

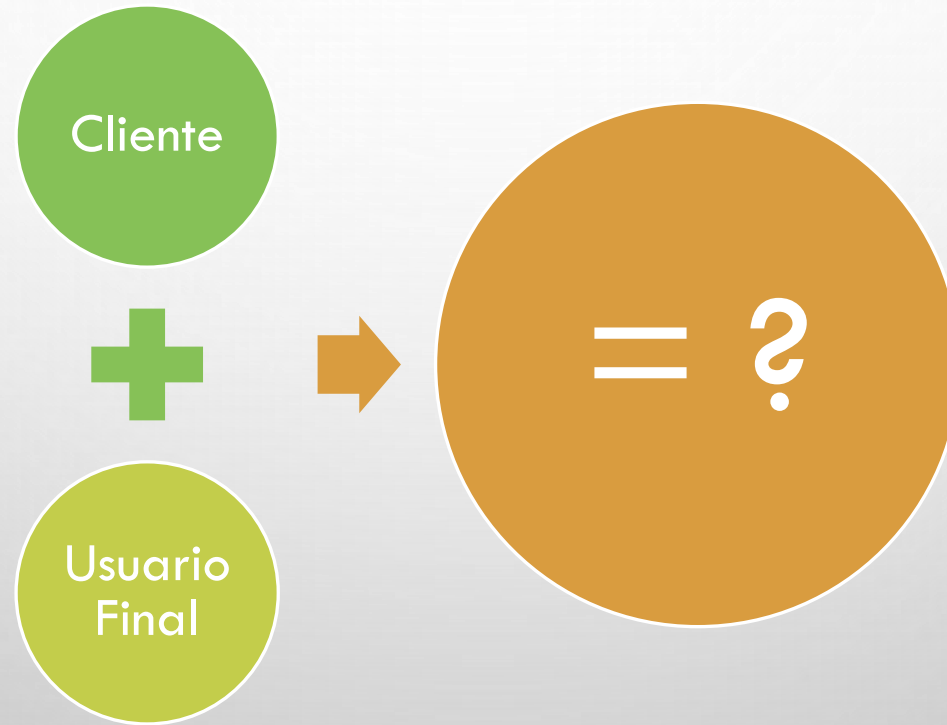


# SISTEMAS DE INFORMACIÓN

UNIDAD II

INGENIERÍA DE SOFTWARE I

# CLIENTE VS USUARIO FINAL



# CLIENTE VS USUARIO FINAL

## CLIENTE = USUARIO FINAL

LOS CLIENTES Y LOS USUARIOS FINALES SON QUIENES TIENEN EL EFECTO MÁS SIGNIFICATIVO EN EL TRABAJO TÉCNICO QUE SE DESARROLLARÁ. EN CIERTOS CASOS.

EL CLIENTE Y EL USUARIO FINAL SON LA MISMA PERSONA, PERO PARA MUCHOS PROYECTOS SON INDIVIDUOS DISTINTOS QUE TRABAJAN PARA DIFERENTES GERENTES EN DISTINTAS ORGANIZACIONES DE NEGOCIOS.

## CLIENTE

UN CLIENTE ES LA PERSONA O GRUPO QUE:

1. SOLICITÓ ORIGINALMENTE QUE SE CONSTRUYERA EL SOFTWARE,
2. DEFINE LOS OBJETIVOS GENERALES DEL NEGOCIO PARA EL SOFTWARE,
3. PROPORCIONA LOS REQUERIMIENTOS BÁSICOS DEL PRODUCTO Y
4. COORDINA LA OBTENCIÓN DE FONDOS PARA EL PROYECTO.

## USUARIO FINAL

ES LA PERSONA O GRUPO QUE:

- A. USARÁ EN REALIDAD EL SOFTWARE QUE SE ELABORE PARA LOGRAR ALGÚN PROPÓSITO DEL NEGOCIO Y
- B. DEFINIRÁ LOS DETALLES DE OPERACIÓN DEL SOFTWARE DE MODO QUE SE ALCANCE EL PROPÓSITO DEL NEGOCIO.

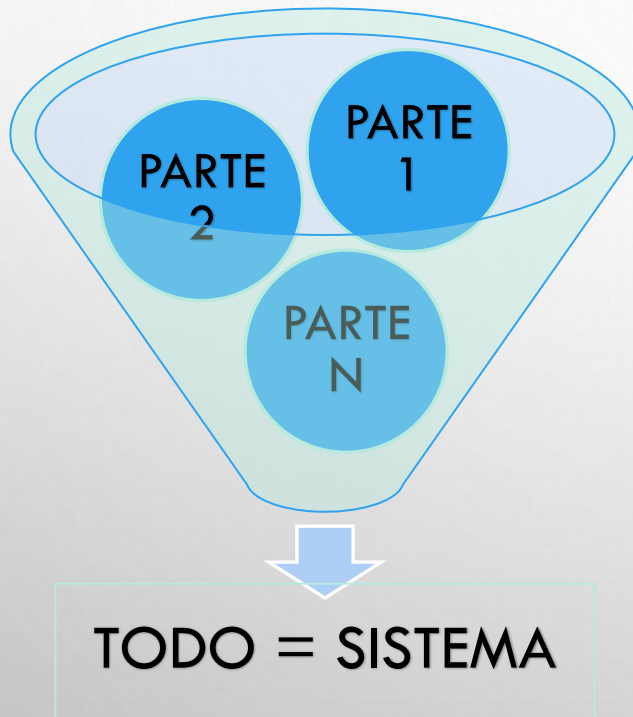
# SISTEMAS DE INFORMACIÓN (SI)

¿ Qué es  
Sistemas?

¿ Qué es  
información?

¿ Qué es el  
sistema de  
información?

# SISTEMAS DE INFORMACIÓN (SI)



- ES UNA ENTIDAD COMPUESTA POR COMPONENTES INTERCONECTADOS QUE TRABAJAN JUNTOS PARA LOGRAR UN OBJETIVO O UNA FUNCIÓN ESPECÍFICA.
- UN CONJUNTO DE ELEMENTOS INTERDEPENDIENTES Y COORDINADOS QUE TRABAJAN EN SINERGIA PARA ALCANZAR UN OBJETIVO COMÚN.

# SISTEMAS DE INFORMACIÓN (SI)

## INFORMACIÓN

- **UN CONJUNTO ORDENADO DE DATOS QUE INTERACTÚAN CON UN OBJETIVO COMÚN.**
- LOS SI SON FUNDAMENTALES PARA LA ADMINISTRACIÓN, RECOLECCIÓN, RECUPERACIÓN, PROCESAMIENTO, ALMACENAMIENTO Y DISTRIBUCIÓN DE DATOS RELEVANTES PARA LOS PROCESOS Y PARTICULARIDADES DE CADA ENTIDAD.
- LA INFORMACIÓN PUEDE SER MANUAL O AUTOMATIZADA Y SE UTILIZA PARA APOYAR LA TOMA DE DECISIONES, LA COORDINACIÓN, EL CONTROL, EL ANÁLISIS Y LA VISUALIZACIÓN DENTRO DE UNA ORGANIZACIÓN.

# SISTEMAS DE INFORMACIÓN (SI)

- SI ES UNA ESTRUCTURA ORGANIZADA QUE SE ENCARGA DE RECOLECTAR, ALMACENAR, PROCESAR Y DISTRIBUIR DATOS CON EL OBJETIVO DE GENERAR INFORMACIÓN SIGNIFICATIVA Y ÚTIL PARA LA TOMA DE DECISIONES Y EL FUNCIONAMIENTO EFICIENTE DE UNA ORGANIZACIÓN.
- ESTOS SISTEMAS ESTÁN COMPUESTOS POR UNA SERIE DE ELEMENTOS INTERRELACIONADOS QUE TRABAJAN EN CONJUNTO PARA GESTIONAR LA INFORMACIÓN DE MANERA COHERENTE Y PROVECHOSA.

# SISTEMAS DE INFORMACIÓN (SI)

- EL PROPÓSITO FUNDAMENTAL DE UN SISTEMA DE INFORMACIÓN ES **FACILITAR EL FLUJO DE DATOS, ASEGURAR SU INTEGRIDAD, DISPONIBILIDAD Y CONFIDENCIALIDAD**, ASÍ COMO **TRANSFORMAR ESOS DATOS EN INFORMACIÓN** RELEVANTE Y ÚTIL PARA LOS USUARIOS FINALES, YA SEAN GERENTES, EMPLEADOS, CLIENTES U OTROS ACTORES INVOLUCRADOS EN LA ORGANIZACIÓN.



# TEORÍA GENERAL DE SISTEMAS (TGS)

- **ES UN ENFOQUE INTERDISCIPLINARIO QUE BUSCA COMPRENDER Y ANALIZAR SISTEMAS COMPLEJOS, DESTACANDO LA INTERRELACIÓN ENTRE SUS COMPONENTES Y SU ENTORNO.**
- TGS SE REMONTA A LA DÉCADA DE 1920, CUANDO EL BIÓLOGO ALEMÁN LUDWIG VON BERTALANFFY COMENZÓ A DESARROLLAR ESTA DISCIPLINA.
- VON BERTALANFFY SE DIO CUENTA DE QUE MUCHAS DE LAS TEORÍAS EXISTENTES EN ESE MOMENTO NO ERAN SUFICIENTES PARA COMPRENDER LA COMPLEJIDAD DE LOS SISTEMAS VIVOS Y PROPUSO UN ENFOQUE INTERDISCIPLINARIO QUE PERMITIERA UN ANÁLISIS MÁS COMPLETO Y HOLÍSTICO DE LOS SISTEMAS.

# TGS: PRINCIPIOS FUNDAMENTALES

- **PRINCIPIO DE TOTALIDAD:** UN SISTEMA DEBE SER ANALIZADO EN SU TOTALIDAD, CONSIDERANDO LAS INTERACCIONES ENTRE SUS DIFERENTES ELEMENTOS.
- **PRINCIPIO DE JERARQUÍA:** LOS SISTEMAS ESTÁN ORGANIZADOS EN DIFERENTES NIVELES JERÁRQUICOS, DESDE LOS ELEMENTOS INDIVIDUALES HASTA EL SISTEMA COMPLETO.
- **PRINCIPIO DE EQUIFINALIDAD:** UN SISTEMA PUEDE ALCANZAR EL MISMO ESTADO FINAL A PARTIR DE DIFERENTES CONDICIONES INICIALES O MEDIANTE DIFERENTES PROCESOS.
- **PRINCIPIO DE RETROALIMENTACIÓN:** SE REFIERE A LA RETROALIMENTACIÓN QUE OCURRE ENTRE LOS DIFERENTES ELEMENTOS DE UN SISTEMA, LO QUE PUEDE INFLUIR EN SU COMPORTAMIENTO Y FUNCIONAMIENTO.
- **PRINCIPIO DE ENTROPÍA:** LOS SISTEMAS TIENDEN A DESORDENARSE CON EL TIEMPO, A MENOS QUE SE LES APLIQUE ENERGÍA O SE REALICE ALGÚN TIPO DE MANTENIMIENTO.

# MODELO DE REQUERIMIENTOS

- LA INGENIERÍA DE SOFTWARE COMIENZA CON UNA SERIE DE TAREAS DE MODELADO QUE CONDUCEN A LA ESPECIFICACIÓN DE LOS REQUERIMIENTOS Y A LA REPRESENTACIÓN DE UN DISEÑO DEL SOFTWARE QUE SE VA A ELABORAR.
- EL MODELO DE REQUERIMIENTOS —UN CONJUNTO DE MODELOS, EN REALIDAD— ES LA PRIMERA **REPRESENTACIÓN TÉCNICA DE UN SISTEMA.**

# MODELO DE REQUERIMIENTOS

- EL MODELADO DE LOS REQUERIMIENTOS UTILIZA UNA COMBINACIÓN DE TEXTO Y DIAGRAMAS PARA ILUSTRARLOS EN FORMA QUE SEA RELATIVAMENTE FÁCIL DE ENTENDER Y, MÁS IMPORTANTE, DE REVISAR PARA CORREGIR, COMPLETAR Y HACER CONGRUENTE.

“LOS REQUERIMIENTOS **NO** SON ARQUITECTURA. NO SON DISEÑO NI LA INTERFAZ DE USUARIO. LOS REQUERIMIENTOS SON LAS **NECESIDADES**.” ANDREW HUNT Y DAVID THOMAS

# MODELADO DE REQUIRIMIENTOS

- TAMBIÉN LLAMADO “ANÁLISIS ESTRUCTURADO”

Considera los datos y los procesos que los transforma son entidades separadas:

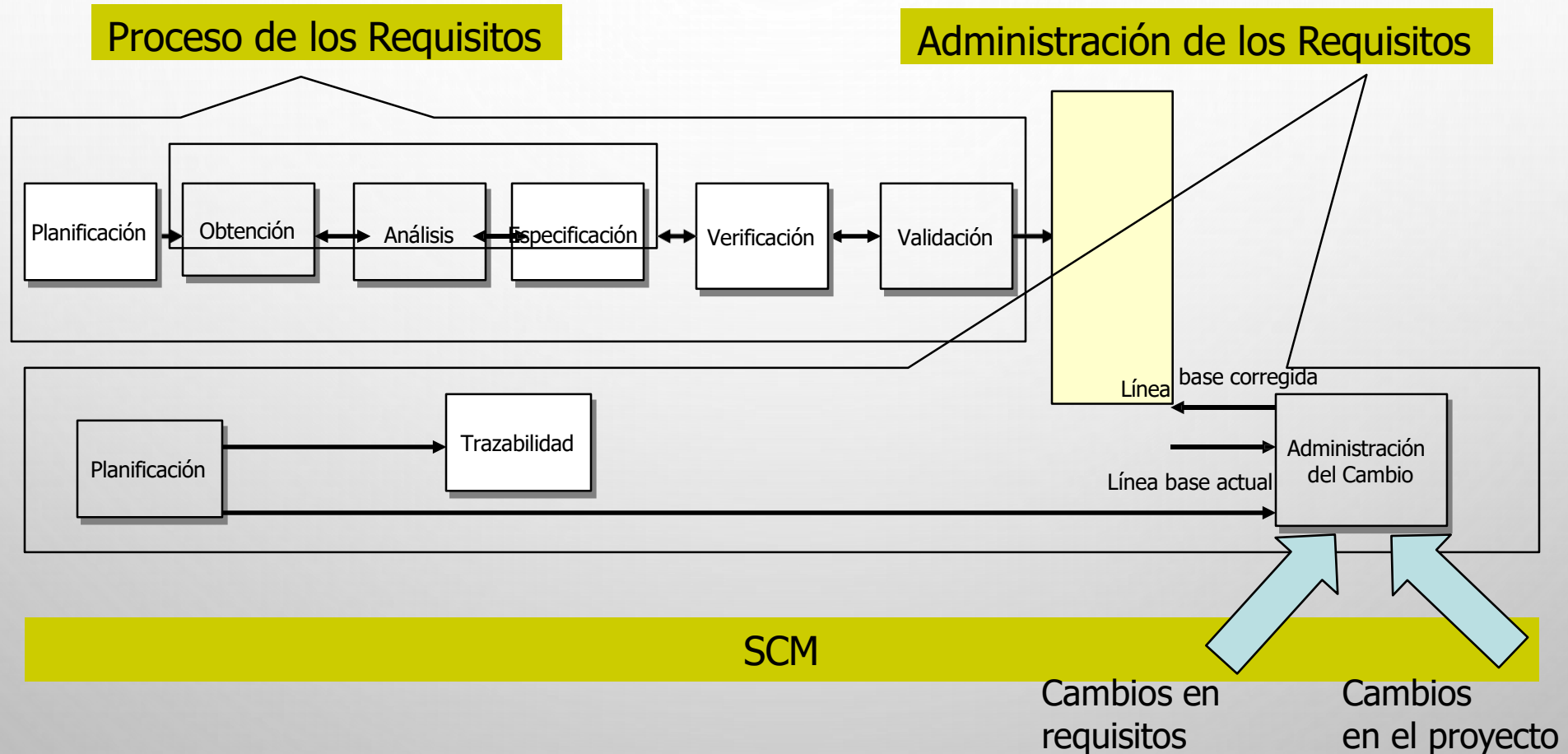


Las entidades datos se modelan de modo que se definan sus atributos y relaciones.



Proceso que manipulan a los objetos datos se modelan en forma que muestre **CÓMO** transforman a los datos a medida que los objetos que se corresponden con ellos fluyen por el sistema.

# MODELADO DE REQUERIMIENTOS

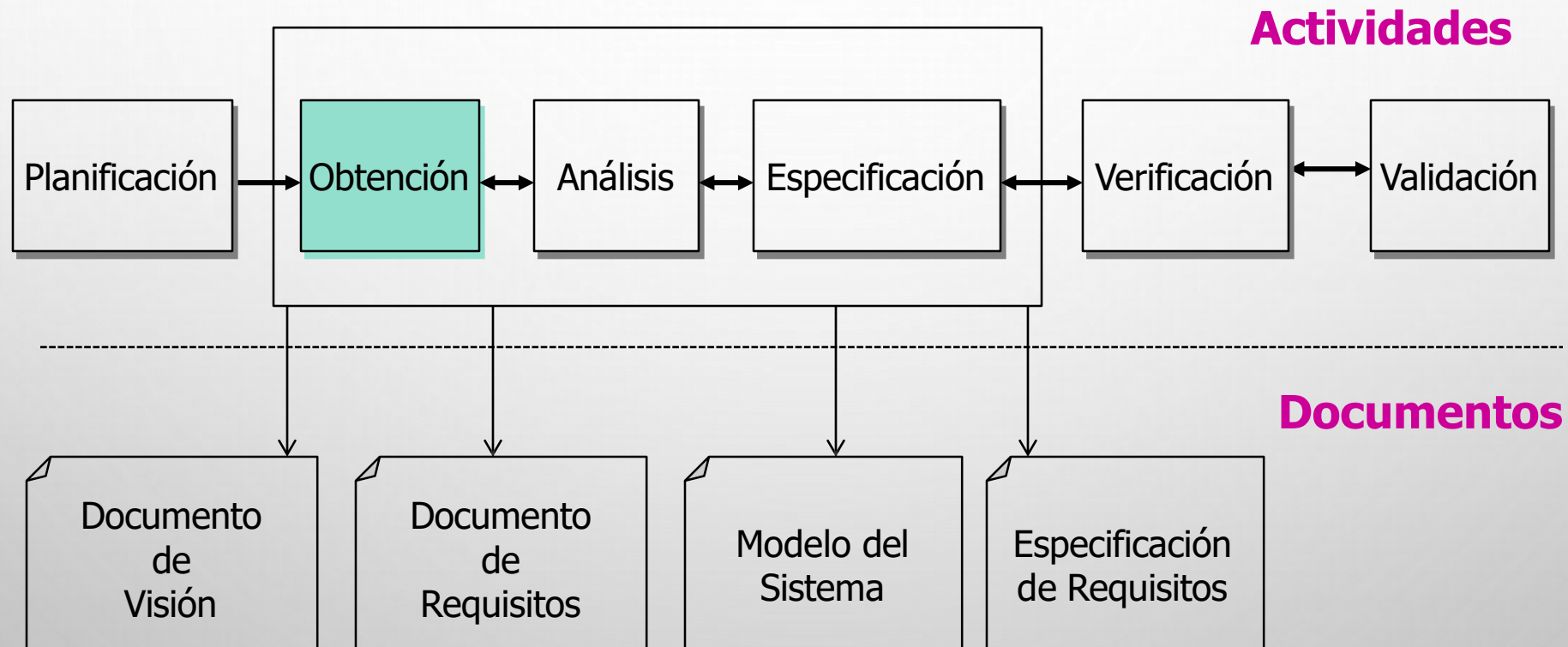


# PARTICIPANTES PROCESO DE REQUIMIENTOS

- Cliente y Usuarios
  - Requisitos adecuados a sus necesidades
- Diseñadores
  - Comprenderlos para lograr diseño que los satisfaga
- Supervisores del Contrato, sugieren:
  - Hitos de Control, cronogramas
- Gerentes del Negocio, entienden:
  - Impacto en la Organización
- Verificadores
  - Comprenderlos para poder verificar si el sistema los satisface



# PROCESO DE REQUIMIENTOS





# QUÉ RELEVAR?

- REQUERIMIENTOS DEL NEGOCIO:
  - OBJETIVOS DEL NEGOCIO DE ALTO NIVEL. RESPONDEN A LA PREGUNTA:
    - ¿CÓMO SERÁ EL MUNDO MEJOR PARA UNA DETERMINADA COMUNIDAD?
  - CATEGORÍAS:
    - BENEFICIOS FINANCIEROS
    - MEJORA DE LAS OPERACIONES DE NEGOCIO
    - POSICIONAMIENTO ESTRATÉGICO / COMPETITIVO
    - ADOPTAR UNA NUEVA LEY
    - NIVELADO / DESARROLLO TECNOLÓGICO.
  - BENEFICIOS PARA LOS USUARIOS FINALES
  - DESCRIPCIÓN DE LA FUNCIONALIDADES EN ALTO NIVEL
  - PRIORIDADES DEL PROYECTO
  - LIMITACIONES DEL PRODUCTO
- **DOCUMENTO DE VISIÓN.** SIRVE PARA ESTABLECER UNA VISIÓN COMÚN CON EL CLIENTE Y PARA DIFUSIÓN DEL PROYECTO.

