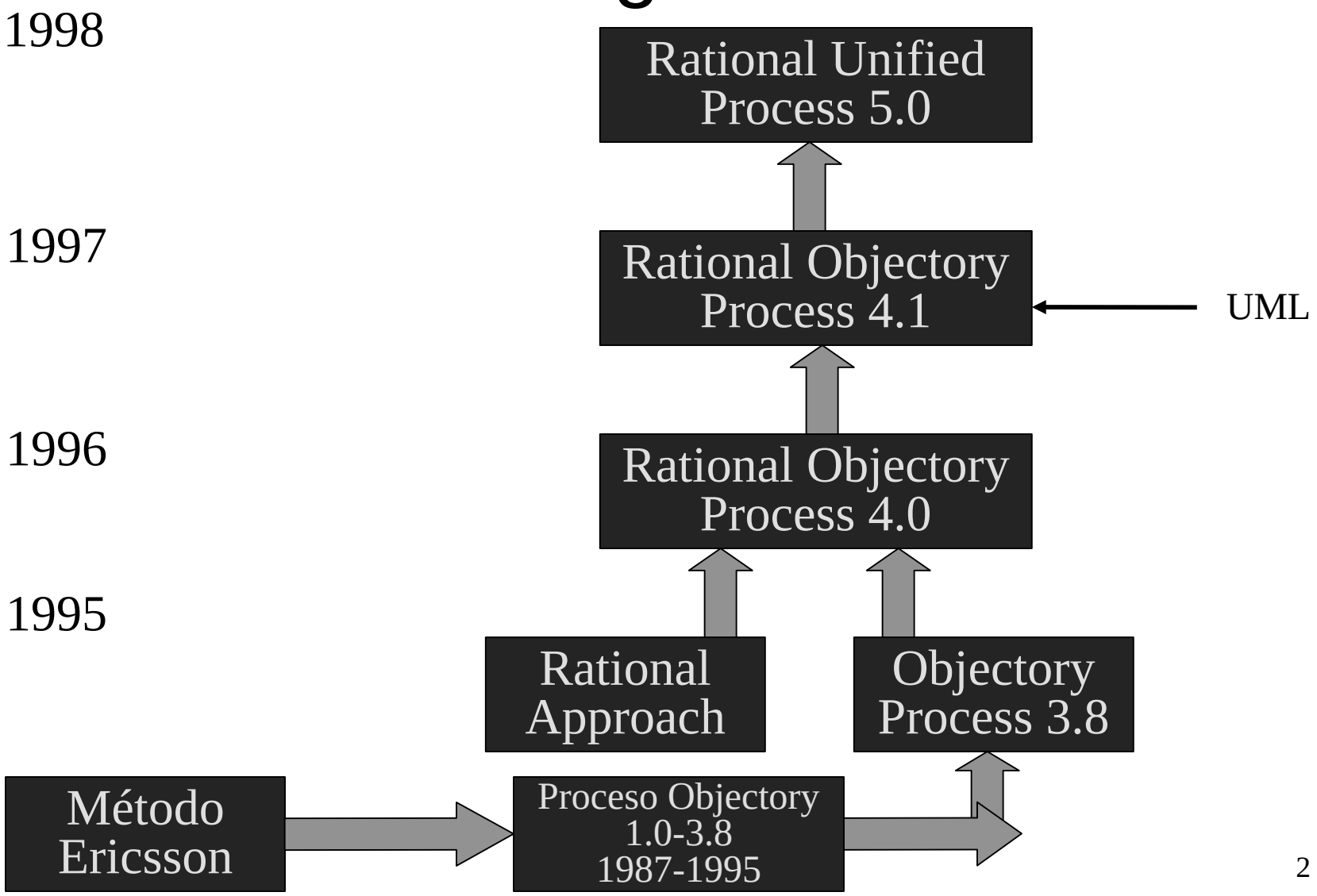
PUD: Proceso de Desarrollo Unificado

Genealogía del PUD



Relación con otros métodos

RUMBAUGH-JACOBSON-BOOCH

- Fortalezas:
  - Rumbaugh: método fuerte para producción modelos de dominio,
  - Jacobson: método fuerte para producción orientada al usuario,
  - Booch: método fuerte para producción detallada modelos de diseño orientados a objetos,

Relación con otros métodos

RUMBAUGH-JACOBSON-BOOCH

- Debilidades:
  - Rumbaugh: Simplista para el espacio de soluciones posibles,
  - Jacobson: No trata detalladamente el diseño orientado a objetos al nivel de Booch,
  - Booch: Se centra en el diseño y no en el analisis.

Relación con otros métodos

RUMBAUGH-JACOBSON-BOOCH

- PUD- Ciclo de vida fruto de fusión de mejores aspectos:
  - Casos de uso y prototipos de Jacobson,
  - Diagramas de clase y prototipos de objetos del mundo real de Rumbaugh,
  - Diagramas de interaccion de Jacobson y diagramas de objetos de Booch, para ver colaboración entre objetos,

Relación con otros métodos

RUMBAUGH-JACOBSON-BOOCH

- PUD- Ciclo de vida fruto de fusión de mejores aspectos:
  - Diagramas de estado para el control en tiempo real,
  - Modelo de clase de Booch y diagramas de clase para saber como construir el sistema.
  - Y otras de las mejores caracteristicas de estos métodos.
- Resultado:PUD- Diseños traceables a traves de cada fase

Herramientas de la metodología

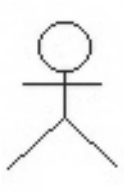
- Herramientas de apoyo al proceso:
  - Gestión de requisitos,
  - Modelado Visual,
  - Herramientas de programación,
  - Aseguramiento de la calidad,
  - Control de versiones, gestión de la configuración, seguimiento de defectos, documentación, gestión del proyecto y automatización de procesos.

UML

Lenguaje de Modelado Unificado

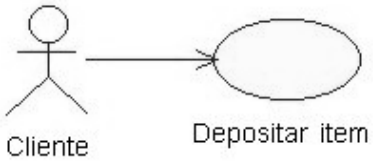
- Define una serie de diagramas:
  - Diagramas de casos de uso
  - Diagramas de clase,
  - Diagramas de secuencia,
  - Diagramas de componentes,
  - Diagramas de despliegue,
  - Diagramas de estado,
  - Diagramas de colaboración

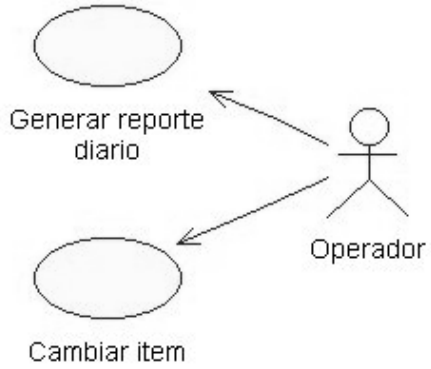
UML: Diagrama de casos de uso

Actor: 

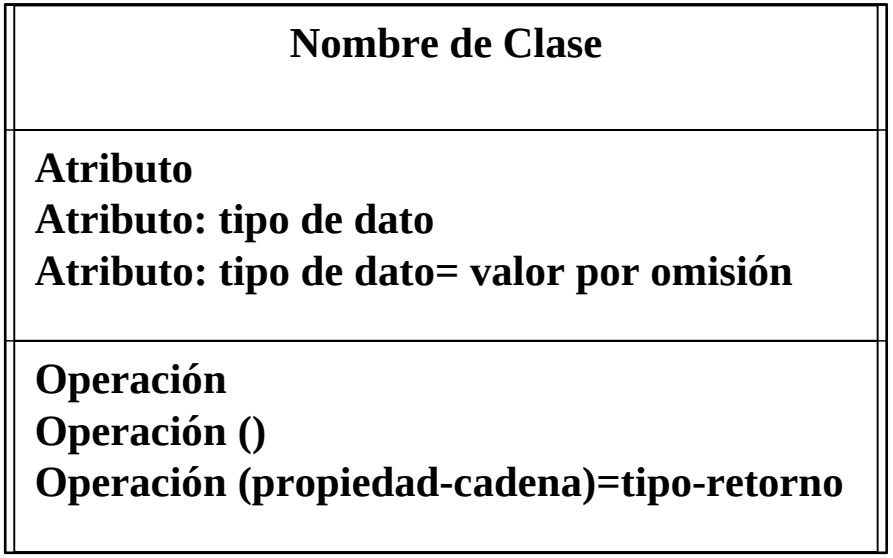
Caso de uso: 

Ejemplo:

