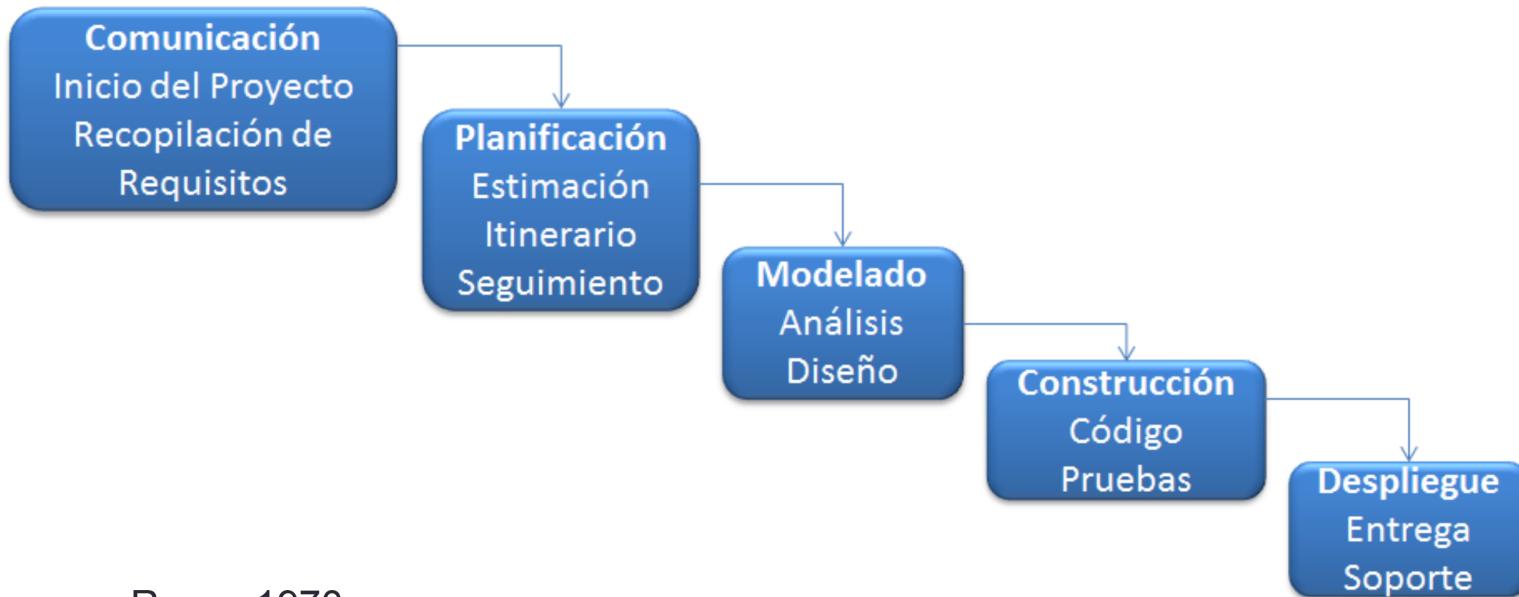


EXAMEN DE TÍTULO INGENIERÍA DE SOFTWARE

Nicolas.rosso@mail_udp.cl

Cascada

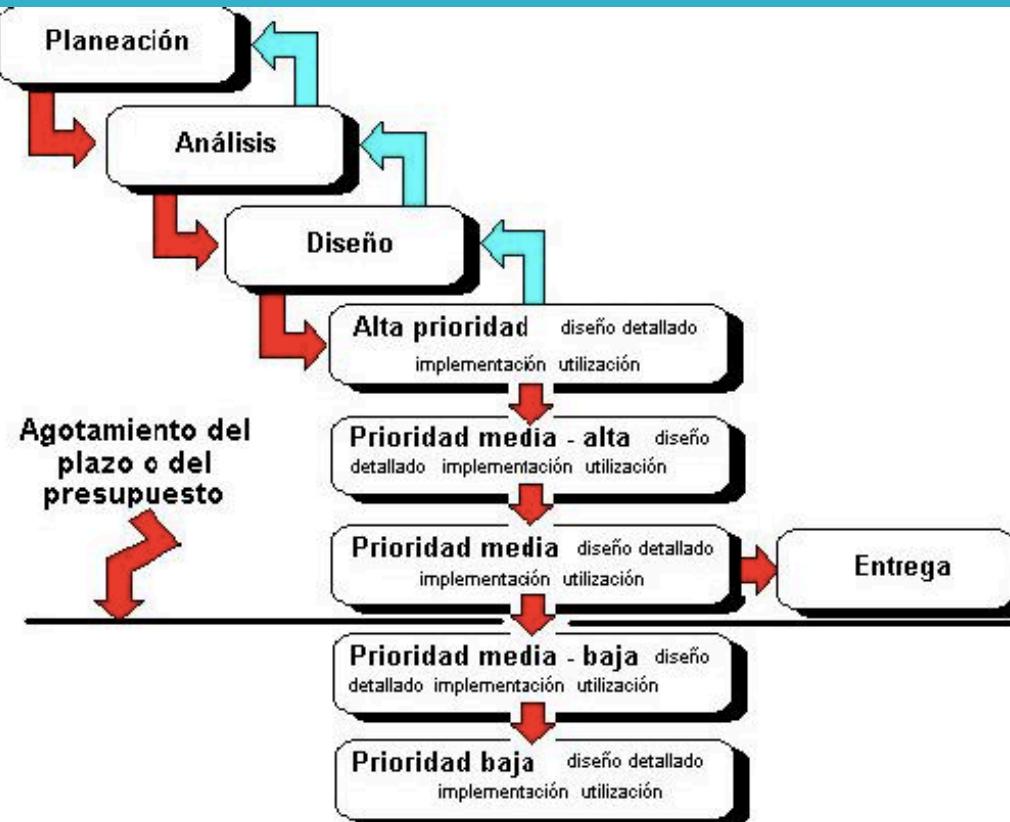
- Proceso de desarrollo secuencial, también denominado desarrollo clásico o lineal



Características

- Muy utilizado en adaptaciones, o mejoras a sistemas existentes.
- Útil para proyectos de requisitos fijos.
- Es raro que un proyecto real sea capaz de seguir esta metodología.
- Es difícil que un cliente conozca los requerimientos de forma fija.
- El cliente debe ser paciente.

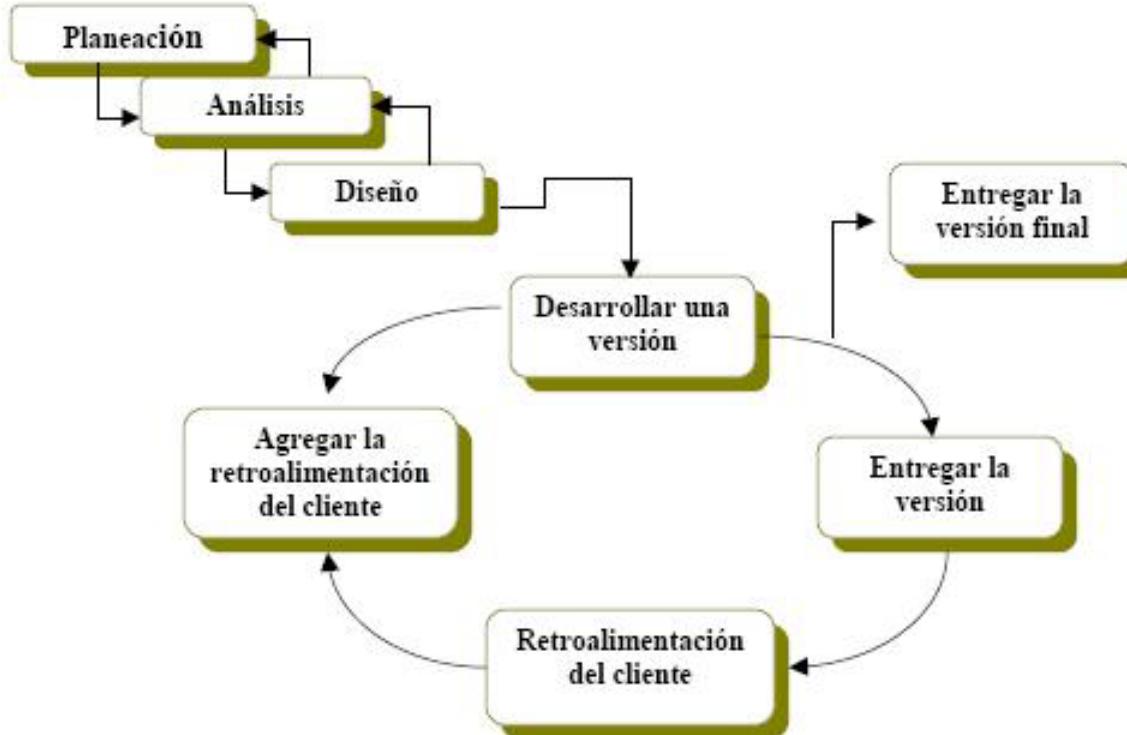
Variaciones: Por Planificación



Ventajas y desventajas

- **Ventajas**
 - Puede ser una estrategia válida para asegurar que se tiene un producto listo en una fecha determinada.
 - Esta estrategia es particularmente útil para las partes del producto que no se quieren realizar en el camino crítico.
- **Desventajas**
 - Si no se completan todas las etapas, se desperdiciará tiempo en la especificación, arquitectura y diseños de prestaciones que no se van a entregar.

