Metodoloxías de Desenvolvemento

Proceso Unificado: Definiciones y Flujos de trabajo

Javier Parapar
@jparapar
javierparapar@udc.es

Revised: Pedro Cabalar

Updated: 23 de octubre de 2017

4 P's: People; Project; Product; Process

1. **People** (la gente)

Si falla la gente, ¡falla todo!

4 P's: People; Project; Product; Process

1. **People** (la gente)

Si falla la gente, ¡falla todo! Factores a tener en cuenta

- Trabajo en **equipo**: tamaño ideal 6-8 personas.
  Clave: **identificar subsistemas** (arquitectura) abordables por equipos de ese tamaño

4 P's: People; Project; Product; Process

1. **People** (la gente)

Si falla la gente, ¡falla todo! Factores a tener en cuenta

- Trabajo en **equipo**: tamaño ideal 6-8 personas.
  Clave: **identificar subsistemas** (arquitectura) abordables por equipos de ese tamaño
- La gente trabaja mejor si **entiende el propósito** de lo que hace dentro del **contexto general** del proyecto

4 P's: People; Project; Product; Process

1. **People** (la gente)

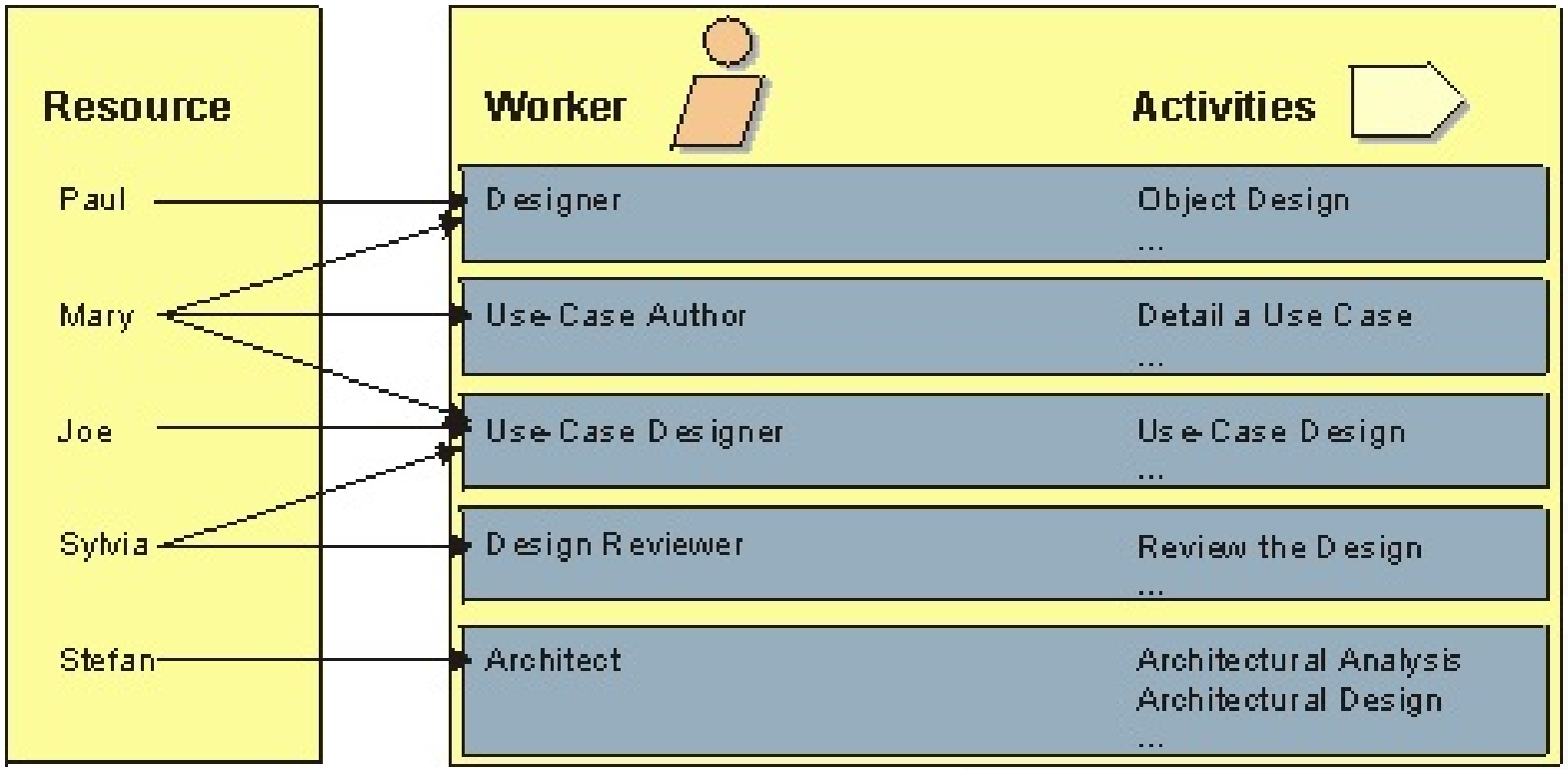
Si falla la gente, ¡falla todo! Factores a tener en cuenta

- Trabajo en **equipo**: tamaño ideal 6-8 personas.
  Clave: **identificar subsistemas** (arquitectura) abordables por equipos de ese tamaño
- La gente trabaja mejor si **entiende el propósito** de lo que hace dentro del **contexto general** del proyecto
- Evitar la visión "nunca conseguiré acabar"$\Rightarrow$ **consecución de metas parciales** (iteraciones)

4 P's: People; Project; Product; Process

1. **People** (cont)

- **persona** (recurso) ≠ trabajador (**worker**) =
- Una persona puede actuar como distintos trabajadores.
- Cada trabajador tiene sus actividades, y las puede desempeñar bajo distintos roles



4 P's: People; Project; Product; Process

2. **Proyecto**

Organización por ciclos que culminan en **entregas** (releases)

- Cada ciclo **planifica**: análisis, diseño, implementación y pruebas

4 P's: People; Project; Product; Process

2. **Proyecto**

Organización por ciclos que culminan en **entregas** (releases)

- Cada ciclo **planifica**: análisis, diseño, implementación y pruebas
- El ciclo se **estructura** en un conjunto de iteraciones.