# QUÉ RELEVAR?

- REGLAS DE NEGOCIO. - EXISTEN INDEPENDIENTEMENTE DEL SOFTWARE. APLICAN AUNQUE SE HAGA MANUALMENTE.
  - POLÍTICAS DE LA ORGANIZACIÓN
  - ESTÁNDARES
  - ALGORITMOS
  - LEYES Y REGULACIONES
- REQUISITOS FUNCIONALES
- REQUISITOS NO FUNCIONALES:
  - ATRIBUTOS DE Q
  - INTERFACES EXTERNAS
  - RESTRICCIONES
- CRITERIOS DE ÉXITO.
  - QUE PUEDAN DESARROLLAR BIEN TAREAS SIGNIFICATIVAS
  - MANEJO DE ERRORES
  - QUE SATISFAGA EXPECTATIVAS DE CALIDAD

# CÓMO OBTENER LOS REQUERIMIENTOS?

- INVESTIGAR ANTECEDENTES
- ENTREVISTAS INDIVIDUALES/GRUPALES
- ENCUESTAS/CUESTIONARIOS
- TORMENTA DE IDEAS
- WORKSHOP
- CASOS DE USO
- OBSERVACIÓN/PARTICIPACIÓN
- PROTOTIPADO

# INVESTIGAR ANTECEDENTES

- Estudio, muestreo, visitas,...
- Buena forma de comenzar un proyecto
- Interna: estructura de la organización, políticas y procedimientos, formularios e informes, documentación de sistemas
- Externa: publicaciones de la industria y comercio, Encuentros profesionales, visitas, literatura y presentaciones de vendedores

## ■ Ventajas

- Ahorra tiempo de otros
- Prepara para otros enfoques
- Puede llevarse a cabo fuera de la organización

## ■ Desventajas

- Perspectiva limitada
- Desactualizado
- Demasiado genérico

# ENTREVISTA INDIVIDUAL / GRUPAL

- Usar para:
  - Entender el problema de negocio
  - Entender el ambiente de operación
  - Evitar omisión de requisitos
  - Mejorar las relaciones con el cliente
- Ventajas
  - Orientado a las personas
  - Interactivo/flexible
  - Rico
- Desventajas
  - Costoso
  - Depende de las habilidades interpersonales
- Pasos para las Entrevistas
  - Seleccionar participantes
    - Aprender tanto como sea posible de antemano
  - Preparar la entrevista
    - Utilizar un patrón de estructura
  - Conducirla
    - Apertura, desarrollo, conclusión
  - Enviar un memo con resultado
  - Seguimiento



# ENCUESTA / CUESTIONARIO

- No substituye la entrevista
  - Antes de usar el enfoque:
    - Determinar la información que se precisa
    - Determinar el enfoque más adecuado:
      - Abierto, cerrado, combinado
      - Múltiple opción, valor en escala, orden relativo
    - Desarrollar cuestionario
    - Probarlo con perfil típico
    - Analizar resultado de las pruebas
  - Su principal uso es para validar asunciones y obtener datos estadísticos sobre preferencias
- |                                   |                              |
|-----------------------------------|------------------------------|
| ■ Ventajas                        | ■ Desventajas                |
| ■ Economía de escala              | ■ Menos rico                 |
| ■ Conveniente para quien contesta | ■ Problemas por no-respuesta |
| ■ Respuestas anónimas             | ■ Esfuerzo de desarrollo     |

# OBSERVACIÓN / PARTICIPACIÓN

- Poco utilizado...
- Antes de usarlo
  - Determinar información necesaria
  - Comunicar a los involucrados
  - Considerar períodos normales y atípicos
  - Planificar las anotaciones
- Ventajas
  - Confiable
  - Muy rico
  - Desarrolla empatía
- Desventajas
  - Efecto Hawthorne
  - Cuidado con generalizar demasiado (sesgo particular/local)

# TORMENTA DE IDEAS (BRAINSTORMING)

- Objetivo: Lograr consenso sobre los requisitos
- Ayuda a la participación de todos los involucrados
- Permite pensar en otras ideas
- Un secretario saca notas de todo lo discutido
- Reglas
  - No se permite criticar ni debatir
  - Dejar volar la imaginación
  - Generar tantas ideas como sea posible
  - Mutar y combinar ideas



# SESIONES DE TRABAJO (WORKSHOP)

- Ámbito para las tormentas de ideas
- Preparación
  - Venderlo a los posibles miembros de la reunión
  - Asegurarse que asisten los stakeholders correctos
  - Estructurar la invitación, el lugar, etc.
  - Enviar material previo a la reunión
    - Doc de requisitos
    - Entrevistas, defectos de los sistemas existentes, etc.
    - Asegurarse de enviar lo necesario, sin exagerar
  - Organizar la Agenda
    - Introducción
    - Tormenta de ideas – generación
    - Tormenta de ideas – reducción
    - Priorización
    - Resumen

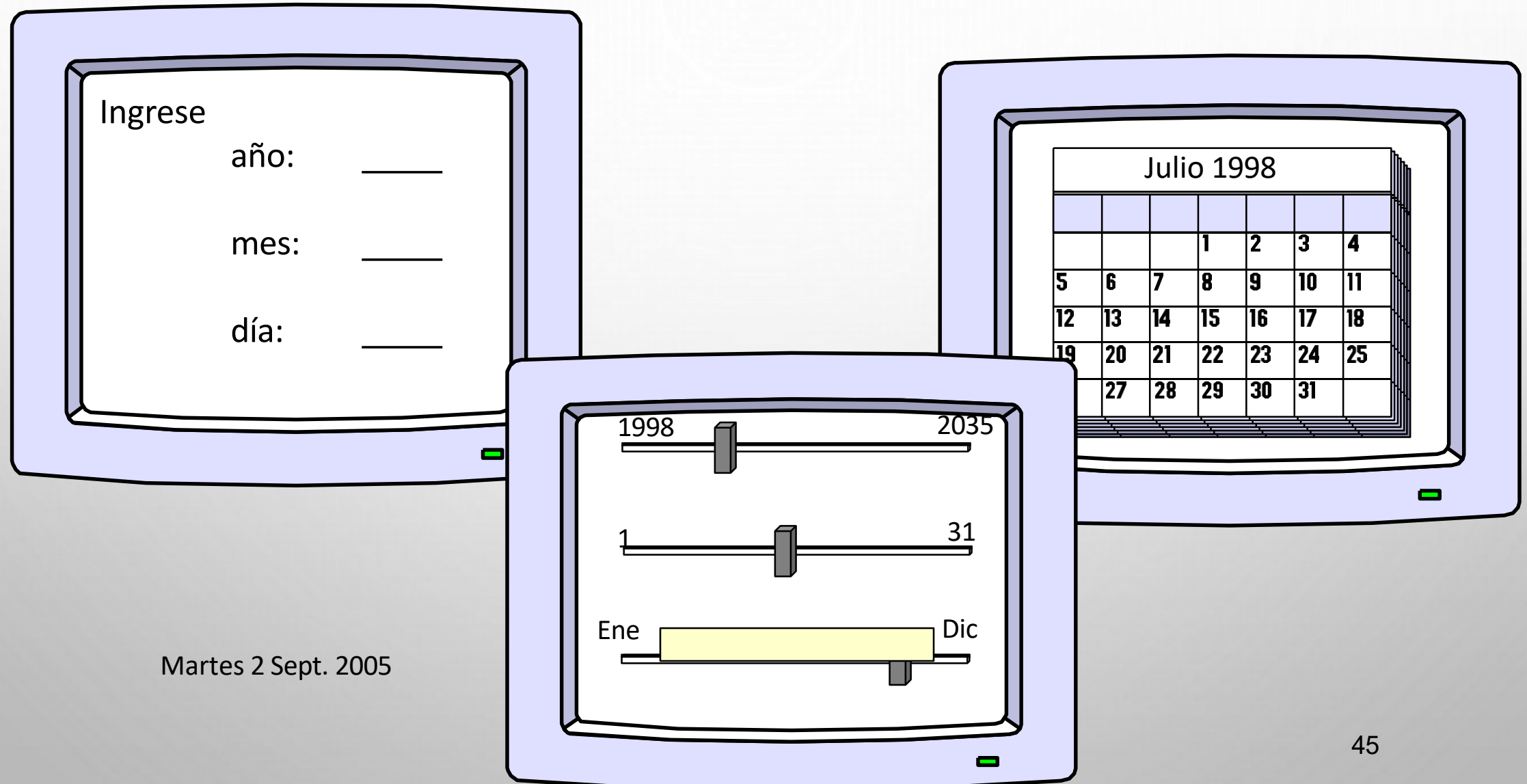
# CASOS DE USO

- Formato simple y estructurado donde los usuarios y desarrolladores pueden trabajar juntos
  - No son de gran ayuda para identificar aspectos no funcionales
  - Mientras se definen los casos de uso, puede ser un buen momento para definir pantallas u otros objetos con los que el usuario interactúa
  - Pueden ser usados en el diseño y en el testing del sistema
- 
- |   |  |
|---|--|
| <ul style="list-style-type: none"><li>■ Usarlo<ul style="list-style-type: none"><li>■ Cuando el sistema está orientado a la funcionalidad, con varios tipos de usuarios</li><li>■ Cuando la implementación se va a hacer OO y con UML</li></ul></li></ul> | <ul style="list-style-type: none"><li>■ No son la mejor elección:<ul style="list-style-type: none"><li>■ Sistemas sin usuarios y con pocas interfaces</li><li>■ Sistemas dominados primariamente por requisitos no funcionales y restricciones de diseño</li></ul></li></ul> |
|---|--|

# PROTOTIPADO

- Implementación parcial, permite a los desarrolladores y usuarios:
  - entender mejor los requisitos
  - cuáles son necesarios, deseables
  - acotar riesgos
- **Prototipo desechable**: El propósito es solo establecer que algo se puede hacer, luego se parte de cero en la construcción, quedando el conocimiento aprendido
- **Prototipo evolutivo**: Es implementado sobre la arquitectura del producto final, el sistema final se obtiene de evolucionar el prototipo
- Aspectos para los que es frecuente construir prototipos:
  - Apariencia y percepción de la interfaz de usuario
  - Arquitectura (riesgos tecnológicos, tiempos de respuesta)
  - Otros aspectos riesgosos

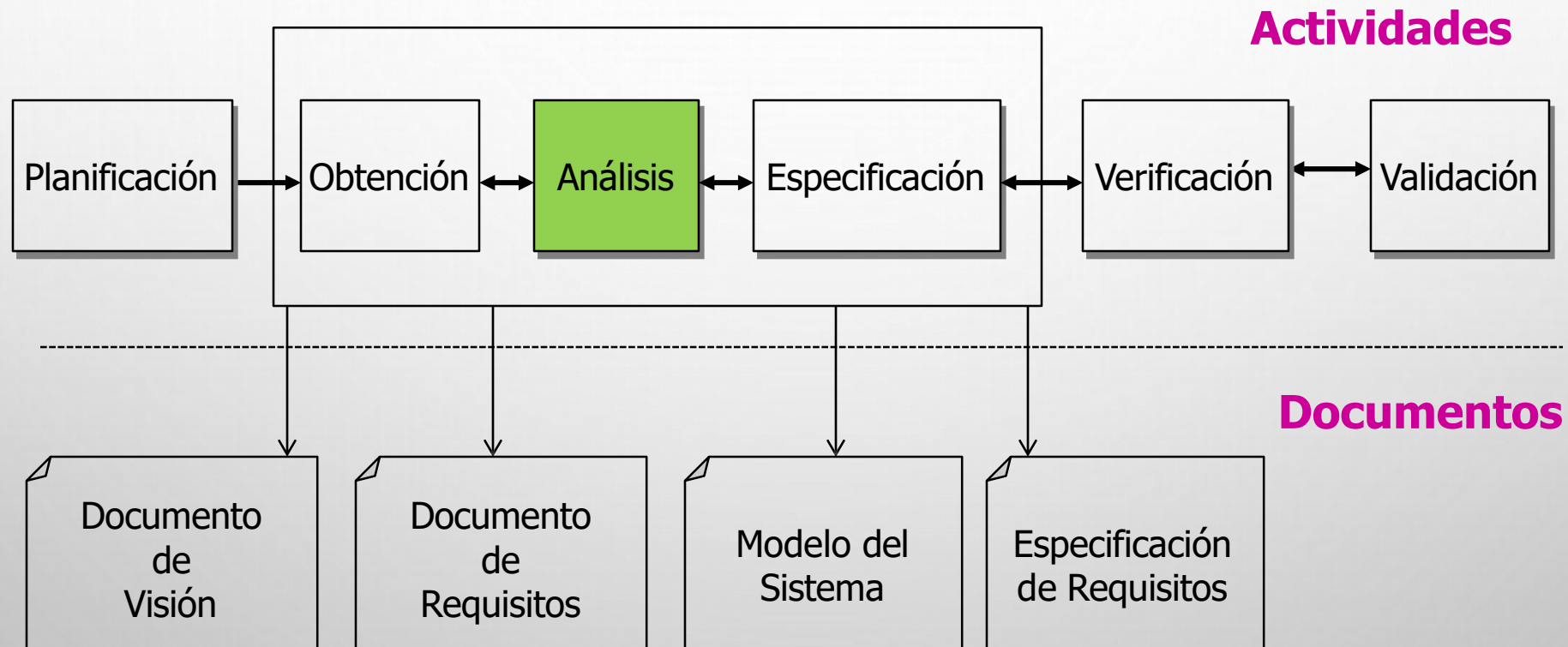
# MISMOS DATOS, PERO...



The background of the slide is a light gray gradient. In the top-left and bottom-right corners, there are several realistic-looking water droplets of various sizes, rendered with soft shadows and highlights. In the top-center, there is a faint, circular watermark logo that appears to be a stylized globe or a similar abstract design.

# ANÁLISIS DE REQUERIMIENTOS

# PROCESO DE REQUIMIENTOS

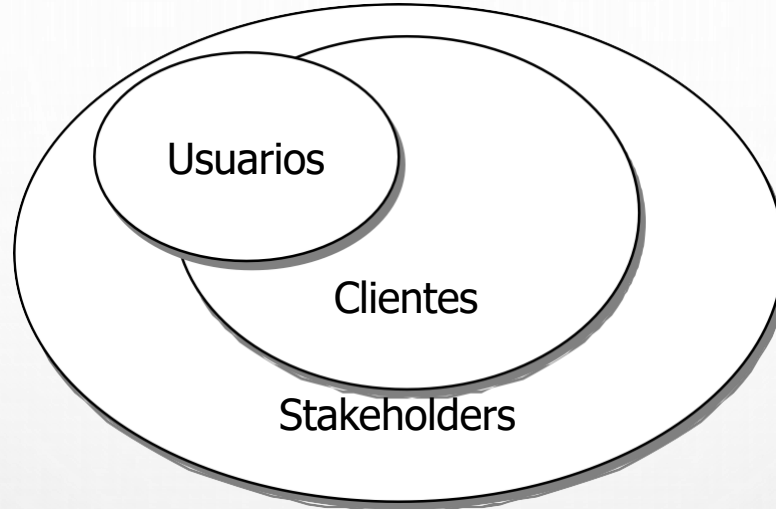


# ANÁLISIS DE REQUISITOS

- Analizar stakeholders / clientes / usuarios
- Crear vistas
- Detallar
- Negociar prioridades
- Buscar reqs que faltan
- Evaluar factibilidad técnica - Prototipos
- Evaluar riesgos de requerimientos – En el Plan



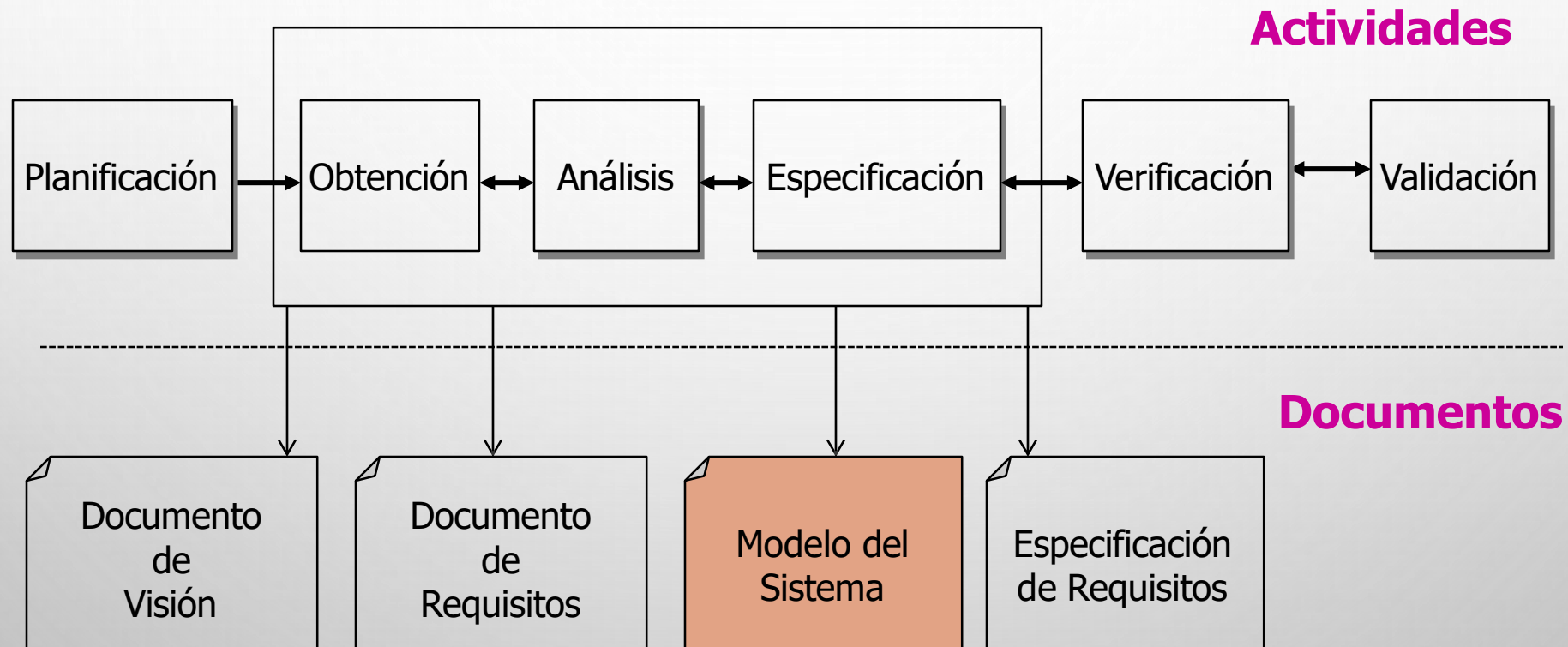
# CLIENTES / USUARIOS



- Clientes:
  - Definir responsable de:
    - resolución de conflictos
    - validación
  - Planificar reuniones de revisión de avance con el responsable.
  - Definir proceso de resolución de conflictos pe. en alcance.
- Usuarios:
  - dividirlos en clases
  - definir representantes
  - definir prototipos
  - acordar responsabilidades y estrategias de colaboración con representantes