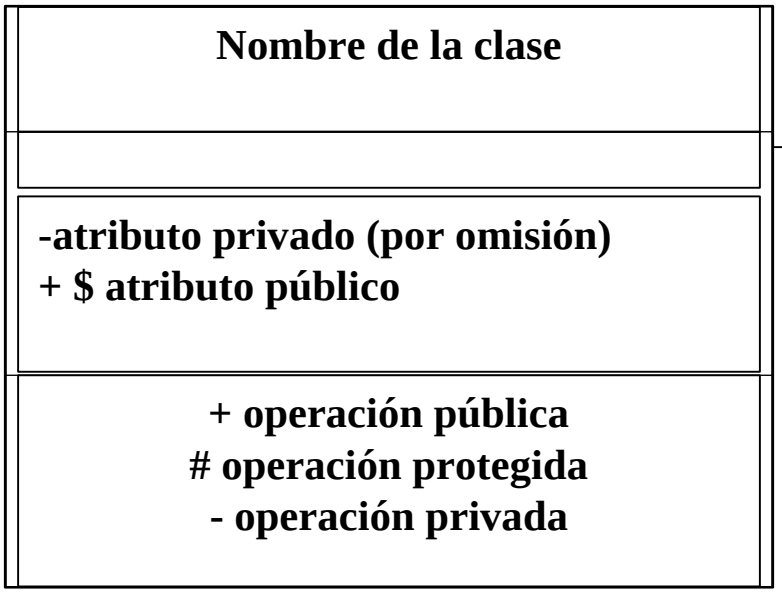




UML: Diagramas de clases



UML: Diagramas de clases



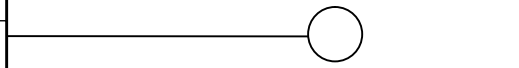
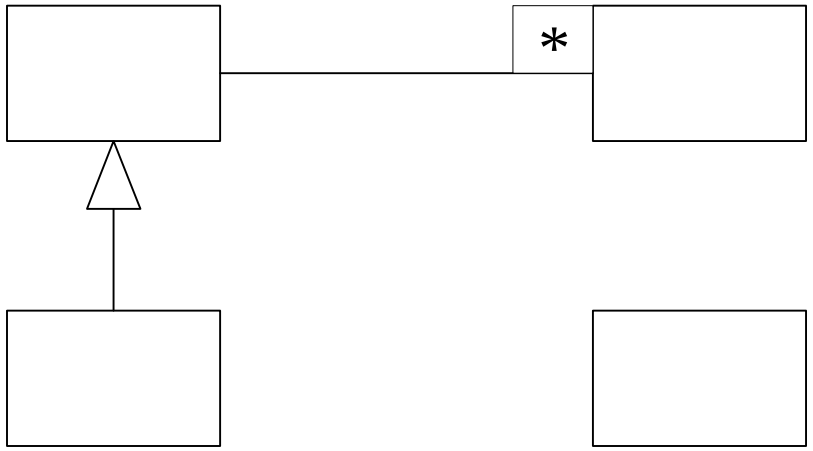


Diagrama de clases



UML: Diagramas de clases

Objeto genérico:

Nombre clase:

Enlaces entre instancias:

Nombre objeto: — **Nombre objeto:**

Objeto específico:

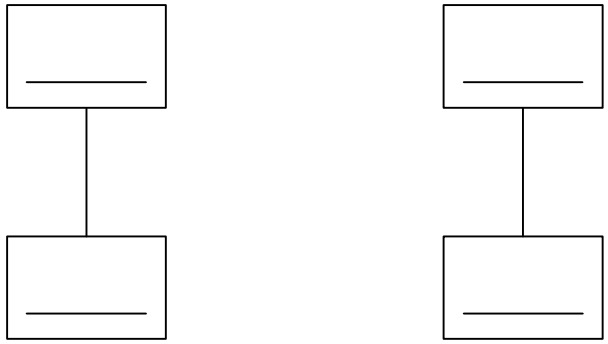
Nombre objeto:

Nombre objeto:Nombre clase

Nombre objeto:Nombre clase

Atributo= valor

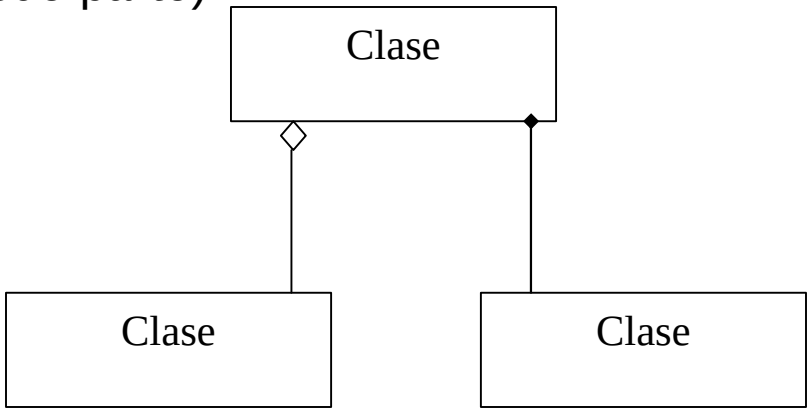
Diagrama de objetos



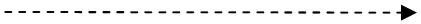
UML: Diagramas de clases

Relaciones:

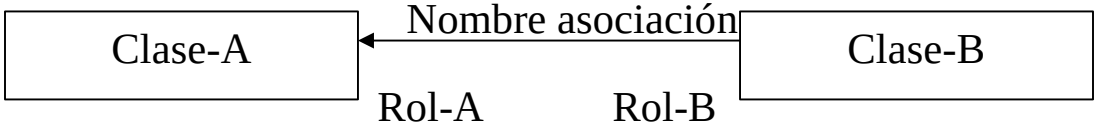
Agregación (Todo parte):



Dependencia:



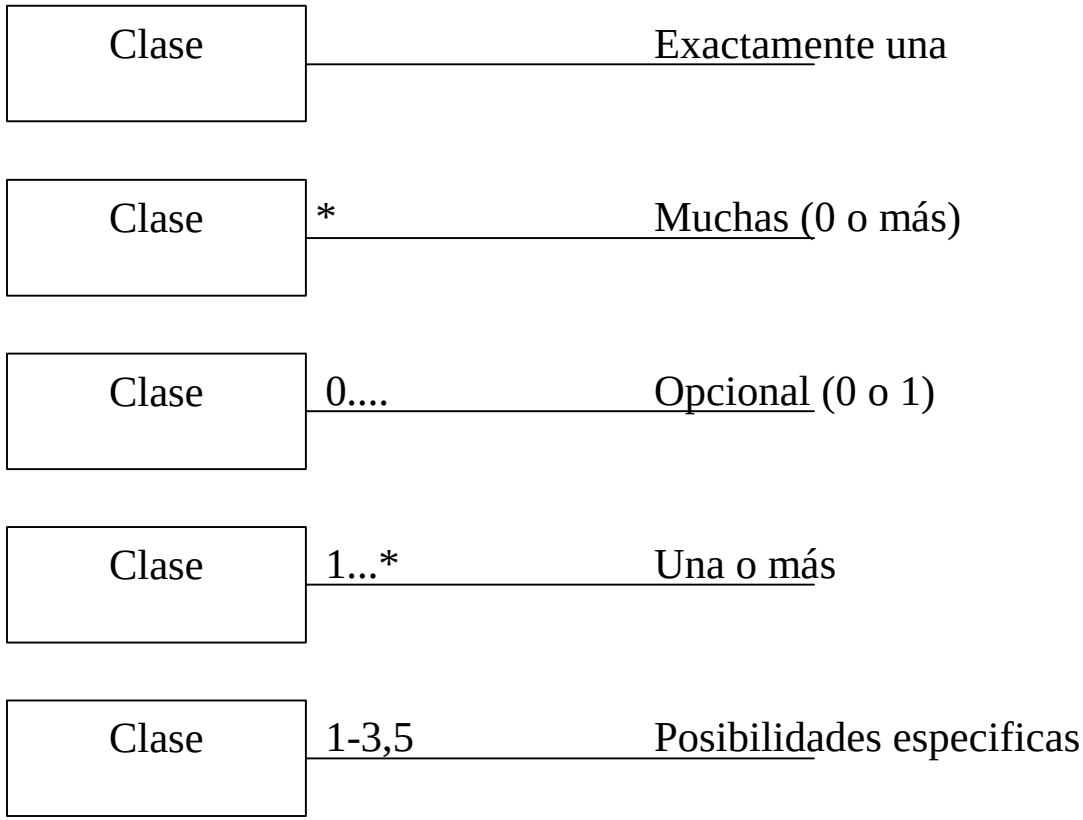
Agregación (Todo parte):