4 P's: People; Project; Product; Process

2. **Proyecto**

Organización por ciclos que culminan en **entregas** (releases)

- Cada ciclo **planifica**: análisis, diseño, implementación y pruebas
- El ciclo se **estructura** en un conjunto de iteraciones.
- Cada iteración es un **miniproyecto** con objetivos parciales y **cambios** a realizar o añadir

4 P's: People; Project; Product; Process

3. **Producto**

Resultado del proceso = **producto SW**

- El sistema SW son los programas y su código fuente

4 P's: People; Project; Product; Process

3. **Producto**

Resultado del proceso = **producto SW**

- El sistema SW son los programas y su código fuente

4 P's: People; Project; Product; Process

3. **Producto**

Resultado del proceso = **producto SW**

- El sistema SW son los programas y su código fuente pero **también** modelos y diagramas, tests, bases de datos, documentos, instalación, manuales, ...
- En general: **todo lo necesario para los trabajadores**.

4 P's: People; Project; Product; Process

3. **Producto**

Resultado del proceso = **producto SW**

- El sistema SW son los programas y su código fuente pero **también** modelos y diagramas, tests, bases de datos, documentos, instalación, manuales, ...
- En general: **todo lo necesario para los trabajadores**.
- Tres **tipos de trabajador**:

4 P's: People; Project; Product; Process

3. **Producto**

Resultado del proceso = **producto SW**

- El sistema SW son los programas y su código fuente pero **también** modelos y diagramas, tests, bases de datos, documentos, instalación, manuales, ...
- En general: **todo lo necesario para los trabajadores**.
- Tres **tipos de trabajador**:
  1. **Máquinas**: ordenadores, dispositivos, herramientas, compiladores, ...

4 P's: People; Project; Product; Process

3. **Producto**

Resultado del proceso = **producto SW**

- El sistema SW son los programas y su código fuente pero **también** modelos y diagramas, tests, bases de datos, documentos, instalación, manuales, ...
- En general: **todo lo necesario para los trabajadores**.
- Tres **tipos de trabajador**:
  1. **Máquinas**: ordenadores, dispositivos, herramientas, compiladores, ...
  2. **Ingenieros SW**: analistas, arquitectos SW, desarrolladores, probadores, administradores, ...

4 P's: People; Project; Product; Process

3. **Producto**

Resultado del proceso = **producto SW**

- El sistema SW son los programas y su código fuente pero **también** modelos y diagramas, tests, bases de datos, documentos, instalación, manuales, ...
- En general: **todo lo necesario para los trabajadores**.
- Tres **tipos de trabajador**:
  1. **Máquinas**: ordenadores, dispositivos, herramientas, compiladores, ...
  2. **Ingenieros SW**: analistas, arquitectos SW, desarrolladores, probadores, administradores, ...
  3. **Involucrados** (stakeholders): usuarios, dirección y gestión, comerciales, autoridades, agencias reguladoras, ...

3. **Producto** (cont)

Definición (Artefacto)

Cualquier información creada, producida, cambiada o usada por los trabajadores para el desarrollo de un sistema SW

3. **Producto** (cont)

Definición (Artefacto)

Cualquier información creada, producida, cambiada o usada por los trabajadores para el desarrollo de un sistema SW

Ejemplos:

- Artefactos de **Ingeniería**: código, modelos, diagramas UML, borradores de interfaces, ...
- Artefactos de **Gestión**: plan de proyecto, diagramas Gantt, listas de tareas, tablas de asignación de recursos, ...

4 P's: People; Project; Product; Process

3. **Producto** (cont)

Un tipo de artefacto de ingeniería esencial en UP son los **modelos**.

