

Tema 3: Ingeniería del Software

Tema 3: Ingeniería del Software
Entornos de Desarrollo

Alejandro Cardo Grau



Entornos de Desarrollo

T3: Ingeniería del Software
ÍNDICE

CRONOLOGÍA 1º/2º TRIMESTRE

		Capítulos del Libro	
			Entornos de Desarrollo
BLOQUE I: INTRODUCCIÓN A LOS ENTORNOS DE DESARROLLO	T1: El Programa Informático	CAPÍTULO 1. DESARROLLO DE SOFTWARE	Pendiente para el 2º trimestre: 1.6 ARQUITECTURA DE SOFTWARE 1.6.1 Patrones de desarrollo 1.6.2 Desarrollo en tres capas
	T2: Entornos de Desarrollo	CAPÍTULO 2. INSTALACIÓN Y USO DE ENTORNOS DE DESARROLLO	
BLOQUE II: ANÁLISIS Y DISEÑO DE PROYECTOS	T3: Ingeniería del Software	CAPÍTULO 1. DESARROLLO DE SOFTWARE	
	T4: Modelado del Proceso	CAPÍTULOS 5 y 6. DISEÑO ORIENTADO A OBJETOS. DIAGRAMAS DE COMPORTAMIENTO	5.1 INTRODUCCIÓN A UML 6.1 Tipos y campo de aplicación 6.2 Diagramas de actividad
	T5: Modelado de Análisis de Requisitos	CAPÍTULO 6. DISEÑO ORIENTADO A OBJETOS. DIAGRAMAS DE COMPORTAMIENTO	6.3 Diagramas de casos de uso

Alejandro Cardo Grau

Entornos de Desarrollo

INDICE**1. Fases Generales del Desarrollo en Proyectos Software**

- 1. Etapas y fases**
- 2. Otros ciclos de vida del desarrollo software**
- 3. Roles de trabajo**

2. Calidad del Software

- 1. Normas y certificaciones.**
- 2. Medidas de la calidad del software.**

3. UML

- 1. Introducción**
- 2. Herramientas CASE para el modelado UML. Estándares.**
- 3. Metodologías Relacionadas con UML**

4. Tipos de Diagramas. Campo de aplicación.**5. Referencias****1. Fases Generales del Desarrollo en Proyectos Software****• Método:**

- Es la **forma de realizar a cabo una tarea** de manera sistemática siguiendo unos pasos previamente establecidos y definidos en unas especificaciones (CÓMO).

• Artefacto:

- Es algo tangible creado con un propósito práctico y siguiendo un **método de desarrollo**.
 - Para producirlo siguen un enfoque sistemático y disciplinado, sujeto a métodos que dictan como deben hacerse.
 - Es un elemento que el proyecto produce o usa mientras se trabaja en busca del producto final.
 - Muchas de las actividades realizadas en el desarrollo de proyectos software están orientadas a:
 - Obtener un producto concreto y ofrecer información en el proceso:
 - Ejemplos: documentos técnicos, diagramas, código fuente, ejecutables, ...

T3: Ingeniería del Software

1. Fases Generales del Desarrollo en Proyectos Software

- **Framework:**

- Es un conjunto de **componentes** que cooperan entre sí y permiten reutilizarse para desarrollar software.
- Parten de un conjunto diseñado específicamente para ser extendido y no como software final.
 - Normalmente se componen de interfaces de comunicación (API).
- Requieren una inversión en coste inicial de aprendizaje (curva de aprendizaje) del equipo de desarrollo, aunque a largo plazo.

Módulo o parte funcional de un sistema
Ejemplo: Plugins IDE

- **Ejemplos:**

- Un framework para construcción de interfaces gráficas de usuario (GUI).
 - El usuario se centra en el desarrollo de la interfaz abstrayéndose de cómo se implementan sus distintos componentes (botones, ventanas, etc.).
- Un framework para el acceso a bases de datos (Hibernate)
 - El usuario se centra en el desarrollo de la aplicación sin necesidad de saber cómo está implementado el modelo físico que compone la base de datos.
- Un framework para la construcción de aplicaciones web (Spring/Struts).

T3: Ingeniería del Software

1. Fases Generales del Desarrollo en Proyectos Software

- **Framework:**

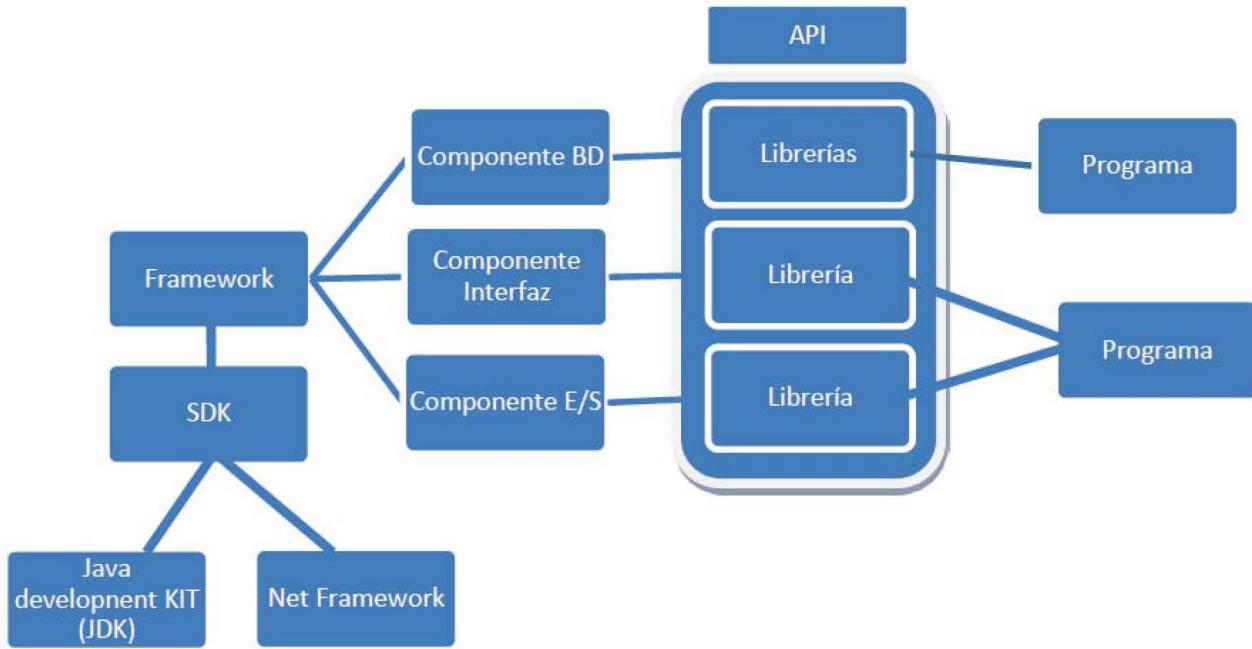
- **Ventajas**

- Suponen una importante reducción de costes a largo plazo.
 - Es necesario valorar la simplicidad y funcionalidad que aporta para el desarrollo.
- Facilitan el propio trabajo del desarrollo del proyecto.
- Desarrollo rápido y uniforme del software que se desarrolla.
- Reutilización de los componentes para otras aplicaciones.

- **Inconvenientes**

- Dependencia del código:
 - Si cambiamos de *framework*, habrá que valorar el coste y riesgo.
- La instalación e implementación consume recursos del sistema.

1. Fases Generales del Desarrollo en Proyectos Software



Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

- **Frameworks CSS:**
 - Los **frameworks CSS** incluyen utilidades para que el diseñador no tenga que trabajar en ningún aspecto genérico del diseño web.
- **Propiedades comunes:**
 - **Neutralizar los estilos** por defecto que se aplican en navegadores web:
 - Hojas de estilo CSS reseteadas (`reset.css`)
 - Hojas personalizadas para la creación de elementos visuales CSS3 atractivos.
 - Manejar correctamente el texto y sus contenidos de forma que:
 - Se vea exactamente igual en múltiples navegadores web
 - Sean **adaptables a cualquier medio** y/o dispositivo móvil/equipo para mejorar su accesibilidad y permitir su acceso global.
 - Crear cualquier estructura compleja o **layout autogenerado** de forma sencilla:
 - Funcionar correctamente en cualquier versión de cualquier navegador.

Alejandro Cardo Grau

Entornos de Desarrollo

The screenshot shows a blog post from 'TemplateMonsterBlog.es' titled 'Mejores 10 HTML/CSS Frameworks en 2014'. The post discusses how the combination of HTML and CSS helps create functional web sites with various features. It highlights the increasing intelligence of frameworks like Bootstrap and Montage.js. Below the text is a grid of screenshots for several frameworks: CLANK, Bootstrap, Semantic UI, and Montage.js. A large banner at the bottom reads 'Top HTML/CSS Frameworks'.

Frameworks CSS: <http://www.templatemonsterblog.es/2014/03/03/mejores-10-htmlcss-frameworks-en-2014>

Alejandro Cardo Grau

Entornos de Desarrollo

- **PREGUNTA 1:**

- El desarrollo de aplicaciones requiere una gran cantidad de esfuerzo por parte de los desarrolladores cuando concierne el entendimiento de nuevas tecnologías, arquitecturas y lenguajes. ¿Qué ventajas aportaría introducir un nuevo framework para el desarrollo de aplicaciones de la empresa?

- **PREGUNTA 2 (TEST):**

- Indicar cuál de las siguientes afirmaciones es cierta:
 - 1) El resultado de obtener un informe estadístico acerca del porcentaje de errores y avisos del código fuente es un artefacto.
 - 2) Para medir cuantitativamente el esfuerzo generado por los programadores se usa el artefacto de registro de horas de trabajo.
 - 3) La larga curva de aprendizaje que supone a los desarrolladores introducir un nuevo framework puede implicar un aumento de costes inicial, por lo que es necesario valorar además la simplicidad junto a las funcionalidades que ofrecen.
 - 4) Todas las anteriores son correctas.

REPASO

- **PREGUNTA 3 (TEST):**

- JQuery es un lenguaje de programación *front-end* que procesa y traduce las instrucciones en tiempo de ejecución y, por tanto, es:
 - 1) Compilado
 - 2) Interpretado
 - 3) Virtual
 - 4) Ninguna de las otras respuestas son correctas..

- **PREGUNTA 4 (TEST):**

- El _____ Quicktime extiende la funcionalidad básica de ver vídeos, en formato MOV, en un navegador web (no sólo para mostrar documentos web):
 - 1) Artefacto
 - 2) Framework
 - 3) Plugin
 - 4) Todas las anteriores son correctas.

Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

- **El ciclo de vida de un proyecto de desarrollo software es el periodo de tiempo que:**
 - Comienza cuando se toma la decisión de desarrollar un producto software y finaliza cuando se entrega el software al cliente.
- **Deben de dejarse claro en las etapas iniciales del proyecto:**
 - Las necesidades a satisfacer por el sistema.
 - Los artefactos entregables del proyecto.
 - El presupuesto y plazo de ejecución.
 - Penalizaciones por retrasos.
 - Restricciones técnicas.
- El proceso más habitual en el desarrollo de un software es el **modelo de ciclo de vida en cascada** (no es el único), el cual consta de 7 etapas.

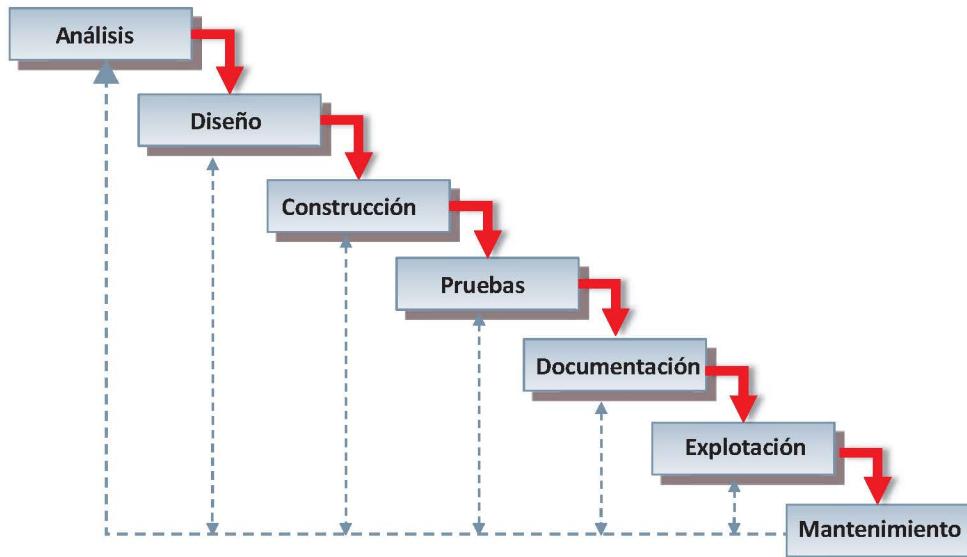


Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software**• Modelo de ciclo de vida en cascada**

- Para empezar una etapa es necesario finalizar la etapa anterior.
- Es difícil obtener todos los requisitos al comienzo.
- Se tarda mucho en disponer del software.
- Es el más fácil de planificar, es el ciclo ideal.



Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software**• Modelo de ciclo de vida en cascada****• Ventajas:**

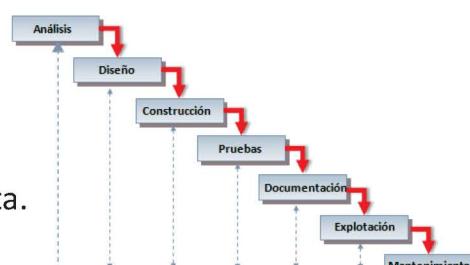
- Fácil de planificar y seguir.
- La calidad del producto resultante es alta.

• Inconvenientes:

- La necesidad de tener todos los requisitos definidos desde el principio:
 - Pueden surgir necesidades imprevistas.
- Es difícil volver atrás si se detectan fallos en una etapa:
 - Será necesario volver a la etapa anterior, realizar ajustes y continuar.
- El producto no está disponible para su uso hasta que no está completamente finalizado.
 - Se tarda mucho en disponer del software.

• Se recomienda cuando:

- El proyecto es similar a alguno realizado con éxito en la empresa.
- Los requisitos son estables, poco variables y bien comprendidos.
- Los clientes no necesitan versiones intermedias.



Alejandro Cardo Grau

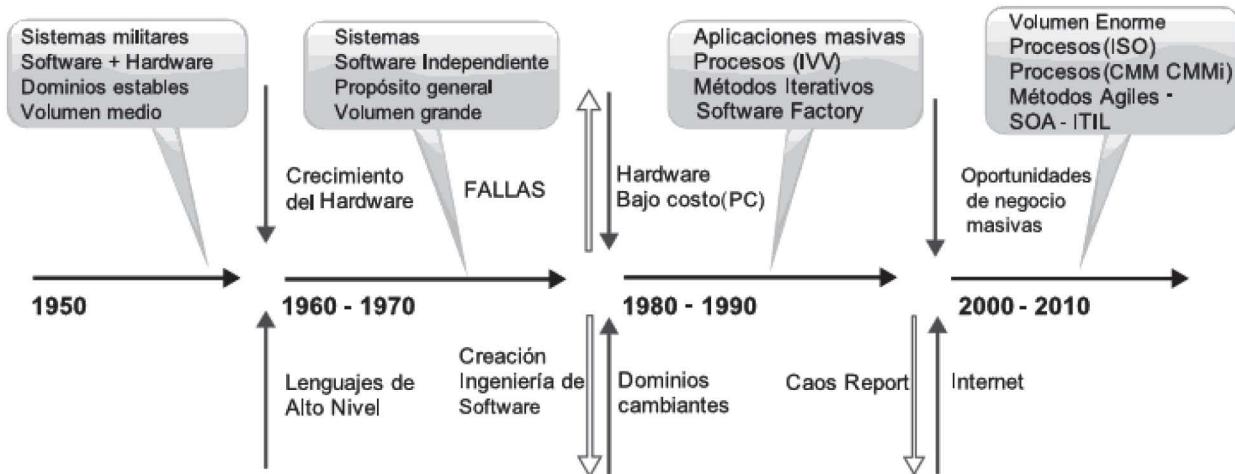
Entornos de Desarrollo

T3: Ingeniería del Software

1. Fases Generales del Desarrollo en Proyectos Software

- Modelos de ciclos de vida evolutivos**

- Conseguir obtener todos los requisitos al comienzo del proyecto es prácticamente imposible.
- Las necesidades de clientes y usuarios evolucionan durante el desarrollo y surgen nuevos requisitos.
- Cuanto mayor es un proyecto, menor es su probabilidad de éxito (**Informe CHAOS**).



