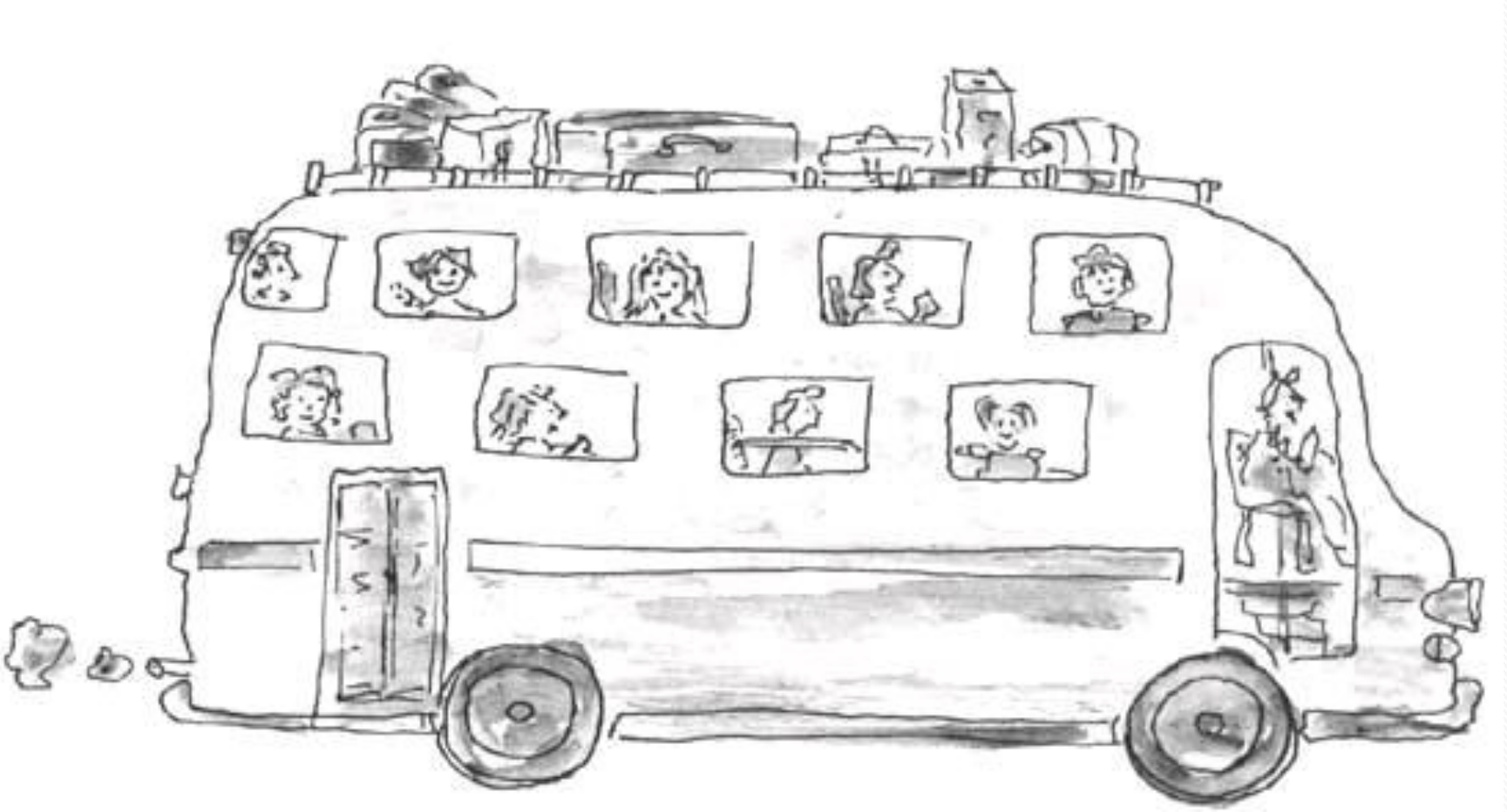
Análisis y Diseño de Sistemas de Información I

Objetivo

Comprender los conceptos fundamentales de **ingeniería de software**, el **ciclo de vida** de un **sistema** y algunos **modelos de desarrollo de software.**

¿Sistema?



Sistema

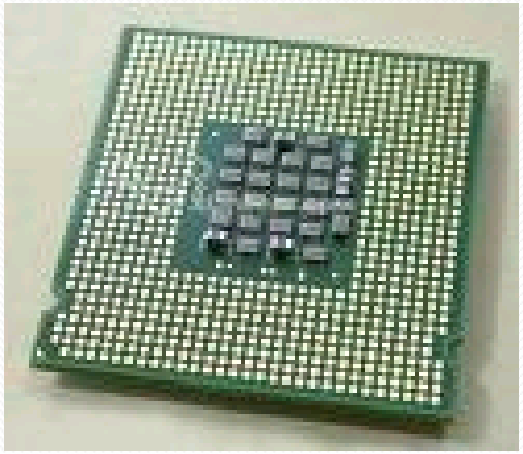
- Un sistema es un conjunto de elementos organizados que interactúan entre sí y con su ambiente, para lograr objetivos comunes, operando sobre información, para producir como salida información.

Clasificación de sistemas

- Sistemas hechos por el hombre: Construidos, organizados y mantenidos por humanos, tales como:
  - Sistemas sociales
  - Sistemas de transporte
  - Sistemas de comunicación
  - Sistemas de manufactura
  - Sistemas financieros, etcétera.

¿Información? ¿Datos?

- La información es un conjunto organizado de datos



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 ... 254	255
1 Bit	0	1															
2 Bit	0	0	1	1													
	0	1	0	1													
4 Bit	0	0	0	0	1	1	1	1	0	0	0	1	0	1	1	1	
	0	0	0	1	1	1	0	0	1	0	1	0	1	1	0	1	
	0	0	1	1	1	0	0	0	1	1	0	1	0	0	1	1	
	0	1	1	1	0	0	0	1	0	0	0	0	1	1	1	1	
8 Bit	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1 ... 1	1
	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0 ... 0	1
	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0 ... 1	1
	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0 ... 1	1
	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0 ... 1	1
	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0 ... 1	1
	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0 ... 1	1
	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0 ... 1	1
	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1 ...	

Sistema de Información

- Conjunto de funciones y procedimientos encaminadas a la captación, desarrollo, recuperación, almacenamiento, etc., de información en el seno de una organización.
- Conjunto de elementos que interactúan entre sí con el fin de apoyar las actividades de una empresa o negocio.

Sistema de Información

- Un sistema de información realiza cuatro actividades básicas:
  - Entrada
  - Almacenamiento
  - Procesamiento
  - Salida de información.

Ingeniería de Software

- IEEE: "Standar Glossary of Software Engineering Terminology“.
- Enfoque sistemático para el desarrollo de operación, mantenimiento y eliminación del software, donde software se define como aquellos programas, procedimientos, reglas y documentación posible asociada con la computación, así como los datos pertenecientes a la operación de un sistema de cómputo.

III. Productos de Software

- Productos genéricos.
  - Productos que son producidos por una organización para ser vendidos al mercado.
- Productos hechos a medida.
  - Sistemas que son desarrollados bajo pedido a un desarrollador específico.
- La mayor parte del gasto del software es en productos genéricos, pero hay más esfuerzo en el desarrollo de los sistemas hechos a medida.

Características de los Productos de Software

- Mantenibles.
  - Debe ser posible que el software evolucione y que siga cumpliendo con sus especificaciones.
- Confiabilidad.
  - El software no debe causar daños físicos o económicos en el caso de fallos.
- Eficiencia.
  - El software no debe desperdiciar los recursos del sistema.
- Utilización adecuada.
  - El software debe contar con una interfaz de usuario adecuada y su documentación.

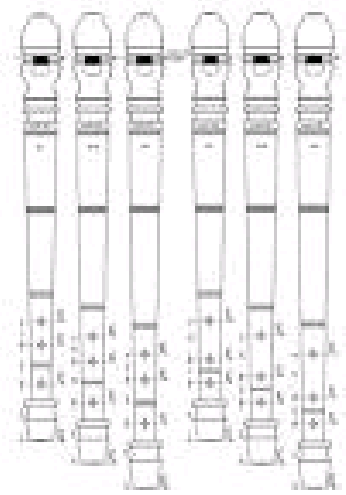
V. Modelo de Ingeniería del Proceso

- **Requisitos**
  - Capturar los aspectos funcionales correspondientes, cómo un usuario interactuaría con el sistema.
- **Análisis**
  - Dar al sistema una estructura robusta y extensible bajo un ambiente de implementación ideal.
- **Diseño**
  - Adoptar y refinar las estructuras al ambiente de implementación particular.
- **Implementación**
  - Codificar el sistema.
- **Pruebas**
  - Validar y verificar el sistema.
- **Integración**
  - Pegar componentes del sistema.
- **Documentación**
  - Describir los distintos aspectos el sistema.
- **Mantenimiento**
  - Extender la funcionalidad del sistema.

VI. Modelos de Desarrollo de Software

¿ Modelo ?

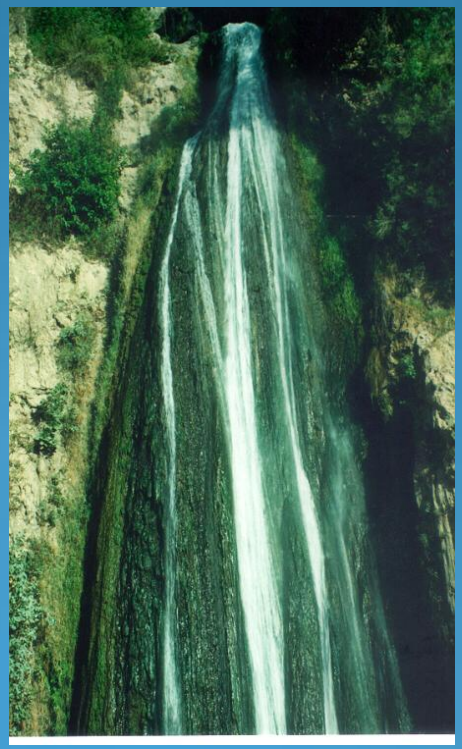
- Representación formal o simplificada del proceso de software.