Variaciones: Modelo evolutivo

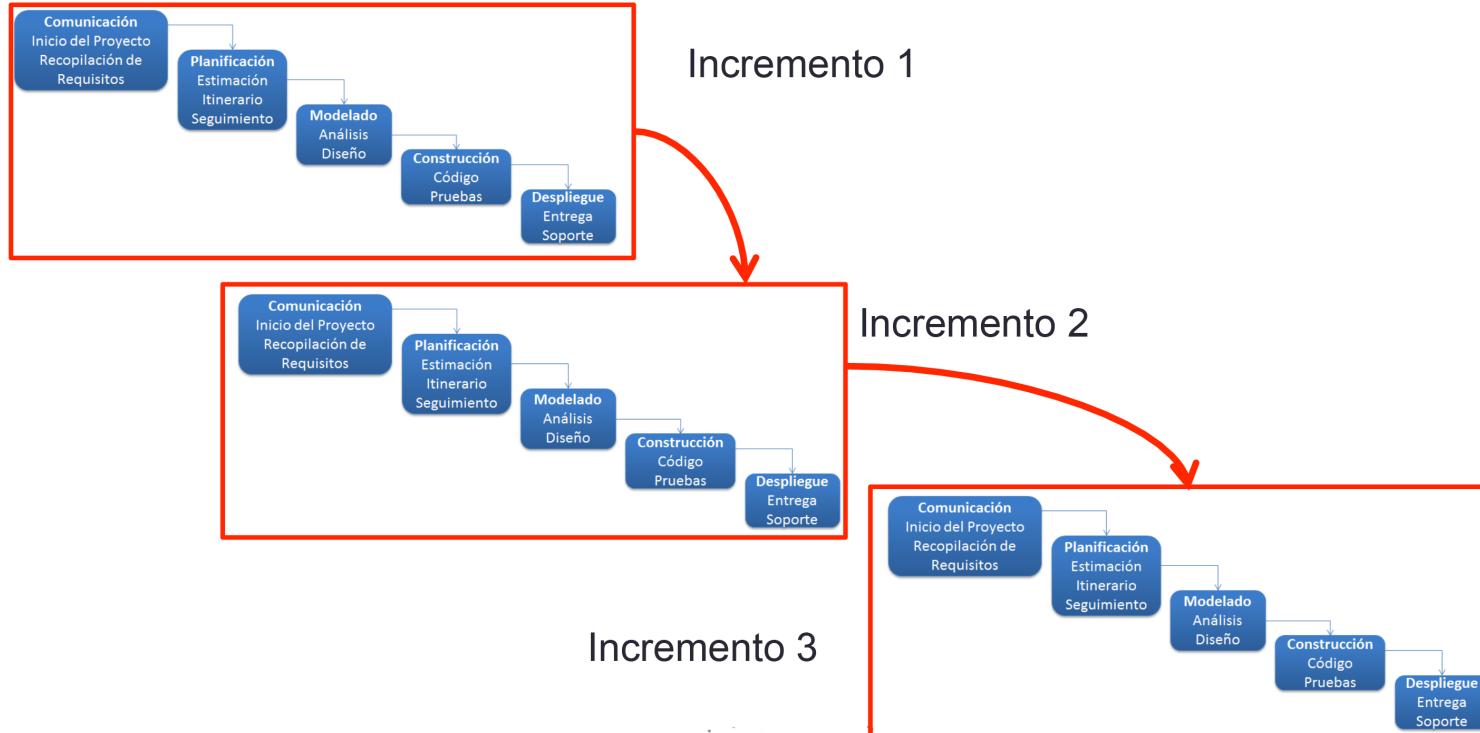


Ventajas y Desventajas

- Ventajas
 - Permite proporcionar una funcionalidad útil en las manos del cliente antes de entregar el 100% del proyecto.
 - Con una planificación cuidadosa, es posible entregar las prestaciones más importantes al principio, y el cliente puede comenzar a usar el sistema en ese punto.
 - Proporciona signos tangibles de progreso en el proyecto.
- Desventajas
 - Depende de la planificación inicial.

Incremental

- Aplica cascada tradicional repetidas veces.



Características

- El primer incremento corresponde al producto esencial.
- El plan se va modificando al entregar cada incremento, para cubrir de mejor manera las necesidades del cliente.
- Se entrega un producto operacional al finalizar cada iteración.
- Útil cuando no se tiene todo el personal necesario para el desarrollo del proyecto.

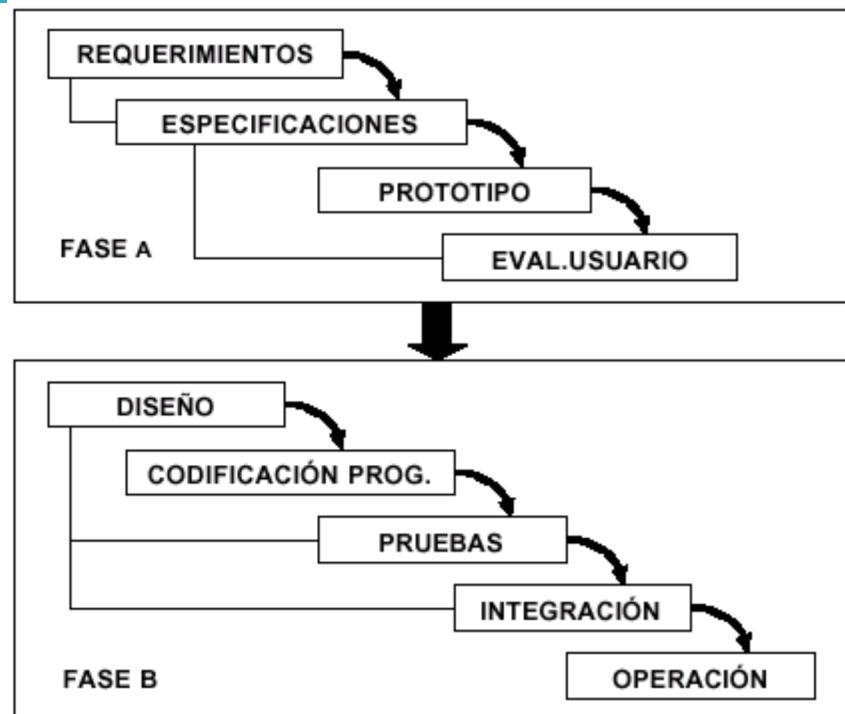
Evolutivos

- El software por su naturaleza evoluciona con el tiempo.
- Los requisitos van cambiando a medida que se desarrolla el sistema.
- Los modelos evolutivos son iterativos.

Modelo de prototipos

- Este paradigma se basa en la producción de una versión operacional del software, para un conjunto de requisitos limitado (excluyéndose parte de la funcionalidad).

Prototipo desecharable



Prototipo evolutivo



Boehm 1984

Desventajas

- Se puede adicionar funcionalidad intrascendente bajo presiones de uso normal.
- Posibilidad de pensar que es en sí una especificación (solo es parte).
- Puede demostrar funcionalidad que no es posible bajo apremios reales.
- El usuario puede creer que tiene frente a sí el sistema completo y operable.
- Posibilidad de subestimar el proyecto, ya que las salidas están pronto disponibles.

Ventajas

- Los usuarios tienen la posibilidad de interactuar con el prototipo y realimentar a los desarrolladores.
- Para los usuarios es más confortable enfrenta a un prototipo que a una especificación (mayor identificación).
- Decrece el esfuerzo de desarrollo entre un 40% y un 70%. Un diseño rápido es posible cuando los requisitos son claros.

Espiral

- Se basa en un acercamiento orientado a la determinación del riesgo en la evolución del software.
- Sus principales actividades son:
 - Planificación
 - Análisis de riesgo
 - Ingeniería
 - Evaluación del cliente

PLANIFICACIÓN

Recolección de Requisitos
y Planificación del
Proyecto (iniciales).

Planificación
basada en los
comentarios del
cliente.

Evaluación del
cliente.

EVALUACIÓN DEL CLIENTE

ANÁLISIS DE RIESGO

Análisis de riesgo basado
los requisitos
iniciales.

Análisis de riesgo
basado en la reacción
del cliente.

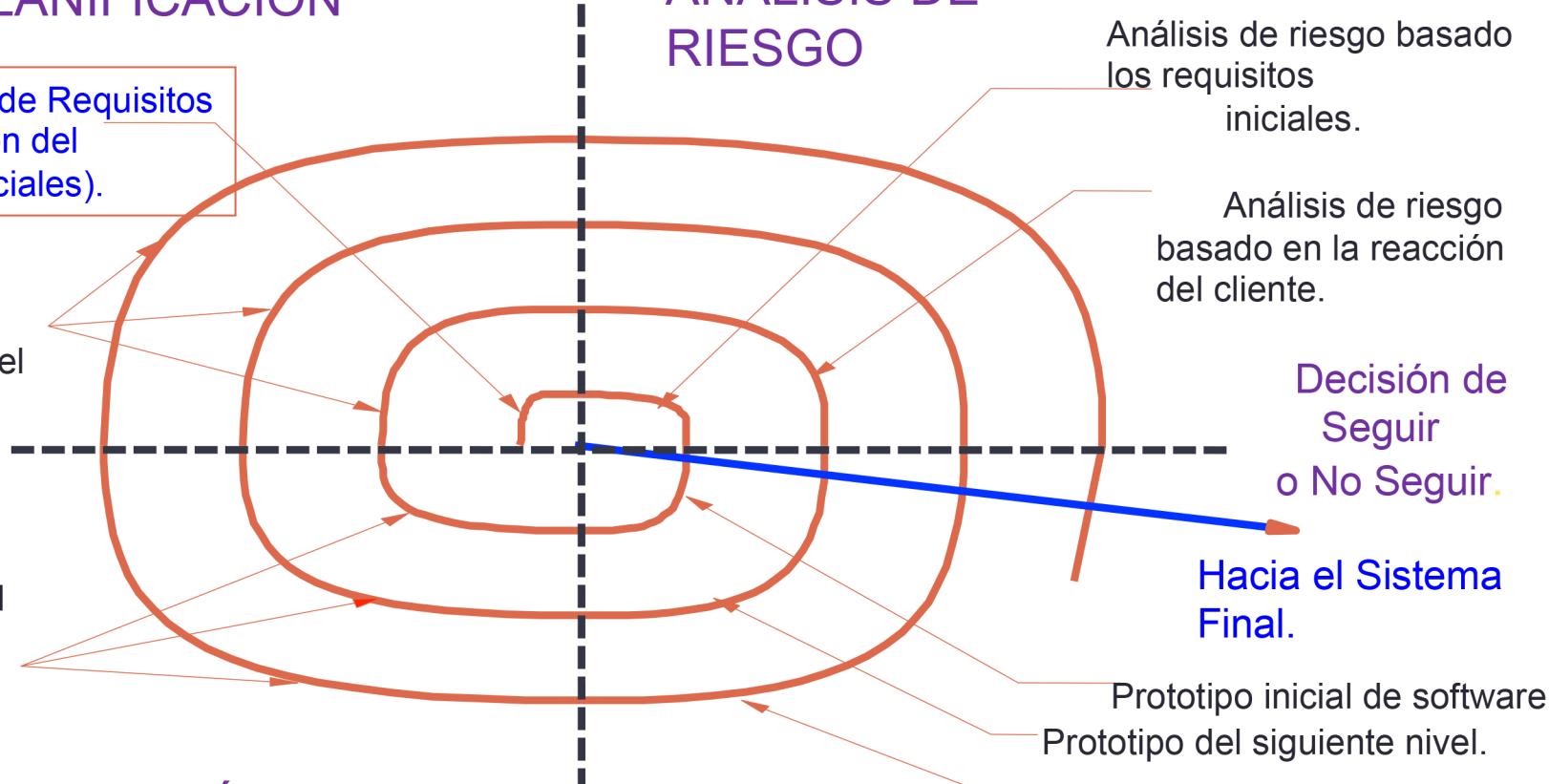
Decisión de
Seguir
o No Seguir.

Hacia el Sistema
Final.

Prototipo inicial de software
Prototipo del siguiente nivel.
Sistema de Ingeniería.

INGENIERÍA

Boehm (1986)



Ventajas y Desventajas

- Desventajas
 - Criticidad del análisis de riesgo.
 - Difícil de “vender” como algo controlable.
- Ventajas
 - Demanda una consideración directa de los riesgos técnicos en todas las etapas del proyecto.
 - Permite la utilización de la creación de prototipos como un mecanismo de reducción del riesgo.
 - Permite utilizar el enfoque de creación de prototipos en cualquier etapa de evolución del producto.

