Facultad de Ciencias de la Computación

Desarrollo de sistemas

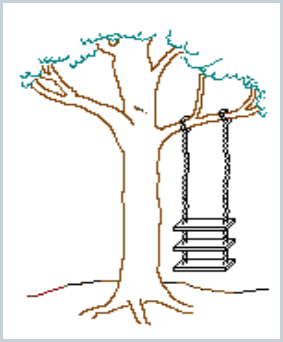
ABRAHAM SÁNCHEZ LÓPEZ
GRUPO MOVIS
FCC-BUAP

Introducción

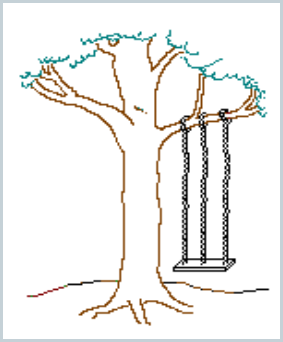
- Actualmente las metodologías de ingeniería de software pueden considerarse como una base necesaria para la ejecución de cualquier proyecto de desarrollo de software que se considere serio, y que necesite sustentarse en algo más que la experiencia y capacidades de sus programadores y equipo.
- Estas metodologías son necesarias para poder realizar un proyecto profesional, tanto para poder desarrollar efectiva y eficientemente el software, como para que sirvan de documentación y se puedan rendir cuentas de los resultados obtenidos.
- Un amplio y buen conocimiento de estas metodologías servirá de base teórica y permitirá comprender completamente todo lo que requiere el análisis, diseño, desarrollo e implementación de un sistema.
- Además es importante, por la demanda que se tiene hoy en día por parte de muchas empresas, el conocimiento de algunas metodologías de desarrollo de software en específico.
- Lo más importante en una primera etapa es poder identificar qué metodología de ingeniería de software se adecúa de la mejor manera a nuestro proyecto, para así lograr el mejor resultado en tiempo y forma.

Desarrollo de software

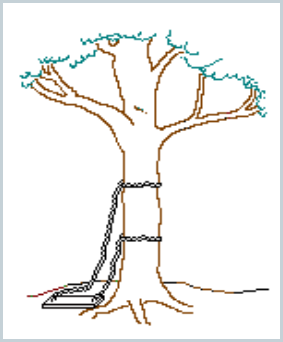
- Y la comunicación?



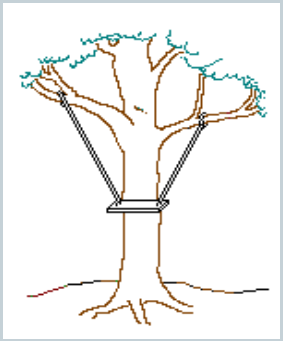
1. Lo que el director desea.



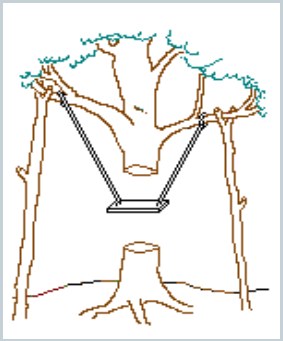
2. Como lo define el director de proyecto.



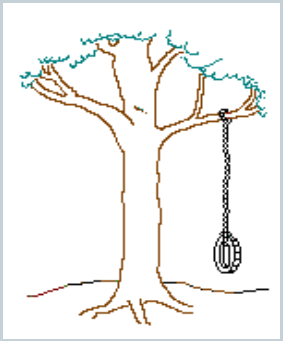
3. Como se diseña el Sistema.



4. Como lo desarrolla el programador.



5. Como se ha realizado la instalación.



6. Lo que el usuario quería.

Metodologías ágiles vs tradicionales, I

- Para asegurar el éxito durante el desarrollo de software, no es suficiente contar con notaciones de modelado y herramientas, hace falta un elemento importante: *la metodología de desarrollo.*
- Esta metodología nos provee una dirección a seguir para correcta aplicación de los demás elementos.
- Generalmente el proceso de desarrollo llevaba asociado un marcado énfasis en el control del proceso mediante una rigurosa definición de roles, actividades y artefactos, incluyendo modelado y documentación detallada.
- Este esquema "tradicional" para abordar el desarrollo de software ha demostrado ser efectivo y necesario en proyectos de gran tamaño (respecto a tiempo y recursos), en donde como sabemos se exige un algo grado de cuidado en el proceso.
- Sin embargo, este enfoque no resulta ser el más adecuado para muchos de los proyectos actuales donde el entorno de sistema es muy cambiante, y además donde se exige reducir drásticamente los tiempos de desarrollo manteniendo al mismo tiempo una alta calidad.