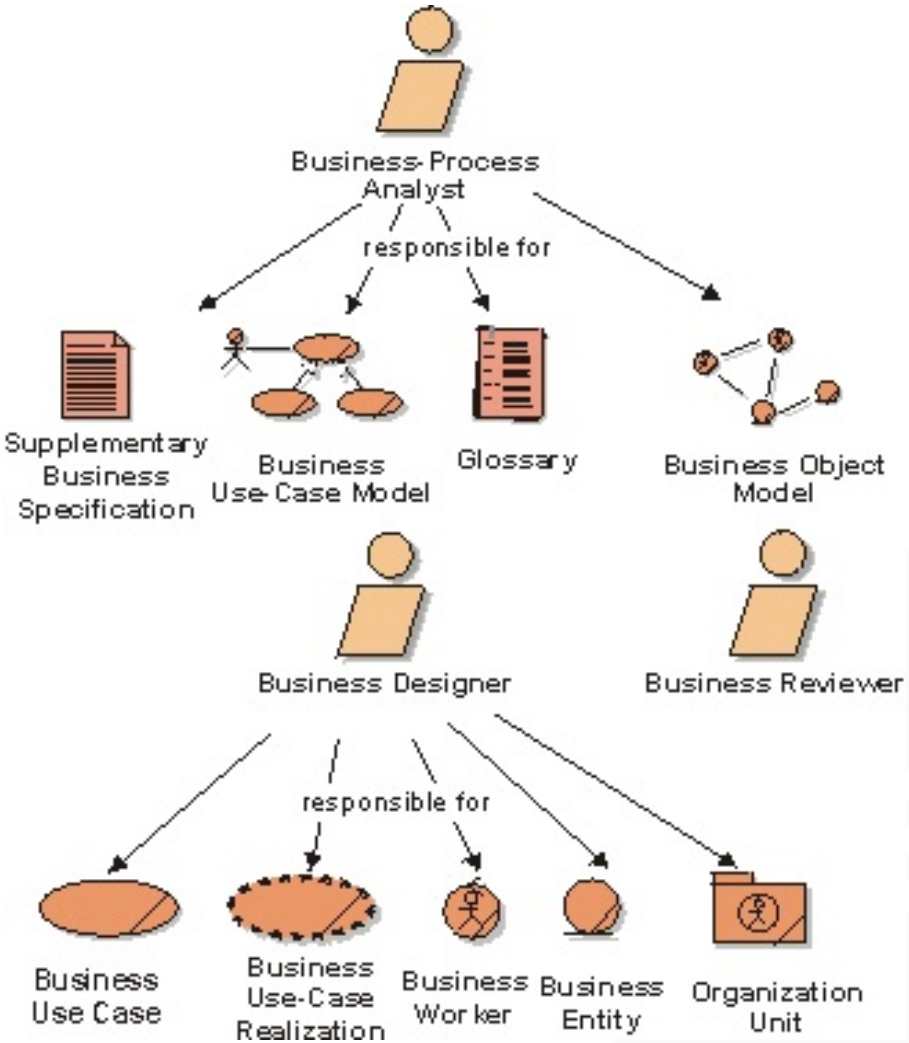
Seis **tipos de modelos**

1. de casos de uso
2. de análisis
3. de diseño
4. de implementación
5. de pruebas
6. de despliegue

4 P's: People; Project; Product; Process

3. **Producto** (cont)

Ejemplo: trabajadores y artefactos bajo su responsabilidad



4. **Proceso**

Definición (Proceso)

*Conjunto de **flujos de trabajo** y las **actividades** asociadas a cada uno que actúan como **plantilla para crear proyectos***

Cada flujo de trabajo es una **meta-especificación**, un diagrama que relaciona:

- **Trabajadores** involucrados
- **Artefactos** usados, modificados o generados
- **Actividades** y su secuencia temporal (diagramas de flujo)

4 P's: People; Project; Product; Process

4. **Proceso** (cont)

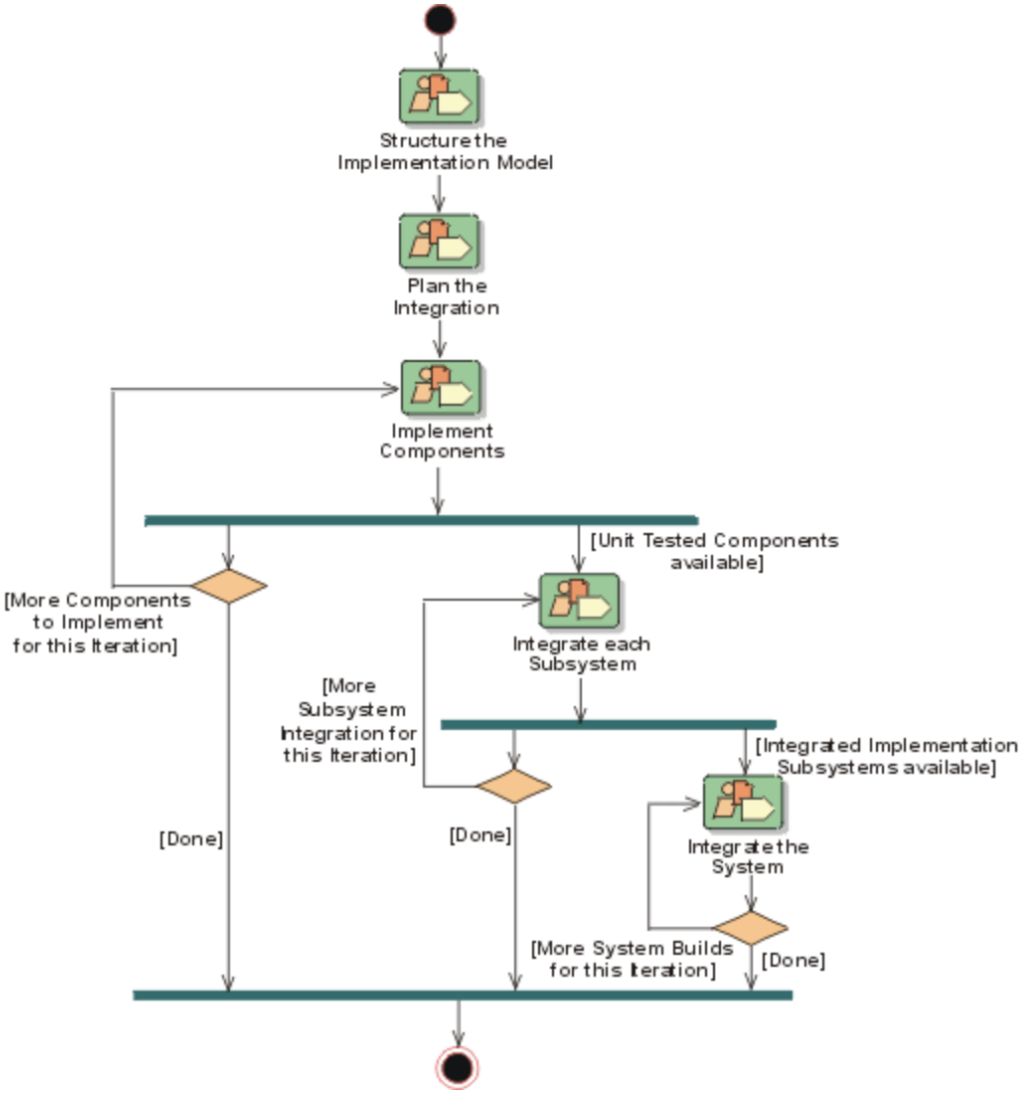
Ventajas de especificar el proceso de desarrollo