Metodologías Ágiles

- Al individuo y las interacciones en el equipo de desarrollo más que a las actividades y las herramientas
- Desarrollar software que funciona más que conseguir una buena documentación ⇒ Minimalismo respecto del modelado y la documentación del sistema
- La colaboración con el cliente más que la negociación de un contrato
- Responder a los cambios más que seguir estrictamente una planificación

Manifiesto

1. La prioridad principal es satisfacer al cliente mediante tempranas y continuas entregas de software que le reporte un valor
2. Dar la bienvenida a los cambios. Las Mas capturan los cambios para que el cliente tenga una ventaja competitiva
3. Entregar frecuentemente software que funcione, desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre una entrega y la siguiente

... Manifiesto

4. La gente del negocio y los desarrolladores deben trabajar juntos a lo largo del proyecto
5. Construir proyecto en torno a individuos motivados. Darles el entorno y el apoyo que necesitan y confiar en ellos para conseguir el trabajo
6. El diálogo cara a cara es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo
7. El software que funciona es la medida principal de progreso

... Manifiesto

8. Los procesos ágiles promueven un desarrollo sostenible. Los promotores, desarrolladores y usuarios deberían ser capaces de mantener una paz constante
9. La atención continua a la calidad técnica y al buen diseño mejora la agilidad
10. La simplicidad es esencial
11. Las mejores arquitecturas, requisitos y diseños surgen de los equipos organizados por sí mismos
12. En intervalos regulares, el equipo reflexiona respecto de cómo llegar a ser más efectivo, y según esto ajusta su comportamiento

Comparación

Metodología Ágil	Metodología No Ágil
Pocos Artefactos	Más Artefactos
Pocos Roles	Más Roles
No existe un contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
Cliente es parte del equipo de desarrollo (además in-situ)	El cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños (< 10 integrantes) y trabajando en el mismo sitio	Grupos grandes
Menos énfasis en la arquitectura	La arquitectura es esencial

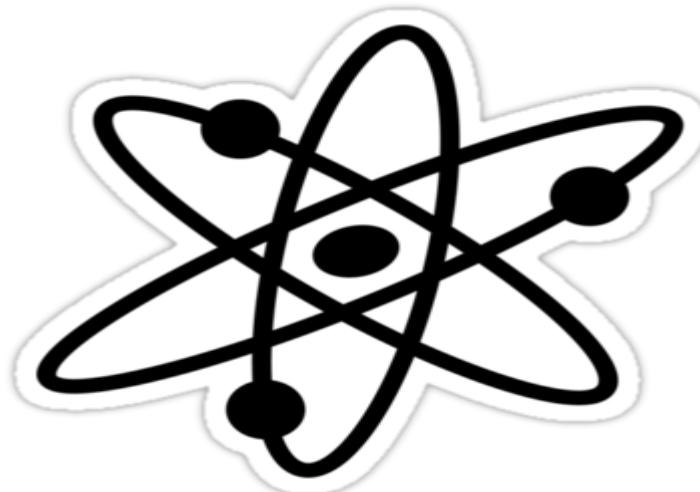


BPMN

Modelando Procesos

□ ¿Qué es un Modelo?

- ✓ Es una abstracción de la realidad
- ✓ Cada especialidad cuenta con una herramienta para generar modelos muy específicos



Elementos claves de un modelo

- Deben ser una buena conceptualización del problema en cuestión
- Permite comunicarse entre los actores del proyecto (cliente, ingenieros, desarrolladores, etc.)

Desventajas de los Modelos

- A veces pueden perder el objetivo del proyecto
 - ✓ Gastar o perder más tiempo del necesario en tener modelo perfecto
 - ✓ Gastar más tiempo del necesario en actualizar el modelo
- Tiene el modelo, pero no tiene la documentación

Modelos Utilizados en Sistemas de Administración Administrativos (SIA)

- Para Modelar Procesos
 - Business Process Modeling Notation – BPMN**



Qué es BPMN

- Es una notación **Gráfica** para dibujar procesos de negocios que permite definir los responsables de cada tarea.

Objetivos

- Entregar una notación estándar que es fácil de entender para todos los stakeholders involucrados en el negocio.
- Crear un lenguaje de negocios común a todos.
- Crear un puente de comunicación entre el proceso, el diseño y la implementación.

Porque se usa BPMN?

- Es ampliamente aceptado
- Reemplaza numerosos modelos de proceso (notación, métodos y lenguajes).
- Herramienta simple de aprender que posee un alto potencial para modelar complejos modelos de negocios.
- Software de modelamiento gratis en la red.
- Proveedor neutral.

¿ Porqué es importante realizar diseño de los procesos?

- Los procesos son el core de cualquier organización, aún así no siempre están claramente definidos, documentados y optimizados.
- La necesidad de un puente de comunicación entre el negocio y TI es más fuerte que nunca.
- La tasa de crecimiento de los negocios incrementa y la presión por volverse más eficiente, por lo que la organización DEBE tener una visión clara de cómo operan sus procesos.

BPMN ofrece una notación que permite documentar los procesos sin ambigüedad.

Introducción BPMN

- BPMN define un Business Process Diagram (BPD) que se basa en una técnica de grafos de flujo. Un modelo de procesos de negocios es una red de gráficos, que son actividades y controles de flujo que definen su orden de rendimiento.
- Las cuatro categorías básicas de elementos son:
 1. Objetos de Flujo
 2. Objetos Conectores
 3. Artefactos
 4. Swimlanes (Carriles de piscinas)

Objetos de Flujo

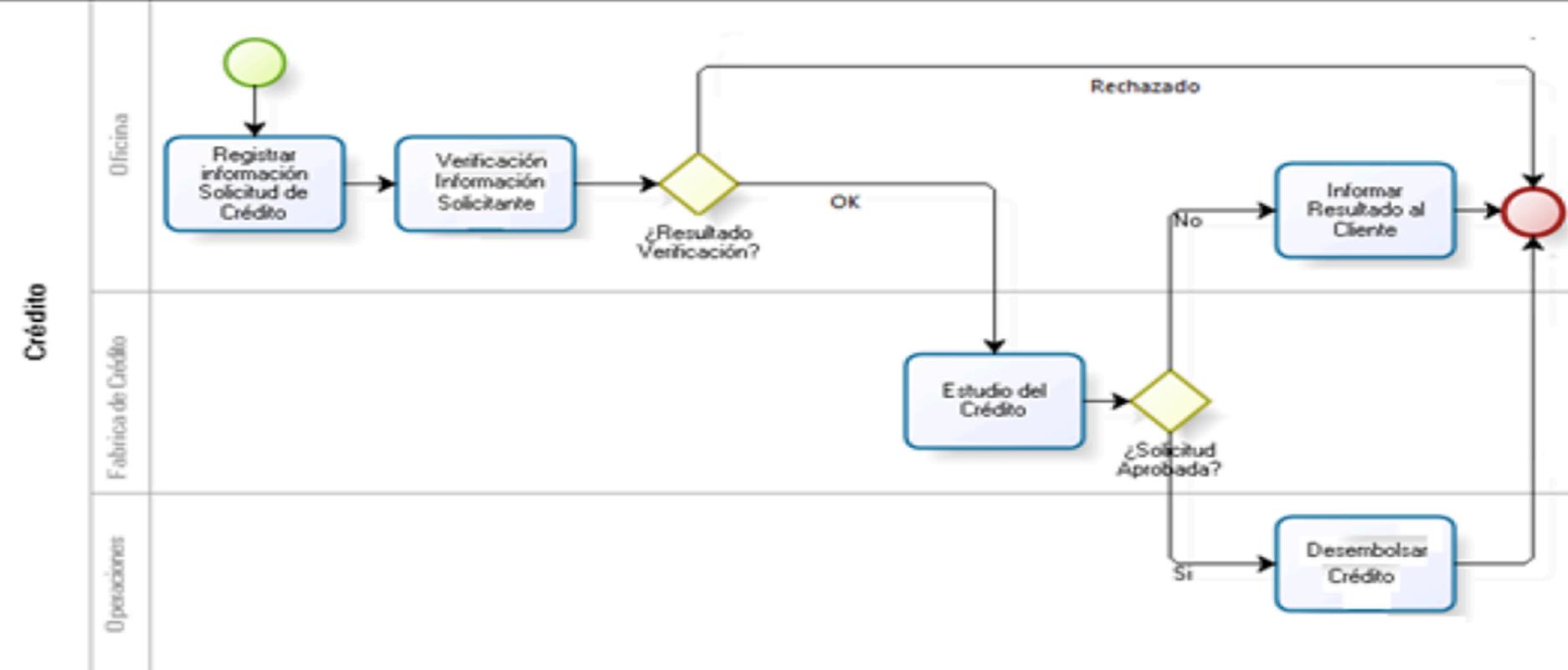
- **Evento:** El algo que “pasa” durante el curso del proceso negocio. Suelen tener una causa o un impacto (resultado)

Problema para explicación

El Proceso de Solicitud Crédito gestiona las actividades necesarias para recibir, analizar y aprobar solicitudes registradas por los clientes de una entidad financiera.

Una versión simplificada de este proceso consta del registro de la solicitud, la verificación de la información del solicitante y el estudio del crédito.

Al registrar la solicitud el cliente manifiesta su interés de adquirir un crédito y presenta la documentación requerida a la entidad. Luego un agente realiza la verificación de la información presentada por el cliente, y posteriormente la fábrica de crédito realiza estudio de la solicitud. Por último se realizan las actividades necesarias para desembolsar el monto solicitado o informar el rechazo de la solicitud al cliente.



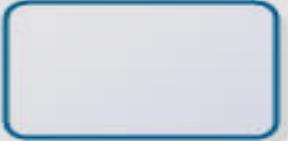
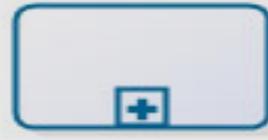
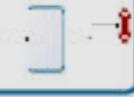
Eventos

TIPO DE EVENTO	NOMBRE BPMN	DEFINICIÓN	NOTACIÓN
Inicio	Start	Como su nombre lo indica, representa el punto de inicio de un proceso.	
Intermedio	Intermediate	Ocurren entre un evento de inicio y de fin. Afectará el proceso pero no lo iniciará o directamente finalizará.	
Fin	End	Indica cuando un proceso termina.	

Objetos de Flujo

- **Actividad:** Término genérico para el trabajo que hace una compañía,. Puede ser atómica (Tarea) o compuesta (Sub-proceso)

Tareas

NOMBRE BPMN	USO	NOTACIÓN
Tarea de Usuario	Es una tarea de "flujo de trabajo" donde un humano realiza una tarea que tiene que ser completada en cierta cantidad de tiempo. Se usa cuando el trabajo durante el proceso no puede ser descompuesto en un nivel más fino dentro del flujo. Tarea de usuario	
Subproceso	Un subprocesso es una actividad compuesta incluida dentro de un proceso. Éste es compuesto dado el hecho que esta figura incluye un conjunto de actividades y una secuencia lógica (proceso), que indica que la actividad mencionada puede ser analizada a un nivel más fino. Se puede colapsar o expandir. Subproceso.	   

Objetos de Flujo

- **Gateway (Compuerta):**

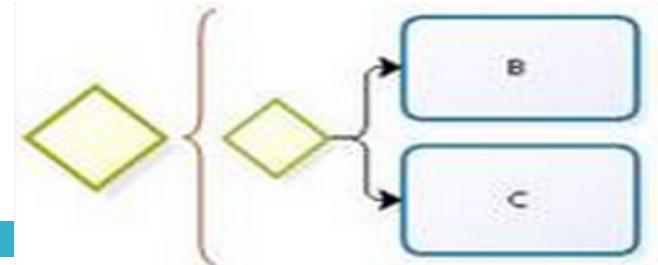
TIPO DE DECISIÓN	DEFINICIÓN	NOTACIÓN
Decisión Exclusiva	Decisión basada en datos del sistema. El mismo elemento se usa para sincronizar esta figura. Decisión Exclusiva	



Compuerta Paralela

Divergencia: Se utiliza cuando varias actividades pueden realizarse concurrentemente o en paralelo.

Convergencia: Permite sincronizar varios caminos paralelos en uno solo. El flujo continúa cuando todos los flujos de secuencia de entrada hayan llegado a la figura.