Metodologías ágiles vs tradicionales, II

- Ante las dificultades para utilizar metodologías tradicionales con estas restricciones de tiempo y flexibilidad, muchos equipos de desarrollo se resignan a no adoptar las *buenas prácticas* de la IS, asumiendo el riesgo que esto conlleva.
- En este contexto, las metodologías ágiles emergen como una posible respuesta para llenar ese vacío metodológico.
- Por estar orientada a proyectos pequeños, las metodologías ágiles constituyen una solución a la medida para ese entorno.
- Aportan una elevada simplificación que a pesar de ello, no renuncia a las prácticas esenciales para asegurar la calidad del producto.

Visión general, I

- La ingeniería (en general, incluyendo la IS) consiste en el análisis, diseño, construcción, verificación y gestión de los productos de su campo de aplicación (software).
- Para construir bien, la IS debe definir el proceso de desarrollo de software.
- En este se pueden distinguir tres fases genéricas:

Fase de definición:

- Se centra en el QUE. Se identifican los requerimientos clave del sistema y del software.
- Esto incluye:
  - Qué información debe ser procesada
  - Qué funcionalidad y rendimiento se desea
  - Qué comportamiento se desea
  - Qué interfaces se van a establecer

Visión general, II

- Aunque, los métodos aplicados durante esta fase varían dependiendo del paradigma de IS que se aplique, de alguna manera tendrán lugar 3 tareas principales:
  - Ingeniería de sistemas o de información
  - Planificación del proyecto de software
  - Ingeniería de requerimientos

Fase de desarrollo

- Se centra en el COMO, es decir, en definir:
  - Cómo se tienen que diseñar las estructuras de datos
  - Cómo se tienen que implementar detalles procedurales
  - Cómo se van a desarrollar las interfaces (ventanas, íconos, cajas de diálogo, etc.)
  - Cómo se va a traducir el diseño en un lenguaje de programación
  - Cómo se va a realizar las pruebas

Visión general, III

- Igual que en la fase anterior, los métodos aplicados en esta fase varían dependiendo del paradigma de IS utilizado. Aunque, siempre deben llevarse a cabo 3 tareas específicas técnicas:
  - Diseño del software
  - Generación de código
  - Prueba del software