# PROCESO DE REQUIMIENTOS



# MODELOS O VISTAS DEL SISTEMA

- Glosario
- Modelos gráficos:
  - Modelo conceptual
  - Diagramas de estado – para entidades complejas que pasen por distintos estados.
  - Diagramas de Flujo de Datos (DFD)
- Prototipos de interfaz gráfica. – Definir docs del prototipo (reqs, diseño, CP)
- Casos de Prueba
- Tablas de Decisión
- Redes de Petri
- Casos de Uso

# DIAGRAMAS

- BPM
- UML
  - Diagramas de Casos de Uso
  - Diagramas de Actividad
  - Diagramas de Estado
  - Modelo de dominio (Diagrama de Clases)
    - Modelo Conceptual


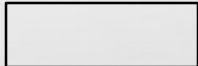
# TABLAS DE DECISIÓN

- Descripción dinámica
- Conjunto de **condiciones posibles** en un cierto instante
- **Estados** donde se verifica una combinación determinada de las condiciones
- **Acciones** a tomar

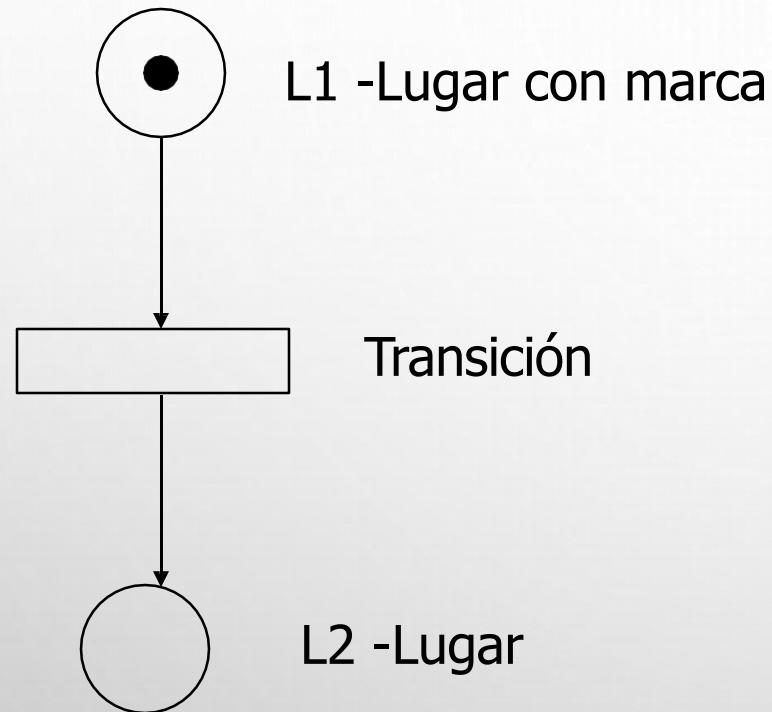
		Estados				
		1	2	3	4	5
Condiciones	Importe > 1000	F	F	V	V	V
	Buenos Antecedentes	V	F	V	V	F
	Ya operó antes	-	-	V	F	-
Acciones	Autorizar Crédito	X		X		
	Analizar antecedentes		X		X	X

- Nro estados =  $2^{\text{nro condiciones}}$  => tablas extensas

# REDES DE PETRI

- Descripción dinámica
- Permiten describir:
  - Concurrencia
  - Sincronización
- Grafo dirigido con dos tipos de nodos:
  - Lugares  y Transiciones 
  - Arcos sólo pueden unir lugares con transiciones y transiciones con lugares

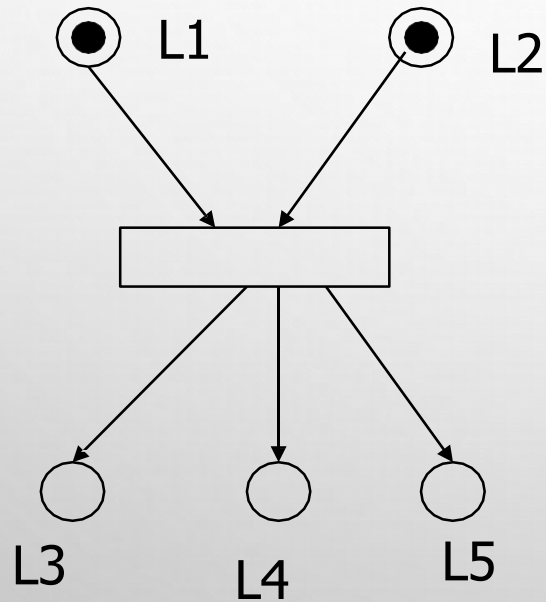
# REDES DE PETRI (2)



- El estado inicial de una red de Petri se le llama marca, está dado por los Tokens (marcas) iniciales
- Significado:
  - Transiciones: Modelan eventos o acciones
  - Lugares con marca: Cumplimiento de una condición
  - Transición activada: Ocurrencia del evento o ejecución de la acción

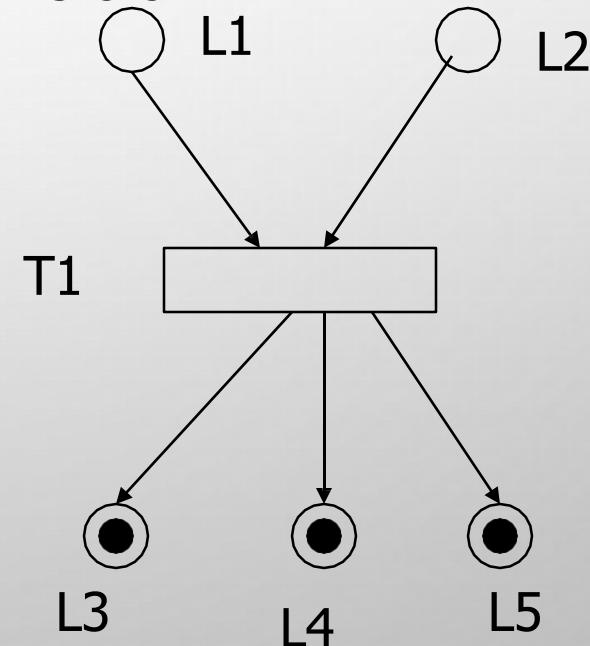
# REDES DE PETRI (3)

- Transición habilitada: Existe al menos un token en cada uno de sus lugares de entrada



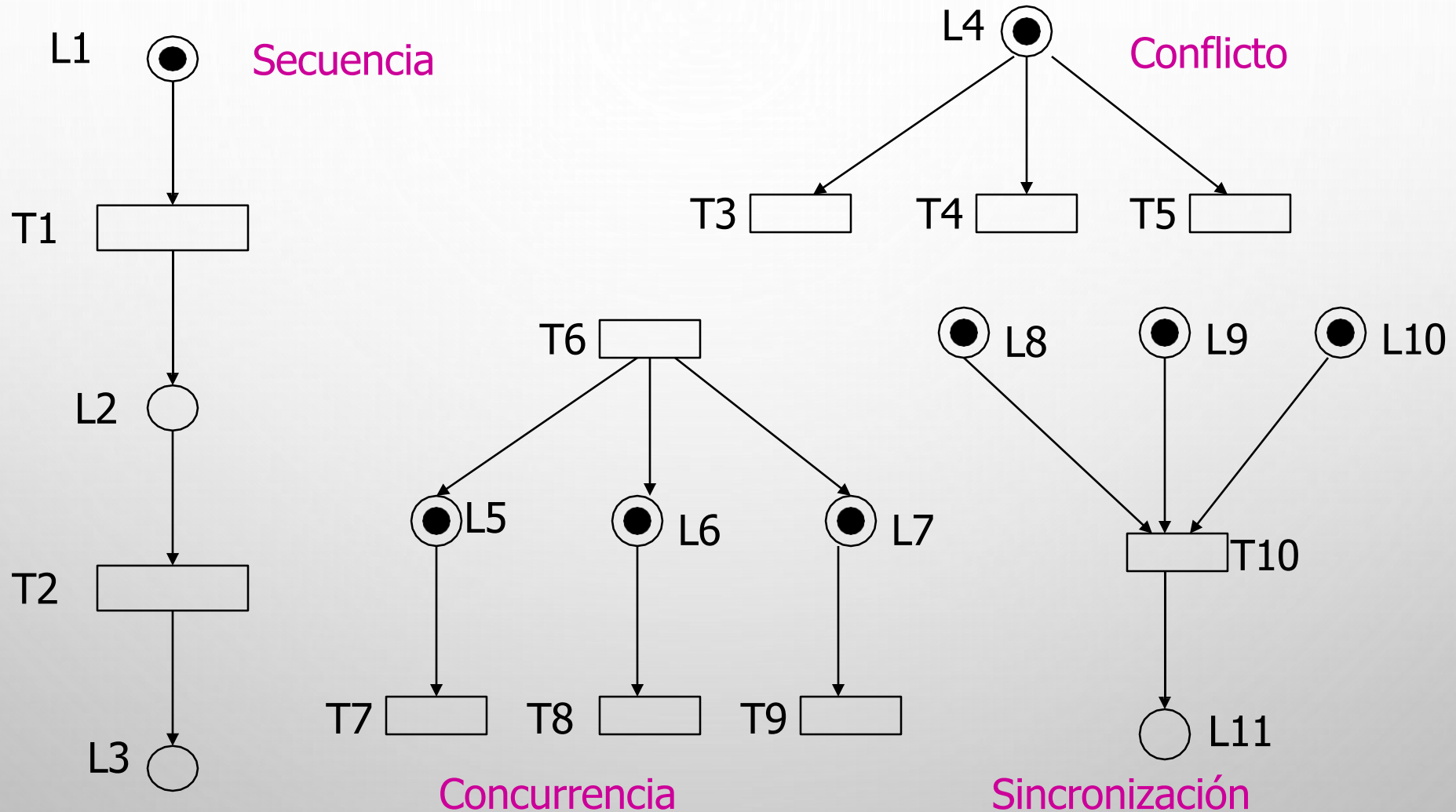
- Estado pronto para activar la transición

Al activarse una transición, los tokens que activaron la transición desaparecen de los lugares de entrada y se generan tokens en los lugares de salida de la transición



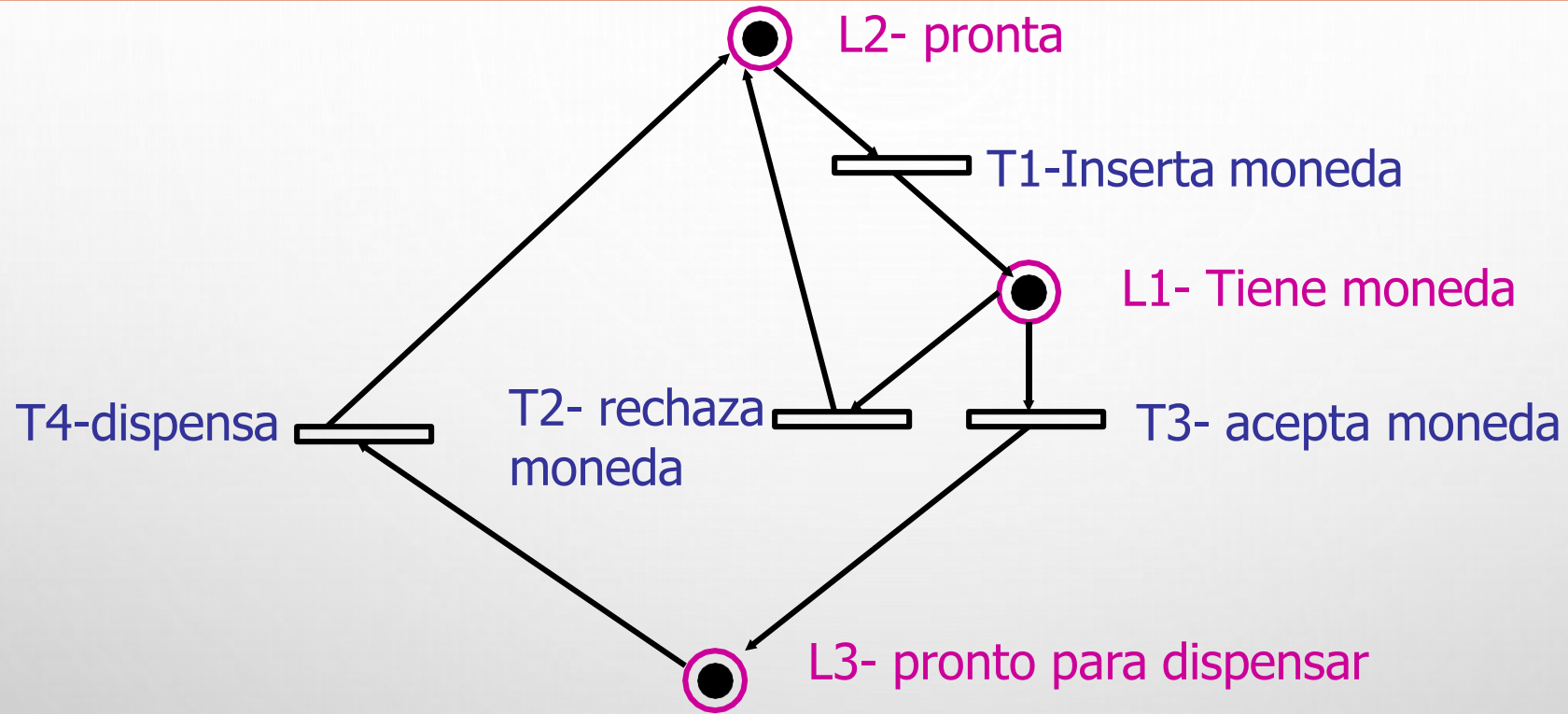


# REDES DE PETRI – ALGUNAS PRIMITIVAS





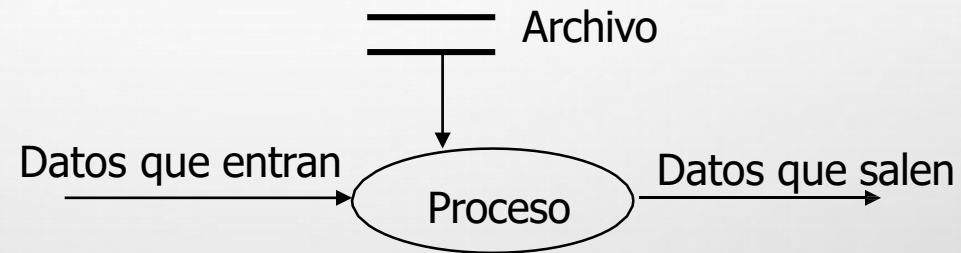
# REDES DE PETRI – EJEMPLO: MÁQUINA DISPENSADORA



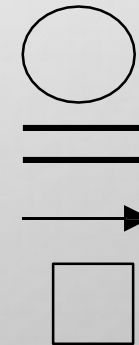
Se dispararon las transiciones: t1,t3,t4  
Otra secuencia posible seria t1,t2

# DIAGRAMA DE FLUJO DE DATOS (DFD)

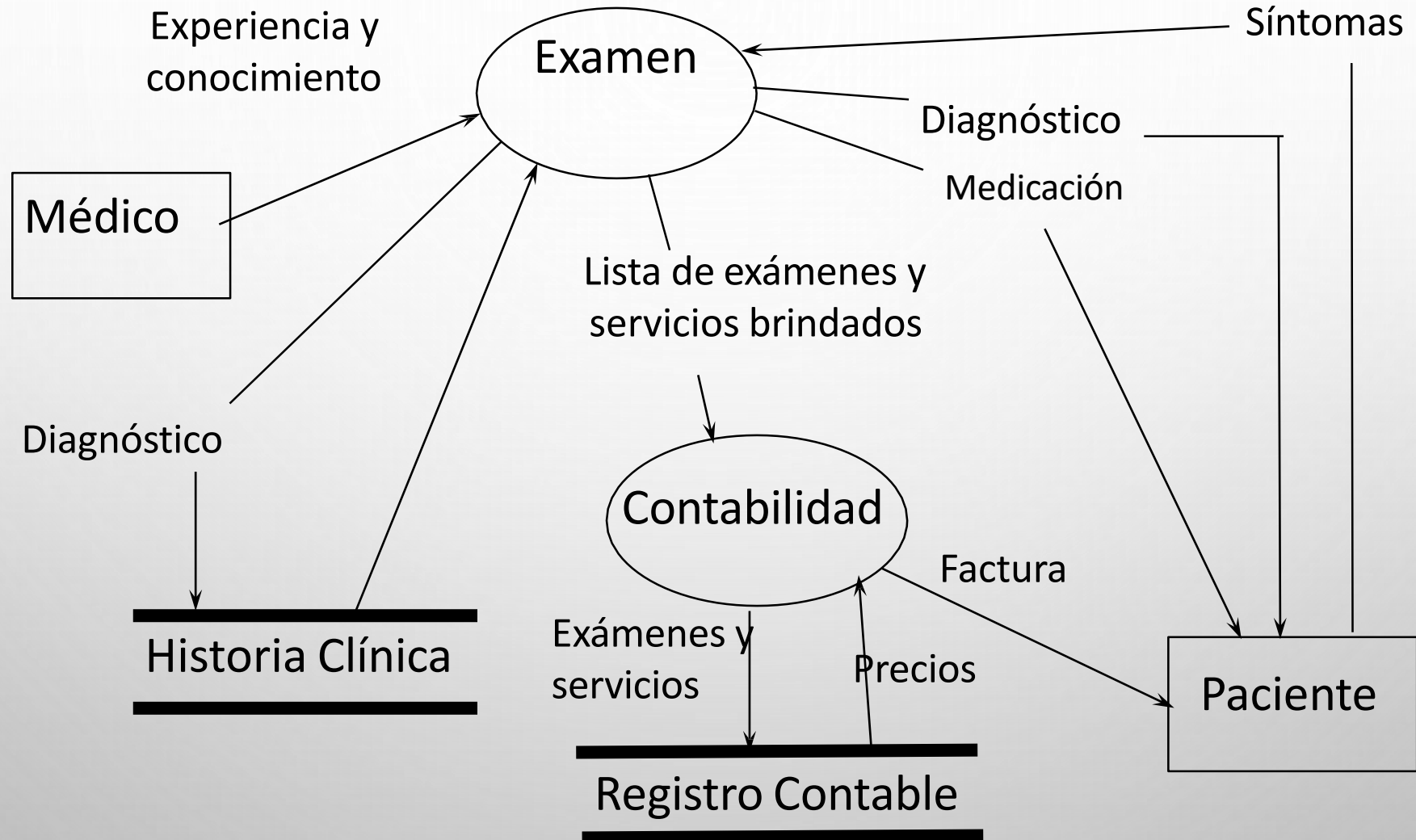
- Descripción dinámica
- Proviene de Metodología de Análisis y Diseño Estructurado
  - Fin de la década del 70
  - Usados en versión original de OMT (Rumbaugh 91), no incorporados a UML
  - Antes de los Casos de Uso era una de las formas más usadas para describir un sistema