Conectores: Secuencia

TIPO DE LINEA	NOMBRE ORIGINAL	DEFINICIÓN	NOTACIÓN
Línea normal	Normal Flow	La línea normal se refiere al flujo que se originan en el inicio, continúa a través de actividades hasta terminar en un evento de salida (por ejemplo el FIN).	
Flujo Condicional	Conditional Flow	Este flujo tiene una condición asignada que define si el flujo es usado. Se puede asignar a cualquier figura en el proceso que requiera evaluar una condición para seguir cierto camino.	

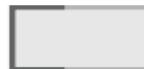
..... ➤ Mensaje

- Las líneas de mensaje representan la interacción entre varios procesos o pools.
- Representan Señales o Mensajes NO flujos de control.
- No todas las líneas de mensaje se cumplen para cada instancia del proceso y tampoco se especifica un orden para los mensajes.

..... Asociaciones

- Se usan para asociar información adicional sobre el proceso. También se usan para asociar tareas de compensación

Artefactos



Anotaciones

- Son utilizados para proporcionar información adicional sobre el proceso.



Grupos

- Se utiliza para agrupar un conjunto de actividades, ya sea para efectos de documentación o análisis, no afecta la secuencia del flujo.



Objetos de Datos

- Permiten mostrar la información que una actividad necesita, como las entradas o las salidas.

Swimlanes (carriles de piscinas)



Pool

- Actúa como contenedor de un proceso
- El nombre del pool puede ser el del proceso o el del participante.
- Representa un Participante Entidad o Role.
- Siempre existe al menos uno, así no se diagrame.



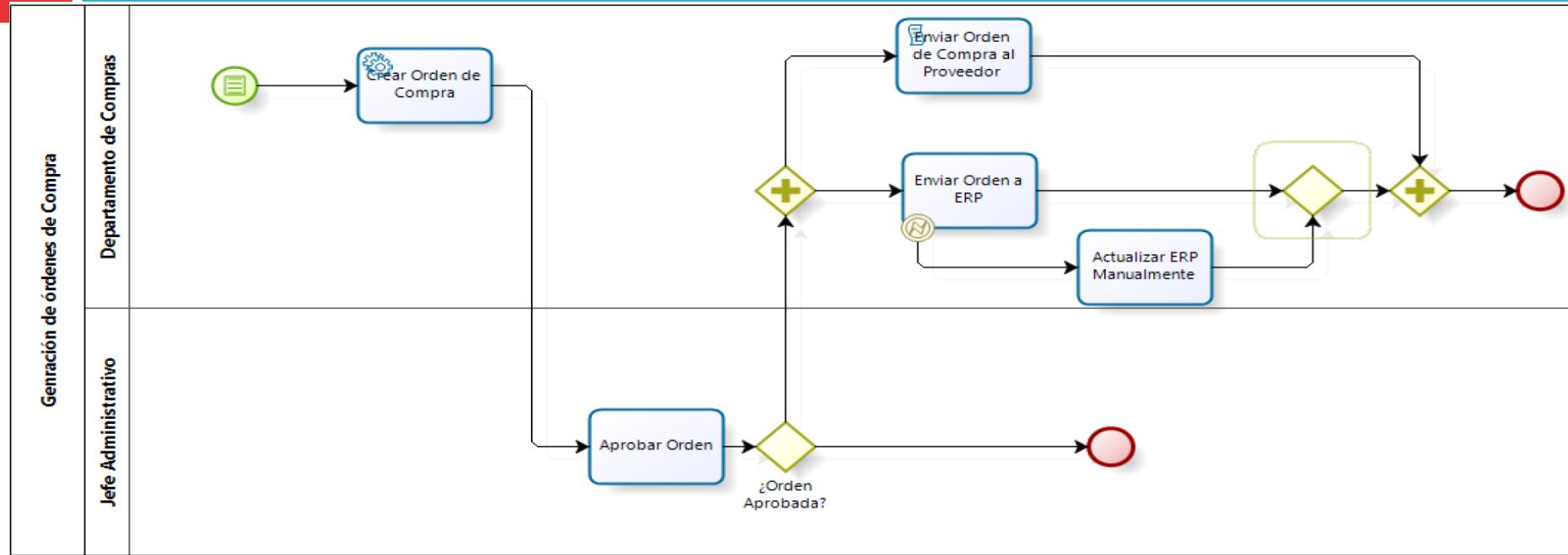
Lane

- Subdivisiones del Pool.
- Representan los diferentes participantes al interior de una organización.

Ejercicio – Órdenes de compra

Este proceso tiene como objetivo generar órdenes de compra automáticamente de acuerdo a los niveles de inventario de una materia prima específica, gestionar su aprobación (Por un Jefe de administración) e ingreso en los sistemas contables de la empresa además de realizar el envío de las mismas a los proveedores.

BPMN



Requisitos

Especificación de Requisitos

- Descripción completa del software a desarrollar
- IEE 830-1998
 - Completa
 - Consistente
 - Inequívoca
 - Correcta
 - Trazable
 - Priorizable
 - Modificable
 - Verifiable

Tabla de Requisitos Funcionales

- ID
- Nombre
- Descripción
- Prioridad: Baja, Media o Alta.
- Categoría: Oculta o Evidente

Requisitos No Funcionales

- Corresponden a los atributos o características de calidad que debe tener el software.
- En 1991, la organización internacional de estándares (ISO) en conjunto con la comisión electrónica Internacional (IEC) propusieron un estándar para la evaluación de calidad del software, denominado ISO 9126.
- El estándar ISO/IEC 9126 se compone de cuatro partes: modelo de calidad, métricas externas, métricas internas y métricas para la calidad en uso.
- En la primera parte, se describen detalladamente seis características y sub-características de calidad para los productos de software.
- Se puede determinar el grado de calidad de estos productos según la evaluación de estas características y sub-características.

Tabla de características

Característica	Sub-Característica
Funcionalidad	Adecuación, Corrección, Interoperabilidad, Seguridad, Conformidad
Fiabilidad	Madurez, Tolerancia a fallos, Recuperabilidad, Conformidad
Usabilidad	Comprensibilidad, Aprendibilidad, Operabilidad, Atractividad, Conformidad
Eficiencia	Comportamiento temporal, Utilización de recursos, Conformidad
Mantenibilidad	Analizabilidad, Cambiabilidad, Estabilidad, Facilidad de prueba, Conformidad
Portabilidad	Adaptabilidad, Instalabilidad, Coexistencia, Reemplazabilidad, Conformidad

Tabla de Requisitos no funcionales

- ID
- Nombre
- Prioridad
- Característica
- Sub-característica
- Descripción.

Sub-Características: Funcionalidad

- Adecuación – Tareas específicas
- Exactitud – Respuestas o efectos
- Interoperabilidad – Otros sistemas
- Seguridad
- Conformidad : a los requisitos funcionales

Sub-Características: Fiabilidad

- Madurez – frecuencia de fallas.
- Recuperabilidad: Medidas para recuperarse de una falla.
- Tolerancia a fallos – Niveles de desempeño, para casos específicos.
- Cumplimiento – normas, estándares.

Sub-Características: Usabilidad

- Aprendizaje: facilidad de aprender a utilizar
- Comprensión: Entender lo que se va a realizar y sus implicancias.
- Operatividad: Como se utiliza en el día a día, ¿es costoso?
- Atractividad: Linda

Sub-Características: Mantenibilidad

- Estabilidad: Como se comporta ante las modificaciones.
- Facilidad de análisis: Indican la forma en que se realizan análisis sobre el sistema para detectar defectos o posibles fallas.
- Facilidad de cambio: Facilidad de pruebas

Sub-Características: Eficiencia

- Tiempo
- Recursos.

Sub-Características: Portabilidad

- Capacidad de instalación: en distintos ambientes.
- Capacidad de reemplazamiento: Fijar bien los límites para que quede claro la forma de modificar las funciones del sistema por otro.
- Adaptabilidad a distintos ambientes.
- Co-Existencia: como funciona con otros ambientes.

Ejemplo

- Sistema de gestión de alumnos y notas de un colegio.
 - Profesores pueden subir notas y anotaciones de alumnos.
 - Pueden generar informes de personalidad
 - Se pueden enviar correos masivos a los padres.
 - Los padres pueden ver las notas y anotaciones de sus hijos.

Ejemplo: Funcionales

ID	Nombre	Prioridad	Categoría	Descripción
RF00	Ingresar al Portal	Alta	Evidente	Los usuarios deben entrar al sistema por medio de un login para poder visualizar la información y/o realizar una acción
RF01	Crear Usuario	Alta	Evidente	El usuario administrador puede crear usuarios
RF02	Modificar Usuario	Baja	Evidente	El usuario administrador puede modificar datos de los usuarios
RF03	Eliminar Usuario	Media	Evidente	El usuario administrador puede eliminar un usuario
RF04	Ingresar Nota	Alta	Evidente	El usuario profesor puede ingresar notas
RF05	Modificar Nota	Baja	Evidente	El usuario profesor puede modificar una nota
RF06	Eliminar Nota	Baja	Evidente	El usuario profesor puede eliminar una nota
RF07	Crear Comunicación	Alta	Evidente	El usuario profesor puede crear una comunicación
RF08	Enviar Comunicación	Alta	Evidente	El usuario profesor puede enviar una comunicación
RF09	Generar informe de notas	Alta	Evidente	El usuario profesor puede generar un reporte con las notas almacenadas para cada alumno

Ejemplo: Funcionales

RF10	Ingresar Anotaciones	Alta	Evidente	El usuario profesor puede ingresar una anotación
RF11	Modificar Anotación	Baja	Evidente	El usuario profesor puede modificar una anotación
RF12	Recibir Anotación	Alta	Evidente	El usuario apoderado puede recibir una anotación
RF13	Generar informe de personalidad	Alta	Evidente	El usuario profesor puede generar un informe de personalidad con las anotaciones del alumno
RF14	Recibir Comunicación	Alta	Evidente	El usuario apoderado puede recibir una comunicación
RF15	Responder Comunicación	Alta	Evidente	El usuario apoderado puede responder una comunicación
RF16	Ingresar Actividad en el calendario	Alta	Evidente	Los usuarios profesor y administrador pueden ingresar actividades en el calendario
RF17	Modificar Actividad en el calendario	Media	Evidente	Los usuarios profesor y administrador pueden modificar una actividad en el calendario
RF18	Eliminar Actividad en el calendario	Baja	Evidente	Los usuarios profesor y administrador pueden eliminar una actividad en el calendario
RF19	Manejo de datos en Base de Datos	Alta	Oculto	Los datos que se crean, eliminan, o modifican alteran sus respectivas tablas en la base de datos.