Modelos Genéricos del Desarrollo de Software

- **Cascada (Lineal Secuencial)**
- **Prototipado**
- **DRA (Reutilización)**
- **Evolutivo (Incremental)**
- **Espiral**
- Espiral Win / Win
- Basado en componentes
- Programación Extrema
- Proceso unificado de desarrollo de software (PUDS)

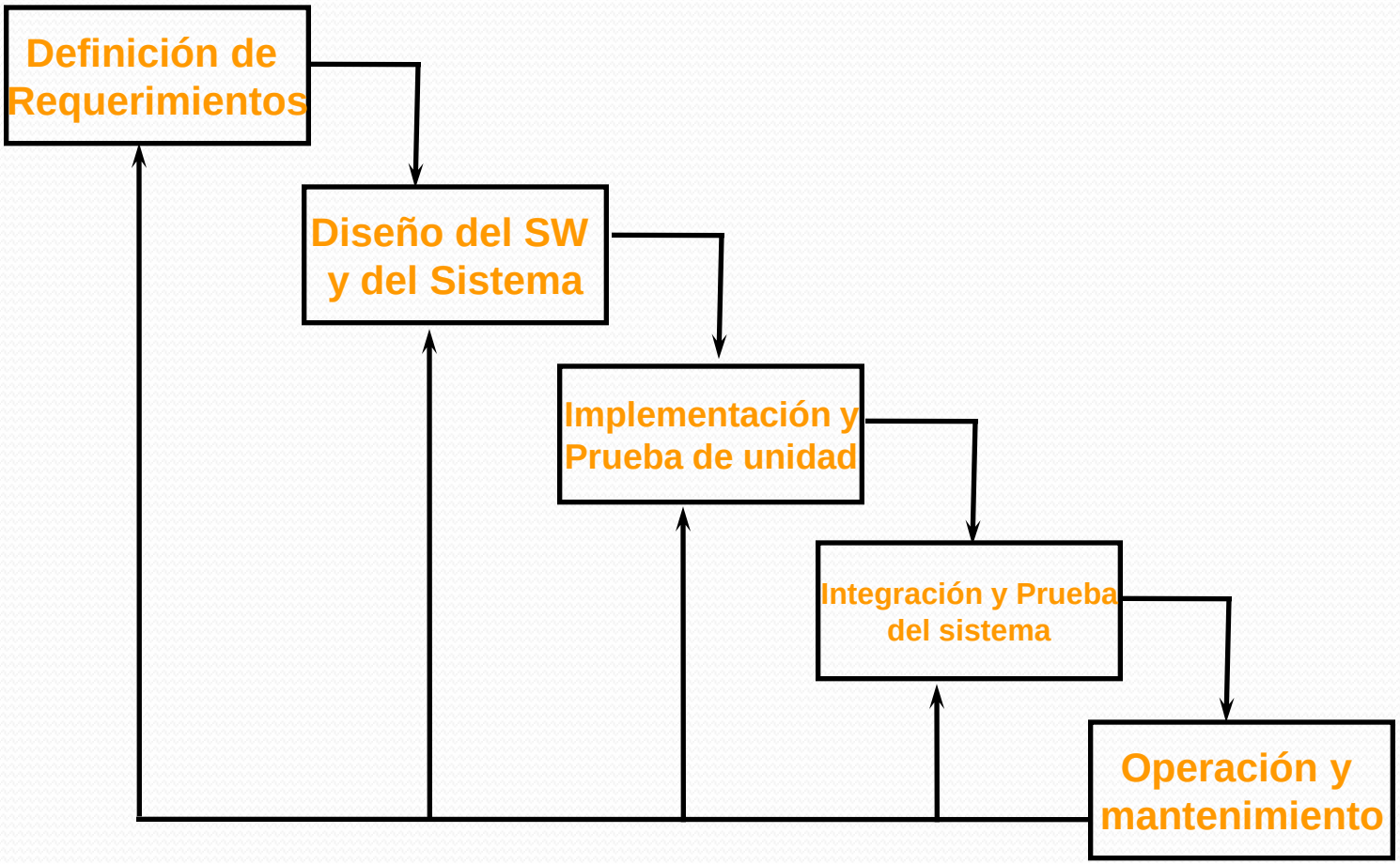
Lineal secuencial, clásico o cascada.



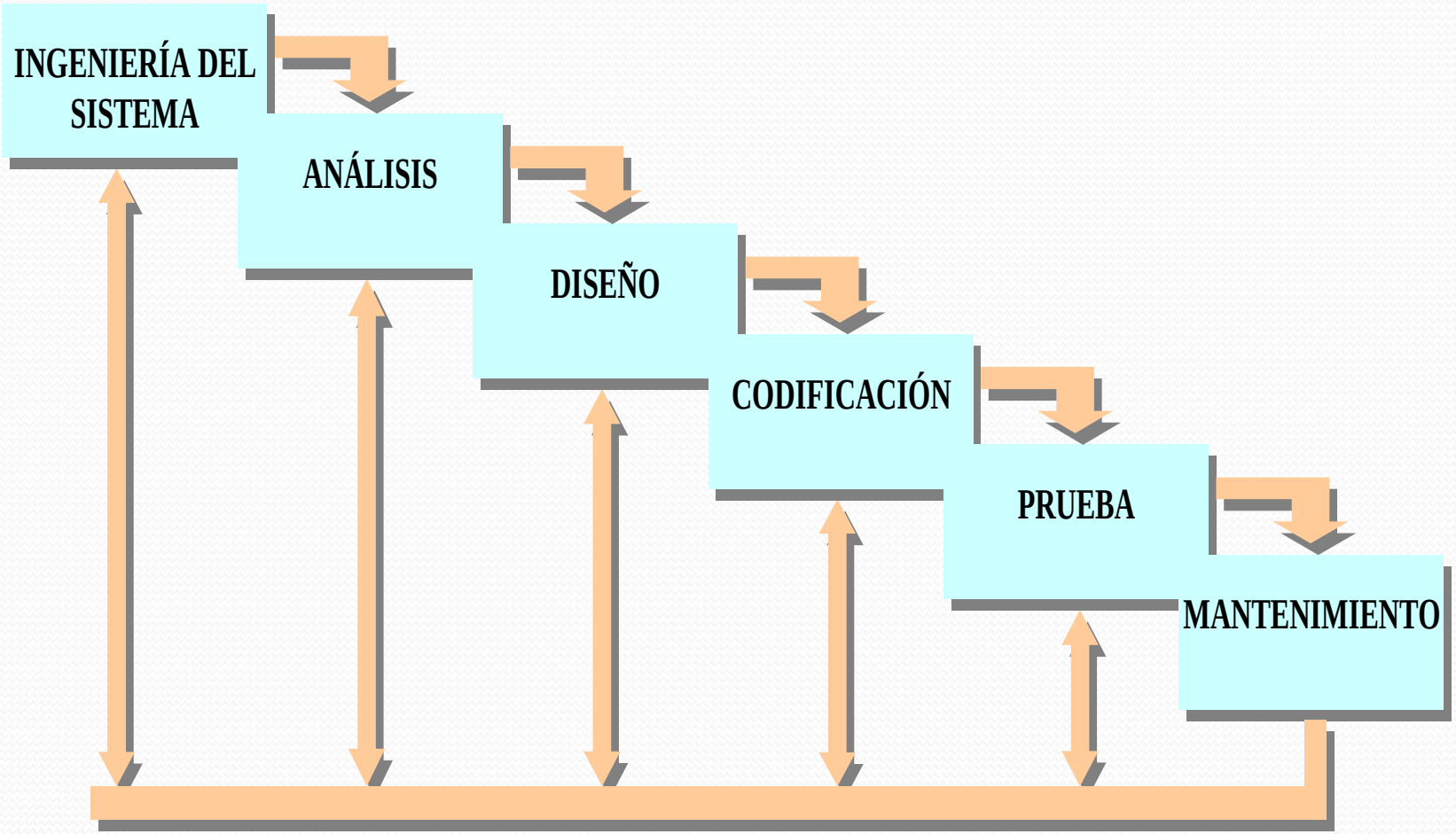
Modelo lineal secuencial

- Las tareas se organizan en cascada, una después de la otra.
- La salida de una etapa es la entrada de la siguiente.
  - Creado en 1970 por el Departamento de la Defensa de EE.UU.
  - Su usó en las primeras aplicaciones es crítica y compleja.

Modelo lineal secuencial



Paradigma clásico. Definición alternativa



Ventajas

- Introduce disciplina al proceso.
- Pospone la implementación hasta que los objetivos estén claros.
- Es el paradigma más usado y conocido en la industria del software.
- Impone puntos de control claros.
- Costo de producción del producto.
- Es un modelo conducido con documentación .

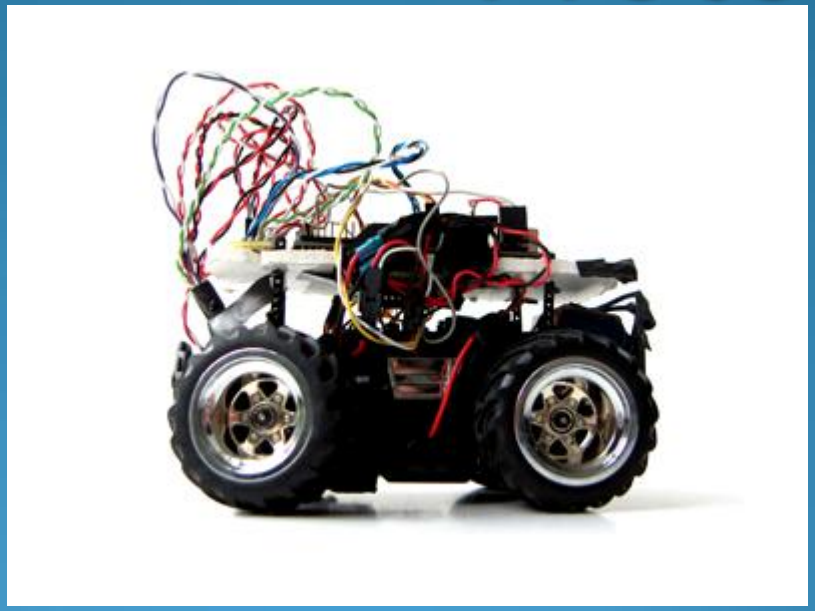
Desventajas

- Los proyectos raramente siguen el flujo secuencial.
- El cliente no puede explicitar inicialmente todos los requisitos
- No existe una versión operativa hasta el final
- Dificultad de hacer cambios entre etapas.
- Alto riesgo en sistemas nuevos debido a problemas en las especificaciones y en el diseño.
- Bajo riesgo para desarrollos bien comprendidos utilizando tecnología conocida.
- La dificultad en esta modelo reside, en la dificultad de hacer cambios entre etapas.
- El cliente debe tener paciencia para ver los resultados.