- Elementos
  - Proceso del sistema que recibe datos y genera otros
  - Archivo de datos (almacenamiento)
  - Flujo de Datos
  - Entidad Externa al sistema a modelar (actor)



# DFD - EJEMPLO



# DFD

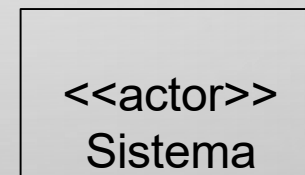
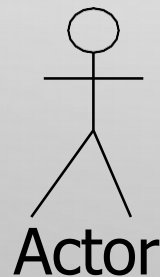
- Permite visualizar cómo fluye la información por el sistema
  - está asociado a una realización particular del sistema
- El diagrama no es suficiente para precisar el comportamiento:
  - por un flujo que entra a un proceso desde un archivo, ¿fluye un registro o todo el archivo?
  - No estipula sincronización, un flujo llega a una entidad externa y otro sale ¿Están relacionados? ¿Uno es respuesta del otro?
- No permite definir procesos (bucles, condiciones, etc.)
- Se complementa con un diccionario de datos que describe:
  - estructura de los flujos y otros detalles
  - los procesos (lenguaje natural estructurado) con lo que el comportamiento queda determinado
- Permite distintos niveles de abstracción, anidando DFDs
- A menudo sistemas legados están documentados con DFD

# CASOS DE USO

- Técnica para entender y describir requisitos
- Los casos de uso son requisitos, describen requisitos funcionales
- Pone el acento en el uso del producto
- Describen como el sistema debe comportarse desde el punto de vista del usuario
- Casos de Uso como caja negra: Especifican que es lo que el sistema debe hacer sin especificar cómo debe hacerlo
- Se describen mediante documentos de texto
- Introducido por Ivar Jacobson (1992)

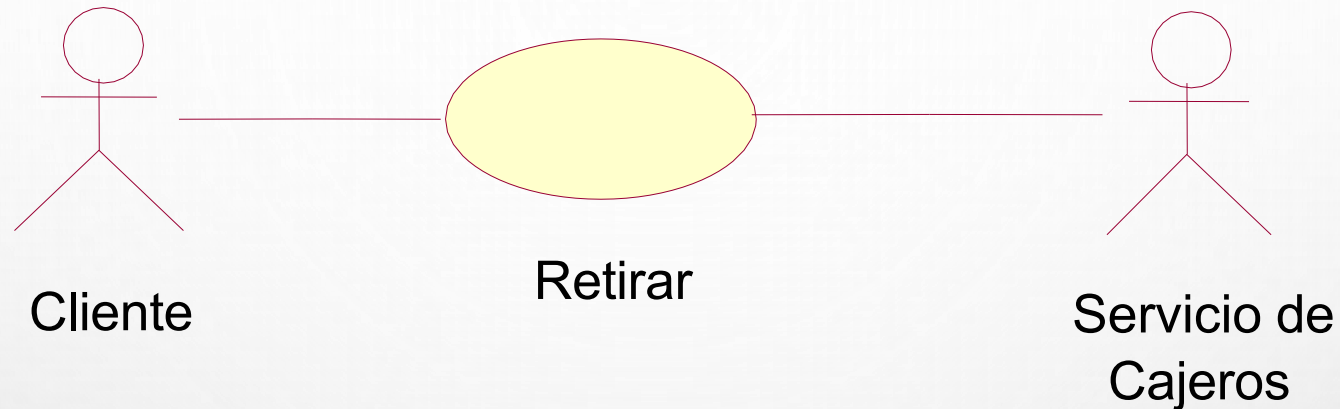
# ACTOR

- Entidad externa que interactúa con el sistema (persona identificada por un rol o sistema externo)
- Actor principal: Sus objetivos son cumplidos al realizar el caso de uso
- Los actores son externos al sistema que vamos a desarrollar.
- Al identificar actores estamos delimitando el sistema
- Usuario: persona que cuando usa el sistema, asume un rol.





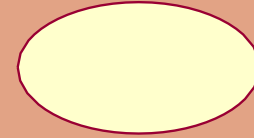
# CAJERO AUTOMÁTICO - EJEMPLO



- **Actor principal:** Cliente
- **Actores:** Servicio de Cajeros
- **Caso de Uso:** Retirar
- **Descripción:** Un cliente de un banco retira dinero de una cuenta a través del cajero automático utilizando una tarjeta bancaria, el Servicio de Cajeros verifica que el PIN sea válido y que el monto de la cuenta sea suficiente para realizar el retiro



# CASO DE USO



Caso de Uso

- **Escenario:**
  - Secuencia de acciones e interacciones entre los actores y el sistema, dando un resultado de valor observable para un actor particular
  - También se conoce como instancia de caso de uso
  - Es una forma particular de usar el sistema, un camino a través de un caso de uso.
- **Caso de uso:** conjunto de escenarios posibles que puede encarar un actor (o varios) con el sistema para el logro de cierto objetivo.
- “Un resultado observable de valor” se basa en entregar sistemas que hagan lo que las personas realmente necesitan.

# CASO DE USO: RETIRAR

## Flujo principal:

1. Cliente inserta una tarjeta bancaria en el lector del CA.
2. El CA lee el código de la tarjeta y verifica que es correcto
3. El CA pide el código de PIN de 4 dígitos
4. EL Cliente ingresa el PIN
5. El CA envía código de Tarjeta y PIN al SC
6. El SC verifica que el PIN sea correcto y contesta: OK
7. El CA despliega las distintas alternativas disponibles: retiro, depósito, consulta
8. El Cliente elige Retiro
9. El CA pide cuenta y monto
10. El Cliente los ingresa
11. CA envía código de Tarjeta, PIN, cuenta y monto al SC
12. El SC contesta: OK
13. El CA dispensa el dinero
14. El CA devuelve la tarjeta
15. El CA imprime el recibo

# CASO DE USO : RETIRAR

## Flujo principal: (otra forma)

Cliente	Sistema	Servicio de Cajeros
1. Inserta una tarjeta bancaria en el lector del CA.		
	2. Lee el código de la tarjeta y verifica que es correcto	
	3 Pide el código de PIN de 4 dígitos	
4 Ingresa el PIN		
	5 – Envía Id. De tarjeta y PIN	
		6 – Verifica que el PIN sea correcto
	7- Despliega las distintas alternativas disponibles	
8- Elige la opción: Retiro		
	9. Pide cuenta y monto	
10- Ingresa cuenta y monto		
	11. Envía al SC el Id. Tarjeta, PIN, cuenta y monto	
		12 Contesta: Continuar (OK)
	13 Dispensa el dinero	
	14 Devuelve la tarjeta	
	15 Imprime recibo	67

# CASOS DE USO

- Forma de encontrarlos: Mirar cada uno de los actores del sistema y preguntarse que es lo que buscan cuando usan el sistema.
- Recomendación: Identificarlos con un verbo.
- Cada caso de uso modela partes de la dinámica.
- Diagrama de Casos de Uso – descripción estática.
- Los casos de uso son independientes del método de diseño que se utilice, y por lo tanto del método de programación, no son parte del análisis OO, pero son una excelente entrada para ello.
- Los casos de uso pueden dirigir el proceso de desarrollo. Guían el diseño, la implementación y la prueba del sistema.

# CASOS DE USO - CONCEPTOS

- **Precondiciones:** Establece que cosas deben ser siempre verdaderas antes de comenzar un caso de uso. No se verifican dentro del caso de uso ya que se asume que son verdaderas dentro de él.
- **Poscondiciones:** Establece que cosas ocurren al completar el caso de uso.
- **Flujo principal:** Describe el escenario del caso de uso de mayor interés para el actor. Típicamente no incluye condiciones ni bifurcaciones.
- **Flujos alternativos:** Son todos los otros escenarios; son bifurcaciones en el flujo principal.
- **Requisitos Especiales:** Son los requisitos no funcionales, atributos de calidad o restricciones específicas relacionadas con el caso de uso.



# CASO DE USO : RETIRAR

## Flujos Alternativos :

### **2A. La tarjeta no es válida**

1. El CA devuelve la tarjeta con el mensaje “tarjeta no válida”
2. Fin CU

### **6A. PIN inválido y menos de 3 intentos**

El Cliente puede realizar tres intentos para ingresar el PIN válido. Sino, el CA retiene la tarjeta.

1. El SC contesta indicando PIN inválido
2. El CA muestra el mensaje “PIN incorrecto” y sigue en punto 3

### **6B. PIN inválido y 3 intentos**

El CA debe retener la tarjeta

1. El SC contesta indicando PIN inválido
2. El CA muestra el mensaje “Se le retiene la tarjeta”
3. Fin CU

### **9A. El CA no tiene dinero**

1. La opción “Retiro” en esta situación no es una alternativa posible, y el CA despliega la advertencia: “Sin dinero”.
2. Fin CU

# CASO DE USO : RETIRAR

## Flujos Alternativos (cont.):

### **11A. Monto insuficiente para el cajero**

El monto indicado por el cliente no puede obtenerse a partir de los billetes de que dispone el CA

- 1 El CA despliega el mensaje “No se cuenta con ese monto en este cajero”
- 2 Vuelve a 9.

### **12A. No hay suficiente saldo en la cuenta.**

1. CA despliega mensaje “Su saldo no permite extraer ese monto”
2. El CA devuelve la tarjeta
3. Fin CU

### **12B. No hay contacto con el Servicio de Cajeros (SC)**

1. CA despliega el mensaje “sin conexión a la red de cajeros”
- 2 . El CA devuelve la tarjeta
3. Fin CU

### **12C. Enlace con el computador central se cae durante la transacción**

Hay que asegurar que el SC considera sólo los retiros efectivamente realizados

### **14A. El dinero no es retirado de la bandeja.**

1. Si después de YY segundos el dinero está todavía en la bandeja, el CA lo recupera y lo deja en el depósito de dinero usado
1. Sigue en 14

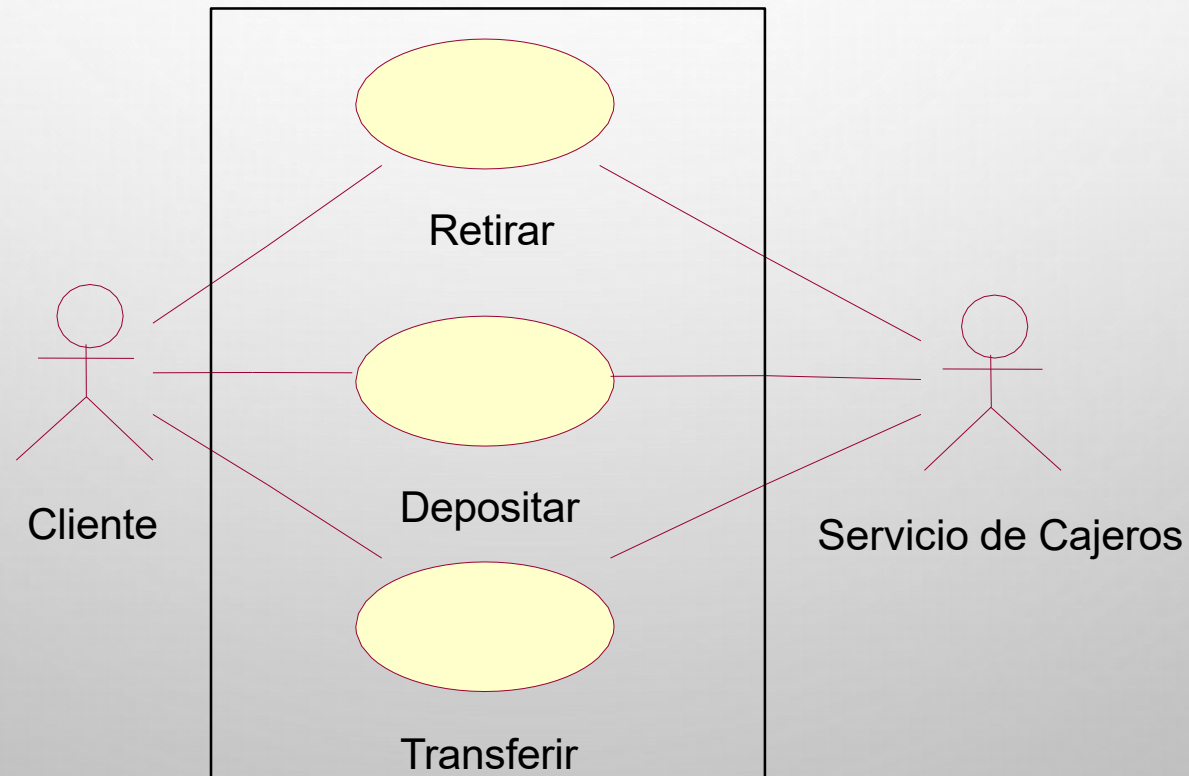
### **14B. La tarjeta se tranca al intentar devolverla.**

1. CA trata de devolverla durante xx segundos.
2. Si en ese tiempo no puede devolverla, CA avisa a mantenimiento
3. Fin CU



# DIAGRAMA DE CASOS DE USO

- UML provee notación para los casos de uso para ilustrar los actores, los casos de uso y las relaciones entre ellos
- Permite realizar un Diagrama del Contexto del Sistema
- Muestra los bordes del sistema

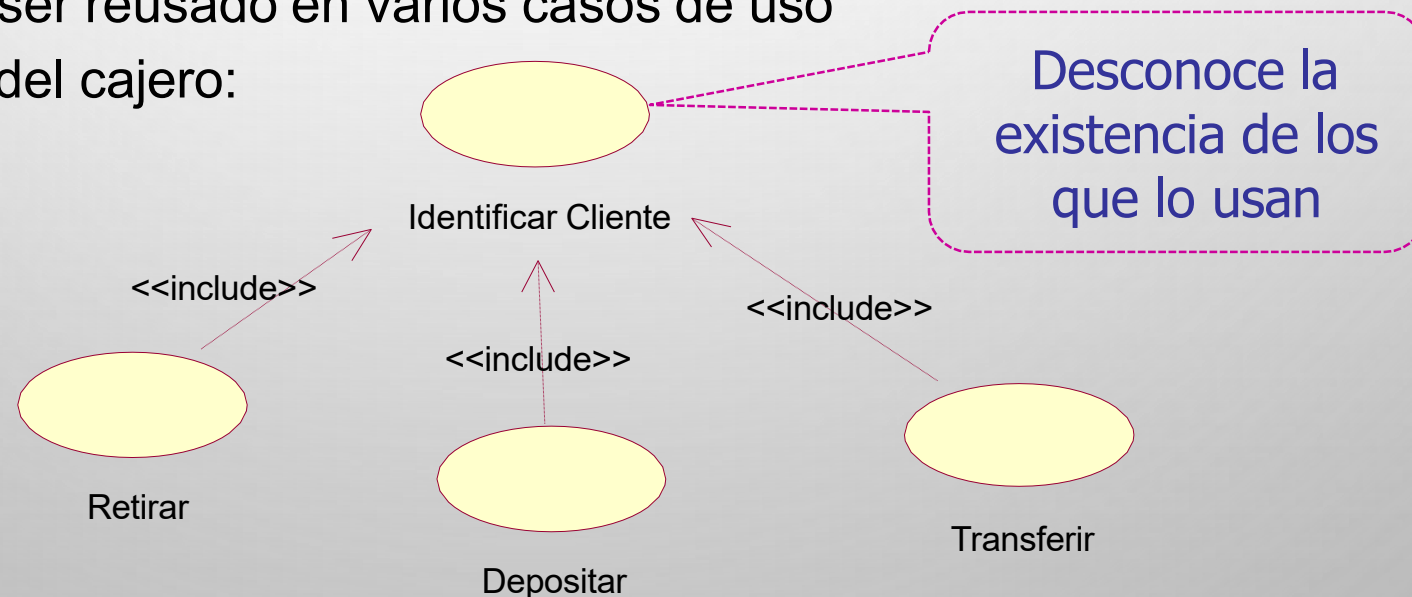


# CONSTRUCCIÓN DEL MODELO - PASOS

- DEFINIR FRONTERA
- IDENTIFICAR ACTORES
- PARA CADA ACTOR, IDENTIFICAR QUÉ COSAS QUIERE HACER
  - CADA UNO VA A DETERMINAR UN CASO DE USO
  - DARLE UN NOMBRE
- DADO UN CASO DE USO
  - IDENTIFICAR SI PARTICIPAN OTROS ACTORES
  - DESCRIBIRLO BREVEMENTE DE FORMA NARRATIVA, CENTRÁNDOSE EN EL
    - FLUJO PRINCIPAL (DISTINTAS VARIANTES DE PRESENTACIÓN Y CONTENIDO)
- UNA VEZ DEFINIDO EL CONJUNTO DE CASOS DE USO RELEVANTE:
  - REFINARLOS INCLUYENDO CONDICIONES ESPECIALES
  - IDENTIFICAR CASOS DE USO COMUNES Y PARTICULARES (“INCLUYE” Y
    - “EXTIENDE”), GENERALIZACIÓN

# RELACIONES ENTRE CU – INCLUDE

- Escenarios comunes a más de un caso de uso
- El caso de uso incluido no depende del caso de uso base
- Cuando una instancia del caso de uso «llega al lugar» donde el comportamiento de otro caso de uso debe ser incluido, ejecuta todo el comportamiento descrito por el caso de uso incluido y luego continúa de acuerdo a su caso de uso original.
- El caso de uso incluido representa comportamiento encapsulado que puede ser reusado en varios casos de uso
- En el caso del cajero:



# CASO DE USO: RETIRAR

## Flujo principal:

1. **Incluye** el caso de uso: Identificar Cliente
2. El CA despliega las distintas alternativas disponibles: retiro, depósito, consulta
3. El Cliente elige Retiro
4. El CA pide cuenta y monto
5. El Cliente los ingresa
6. CA envía código de Tarjeta, PIN, cuenta y monto al SC
7. El SC contesta: OK
8. El CA dispensa el dinero
9. El CA devuelve la tarjeta
10. El CA imprime el recibo

# CASO DE USO: IDENTIFICAR CLIENTE

## Descripción Breve:

Verifica que la tarjeta y el PIN sean válidos

## Flujo Principal:

1. Cliente inserta una tarjeta bancaria en el lector del CA.
2. El CA lee el código de la tarjeta y verifica que es correcto
3. El CA pide el código de PIN de 4 dígitos
4. EL Cliente ingresa el PIN
5. El CA envía código de Tarjeta y PIN al SC
6. El SC verifica que el PIN sea correcto y contesta: OK

## Flujos Alternativos:

### **2A. La tarjeta no es válida**

1. El CA devuelve la tarjeta con el mensaje “tarjeta no válida”
2. Fin CU

### **6A. PIN inválido y menos de 3 intentos**

El Cliente puede realizar tres intentos para ingresar el PIN válido. Sino, el CA retiene la tarjeta.