Ejemplo: Funcionales

RF20	Ingresar Foto	Alta	Evidente	El usuario administrador puede ingresar una foto
RF21	Modificar Foto	Media	Evidente	El usuario administrador puede modificar una foto
RF22	Eliminar Foto	Baja	Evidente	El usuario administrador puede eliminar una foto
RF23	Ingresar Asignatura	Alta	Evidente	El usuario administrador puede ingresar una asignatura
RF24	Modificar Asignatura	Media	Evidente	El usuario administrador puede modificar una asignatura
RF25	Eliminar Asignatura	Baja	Evidente	El usuario administrador puede eliminar una asignatura
RF26	Ingresar Curso	Alta	Evidente	El usuario administrador puede ingresar un curso
RF27	Modificar Curso	Media	Evidente	El usuario administrador puede modificar un curso
RF28	Eliminar Curso	Baja	Evidente	El usuario administrador puede eliminar un curso

Ejemplo: No Funcionales

ID	Nombre	Prioridad	Característica	Subcaracterística	Descripción
RNF00	Tiempo de respuesta	Alta	Eficiencia	Comportamiento temporal, conformidad	El sistema debe responder en menos de 5 segundos cuando se realiza una acción o alguna petición.
RNF01	Interoperabilidad	Alta	Portabilidad	Adaptabilidad, conformidad	Portal se debe adaptar a los dispositivos móviles, ya sea smartphones o tablets, a la hora de que los usuarios quieran ingresar al portal.
RNF03	Interfaz simple	Alta	Usabilidad	Atractividad	Portal debe ser simple y atractivo para el usuario a la hora de que lo use.
RNF04	Definir roles	Alta	Funcionalidad	Adecuación	Portal debe tener 3 roles fundamentales; admin, profesores y apoderados.

Ejemplo: No Funcionales.

RNF05	Recuperar estados al fallar el sistema	Alta	Fiabilidad	Tolerancia a fallos	Si es que falla, se recupera el avance que se ha hecho hasta la última modificación de la base de datos.
RNF06	Notas de alumnos	Alta	Funcionalidad	Seguridad, conformidad	Un apoderado no puede ver las notas de los alumnos que no estén asociados a su hijo.
RNF07	Mantenimiento	Alta	Mantenibilidad	Cambiabilidad, analizabilidad, conformidad	Se debe tener una buena trazabilidad para las posibles mantenciones o cambios que se querrán hacer a futuro.
RNF08	Manejo en el portal	Baja	Usabilidad	Operatividad, conformidad	Que sea fácil para el apoderado llegar a lo que está buscando en el portal.
RNF09	Validación de datos	Alta	Funcionalidad	Corrección	Que los datos ingresados en los distintos campos del sistema sean válidos con el formato requerido y/o validación correspondiente.

Casos de uso

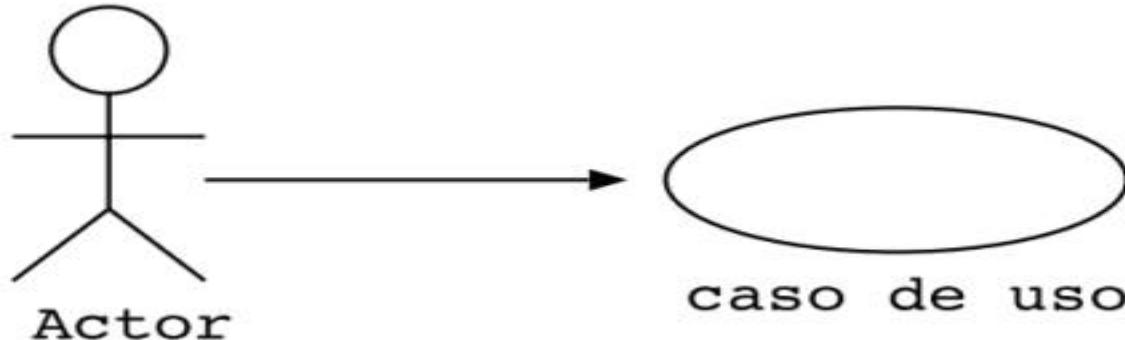
Definición

- Es una estructura que ayuda a los analistas a trabajar con los usuarios para determinar la forma en que se usará un sistema.
- Con un conjunto de casos de uso se puede obtener un bosquejo del sistema en términos de lo que los usuarios intentan hacer con él.

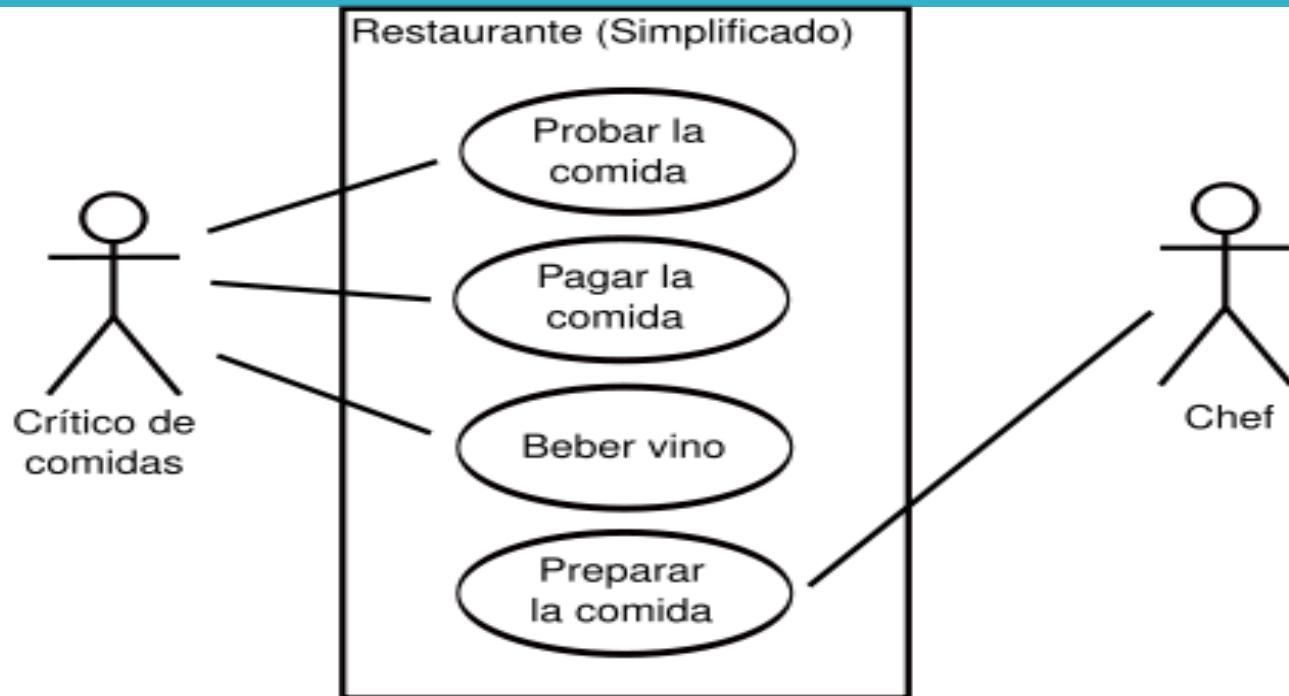
Características

- Fáciles de comprender/validar por los usuarios.
- Guían todo el proceso de desarrollo.

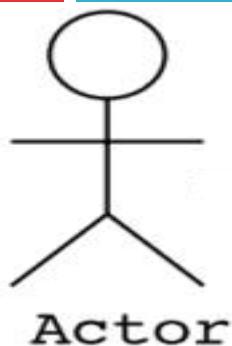
□ Ayudan a organizar el desarrollo de manera jerárquica.



Diagramas



Actor



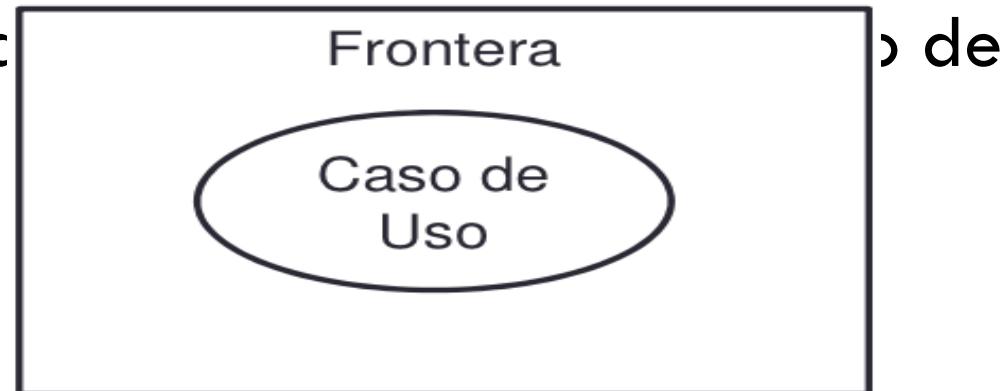
- ❑ La misma persona física puede interpretar varios papeles como actores distintos
- ❑ El nombre del actor describe el papel desempeñado
- ❑ Los casos de uso son lo que sucede cuando el actor interactúa con el sistema
- ❑ El actor usa el sistema para conseguir un objetivo
- ❑ Al registrar todas las formas en que el sistema se usa (Casos de Uso) acumulamos todos los objetivos o requerimientos del sistema.

Caso de uso

- Muestran los requerimientos funcionales en una forma fácil de leer y de trazar.
- Representan los objetivos de la interacción entre el actor y el sistema. Este objetivo representa algo medible y relevante para el actor.
- Registra un conjunto de caminos (escenarios) que atraviesa un actor desde el comienzo del caso hasta la meta (escenarios exitosos).
- Registra un conjunto de caminos (escenarios) que atraviesa un actor desde el comienzo del caso pero que por alguna razón no consigue el objetivo (escenarios de falla)

Frontera

- Es el límite físico y/o lógico para el caso de uso.
Representa la frontera del sistema modelado.
- Los actores pueden ser internos o externos a la
frontera del caso de uso.
- Una frontera puec
uso.