Ejemplo de aplicación

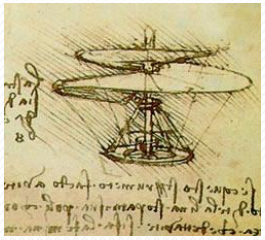
- Algunas aplicaciones para implementar el modelo lineal secuencial:
  - Problemas medianos / chicos.
  - Problemas con requerimientos estables.
  - Reingeniería de sistemas.
- Es recomendable cuando los requerimientos están totalmente comprendidos.

Prototipos



¿ Qué es un prototipo?

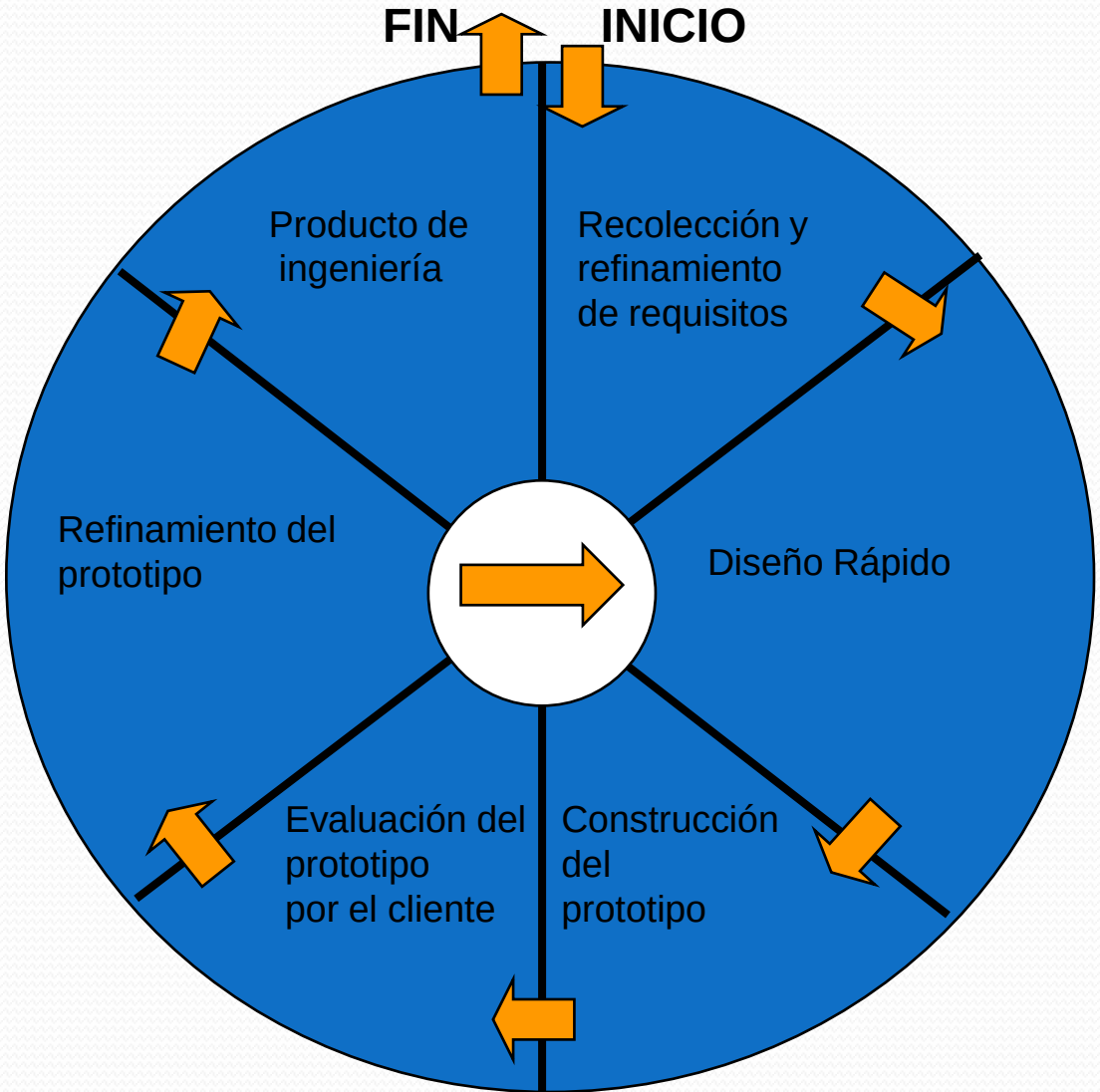
- Un prototipo es una representación limitada del diseño de un producto que permite a las partes responsables de su creación experimentar su uso, probarlo en situaciones reales y explorar su uso.
- Un prototipo puede ser cualquier cosa, desde un trozo de papel con sencillos dibujos a un complejo software.



Modelo de prototipos

- Su principal objetivo es la obtención de requerimientos y diseño.
- Reducir el riesgo de que la aplicación no se ajuste a las necesidades del cliente.
- De la revisión de cada etapa se pueden revisar las etapas anteriores.

Modelo de prototipos



Fases del Modelo

1. Recolección y refinamiento de requisitos.
2. Diseño rápido
3. Construcción del prototipo
4. Evaluación del prototipo por el cliente.
5. Refinamiento del prototipo.
6. Producto de Ingeniería

Ventajas

- El prototipo se usa para obtener los requerimientos del usuario.
- Su principal propósito es obtener y validar los requerimientos esenciales, manteniendo abiertas las opciones de implementación. Esto implica que se deben tomar los comentarios de los usuarios, pero también se debe volver a los objetivos para no perder la atención.
- Bajo riesgo para nuevas aplicaciones debido a que las especificaciones y el diseño se llevan acabo paso a paso.

Desventajas

- El cliente ve el prototipo y lo confunde con el sistema real.
- El cliente no entiende la necesidad de volver a hacerlo.
- El equipo de desarrollo toma decisiones rápidas para poder construir el prototipo , qué mas tarde son difíciles de revertir (por ejemplo el lenguaje de programación).
- Alto riesgo debido a falta de visibilidad

Ejemplo de aplicación

- Este modelo de desarrollo es recomendable utilizarlo cuando se trata con clientes que tienen poca experiencia en el desarrollo de sistemas de información que no definen claramente lo que quieren.
- Es recomendable cuando existen grandes riesgos técnicos.

Desarrollo Rápido de Aplicaciones

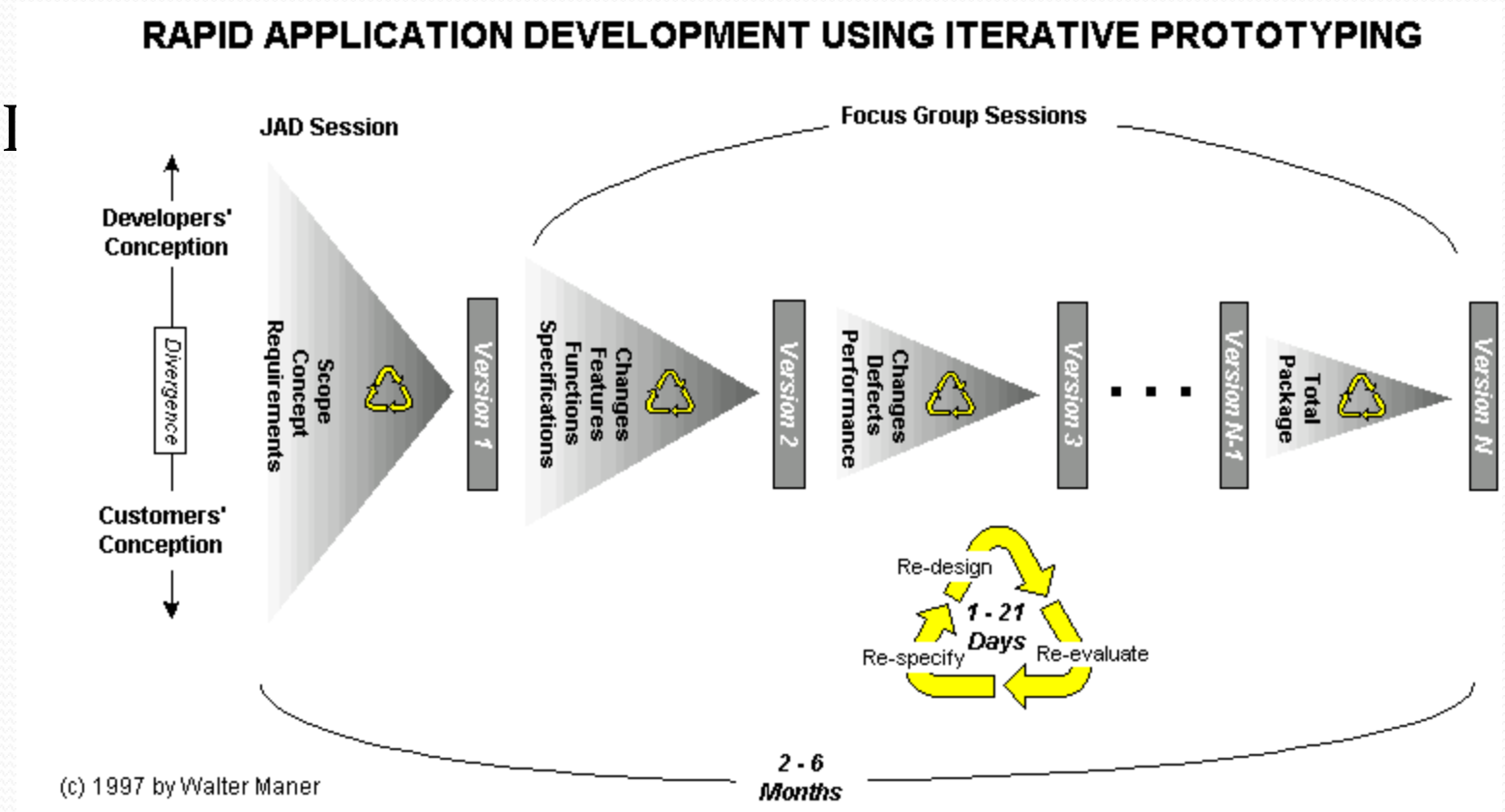
DRA

DRA

- Proceso de desarrollo de software que permite construir sistemas utilizables en poco tiempo, normalmente de 60 a 90 días, bajo ciertas restricciones.
- Prevenir presupuestos rebasados
- Prevenir incumplimiento de fechas
- Llegar a un acuerdo temprano entre cliente y desarrolladores con respecto al SW.