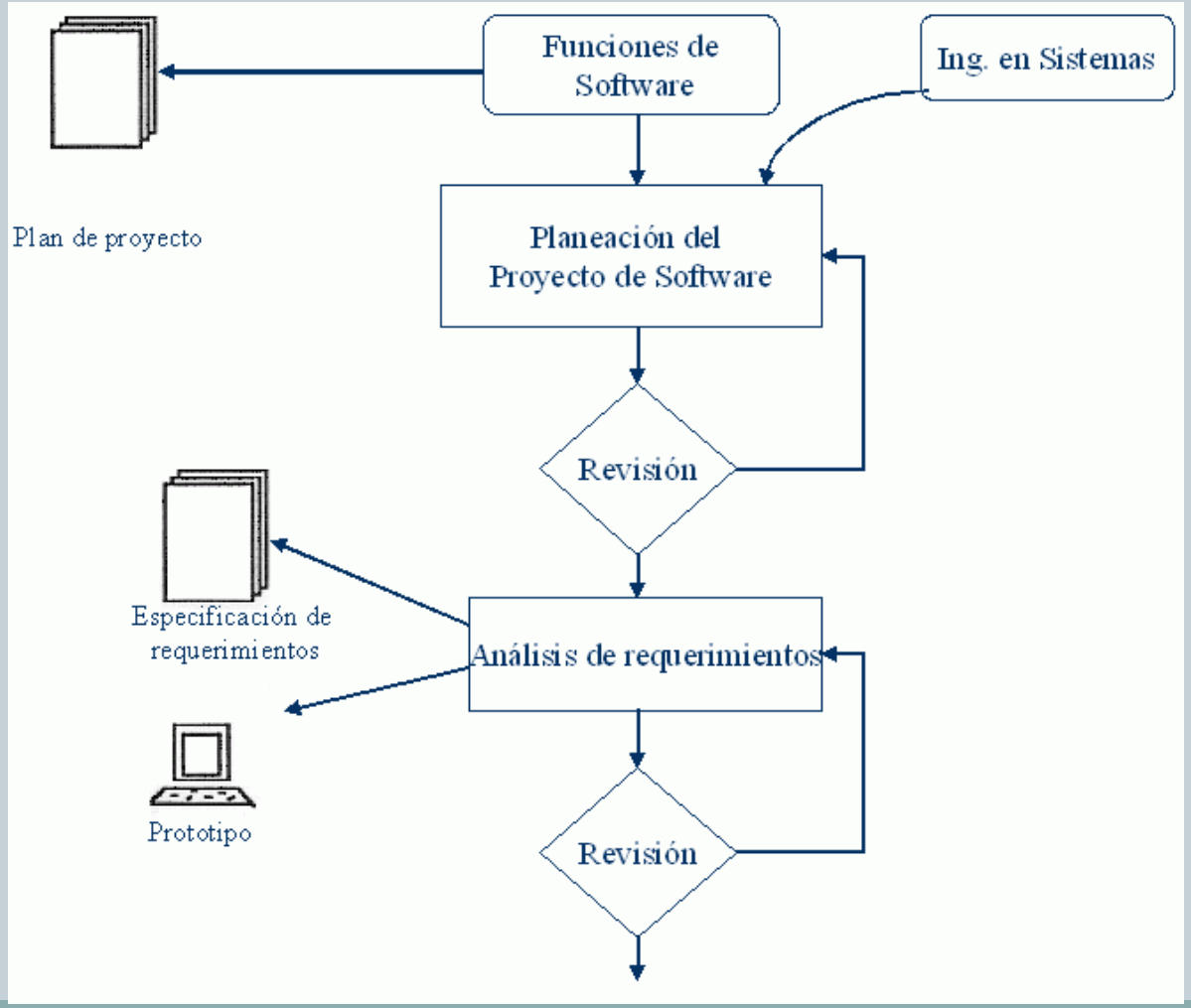
Fase de mantenimiento

- Se centra en el CAMBIO:
  - Asociado a la corrección de errores
  - A adaptaciones requeridas a medida que evoluciona el entorno del software
  - Debidos a las mejoras producidas por los requerimientos cambiantes del cliente. Por ejemplo, el cliente quiere algo más
- Esta fase vuelve a aplicar los pasos de las fases de definición y de desarrollo, pero en el contexto del software ya existente.

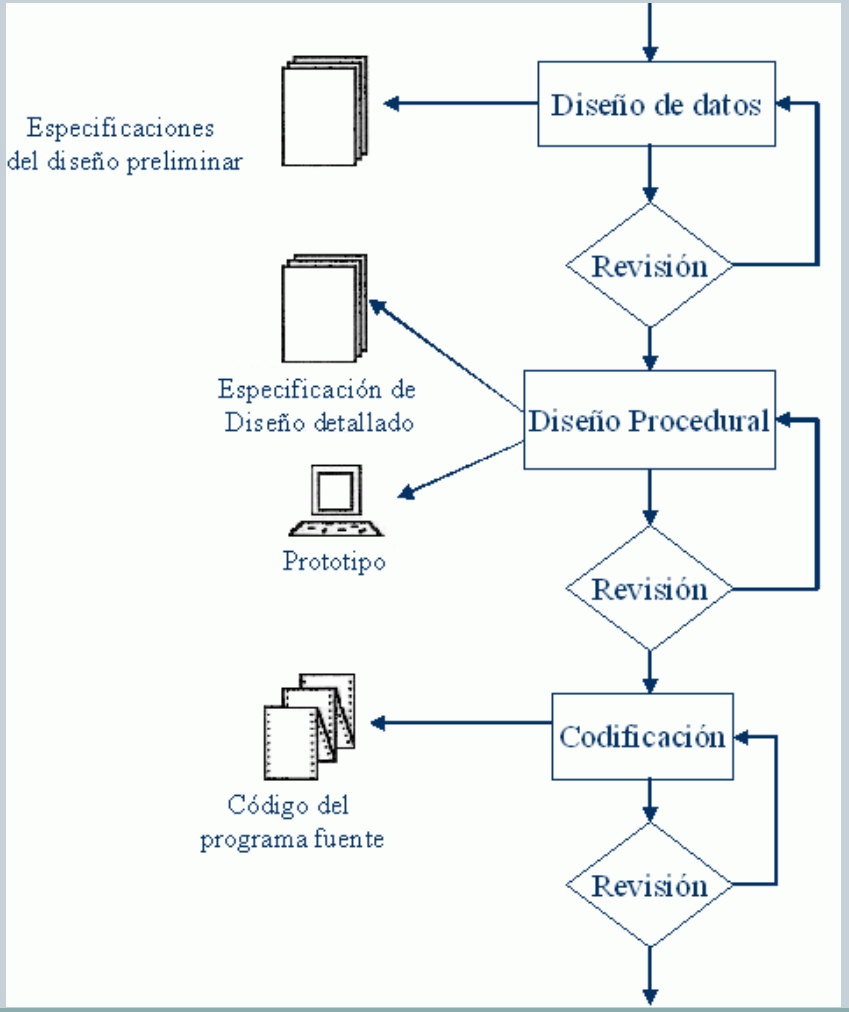
Visión general, IV

- Durante esta fase se encuentran 4 tipos de cambios:
- **Corrección**: de los defectos del software encontrados por el cliente (mantenimiento correctivo)
- **Adaptación**: a los cambios del entorno original para el que se desarrolló el software (mantenimiento adaptativo). Por ejemplo, sistema operativo, reglas de la empresa, etc.
- **Mejora**: por descubrir funciones adicionales que van a producir más beneficios (mantenimiento perfectivo).
- **Prevención**: implica hacer cambios en los programas con el fin de que se puedan corregir, adaptar y mejorar más fácilmente (mantenimiento preventivo).

