecto	Habrá un cambio de gestión organizacional con diferentes prioridades.
No disponibilidad del hardware	Proyecto	El hardware esencial para el proyecto no será entregado a tiempo.
Cambio de requerimientos	Proyecto y producto	Habrá más cambios en los requerimientos de lo esperado.
Retrasos en la especificación	Proyecto y producto	Las especificaciones de las interfaces esenciales no estarán a tiempo.
Subestimación del tamaño	Proyecto y producto	El tamaño del sistema se ha subestimado.
Bajo rendimiento de la herramienta CASE	Producto	Las herramientas CASE que ayudan al proyecto no tienen el rendimiento esperado.
Cambio de tecnología	Negocio	Un producto competitivo se pone en venta antes de que el sistema se complete.
Competencia del producto	Negocio	La tecnología fundamental sobre la que se construirá el sistema se sustituye por nueva tecnología.

El proceso de gestión de riesgos

- La identificación de riesgos
  - Identificar riesgos empresariales, en los proyectos y productos;
- Análisis de riesgos
  - Evaluar la probabilidad y consecuencias de estos riesgos;
- Planificación de riesgo
  - Elaborar planes para evitar o minimizar los efectos de la situación de riesgo;
- Supervisión de riesgos
  - Controlar los riesgos a lo largo del proyecto;

El proceso de gestión de riesgos



La identificación de riesgos

- Riesgos tecnológicos.
- Riesgos de personal.
- Riesgos organizacionales.
- Riesgos de requerimientos.
- Riesgos de estimación.

Riesgos y los tipos de riesgo

Tipo de riesgo	Posibles riesgos
Tecnología	La base de datos que se utiliza en el sistema no puede procesar muchas transacciones por segundo como se esperaba. Los componentes de software que deben reutilizarse contienen defectos que limitan su funcionalidad.
Personal	Es imposible reclutar personal con las habilidades requeridas para el proyecto. El personal clave está enfermo y no disponible en momentos críticos. La capacitación solicitada para el personal no está disponible.
Organizacional	La organización se reestructura de tal forma que una gestión diferente se responsabiliza del proyecto. Los problemas financieros de la organización fuerzan a reducciones en el presupuesto del proyecto.
Herramientas	Es ineficiente el código generado por las herramientas CASE. Las herramientas CASE no se pueden integrar.
Requerimientos	Se proponen cambios en los requerimientos que requieren rehacer el diseño. Los clientes no comprenden el impacto de los cambios en los requerimientos.
Estimación	El tiempo requerido para desarrollar el software está subestimado. La tasa de reparación de defectos está subestimada. El tamaño del software está subestimado.

Análisis de riesgos

- Evaluar la probabilidad y gravedad de cada riesgo.
- Probabilidad de riesgo puede ser muy baja, baja, moderada, alta o muy alta.
- Los efectos del riesgo pueden ser catastróficos, serios, tolerables o insignificantes.

Análisis de riesgos (i)

Riesgo	Probabilidad	Efectos
Los problemas financieros de la organización fuerzan a reducir el presupuesto del proyecto.	Baja	Catastrófico
Es imposible reclutar personal con las habilidades requeridas para el proyecto.	Alta	Catastrófico
El personal clave está enfermo y no disponible en momentos críticos.	Moderada	Serio
Los componentes de software que deben reutilizarse contienen defectos que limitan su funcionalidad.	Moderada	Serio
Se proponen cambios en los requerimientos que requieren rehacer el diseño.	Moderada	Serio
La organización se reestructura de tal forma que cambia el grupo de gestión.	Alta	Serio

Análisis de riesgo (ii)