- **Referencia** para ingenieros SW: qué hay que hacer en cada momento
- Simplifica el **aprendizaje** del proceso por nuevos desarrolladores
- Facilita **movilidad** de trabajadores
- Para otros involucrados (usuarios, clientes, ...): **clarifica** el avance del proceso
- Mayor **fiabilidad en estimaciones**

`http://sce.uhcl.edu/helm/rationalunifiedprocess/`

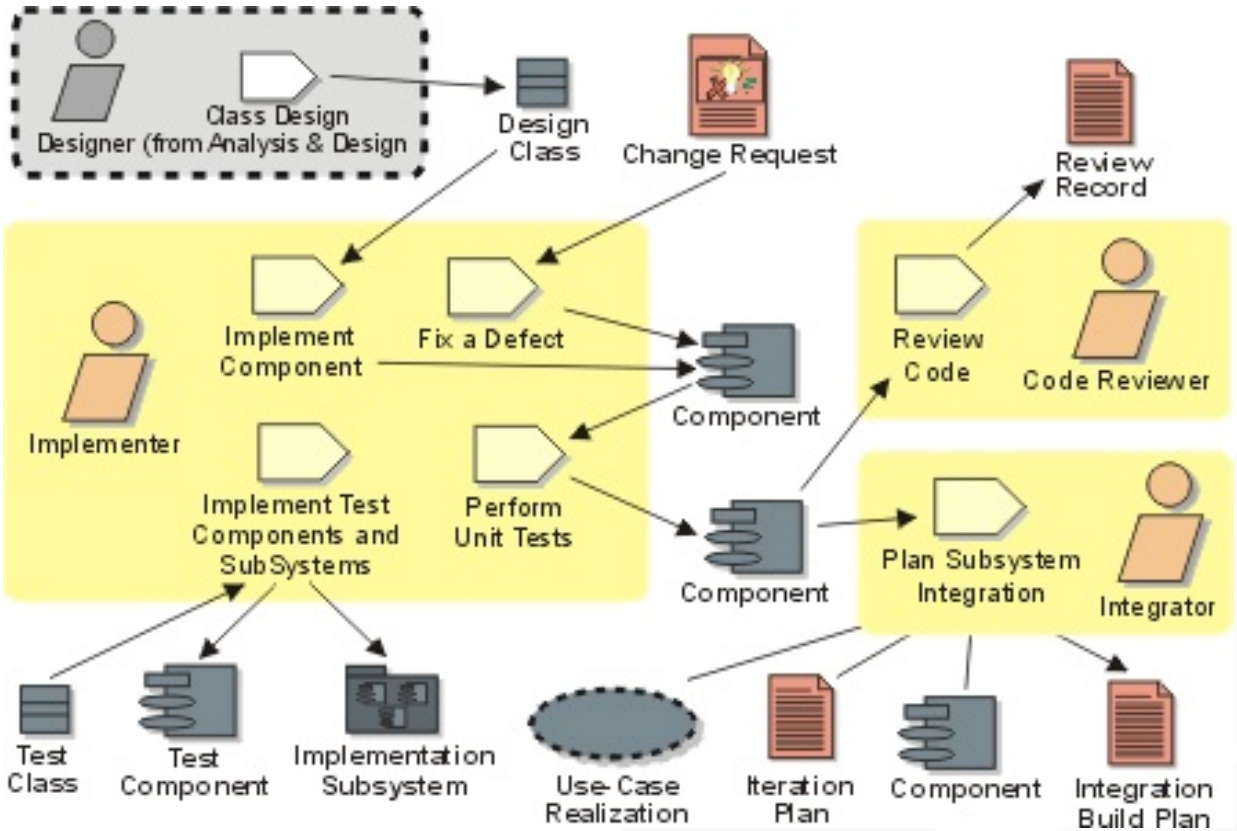
4. **Proceso**

Ejemplo: flujo de trabajo de implementación



4. **Proceso**

Ejemplo: detalle de "Implementar Componentes"



Flujos de trabajo

Podemos definir los principales flujos de trabajo para la fase del proyecto de **construcción** del software.

1. Captura de **requisitos** como casos de uso $\mapsto$ Modelo de Casos de Uso

Flujos de trabajo

Podemos definir los principales flujos de trabajo para la fase del proyecto de **construcción** del software.

1. Captura de **requisitos** como casos de uso $\mapsto$ Modelo de Casos de Uso
2. **Análisis** de los Casos de Uso $\mapsto$ Modelo de Análisis

Flujos de trabajo

Podemos definir los principales flujos de trabajo para la fase del proyecto de **construcción** del software.