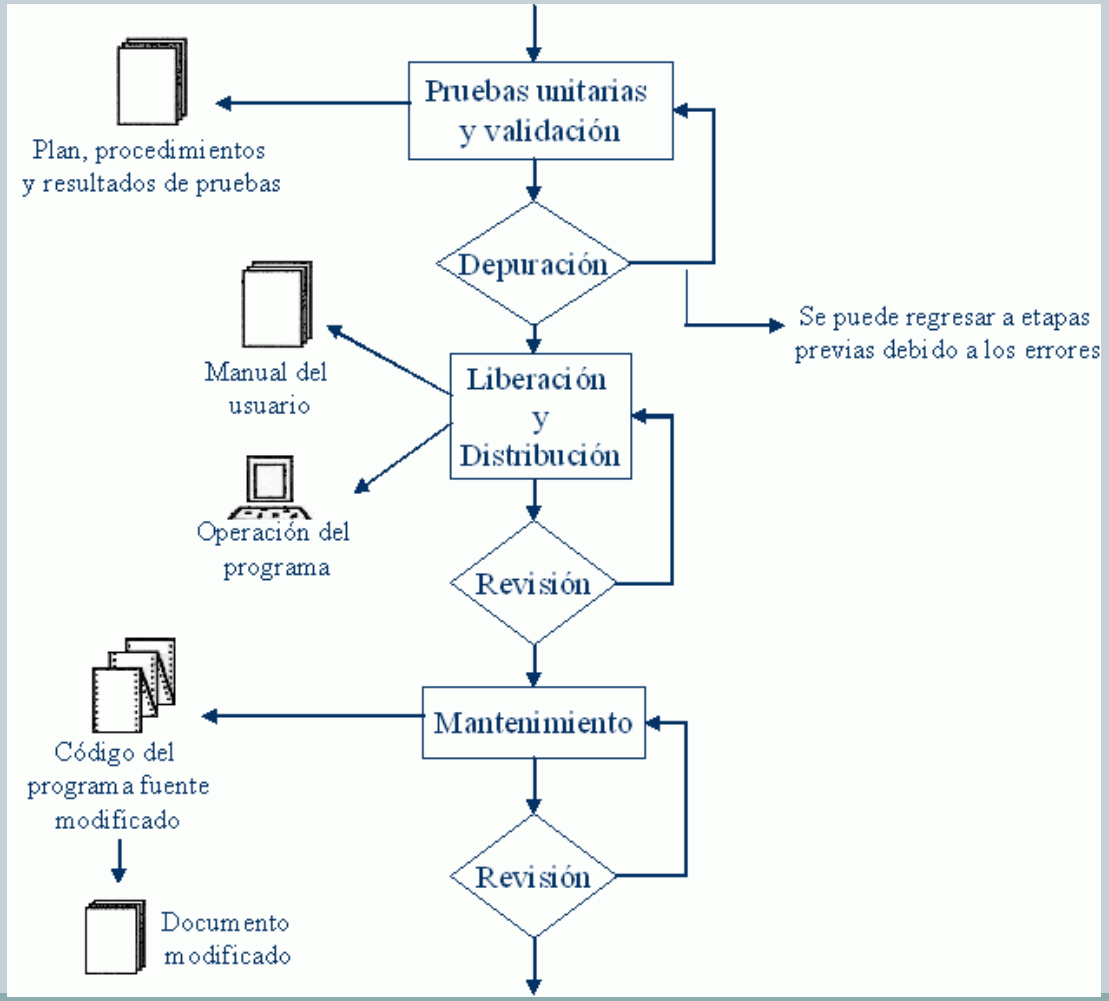
Fase de definición y planeación



Fase de desarrollo



Fase de mantenimiento



Paradigmas de la IS clásicos

Existen varios paradigmas, los más conocidos son:

- El modelo en cascada
- Modelos de proceso incrementales
  - El modelo incremental
  - El modelo DRA
- Modelos de proceso evolutivos
  - Construcción de prototipos
  - El modelo en espiral
  - El modelo de desarrollo concurrente
- Modelos especializados de proceso
  - Desarrollo basado en componentes
  - El modelos de métodos formales
  - Desarrollo de software orientado a aspectos
- El proceso unificado