Alejandro Cardo Grau

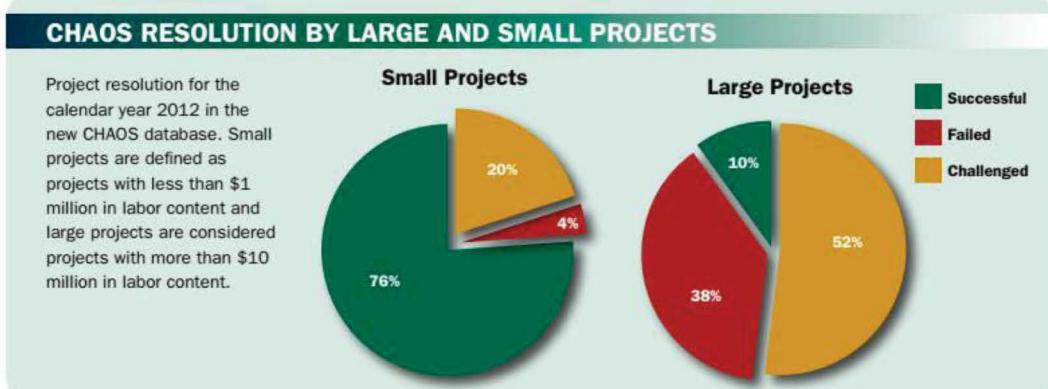
Entornos de Desarrollo

T3: Ingeniería del Software

1. Fases Generales del Desarrollo en Proyectos Software

- Modelos de ciclos de vida evolutivos**

	2004	2006	2008	2010	2012	RESOLUTION
Successful	29%	35%	32%	37%	39%	Project resolution results from CHAOS research for years 2004 to 2012.
Failed	18%	19%	24%	21%	18%	
Challenged	53%	46%	44%	42%	43%	



CHAOS Manifesto (2013): <http://www.versionone.com/assets/img/files/CHAOSManifesto2013.pdf>

Alejandro Cardo Grau

Entornos de Desarrollo

- **Modelos de ciclos de vida evolutivos**

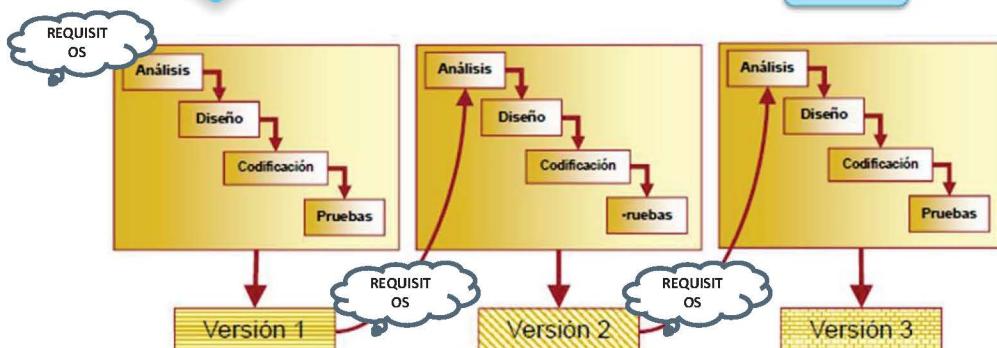
- Los ciclos de vida evolutivos afrontan estos problemas mediante ciclos: **requisitos → desarrollo → evaluación**.
 - Permiten desarrollar versiones cada vez más completas hasta llegar a un producto final deseado.
 - Las necesidades del usuario no están completas y se requieren de una vuelta para planificar y diseñar después de cada entregable.
 - **Principales modelos de ciclo de vida evolutivos:**
 - Iterativo Incremental
 - Espiral

- **Modelos de ciclos de vida evolutivos**

Evolutivo Espiral



Evolutivo Incremental



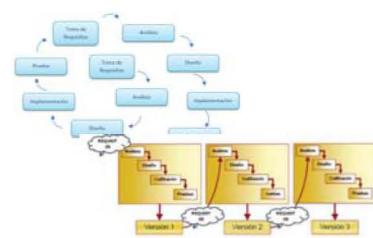
T3: Ingeniería del Software

1. Fases Generales del Desarrollo en Proyectos Software

- Modelos de ciclos de vida evolutivos**

- Ventajas:**

- No se necesita conocer todos los requisitos al comienzo.
 - Permite la entrega temprana al cliente de partes operativas.
 - Las entregas facilitan la realimentación de los próximos entregables.



- Inconvenientes:**

- Es difícil estimar el esfuerzo y el coste final necesario.
 - El coste del proyecto aumenta a medida que va evolucionando.
 - Se tiene el riesgo de no acabar nunca.
 - Planificación extra necesaria en cada iteración o vuelta.

- Se recomienda cuando:**

- Proyectos de gran tamaño y que necesitan constantes cambios.
 - Los requisitos no están definidos y es posible cambios futuros.
 - El desarrollo de proyectos se orientan a la construcción del prototipado móvil.

Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.1 Etapas y fases

- Análisis:** Especifica las necesidades o requisitos del software que hay que desarrollar. Interactúa en alta medida con el cliente.



- Técnicas iniciales para captura de requisitos:**

- Entrevistas:**

- Técnica tradicional que consiste en hablar con el cliente.



- Desarrollo conjunto de aplicaciones (JAD: Joint Application Development):**

- Tipo de entrevista estructurada aplicable a grupo de personas.
 - Cada persona juega un rol concreto.



- Braimstorming**

- Tipo de reuniones en grupo cuyo objetivo es generar ideas desde diferentes puntos de vista.

Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.1 Etapas y fases

- **Artefactos principales:**

- Documento de Especificación de Requisitos del Software (ERS)
- Documentos de Análisis de Requisitos del Sistema

1. Introducción

- 1.1. Propósito
- 1.2. Ámbito del Sistema
- 1.3. Definiciones, Acrónimos y Abreviaturas
- 1.4. Referencias
- 1.5. Visión General del Documento

2. Descripción General

- 2.1. Perspectiva del Producto
- 2.2. Funciones del Producto
- 2.3. Características de los Usuarios
- 2.4. Restricciones
- 2.5. Suposiciones y Dependencias
- 2.6. Requisitos Futuros

3. Requisitos Específicos

- 3.1. Interfaces Externas
- 3.2. Funciones
- 3.3. Requisitos de Rendimiento
- 3.4. Restricciones de Diseño
- 3.5. Atributos del Sistema
- 3.6. Otros Requisitos

4. Apéndices

Índice del Documento de Especificación de Requisitos del Software (ERS) (IEEE.830)

Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.1 Etapas y fases

- **Técnicas para la representación y definición del análisis de requisitos:**

- **Casos de uso**
- **Prototipos**
- Diagrama Entidad-Relación (ERD)
- Diagramas Flujo de Datos (DFD)
- Diccionario de Datos (puede ser ampliado en la fase posterior):
 - Descripción detallada de los **datos utilizados o producidos** en el sistema.
 - Usado también en fases posteriores por los S.G.B.D. para determinar que datos son usados en el modelo relacional diseñado.
- Dependiendo de la metodología aplicada se aplicarán **artefactos diversos**.

Nombre de la Clase	Asistencia			
Objetivo				
Tipo Clase	Class	X	Abstracta	Parametrizada
ATRIBUTOS				
Nombre	Tipo	Longitud	Valor Inicial	Visibilidad
Registro	Int	10	0	Privado
Tipo_asistencia	char	1	0	Privado
fecha	date	dd/mm/yyyy	00/00/0000	Privado
hora	time	hh:mm:ss	00:00:00	Privado

Ejemplo de modelo de diccionario de datos

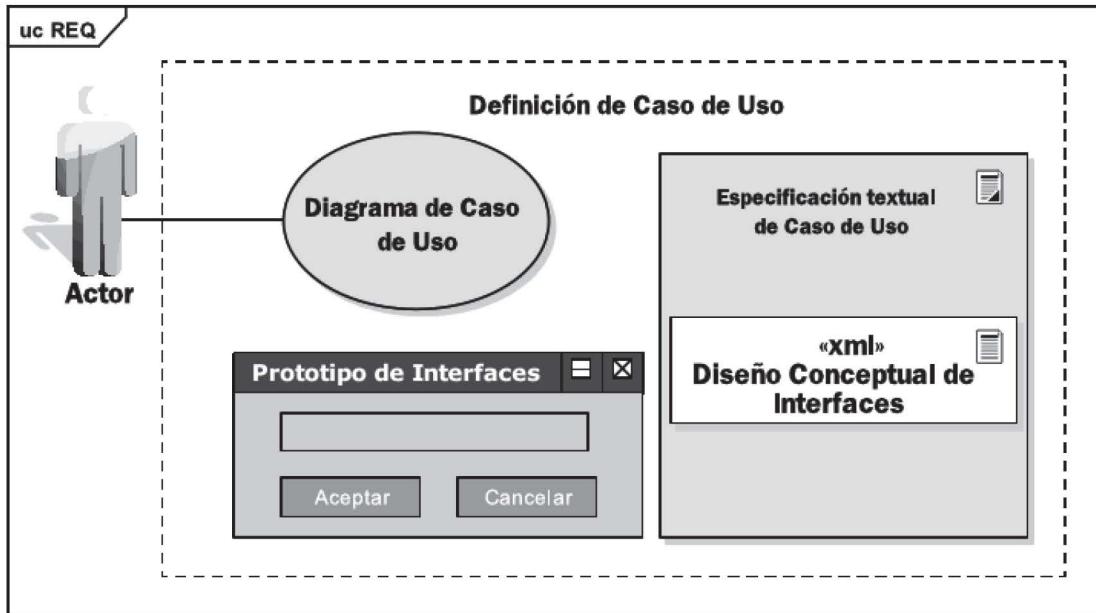
Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.1 Etapas y fases

- Análisis de casos de uso para prototipos de interfaces



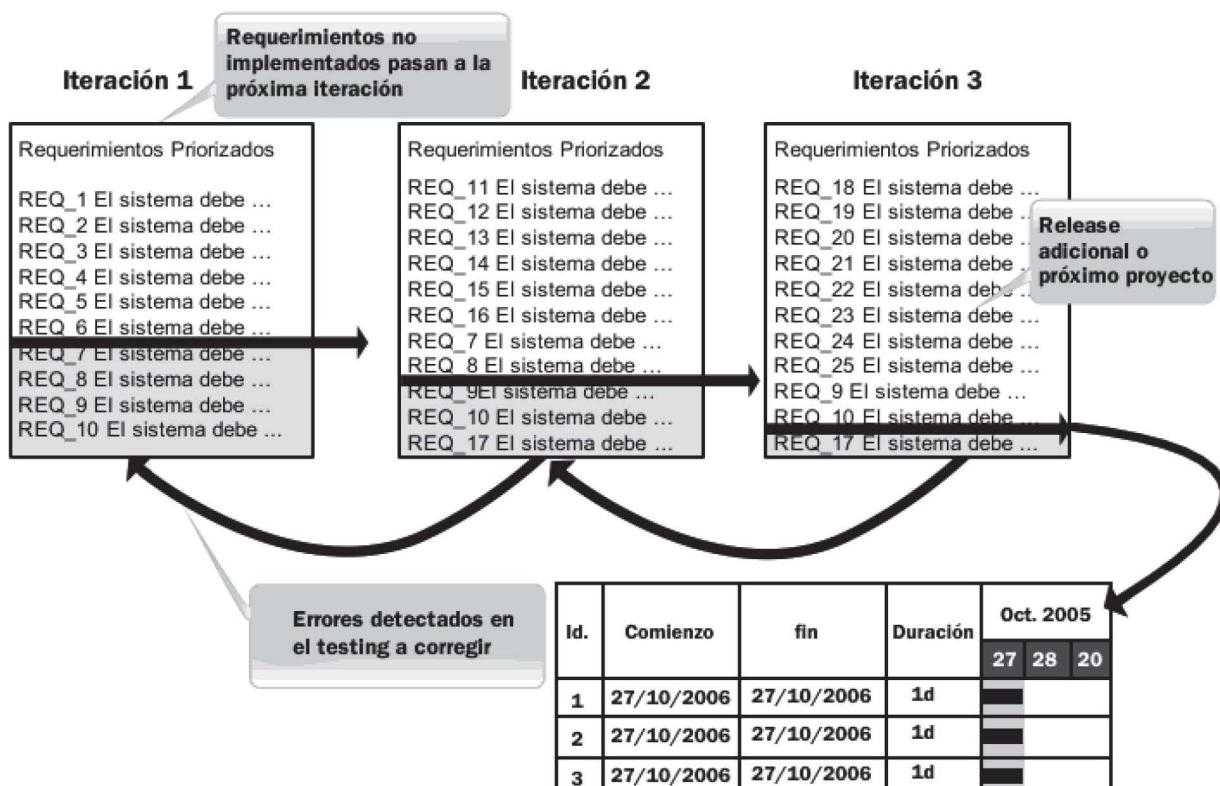
Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.1 Etapas y fases

- Listado de requisitos del análisis usando ciclos evolutivos



Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.1 Etapas y fases

- **Diseño:**

- En esta etapa se pretende determinar el funcionamiento de una forma global y general.
- Se diseña la relación entre los recursos y diagramas analizados en la fase de análisis:
- Se diseñan:
 - Estructuras de datos
 - Prototipos funcionales
 - El modelo relacional de la base de datos
 - El diagrama de clases del sistema (uso de patrones del diseño)
 - La interfaz gráfica del usuario aplicando guías de estilo
 - La arquitectura del software
 - Procedimientos principales a seguir para la posterior codificación.
- Ejemplos: seleccionar el lenguaje de programación, decidir cual es el Sistema Gestor de Bases de Datos (S.G.B.D.) para el diseño de la B.D., la arquitectura de componentes a seguir, etc.

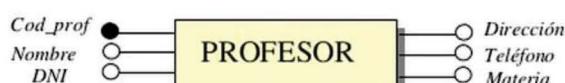
Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.1 Etapas y fases

- **Transformación de ERD a Modelo Relacional**



```

CREATE TABLE Profesor (
    Cod_Profesor Códigos,
    Nombre Nombres,
    DNI DNIS, NOT NULL
    Dirección Lugares,
    Teléfono Nos_Teléfono,
    Materia Materias,
    PRIMARY KEY (Cod_Profesor),
    UNIQUE (DNI));
  
```

Cod_prof	Nombre	DNI	Dirección	Teléfono	Materia
00001	Juan	12223433	Rios Rosas, 23	670123123	Ing. Software
00002	Coral	54656754	Alarcos, 8	567983456	Bases de datos
00003	Belén	53567523	Getafe, 4	6º9267854	Orientación objetos
00004	Goyo	97856757	Pez, 102	679345763	Sistemas operativos
.
.
03568	Roberto	34534522	Fundación, 10	639456239	Redes

Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.1 Etapas y fases



Los prototipos se orientan hacia la GUI, pueden ser: desechables, rápidos, básicos, acordes a los requisitos y a veces sin criterios de calidad. Van evolucionando hacia la GUI final.

Bocetos analizados y prototipos funcionales del diseño para su evaluación con el cliente en ambas plataformas móviles y de acuerdo a guías de estilo

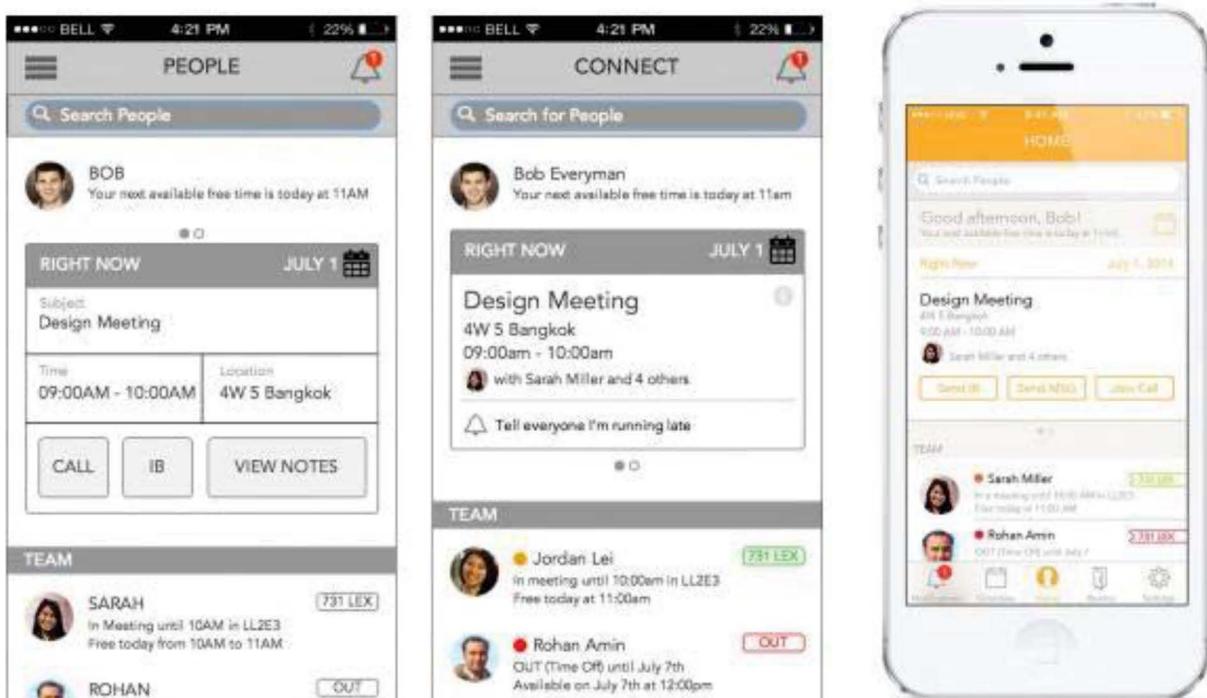
Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.1 Etapas y fases

- **Prototipos funcionales diseñados**



Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.1 Etapas y fases

- **Codificación:**

- El programador recibe las especificaciones del diseño y las transforma en un conjunto de instrucciones escritas (código fuente) usando para la implementación del proyecto:
 - Los lenguajes de programación.
 - Los entornos de desarrollo necesarios y sus compiladores asociados.
 - Los *plugins*, librerías, frameworks y módulos externos requeridos.
 - Las plataformas o sistemas operativos dirigidos en el desarrollo.



Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.1 Etapas y fases

- **Pruebas:** Buscan comprobar la calidad y estabilidad del código realizado:
 - Confirmando que la codificación ha sido exitosa y que no hay errores.
 - Comprobando que el software hace lo que debe hacer.
- **Documentación:** Debe mostrar una información completa y de calidad para ejecutar y manejar la aplicación final.
 - Documentación dirigida al usuario (**manual de usuario**).
 - Documentación dirigida al equipo técnico:
 - Documentación técnica de instalación y configuración para su implantación.
 - Documentación técnica de desarrollo para su codificación e integración.
 - Generador automatizado de documentación: Herramienta que genera la documentación de un software en un formato específico (generalmente en HTML) a partir de los comentarios embebidos del código:
 - 1) Analiza el código fuente.
 - 2) Extrae los comentarios.
 - 3) Les da el formato deseado.
 - 4) Genera una salida que pone a disposición del usuario.
- **Explotación:**
 - Preparar la distribución, despliegue, implantación y puesta en marcha.
 - Comprobar la compatibilidad entre sistemas para su integración final.

Alejandro Cardo Grau

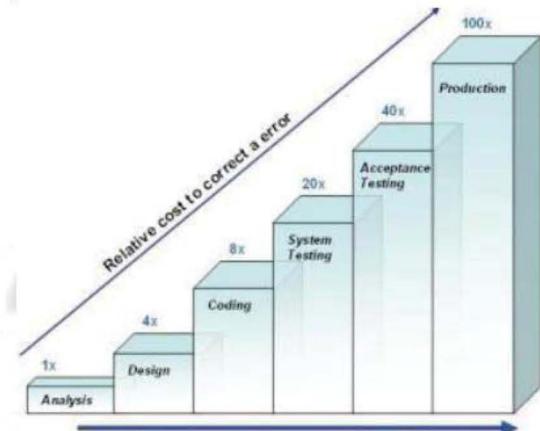
Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

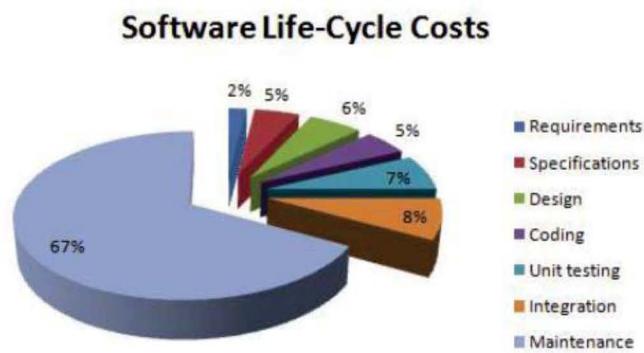
1.1 Etapas y fases

- **Mantenimiento:**

- Gestión de control de incidencias para solucionar errores producidos.
- Adaptar a cambios en el entorno tecnológico (*hardware*, sistema operativo, comunicaciones, bases de datos, cambios de versión, etc.)