1. Captura de **requisitos** como casos de uso $\mapsto$ Modelo de Casos de Uso
2. **Análisis** de los Casos de Uso $\mapsto$ Modelo de Análisis
3. **Diseño** desde los Casos de Uso $\mapsto$ Modelo de Diseño

Flujos de trabajo

Podemos definir los principales flujos de trabajo para la fase del proyecto de **construcción** del software.

1. Captura de **requisitos** como casos de uso $\mapsto$ Modelo de Casos de Uso
2. **Análisis** de los Casos de Uso $\mapsto$ Modelo de Análisis
3. **Diseño** desde los Casos de Uso $\mapsto$ Modelo de Diseño
4. **Implementación** desarrollando Componentes $\mapsto$ Modelo de Implementación

Flujos de trabajo

Podemos definir los principales flujos de trabajo para la fase del proyecto de **construcción** del software.

1. Captura de **requisitos** como casos de uso $\mapsto$ Modelo de Casos de Uso
2. **Análisis** de los Casos de Uso $\mapsto$ Modelo de Análisis
3. **Diseño** desde los Casos de Uso $\mapsto$ Modelo de Diseño
4. **Implementación** desarrollando Componentes $\mapsto$ Modelo de Implementación
5. **Prueba** y escenarios de verificación $\mapsto$ Modelo de Prueba

Flujos de trabajo

Existen además los siguientes flujos:

1. Modelado de negocio

Flujos de trabajo

Existen además los siguientes flujos:

1. Modelado de negocio
2. Captura de requisitos

Flujos de trabajo

Existen además los siguientes flujos:

1. Modelado de negocio
2. Captura de requisitos
3. Gestión del proyecto

Flujos de trabajo

Existen además los siguientes flujos:

1. Modelado de negocio
2. Captura de requisitos
3. Gestión del proyecto
4. Configuración y cambio

Flujos de trabajo

Existen además los siguientes flujos:

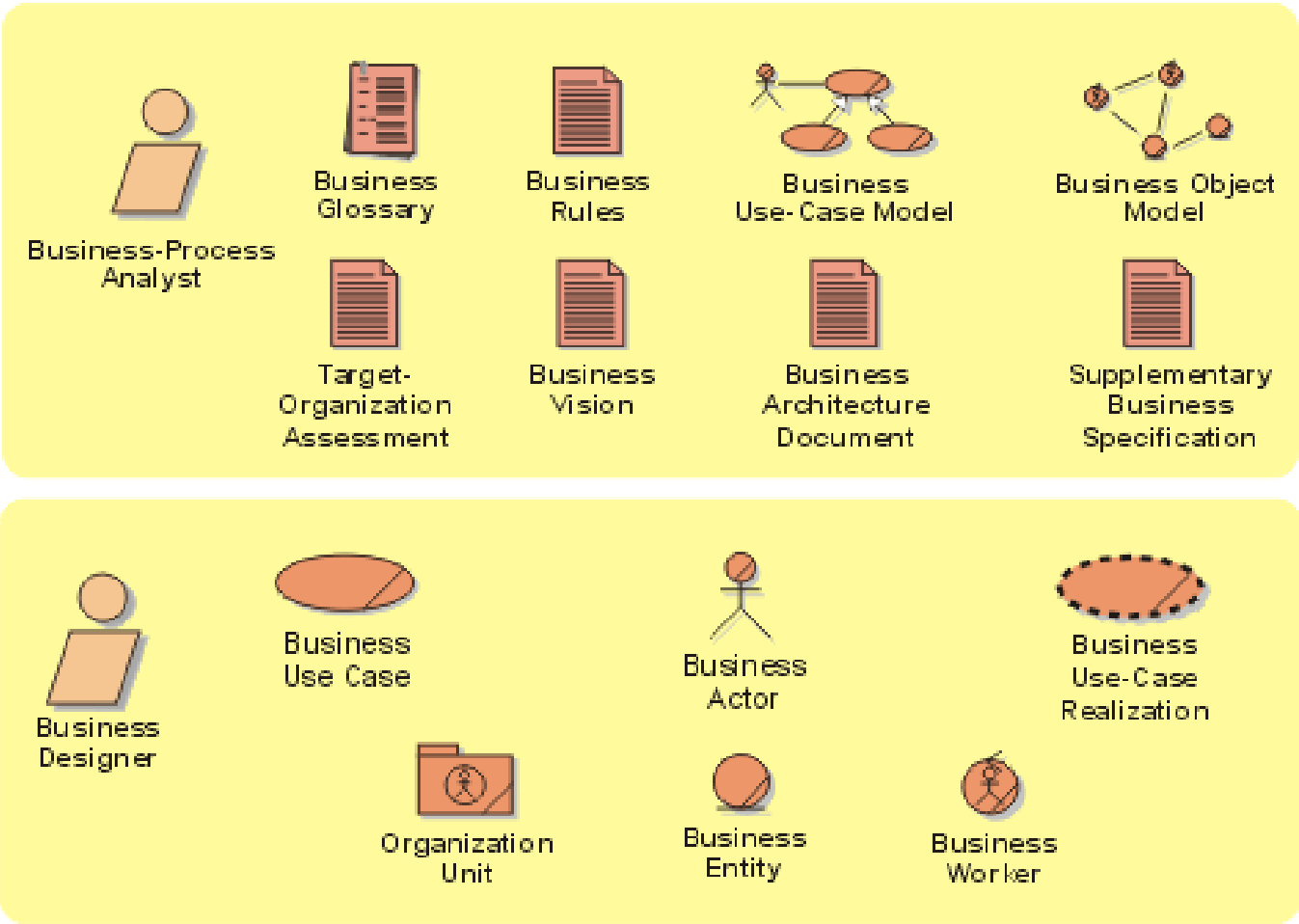
1. Modelado de negocio
2. Captura de requisitos
3. Gestión del proyecto
4. Configuración y cambio
5. Despliegue