Paradigmas ágiles, I

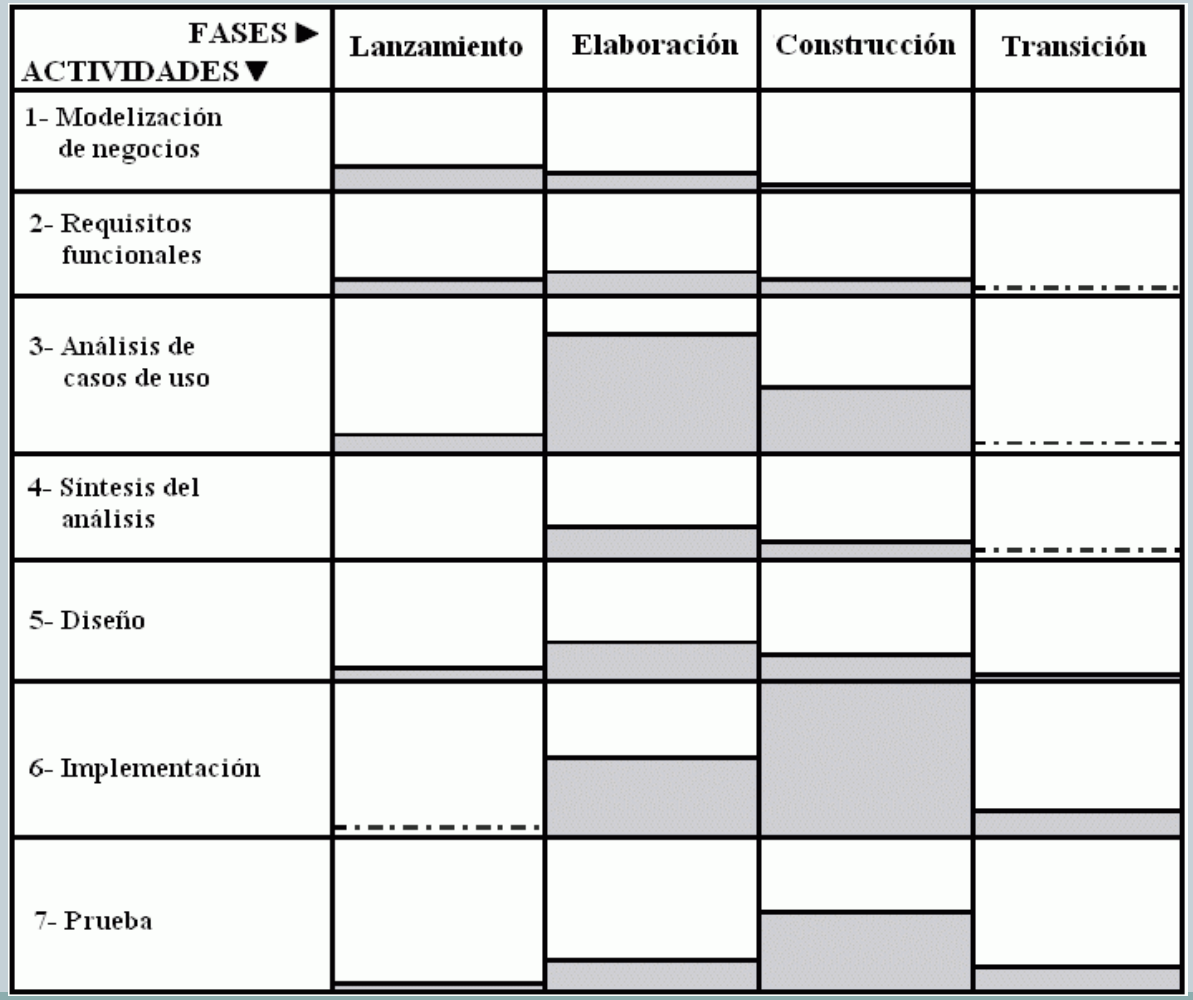
- La siguiente tabla resume algunas de las metodologías ágiles.

Metodología	Acrónimo	Creación	Tipo de modelo	Característica
Adaptive software Development	ASD	Highsmith, 2000	Prácticas + ciclo de vida	Inspirado en sistemas adaptativos complejos.
Agile Modeling	AM	Ambler, 2002	Metodología basada en la práctica	Suministrad modelado ágil a otros métodos.
Cristal Methods	CM	Cockburn, 1998	Familia de metodologías	Metodología ágil con énfasis en modelo de ciclos.
Agile RUP	dX	Booch, Martin, Newkirk, 1998	Framework/Disciplina	XP dando vuelta con artefactos RUP.
Dynamic Solutions Delivery Model	DSDM	Stapleton, 1997	Framework/modelo de ciclo de vida	Creado por 16 expertos en RAD
Evolutionary Project Management	EVO	Gilb, 1976	Framework adaptativo	Primer método ágil existente
eXtreme Programming	XP	Beck, 1999	Disciplina en prácticas de ingeniería	Método ágil radical

Paradigmas ágiles, II

Metodología	Acrónimo	Creación	Tipo de modelo	Característica
Feature-Driven Development	FDD	De Luca & Coad, 1998 Palmer & Felsing, 2002	Metodología	Método ágil de diseño y construcción.
Lean Development	LD	Charette, 2001	Forma de pensar – modelo logístico	Metodología basada en procesos productivos.
Rapid Development	RAD	McConnell, 1996	Revisión de técnicas y modelos	Selección de mejores prácticas, no es un método.
Microsoft Solutions Framework	MSF	Microsoft, 1994	Lineamientos, disciplinas, prácticas	Framework de desarrollo de soluciones.
Scrum	Scrum	Sutherland, 1994 Schwaber, 1995	Proceso – Framework de administración	Complemento de otros métodos, ágiles o no.