1. El SC contesta indicando PIN inválido
2. El CA muestra el mensaje “PIN incorrecto”
3. Sigue en punto 3

### **6B. PIN inválido y 3 intentos**

El CA debe retener la tarjeta

1. El SC contesta indicando PIN inválido
2. El CA muestra el mensaje “Se le retiene la tarjeta”
3. Fin CU

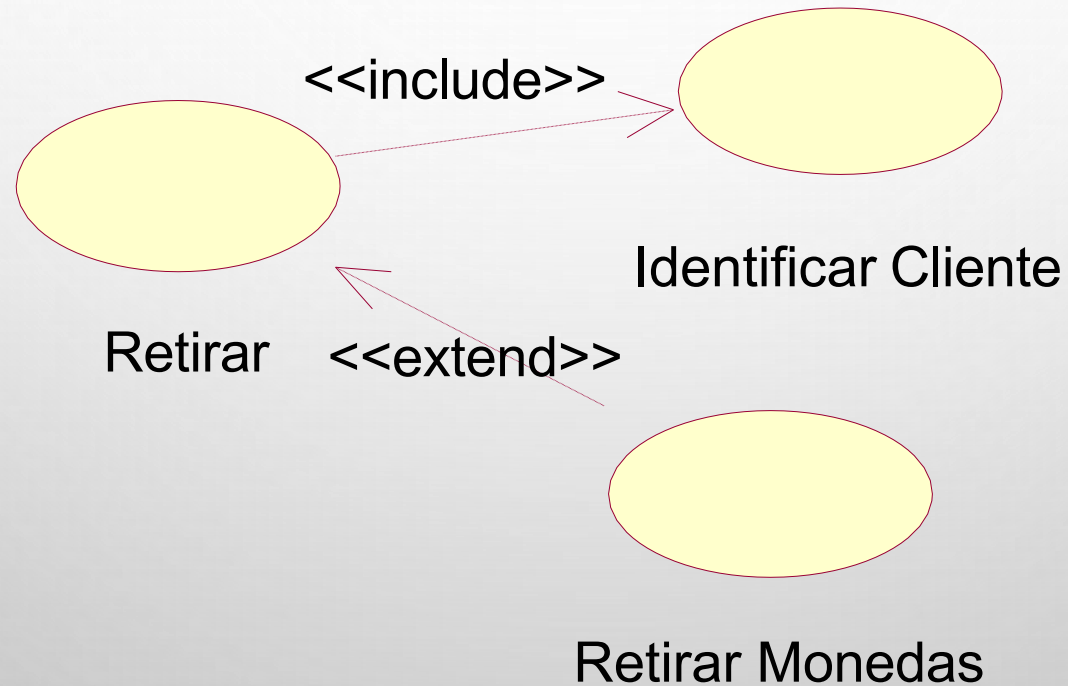


# RELACIONES ENTRE CU – EXTEND

- Es un fragmento de un caso de uso, que agrega comportamiento a otro caso de uso.
- Se usan para explicar escenarios que sería complejo presentar como flujo alternativo, o que se desea destacar.
- Representan una parte de la funcionalidad del caso que no siempre ocurre (condicional).
- Se ejecuta solo si la condición se cumple.
- El caso de uso extendido referencia a su caso de uso base.
- Punto de extensión: Punto dentro del caso de uso, donde se puede insertar comportamiento adicional.
- Al terminar el caso de uso extendido, se vuelve al caso de uso base, en la sentencia siguiente al punto de extensión.

# EXTEND - EJEMPLO

- El cliente puede querer retirar monedas además de billetes





# CASO DE USO : RETIRAR

## Flujo principal:

1. **Incluye** el caso de uso: Identificar Cliente
2. El CA despliega las distintas alternativas disponibles: retiro, depósito, consulta
3. El Cliente elige Retiro
4. El CA pide cuenta y monto
5. El Cliente los ingresa
6. CA envía código de Tarjeta, PIN, cuenta y monto al SC
7. El SC contesta: OK
8. El Cliente pide dispensar el dinero
9. El CA dispensa el dinero
10. El CA devuelve la tarjeta
11. El CA imprime el recibo

## Puntos de Extensión:

Retiro de Monedas: En el punto 8 del flujo principal

# CASO DE USO: RETIRAR MONEDAS

**Descripción Breve:** El cliente opcionalmente puede querer retirar monedas

Punto de extensión  
indicado por un nombre

## **Flujo Principal:**

Extensión de **Retirar** en el punto **Retirar Monedas**, el cliente también puede elegir “monedas”, en ese caso:

1. El Cliente elige retirar monedas, especificando tipos de monedas y la cantidad de rollos para cada uno.
2. El CA calcula el importe a retirar para cada moneda y el total y lo muestra
3. El Cliente confirma

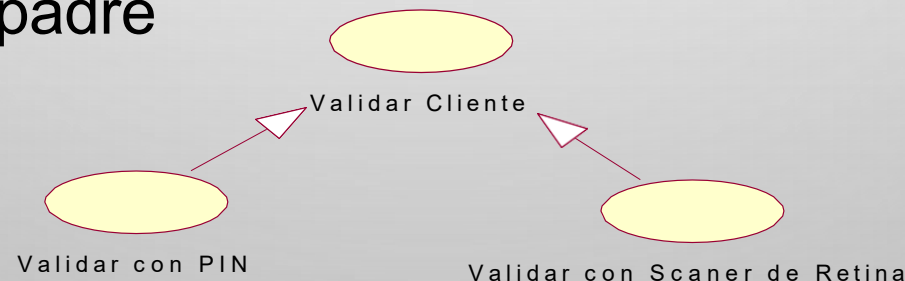
## **Flujos Alternativos:**

3A El cliente puede querer cambiar la selección, se vuelve a 1

G1 El cliente cancela el retiro de monedas. Fin CU Retirar Monedas

# RELACIONES ENTRE CU – GENERALIZACIÓN

- Algunas veces existe más de un escenario principal para un caso de uso
- Se puede crear un caso de uso abstracto, crear un caso de uso para cada escenario principal y que estos hereden del caso abstracto
- El caso de uso hijo hereda los escenarios, puntos de extensión y relaciones definidos en el caso de uso padre
- El caso de uso hijo puede definir nuevas operaciones, como también redefinir o enriquecer con nuevas secuencias de acciones operaciones ya existentes en el caso de uso padre



# ACTIVIDADES

- Encontrar actores y casos de uso
- Priorizar los casos de uso
- Detallar un caso de uso
- Estructurar el modelo de casos de uso

# CASOS DE USO - NIVEL DE DETALLE

¿A qué nivel se deben expresar los CU en el análisis de requerimientos?

- Enfocarse en CU al nivel de Proceso de Negocio Elemental (**elementary business process (EBP)**):

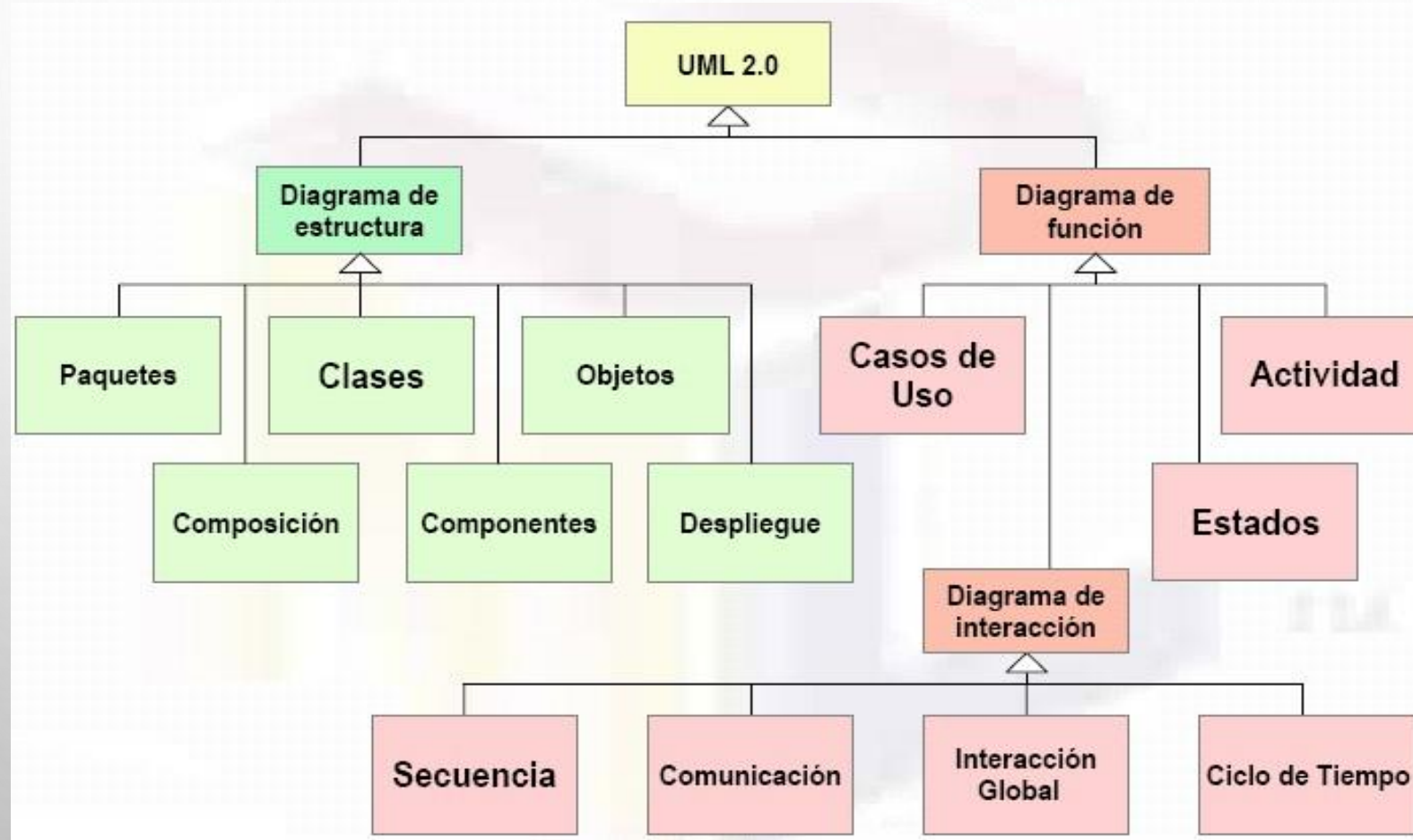
Una tarea ejecutada por una persona en un lugar en un determinado momento, en respuesta a un evento del negocio, que agrega valor de negocio medible y deja los datos en estado consistente.

- Que cumpla un objetivo del usuario.
- Error común: definir muchos CU a nivel demasiado bajo: por cada paso o subtarea dentro de un EBP.
- Excepciones: Pe. casos de uso incluidos

- Unified Modeling Language
- Objetivo: Proveer un lenguaje común que puede ser usado para el desarrollo de software
- Lenguaje que permite:
  - Visualizar: La comunicación es a través de gráficos
  - Especificar: construyendo modelos para el análisis, diseño, implementación
  - Construir: Permite la generación de código a partir de un modelo UML, y la construcción de un modelo a partir del código (ingeniería reversa)
  - Documentar: Permite la documentación completa de todo el sistema
- Aprobado como estándar por la OMG en 1997
- Actualmente se encuentra en la versión 2.5.1 (2017)



# DIAGRAMAS EN UML





# TIPOS DE DIAGRAMAS

- **Modelo Estático**

- Construye y documenta los aspectos estáticos de un sistema.
- Refleja la estructura básica y estable de un sistema software.
- Crea una representación de los principales elementos del dominio del problema

- **Modelo Dinámico**

- Crea los diagramas que muestran el comportamiento de un sistema
- Para requisitos se utilizan los siguientes diagramas:
  - Diagrama de Casos de Uso
  - Diagrama de Clases (Modelo Conceptual)
  - Diagrama de Actividad
  - Diagramas de Estados

# DIAGRAMA DE CASOS DE USO

- Permite visualizar en una forma compacta los casos de uso del sistema y que actores participan en cada caso de uso
- Presenta las relaciones que existen entre los casos de uso
- Muestra los límites del sistema
- Visión estática de los Casos de Uso de un sistema
- Consta de los siguientes elementos:
  - Actor
  - Caso de Uso
  - Relaciones
    - Include
    - Extend
    - Generalización

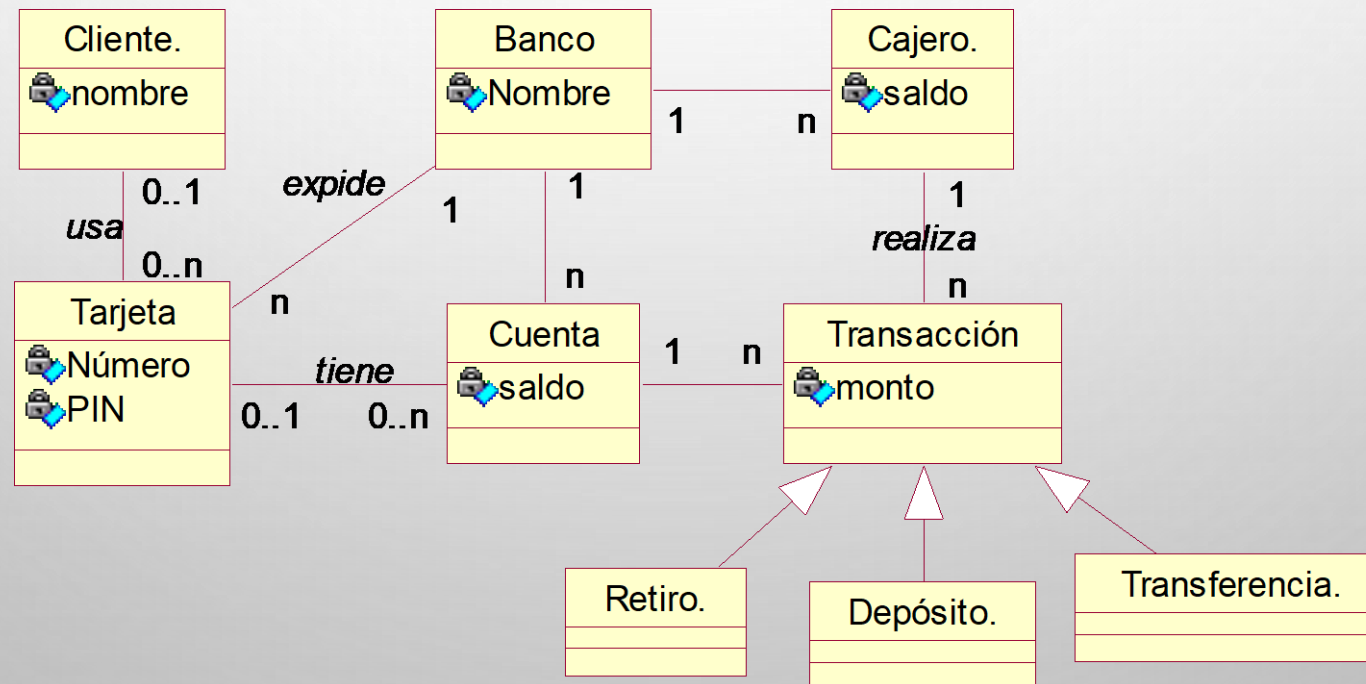
# DIAGRAMA DE CLASES

Nombre Clase
Atributos
Operaciones

- Muestra las clases e interfaces que componen el sistema y las relaciones que existen entre ellas
- Muestra aspectos estáticos
- Clase: conjunto de objetos que comparten:
  - Atributos
  - Operaciones
  - Relaciones
  - Semántica
- **Modelo de Dominio (Conceptual):** ayudan a entender los conceptos del dominio del problema y el vocabulario del mismo. Se excluyen detalles referentes a la implementación o al lenguaje de programación.
- **Diagramas de clases de implementación:** muestran todos los métodos y atributos necesarios para implementar cada clase. Es un diagrama dependiente de la implementación y del lenguaje.

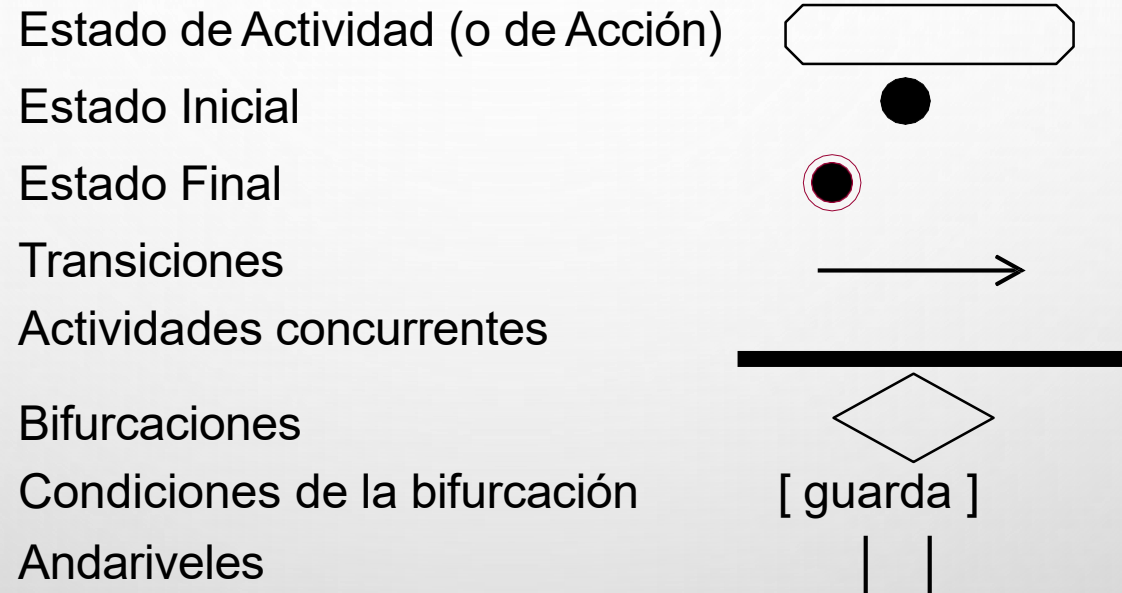
# MODELO DEL DOMINIO (CONCEPTUAL)

- Permite describir las entidades que conforman el dominio, sus relaciones y atributos
- Se representan los conceptos del dominio
- Muestra aspectos estáticos