UML: Diagramas de clases

Relaciones:

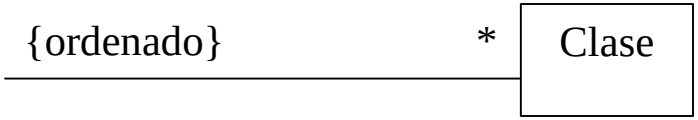
Multiplicidad:



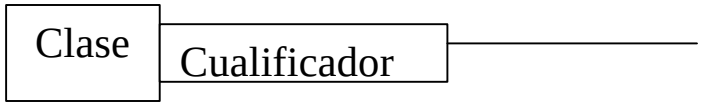
UML: Diagramas de clases

Relaciones:

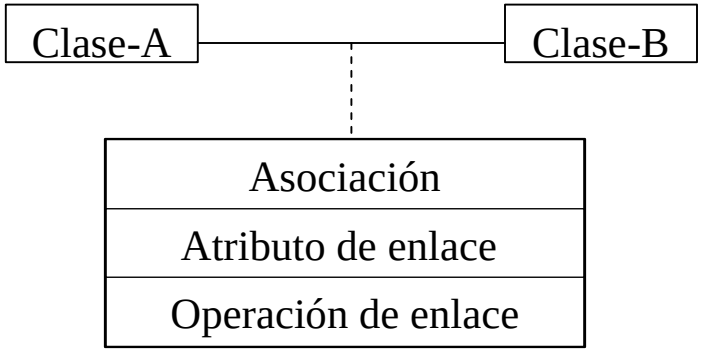
Restricciones:



Asociación cualificada:



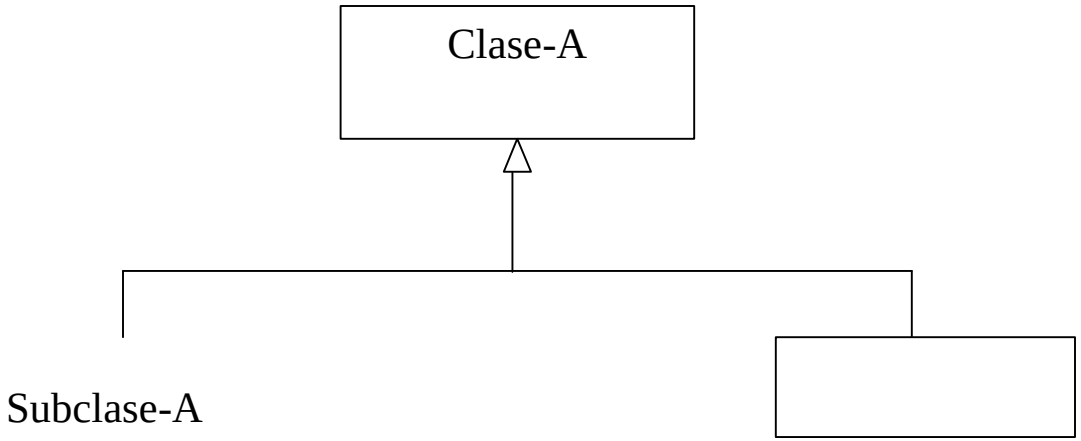
Clase asociación:



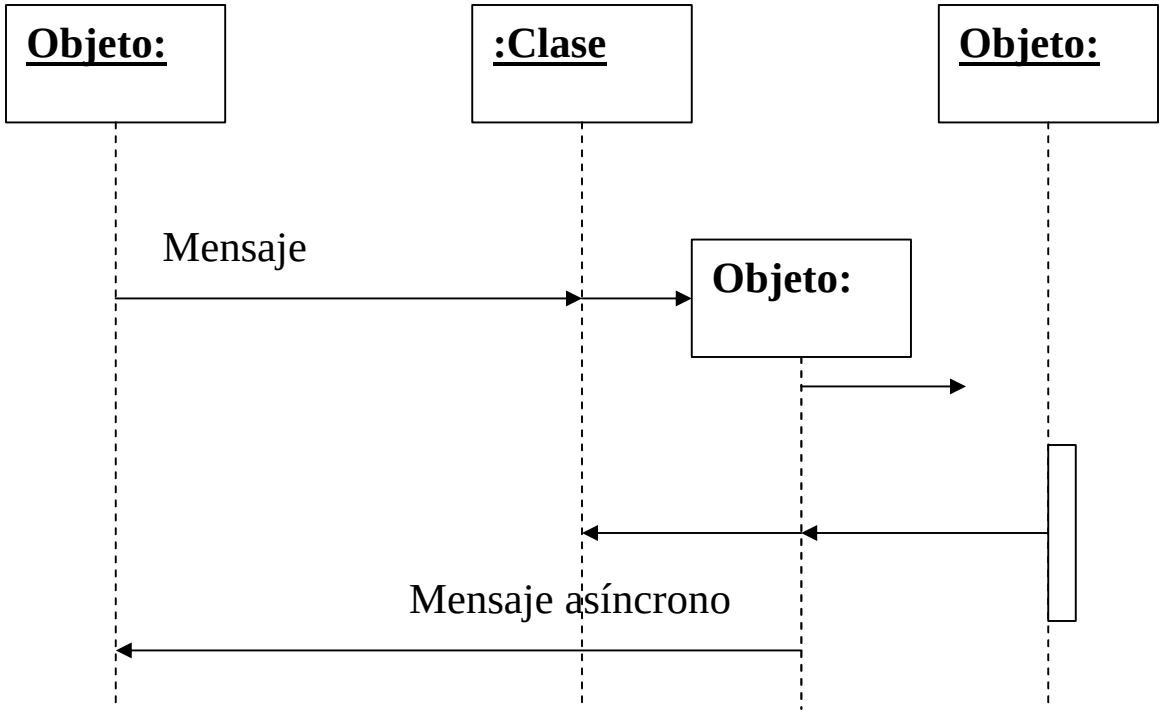
UML: Diagramas de clases

Relaciones:

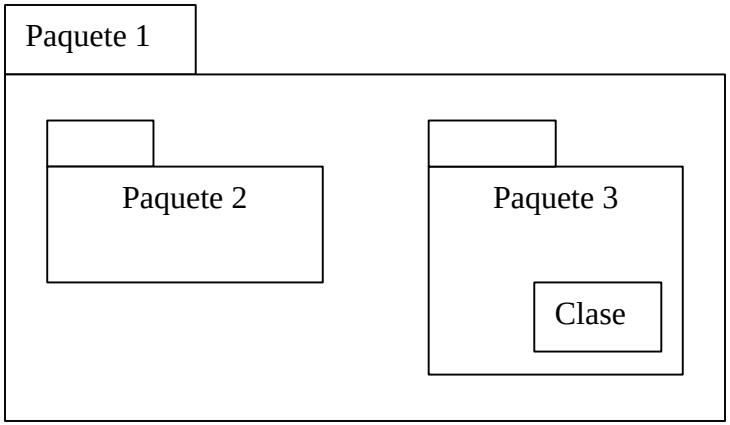
Herencia:

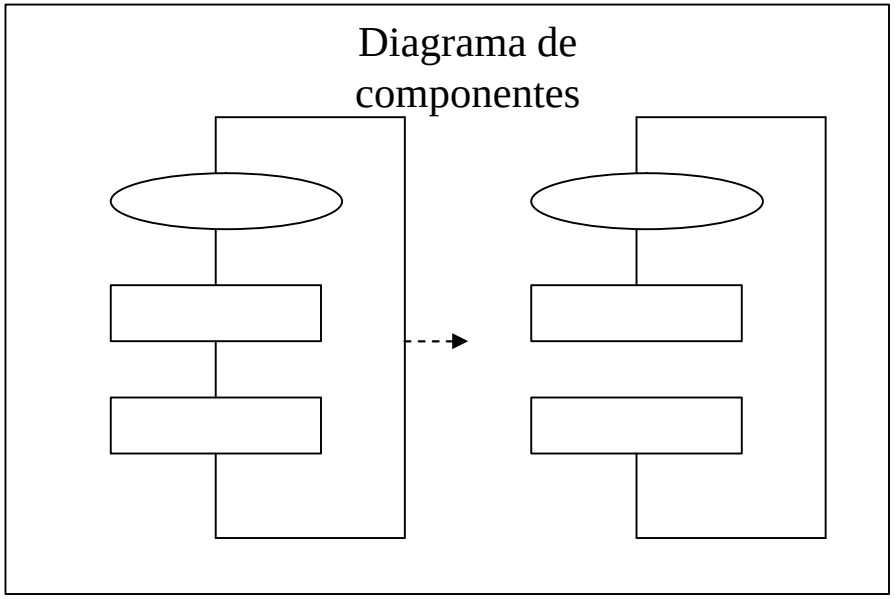


UML: Diagramas de secuencia

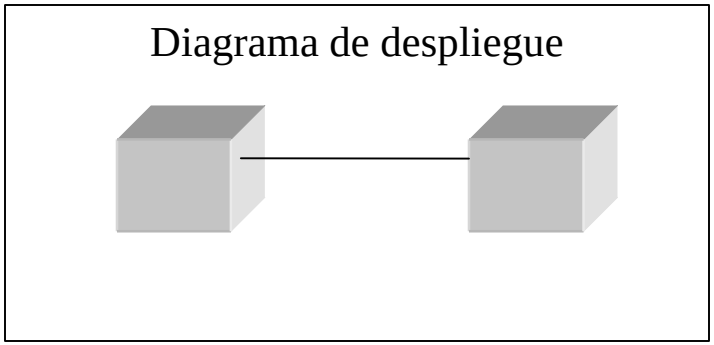


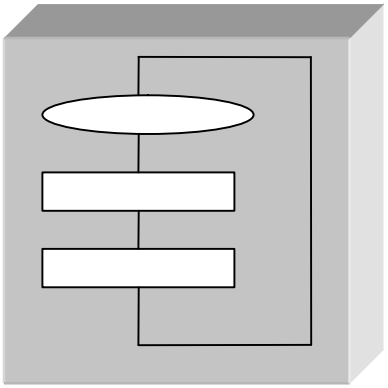
UML: Diagramas de componentes



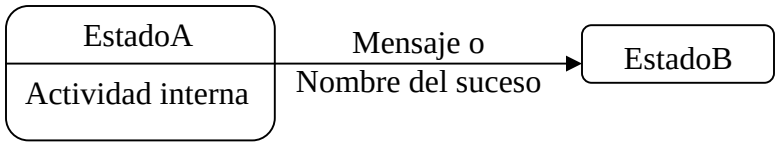


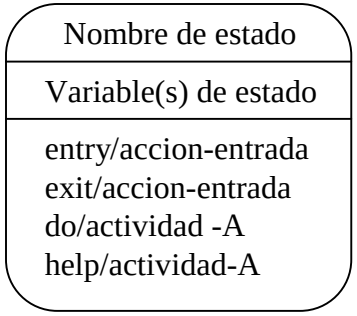
UML: Diagramas de despliegue

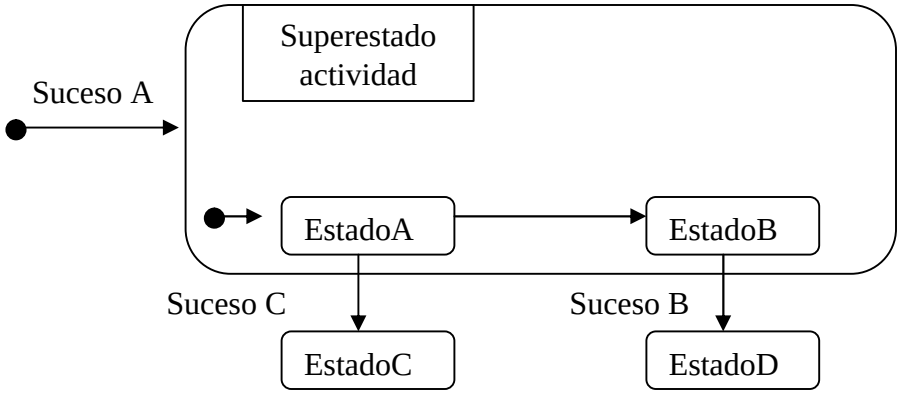




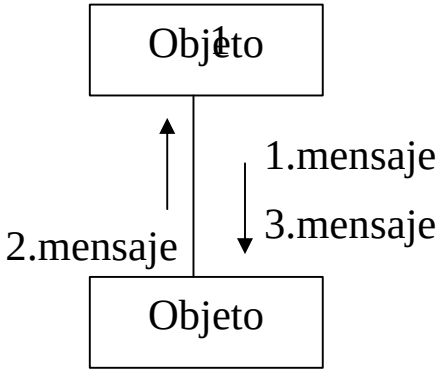
UML: Diagramas de estados

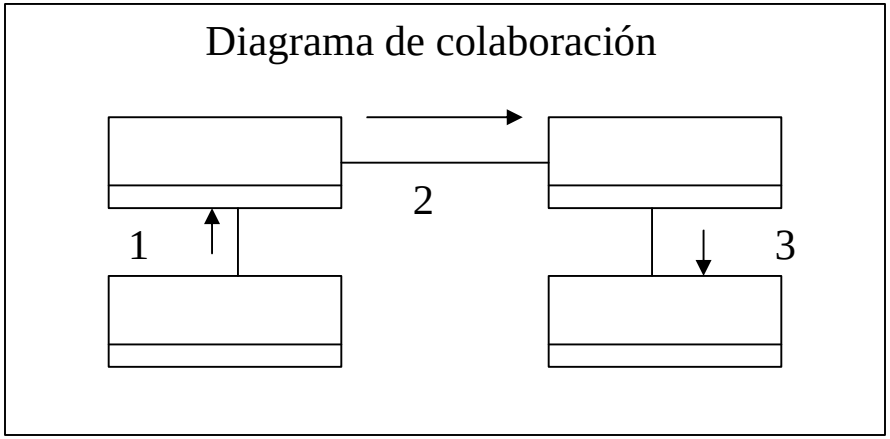






UML:Diagramas de colaboración





Principios PUD

- PUD se basa en los siguientes principios de desarrollo:
  - Desarrollo Iterativo del Software,
  - Administración de Requerimientos,
  - Uso de Arquitecturas Basadas en Componentes,
  - Modelado Visual del Software,
  - Verificación de la Calidad del Software,
  - Control de Cambios.

Principios PUD
Desarrollo Iterativo

- El software moderno es complejo y novedoso, así que no es realista usar un modelo lineal de desarrollo como el de cascada.
- Un proceso iterativo permite una comprensión creciente de los requerimientos a la vez que se va haciendo crecer el sistema.
- PUD sigue un modelo iterativo que aborda las tareas más arriesgadas primero.
- Así se logra reducir los riesgos del proyecto y tener un subsistema ejecutable tempranamente.

Principios PUD
Administración de Requerimientos

- PUD describe cómo:
  - obtener los requerimientos,
  - organizarlos,
  - documentar requerimientos de funcionalidad y restricciones,
  - rastrear y documentar decisiones,
  - captar y comunicar requerimientos del negocio,
  - Detectar más facilmente las incosistencias
- Los casos de uso y los escenarios indicados por el proceso han probado ser una buena forma de captar requerimientos y guiar el diseño, la implementación y las pruebas.

Principios PUD

Arquitecturas Basadas en Componentes

- El proceso se basa en diseñar tempranamente una arquitectura base ejecutable.
- La arquitectura debe ser:
  - flexible,
  - fácil de modificar,
  - intuitivamente comprensible,
  - promueve la reutilización de componentes.
- Apoya el desarrollo basado en componentes, tanto nuevos como preexistentes.
- Componente: un elemento del software con función y limites claros

Principios PUD
Modelado Visual

- Modelado visual de la estructura y el comportamiento de la arquitectura y los componentes.
- Las herramientas de modelado visual permiten:
  - ocultar detalles,
  - la comunicación en el equipo de desarrollo,
  - analizar la consistencia:
    - entre las componentes,
    - entre diseño e implementación.
- UML es la base del modelado visual del PUD.

Principios PUD
Verificación de Calidad

- No sólo la funcionalidad es esencial, también el rendimiento y la confiabilidad.
- PUD ayuda a planificar, diseñar, implementar, ejecutar y evaluar pruebas que verifiquen estas cualidades.
- La prueba es vital para detectar errores antes de la implementación.
- El aseguramiento de la calidad es parte del proceso de desarrollo y no la responsabilidad de un grupo independiente.

Principios PUD

Control de Cambios

- El control es esencial en en desarrollo sotware para evitar caos.
- Los cambios son inevitables, pero es necesario evaluar si éstos son necesarios y investigar su impacto.
- PUD indica como controlar, investigar y monitorizar los cambios dentro del proceso iterativo de desarrollo.

¿Qué es PUD?