Enfoque UP7



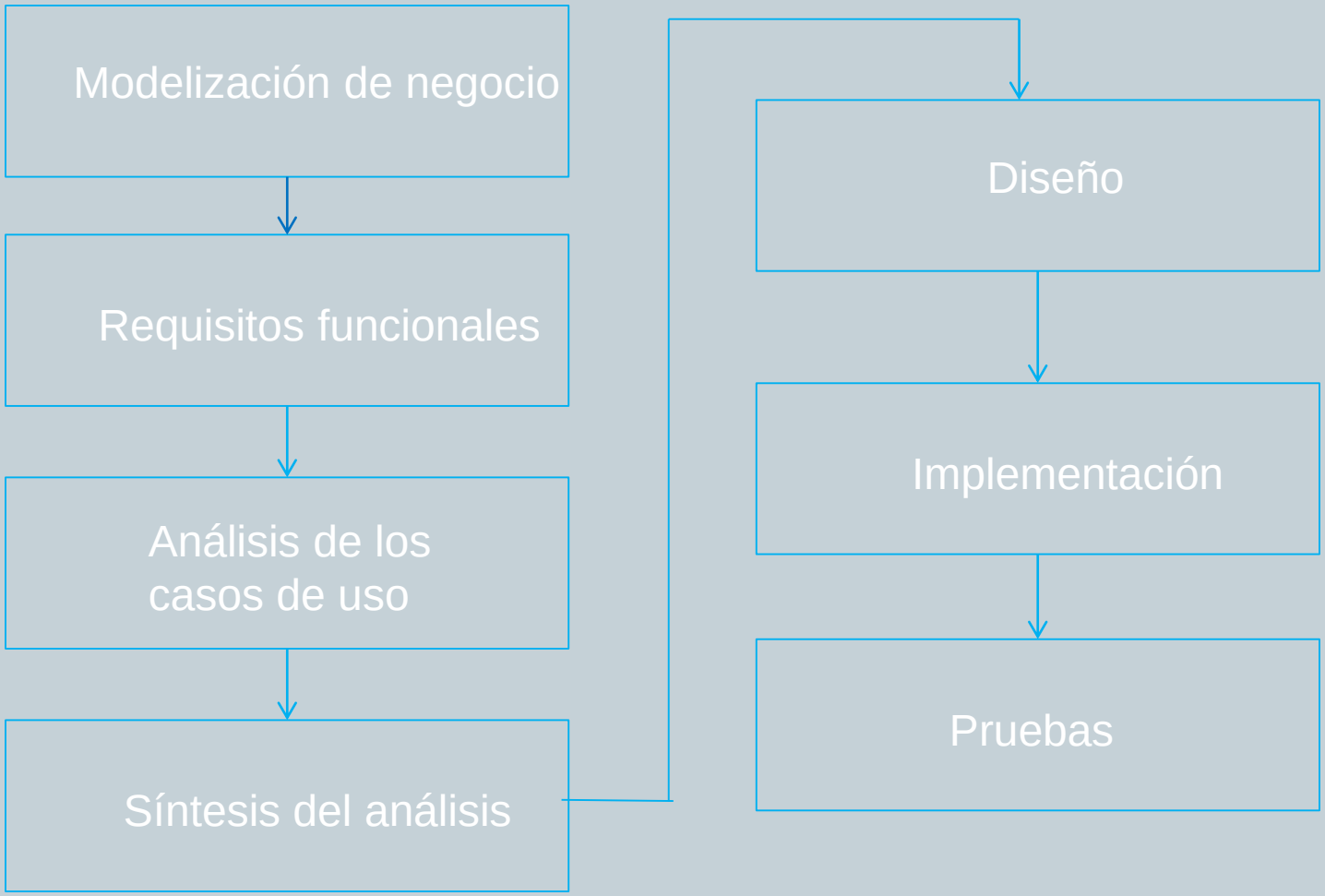
Algunos detalles de UP7, I

- Es importante notar que sobre el esquema, la proporción que representa cada actividad en relación al conjunto de la carga de desarrollo de un proyecto se ha respetado gráficamente.
- Dada nuestra experiencia y de las proporciones habitualmente constatadas en la profesión, hemos adoptado la repartición indicativa siguiente para los volúmenes de esfuerzo consagrado en las actividades del proyecto.
  - Modelización de negocio : 5%
  - Requisitos funcionales : 5%
  - Análisis de los casos de uso : 20%
  - Síntesis del análisis : 5%
  - Diseño : 10%
  - Implementación : 40%
  - Pruebas : 15%
- Es importante comprender para cada proyecto sus especificidades en términos por ejemplo de la complejidad funcional, restricciones técnicas y competencia de recursos afectados.