DRA

- I


RAPID APPLICATION DEVELOPMENT USING ITERATIVE PROTOTYPING
JAD Session
Focus Group Sessions
Developers' Conception
Divergence
Customers' Conception
Scope
Concept
Requirements
Version 1
Changes
Features
Functions
Specifications
Version 2
Changes
Defects
Performance
Version 3
Version N-1
Total
Package
Version N
Re-design
1 - 21 Days
Re-specify
Re-evaluate
2 - 6 Months
(c) 1997 by Walter Maner

Fases DRA

- Modelado de Gestión
- Modelado de datos
- Modelado de proceso
- Generación de aplicaciones
- Prueba y entrega

Ventajas

- Velocidad del desarrollo: Los aumentos de la velocidad son debido al uso de la herramienta CASE.
- Calidad: según lo definido por el RAD, es el grado al cual un uso entregado resuelve las necesidades de usuarios así como el grado al cual un sistema entregado tiene costes de mantenimiento bajos. El RAD aumenta calidad con la implicación del usuario en las etapas del análisis y del diseño.

Desventajas

- Características reducidas.
- Escalabilidad reducida: debido a que el RAD se desarrolló como prototipo.

Ejemplo de aplicación

- Se puede utilizar cuando se desarrolla:
  - Una aplicación no crítica.
  - Una aplicación independiente.
  - Alcance del proyecto limitado.
  - Distribución limitada.
  - Utilizando bibliotecas preexistentes.

Herramientas RAD

- Microsoft Visual Studio
- NetBeans
- Microsoft Visual Basic
- Clarion
- Oracle
- Coldfusion

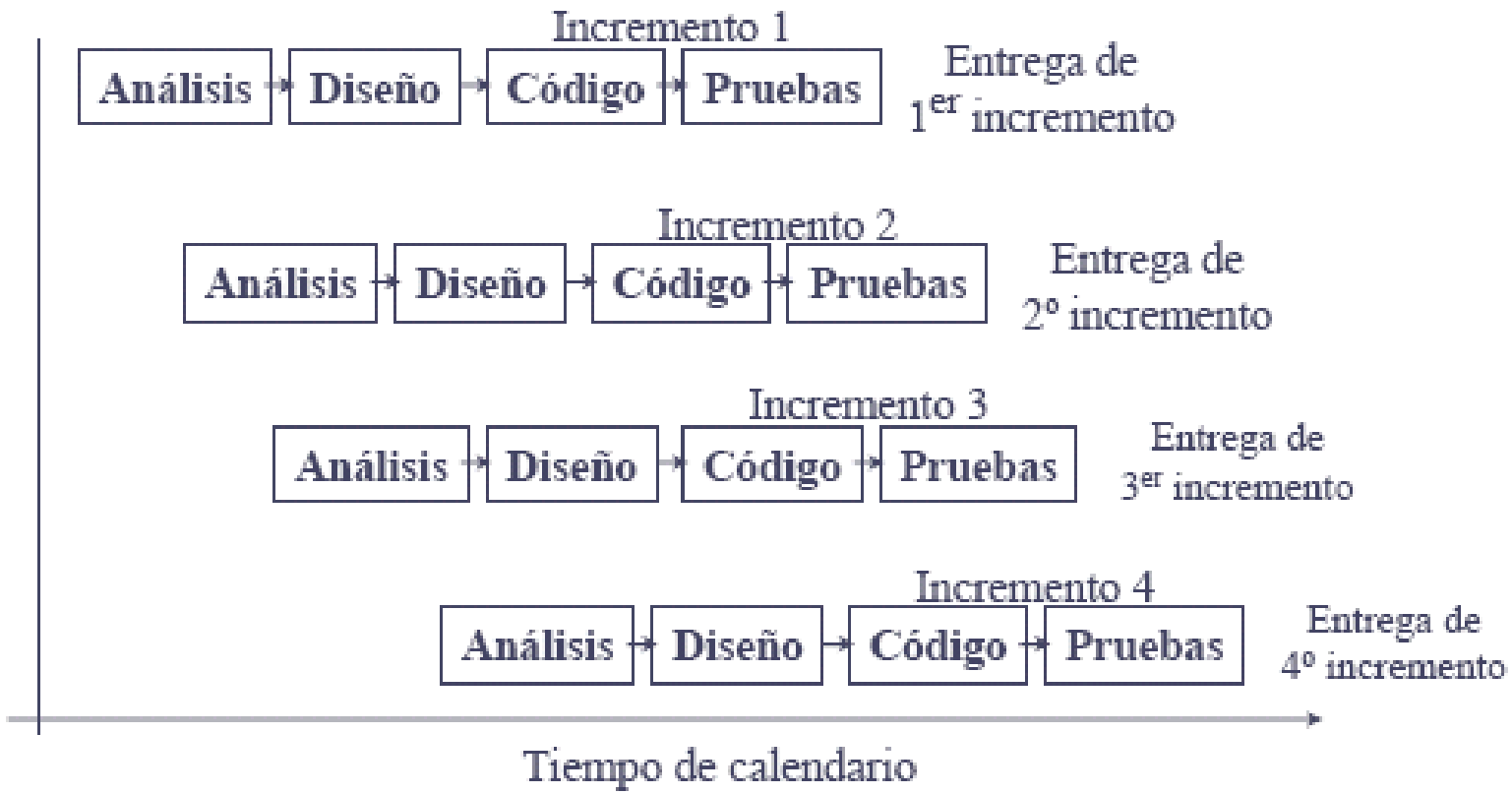
Incremental

Modelo evolutivo

Incremental

- Es un modelo cuyas etapas consisten en incrementos expandidos de un producto de software operativo, conducidos por la evolución determinada según la experiencia operativa.
- Los incrementos se distribuyen a medida que se complementan.
- Incremento integrado: es una unidad de software autocontenida que realiza algún propósito útil.

Incremental



Fases Incremental

- Son las mismas que en modelo en cascada.

Ventajas

- Se puede financiar el proyecto por partes.
- Apropiado para proyectos grandes de larga duración.
- No se necesita tanto personal al principio como para una implementación completa

Desventajas

- Para proyectos grandes requiere recursos humanos suficientes.
- Los clientes y desarrolladores deben estar comprometidos en las rápidas actividades.
- Si el sistema no se puede modularizar será problemático.
- No es adecuado con riesgos técnicos altos.
- Se pueden aumentar los costos debido a pruebas.

Ejemplo de aplicación

- Apropiado para proyectos grandes de larga duración.

Espiral

- El modelo en espiral trata de mejorar los ciclos de vida clásicos.
- Incorpora objetivos de calidad y gestión de riesgos.
- Permite iteraciones, vuelta atrás y finalizaciones rápidas.
- Cada ciclo empieza identificando:
  - Los objetivos de la porción correspondiente
  - Las alternativas
  - Restricciones
- Cada ciclo se completa con una revisión que incluye todo el ciclo anterior y el plan para el siguiente.

Espiral

- El modelo combina las actividades del desarrollo con la gestión de riesgos , para minimizar y controlar el riesgo.
- Riesgo: circunstancias potencialmente adversas que pueden perjudicar el proceso de desarrollo y la calidad de los productos.
- Administración del riesgo: disciplina cuyos objetivos son manejar y eliminar items del riesgo del software antes que se transformen en amenaza.