El error es usualmente 100 veces más caro de corregir en la fase de mantenimiento que en la fase de análisis (Barry Boehm - 1981)



Coste del Mantenimiento (Fuente:
<http://thibautvs.com/blog/?tag=maintenance>)

Alejandro Cardo Grau

Entornos de Desarrollo

T3: Ingeniería del Software

REPASO

- **PREGUNTA 1:**

- El desarrollo de aplicaciones requiere una gran cantidad de esfuerzo por parte de los desarrolladores cuando se definen los requisitos en la etapa de análisis. ¿Qué ciclo de vida adoptarías para obtener un producto satisfactorio si los requisitos no son muy cambiantes, el equipo es inexperto y subcontratado y no se tiene la participación continua del cliente?

- **PREGUNTA 2 (TEST):**

- Indicar cuál de las siguientes afirmaciones es cierta:

- 1) En los ciclos de vida evolutivos se necesitan conocer todos los requisitos al comienzo.
- 2) Es muy común en el ciclo de vida en cascada generar únicamente artefactos basados en documentos entregables.
- 3) Los ciclos de vida evolutivos permiten ofrecer un modelo ejecutable que implementa parte de la funcionalidad del software.
- 4) Todas las anteriores son correctas.

Alejandro Cardo Grau

Entornos de Desarrollo

REPASO

- **PREGUNTA 3 (TEST):**

- Cuando los requisitos son estables es conveniente aplicar en el desarrollo del proyecto software un ciclo de vida:
 - 1) Cascada
 - 2) Evolutivo
 - 3) Evolutivo incremental
 - 4) Ninguna de las otras respuestas son correctas.

- **PREGUNTA 4 (TEST):**

- ¿Qué ciclo de vida es apropiado cuando se necesita una versión inicial del software a desarrollar?:
 - 1) Cascada
 - 2) Evolutivo
 - 3) Hay dos respuestas que son correctas.
 - 4) Ninguna de las otras respuestas son correctas.

REPASO

- **PREGUNTA 5 (TEST):**

- Indicar cuál de las siguientes afirmaciones es cierta:
 - 1) En la fase del diseño se especifica que hay que hacer.
 - 2) En la fase del diseño se especifica cómo hacerlo.
 - 3) En la fase del diseño se realiza el proceso de programación.
 - 4) Ninguna de las otras respuestas son correctas.

- **PREGUNTA 6 (TEST):**

- Dado un proyecto corto, bien definido y teniendo en cuenta que el equipo de desarrollo tiene experiencia en proyectos similares. El ciclo de vida más adecuado sería:

- 1) Cascada
- 2) Evolutivo
- 3) Hay dos respuestas que son correctas.
- 4) Ninguna de las otras respuestas son correctas.

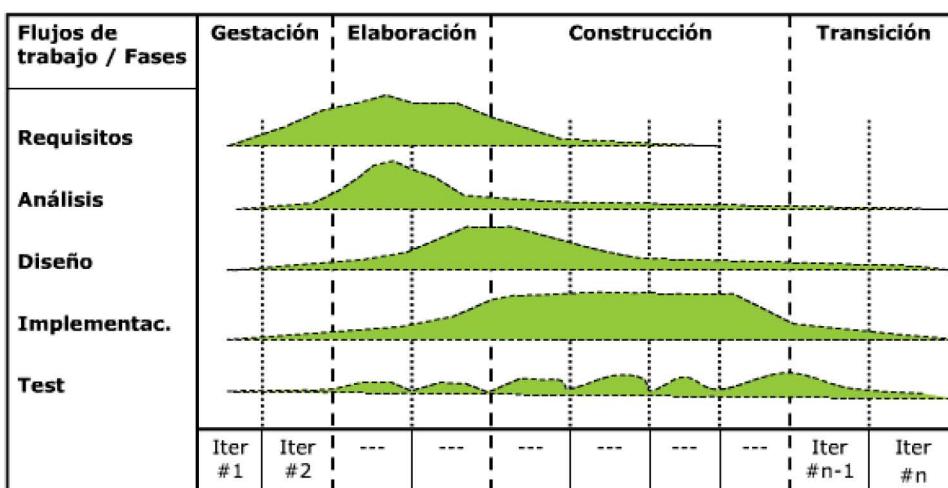
REPASO

- **PREGUNTA 7 (TEST):**
 - ¿Qué técnica emplearías para obtener y validar los requisitos?:
 - 1) Prototipo
 - 2) Caso de uso
 - 3) Hay dos respuestas que son correctas.
 - 4) Ninguna de las otras respuestas son correctas.
 - **PREGUNTA 8 (TEST):**
 - En el software a medida ¿Quién decide las necesidades?:
 - 1) El analista
 - 2) El cliente
 - 3) El propietario
 - 4) El programador

1. Fases Generales del Desarrollo en Proyectos Software

1.2 Otros ciclos de vida del desarrollo software

- **Otras metodologías de desarrollo software:**
 - **Proceso Unificado:**
 - Adopta el ciclo de vida evolutivo: iterativo incremental.
 - En una **misma fase** de iteración se realizan **varios flujos de trabajo**.
 - Se obtiene una versión evaluable del software en cada iteración.
 - Basado en el lenguaje especificativo y descriptivo de diagramas UML.



1. Fases Generales del Desarrollo en Proyectos Software

1.2 Otros ciclos de vida del desarrollo software

- **Otras metodologías de desarrollo software:**

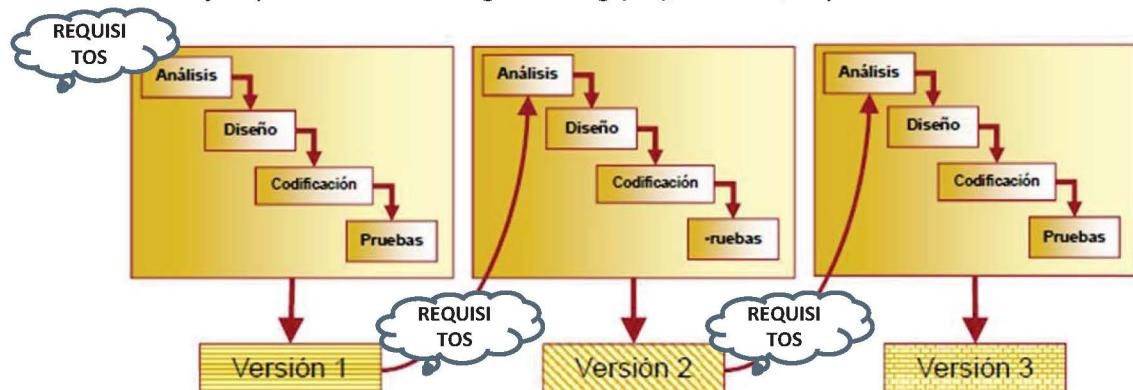
- **CMMI**

- **Métrica 3:**

- Usado como modelo de desarrollo en las aplicaciones de las administraciones públicas, junto a CMMI.
- Basado en las normativas de calidad **ISO/IEC 12207 (Information Technology - Software Life Cycle Processes)** e **ISO/IEC 15504**.

- **Metodologías ágiles:**

- Los ciclos se repiten hasta obtener un producto software satisfactorio.
- En cada iteración se incorporan nuevas peticiones de clientes (**requisitos**)
- Ejemplos: Extreme Programming (XP), SCRUM, Crystal Methods.



Alejandro Cardo Grau

Entornos de Desarrollo

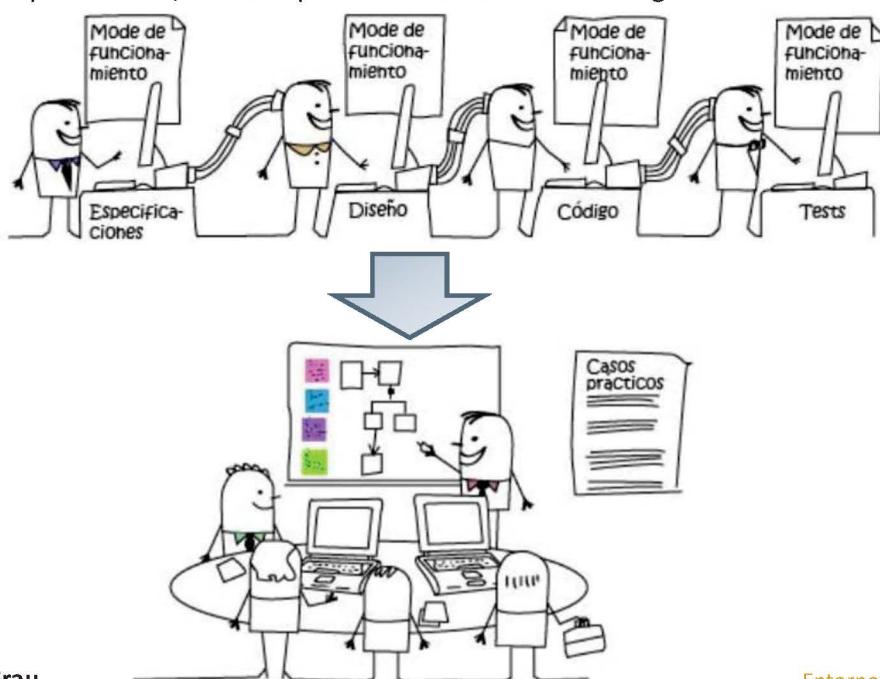
1. Fases Generales del Desarrollo en Proyectos Software

1.2 Otros ciclos de vida del desarrollo software

- **Metodologías ágiles:**

- Es un **conjunto de principios**, cuyos objetivos principales son:

- Potenciar el papel de las relaciones humanas y la participación del cliente.
- Premiar la negociación y colaboración directa entre cliente-desarrollador, por encima, del cualquier contrato de carácter legal.



Alejandro Cardo Grau

Entornos de Desarrollo

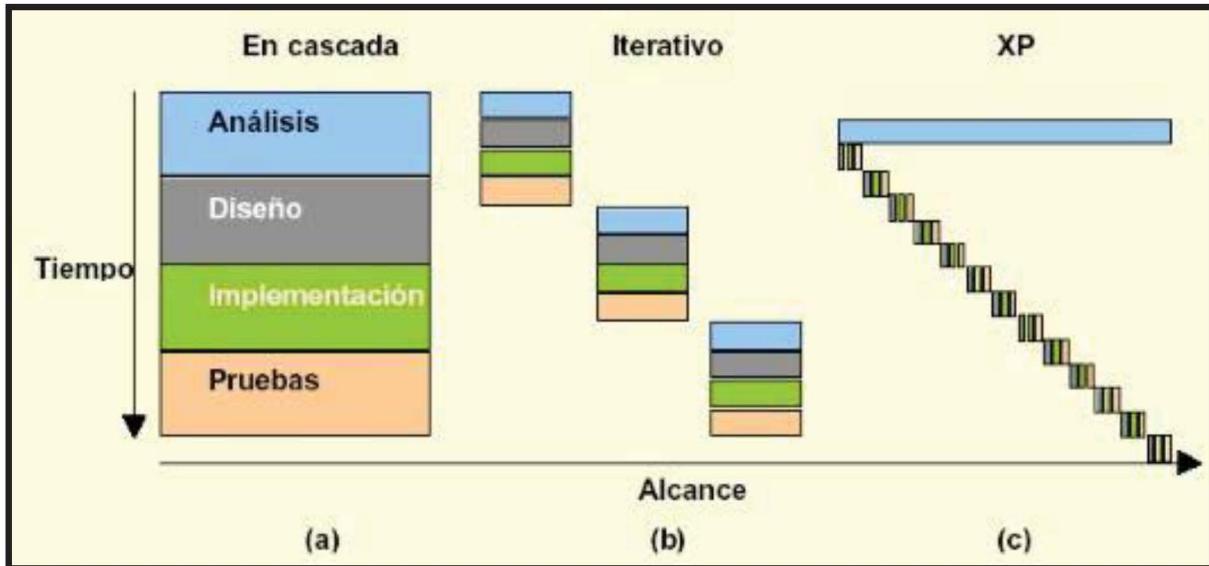
1. Fases Generales del Desarrollo en Proyectos Software

1.2 Otros ciclos de vida del desarrollo software

- **Metodologías ágiles:**

- **Su aparición surge de:**

- La necesidad de responder rápida y eficazmente a los cambios producidos en los requisitos de un proyecto.
- Ajustar los márgenes de tiempo de entrega ante los cambios producidos ofreciendo un software directamente funcional.



Alejandro Cardo Grau

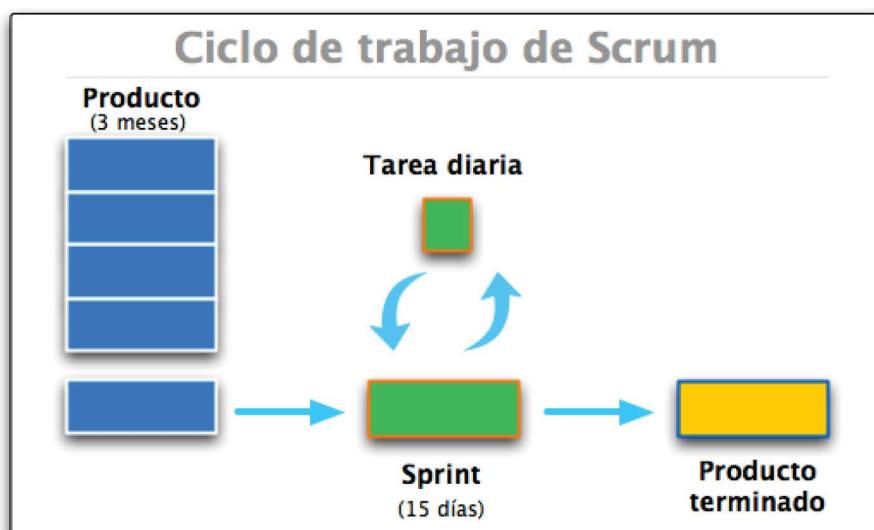
Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.2 Otros ciclos de vida del desarrollo software

- **Metodologías ágiles:**

- Es más deseable realizar **fases y entregas cortas** que permitan producir entregables que funcionen:
 - A pesar de que las entregas no están preparadas para la explotación, permitirán al cliente validar el trabajo que se está haciendo.
 - La mejor métrica para medir la calidad del software producido es la propia satisfacción del cliente.



Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.2 Otros ciclos de vida del desarrollo software

- **Manifiesto Ágil: Principios de las metodologías ágiles**

- Simplicidad.
- Un SW funcionando es la principal medida del progreso.
- Colaboración cercana y diaria entre clientes y desarrolladores.
- Satisfacer al cliente con entregas rápidas de SW funcional.
- El SW se entrega de forma frecuente (semanas VS meses).
- Los equipos con líder se autoorganizan.
- Un proyecto se construye con confianza y personas motivadas.
- Atención continua a la excelencia técnica y el buen diseño.
- Adaptación regular a las circunstancias cambiantes.
- Cambios tardíos en los requisitos pueden ser incluso bienvenidos.

<http://agilemanifesto.org/iso/es>



Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.2 Otros ciclos de vida del desarrollo software

Metodologías Tradicionales	Metodologías Ágiles
Basadas en normas que provienen de estándares seguidos por el entorno de desarrollo.	Basadas en heurísticas provenientes de la práctica de desarrollo.
Ofrecen ciertas resistencias a los cambios.	Están especialmente preparadas para los cambios durante el proyecto.
Impuesta extemamente.	Impuesta internamente (por el equipo).
Procesos mucho más controlados, poseen numerosas políticas y/o normas.	Procesos no controlados, con pocos principios.
Existe un contrato prefijado.	No existe un contrato tradicional o al menos es bastante flexible.
El cliente interactúa con el equipo mediante reuniones.	El cliente es parte del equipo de desarrollo.
Grupos grandes y posiblemente distribuidos.	Grupos pequeños (< 10) y trabajando en el mismo sitio.
Más roles y artefactos.	Pocos roles y artefactos.
La arquitectura del software es esencial y se expresa mediante modelos.	Menos énfasis en la arquitectura del software.

Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.2 Otros ciclos de vida del desarrollo software

- **Otras metodologías de desarrollo software:**

- **SCRUM:** Es un tipo de marco de trabajo de **metodología ágil** basada:
 - En **iteraciones de corta duración** de (máximo 30 días) (**sprints**) que producen código entregable para favorecer la comunicación entre clientes y usuarios.



Alejandro Cardo Grau

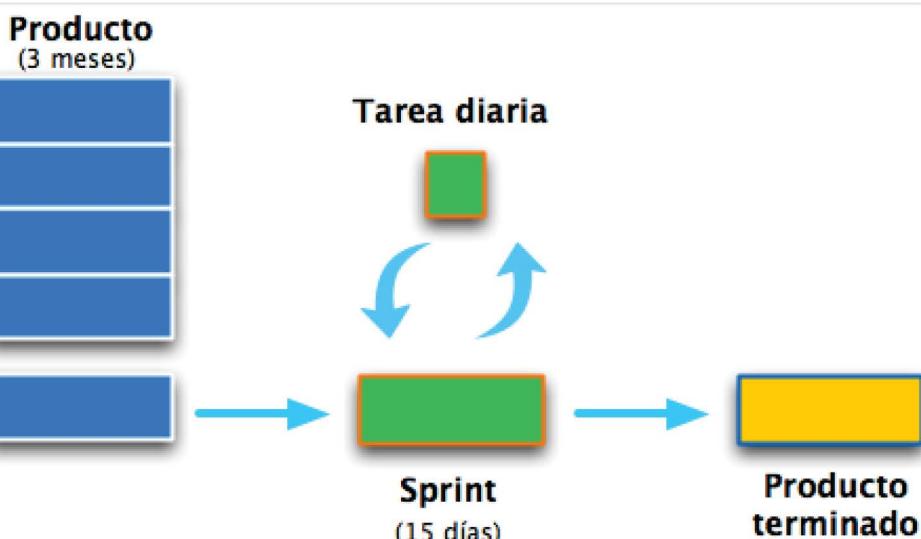
Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.2 Otros ciclos de vida del desarrollo software

Ciclo de trabajo
SCRUM

Ciclo de trabajo de Scrum



Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.2 Otros ciclos de vida del desarrollo software

Ciclo de trabajo SCRUM

- 1) **Toma de requisitos al cliente.** Para cada requisito principal se crea un bloque de trabajo, llamado historia
- 2) El cliente ordena los **bloques de trabajo** en una pila de producto según su prioridad de entrega.
- 3) El equipo de trabajo toma un **grupo de historias**, con el que trabajan durante una iteración o *sprint*.
- 4) Una vez finalizado un *sprint* entregan al cliente el resultado del trabajo. Se vuelve al punto 2º hasta terminar la pila de producto.

Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

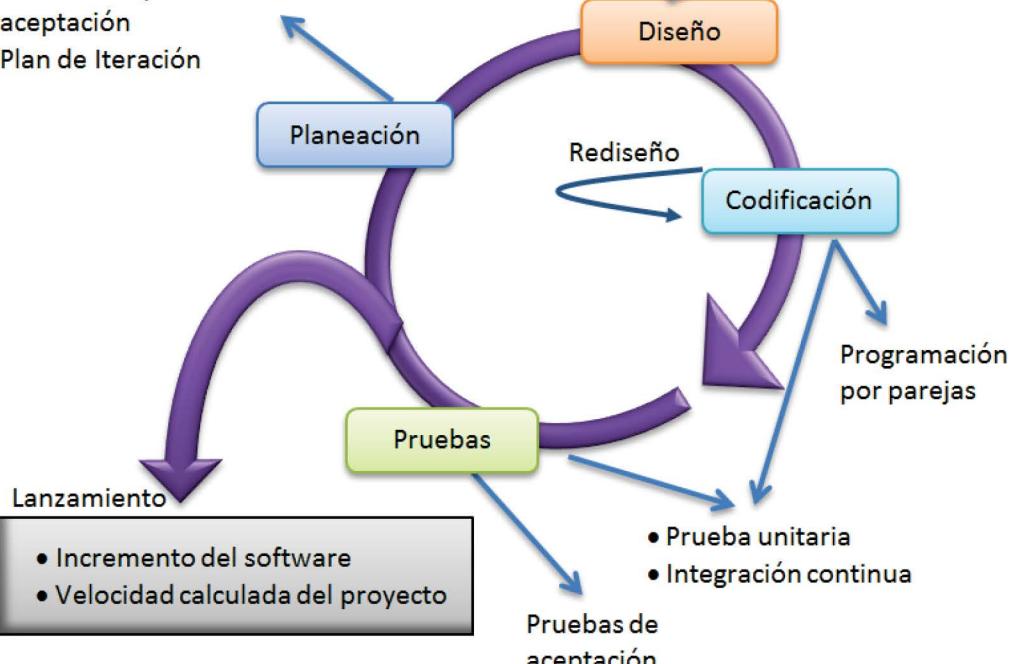
1.2 Otros ciclos de vida del desarrollo software

Extreme Programming (XP)

- **Metodologías ágiles:**

- Historias del usuario
- Valores
- Criterios de pruebas de aceptación
- Plan de Iteración

- Diseño simple
- Tarjetas CRC
- Soluciones en punto
- Prototipos



Alejandro Cardo Grau

Entornos de Desarrollo

1. Fases Generales del Desarrollo en Proyectos Software

1.3 Roles de trabajo

Rol y Equipo	Descripción	Fase en la que participa
 Analista de sistemas	Responsable de interactuar con clientes y usuarios para obtener sus necesidades y desarrollar requisitos	ANÁLISIS
 Diseñador software	Realiza, en función del análisis de un software, el diseño de la solución que hay que desarrollar.	DISEÑO
 Analista programador	Aporta una visión general más detallada diseñando una solución más amigable para la codificación y participando activamente en ella.	DISEÑO CODIFICACIÓN
 Programador	Escribe el código fuente en función del estudio de analistas y diseñadores.	CODIFICACIÓN
 Arquitecto software	Conoce e investiga las tecnologías revisando que todo el procedimiento se lleva a cabo de la mejor forma y los apropiados recursos.	ANÁLISIS DISEÑO DOCUMENTACIÓN EXPLOTACIÓN

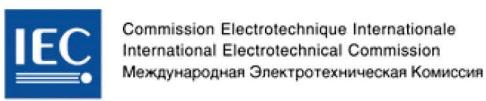
Alejandro Cardo Grau

Entornos de Desarrollo

2. Calidad del Software

2.1 Normas y certificaciones

- En Informática tienen estatus legal para definir estándares:
 - ISO Asociación Internacional para Estándares
 - IEC Comisión Electrotécnica Internacional
 - ANSI Instituto Nacional Americano para Estándares
 - IEEE Instituto Ingenieros Eléctricos-Electrónicos Americano
 - CEN Comité Europeo para la Estandarización
 - W3C Consorcio para la World Wide Web



Alejandro Cardo Grau

Entornos de Desarrollo

2. Calidad del Software

2.1 Normas y certificaciones



International Organization for Standardization

- La **calidad del software [IEEE.Std.610-1990]**:
 - Es el grado con el que un sistema, componente o proceso cumple:
 - Los requisitos especificados.
 - Las necesidades o expectativas del cliente o usuario.
- La organización internacional de estándares, ISO, ha producido una serie de estándares para la gestión y aseguramiento de la calidad conocidos colectivamente como **ISO 9000**.
 - Las normas de la serie **ISO 9000** han sido adoptadas sin modificación como normas europeas (**serie EN 29000**) y como normas españolas (**serie UNE 66-900**).
- **ISO 9000:3**
 - **Modelo de calidad para el desarrollo, aplicación y mantenimiento del software.**

Alejandro Cardo Grau

Entornos de Desarrollo

2. Calidad del Software

2.1 Normas y certificaciones



International Organization for Standardization



Sistema de Gestión de la Calidad para la Mejora Continua

Alejandro Cardo Grau

Entornos de Desarrollo

- **EJERCICIO 1 (FORO):**

- Determina, al menos, un artefacto generado en cada una de las etapas que se compone los ciclos de vida vistos en clase detallando algún ejemplo, imagen o enlace del artefacto generado en un proyecto software que encuentres de referencia y describiendo, al menos, **6 ejemplos de artefactos:**

- Nombre de artefacto
- Descripción y uso
- Etapa o fase del ciclo
- Roles de trabajo que participan
- Imagen, enlace web o ejemplo gráfico
- Página web del proyecto de referencia (puede ser común)

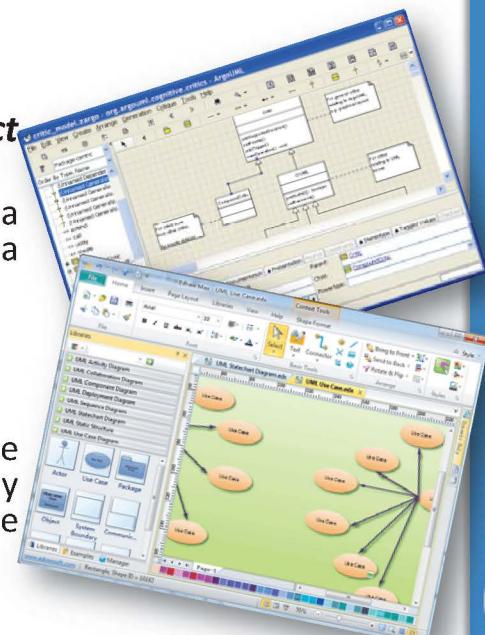
- **EJERCICIO 2 (FORO):**

- En el PDF aportado se analiza la problemática que ha surgido por la implantación y mantenimiento del sistema de gestión de Diraya. Éste es uno de los grandes ejemplos del coste de mantenimiento de los sistemas en función de los errores y la adaptación a nuevas versiones que ha supuesto a los contribuyentes un gasto excesivo. ¿Conoces algún proyecto donde se haya invertido mucho dinero y los costes reales asumidos no hayan sido provechosos en función del objetivo esperado?
- http://www.regulacioninformatica.org/wiki/images/b/b5/Ponencia_-_Antonio_Llergo.pdf

3. UML

3.1 Introducción

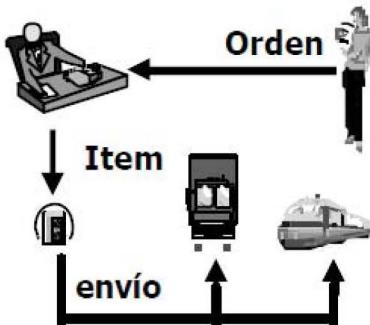
- **UML (Unified Modeling Language)** es un estándar para el modelado conceptual que define 14 tipos de diagramas.
- **Objetivo:** Método formal para modelar sistemas software en desarrollo a partir de la descripción y generación de diagramas.
 - No define documentos textuales
- **Gestionado por la OMG (Object Management Group):**
 - **Orígenes:** UML surge como resultado de la fusión de varias propuestas previas en la década de los 90.
- **Aplicación:**
 - Usado en empresas de desarrollo software.
 - Debido a que los proyectos software que se abordan en empresas son amplios y complejos, las soluciones y modelos se simplifican con UML.
 - Multitud de herramientas CASE UML.



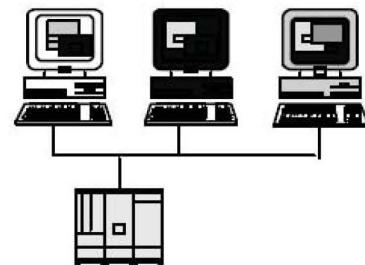
3. UML

3.1 Introducción

"El modelado captura las partes esenciales del sistema"



Proceso de Negocios



Sistema Computacional

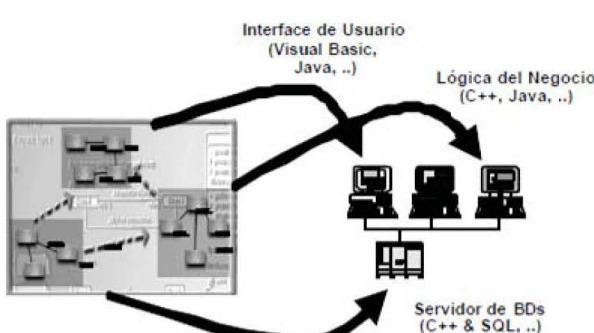
Alejandro Cardo Grau

Entornos de Desarrollo

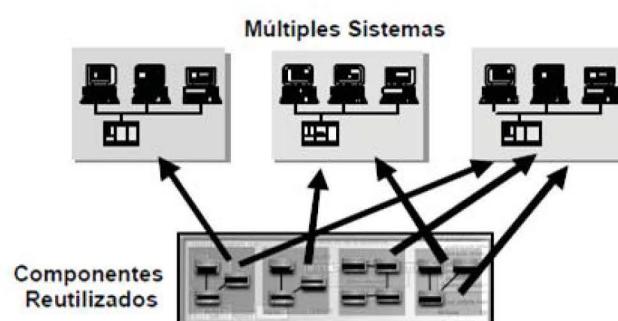
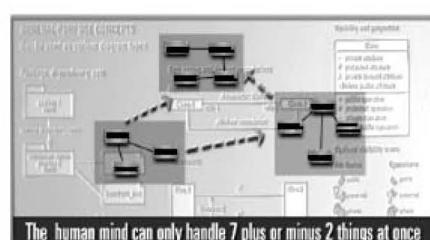
3. UML

3.1 Introducción

Manejar la complejidad



"Modelar el sistema independientemente del lenguaje de implementación"



Promover la Reutilización

Alejandro Cardo Grau

Entornos de Desarrollo

3. UML

3.1 Introducción

- El código fuente de la implementación del sistema es el modelo más detallado del sistema (y además es ejecutable). Sin embargo, se requieren otros modelos para llegar a él.



- Un diagrama UML es una representación gráfica de una colección de elementos del modelado que se dibuja para capturar una vista del sistema.
- Para modelar un diagrama UML se debe:
 - Describir aspectos importantes del sistema con un apropiado nivel de detalle.
 - Identificar las relaciones de trazabilidad entre requisitos y diagramas.

Alejandro Cardo Grau

Entornos de Desarrollo

3. UML

3.2 Herramientas CASE para el modelado UML. Estándares.

- Herramientas CASE para el modelado UML:
 - Aplicación:
 - Las herramientas de ingeniería de software asistida por computadora (CASE), ayudan al proceso del desarrollo del proyecto software durante todos los pasos del ciclo de vida.
 - Herramientas CASE UML:
 - Integradas al IDE:
 - Eclipse:
 - ArgoEclipse
 - UML2 Tools SDK
 - Más plugins: <http://marketplace.eclipse.org/search/site/uml>
 - Microsoft Visual Studio .NET
 - NetBeans IDE UML
 - Independientes:
 - Microsoft Visio
 - ArgoUML
 - UMLPad
 - Generadores Online:
 - [YUML](#)
 - [WebSequenceDiagram](#) (Diagramas de secuencia)

Alejandro Cardo Grau

Entornos de Desarrollo

3. UML

3.2 Herramientas CASE para el modelado UML. Estándares.

Features [edit]									
Name	UML 2	MDA	XMI	Templates	Languages generated	Reverse engineered languages [clarification needed]	Can be integrated with	Details	
AgileJ StructureViews	No	No	Custom reverse-engineered class-diagrams — Java/Eclipse/Agile.		Unknown	Unknown	Java	Eclipse	Batch production of diagrams, Emphasis on filtering, Diagram tailoring while viewing in a browser
Altova UModel	Yes	Yes	Yes	Yes	Java, C#, Visual Basic	Java, C#, Visual Basic	Eclipse, Visual Studio	Also supports business process modeling, SysML, and database modeling	
ArgoUML	No	Unknown	Yes	Unknown	C++, C#, Java, PHP4, PHP5, Python, Ruby	Java (other languages with plugins)	Unknown	Closely follows the UML standard	
Artisan Studio	Yes	Yes	Yes	Yes	Ada, C, C++, C#, Java,., IDL, SQL, VB	Ada, C, C++, C#, Java,., IDL, SQL, VB	Mathworks Simulink, DOORS, Microsoft Word/Excel	Runs live on a highly scalable, multi-user database. UML, SysML & UPDM modeling. Diagram template driven code synchronization.	
astah*	Yes	Unknown	Yes	Unknown	Java, C++, C#	Java, C++, C#		Mind Mapping, ER Diagram, DFD, Flowchart, CRUD, Traceability Map, Requirement Diagram and Requirement table. Provides API and Plugins. RTF, HTML Export.	
ATL	Yes	No	Yes	No	Unknown	Unknown	Available from the Eclipse M2M project (Model to Model).	Can transform UML & EMF models into other models. It has a repository of transformations called ZOO about a large set of common industrial concerns and educational labs.	
Borland Together	Yes	Yes	No	Yes	Java 6, C++, CORBA	Unknown	Eclipse and MS VS.NET 2005		
BOUML	Yes	Yes	Yes	Yes	C++, Java, PHP, IDL, Python	C++, Java, PHP	Unknown	Solid code roundtrip, fast.	
CaseComplete	Unknown	Unknown	Export	Unknown	Unknown	Unknown	Unknown	Provides management and editing of use cases, their flow of events, and referenced requirements. Supports use case and activity diagrams.	
Creately for UML	Unknown	Unknown	Unknown	Unknown	Unknown	Unknown	Unknown		

Comparativa de Herramientas de Modelado UML (2013)

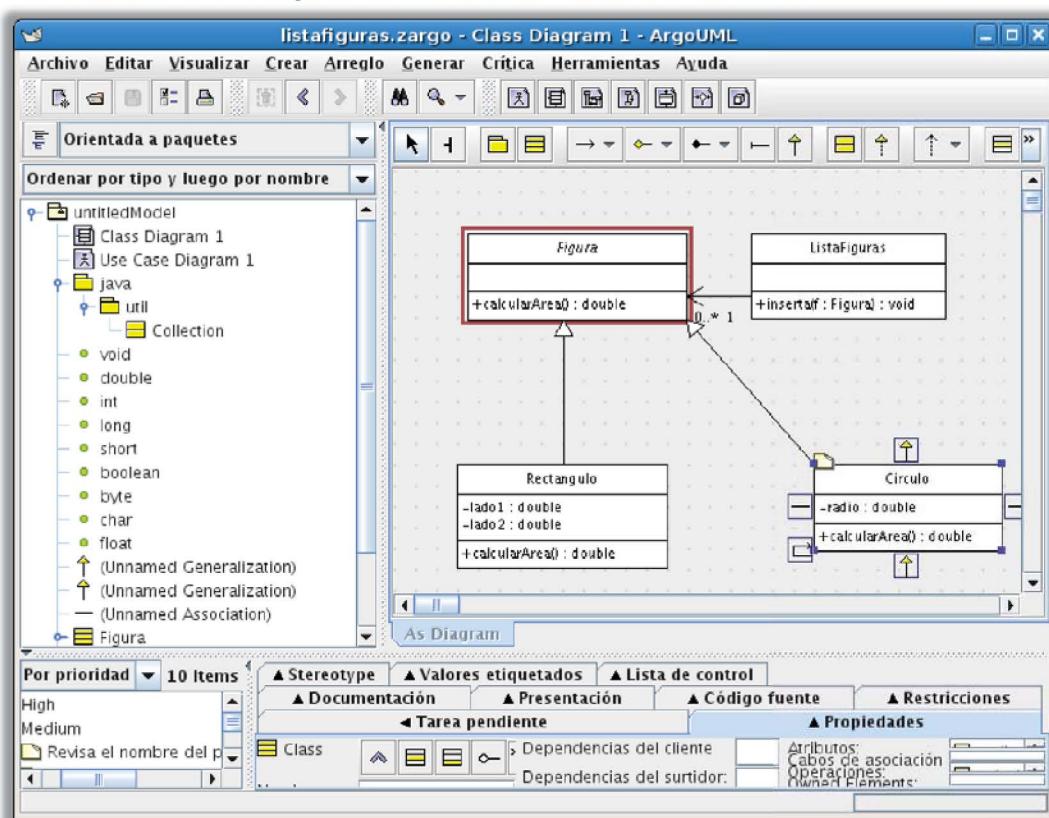
http://en.wikipedia.org/wiki/List_of_UML_tools