- Para definir lo que es el proceso unificado, podríamos resumirlo en tres frases clave:
  - El PUD está dirigido por casos de uso
  - El PUD está centrado en la arquitectura
  - El PUD es iterativo e incremental

- A continuación, pasamos a explicar cada una de estas definiciones...

Dirigido por Casos de Uso

- Podríamos decir que un **Caso de Uso** es un fragmento de funcionalidad del sistema que proporciona al usuario un resultado importante
  - Se utilizan para representar los requisitos funcionales.
  - No sólo inician el proceso de desarrollo, sino que le proporcionan un hilo conductor.
- El proceso avanza a través de una serie de flujos de trabajo que parten de los casos de uso.
- Los casos de uso se especifican, se diseñan, y los casos de uso finales son la fuente a partir de la cual se constituyen los casos de prueba.

Está centrado en la arquitectura

- La arquitectura es una vista del diseño completo con las características más importantes resaltadas, dejando los detalles de lado.
- El valor de la arquitectura depende de las personas que se hayan responsabilizado de su creación

Relación Casos de Uso - Arquitectura

- Cada producto tiene tanto una función como una forma
- Ambas fuerzas deben equilibrarse para lograr un producto de éxito
- La función corresponde a los casos de uso y la forma a la arquitectura
- Los casos de uso y la arquitectura deben evolucionar en paralelo.

EL PUD es iterativo e incremental

- Un producto sw puede conllevar mucho tiempo
- Resulta más práctico dividirlo en pequeñas partes o miniproyectos
- Cada miniproyecto es una iteración que acaba en un incremento
- Las iteraciones hacen referencia a pasos en el flujo de trabajo y los incrementos al crecimiento del producto

EL PUD es iterativo e incremental

- La iteración trata un grupo de casos de uso que juntos amplían la utilidad del producto desarrollado hasta ahora e identifica los riesgos más importantes
- En cada iteración, se identifican y especifican los casos de uso más relevantes, se crea un diseño utilizando la arquitectura como guía, se implementa el diseño como componentes y se verifica que dichos componentes satisfacen los casos de uso

La vida del Proceso Unificado

- Se repite a lo largo de una serie de ciclos que constituyen la vida de un sistema. Cada ciclo concluye con una versión del producto para los clientes.
- Cada ciclo consta de cuatro fases: Inicio, elaboración, construcción y transición. Como vimos anteriormente, cada fase se subdivide en iteraciones.

Conceptos Importantes

- Artefacto: Pieza de información tangible que:
  - Es creada, modificada y usada por los trabajadores al realizar las actividades
  - Representa un área de responsabilidad
  - Es candidata a ser tenida en cuenta para el control de la configuración.
- Un artefacto puede ser un modelo, un elemento de un modelo o un documento.

Conceptos Importantes

- Actividad: Unidad tangible de trabajo realizada por un trabajador en un flujo de trabajo de forma que:
  - Implica una responsabilidad bien definida para el trabajador
  - Produce un resultado bien definido basado en una entrada bien definida
  - Representa una unidad de trabajo con límites bien definidos

Conceptos Importantes

- Modelo: Es una abstracción del sistema, especificando el sistema modelado desde un cierto punto de vista y en un determinado nivel de abstracción
  - Son abstracciones del sistema que construyen los arquitectos y desarrolladores
- Un modelo siempre identifica el sistema que está modelando

El producto

- Cada ciclo produce una nueva versión del sistema, y cada versión es un producto preparado para su entrega.
- Consta de un cuerpo de código fuente incluido en componentes que puede compilarse y ejecutarse, además de manuales y otros productos asociados.

Las personas

- El resultado final de un proyecto software es un producto que toma forma durante su desarrollo gracias a la intervención de muchos tipos distintos de personas.
- Los principales autores de un proyecto software son los arquitectos, desarrolladores, ingenieros de prueba y el personal de gestión que les da soporte, además de clientes, usuarios y otros interesados.

Trabajadores

- La gente llega a ocupar muchos puestos diferentes en una organización de desarrollo de software
- Se conoce como **trabajador** a los puestos a los cuales se pueden asignar personas y los cuales esas personas aceptan
- Un trabajador puede representar a un conjunto de personas que trabajan juntas, p. ej. Grupo de arquitectura - arquitecto.

Fases dentro de un ciclo

- Cada ciclo se desarrolla a lo largo del tiempo
- Este tiempo se subdivide a su vez en cuatro fases
- A través de una secuencia de modelos, los implicados visualizan lo que está sucediendo en esas fases
- Dentro de cada fase, los directores o los desarrolladores pueden descomponer adicionalmente el trabajo

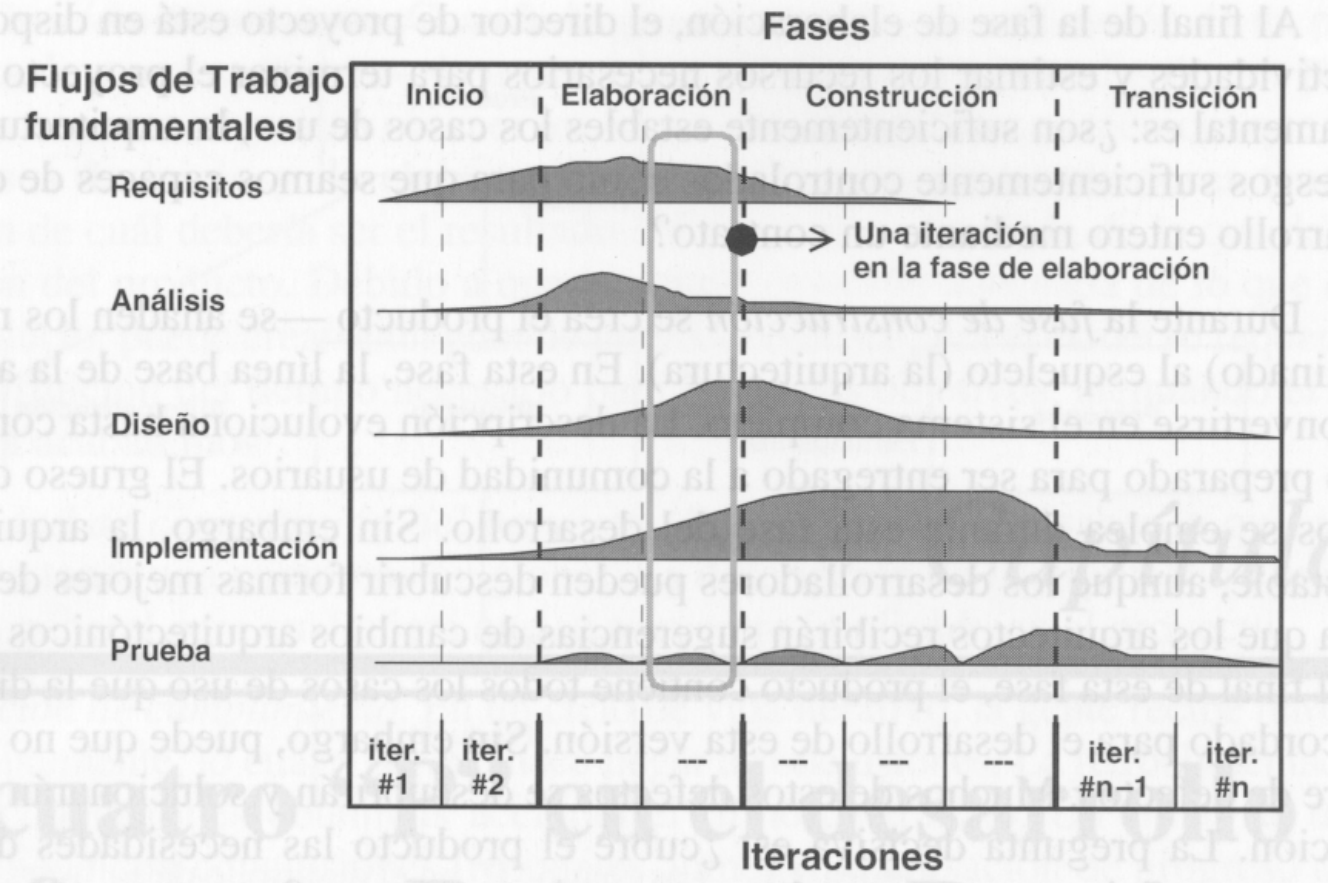
Fases dentro de un ciclo (Hitos)

- Cada fase termina con un hito
- Cada hito se determina por la disponibilidad de un conjunto de artefactos
- Uno de los objetivos más importantes de los hitos es que los directores deben tomar ciertas decisiones cruciales antes de pasar a la siguiente fase

Fases dentro de un ciclo

- Al final se obtiene un conjunto de datos a partir del seguimiento del tiempo y esfuerzo consumido en cada fase.
- Estos datos son útiles para la estimación del tiempo y de los recursos humanos para otros proyectos, la asignación de recursos mientras dura el proyecto y para controlar el progreso contrastado con las planificaciones.