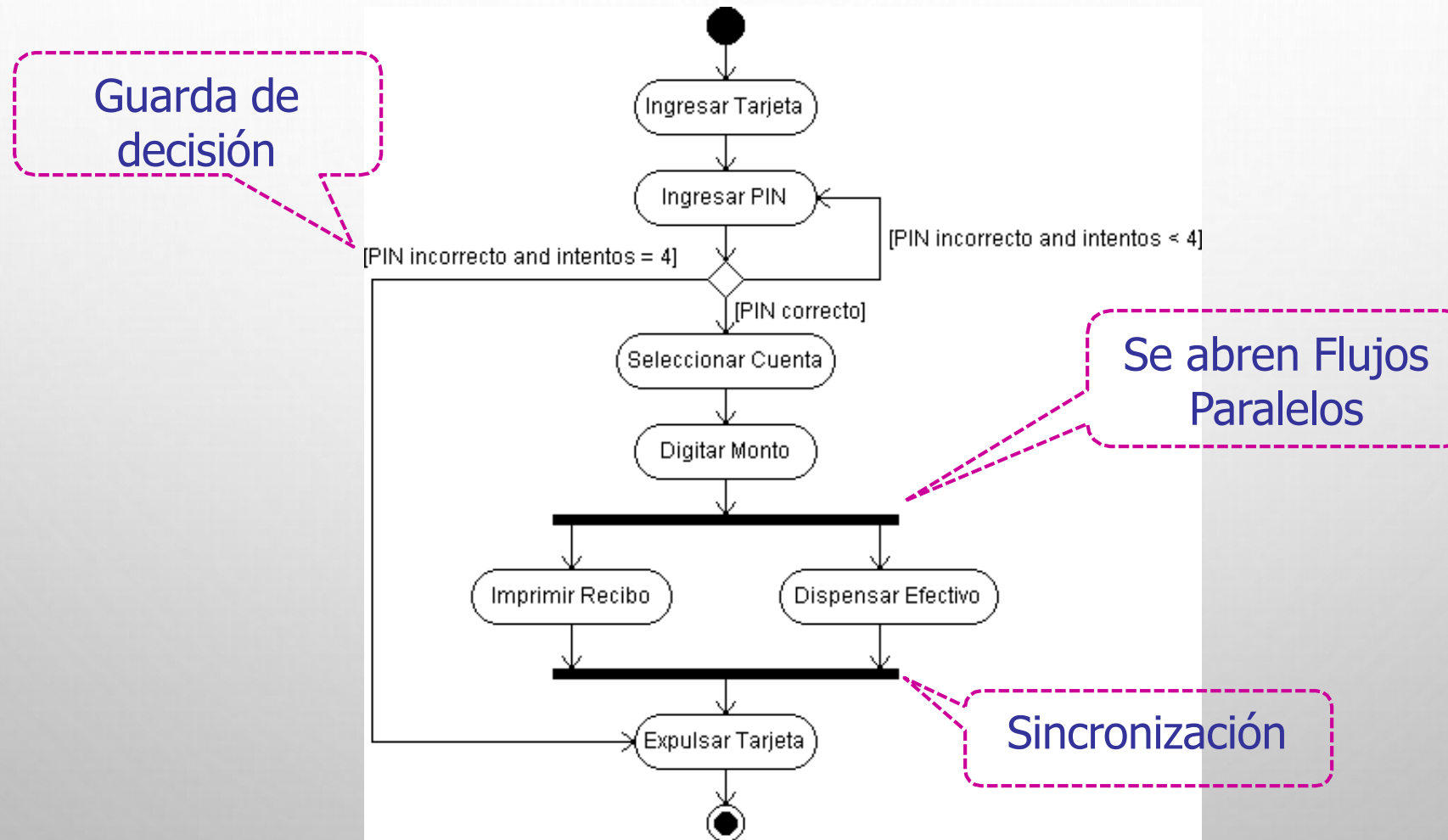
# DIAGRAMA DE ACTIVIDAD

- Se construye para modelar el flujo del control (workflow)
- Elementos:



- Permite modelar el flujo del trabajo
  - En un sistema
  - En una organización

# DIAGRAMA DE ACTIVIDAD - EJEMPLO

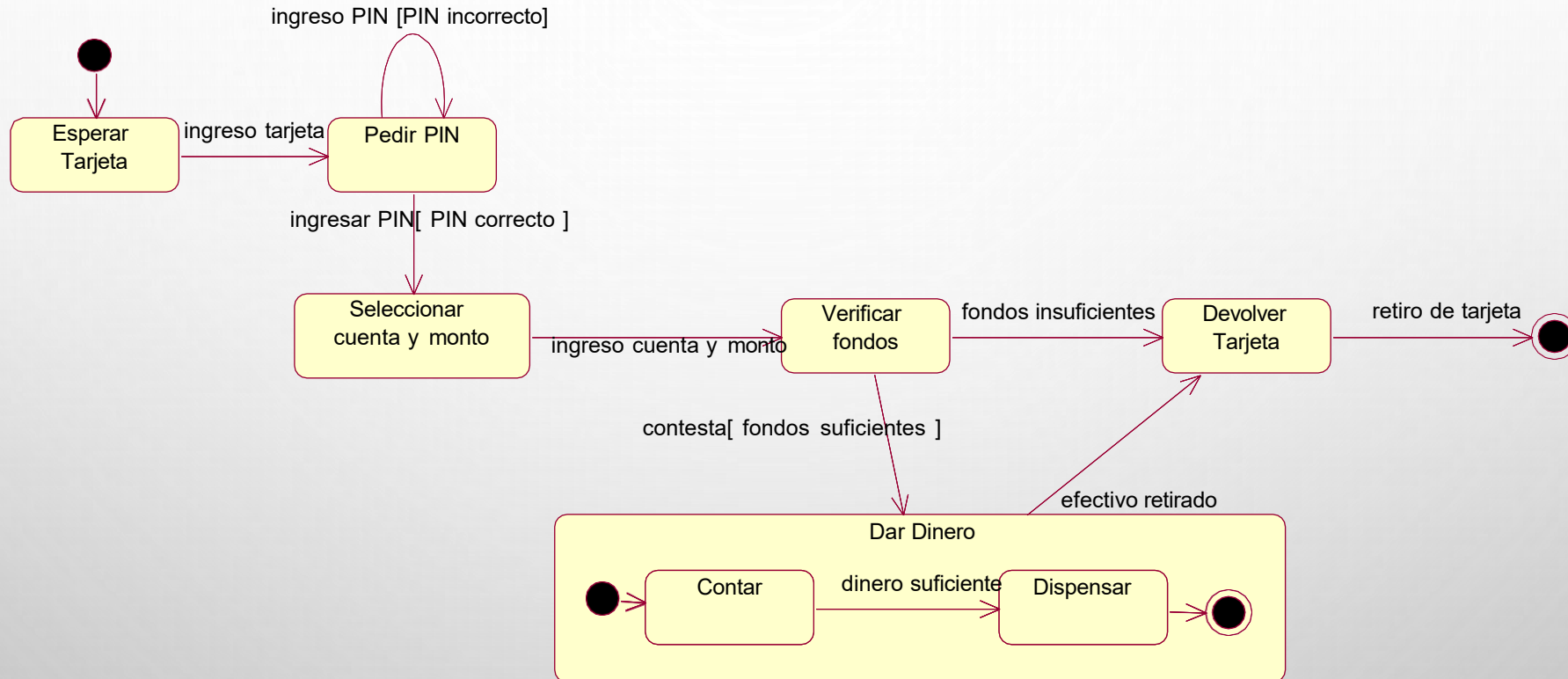




# DIAGRAMA DE ESTADOS

- Muestra el comportamiento de un objeto representando los estados en que se puede encontrar y los eventos que le hace pasar de uno a otro.
- Se utiliza para:
  - Modelar el estado interno de una entidad
  - Modelar el estado de un caso de uso
- Da una vista dinámica del sistema
- Permite:
  - Anidamiento: un estado con subestados
  - Estados paralelos: reduce el nro. de estados necesarios en el modelo
  - Condiciones de bifurcación

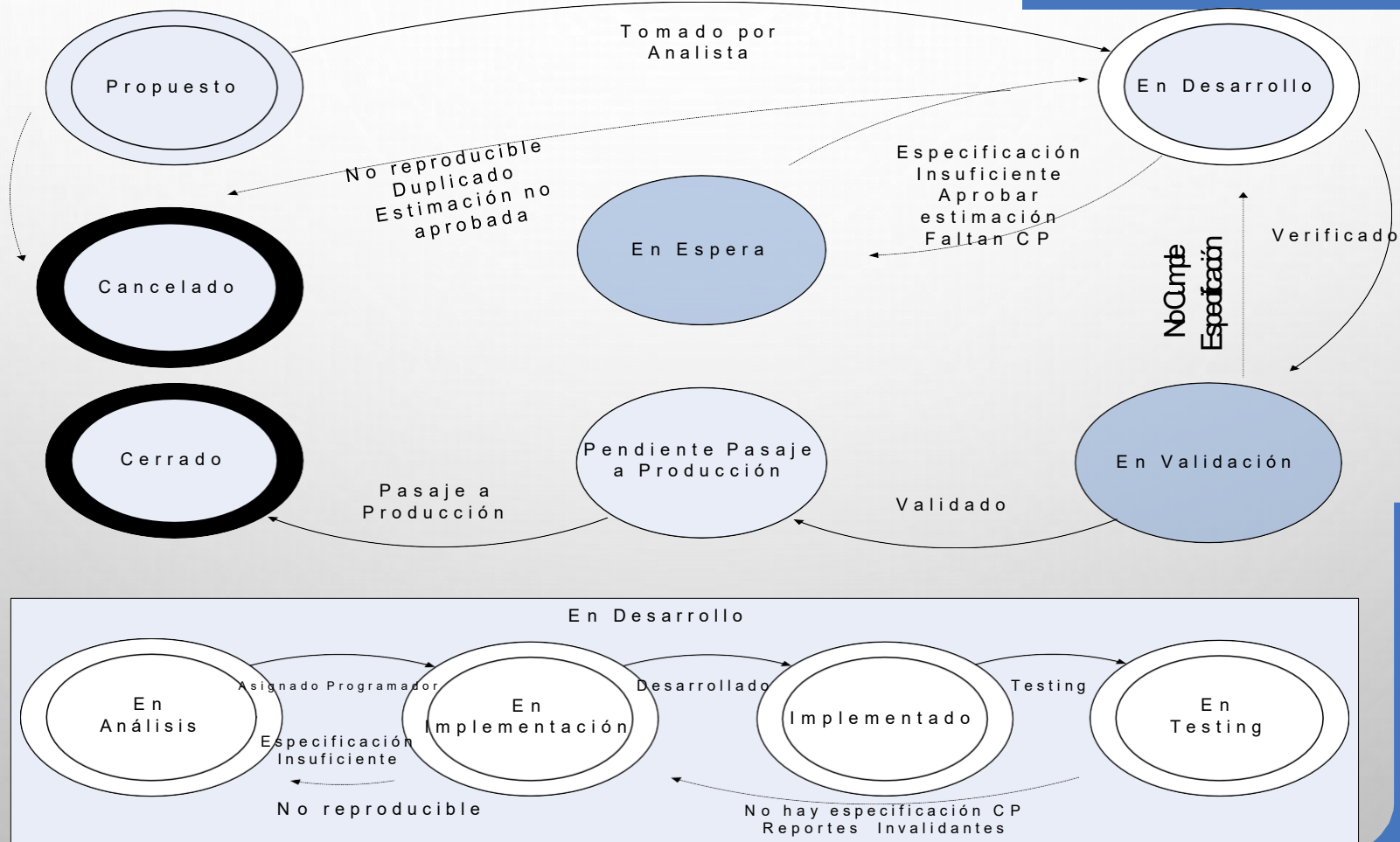
# DIAGRAMA DE ESTADOS – EJEMPLO 1



# DIAGRAMA DE ESTADOS –EJEMPLO 2

## Ciclo de Vida de Incidentes

Wednesday, April 16, 2008



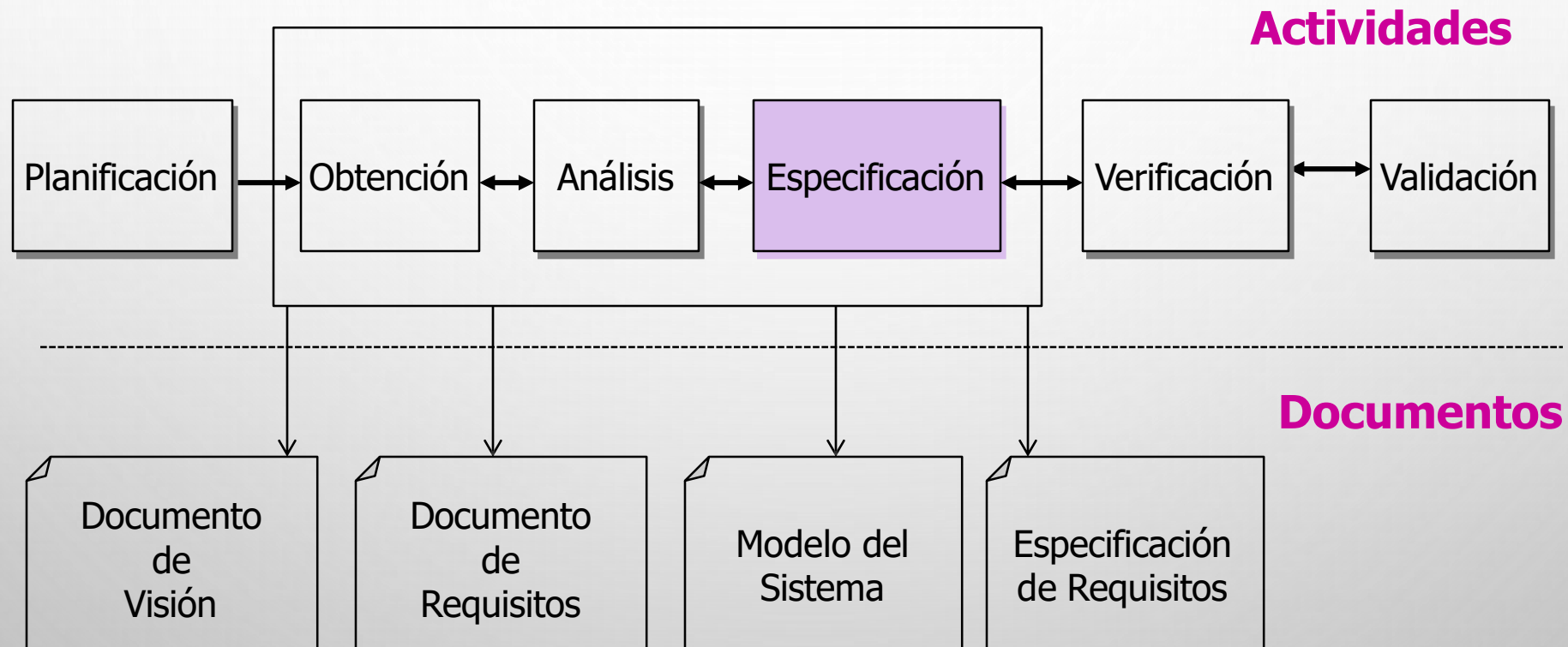
# ELECCIÓN DE UNA TÉCNICA PARA MODELAR REQUISITOS

- No existe un único enfoque aplicable a todos los sistemas, depende de cada proyecto
- Puede ser necesario combinar varios enfoques

The background of the slide is a light gray gradient. In the top-left and bottom-right corners, there are several realistic-looking water droplets of various sizes, some overlapping. A faint, circular watermark is visible in the upper center of the slide.

# ESPECIFICACIÓN DE REQUERIMIENTOS

# PROCESO DE REQUIMIENTOS





# LENGUAJES DE NOTACIÓN

- Lenguaje Natural
  - Comprensible para el Cliente/Usuario
  - Ambiguo (glosario)
  - Poca legibilidad (plantilla, formateo del texto)
  - Difícil de tratar (Verificar correctitud, consistencia, completitud)
- Notaciones Especiales (más formales)
  - Poca o ninguna ambigüedad
  - Facilita tratamiento
  - Necesidad de entrenamiento en la notación
  - Dificultades de comprensión por Cliente/Usuario

# NOTACIONES ESPECIALES

- Gráficas vs. Basadas en texto
- Estáticas vs. Dinámicas
- Descripciones Estáticas
  - Se especifican entidades y sus atributos, los requisitos se pueden ver como las relaciones entre las entidades.
  - No describe como cambian las relaciones con el tiempo
- Descripciones Dinámicas
  - Especifican estados y las transiciones entre estados en el tiempo

# DOCUMENTACIÓN DE REQUERIMIENTOS

- Qué documentar:
  - lo que hace el sistema actual
  - lo que el cliente pide
  - lo que el sistema va a hacer
  - criterios de aceptación
  - criterios de verificación
- Recomendaciones:
  - agrupar por temas
  - formular los reqs como reqs positivos y no negativos
  - expresarlos en voz activa y no pasiva
  - indicar si se está documentando solo lo que va en el alcance o todo lo que se pidió.
  - representar reqs. con múltiples vistas (ejemplo de los ciegos y el elefante).

# DOCUMENTOS DE REQUERIMIENTOS

- **Definición de Requisitos:** lista completa de lo que el cliente espera que el sistema haga, escrita de forma que el cliente la pueda entender
  1. Se debe proveer un medio para acceder a archivos externos creados por otras herramientas
- **Especificación de Requisitos (SRS):** reformula la definición en términos técnicos para que los diseñadores puedan comenzar el diseño
  1. Se proveerá al usuario los recursos para definir el tipo de archivo externo
  2. Cada tipo de archivo tendrá una herramienta asociada y un ícono que lo identifica
  3. Cuando el usuario seleccione el ícono que representa un archivo externo, el efecto es aplicar la herramienta asociada con ese tipo de archivo al archivo seleccionado

# DOCUMENTOS DE REQUERIMIENTOS (2)

- USAR UN MISMO DOCUMENTO: ENTENDIMIENTO COMÚN ENTRE CLIENTE, USUARIO, ANALISTAS, DESARROLLADORES
- USAR DOS DOCUMENTOS: SE DEBE APLICAR GESTIÓN DE CONFIGURACIÓN: NECESARIA PARA ASEGURAR LA CORRESPONDENCIA ENTRE AMBOS (SI EXISTEN POR SEPARADO)
- PERMITE SEGUIR LA PISTA Y CORRESPONDENCIA ENTRE:
  - DEFINICIÓN DE REQUISITOS
  - ESPECIFICACIÓN DE REQUISITOS
  - MÓDULOS DE DISEÑO
  - CÓDIGO QUE IMPLEMENTA LOS MÓDULOS
  - PRUEBAS PARA VERIFICAR LA FUNCIONALIDAD
  - DOCUMENTOS QUE DESCRIBEN EL SISTEMA

# DOCUMENTO DEFINICIÓN DE REQUERIMIENTOS

- Registrar los requisitos en los términos del cliente
  - 1. Delinear el propósito general del sistema: Incluir referencias a otros sistemas, glosario y abreviaciones
  - 2. Describir el contexto y objetivos del desarrollo del sistema
  - 3. Delinear visión global del sistema: Incluir restricciones generales
  - 4. Definir en detalle las características del sistema propuesto, definir la frontera del sistema e interfaces.
  - 5. Discutir el ambiente en el que el sistema va a operar (hardware, comunicaciones, personal).



# CARACTERÍSTICAS DE UNA BUENA ESPECIFICACIÓN SRS (IEEE 830)

- **Correcta / Válida:** Todos los req. son requeridos en el sistema.
  - No existe herramienta que asegure esto.
  - Validado por el cliente (que efectivamente refleje sus necesidades).
  - Revisar que sea consistente contra otros documentos existentes (pe. especificación de reqs. del sistema).
- **No Ambigua:** Todo req tiene una única interpretación.
  - Incluir glosario.
  - No ambigua para quienes lo crearon y para quienes lo usan.
- **Completa:** Incluye:
  - Todos los requisitos asociados con funcionalidad, desempeño, restricciones de diseño, atributos o interfaces externas.
  - Definición de respuestas del sw a todo posible datos de entrada (válidos o inválidos) en toda clase de situaciones realizables.
  - No hay referencias sin definir en la especificación.
  - La frase “a determinar” indica SRS no completa. Ocasionalmente necesaria; describir:
    - condiciones que causan que no se sepa aún.
    - qué se debe hacer para determinar lo que falta, quién y cuándo.

# CARACTERÍSTICAS DE UNA BUENA ESPECIFICACIÓN SRS (IEEE 830)

- Consistente internamente: Los requisitos no son contradictorios entre sí. Probables conflictos:
  - entre características de entidades. Pe. color de las luces, formatos distintos
  - conflicto lógico o temporal entre dos acciones. Pe. multiplicar o sumar; en forma simultánea o consecutiva.
  - diferentes términos para describir el mismo objeto.
- Ordenados por grado de importancia y/o estabilidad – identificador.
  - Importancia: esencial / deseado
  - Estabilidad: cantidad de cambios esperados
  - Necesidad: esencial / condicional / opcional
    - Esencial (condiciona aceptación del sw)
    - Condicional (valor agregado)
    - Opcional (puede o no valer la pena; se aceptan propuestas alternativas)

# CARACTERÍSTICAS DE UNA BUENA ESPECIFICACIÓN SRS (IEEE 830)

- Verificable: Un requerimiento es verificable si existe un proceso finito de costo accesible para determinar que el sistema lo cumple
  - Usar términos concretos y cantidades medibles.
  - Preparar pruebas para demostrar que se cumplen. Si no se puede, eliminar o revisar el requisito.
- Modificables: Su estructura y estilo son tales que cualquier cambio en los requisitos puede ser hecho fácilmente en forma completa y consistente.
  - Organización coherente y fácil de usar (tablas, índices, refs. cruzadas)
  - No redundante.
    - Ventajas de redundancia: lo hace más legible.
    - Desventajas: difícil de mantener
    - Si la uso: referencias cruzadas
  - Expresar cada req. separadamente.