Alejandro Cardo Grau

Entornos de Desarrollo

3. UML

3.2 Herramientas CASE para el modelado UML. Estándares.



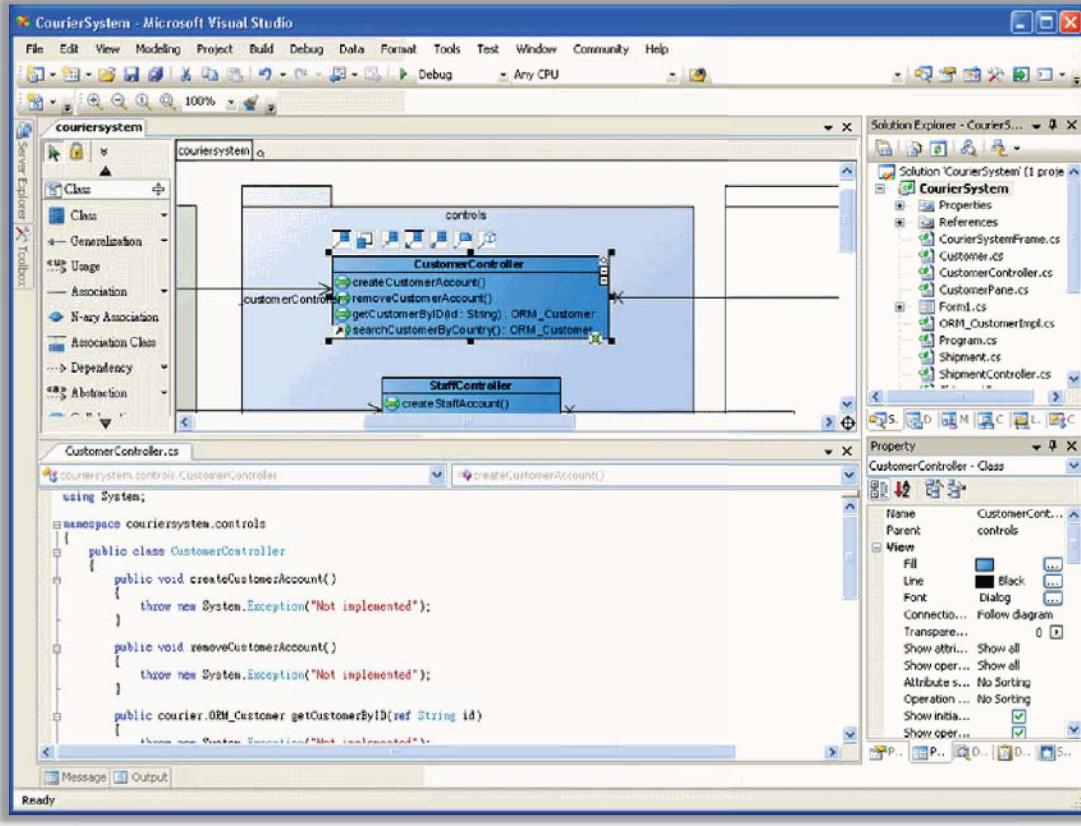
ArgoUML: <http://argouml.tigris.org> (necesario máquina virtual JRE)

Alejandro Cardo Grau

Entornos de Desarrollo

3. UML

3.2 Herramientas CASE para el modelado UML. Estándares.



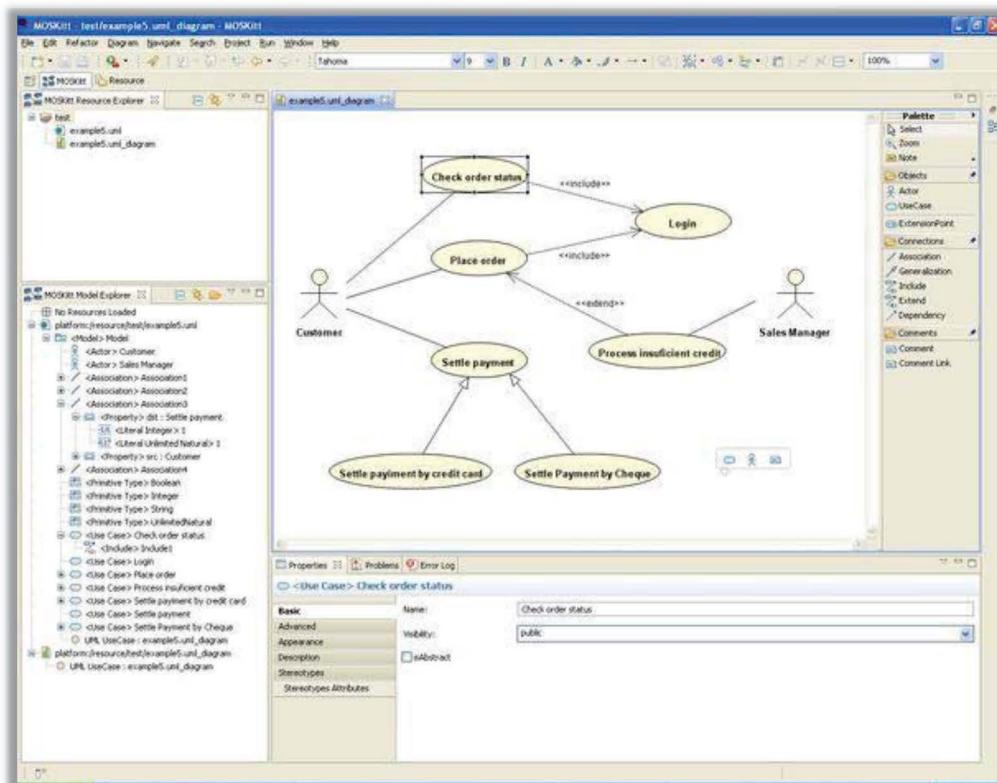
Microsoft Visual Studio .NET

Alejandro Cardo Grau

Entornos de Desarrollo

3. UML

3.2 Herramientas CASE para el modelado UML. Estándares.



Proyecto MOSSKit (adaptación española del UML2 Tools SDK para el IDE Eclipse):

<http://www.prodevelop.es/es/productos/moskitt>

Alejandro Cardo Grau

Entornos de Desarrollo

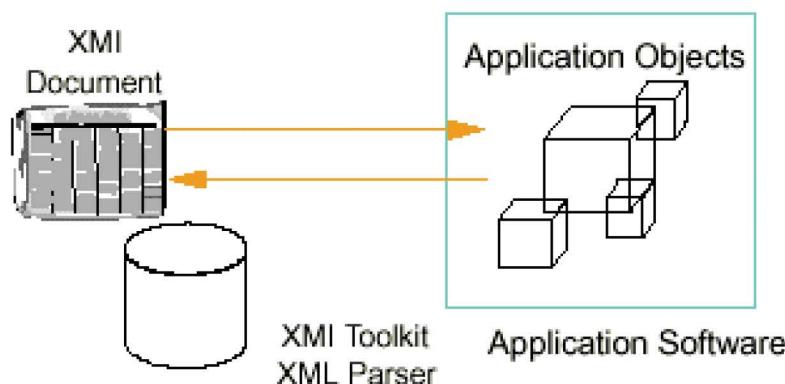
3. UML

3.2 Herramientas CASE para el modelado UML. Estándares.

- **Estándares:**

- **XMI:**

- Para evitar problemas en el intercambio de diagramas generados por diferentes herramientas CASE (donde participan múltiples equipos de trabajo) se ideó el lenguaje de marcas XMI (*XML Metadata Interchange*) basado en XML.
- Las últimas herramientas CASE UML permiten importar, exportar y compartir modelos de diagramas UML XMI, aunque con limitaciones.
- XMI genera una imagen vectorial del diagrama UML, basada en **SVG**.

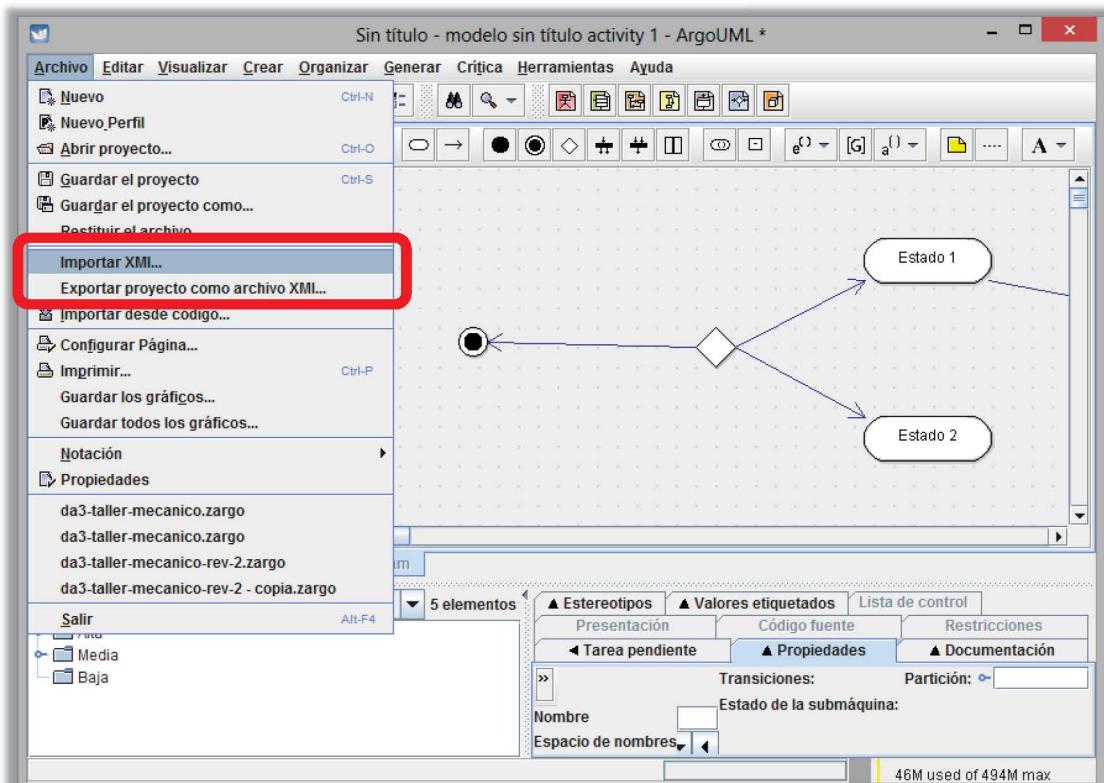


Alejandro Cardo Grau

Entornos de Desarrollo

3. UML

3.2 Herramientas CASE para el modelado UML. Estándares.



Importación y Exportación de Modelos XMI UML en la herramienta CASE ArgoUML

Alejandro Cardo Grau

Entornos de Desarrollo

3. UML

3.3 Metodologías relacionadas con UML

- **Metodologías relacionadas:**

- CMMI
- **Metodologías ágiles**

- XP
- SCRUM

- **Métrica 3**

- Usado como modelo de desarrollo en las aplicaciones de las administraciones públicas, junto a CMMI.
- Basado en normativas de calidad ISO/IEC 12207 (*Information Technology - Software Life Cycle Processes*) así como en la norma ISO/IEC 15504.

- **Características Métrica 3:**

- Establece actuaciones en un conjunto de **procesos, actividades y tareas** para garantizar la calidad final de un producto software.
- La metodología comprende tanto el paradigma de **desarrollo orientado a objetos** como el estructurado.

3. UML

3.3 Metodologías relacionadas con UML

- **Métrica 3 - Etapas y Procesos Principales:**

- **Etapas y procesos principales:**

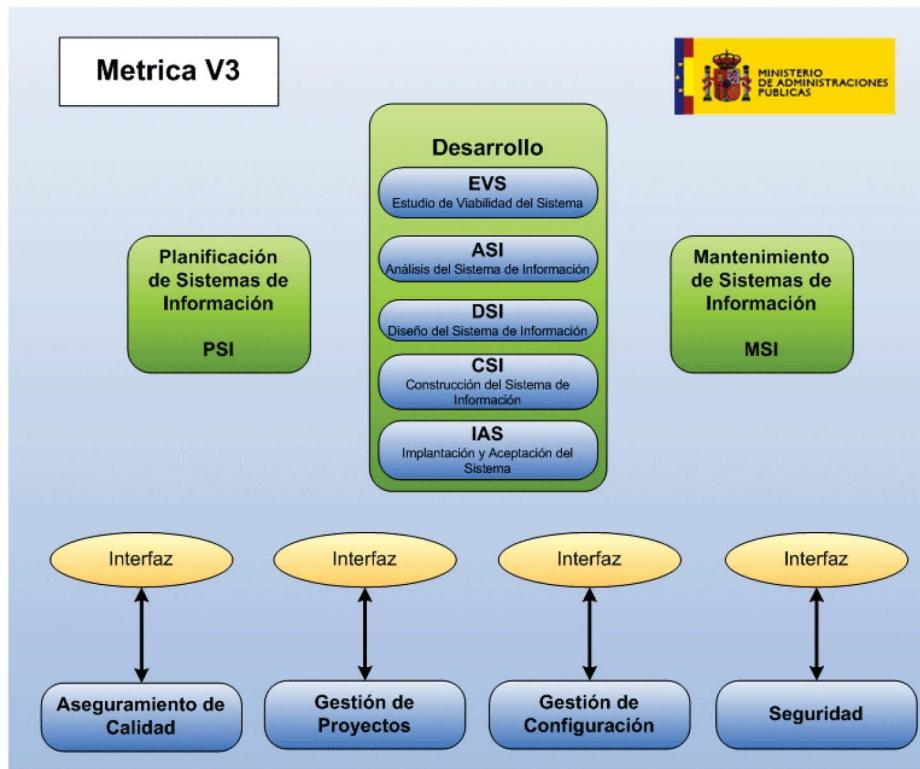
- PLANIFICACIÓN DE SISTEMAS DE INFORMACIÓN (**PSI**):
 - Proporciona un marco estratégico inicial de referencia.
- DESARROLLO DE SISTEMAS DE INFORMACIÓN.
- MANTENIMIENTO DE SISTEMAS DE INFORMACIÓN (**MSI**).

- Dentro del **DESARROLLO DE SISTEMAS DE INFORMACIÓN**:

- Estudio de Viabilidad del Sistema (**EVS**):
 - Se estudian las necesidades y la arquitectura tecnológica básica necesaria, teniendo en cuenta restricciones económicas, técnicas, legales y operativas.
- **Análisis del Sistema de Información (ASI)**:
 - Especificación detallada del sistema de información.
- **Diseño del Sistema de Información (DSI)**:
 - Definición de la arquitectura del sistema y especificación detallada de los componentes del mismo.
- Construcción del Sistema de Información (**CSI**):
 - Implementación, **codificación** y ejecución de pruebas de los componentes.
- Implantación y Aceptación del Sistema (**IAS**)

3. UML

3.3 Metodologías relacionadas con UML



Metodología Métrica v3 - Procesos

Alejandro Cardo Grau

Entornos de Desarrollo

3. UML

3.3 Metodologías relacionadas con UML

- Métrica 3 - UML:**

- Usaremos el modelado visual orientado a objetos de los **diagramas UML**, que veremos durante el curso, para:

Diagrama UML	Descripción	Proceso Métrica
Diagrama de actividad	Modelar los procesos de negocio y flujos de trabajo del sistema	ASI
Diagrama de casos de uso	Definir los requisitos y cómo interactúan los diferentes actores en el sistema	ASI
Diagrama de clases	Diseñar una solución estructural para construir el sistema y modelar el esquema lógico de la base de datos	ASI (modelo estático: clases , atributos y relaciones) DSI (clases detalladas)
Diagrama de estados Diagrama de secuencia	Modelar el comportamiento dinámico del sistema y los estados	ASI DSI

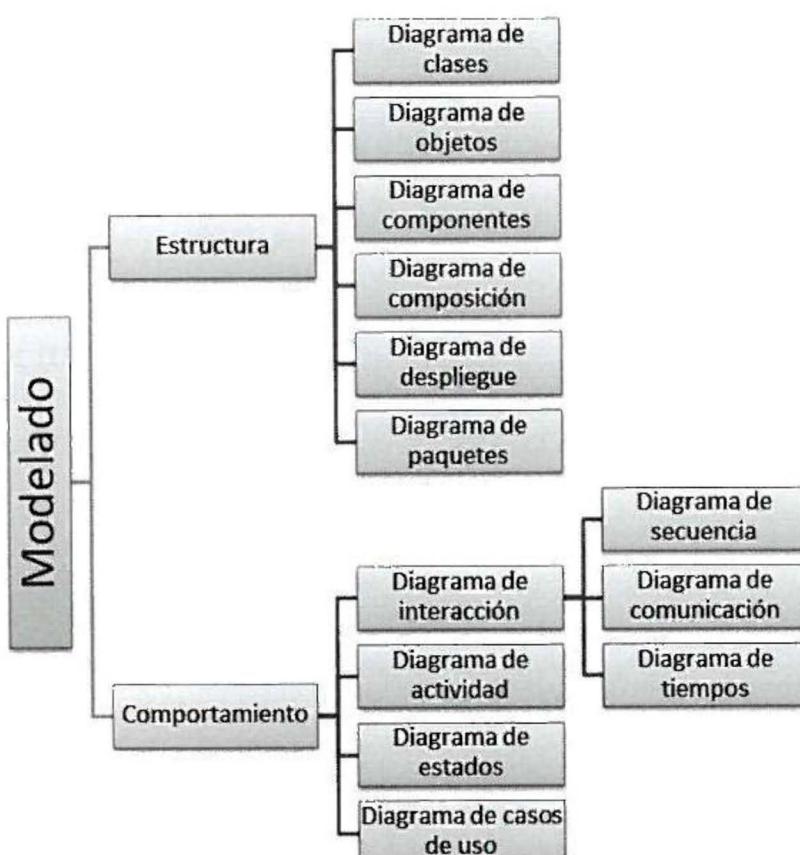
Alejandro Cardo Grau

Entornos de Desarrollo

- **EJERCICIO 3 (FORO):**

- ¿Sabéis lo que es unalicitación o propuesta de oferta para un desarrollo de proyecto software? Son personal laboral dedicado exclusivamente a la búsqueda de nuevas ofertas, donde examinan la propuesta a través de un pliego de condiciones realizadas de otras empresas multinacionales o de la Administración Pública para el desarrollo y mantenimiento de un proyecto software.
- Una vez ese pliego de condiciones técnicas sea conocido por las empresas. Las consultoras informáticas y de gestión se encargan de ofrecer distintas ofertas a partir de un análisis previo y un estudio intenso previo inicial de lo serán las primeras partes del ERS (Documento Requisitos del Sistema). Si la consultora es elegida para llevar a cabo el proyecto, por haber diseñado una buena oferta, realizará primeramente un buen ERS como factor prioritario (ajustado a la oferta dada).
- ¿Conoceis algún ejemplo de enlace web sobre propuesta de proyecto software a través de un pliego de condiciones de Administraciones Públicas o de empresas multinacionales? ¿Qué empresa fue la elegida en mantener ese proyecto? ¿Qué demandaron para hacer la oferta entre las empresas?
- Comentad alguna de las propuestas que existen o han existido en el desarrollo de proyectos software o de características relacionadas con la mejora de la arquitectura tecnológica de los sistemas.
 - Se adjunta un ejemplo de pliego de condiciones técnicas.

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.