EL proceso


Fases
Flujos de Trabajo fundamentales
Inicio
Elaboración
Construcción
Transición
Requisitos
Una iteración en la fase de elaboración
Análisis
Diseño
Implementación
Prueba
iter. #1
iter. #2

---
iter. #n–1
iter. #n
Iteraciones

El proceso

- La figura muestra en la columna de la izquierda los flujos de trabajo (requisitos, análisis, diseño, implementación y pruebas).
- Las curvas son una aproximación de hasta donde se llevan a cabo los flujos de trabajo en cada fase (Recuérdese que cada fase se divide normalmente en iteraciones)
- Una iteración típica pasa por los cinco flujos de trabajo

Flujos de Trabajo

- Hay cinco flujos de trabajo:
  - Captura de Requisitos
  - Análisis
  - Diseño
  - Implementación
  - Prueba

Captura de Requisitos

- El esfuerzo principal consiste en desarrollar un modelo del sistema que se va a construir
  - La utilización de casos de uso es una forma apropiada de hacerlo
- Los artefactos fundamentales en este flujo son el modelo de casos de uso, prototipos de interfaz de usuario...

Captura de Requisitos

- Los trabajadores que actúan son:
  - Analista de sistemas: Responsable del conjunto de requisitos modelados en los casos de uso y de delimitar el sistema
  - Especificador de casos de uso: Ayudan a los analistas de sistemas en el trabajo de capturar los requisitos. Deben trabajar estrechamente con los usuarios de los casos de uso

Captura de Requisitos

- Diseñador de Interfaz de Usuario: Dan forma visual a las interfaces de usuario
- Arquitecto: Describe la vista de la arquitectura del modelo de casos de uso. Esta vista es una entrada importante para planificar las iteraciones.

Captura de Requisitos (Flujo de Trabajo)

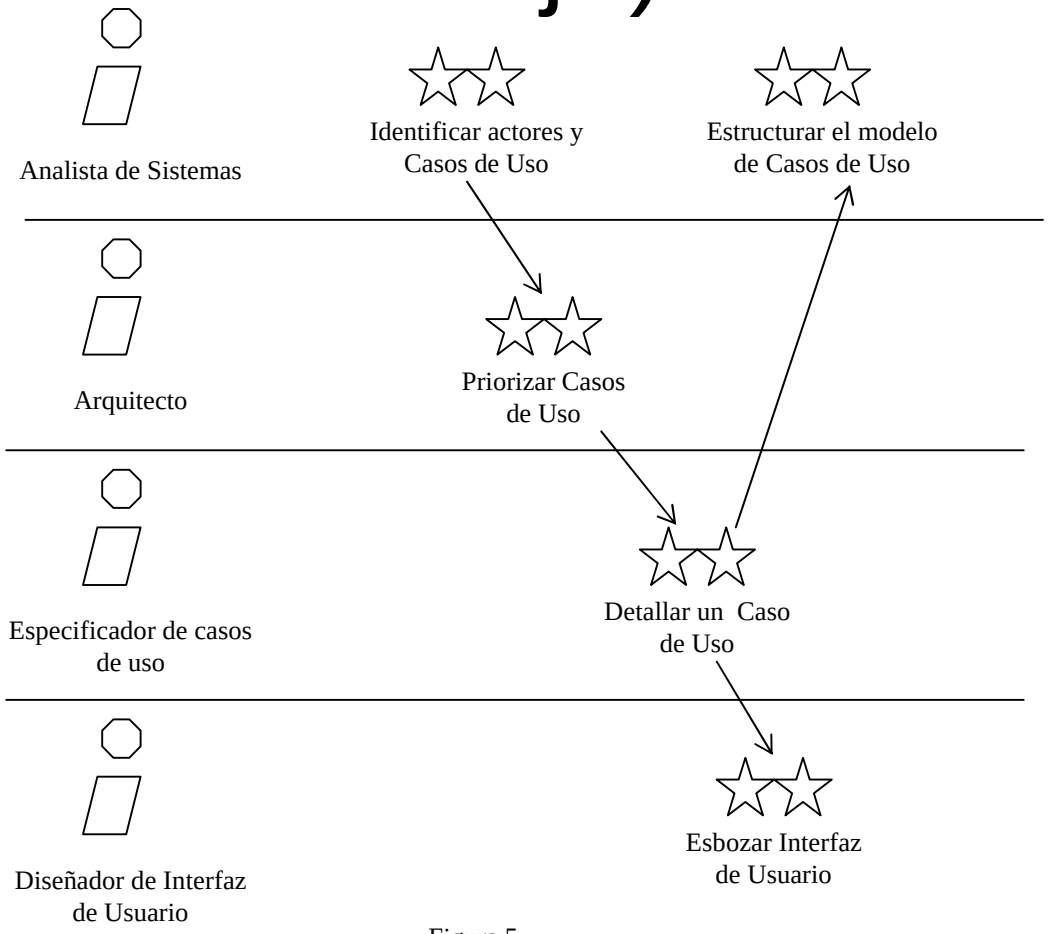


Figura 5

Captura de Requisitos (Flujo de Trabajo)

- EL diagrama utiliza calles para mostrar qué trabajadores ejecutan qué actividades
- Cada actividad se sitúa en el mismo campo que el trabajador que las ejecuta
- Los flujos de trabajo son una secuencia de actividades ordenada
  - Una actividad produce una salida que sirve como entrada de la siguiente actividad

Análisis

- Se analizan los requisitos que se describieron en la captura de requisitos, refinándolos y estructurándolos
- Para ello se utiliza el **modelo de análisis**, que ayuda a refinar los requisitos y permite razonar sobre los aspectos internos del sistema.

Análisis

- Los artefactos fundamentales en este flujo son el modelo de análisis, las clases del análisis, la realización del caso de uso-análisis, el paquete del análisis y la descripción de la arquitectura

Análisis

- Los trabajadores que actúan son:
  - Arquitecto: Responsable de la integridad del modelo de análisis, garantizando que éste sea correcto, consistente y legible como un todo. También es responsable de la arquitectura del modelo de análisis
  - Ingeniero de casos de uso: Responsable de la integridad de una o más realizaciones de casos de uso

Análisis

- Ingeniero de componentes: Define y mantiene las relaciones, atributos, relaciones y requisitos especiales de una o varias clases del análisis

Análisis (Flujo de Trabajo)

- Los arquitectos comienzan la creación del modelo de análisis identificando los paquetes del análisis principales, las clases de entidad evidentes y los requisitos comunes.
- Los ingenieros de casos de uso realizan los casos de uso en términos de las clases del análisis participantes.

Análisis (Flujo de Trabajo)

- Posteriormente, los ingenieros de componentes especifican estos requisitos y los integran dentro de cada clase, creando responsabilidades, atributos y relaciones consistentes para cada clase

Diseño

- Se modela el sistema y se encuentra su forma para que soporte todos los requisitos que se le suponen.
- Una entrada esencial en el diseño es la salida del análisis.
- El diseño es el centro de atención al final de la fase de elaboración y el comienzo de las iteraciones de construcción.

Diseño

- Los artefactos fundamentales en este flujo son el modelo de diseño, la clase del diseño, la realización de casos de uso-diseño, el subsistema de diseño, la interfaz, la descripción de la arquitectura y el modelo de despliegue

Diseño

- Los trabajadores que actúan son:
  - Arquitecto: Responsable de la integridad de los modelos de diseño y de despliegue, garantizando que son correctos, consistentes y legibles en su totalidad.
  - Ingeniero de casos de uso: Responsable de la integridad de una o más realizaciones de casos de uso-diseño

Diseño

- Ingeniero de componentes: Define y mantiene las operaciones, métodos, atributos, relaciones y requisitos de implementación de una o más clases del diseño.

Diseño (Flujo de Trabajo)

- Los arquitectos inician la creación de los modelos de diseño y despliegue
- Después, los ingenieros de casos de uso realizan cada caso de uso en términos de clases y subsistemas del diseño de participantes y sus interfaces
- Los ingenieros de componentes especifican a continuación los requisitos y los integran dentro de cada clase,

Diseño (Flujo de Trabajo)

- A lo largo del diseño, los desarrolladores identificarán nuevos candidatos para ser subsistemas, interfaces, clases y mecanismos de diseño genéricos, y los ingenieros de componentes responsables de los subsistemas individuales deberán refinarlos y mantenerlos.