Algunos detalles de UP7, II

- Las actividades 6 y 7, “Implementación” y “Pruebas”, están presentes en este esquema para cubrir en este nivel la totalidad de las actividades en referencia al UP, pero no se discutirán en este curso.
- Este enfoque es por definición iterativo.
- Es también posible, si fuera necesario, tratar las iteraciones al interior de cada fase.
- Cada iteración debe concluirse con un producto entregable bajo la forma de una maqueta o de un prototipo.
- Las iteraciones sucesivas permiten afinar los resultados de cada fase.

Esquema detallado del enfoque, I



Esquema detallado del enfoque, II

1. **Modelización de negocios**
   1.1 Elaboración del esquema de contexto del dominio de estudio (FG1)
   1.2 Elaboración del diagrama de actividad (FG2)
   1.3 Elaboración del diagrama de clases de negocio (FG3)
2. **Requisitos funcionales**
   2.1 Elaboración del diagrama de casos de uso de sistema (FG4)
   2.2 Elaboración de los diagramas de secuencia del sistema (FG5)
   2.3 Elaboración de esquema de navegación general (FG6)
3. **Análisis de casos de uso**
   3.1 Elaboración del diagrama de casos de uso (FG7)
   3.2 Descripción de los casos de uso (FG8)
   3.3 Elaboración de los diagramas de secuencia (FG9)
   3.4 Elaboración de los diagramas de estado-transición (FG10)
   3.5 Elaboración de las interfaces de usuario (FG11)
   3.6 Elaboración de los diagramas de clase (FG12)

Esquema detallado del enfoque, III

4. **Síntesis del análisis**
   - 4.1 Elaboración del diagrama de clase recapitulativo (FG13)
   - 4.2 Elaboración de la matriz de validación (FG14)
5. **Diseño**
   - 5.1 Realización de la elección de técnicas (FG15)
   - 5.2 Elaboración de los diagramas de secuencia técnicos (FG16)
   - 5.3 Elaboración de los diagramas de clase técnicos (FG17)
   - 5.4 Elaboración del diagrama de paquetes (FG18)
6. **Implementación**
   - Actividad no detallada en este curso
7. **Pruebas**
   - Actividad no detallada en este curso

Ficha guía 1

FICHA GUIA 1 – FG1	
Actividad 1: Modelización de negocio **Sub-Actividad 1.1:** **Elaboración del esquema de contexto del dominio de estudio**	**Proyecto:** **Fecha:**
Objetivo	Posicionar el sistema a estudiar en el seno de los procesos de la empresa.
Punto de partida	Bosquejo funcional del requisito considerado.
Punto de llegada	Sistema a estudiar posicionado en relación a los procesos de la empresa y perímetro funcional definido.
Enfoque de elaboración	
1 Identificar los procesos conexos en el sistema estudiado. 2 Determinar las interacciones entres los procesos conexos y el sistema estudiado. 3 Precisar el perímetro del sistema a estudiar (definición de los subconjuntos funcionales).	
Recomendación: Poner en evidencia el subconjunto a estudiar (subconjunto gris/líneas punteadas) en el esquema de contexto.	

Ficha guía 3

FICHA GUIA 3 – FG3	
Actividad 1: Modelización de negocio **Sub-Actividad 1.1:** **Elaboración del diagrama de clases (DCL) de negocio**	**Proyecto:** **Fecha:**
Objetivo	Definir los conceptos de negocio del dominio bajo la forma de clases.
Punto de partida	Actores identificados y procesos de negocio del sistema estudiado definidos (flujo de control, flujo de datos).
Punto de llegada	Conceptos de negocio identificados en el DCL de negocio y el glosario de negocio.

Enfoque de elaboración

1 Identificar los conceptos del dominio bajo la forma de clases, tomando como base los conceptos definidos en el diagrama de actividad.
2 Precisar los principales atributos útiles en la comprensión de expertos de negocio.
3 Determinar las relaciones entre las clases:
- nombre de la asociación,
- multiplicidad.

4 Describir de manera general los conceptos del dominio con el fin de obtener un glosario de negocio.

Recomendaciones:

1 Limitarse en este nivel a los conceptos estructurantes para el sistema en estudio.
2 Elaborar en síntesis un expediente de modelado de negocio.

Ficha guía 7

FICHA GUIA 7 – FG7		
Actividad 3: Análisis de casos de uso **Sub-Actividad 3.1:** **Elaboración del diagrama de casos de uso**		**Proyecto:** **Fecha:**
Objetivo	Definir la totalidad de los casos de uso (negocios y computacionales). Los casos de uso computacionales son funciones complementarias que no se han identificado durante la actividad anterior (ejemplo, un modulo de administración).	
Punto de partida	Requisitos de negocios, interacciones actores de negocios/sistema, IHC definidos.	
Punto de llegada	Todos los casos de uso definidos en el DCU.	