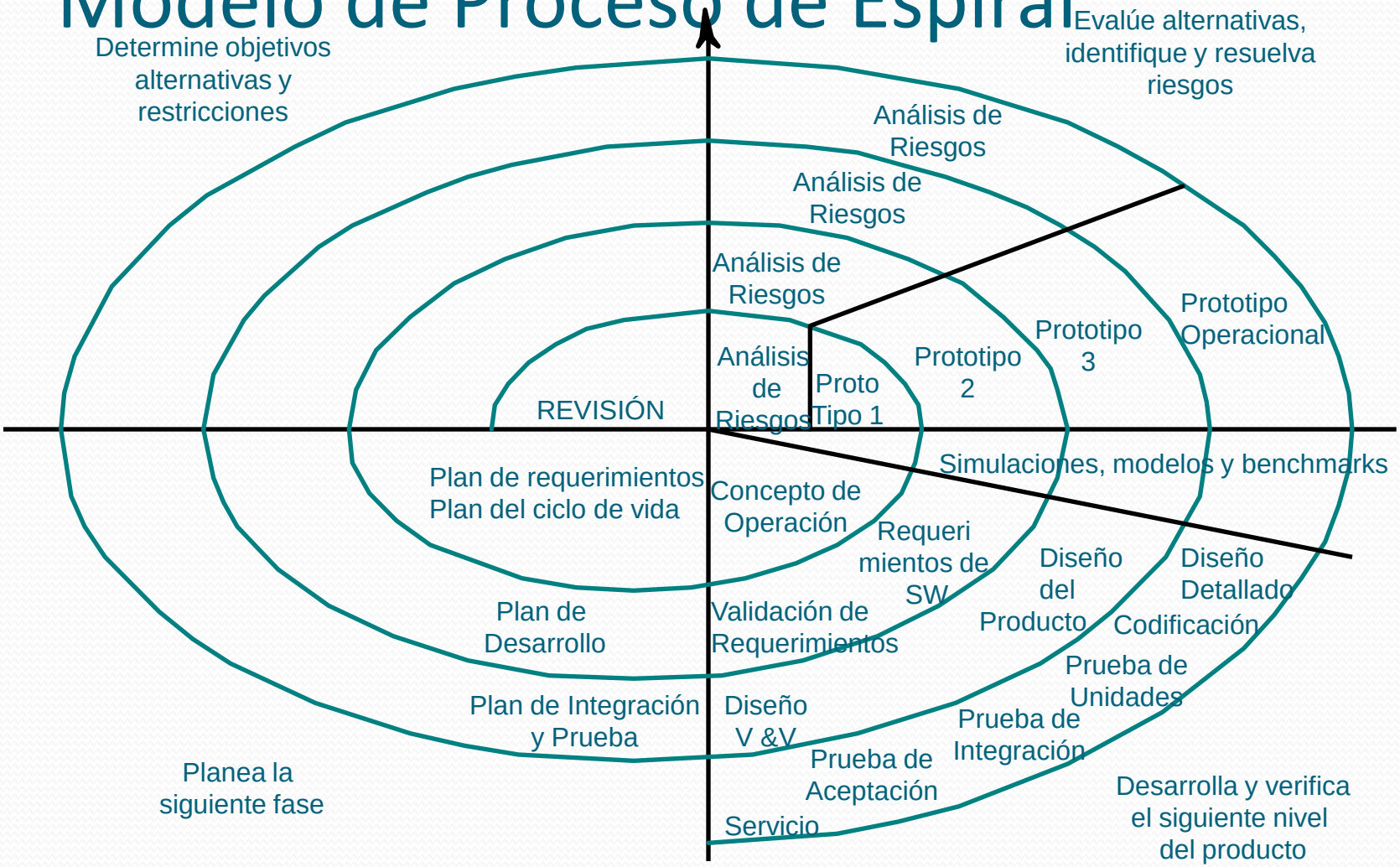
Espiral

- 1era Etapa
  - Identifica los objetivos de la porción del producto bajo consideración, en términos de la calidad a lograr.
  - Plantear alternativas: comprar, diseñar, reusar y las restricciones a dichas alternativas.
- 2da Etapa
  - Se evalúan las alternativas y se identifican las áreas de riesgo potencial.
  - La evaluación del riesgo puede requerir distintas clases de actividades a ser planificadas (simulación, prototipos)

Espiral

- 3era Etapa
  - Consiste del desarrollo y verificación del próximo nivel del producto. El proceso está conducido por el riesgo.
- 4ta Etapa
  - Se revisan los resultados para planificar la próxima vuelta en espiral.

Modelo de Proceso de Espiral



Fases Espiral

- **Comunicación con el cliente**: Permite establecer comunicación entre el desarrollador y el cliente.
- **Planificación**: Para definir los recursos , el tiempo y otras informaciones relacionadas con el proyecto.
- **Análisis de riesgos**: Para evaluar riesgos técnicos y operativos.
- **Ingeniería**: Para construir una o más presentaciones de la aplicación.
- **Construcción y adaptación**: Para construir, probar, instalar y proporcionar soporte al usuario.
- **Evaluación del cliente**: Para obtener la reacción del cliente según la evaluación de las representaciones del software.

Ventajas

- El análisis del riesgo se hace de forma explícita y clara.
- Une los mejores elementos de los restantes modelos.
- Reduce riesgos del proyecto.
- Incorpora objetivos de calidad.
- Integra el desarrollo con el mantenimiento.
- Constituye un enfoque realista del desarrollo de sistemas de software de gran escala. En la medida que progresa cliente y desarrollador comprenden y manejan mejor los riesgos.
- Considera los riesgos técnicos en todas sus etapas.

Ventajas

- Centra su atención en la reutilización de componentes y eliminación de errores en información descubierta en fases iniciales.
- Los objetivos de calidad son el primer objetivo.
- Integra desarrollo con mantenimiento.
- Provee un marco de desarrollo de hardware/software.

Desventajas

- Genera mucho tiempo en el desarrollo del sistema.
- Modelo costoso.
- Requiere experiencia en la identificación de riesgos.
- No existe demasiada experiencia en su uso.

Desventajas

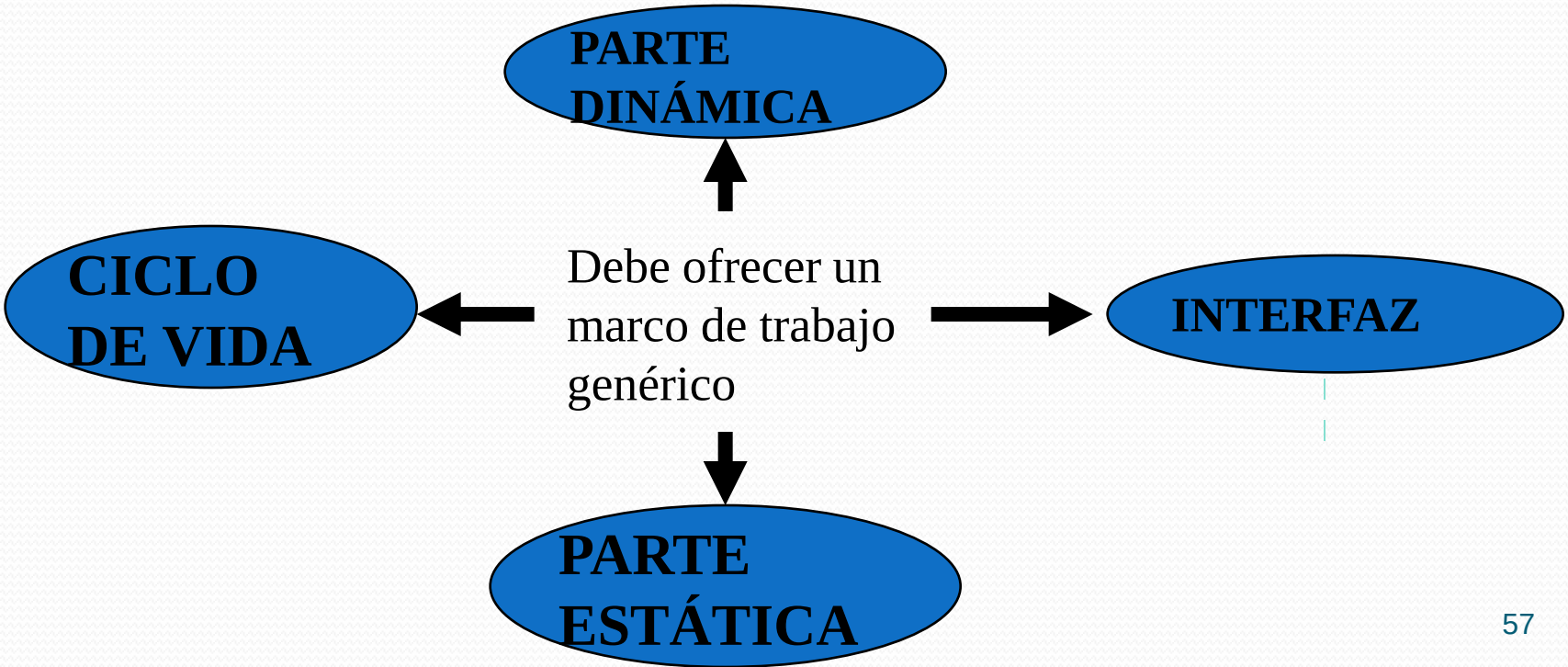
- El desarrollo contractual especifica el modelo del proceso y los resultados a entregar por adelantado.
- Requiere de experiencia en la identificación de riesgos.
- Requiere refinamiento para uso generalizado.

Introducción al proceso unificado de desarrollo de software

PUDS

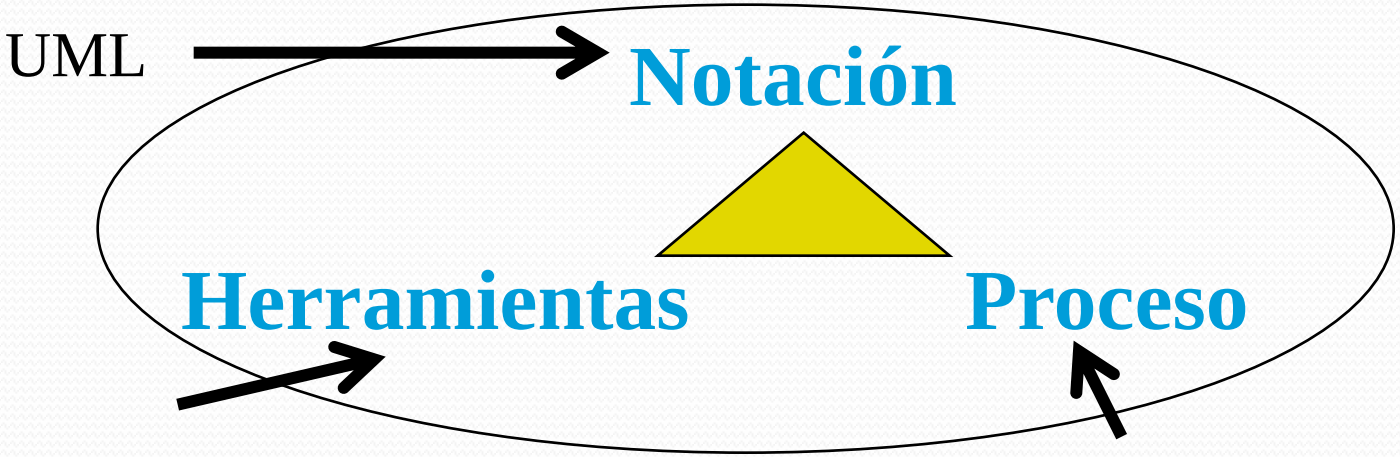
El proceso unificado de desarrollo de software