# CARACTERÍSTICAS DE UNA BUENA ESPECIFICACIÓN SRS (IEEE 830)

- Trazables: El origen de cada requerimiento es claro, y es posible seguirle la pista en futuros desarrollos o mejora de la documentación
    - Trazabilidad hacia atrás: en versiones previas
    - Trazabilidad hacia adelante: documentos posteriores:
      - requiere IDENTIFICADOR ÚNICO.
- 
- Realistas / Factibles
    - Ej.: tiempo de respuesta local=remoto
    - Ej.: El cliente quiere adelantarse a la tecnología
  - Entendibles: Tanto por los usuarios como por los desarrolladores

# CARACTERÍSTICAS DE UNA BUENA ESPECIFICACIÓN SRS (IEEE 830)

- Ejemplos ambigüedad:
  - “La página web finalmente obtenida deberá permitir que toda su funcionalidad esté disponible utilizando los principales navegadores del mercado, como Explorer, Firefox, Opera y Chrome.”
  - “La solución deberá ser multiplataforma, encontrándose certificada para plataformas Linux, AIX y Windows.”
  - “Dada la naturaleza de este sistema de información, es requisito indispensable que la solución trabaje en alta disponibilidad.”



# CARACTERÍSTICAS DE UNA BUENA ESPECIFICACIÓN SRS (IEEE 830)

- Ejemplos ambigüedad:
  - “La página web finalmente obtenida deberá permitir que toda su funcionalidad esté disponible utilizando los principales navegadores del mercado, como Explorer, Firefox, Opera y Chrome.”
    - ¿Qué versión de Firefox o de los otros navegadores?
  - “La solución deberá ser multiplataforma, encontrándose certificada para plataformas Linux, AIX y Windows.”
  - “Dada la naturaleza de este sistema de información, es requisito indispensable que la solución trabaje en alta disponibilidad.”



# CARACTERÍSTICAS DE UNA BUENA ESPECIFICACIÓN SRS (IEEE 830)

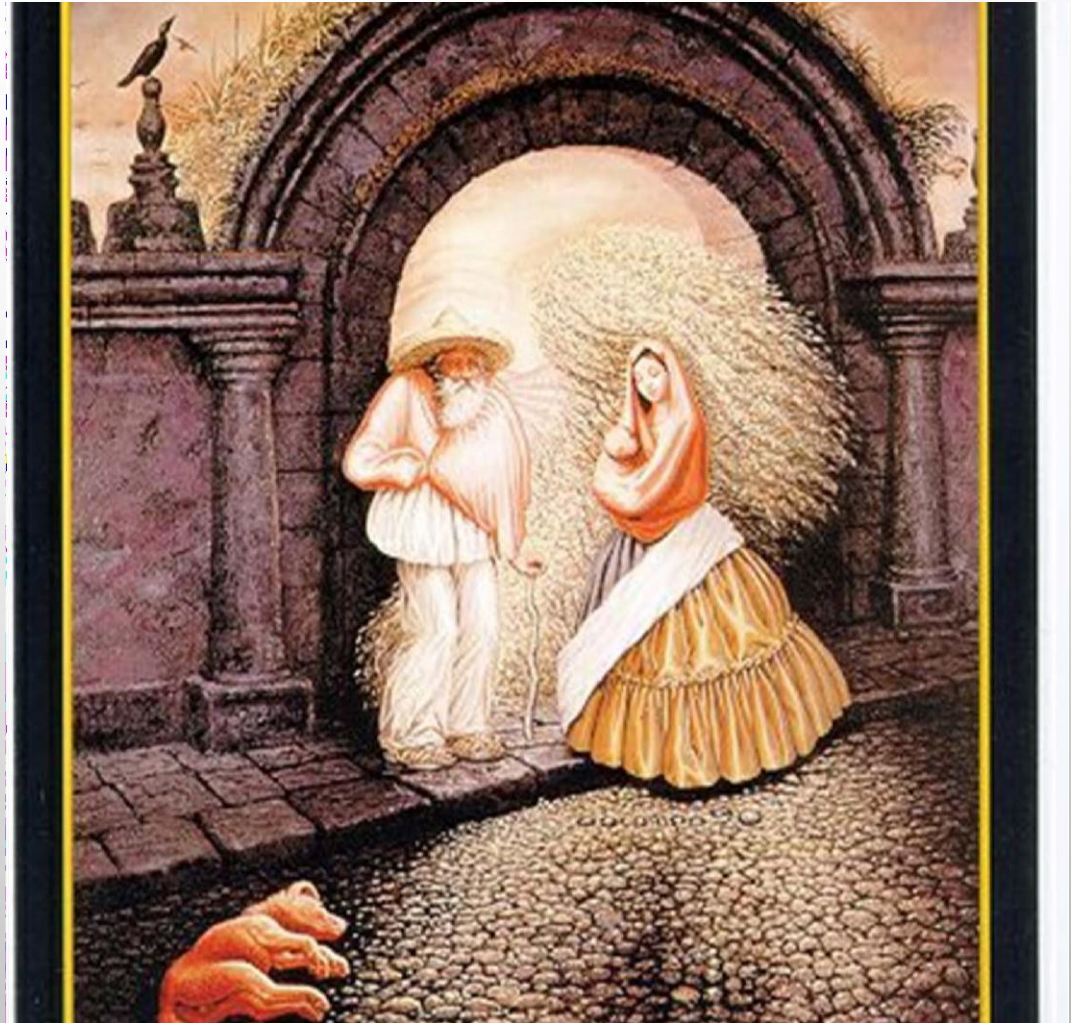
- Ejemplos ambigüedad:
  - “La página web finalmente obtenida deberá permitir que toda su funcionalidad esté disponible utilizando los principales navegadores del mercado, como Explorer, Firefox, Opera y Chrome.”
  - “La solución deberá ser multiplataforma, encontrándose certificada para plataformas Linux, AIX y Windows.”
    - ¿Qué versión de “plataformas” Linux y demás? Porque de unas versiones a otras hay enorme diferencia.
  - “Dada la naturaleza crítica de la información, es requisito indispensable que la solución trabaje en alta disponibilidad.”

# CARACTERÍSTICAS DE UNA BUENA ESPECIFICACIÓN SRS (IEEE 830)

- Ejemplos ambigüedad:
  - “La página web finalmente obtenida deberá permitir que toda su funcionalidad esté disponible utilizando los principales navegadores del mercado, como Explorer, Firefox, Opera y Chrome.”
  - “La solución deberá ser multiplataforma, encontrándose certificada para plataformas Linux, AIX y Windows.”
  - “Dada la naturaleza de este sistema de información, es requisito indispensable que la solución trabaje en alta disponibilidad.”
    - ¿Alta disponibilidad? ¿en qué porcentaje de tiempo de funcionamiento / año? ¿un 90%? ¿98%? La diferencia es tremendamente enorme.

# CARACTERÍSTICAS DE UNA BUENA ESPECIFICACIÓN SRS (IEEE 830)

- Dificultades para
- comprender los
- requerimientos:

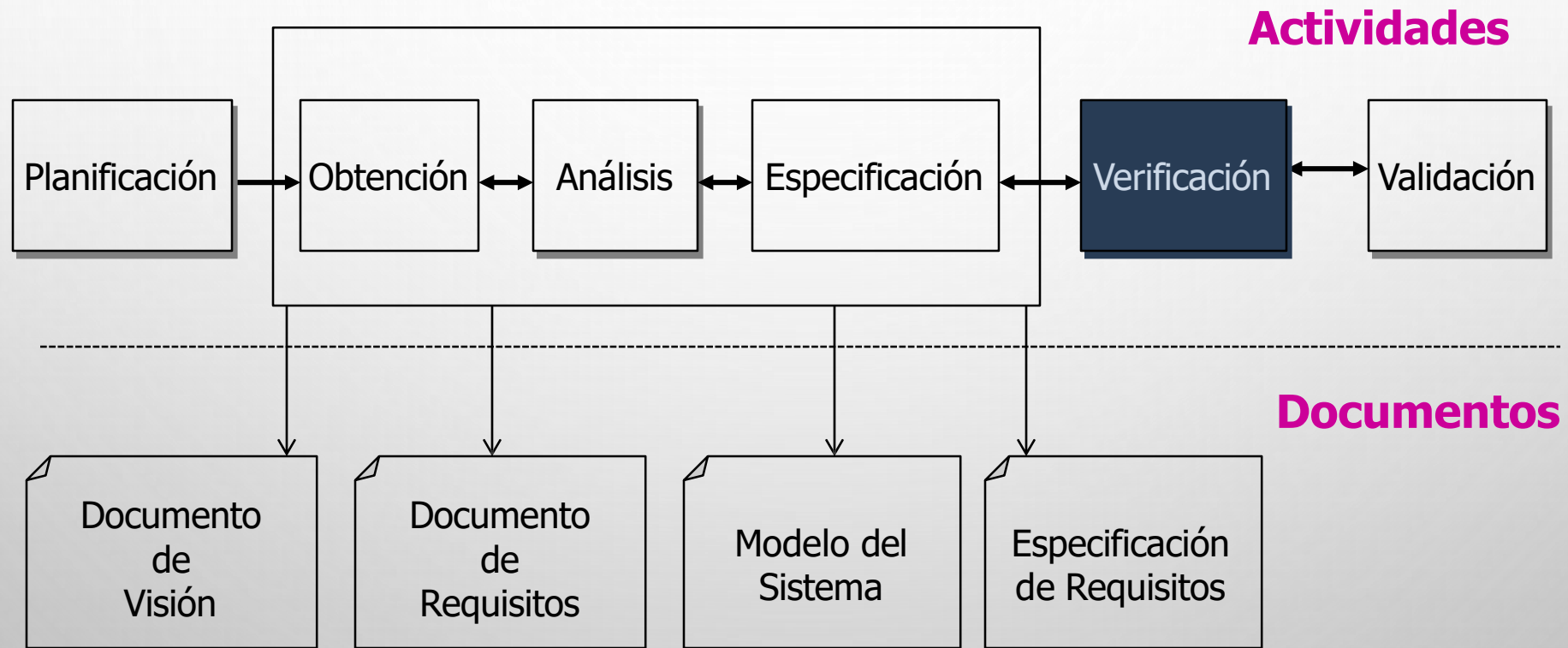


The image features a light gray background with a subtle radial gradient. In the top-left and bottom-right corners, there are clusters of realistic water droplets of various sizes, rendered with soft shadows and highlights. Faintly visible in the upper center is a circular logo or seal, which appears to be the official emblem of the United Nations, though its details are too light to discern clearly.

# VERIFICACIÓN DE REQUERIMIENTOS



# PROCESO DE REQUIMIENTOS



# VERIFICACIÓN

- Se verifica en el documento de requisitos:
  - Consistencia: que no haya contradicciones
  - Completitud: que no falte nada. Chequear por:
    - Omisiones. Hacer árboles de decisión para ver que estén todas las opciones detalladas.
    - Límites. Más claro con tabla, ahí se ve que no falta ninguno.
  - Necesidad
  - Ambigüedades
  - Realismo o Factibilidad: que se puedan implementar con la tecnología, presupuesto y calendario existentes.
  - Verificabilidad: que se pueda diseñar conjunto de pruebas para demostrar que el sistema cumple esos requisitos. Cuidado con adjetivos y adverbios.
  - Comprensibilidad: que los usuarios finales lo entiendan
  - Adaptabilidad: que el requisito se pueda cambiar sin afectar a otros.
  - Trazabilidad: que esté establecido el origen. Incluye verificar trazabilidad entre la especific. y el doc. de requisitos.



# VERIFICACIÓN DE REQUISITOS NO FUNCIONALES

- Son difíciles de verificar
- Se deben expresar de manera cuantitativa utilizando métricas que se puedan probar de forma objetiva (esto es IDEAL)

Propiedad	Medida
Rapidez	Transacciones por seg
Tamaño	KB
Fiabilidad	Tiempo promedio entre fallas
Portabilidad	Número de sistemas, especificar
Facilidad de uso	Tiempo de capacitación

- Para los usuarios es difícil especificarlos en forma cuantitativa

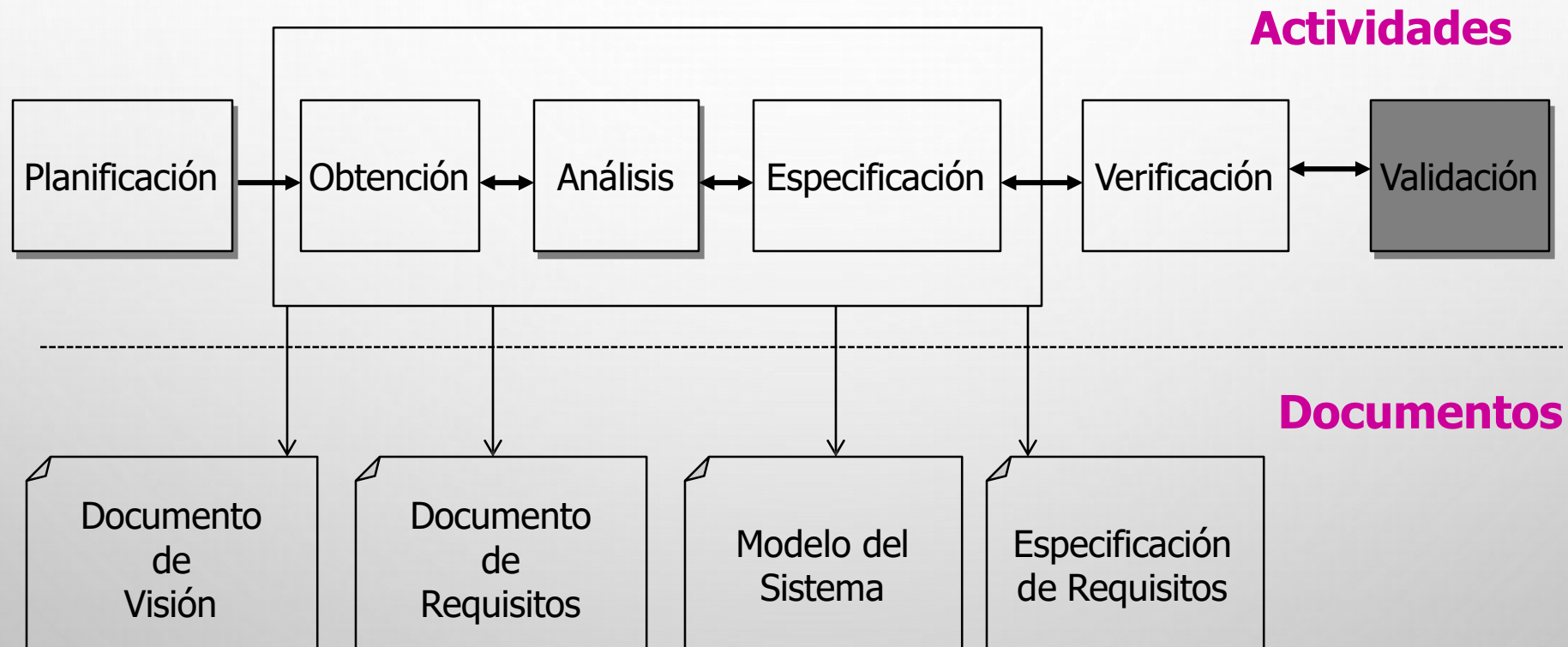
# VERIFICACIÓN DE REQUISITOS

- Definir quiénes, cómo, cuándo.
- Cómo:
  - revisiones formales (en grupo)
  - revisiones por pares
  - listas de comprobación

The background of the slide is a light gray gradient. In the top-left and bottom-right corners, there are several realistic-looking water droplets of various sizes, some overlapping. A faint, circular watermark is visible in the upper center of the slide.

# VALIDACIÓN DE REQUERIMIENTOS

# PROCESO DE REQUIMIENTOS



# VALIDACIÓN

- Proceso por el cual se determina si la especificación es consistente con las necesidades del cliente
- Se verifica en el documento de requisitos:
  - Validez: que el usuario valide qué es lo que quiere
- Planificar quién (qué stakeholder) va a validar qué artefacto cómo (técnica).
- Ejecutar
- Registrar – Reporte de validación / Firma

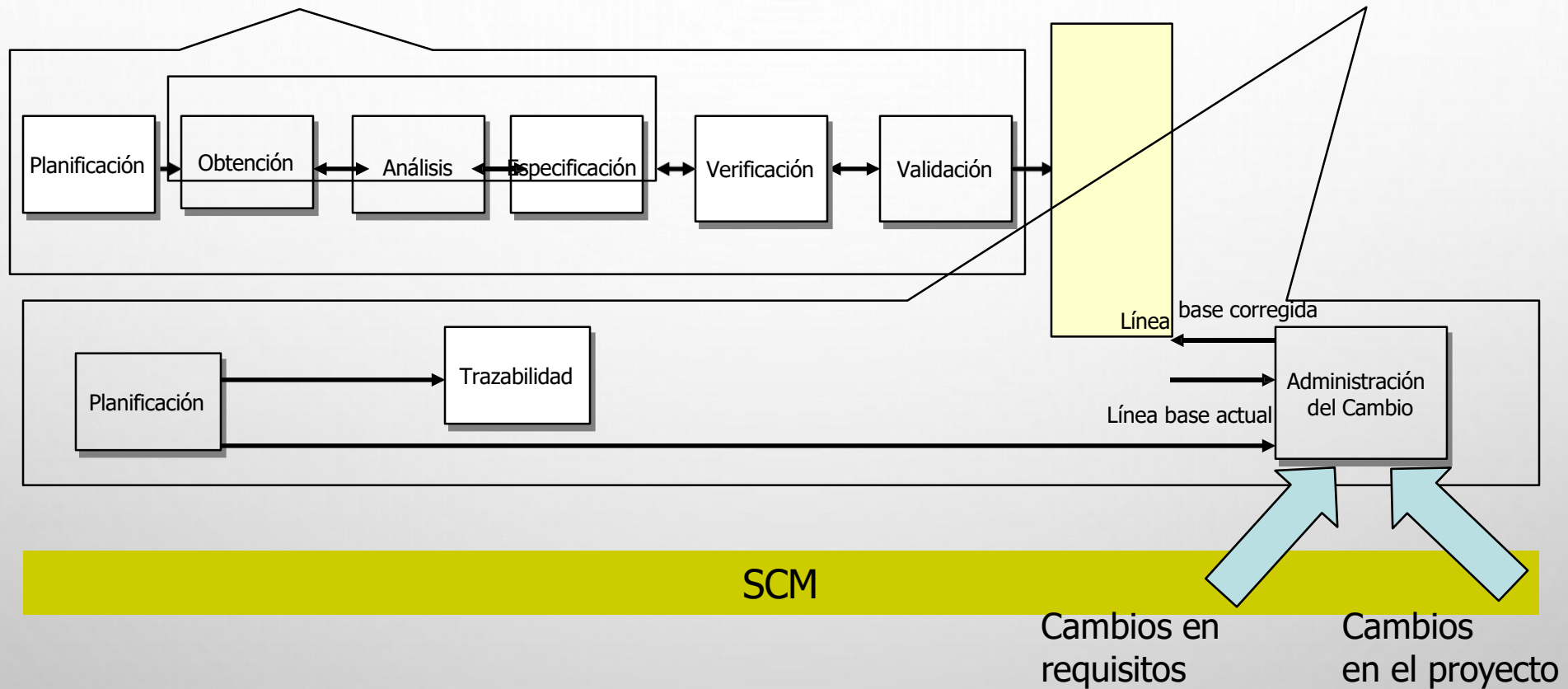
# VALIDACIÓN

- TÉCNICAS DE VALIDACIÓN:
  - MANUALES
    - LECTURA, REFERENCIAS CRUZADAS MANUALES
    - INSTANCIAS DE VALIDACIÓN FORMAL: ENTREVISTAS, REVISIONES.
    - LISTAS DE COMPROBACIÓN
    - MODELOS MANUALES PARA VERIFICAR FUNCIONES Y RELACIONES
    - ESCENARIOS
    - GENERACIÓN DE CASOS DE PRUEBA
    - PRUEBAS MATEMÁTICAS: SI SE USÓ UN LENGUAJE FORMAL, POR EJEMPLO Z
  - AUTOMATIZADAS
    - REFERENCIAS CRUZADAS AUTOMÁTICAS: SI LOS REQUISITOS SE EXPRESAN FORMALMENTE, LAS HERRAMIENTAS CASE VERIFICAN SU CONSISTENCIA
    - EJECUCIÓN DE MODELOS
    - CONSTRUCCIÓN DE PROTOTIPOS