Flujos de trabajo

Existen además los siguientes flujos:

1. Modelado de negocio
2. Captura de requisitos
3. Gestión del proyecto
4. Configuración y cambio
5. Despliegue
6. Entorno

Modelado de Negocio



Modelado de Negocio

- El objetivo es **comprender** la estructura y dinámica de la **organización** que requiere el software, asegurar que clientes, usuarios finales, y desarrolladores tienen un entendimiento común, comprender problemas e identificar potenciales mejoras, y derivar los requerimientos para el sistema

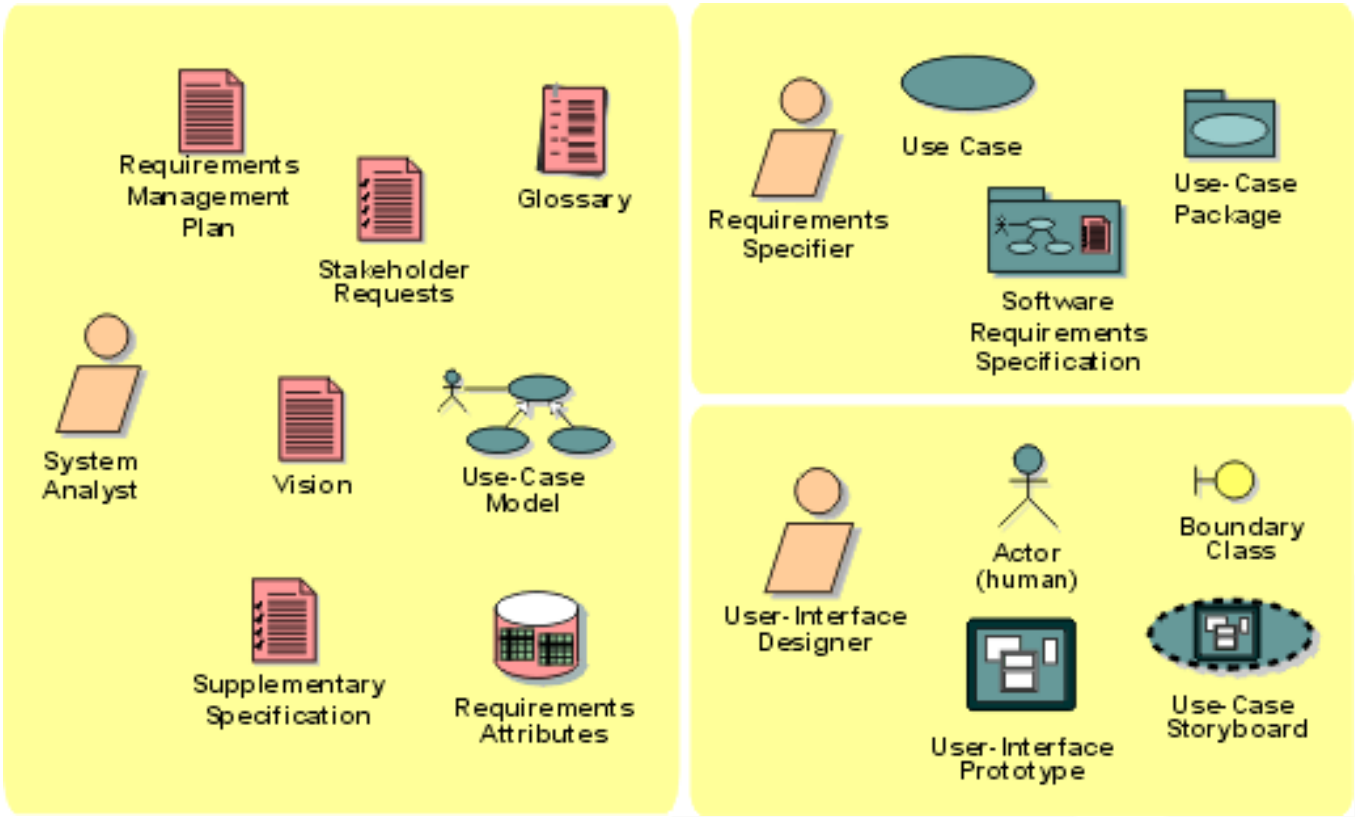
Modelado de Negocio

- El objetivo es **comprender** la estructura y dinámica de la **organización** que requiere el software, asegurar que clientes, usuarios finales, y desarrolladores tienen un entendimiento común, comprender problemas e identificar potenciales mejoras, y derivar los requerimientos para el sistema
- Plantea también que el esfuerzo de modelado del negocio puede tener distinto **alcance** dependiendo del contexto y **necesidades** de la organización