Implementación

- En la implementación se empieza con el resultado del diseño y se implementa el sistema en términos de componentes, es decir, ficheros de código fuente, scripts, ficheros de código binario, ejecutables y similares.

Implementación

- Es el centro durante las iteraciones de construcción, aunque también se lleva a cabo trabajo de implementación durante la fase de elaboración, para crear la línea base ejecutable de la arquitectura, y durante la fase de transición, para tratar defectos tardíos como los encontrados con distribuciones beta del sistema

Implementación

- Los artefactos fundamentales en este flujo son el modelo de implementación, los componentes, el subsistema de implementación, la descripción de la arquitectura y el plan de integración de construcciones

Implementación

- Los trabajadores que actúan son:
  - Arquitecto: Responsable de la integridad del modelo de implementación y asegura que el modelo como un todo es correcto, completo y legible.
  - Ingeniero de componentes: define y mantiene el código fuente de uno o varios componentes, garantizando que cada componente implementa la funcionalidad correcta

Implementación

- Integrador de sistemas: Entre sus responsabilidades se incluye el planificar la secuencia de construcciones necesarias en cada iteración y la integración de cada construcción cuando sus partes han sido implementadas. La planificación da lugar a un plan de integración de construcciones.

Implementación (Flujo de Trabajo)

- El objetivo principal de la implementación es implementar el sistema.
- Este proceso es iniciado por el arquitecto esbozando los componentes clave en el modelo de implementación
- A continuación, el integrador de sistemas planea las integraciones de sistema necesarias en la presente iteración como una secuencia de instrucciones

Implementación (Flujo de Trabajo)

- Para cada construcción, el integrador de sistemas describe la funcionalidad que debería ser implementada y qué partes del modelo de implementación se verán afectadas
- Los requisitos sobre los subsistemas y componentes en la construcción son entonces implementados por ingenieros de componentes.

Implementación (Flujo de Trabajo)

- Los componentes resultantes son probados y pasados al integrador de sistemas para su integración
- Entonces, el integrador de sistemas integra los nuevos componentes en una nueva construcción y la pasa a los ingenieros de pruebas de integración para llevar a cabo pruebas de integración.

Implementación (Flujo de Trabajo)

- Por último, los desarrolladores inician la implementación de la siguiente construcción, tomando en consideración los defectos de la construcción anterior.

Prueba

- En el flujo de trabajo de la prueba se verifica el resultado de la implementación probando cada construcción, incluyendo tanto construcciones internas como intermedias, así como las versiones finales del sistema a ser entregadas a terceros.
- Las pruebas se llevan a cabo sobre todo cuando una construcción es sometida a pruebas de integración y de sistema.

Prueba

- Los artefactos fundamentales en este flujo son el modelo de pruebas, el caso de pruebas, el procedimiento de prueba, el componente de prueba, el plan de prueba, el defecto y la evaluación de pruebas

Prueba

- Los trabajadores que actúan son:
  - Diseñador de pruebas: es responsable de la integridad del modelo de pruebas, asegurando que el modelo cumple con su propósito. Los diseñadores de pruebas también planean las pruebas, lo que significa que deciden los objetivos de prueba apropiados y la planificación de las pruebas.

Prueba

- Ingeniero de componentes: son responsables de los componentes de prueba que automatizan algunos de los procedimientos de prueba.
- Ingeniero de pruebas de integración: son los responsables de realizar las pruebas de integración que se necesitan para cada construcción producida en el flujo de trabajo de la implementación. Las pruebas de integración se realizan para verificar que los componentes integrados en una construcción funcionan correctamente juntos.

Prueba

– Ingeniero de pruebas de sistema: Es responsable de realizar las pruebas de sistema necesarias sobre una construcción que muestra el resultado de una iteración completa. Las pruebas de sistema se llevan a cabo principalmente para verificar las interacciones entre los actores y el sistema.

Prueba (Flujo de Trabajo)

- El principal objetivo de la prueba es realizar y evaluar las pruebas como se describe en el modelo de pruebas.
- Los ingenieros de pruebas inician esta tarea planificando el esfuerzo de prueba en cada iteración, y describen entonces los casos de prueba necesarios y los procedimientos de prueba correspondientes para llevar a cabo las pruebas

Prueba (Flujo de Trabajo)

- Si es posible, los ingenieros de componentes crean a continuación los componentes de prueba para automatizar algunos de los procedimientos de prueba.
- Con estos casos, procedimientos y componentes de prueba como entrada, los ingenieros de pruebas de integración y de sistema prueban cada construcción y detectan cualquier defecto que encuentren

Prueba (Flujo de Trabajo)

- Los defectos sirven como realimentación tanto para otros flujos de trabajo como el de diseño y el de implementación, como para los ingenieros de pruebas para que lleven a cabo una evaluación sistemática de los resultados de las pruebas.

Fase de Inicio

- El objetivo de la fase de inicio es desarrollar el análisis de negocio hasta el punto necesario para justificar la puesta en marcha del proyecto.
- Lo que se intenta es hacer una idea de la arquitectura para asegurarse de que existe una arquitectura que puede soportar el ámbito del sistema.

Fase de Inicio

- Ejecución de los flujos de trabajo fundamentales:
  - Recopilación de requisitos: Este flujo de trabajo incluye identificar y detallar los casos de uso pertinentes en cada fase
  - Análisis: Los objetivos generales del flujo de trabajo de análisis son analizar los requisitos, refinarlos y estructurarlos en un modelo de objetos que sirva como primera impresión del modelo de diseño. En esta fase, el resultado es un modelo inicial de análisis.

Fase de Inicio

- Diseño: el objetivo principal del flujo de trabajo de diseño es esbozar un modelo de diseño de la arquitectura candidata, con el objeto de incluirlo en la descripción de la arquitectura preliminar.
- Implementación: La actividad en este flujo de trabajo depende de decisiones que el jefe de proyecto haya tomado anteriormente. En situaciones normales, se finalizará esta fase con la descripción de la arquitectura candidata, en cuyo caso, seguir con el flujo de trabajo de implementación no es necesario.

Fase de Inicio

– Prueba: En paralelo con las actividades de análisis, diseño e implementación, los ingenieros de pruebas se van poniendo al corriente de la naturaleza general del sistema propuesto, van considerando qué pruebas requerirá y van desarrollando algunos planes provisionales de pruebas

Productos de la fase de Inicio

- Una lista de características
- Una primera versión del modelo de negocio (o del dominio) que describe el contexto del sistema
- Un esbozo de los modelos que representan una primera versión del modelo de casos de uso, el modelo de análisis y el modelo de diseño.
- Un primer esquema de la descripción de una arquitectura candidata, que perfila las vistas de los modelos de casos de uso, análisis, diseño e implementación.

Productos de la fase de Inicio

- Posiblemente, un prototipo exploratorio que muestra el uso del nuevo sistema
- Una lista inicial de riesgos y una clasificación de casos de uso
- Los rudimentos de un plan para el proyecto en su totalidad, incluyendo el plan general de las fases
- Un primer borrador del análisis de negocio, que incluye: Contexto del negocio y criterios de éxito.

Fase de Elaboración

- Al comienzo de la fase de elaboración, se recibe de la fase de inicio un plan para la fase de elaboración, un modelo de casos de uso parcialmente completo y una descripción de la arquitectura candidata.
- También se dispone de un prototipo que muestre el funcionamiento del sistema.
- Sin embargo, no se puede pretender construir sobre ese prototipo

Fase de Elaboración

- Ejecución de los flujos de trabajo fundamentales:
  - Recopilación de requisitos: En este punto encontraremos, estableceremos la prioridad y estructuraremos los casos de uso.
  - Análisis: Durante la fase de inicio, comenzamos a hacer un borrador del modelo de análisis. Ahora, construiremos sobre este borrador, pero podemos descubrir que es necesario desechar partes sustanciales de él.

Fase de Elaboración

- Diseño: El arquitecto es responsable del diseño de los aspectos arquitectónicamente significativos del sistema, tal como están descritos en la vista de la arquitectura del modelo de diseño. La vista de la arquitectura del modelo de diseño incluye subsistemas, clases, interfaces y realizaciones de casos de uso arquitectónicamente significativos, incluidos éstos en la vista del modelo de casos de uso

Fase de Elaboración

- Implementación: Este flujo de trabajo y prueba los componentes arquitectónicamente significativos a partir de los elementos de diseño arquitectónicamente significativos. El resultado es la línea base de la arquitectura, implementada normalmente a partir de menos del 10 por ciento de los casos de uso.
- Prueba: Aquí el objetivo es asegurarse de que los subsistemas de todos los niveles y de todas las capas funcionen. Por supuesto, sólo podemos probar los componentes ejecutables.