- Es un proceso ORIENTADO A OBJETOS
- El proceso es:
  - Guiado por *casos de uso*
  - Centrado en la *arquitectura*
  - Con un ciclo de vida *iterativo* e *incremental*



El proceso unificado de desarrollo de software

- El Proceso Unificado de Desarrollo usa UML



- RATIONAL ROSE
- VISIO

PROCESO UNIFICADO DE DESARROLLO DE RATIONAL

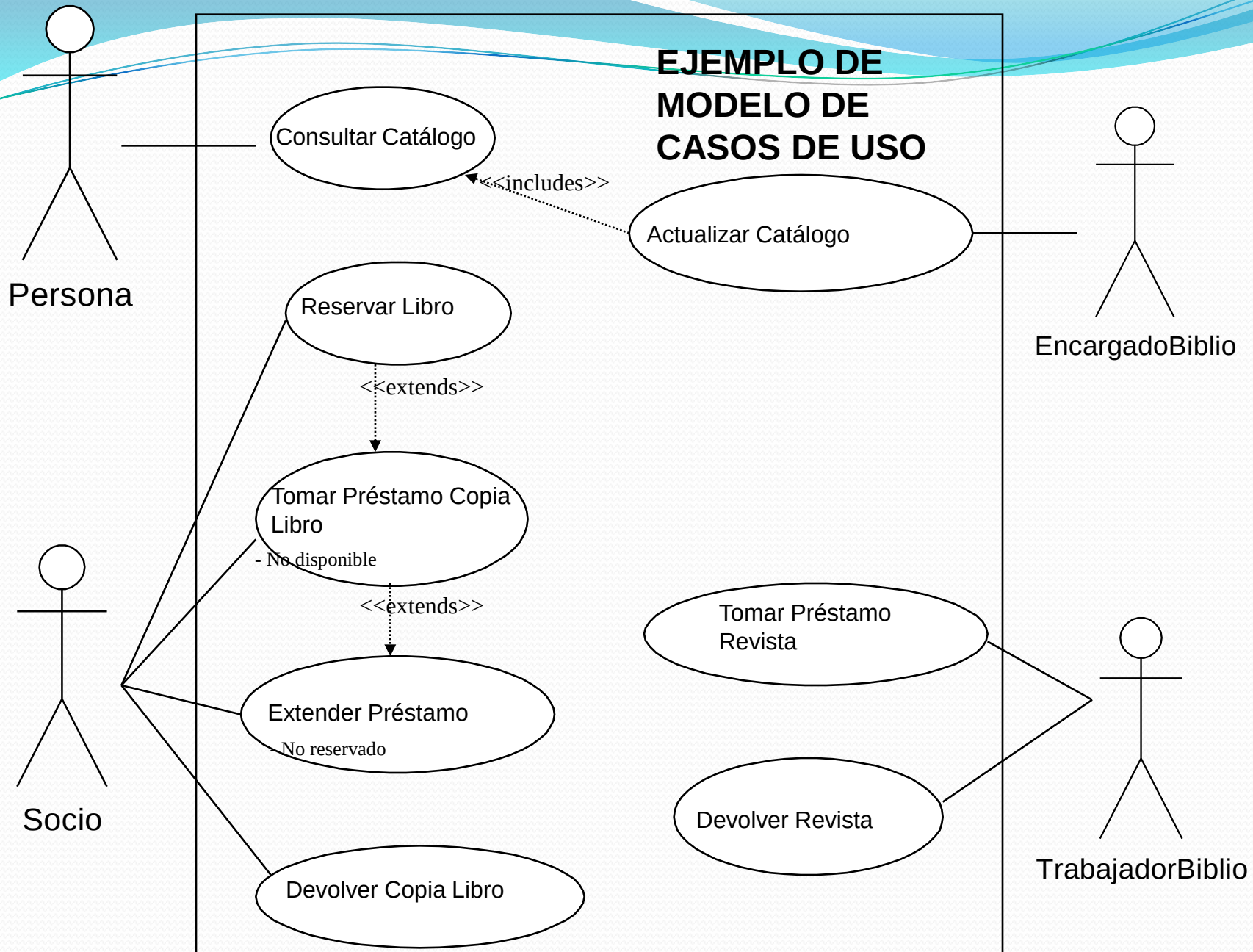
1. Guiado por *casos de uso*

- Los sistemas se crean para dar servicio a los **usuarios.**
  - Qué REQUISITOS se necesitan
  - Un CASO de USO es una pieza de **FUNCIONALIDAD** de un sistema que le proporciona a algún **USUARIO** un **RESULTADO o VALOR**.

Casos de uso

- Todos juntos constituyen el **modelo de casos de uso (MCU)**
  - FUNCIONALIDAD COMPLETA
  - PARA TODOS LOS USUARIOS

EJEMPLO DE MODELO DE CASOS DE USO

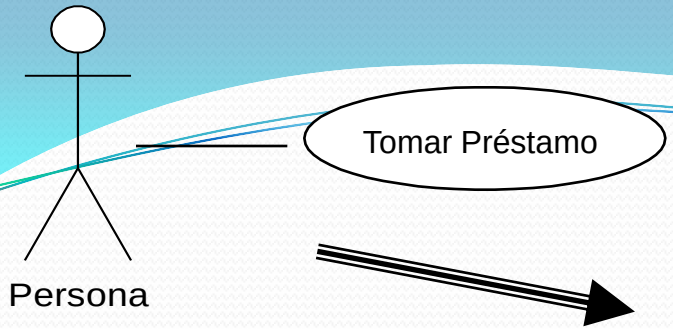


Desarrollo guiado por casos de uso (CU)

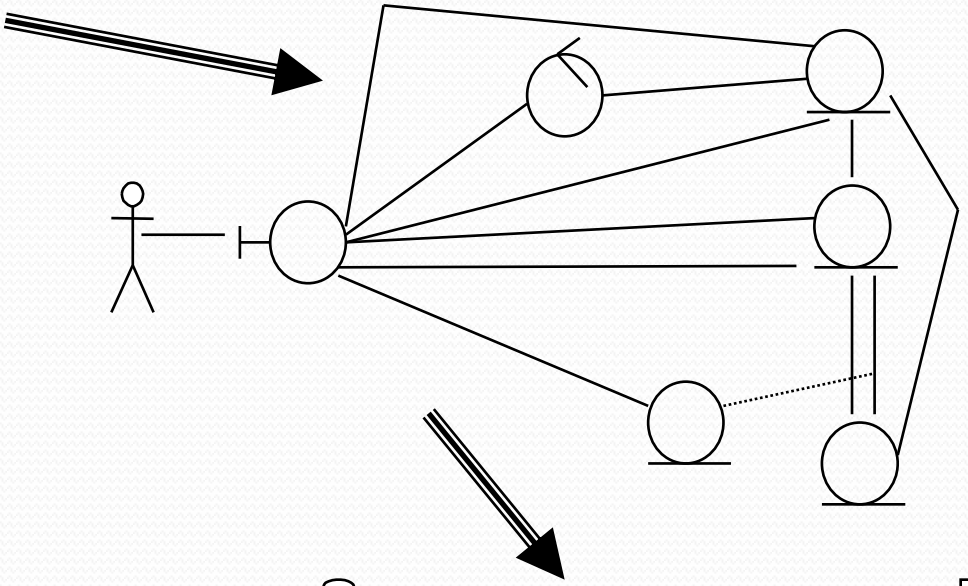
LOS CASOS DE USO:

- CAPTURAN REQUISITOS
- SE ESPECIFICAN (ANALIZAN)
- SE DISEÑAN
- SE IMPLEMENTAN
- Y SE PRUEBAN

Desarrollo guiado por CASOS DE USO

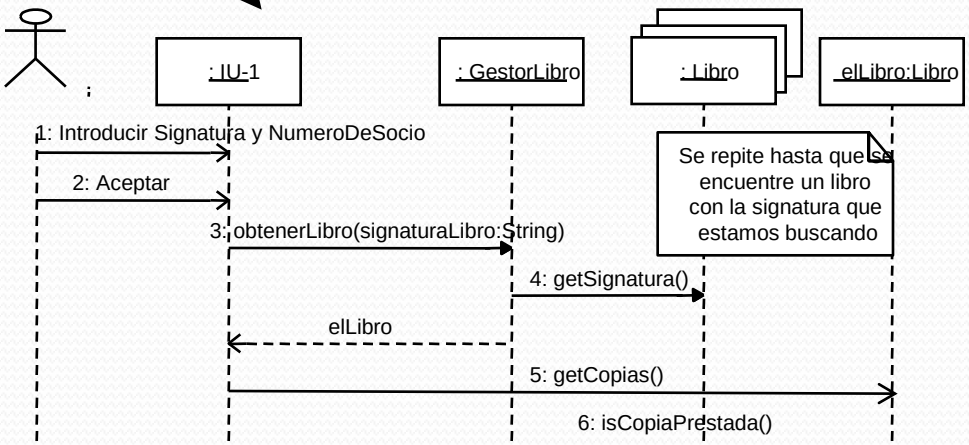


1.- CASO DE USO



2.- ANÁLISIS DEL CASO DE USO

3.- DISEÑO DEL CASO DE USO



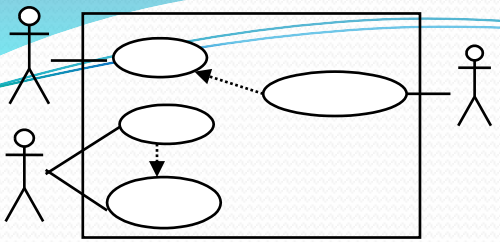
4.- IMPLEMENTACIÓN DEL CASO DE USO

5.- PRUEBA DEL CASO DE USO

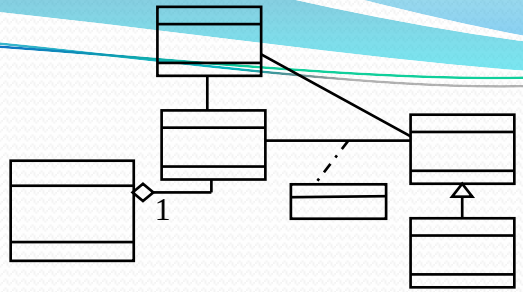
2. Centrado en la arquitectura

- La **arquitectura** de un sistema software es un extracto de los modelos del sistema
  - Extracto: VISTA DE CADA MODELO
- que da una idea de qué **forma** que tiene el **sistema** completo