4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

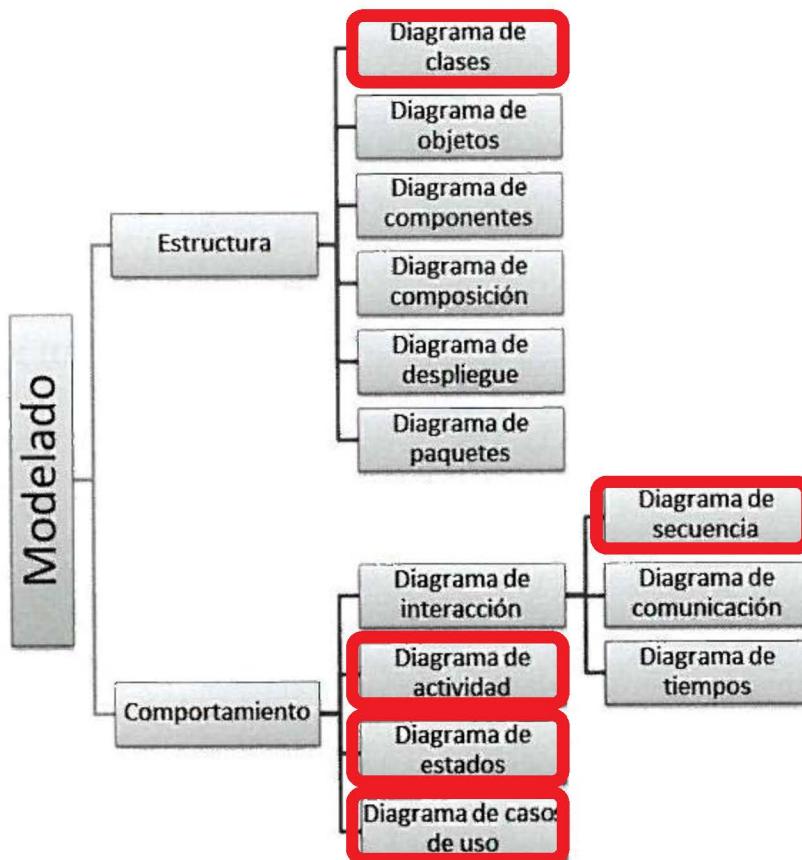
- **Modelado estático o de estructura:**

- Aplicación:
 - Conceptos del dominio del problema y las relaciones entre éstos.
 - El **dominio del problema** fija el conjunto necesario de conceptos interrelacionados para modelarse mediante:
 - Entender el negocio del cliente
 - Entender las necesidades requeridas → Tipos de Requisitos
 - Proponer una solución adecuada.
- Tipos de diagramas:
 - **Diagramas de clases**

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

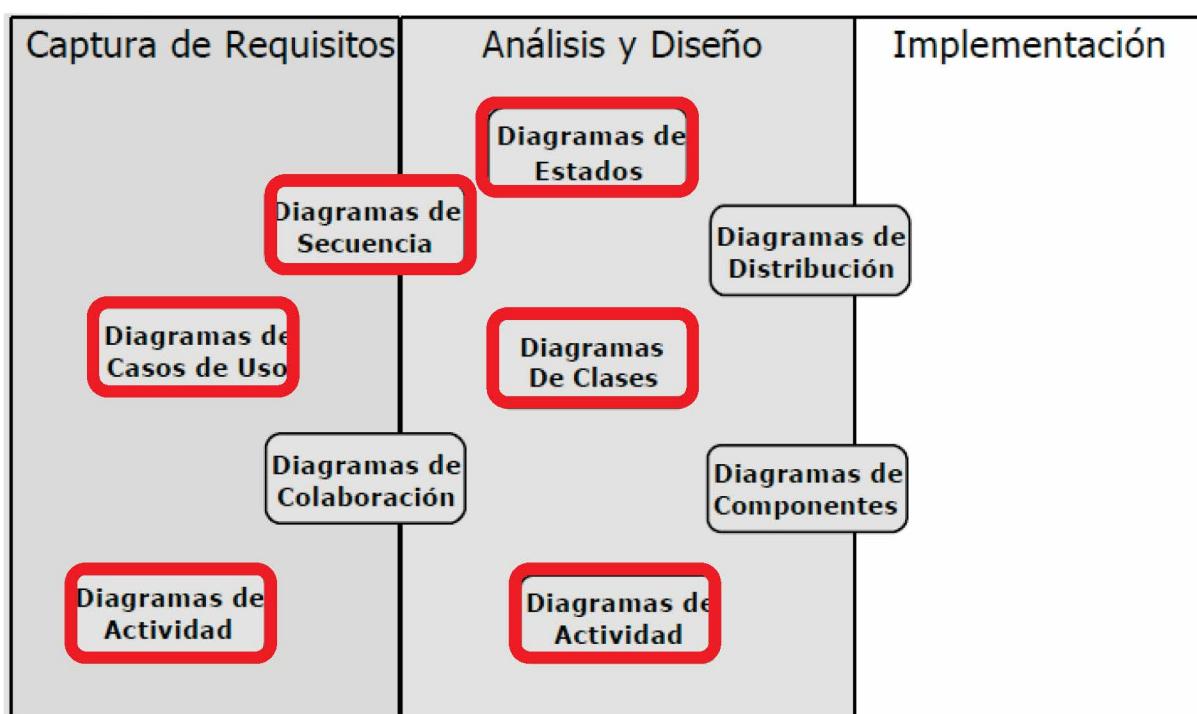
- **Modelado dinámico o del comportamiento:**

- Aplicación:
 - Comportamiento dinámico del sistema en función del tiempo.
 - Captura los tipos de interacción y el estado instantáneo de un modelo mientras “se ejecuta” a través del tiempo
- Tipos de diagramas:
 - **Diagramas de actividad**
 - **Diagramas de estados**
 - **Diagrama de casos de uso**
 - Diagramas de interacción:
 - **Diagramas de secuencia**
 - **Diagramas de colaboración**



Alejandro Cardo Grau

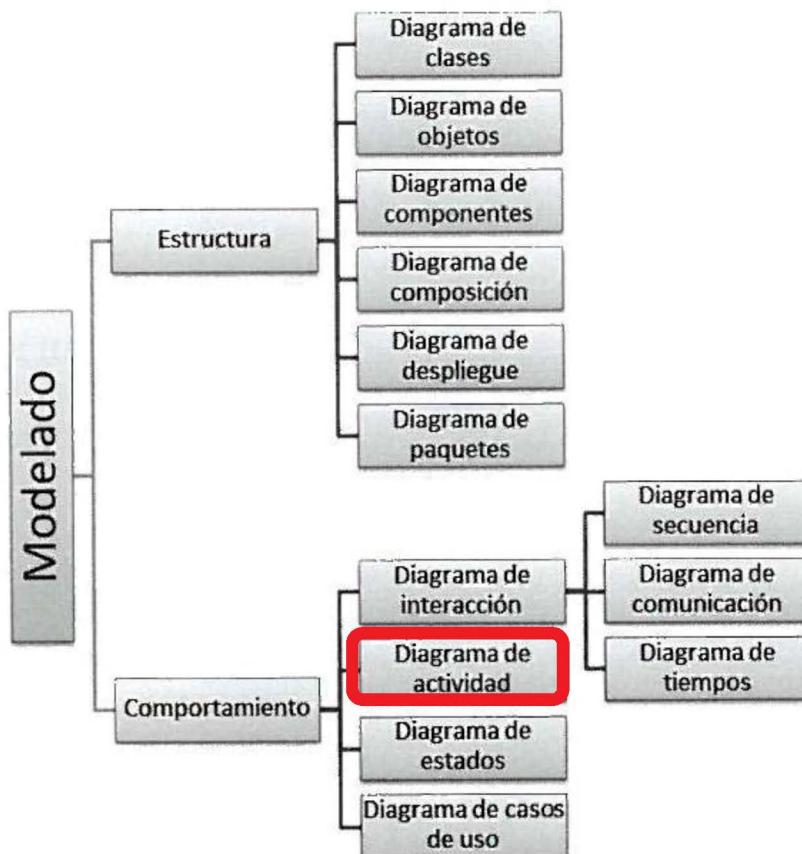
Entornos de Desarrollo



Tipos de diagramas UML y su extensión usados en las fases del ciclo de vida clásico

Alejandro Cardo Grau

Entornos de Desarrollo

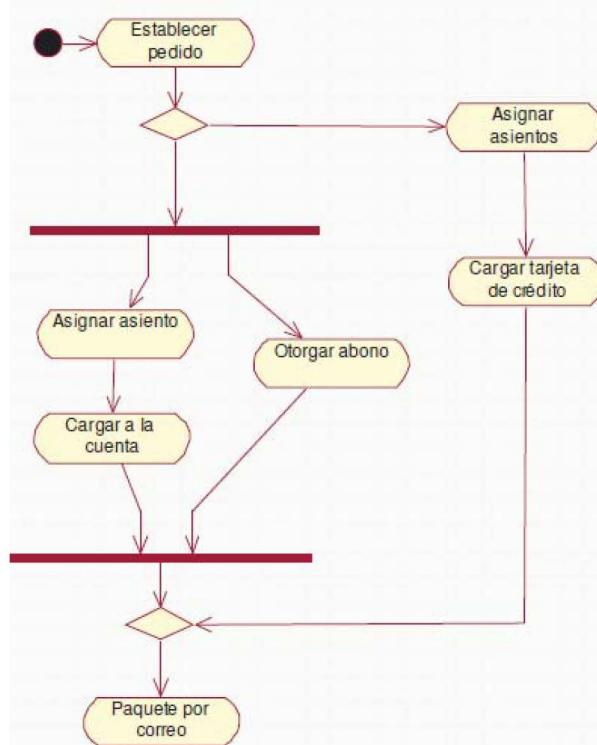
4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

- **Diagrama de actividades:**
 - Modela una **secuencia o flujos de trabajo (workflow)** de cada proceso de negocio, mediante pasos secuenciales y concurrentes.
 - Modelan el **comportamiento a alto nivel** de la ejecución de un sistema:
 - No profundiza en los detalles internos de los mensajes, como hacen los diagramas de interacción.
 - Características:
 - La mayoría de los estados son **actividades**.
 - Las **transiciones** se disparan por la terminación de acciones.



Proceso de negocio:
Proceso de gestión de pedidos

Alejandro Cardo Grau

Entornos de Desarrollo

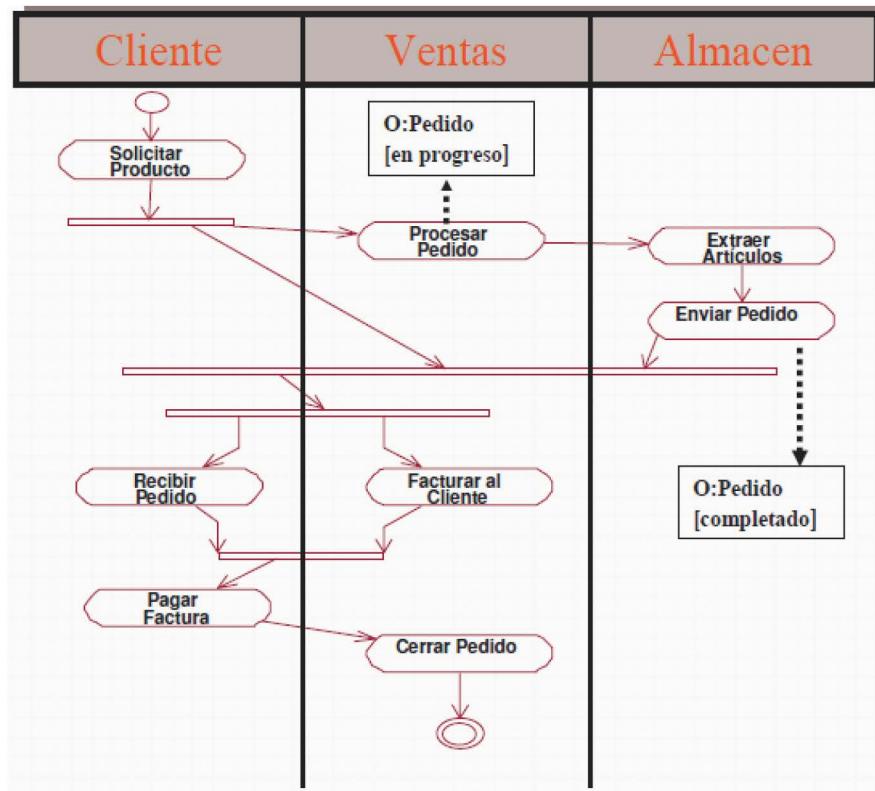
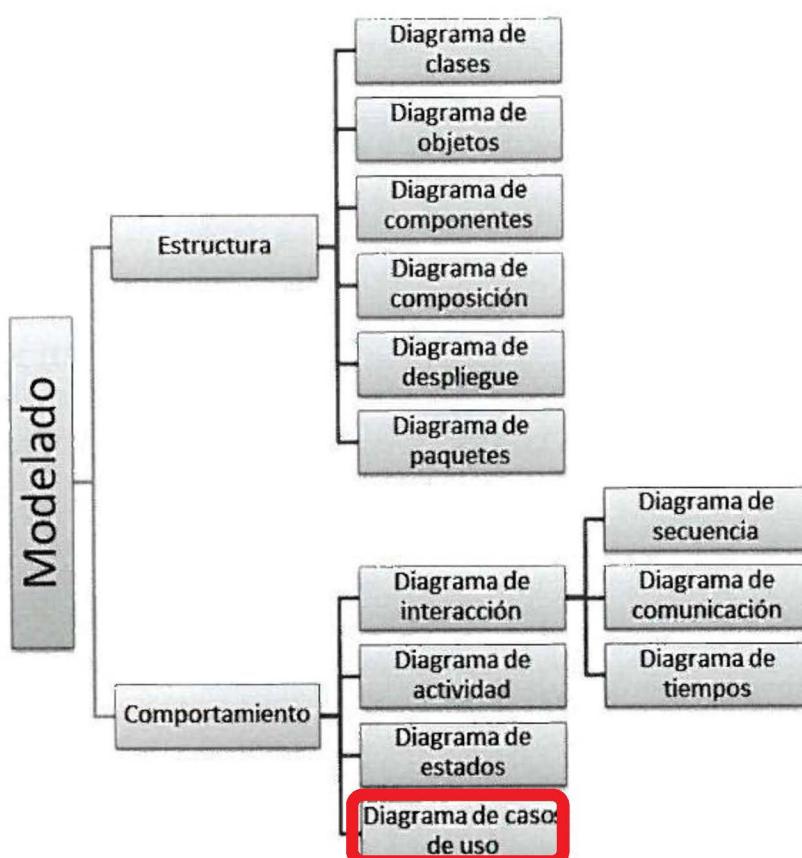


Diagrama de actividades

Alejandro Cardo Grau

Entornos de Desarrollo



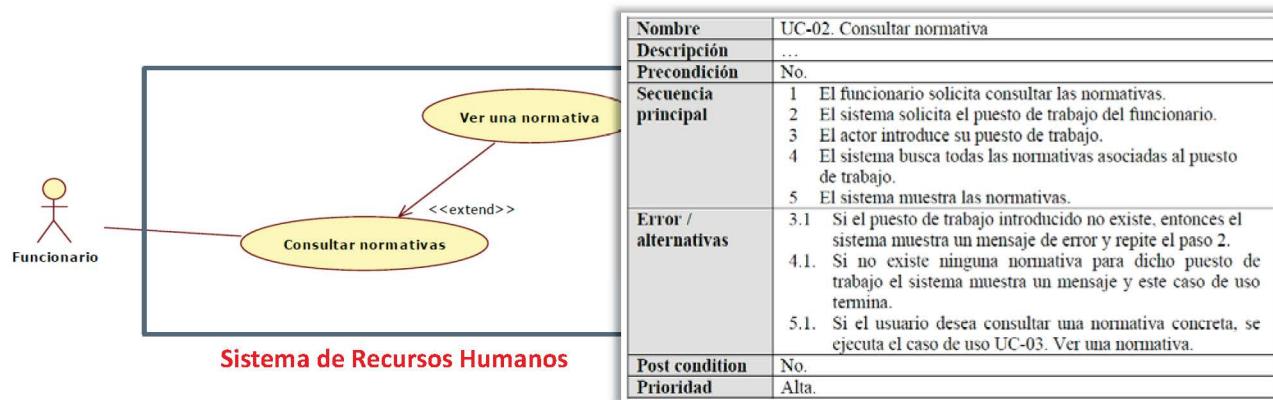
Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

- **Diagrama de casos de uso:**

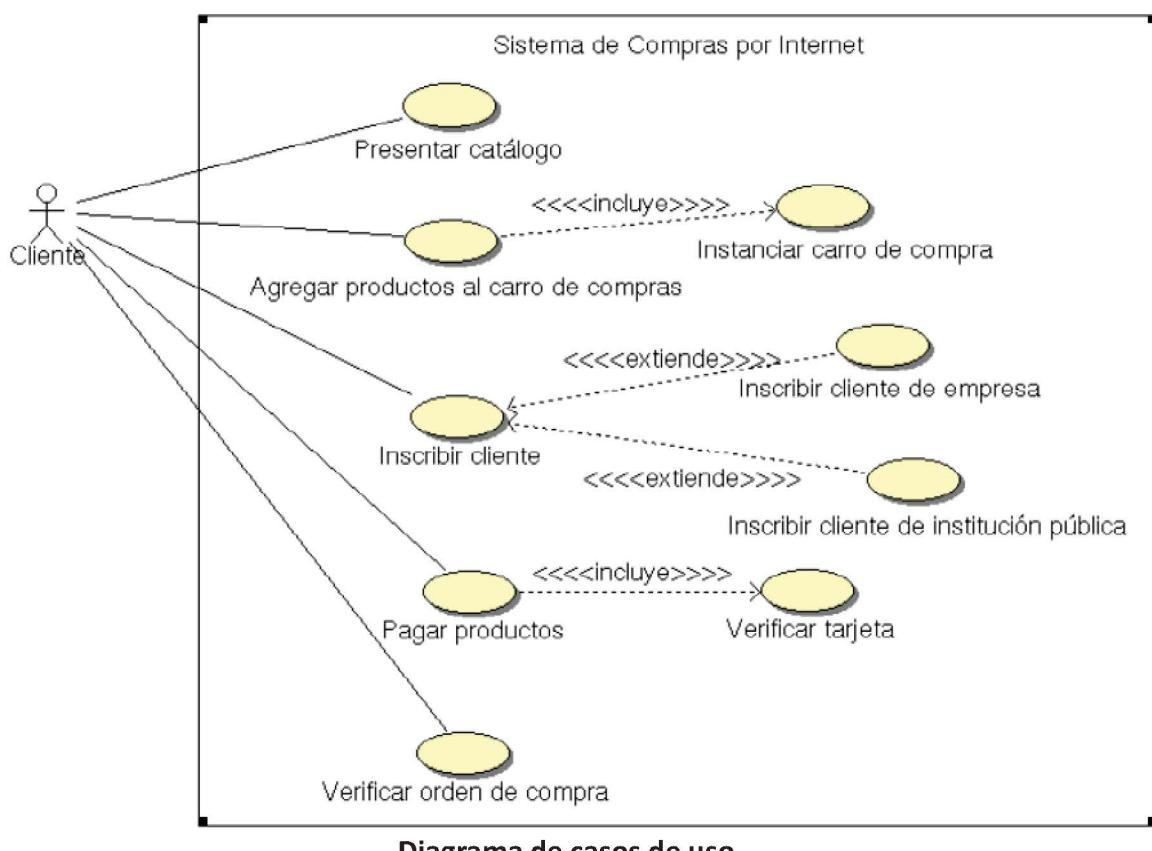
- Los diagramas de casos de uso representan cómo interactúan los diferentes **actores** en un **sistema** en cada caso de uso.
- Es una forma de representar los **requisitos funcionales** del sistema, es decir, definen las funciones del sistema mediante:
 - Las **acciones** que puede realizar cada actor dentro del sistema.
 - Una **acción que representa a un caso de uso**.



Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.



Alejandro Cardo Grau

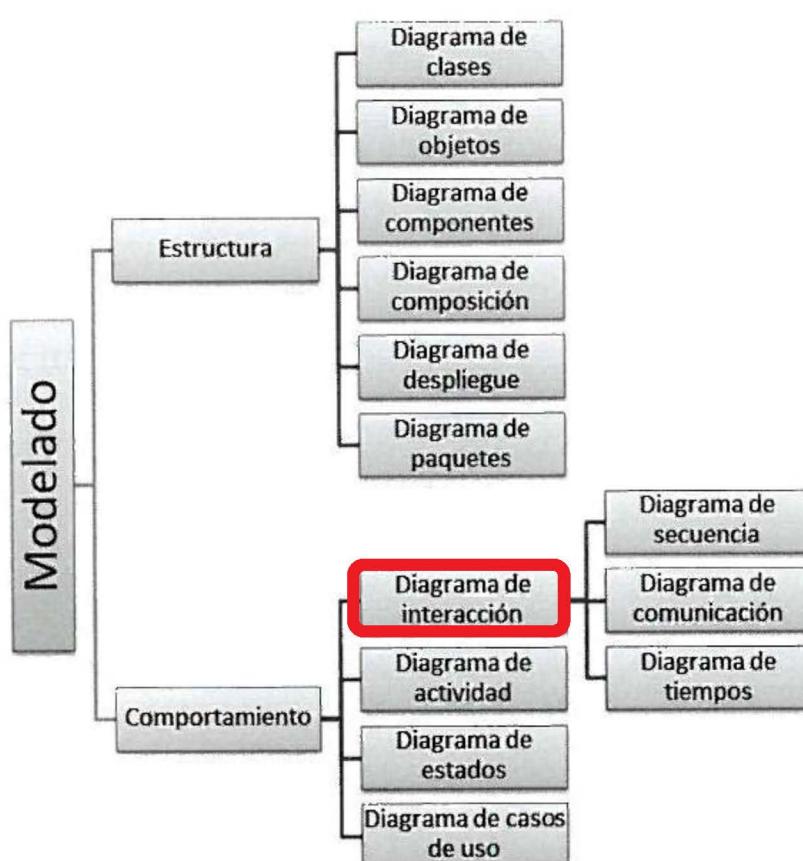
Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.**Caso de uso detallado: Consultar normativa**

- La documentación especifica las interacciones de los casos de uso definidas en secuencia de pasos:
 - Principio del análisis del comportamiento del sistema.



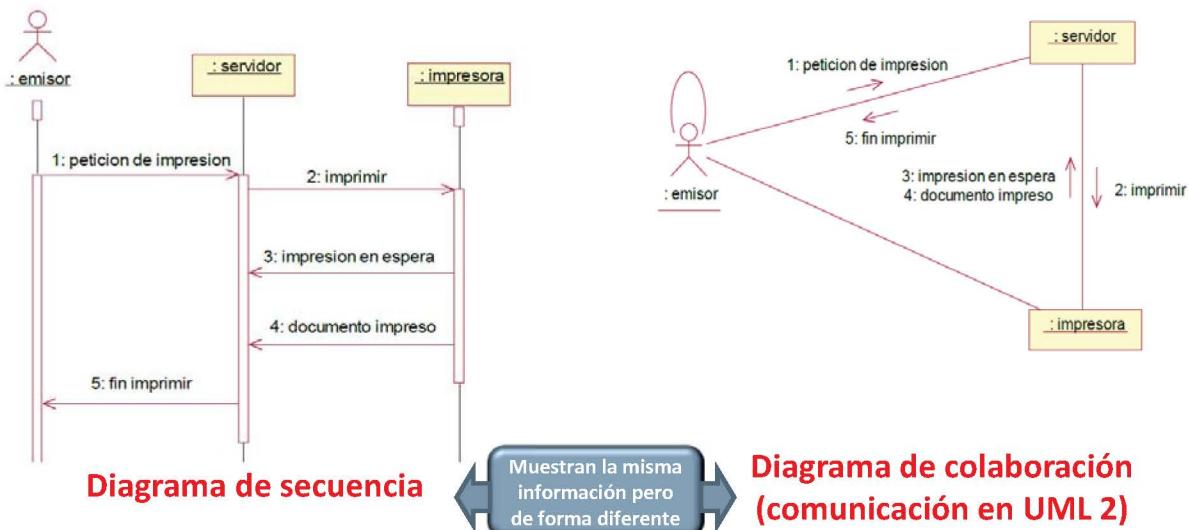
Nombre	UC-02. Consultar normativa
Descripción	...
Precondición	No.
Secuencia principal	<ol style="list-style-type: none"> El funcionario solicita consultar las normativas. El sistema solicita el puesto de trabajo del funcionario. El actor introduce su puesto de trabajo. El sistema busca todas las normativas asociadas al puesto de trabajo. El sistema muestra las normativas.
Error / alternativas	<ol style="list-style-type: none"> Si el puesto de trabajo introducido no existe, entonces el sistema muestra un mensaje de error y repite el paso 2. Si no existe ninguna normativa para dicho puesto de trabajo el sistema muestra un mensaje y este caso de uso termina. Si el usuario desea consultar una normativa concreta, se ejecuta el caso de uso UC-03. Ver una normativa.
Post condition	No.

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

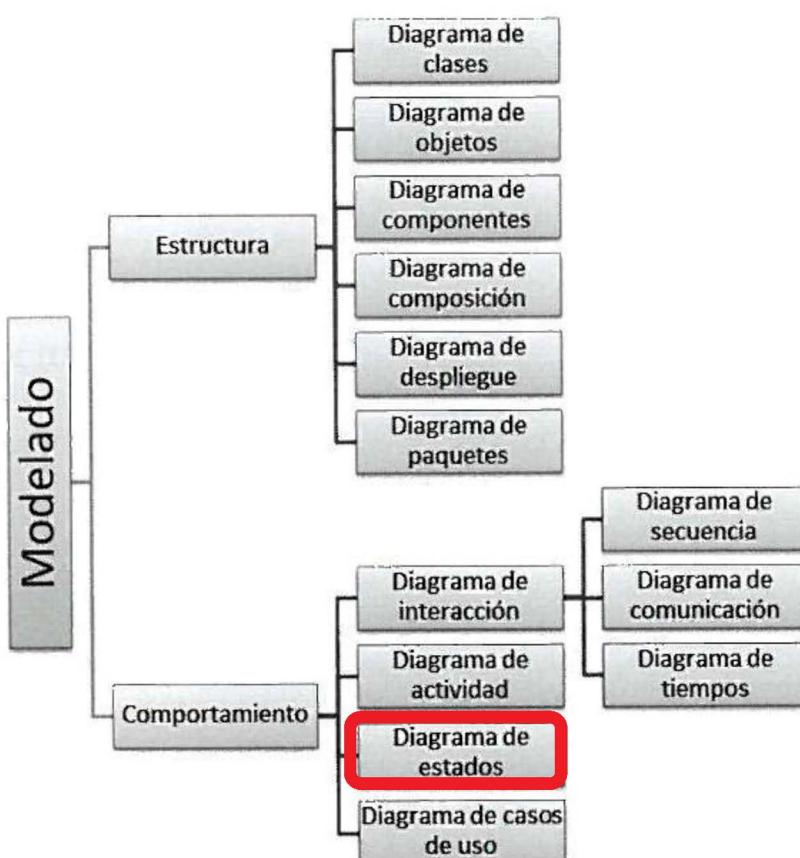
- **Diagramas de interacción:**

- Los diagramas de interacción representan en detalle un escenario para un caso de uso.
- Permiten verificar la coherencia del sistema validándolo a través del **paso de mensajes** entre objetos.
- **Tipos:**



Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

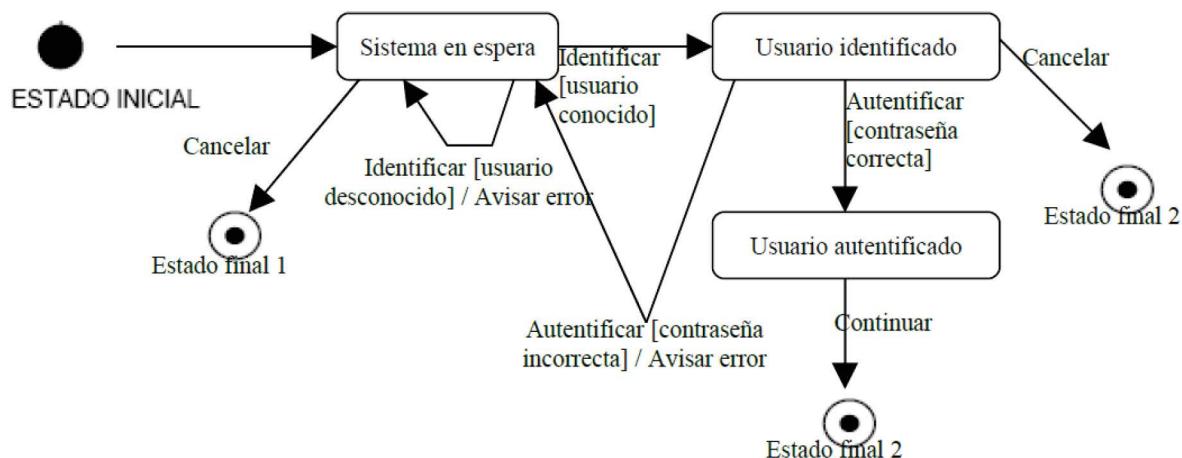
Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

- **Diagrama de estados:**

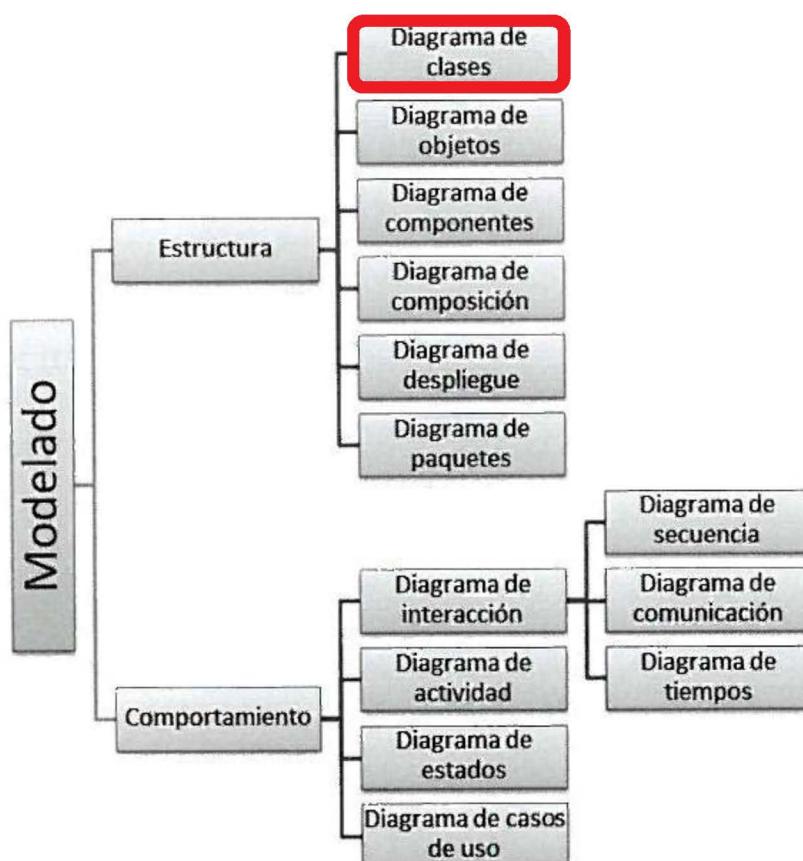
- Modelan como afecta un escenario a los **estados** que se van tomando, a partir de la respuesta de **eventos**.
- **Grafo de estados y transiciones** que describe la respuesta de un objeto, un caso de uso o de un sistema completo.



Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.



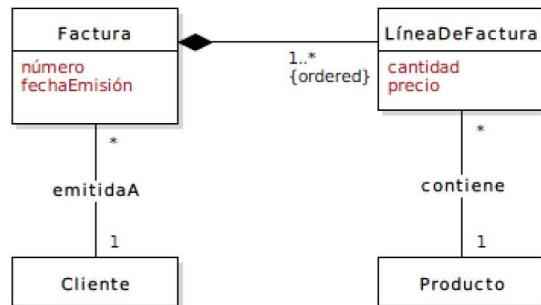
Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

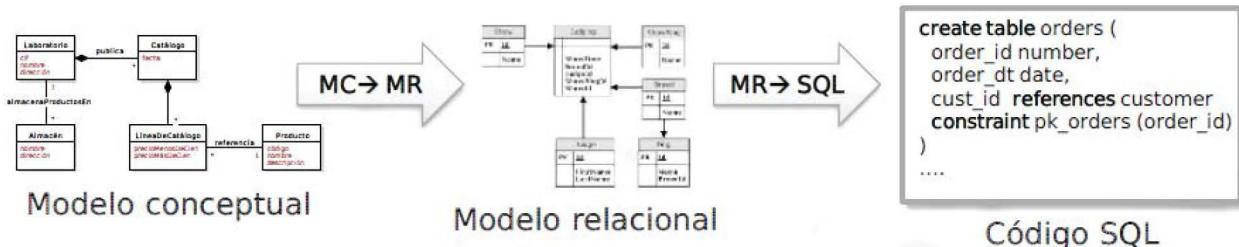
• Diagrama de clases:

- Representa el modelo conceptual y estático del sistema con sus:
 - Relaciones entre clases (enlaces).
- La definición de **clase** incluye **atributos y métodos** que definen:
 - Propiedades
 - Comportamiento



• Transformación entre modelos en BD relacionales (MDD):

- Hay herramientas CASE UML donde el modelo conceptual del diagrama de clases se puede transformar a un **modelo relacional** (**diagrama E/R**) para obtener automáticamente su **código SQL**.

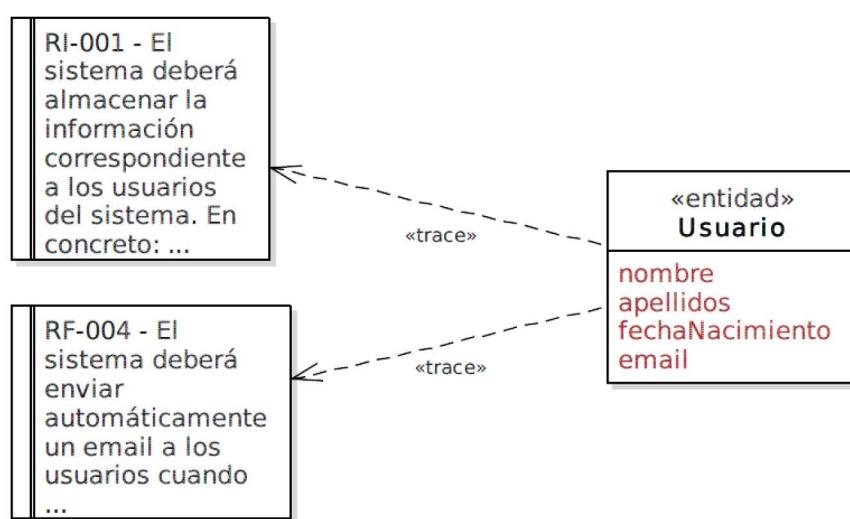


Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

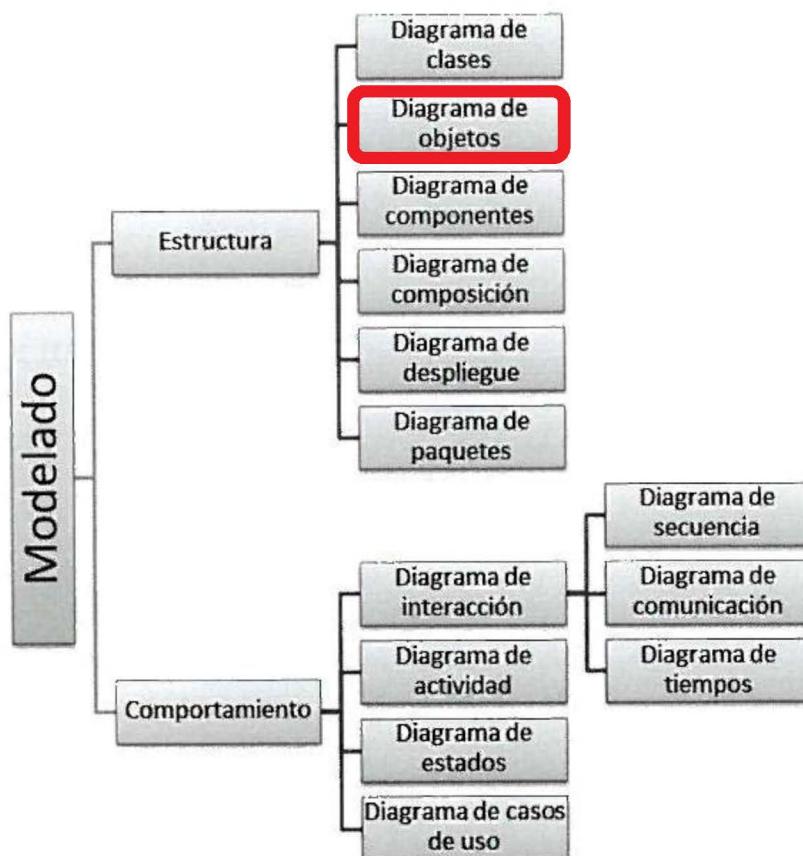
- El diagrama de clases contendrá la estructura de las diferentes clases del sistema que serán posteriormente implementadas a partir de los otros diagramas de estructura y comportamiento.
- Todo elemento del diagrama de clase debe estar relacionado y trazado para aquellos requisitos analizados que lo justifican.



Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.



Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

- **Diagrama de objetos:**

- Permiten enlazar la relación entre las instancias de las clases (**objetos**) para un escenario concreto y en algún punto del tiempo



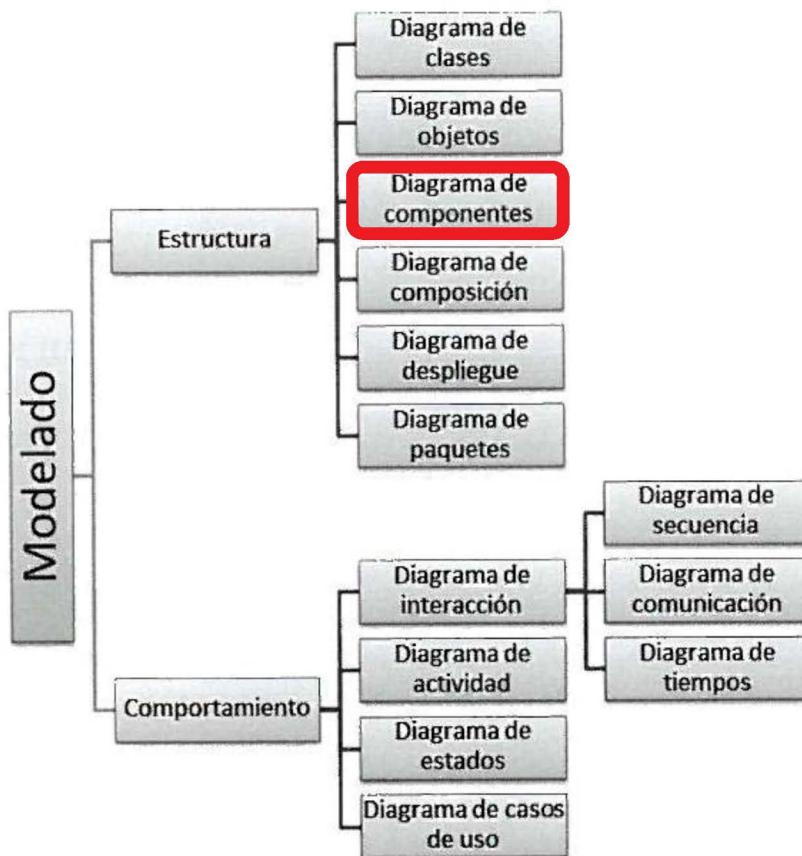
Diagrama de objetos



Diagrama de clases

Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

Alejandro Cardo Grau

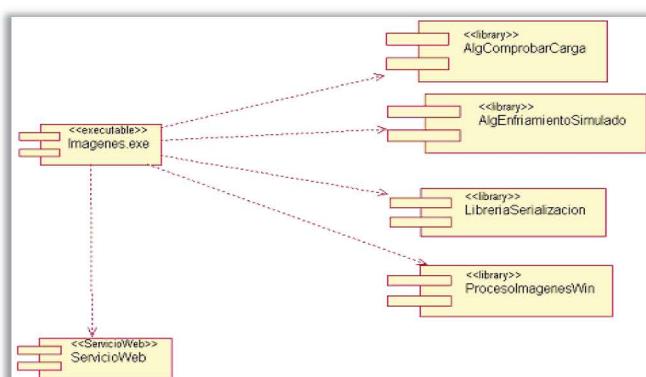
Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

- **Diagrama de componentes:**

- Los **componentes** representan el modelado de los elementos físicos que pueden hallarse en un nodo:

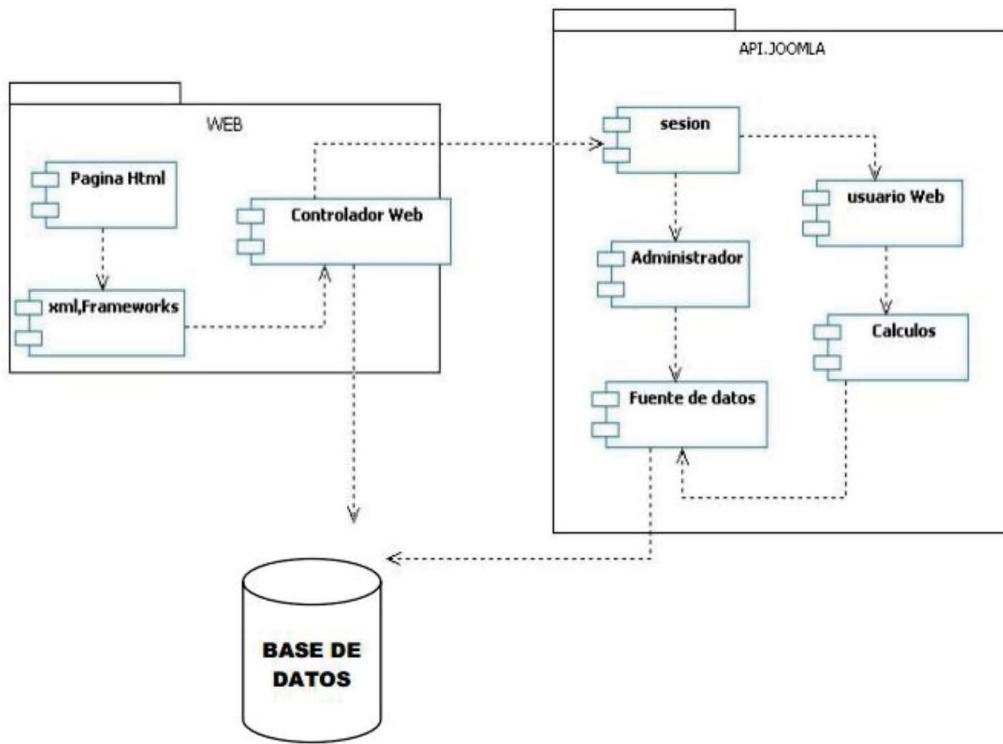
- Ejecutables
- Bibliotecas
- Tablas
- Archivos
- Documentos



- Las **relaciones de dependencia** se refieren en los diagramas de componentes para indicar que un componente usa los servicios ofrecidos por otro componente.
- Los componentes definen abstracciones, con interfaces bien definidas y que facilitan su sustitución posterior.

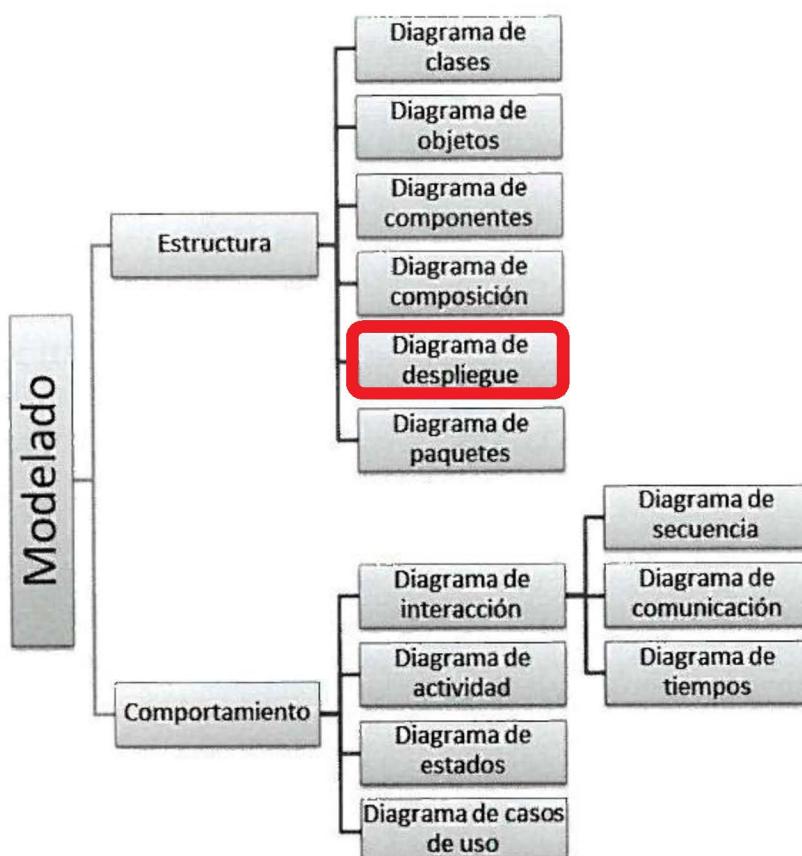
Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.**Ejemplo de Diagrama de Componentes**

Alejandro Cardo Grau

Entornos de Desarrollo

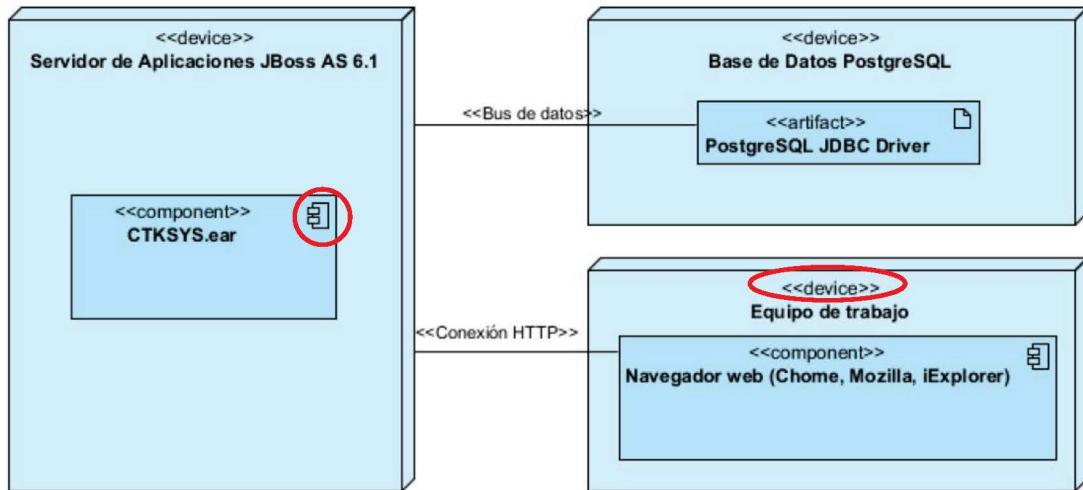
4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

Alejandro Cardo Grau

Entornos de Desarrollo

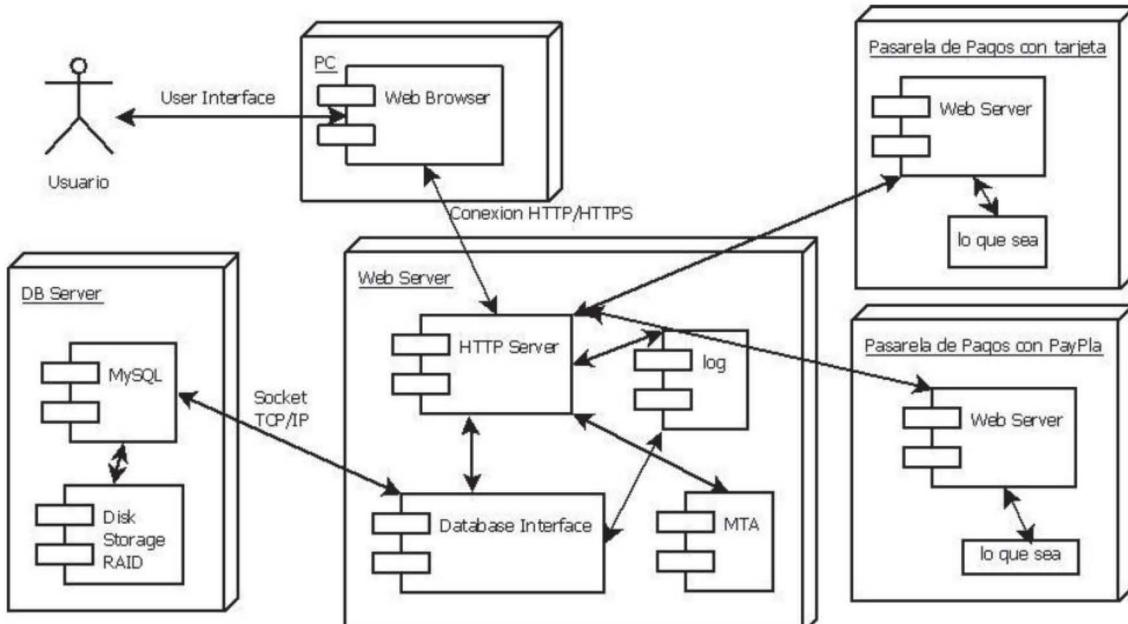
4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.**• Diagrama de despliegue:**

- Describe la **arquitectura tecnológica** del sistema usada para modelar las relaciones entre:
 - El hardware requerido del sistema: dispositivos e interconexión.
 - La asignación de los componentes software (procesos, ficheros, etc)
- Elementos:** nodo, componente, conexión, esteorotipo



Alejandro Cardo Grau

Entornos de Desarrollo

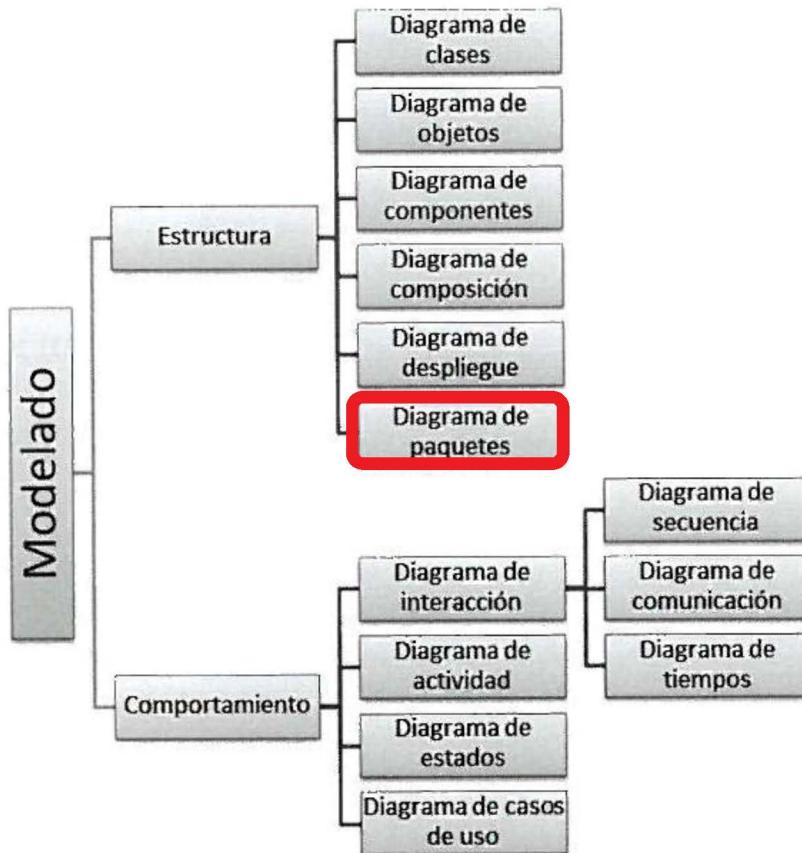
4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

Ejemplo de Diagrama de Despliegue

Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

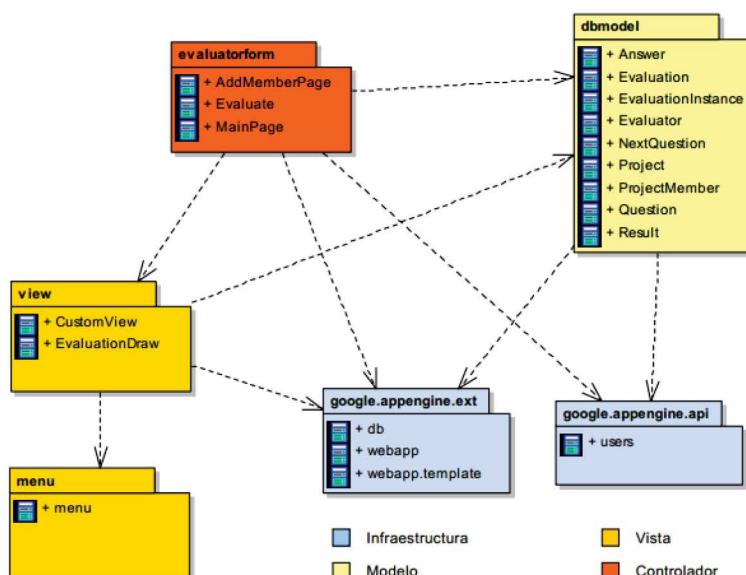


Alejandro Cardo Grau

Entornos de Desarrollo

4. TIPOS DE DIAGRAMAS. CAMPO DE APLICACIÓN.

- **Diagrama de paquetes:**
 - Un **paquete** es un grupo de elementos del modelo relacionados.
 - Puede contener uno o más tipos de clases, caso de uso, etc.
 - Un paquete puede contener otros paquetes, sin límite de anidamiento pero cada elemento pertenece a un sólo un paquete



Alejandro Cardo Grau

Entornos de Desarrollo

- **PREGUNTA 1:**

- ¿A qué tipo de diagrama UML corresponde aquel que describe el conjunto de dispositivos e interconexión de los mismos dentro del diseño de la arquitectura tecnológica del sistema?

- **PREGUNTA 2:**

- ¿A qué tipo de diagrama UML corresponde aquel que representa las funciones realizadas por los actores y el sistema?

- **PREGUNTA 3 (TEST):**

- Indicar cuál de las siguientes afirmaciones es FALSA:

- 1) El diagrama de actividad representa el flujo de control la secuencia de ejecución de acciones que describen un proceso de negocio.
- 2) El diagrama de clases es un tipo de diagrama UML que representa el modelado de comportamiento dinámico del sistema.
- 3) Tanto los diagramas de casos de uso como los de actividad se definen en el proceso de Análisis de Sistemas de Información (ASI) de Métrica.

5. REFERENCIAS



5. REFERENCIAS

- Casado Iglesias, C. (2012). *Entornos de Desarrollo*. CFGS. Editorial Ra-Ma.
- SÁNCHEZ, Salvador et. al. (2011): *Ingeniería del software. Un enfoque desde la guía SWEBOk*. Editorial Garceta.
- *Entornos de Desarrollo (IDE)*: <http://programacion-laura.blogspot.com.es/2011/08/entorno-de-desarrollo-integrado-ide.html>
- *Nuevas profesiones, aquí si hay empleo*: <http://www.abc.es/20120229/economia/abci-nuevas-profesiones-futuro-empleo-201202281538.html>