Enfoque de elaboración

1 Identificar todos los actores del sistema tomando como base aquellos definidos en el DCU del sistema de la actividad anterior. Los actores computacionales aparecen en este nivel del análisis (ejemplo Administrador de una aplicación).

2 Identificar todos los casos de uso tomando como base aquellos definidos en el DCU del sistema de la actividad anterior. Estos son a menudo descompuestos en un nivel más detallado. Los casos de uso computacionales aparecen en este nivel del análisis.

3 representar las interacciones entre los actores y los casos de uso.

4 Definir las dependencias entre los casos de uso.

Recomendación:

Tratar de limitar el número de casos de uso. No pasar aproximadamente 10 por cada nivel de descripción.

Ficha guía 13

FICHA GUIA 13 – FG13	
Actividad 4: Síntesis del análisis **Sub-Actividad 4.1:** **Elaboración del diagrama de clase recapitulativo**	**Proyecto:** **Fecha:**
Objetivo	Reagrupar el conjunto de las clases en un solo diagrama para tener una visión global del sistema estudiado.
Punto de partida	Todos los diagramas de clases de los casos de uso estudiados.
Punto de llegada	Reagrupamiento de las clases en un solo DCL.

Enfoque de elaboración

1 Recuperar el conjunto de los DCL de todos los CU.

2 Agrupar, para las mismas clases, los atributos y operaciones de cada clase definidas en los DCL por CU. Cada clase debe describirse de manera exhaustiva,

3 Agrupar todas las asociaciones entre las clases definidas en los DCL por CU.

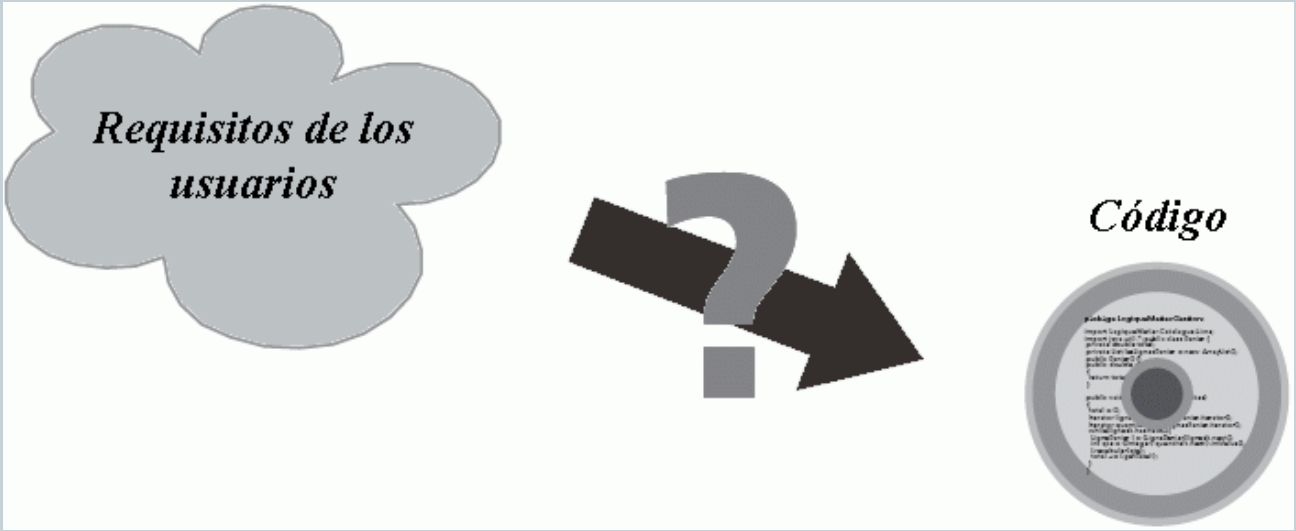
4 Determinar las relaciones entre diferentes DCL de los CU. Colocar las nuevas relaciones necesarias en el DCL recapitulativo.

Recomendación:

Asegurarse que el DCL recapitulativo sea legible para los actores de negocio al menos en el nivel del nombre de clases y de las principales relaciones. Establecer, el caso que falla, un DCL recapitulativo en dos niveles de lectura.

Propuesta de desarrollo web

- El problema fundamental que discutiremos a lo largo del curso es el siguiente:
- Como pasar de los requisitos de los usuarios al código de la aplicación?
- En otras palabras: tengo una buena idea de lo que mi aplicación debe hacer, las funcionalidades esperadas por los futuros usuarios.
- ¿Cómo obtener lo más eficazmente posible un código computacional operativo, completo, probado, y que responda perfectamente al requisitos? .



Esquema completo del proceso