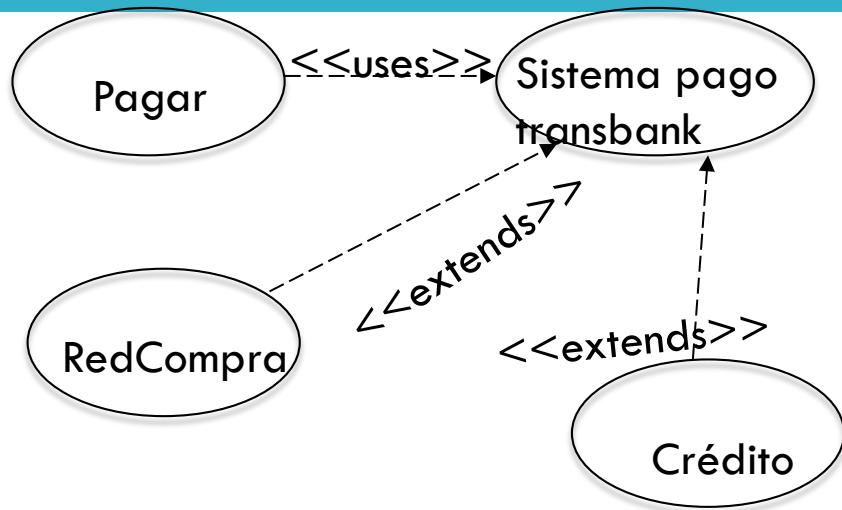


Identificación

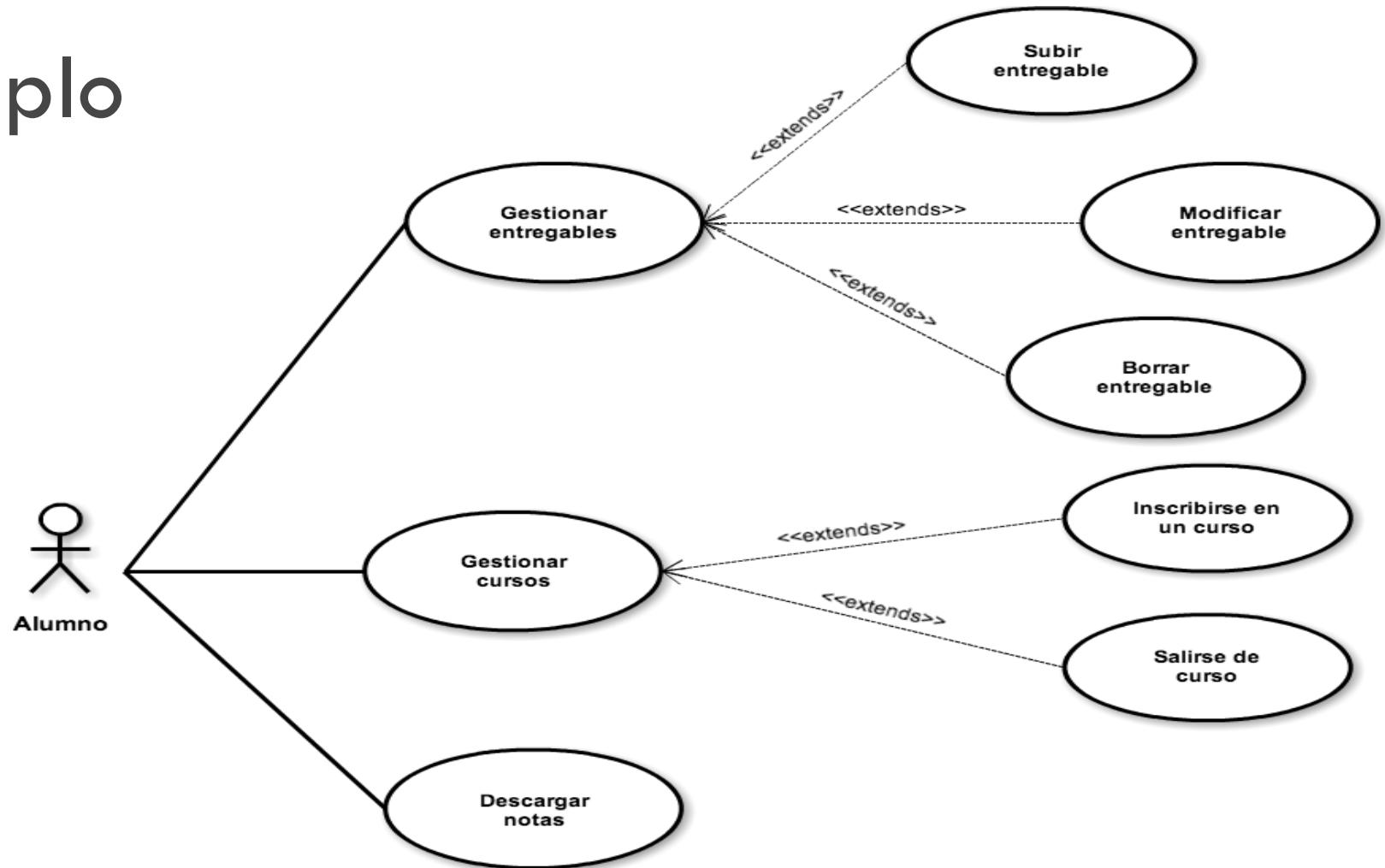
- Un método de identificación se basa en actores:
 1. Se identifican los actores relacionados con un sistema o empresa.
 2. Para cada actor se identifican los procesos que inician o en los cuales participan.
- Otro método de identificación se basa en eventos:
 1. Se identifican los eventos externos a los que un sistema ha de responder.
 2. Se relacionan los eventos con los actores y con los casos de uso.

Estereotipos

- <<uses>> o <<include>>: indica que para que un caso de uso en específico (desde el que sale la flecha) se ejecute, tendrá que ocupar otro de forma previa (al que llega la flecha).
- <<extends>>: es una especificación de un caso de uso, se lee como que un caso de uso (desde el que sale) extiende de



Ejemplo



Caso de uso de alto nivel

Caso de uso: <Nombre>

Actores: <actor1>,, <actor n>

Tipo: <tipo del caso>, se asume es tipo esencial, sólo se especifica Primario, secundario u opcional.

Descripción: <una breve descripción del proceso>

Caso de uso expandido

Caso de uso: <Nombre>

Actores: <actor1>,, <actor n>

Propósito: <propósito>

Tipo: <tipo del caso>, se asume es tipo esencial, solo se especifica Primario, secundario u opcional.

Resumen: <una breve descripción del proceso>

Referencias cruzadas: <casos o funciones del sistema relacionadas con el cu especificado>

Caso de uso expandido

- **Curso Normal de los Eventos**

Acción del actor

1° Acción 1

.

.

Nº Acción n

Respuesta del sistema

1° Respuesta 1

.

.

Nº Respuesta n

- **Cursos Alternativos:**

- <Línea del curso que posee alternativa>: < Acción alternativa>

.

- **Sección:** <Nombre Sección Alternativa>

Diagrama de interacción

- Los diagramas de interacción muestran cómo se **comunican** los objetos en una interacción.
- Los objetos interactúan para realizar **colectivamente** los servicios ofrecidos por las aplicaciones.
- Los diagramas de interacción son modelos que describen la manera en que colaboran grupos de objetos para representar cierto comportamiento.

Diagramas de secuencia

- Muestra la secuencia cronológica de mensajes entre objetos durante un escenario concreto.
- Cada objeto tiene una línea vertical denominada línea de vida del objeto. Esta línea representa la vida del objeto durante la interacción.
- Cada mensaje se representa mediante una flecha entre las líneas de vida de dos objetos.
- El tiempo transcurre de arriba abajo.
- Cuando existe demora entre el envío y la atención se puede indicar usando una línea oblicua.

Ejemplo

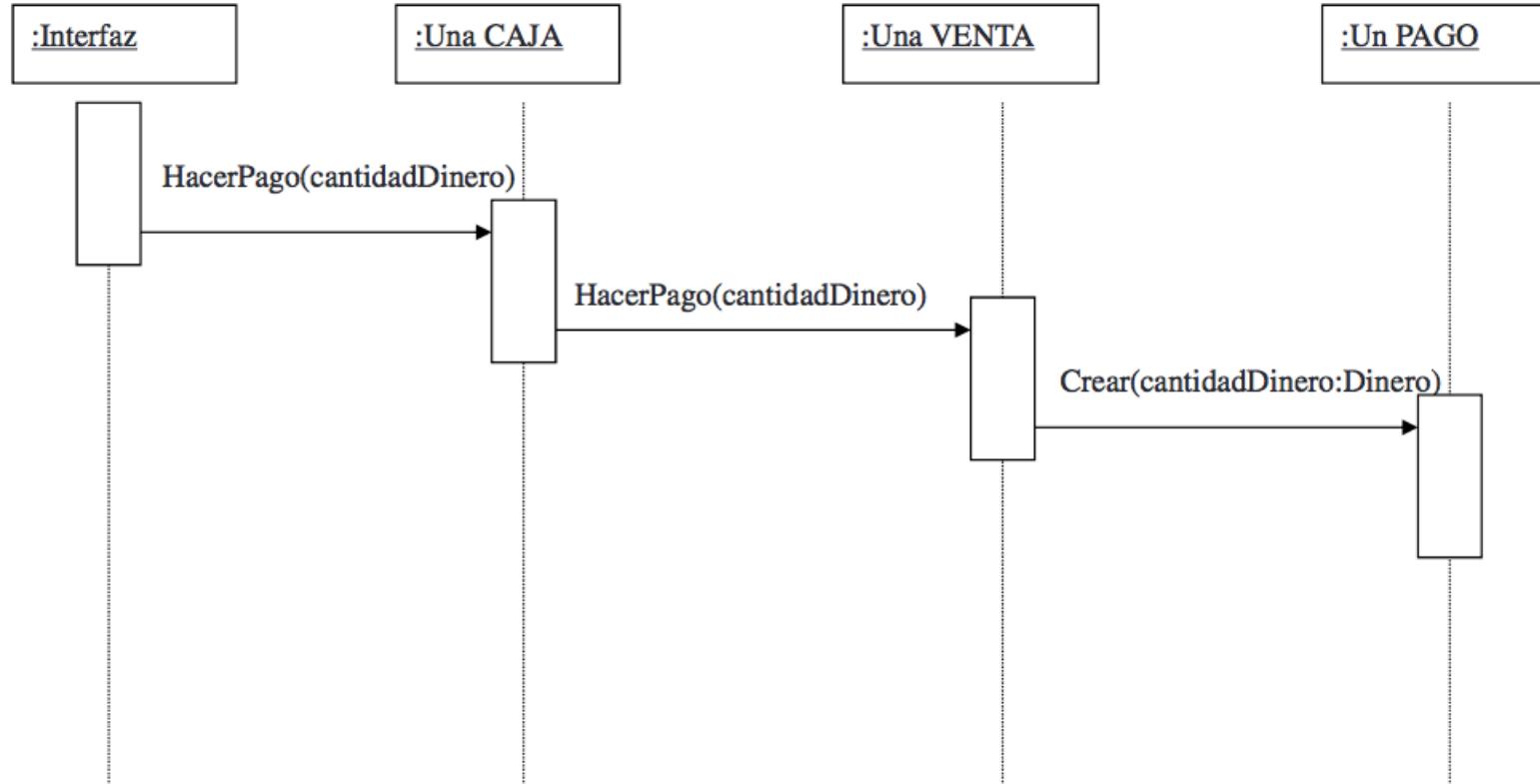


Diagrama de colaboración

- El contexto de una interacción comprende los argumentos, las variables locales creadas en ejecución y los enlaces entre los objetos que participan en la interacción.
- La colaboración se realiza mediante el intercambio de mensajes. Un mensaje desencadena una acción en el objeto destinatario.
- Son útiles en la fase exploratoria para identificar objetos y métodos.