Centrado en la ARQUITECTURA

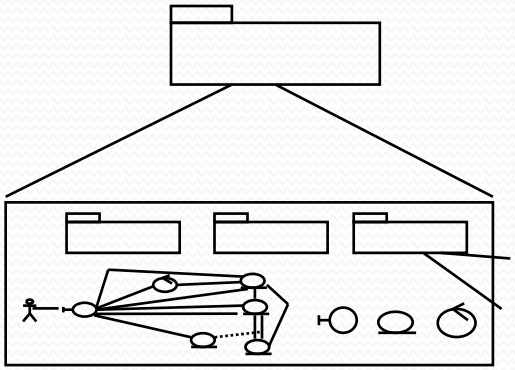


VISTA DEL MODELO DE CASOS DE USO

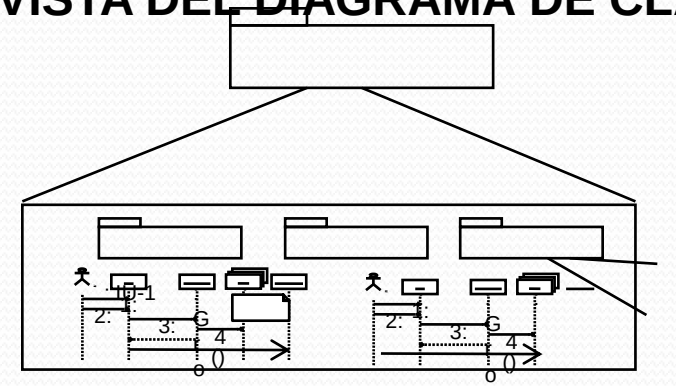


VISTA DEL MODELO DEL DOMINIO /

VISTA DEL DIAGRAMA DE CLASES



VISTA DEL MODELO DEL ANÁLISIS



VISTA DEL MODELO DEL DISEÑO

+ VISTAS DEL MODELO DE IMPLEMENTACIÓN Y PRUEBAS

SON VISTAS DE LOS MODELOS (NO MODELOS COMPLETOS).

SÓLO APARECEN LOS QUE CORRESPONDEN

3. Ciclo de vida *iterativo* e *incremental*

- ITERATIVO
  - Se repiten VARIOS MINIPROYECTOS
- INCREMENTAL
  - Cada miniproyecto AMPLIA EL PRODUCTO

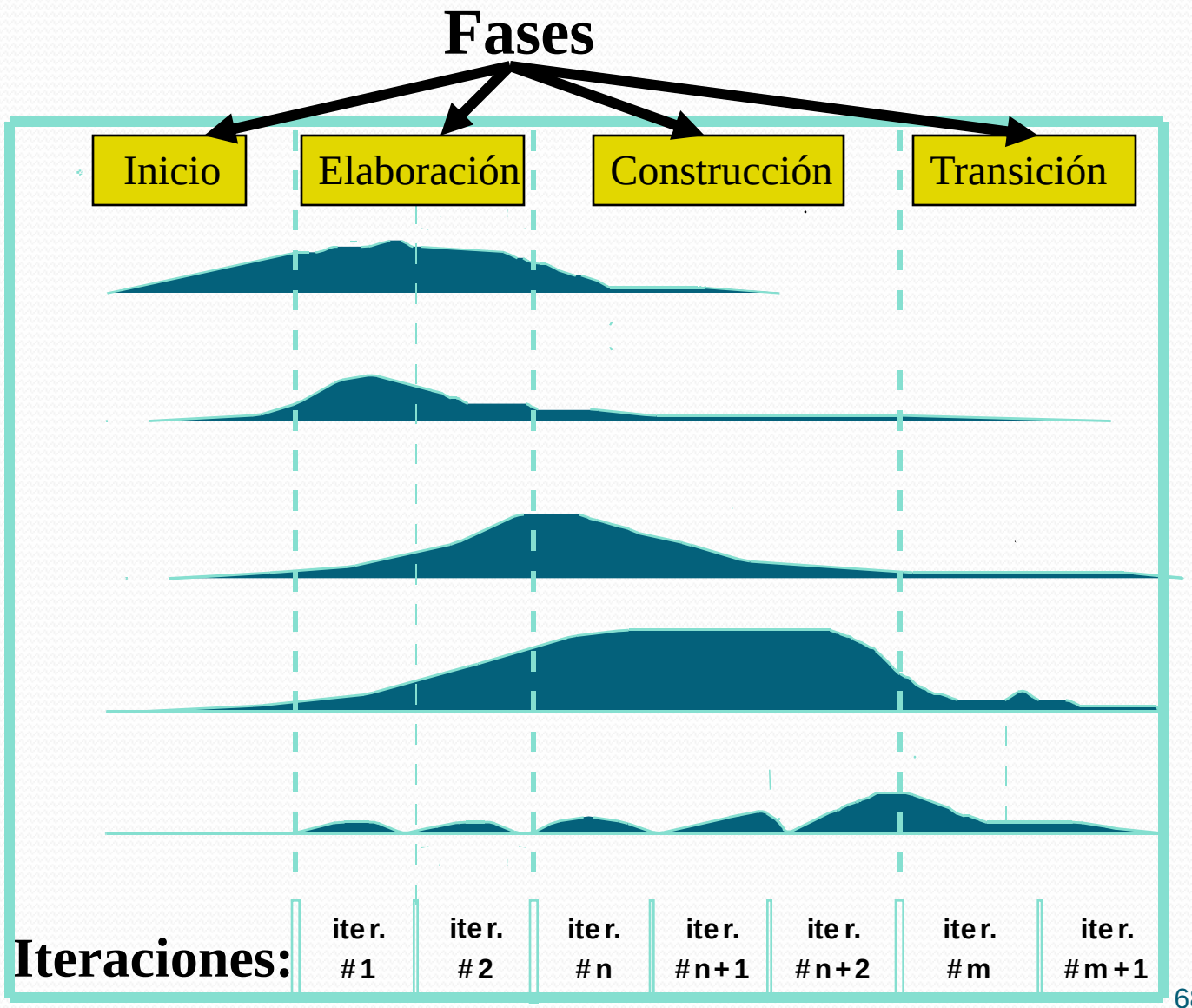
El CV del proceso unificado

- UN CICLO DE VIDA SE REPITE A LO LARGO DEL TIEMPO
- TRAS CADA CICLO DE VIDA ➔ VERSIÓN NUEVA DEL PRODUCTO
- UN CICLO DE VIDA SE DIVIDE EN FASES
- CADA FASE SE DIVIDE EN ITERACIONES
- EN CADA ITERACIÓN SE REALIZAN FLUJOS DE TRABAJO

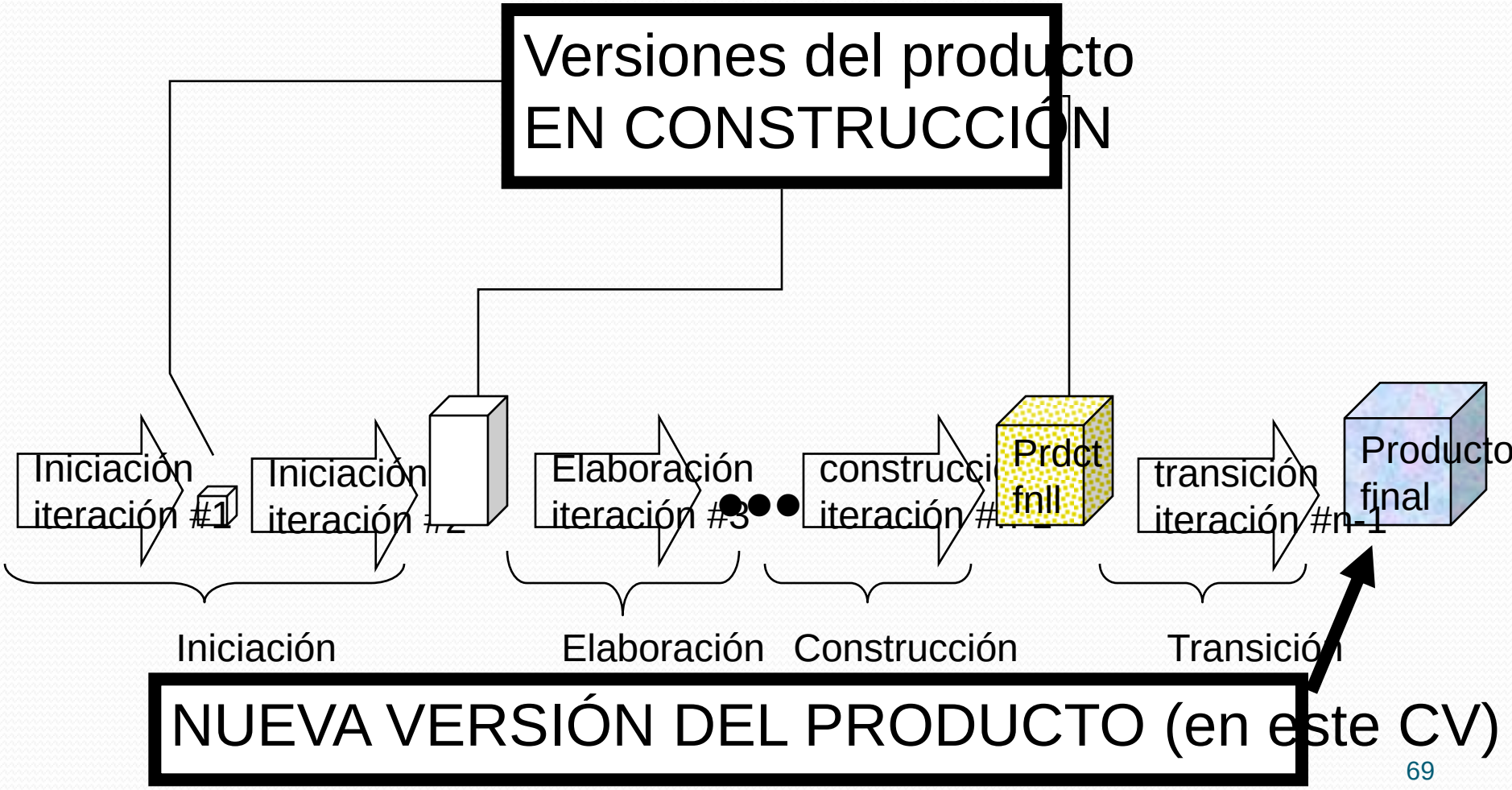
El CV del proceso unificado

Flujos de trabajo: Actividades

- Requisitos
- Análisis
- Diseño
- Implementación
- Prueba



El CV del proceso unificado



El producto
(del proceso unificado)