Riesgo	Probabilidad	Efectos
La base de datos que se utiliza en el sistema no puede procesar muchas transacciones por segundo como se esperaba.	Moderada	Serio
El tiempo requerido para desarrollar el software está subestimado.	Alta	Serio
Las herramientas CASE no se pueden integrar.	Alta	Tolerable
Los clientes no comprenden el impacto de los cambios en los requerimientos.	Moderada	Tolerable
La capacitación solicitada para el personal no está disponible.	Moderada	Tolerable
La tasa de reparación de defectos está subestimada.	Moderada	Tolerable
El tamaño del software está subestimado.	Alta	Tolerable
Es ineficiente el código generado por las herramientas CASE.	Moderada	insignificante

Planificación de riesgos

- Considera cada uno de los riesgos y desarrolla una estrategia para gestionar cada riesgo.
- Estrategias de prevención
  - La probabilidad de que se producirá el riesgo se reduce;
- Estrategias de minimización
  - El impacto del riesgo en el proyecto o el producto se redujo;
- Planes de contingencia
  - Si se plantea el riesgo, planes de contingencia son los planes para hacer frente a ese riesgo;

Estrategias de gestión de riesgos (i)

Riesgo	Estrategia
Problemas financieros de la organización	Preparar un documento breve para el gestor principal que muestre que el proyecto hace contribuciones muy importantes a las metas del negocio.
Problemas de reclutamiento	Alertar al cliente de las dificultades potenciales y los posibles retrasos, investigar la compra de componentes.
Enfermedad de personal	Reorganizar el equipo de tal forma que haya solapamiento en el trabajo y las personas comprendan el de los demás.
Componentes defectuosos	Reemplazar los componentes defectuosos con los comprados de fiabilidad conocida.

Estrategias de gestión del riesgo (ii)

Riesgo	Estrategia
Cambios de los requerimientos	Rastrear la información para valorar el impacto de los requerimientos, maximizar la información oculta en ellos.
Reestructuración organizacional	Preparar un documento breve para el gestor principal que muestre que el proyecto hace contribuciones muy importantes a las metas del negocio.
Rendimiento de la base de datos	Investigar la posibilidad de comprar una base de datos de alto rendimiento.
Tiempo de desarrollo subestimado	Investigar los componentes comprados y la utilización de un generador de programas.

Supervisión de riesgos

- Evaluar periódicamente cada uno de los riesgos identificados y decidir si es cada vez menos o más probable.
- También evaluar si los efectos del riesgo han cambiado.
- Cada riesgo clave debe ser discutido en las reuniones de gestión de progreso.

Factores de riesgo

Tipo de riesgo	Factores potenciales
Tecnología	Entrega retrasada del hardware o de la ayuda al software, muchos problemas tecnológicos reportados.
Personal	Baja moral del personal, malas relaciones entre los miembros del equipo, disponibilidad de empleo.
Organizacional	Chismorreo organizacional, falta de acciones por el gestor principal.
Herramientas	Rechazo de los miembros del equipo para utilizar herramientas, quejas acerca de las herramientas CASE, peticiones de estaciones de trabajo más potentes.
Requerimientos	Peticiones de muchos cambios en los requerimientos, quejas del cliente.
Estimación	Fracaso en el cumplimiento de los tiempos acordados, y en la eliminación de defectos reportados.

Puntos clave

- Buena gestión de los proyectos es esencial para el éxito del proyecto.
- El carácter intangible de software causa problemas para la gestión.
- Los administradores tienen diversas funciones, pero sus actividades más importantes son la planificación, estimación y calendarización.
- Planificación y estimación son procesos iterativos que continuarán durante todo el curso de un proyecto.

Puntos clave

- Un hito de un proyecto es un resultado predecible de una actividad en el que se debe presentar un informe oficial de los progresos realizados en la gestión.
- La calendarización del proyecto comprende la preparación de diversas representaciones gráficas que muestran las actividades del proyecto, su duración y la dotación de personal.
- La gestión del riesgo se refiere a la identificación de los riesgos que puedan afectar al proyecto y la planificación para asegurar que estos riesgos no se desarrollen en principales amenazas.