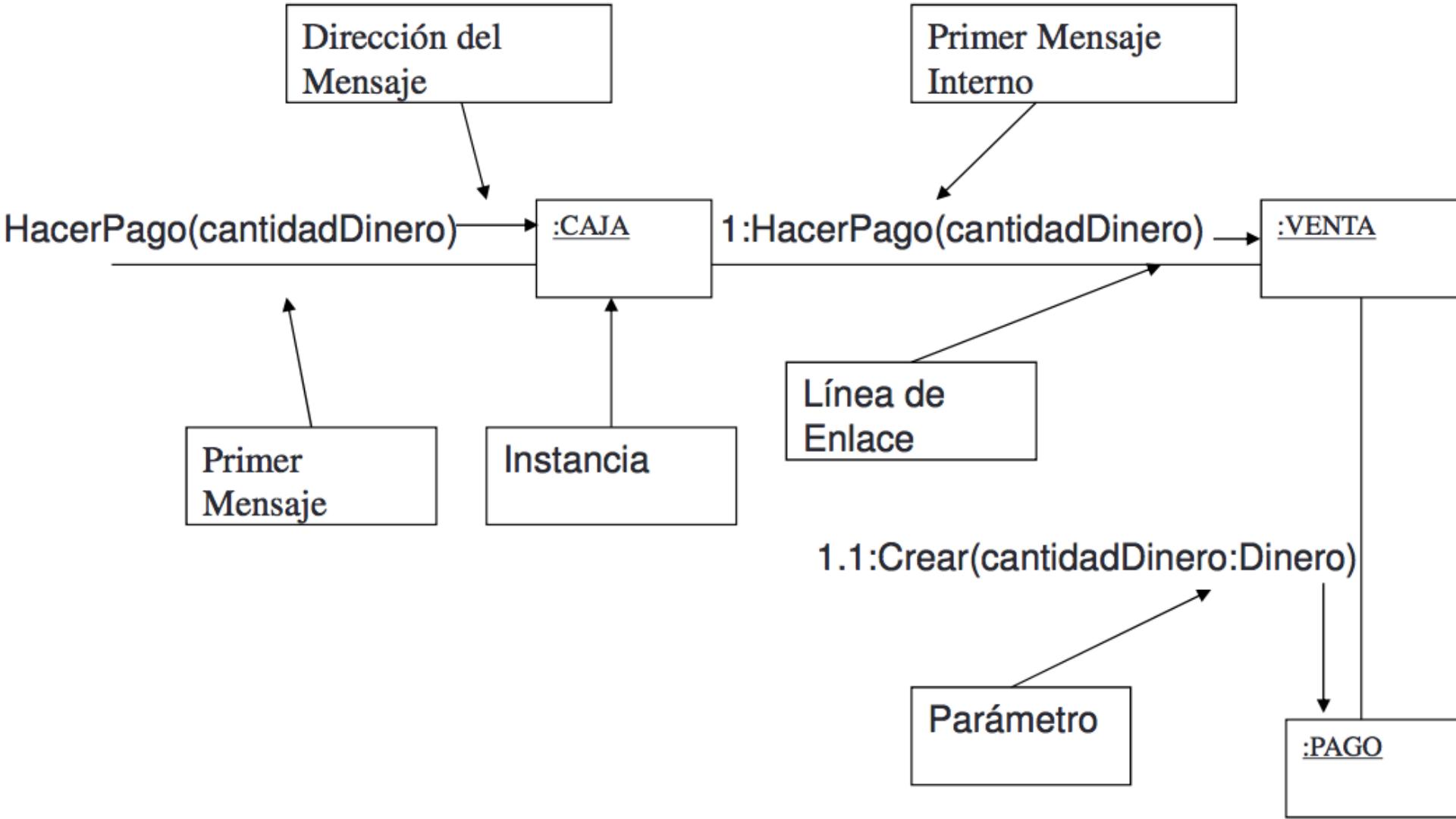


Diagrama de clases

Descripción

- Diagrama de estructura estática, que muestra las clases de un sistema, y como interactúan entre ellas.
- Puede mostrar a la vez lo que el sistema puede hacer, y dar una idea abstracta para la construcción.

¿Qué podemos hacer?

- Identificar clases.
- Relacionar estas clases.
- Identificar atributos.
- Identificar lo que pertenece al sistema y lo que no.

Elementos

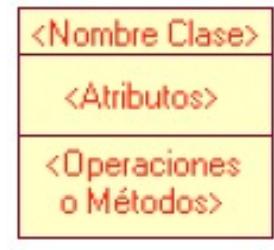
□ Clase

Es la unidad básica que encapsula toda la información de un Objeto (un objeto es una instancia de una clase). A través de ella podemos modelar el entorno en estudio (una Casa, un Auto, una Cuenta Corriente, etc.).

- En UML, una clase es representada por un rectángulo que posee tres divisiones:

En donde:

- **Superior:** Contiene el nombre de la Clase
- **Intermedio:** Contiene los atributos (o variables de instancia) que caracterizan a la Clase (pueden ser private, protected o public).
- **Inferior:** Contiene los métodos u operaciones, los cuales son la forma como interactúa el objeto con su entorno (dependiendo de la visibilidad: private, protected o public).

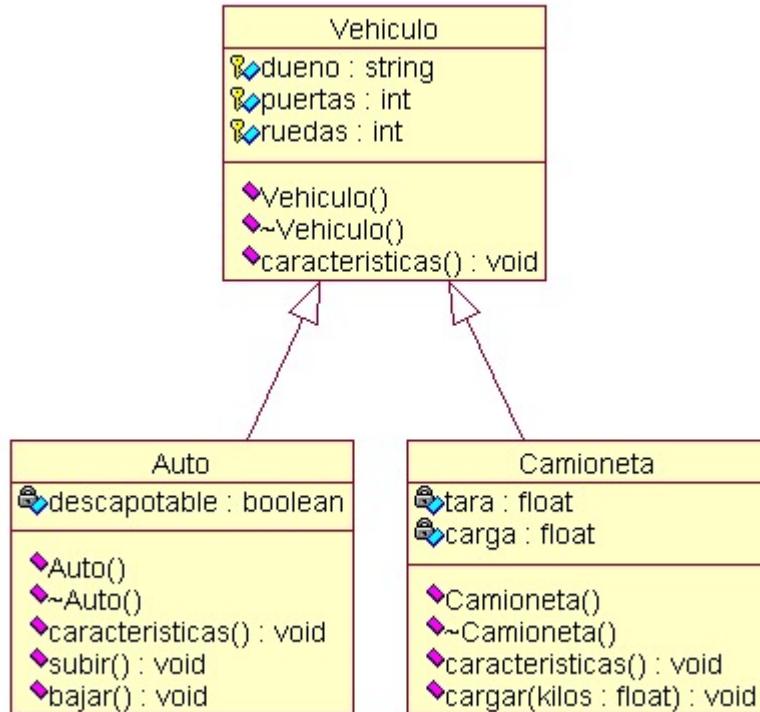


- **Atributos:** son valores que corresponden a un objeto, como color, material, cantidad, ubicación. Generalmente se conoce como la información detallada del objeto. Ejemplo: el objeto es una puerta, sus propiedades o atributos serían: la marca, tamaño, color y peso.
- Tipos de atributos:
 - **public** (+): Indica que el atributo será visible tanto dentro como fuera de la clase, es decir, es accesible desde todos lados.
 - **private** (-): Indica que el atributo sólo será accesible desde dentro de la clase (sólo sus métodos lo pueden utilizar).
 - **protected** (#): Indica que el atributo no será accesible desde fuera de la clase, pero si podrá ser accedido por métodos de la clase además de las subclases que se deriven (ver herencia).

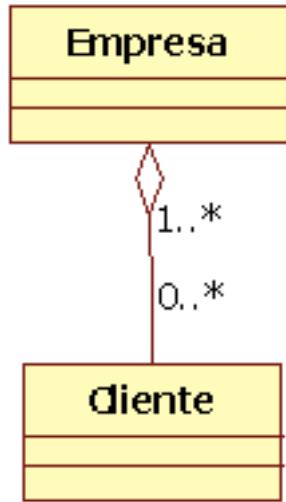
Herencia

□ Herencia (Especialización/Generalización):

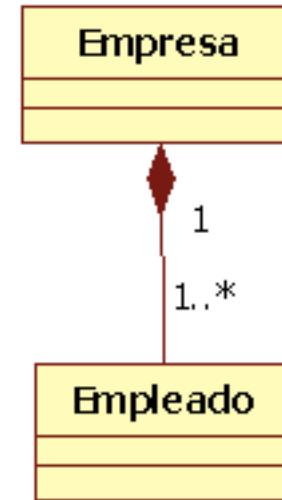
Indica que una subclase hereda los métodos y atributos especificados por una Super Clase (también llamada clase padre), por ende la Subclase además de poseer sus propios métodos y atributos, poseerá las características y atributos visibles de la Super Clase (public y protected).



Aggregación y composición



Aggregación



Composición

Ejemplo

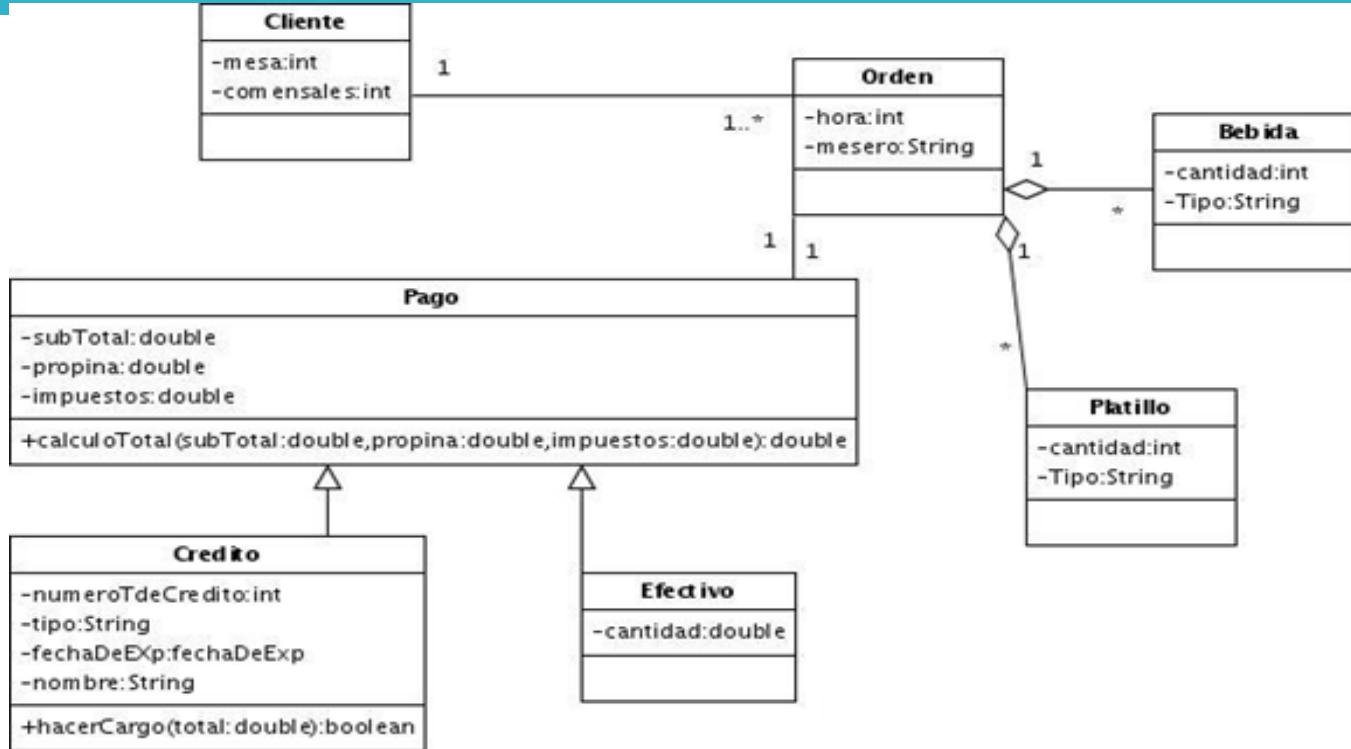


Diagrama de transición de estados

Diagrama de transición de estados

93

- Describe el ciclo de vida completo de un objeto, con los estados válidos que puede tener y las acciones que pueden ocurrir durante su vida.