- NO ES SÓLO CÓDIGO EJECUTABLE
- SON LOS MODELOS O REPRESENTACIÓN DEL SOFTWARE
- DEBE AJUSTARSE A TODAS LAS PERSONAS IMPLICADAS

Fases dentro del CV del proceso unificado

- FASE: PARTE DE UN CV
- CADA FASE TERMINA EN UN HITO
  - HAY ARTEFACTOS DISPONIBLES (SEGÚN LO PLANIFICADO)
  - LOS RESULTADOS EN LOS HITOS PERMITEN GESTIONAR

Fases dentro del CV del proceso unificado

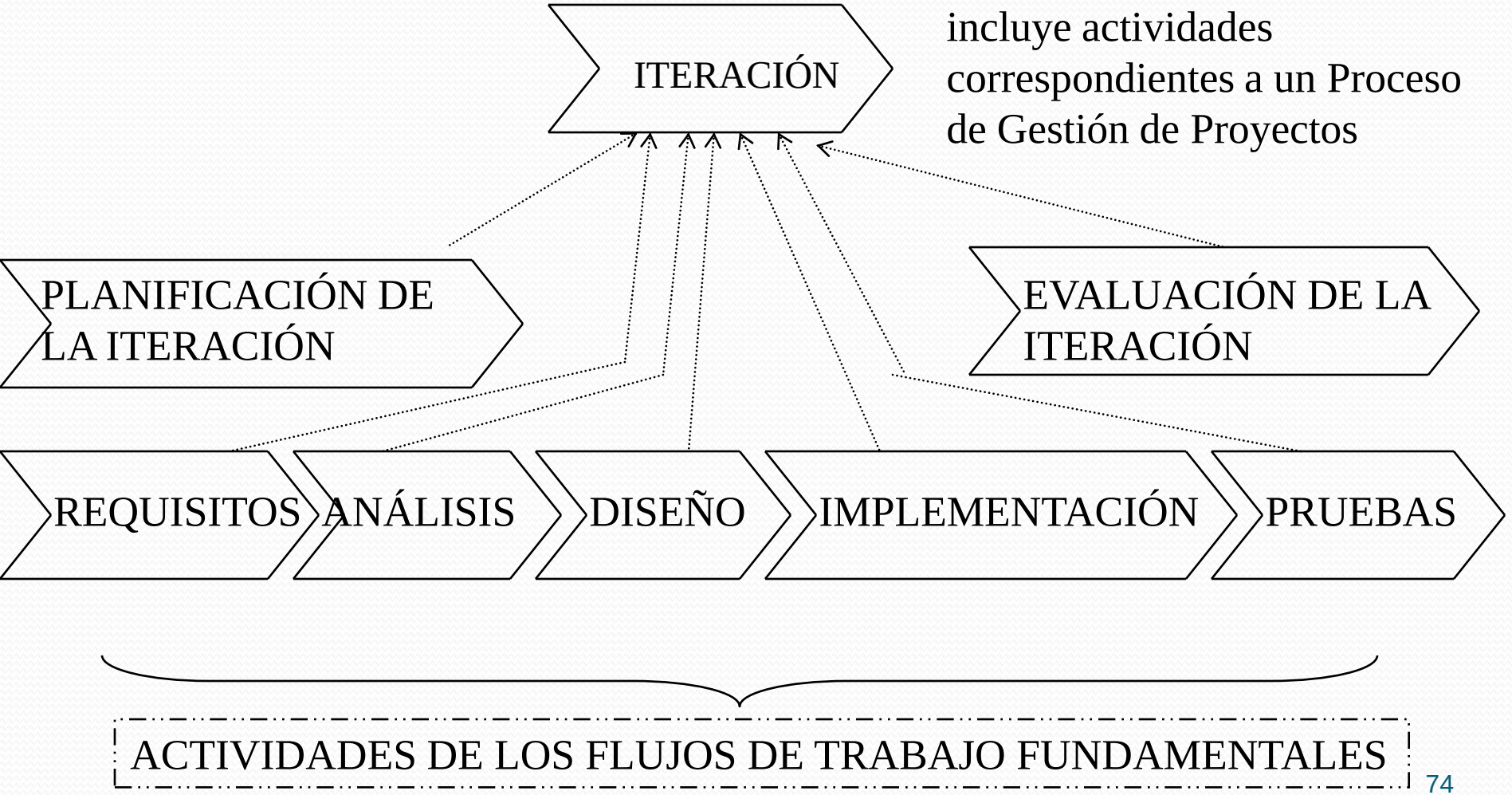
- **INICIACIÓN:**
  - DESCRIBIR PRODUCTO FINAL / ANÁLISIS DEL NEGOCIO
  - IDENTIFICAR RIESGOS MÁS IMPORTANTES
  - ESTABLECER PLANIFICACIÓN INICIAL DEL PROYECTO
  - DECIDIR SI SE CONTINÚA
- **ELABORACIÓN:**
  - ESTABLECER PLAN Y ARQUITECTURA ESTABLE
- **CONSTRUCCIÓN:** DESARROLLAR EL PRODUCTO
- **TRANSICION:** PROPORCIONAR SISTEMA A USUARIOS

Iteraciones

- CADA FASE SE DIVIDE EN ITERACIONES
- CADA ITERACIÓN
  - MINIPROYECTO (EN CASCADA) QUE EJECUTA FLUJOS DE TRABAJO
  - PRODUCE UN INCREMENTO EN PRODUCTO
    - TAL Y COMO ESTABA
- SE REDUCE EL RIESGO
  - SE PUEDE PERDER SÓLO LO REALIZADO EN ESA ITERACIÓN

Iteraciones

Como se puede ver, el Proceso Unificado de Desarrollo incluye actividades correspondientes a un Proceso de Gestión de Proyectos



Flujos de trabajo

- **CAPTURA DE REQUISITOS:**
  - IDENTIFICAR REQUISITOS DEL SISTEMA
  - CONSTRUIR UN MODELO DEL MISMO
    - MODELO DE CASOS DE USO
    - MODELO DEL DOMINIO (o NEGOCIO)
- **ANÁLISIS:**
  - ESPECIFICAR REQUISITOS
  - CONSTRUIR MODELO DEL ANÁLISIS

Flujos de trabajo

- **DISEÑO:**
  - ENCONTRAR LA FORMA DEL SISTEMA (SOLUCIÓN)
  - CONSTRUIR MODELO DEL DISEÑO
- **IMPLEMENTACIÓN:**
  - CODIFICAR EL DISEÑO (SOLUCIÓN)
  - CONSTRUIR MODELO DE IMPLEMENTACIÓN
- **PRUEBAS:**
  - VERIFICAR LA IMPLEMENTACIÓN
  - CONSTRUIR MODELO DE PRUEBAS

ANEXO

Fases: Iniciación

Establecer la planificación del proyecto

- ¿Qué va a hacer el sistema para cada uno de sus usuarios principales?
  - Un MCU simplificado con los CU más críticos
- ¿Cómo sería la arquitectura para un sistema como ese?
  - Borrador con los subsistemas principales
- ¿Cuál es el plan y cuánto va a costar desarrollar el producto?
  - Identificar los riesgos principales y priorizarlos, planificar elaboración y presupuesto aproximado

ANEXO

Fases: Elaboración

Establecer un plan para el proyecto y una arquitectura correcta

- Especificar en detalle los CU + críticos
- Diseñar la arquitectura
  - Mediante vistas de todos los modelos del SI
  - Vista arquitectónica de MCU, M. Análisis, M. Diseño, M. Implementación (con los componentes que demuestran que la arquitectura es ejecutable) y M. Distribución.
- *Al final de esta fase se debe poder planificar las actividades y estimar los recursos para poder completar el proyecto. ¿Son los CU, arquitectura y planes lo suficientemente estables y los riesgos bajo control suficiente para firmar un contrato para terminar el trabajo de desarrollo?*

ANEXO

Fases: Construcción

Desarrollar el sistema

- Se construye el producto. En esta fase:
  - La arquitectura se completa para construir un sistema bien cimentado
  - La visión evoluciona hasta convertirse en un producto preparado para los usuarios
  - Es donde se gastan la mayoría de los recursos
  - La arquitectura del sistema es estable. Sin embargo, se pueden realizar cambios mínimos a la misma.
  - ¿El producto se ajusta suficientemente a las necesidades de los usuarios de algunos usuarios como para enviarselo ya?

Fases: Transición

Proporcionar el sistema a los usuarios finales

- El producto se encuentra en fase *beta*
  - Un grupo reducido de usuarios experimentados prueba el producto e informa de los defectos y deficiencias y sugieren mejoras.
  - Los desarrolladores corrigen las deficiencias e incorporan algunas de las mejoras propuestas en una versión para un grupo de usuarios mayor.
  - En esta fase se encuentran actividades como la venta, formación de los usuarios, ofrecimiento de ayuda en línea y corrección de defectos descubiertos tras la implantación. Los defectos: (1) los que justifican la aparición de una nueva versión del sistema, (2) los que se pueden dejar para la siguiente versión que se cree.