## Proceso de los Requerimientos

## Administración de los Requerimientos



# ADMINISTRACIÓN DE LOS REQUERIMIENTOS

- Los requisitos cambian, debido a:
  - Muchos usuarios
  - Quienes pagan por el sistema y los usuarios no son las mismas personas
  - Cambios en el negocio
  - Cambios en la tecnología
- Proceso de comprender y controlar los cambios en los requisitos del sistema
- Se hace en paralelo con el Proceso de Requisitos.
- Tres etapas:
  - Planificación: Se realiza al comenzar el análisis de requisitos
  - Administración del cambio: Comienza una vez que se tiene una primera versión del documento de requisitos
  - Trazabilidad: Se mantiene a lo largo del proceso de requisitos

# PLANIFICACIÓN

- Muchas actividades son tomadas de las técnicas de SCM  
Se debe decidir sobre:
  - **Identificación de Requisitos:** Cada requisito debe identificarse en forma única, para poder ser referenciado por otros.
    - Ejemplo:
      - <Tipo> <nro> donde Tipo: F – Funcional, D- Datos, etc.
      - Ejemplo: F12
  - **Proceso de Administración del Cambio:** Actividades que evalúan el impacto y costo del cambio
  - **Políticas de Trazabilidad:** Definen qué relaciones entre reqs. y el diseño se deben registrar y cómo se van a mantener.
  - **Herramientas CASE:** De soporte para:
    - Almacenar los requisitos
    - Administrar el cambio
    - Administrar de la trazabilidad

SCM=  
Gestión  
de  
Cadena  
de  
Suministros

# TRAZABILIDAD

- Información de rastreo que se debe mantener
  - **La fuente**: Quién propuso el requerimiento y porqué
  - **Requisitos dependientes**: Vincula los requisitos dependientes entre sí, se usa para el análisis del cambio
  - Trazabilidad **entre artefactos** distintos, qué versión se corresponde con cuál. Pe.:
    - Rastreo reqs - CU
    - **Rastreo al diseño**: Vincula el req con los módulos de diseño que lo implementan
- Uso de matrices de trazabilidad

# ADMINISTRACIÓN DEL CAMBIO

- El cambio va a ocurrir.
- Objetivos del control de cambios:
  - Manejar el cambio y asegurar que el proyecto incorpora los cambios correctos por las razones correctas.
  - Anticipar y acomodar los cambios para producir la mínima interrupción y costo.
- Si los reqs cambian mucho dp de LB →
  - relevamiento incompleto/inefectivo
  - o acuerdo prematuro

# ADMINISTRACIÓN DEL CAMBIO

- Cuando se propone un cambio, debe evaluarse el impacto.
- Etapas:
  1. Especificación del cambio
  2. Evaluar impacto - Análisis del cambio y costo:
    - Se usa la información del rastreo
    - Se calcula el costo en términos de modificaciones a:
      - Docs de requisitos
      - Diseño e implementación
  3. Discutir costo con cliente.
  4. Implementar el cambio: se modifican los artefactos necesario
- Siguiendo estos pasos se logra
  - Todos los cambios se tratan en forma consistente
  - Los cambios a los docs de requisitos se hacen en forma controlada



# PROCEDIMIENTO DE CONTROL DE CAMBIOS

- Establecer procedimiento de control de cambios:
  - quién - Comité de Control de Cambios (CCC)
  - documentar:
    - integración del CCC
    - alcance de autoridad
    - procedimientos operativos (pe. evaluar impacto)
    - proceso de toma de decisiones

# GESTIÓN DE LA CONFIGURACIÓN DE LOS REQUISITOS

- Tiene que haber un responsable
- Control de versiones: Definir:
  - Ítems de configuración
  - Procedimientos
- Línea Base. Definición:
  - Conjunto de especificaciones y /o productos que han sido revisados formalmente y acordados, que sirven de base para desarrollo futuro, y que solo pueden ser cambiados a través de procedimientos formales de control de cambios.

# LÍNEA BASE DE REQUISITOS

- LB de reqs, arranca cuando se decide que son suficientemente buenos como para arrancar diseño y construcción.
- Sobre LB planifico cronograma y costo.
- Asociada a la liberación de un producto. Debo poder recomponer la liberación.
- Definir:
  - qué artefactos van en Línea Base
  - cuándo entran

# MEDIR Y EVALUAR REQUISITOS

- Medir características de los requisitos para obtener detalles
  - Proceso de los Requisitos
  - Calidad de los Requisitos
- Las mediciones van a estar relacionadas con:
  - Producto (de los requisitos)
    - tamaño, calidad, atributos técnicos, ....
  - Proceso
    - actividades,...
  - Recursos
    - personas, equipos, tiempo, dinero,...

# MEDIR Y EVALUAR REQUISITOS

- Medir
  - # Requisitos
    - Entrada para estimación del producto
  - # Cambios introducidos
    - Requisitos Agregados, Modificados, Desechados en el tiempo
    - Estabilidad
  - # Requisitos por tipo de requisitos
    - Permite luego ver en qué parte se encuentra el cambio
  - # Requisitos validados
- Tamaño del producto y del proyecto (ej.:PF, LoC)
  - planificar



RESUMIENDO...



# MODELO DE REQUERIMIENTOS

- EL MODELADO DE LOS REQUERIMIENTOS DESDE TRES PERSPECTIVAS DISTINTAS:
  - MODELOS BASADOS EN EL **ESCENARIO**,
  - MODELOS DE **DATOS** (INFORMACIÓN) Y
  - MODELOS BASADOS EN LA **CLASE**.
- CADA UNA REPRESENTA A LOS REQUERIMIENTOS EN UNA “DIMENSIÓN” DIFERENTE, CON LO QUE AUMENTA LA PROBABILIDAD DE DETECTAR ERRORES, DE QUE AFLOREN LAS INCONSISTENCIAS Y DE QUE SE REVELEN LAS OMISIONES.

# MODELO DE REQUERIMIENTOS

- LA ACCIÓN DE MODELAR LOS REQUERIMIENTOS DA COMO RESULTADO UNO O MÁS DE LOS SIGUIENTES TIPOS DE MODELO:
  - MODELOS BASADOS EN EL ESCENARIO DE LOS REQUERIMIENTOS
  - MODELOS ORIENTADOS A CLASES
  - MODELOS ORIENTADOS AL FLUJO
  - MODELOS DE COMPORTAMIENTO

# MODELO DE REQUERIMIENTOS

LA ATENCIÓN SE CENTRA EN **QUÉ**, NO EN CÓMO.

- ¿**QUÉ** INTERACCIÓN DEL USUARIO OCURRE EN UNA CIRCUNSTANCIA PARTICULAR?,
- ¿**QUÉ** OBJETOS MANIPULA EL SISTEMA?,
- ¿**QUÉ** FUNCIONES DEBE REALIZAR EL SISTEMA?,
- ¿**QUÉ** COMPORTAMIENTOS TIENE EL SISTEMA?,
- ¿**QUÉ** INTERFACES SE DEFINEN? Y
- ¿**QUÉ** RESTRICCIONES SON APLICABLES?

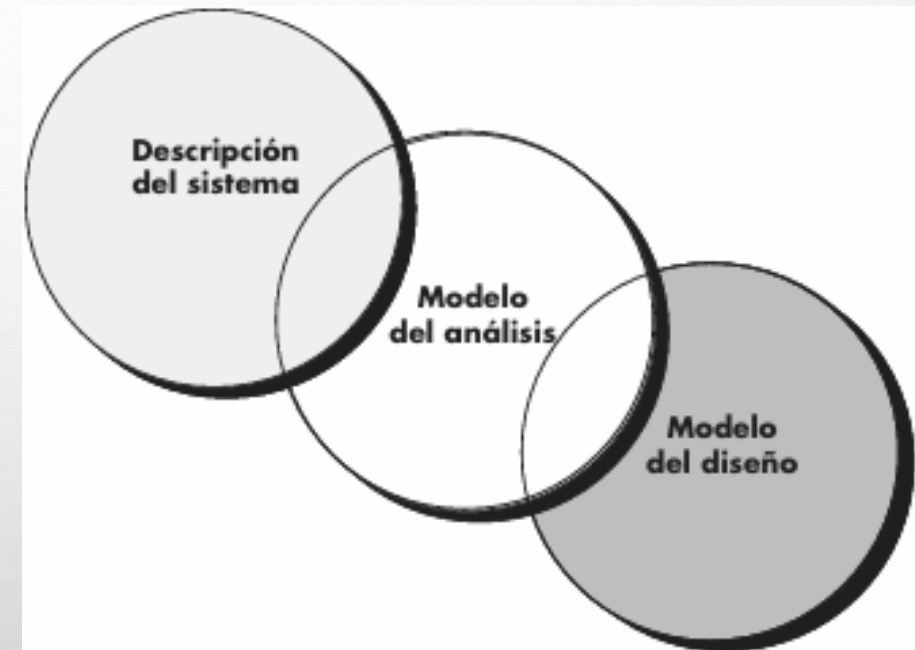
A large, bold, red word "Qué!" is centered within a white rectangular box. The word is written in a sans-serif font, with the exclamation mark being slightly larger than the letters. The box is positioned on the right side of the slide, overlapping the light gray background.

# MODELO DE REQUERIMIENTOS: OBJETIVOS

- 1) DESCRIBIR LO QUE REQUIERE EL CLIENTE,
- 2) ESTABLECER UNA BASE PARA LA CREACIÓN DE UN DISEÑO DE SOFTWARE Y
- 3) DEFINIR UN CONJUNTO DE REQUERIMIENTOS QUE PUEDAN VALIDARSE UNA VEZ CONSTRUIDO EL SOFTWARE.

# MODELO DE REQUERIMIENTOS

**EL MODELO DE ANÁLISIS** ES UN PUENTE ENTRE LA DESCRIPCIÓN EN EL NIVEL DEL SISTEMA QUE SE CENTRA EN ÉSTE EN LO GENERAL O EN LA FUNCIONALIDAD DEL NEGOCIO QUE SE LOGRA CON LA APLICACIÓN DE SOFTWARE, HARDWARE, DATOS, PERSONAS Y OTROS ELEMENTOS DEL SISTEMA Y UN DISEÑO DE SOFTWARE

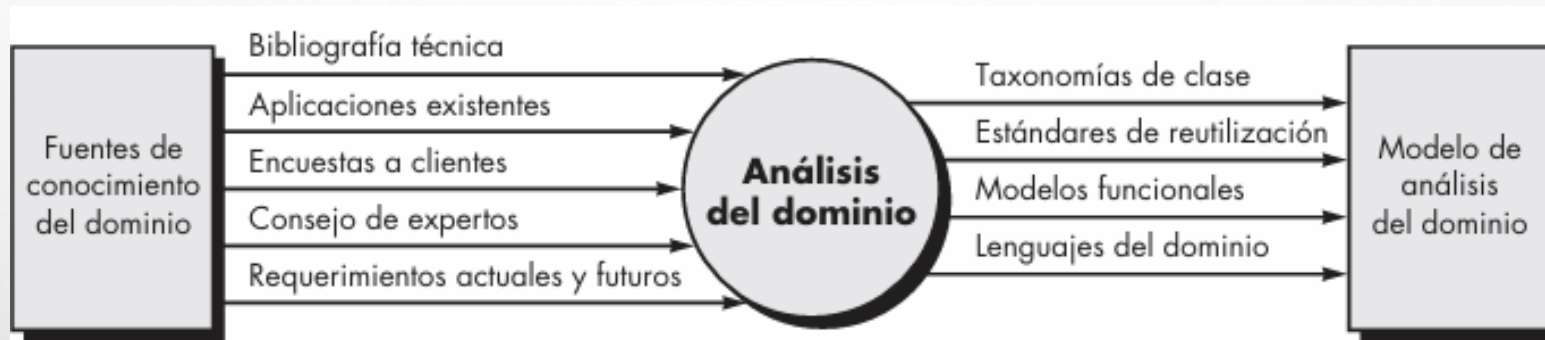


# ANÁLISIS DE DOMINIO

- EL ANÁLISIS DEL DOMINIO DEL SOFTWARE ES LA IDENTIFICACIÓN, ANÁLISIS Y ESPECIFICACIÓN DE LOS REQUERIMIENTOS COMUNES, A PARTIR DE UN DOMINIO DE APLICACIÓN ESPECÍFICA, NORMALMENTE PARA USARLO VARIAS VECES EN MÚLTIPLES PROYECTOS DENTRO DEL DOMINIO DE LA APLICACIÓN [...]
- [EL ANÁLISIS DEL DOMINIO ORIENTADO A OBJETOS ES] LA IDENTIFICACIÓN, ANÁLISIS Y ESPECIFICACIÓN DE CAPACIDADES COMUNES Y REUTILIZABLES DENTRO DE UN DOMINIO DE APLICACIÓN ESPECÍFICA EN TÉRMINOS DE OBJETOS, CLASES, SUBENSAMBLES Y ESTRUCTURAS COMUNES.
  - FIRESMITH [FIR93]

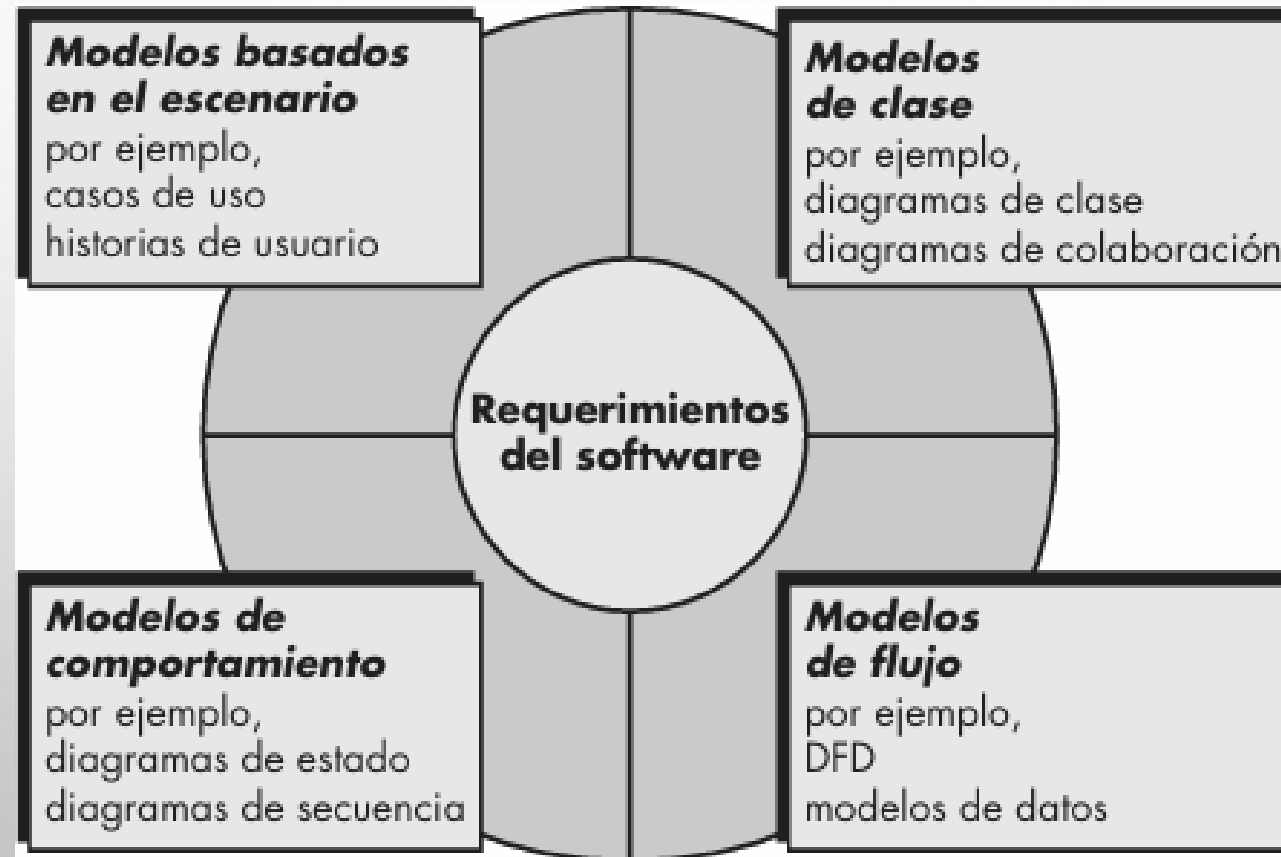


# ANÁLISIS DE DOMINIO



- EL PAPEL DEL ANALISTA DE DOMINIO<sup>5</sup> ES DESCUBRIR Y DEFINIR PATRONES DE ANÁLISIS, CLASES DE ANÁLISIS E INFORMACIÓN RELACIONADA QUE PUEDA SER UTILIZADA POR MUCHA GENTE QUE TRABAJE EN APLICACIONES SIMILARES, PERO QUE NO SON NECESARIAMENTE LAS MISMAS.
- LA FIGURA ILUSTRA ENTRADAS Y SALIDAS CLAVE PARA EL PROCESO DE ANÁLISIS DEL DOMINIO. LAS FUENTES DE CONOCIMIENTO DEL DOMINIO SE MAPEAN CON EL FIN DE IDENTIFICAR LOS OBJETOS QUE PUEDEN REUTILIZARSE A TRAVÉS DEL DOMINIO.

# MODELADO DE REQUERIMIENTOS



# MODELOS BASADOS EN EL ESCENARIO

- AUNQUE EL ÉXITO DE UN SISTEMA O PRODUCTO BASADO EN COMPUTADORA SE MIDE DE MUCHAS MANERAS, LA SATISFACCIÓN DEL USUARIO OCUPA EL PRIMER LUGAR DE LA LISTA. SI SE ENTIENDE CÓMO DESEAN INTERACTUAR LOS USUARIOS FINALES (Y OTROS ACTORES) CON UN SISTEMA, EL EQUIPO DEL SOFTWARE ESTARÁ MEJOR PREPARADO PARA CARACTERIZAR ADECUADAMENTE LOS REQUERIMIENTOS Y HACER ANÁLISIS SIGNIFICATIVOS Y MODELOS DEL DISEÑO.
- EL MODELADO DE LOS REQUERIMIENTOS CON UML COMIENZA CON LA CREACIÓN DE ESCENARIOS EN FORMA DE CASOS DE USO, DIAGRAMAS DE ACTIVIDADES Y DIAGRAMAS TIPO CARRIL DE NATACIÓN.



**BUENA SEMANA...**