- Estado inicial



- Estado intermedio

Pagado

- Estado Final



- Transición

[precondition] event / [postcondition]



Ejemplo

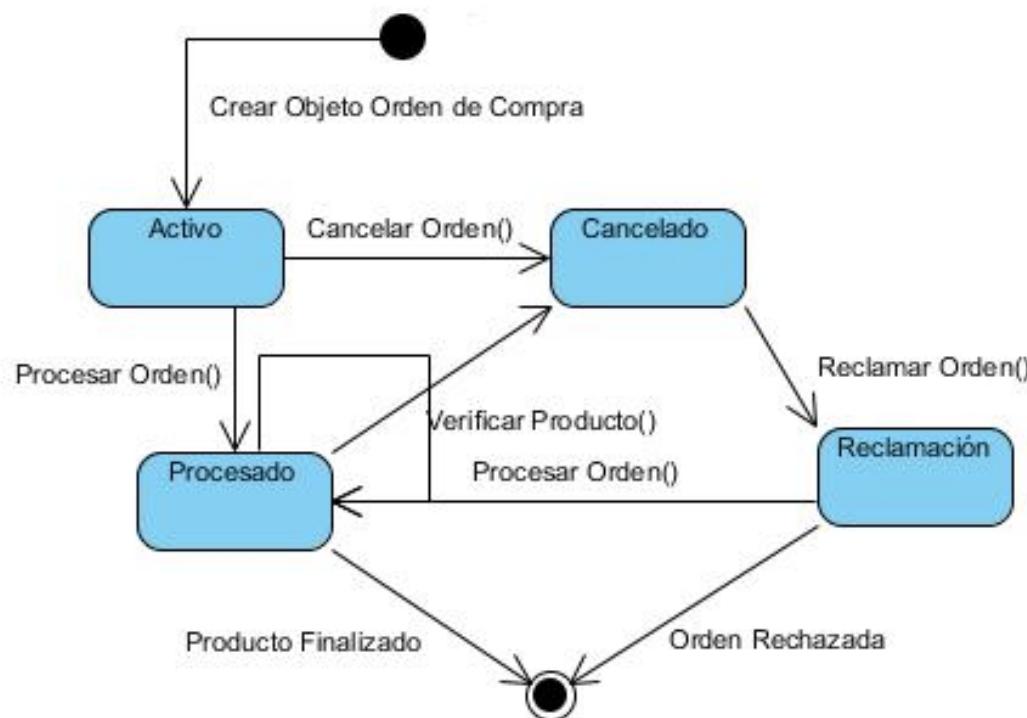


Diagrama de componentes

Diagrama de componentes

96

- Son utilizados para modelar la arquitectura de un sistema.
- Un componente representa una parte modular del sistema, que encapsula el estado y el comportamiento de un conjunto de clases.
- Un componente se comunica con otros componentes mediante dependencias e interfaces.

Diagrama de componentes

97

- Un componente con dos interfaces entregadas y 3 interfaces requeridas.

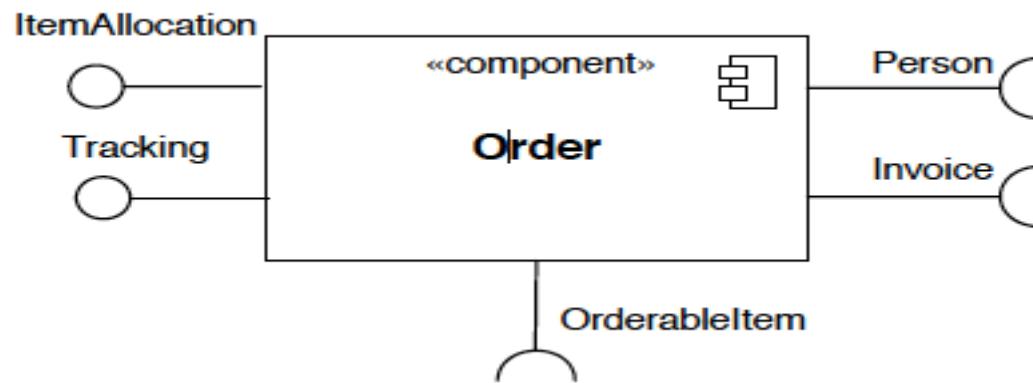


Diagrama de componentes

98

□ Representación interna de un componente

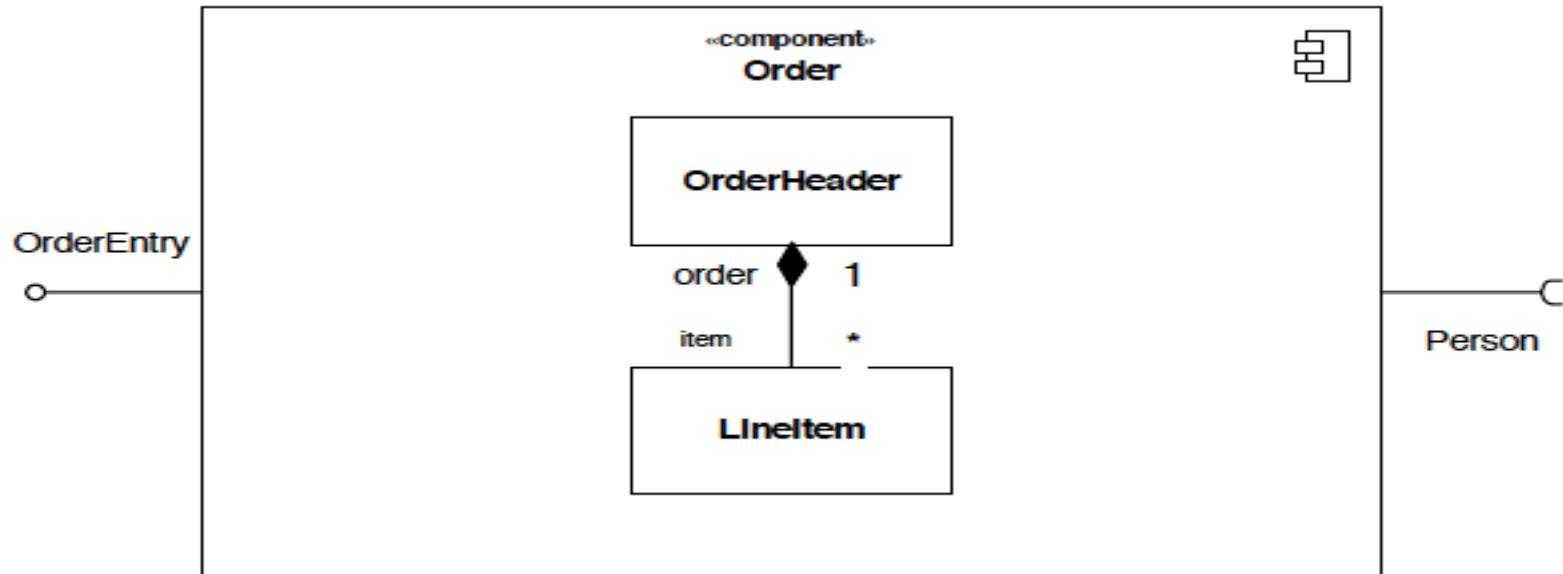


Diagrama de componentes

99

□ Dependencias entre componentes

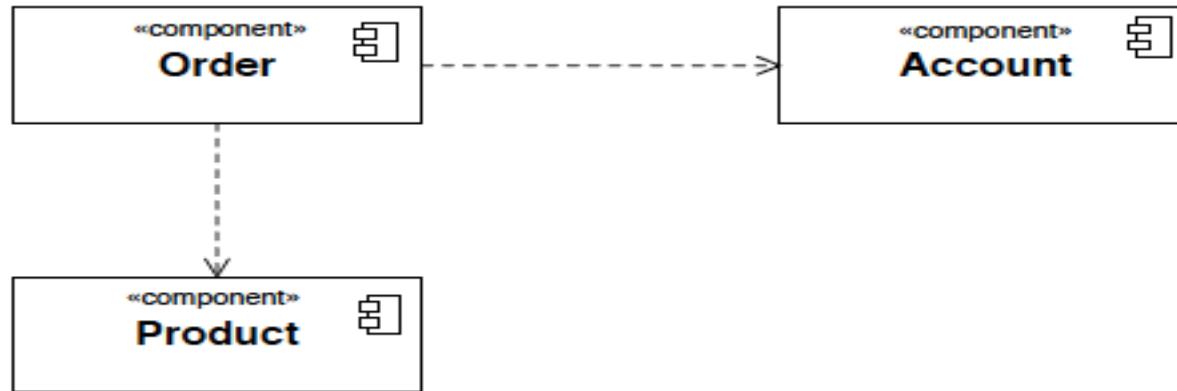


Diagrama de despliegue

Diagrama de despliegue

101

- Se utiliza para representar cómo serán distribuidos los componentes en la arquitectura del software.
- Se deben especificar los artefactos, que son piezas de información físicas relacionadas a los componentes de la arquitectura.
- Además, es necesario especificar los nodos físicos que conformarán la arquitectura.

Diagrama de despliegue

102

□ Notación para artefacto.

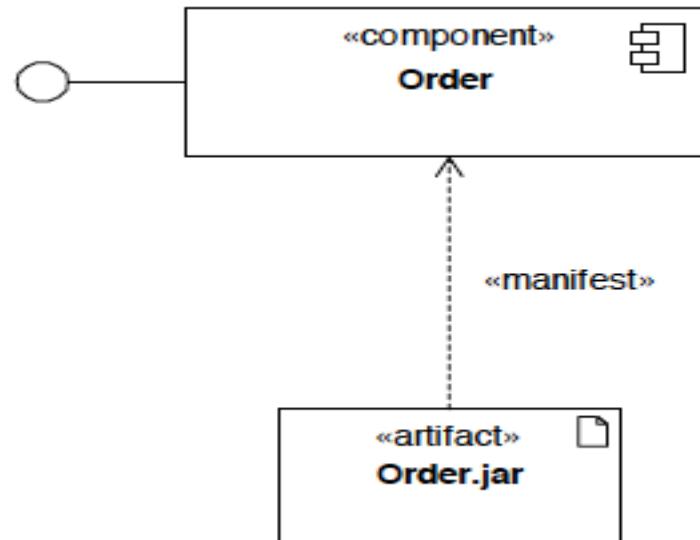
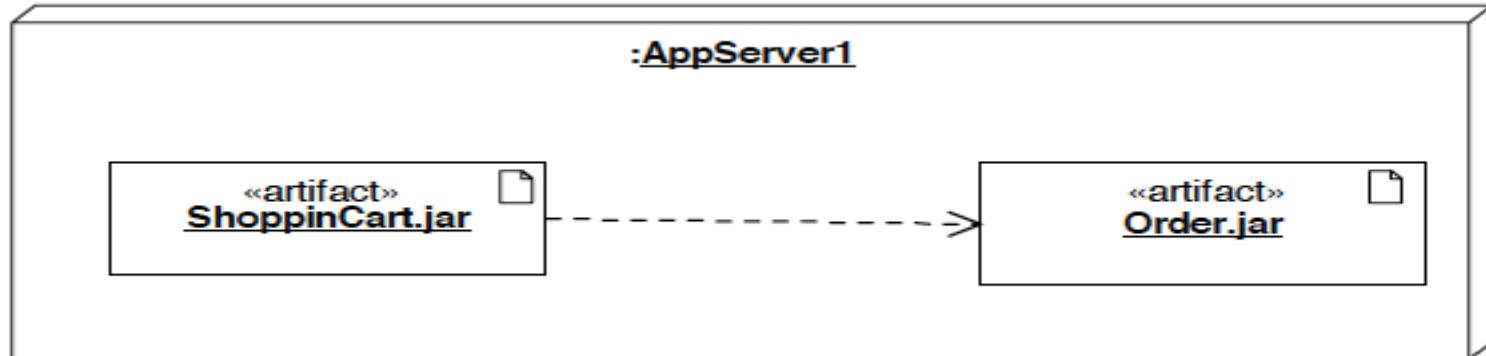


Diagrama de despliegue

103

□ Notación para nodo.



Referencias

- Clases ingeniería de software – Beatriz Marín – UDP
- Unified Modeling Language Reference Manual, the 2nd Edition, 2004, OMG