Productos de la Fase de Elaboración

- Preferiblemente un modelo completo de negocio que describe el contexto del sistema
- Una nueva versión de todos los modelos: casos de uso, análisis, diseño, despliegue e implementación.
- Una línea base de la arquitectura
- Una descripción de la arquitectura, incluyendo vistas de los modelos de casos de uso, análisis, diseño, despliegue e implementación
- Una lista de riesgos actualizada

Productos de la Fase de Elaboración

- El plan del proyecto para las fases de construcción y transición
- Un manual de usuario preliminar
- El análisis de negocio completo, incluida la apuesta económica

Fase de Construcción

- El propósito primordial de esta fase es dejar listo un producto software en su versión operativa inicial, a veces llamada “versión beta”.
- El producto debería tener la calidad adecuada para su aplicación y asegurarse de cumplir los requisitos.
- La construcción debería tener lugar dentro de los límites del plan de negocio.

Fase de Construcción

- Recordemos que para cumplir los objetivos de la fase de elaboración, se han recopilado casi todos los requisitos, pero aún no han sido detallados completamente. Esto es lo que se hará en la fase de construcción.

Fase de Construcción

- Ejecución de los flujos de trabajo fundamentales:
  - Captura de requisitos: en la fase de construcción se recorrerá todo el camino hasta lograr el sistema inicial con capacidad operativa, así que hay que realizar la recopilación completa de requisitos
  - Análisis: Aquí consideraremos de nuevo las actividades de análisis de la arquitectura, analizar un caso de uso, analizar una clase y analizar un paquete, iniciadas en la fase de elaboración.

Fase de Construcción

- Diseño: Normalmente, en esta fase se diseña e implementa el 90 por ciento restante de los casos de uso (aquellos que no fueron utilizados para desarrollar la línea base de la arquitectura).
- Implementación: Implementa y lleva a cabo las pruebas de unidad de todos los componentes, trabajando principalmente a partir del modelo de diseño. El resultado es la versión operativa inicial, que representa el 100 por 100 de los casos de uso. El proyecto lleva a cabo la mayor parte del trabajo de la fase de construcción, construyendo los componentes.

Fase de Construcción

– Prueba: Los esfuerzos de los ingenieros de pruebas para descubrir lo que puede ser comprobado de forma efectiva y para desarrollar casos de prueba y procedimientos de prueba para hacerlo, tendrán su fruto en la fase de construcción. Ésta es una actividad fundamental de esta fase

Productos de la fase de Construcción

- El plan de proyecto para la fase de transición
- El sistema software ejecutable. Ésta es la construcción final de la fase
- Todos los artefactos, incluyendo los modelos del sistema
- La descripción de la arquitectura, mínimamente modificada y actualizada
- Una versión preliminar del manual de usuario, lo suficientemente detallado como para guiar a los usuarios de la beta

Fase de Transición

- Una vez que el proyecto entra en la fase de transición, el sistema ha alcanzado la capacidad operativa inicial.
- El jefe de proyecto ha considerado que el sistema ofrece la confianza suficiente como para operar en el entorno del usuario, aunque no sea necesariamente perfecto

Fase de Transición

- El usuario puede descubrir con retraso la necesidad de determinadas características.
- Si son muy importantes y casan bien con el producto existente, el jefe de proyecto puede aceptar añadirlas. Sin embargo, los cambios deben ser lo suficientemente pequeños como para que puedan ser introducidos sin afectar seriamente el plan del proyecto.

Fase de Transición

- Si una característica propuesta afecta a la planificación, su necesidad debe ser aguda. En la mayoría de los casos, consideramos que es mejor añadirla a la lista de características y dejarla para el siguiente ciclo de desarrollo, es decir, para el desarrollo de la siguiente versión del sistema.

Fase de Transición

- Ejecución de los flujos de trabajo fundamentales:
  - En esta fase, la actividad es baja en los cinco flujos de trabajo.
  - Como casi todo el trabajo se realizó en la fase de construcción, el nivel de actividad en esta fase es bajo, justo lo necesario para corregir los problemas encontrados durante las pruebas en el entorno del usuario.

Fase de Transición

– No debería haber apenas trabajo en los flujos de trabajo de requisitos, análisis y diseño. Normalmente, las actividades de diseño disminuyen durante la fase de transición. La atención se desplaza a la corrección de defectos para eliminar los fallos que ocurran durante el uso inicial, a asegurarse de que las correcciones en sí están bien, y a hacer pruebas de regresión para asegurar que estas modificaciones no introduzcan nuevos defectos.

Productos de la fase de Transición

- El propio software ejecutable, incluyendo el software de instalación
- Documentos legales como contratos, licencias, renuncias de derechos y garantías
- La versión completa y corregida de línea base de la versión del producto, incluyendo todos los modelos del sistema
- La descripción completa y actualizada de la arquitectura
- Manuales y material de formación del usuario final, del operador y del administrador del sistema

Productos de la fase de Transición

- Referencias para la ayuda del cliente, acerca de dónde encontrar más información, cómo informar de defectos o dónde encontrar información sobre defectos y actualizaciones.

Evaluación de la metodología (Situación del problema)

- Los clientes son empresas u organizaciones de mayor o menor envergadura que se ven en algún momento necesitados de algún software que cubra o mejore alguno de los aspectos de su funcionamiento interno.
- El punto de partida a la hora de capturar los requisitos depende del cliente, tipo de sistema y organización con la que el analista se enfrenta.

Evaluación de la metodología (Situación del problema)

- Los clientes pueden ser de varios tipos desde el punto de vista de la información que puedan proporcionar:
  - Clientes que proporcionan al analista una lista detallada de requisitos propuestos junto con otros detalles que especifican en el momento de la entrevista
  - Clientes que solo proporcionan una vaga idea muy poco detallada de lo que en ultima instancia quieren.

Evaluación de la metodología (Situación del problema)

- El nivel de compromiso que debe tener el cliente hacia el proyecto debe ser máximo
- Tras la obtención de una especificación de requisitos completa, éstos deben ser contrastados, negociados y repasados con el cliente para eliminar detalles insignificantes y quedarse con las verdaderas características y aspectos del problema.

Evaluación de la metodología (Situación del Resolvedor)

- El conjunto de valores del resolvedor debe ser amplio, pero debe ocupar un lugar de privilegio en ellos la capacidad parar trabajar en grupo y el compañerismo
  - Esto es debido a que intervienen una gran cantidad de personas en el desarrollo
- El resolvedor debe poseer conocimientos técnicos y abstractos:

Evaluación de la metodología (Situación del Resolvedor)

- Los abstractos deben ser, entre otros, las características esenciales de las entidades que las distinguen de cualquier otra clase de entidades
- Los técnicos deberían ser todo lo relacionado con UML, además de los flujos de trabajo y fases en las que se desarrolla el Proceso Unificado.

Evaluación de la metodología (Situación del Resolvedor)

- La metodología presenta una gran variedad de modelos, entre los que podemos distinguir el modelo de análisis, modelo de casos de uso, modelo de despliegue, modelo de diseño, modelo de implementación, modelo de negocio, modelo del dominio y modelo en cascada.

Evaluación de la metodología (Situación del Resolvedor)

- El nivel de experiencia que debe poseer el usuario de la metodología depende, en gran parte, del trabajador.
- En esta metodología, no se habla de roles, sino de trabajadores. Es por ello que una persona puede desempeñar varios trabajadores a lo largo del proceso, por lo que ningún integrante tiene asignado un rol específico.

Evaluación de la metodología

- La metodología proporciona la posibilidad de ir formando el contorno poco a poco en cada iteración
- La experiencia de los usuarios es imprescindible, ya que en ella se basa la pericia que tenga el usuario para poder diseñar las soluciones ahorrando tiempo.

Bibliografía

- “El proceso unificado de desarrollo de software”. Ivar Jacobson, Grady Booch y James Rumbaugh. Ed. Addison Wesley
- “El lenguaje Unificado de Modelado” Ivar Jacobson, Grady Booch y James Rumbaugh. Ed. Addison Wesley
- “Modelado y diseño orientado a objetos”. James Rumbaugh y otros. Ed. Prentice-Hall

Bibliografía

- www.rational.com
- http://fdomingu.escet.urjc.es/docencia/IS
- www.dcc.uchile.cl/~luguerre/cc51h/clase23.htm
- www.vico.org
- http://bogart.sip.ucm.es/~pablo/ingen_software