Modelado de Negocio

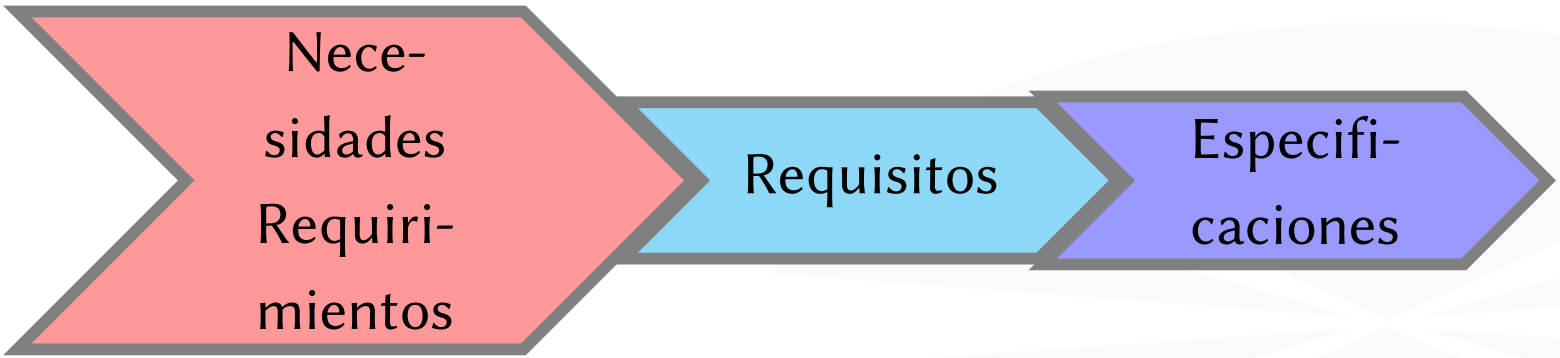
- El objetivo es **comprender** la estructura y dinámica de la **organización** que requiere el software, asegurar que clientes, usuarios finales, y desarrolladores tienen un entendimiento común, comprender problemas e identificar potenciales mejoras, y derivar los requerimientos para el sistema
- Plantea también que el esfuerzo de modelado del negocio puede tener distinto **alcance** dependiendo del contexto y **necesidades** de la organización
- Como elementos para modelar los procesos del negocio, propone los **Casos de Uso del Negocio** descripción textual y los Diagramas de Actividad como notación gráfica para los mismos

Captura de Requisitos



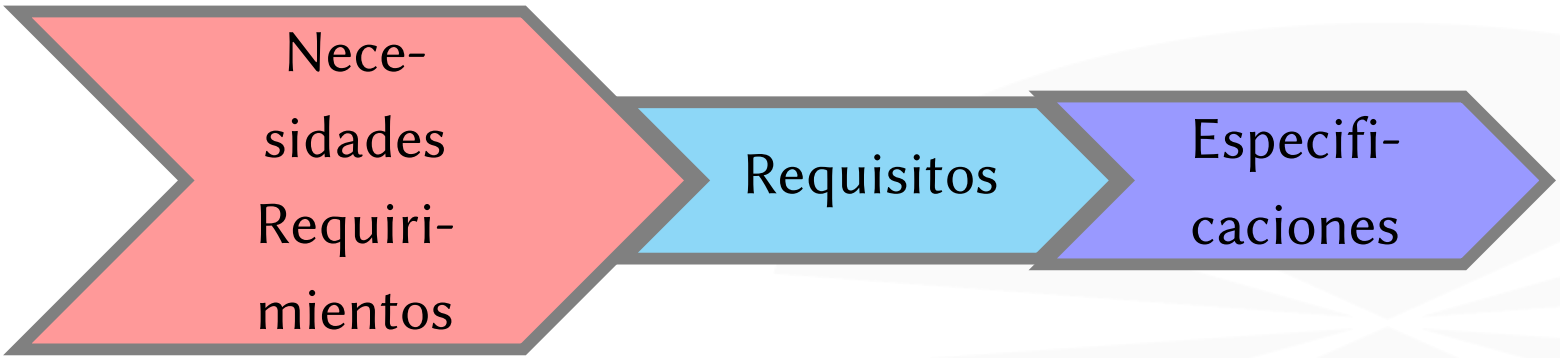
Captura de Requisitos

- Los objetivos de la captura de requisitos son:



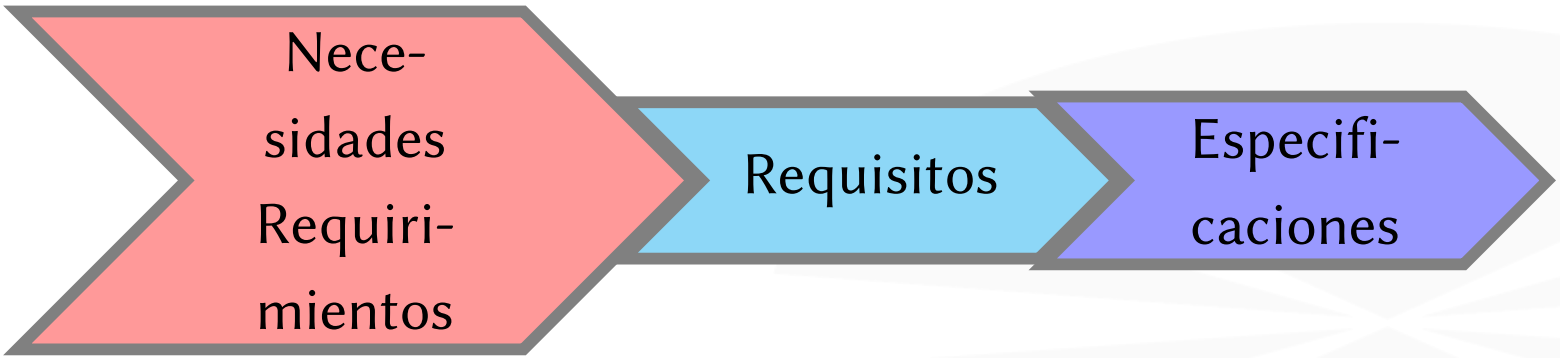
Captura de Requisitos

- Los objetivos de la captura de requisitos son:
  - **Encontrar los verdaderos requisitos**:



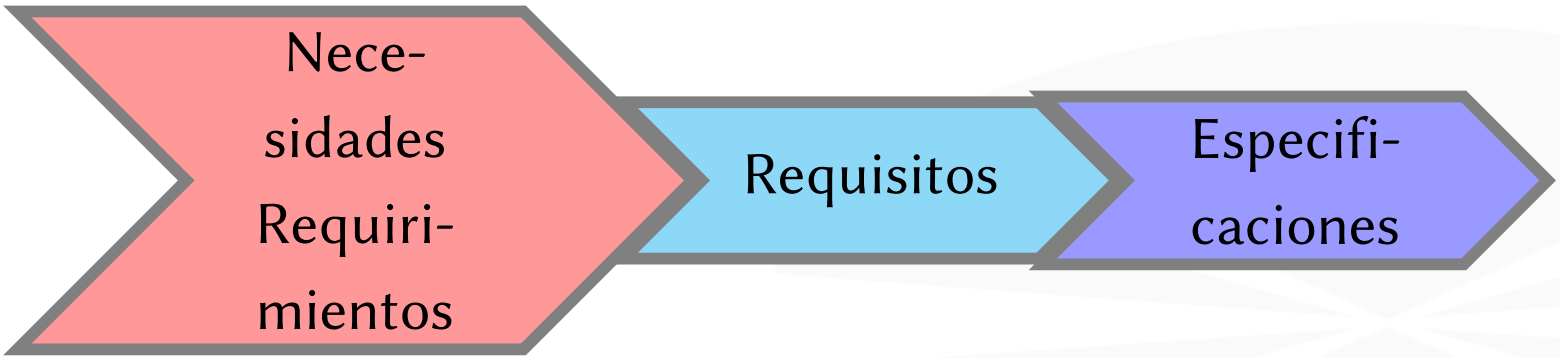
Captura de Requisitos

- Los objetivos de la captura de requisitos son:
  - **Encontrar los verdaderos requisitos**:
    - Su implementación añadirán el valor esperado por los usuarios y estará globado en los objetivos de la organización



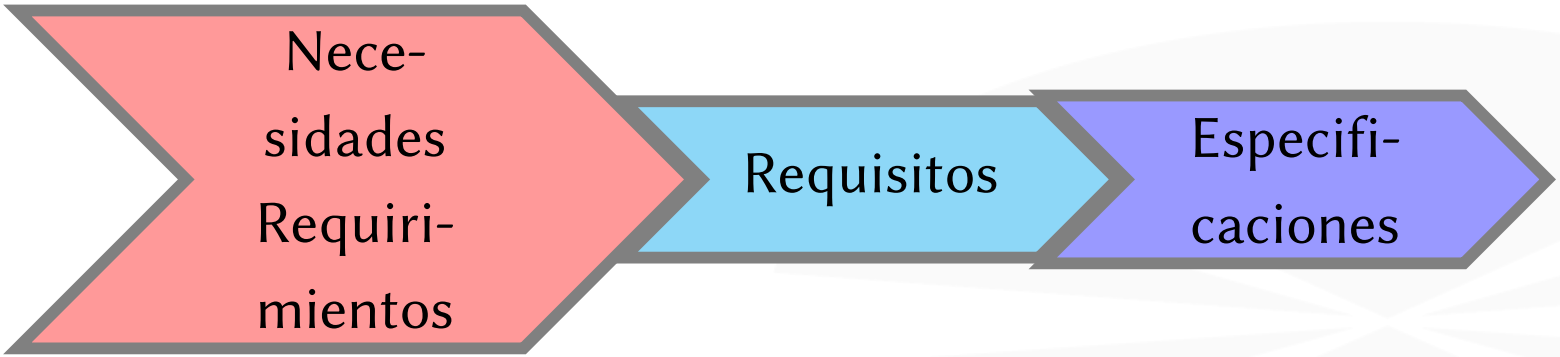
Captura de Requisitos

- Los objetivos de la captura de requisitos son:
  - **Encontrar los verdaderos requisitos**:
    - Su implementación añadirán el valor esperado por los usuarios y estará globado en los objetivos de la organización
  - **Representarlos adecuadamente para usuarios, clientes y desarrolladores**:



Captura de Requisitos

- Los objetivos de la captura de requisitos son:
  - **Encontrar los verdaderos requisitos**:
    - Su implementación añadirán el valor esperado por los usuarios y estará globado en los objetivos de la organización
  - **Representarlos adecuadamente para usuarios, clientes y desarrolladores**:
    - La descripción será comprensible y bien estructurada



Flujos de trabajo

- Un sistema puede tener diferentes tipos de usuarios:

Flujos de trabajo

- Un sistema puede tener diferentes tipos de usuarios:
  - **Cada usuario se representa como un actor**:

Flujos de trabajo

- Un sistema puede tener diferentes tipos de usuarios:
  - **Cada usuario se representa como un actor**:
    - Utilizan el sistema interactuando con los casos de uso

Flujos de trabajo

- Un sistema puede tener diferentes tipos de usuarios:
  - **Cada usuario se representa como un actor**:
    - Utilizan el sistema interactuando con los casos de uso
  - **Un caso de uso es una secuencia de acciones**:

Flujos de trabajo

- Un sistema puede tener diferentes tipos de usuarios:
  - **Cada usuario se representa como un actor**:
    - Utilizan el sistema interactuando con los casos de uso
  - **Un caso de uso es una secuencia de acciones**:
    - El sistema las procesa para ofrecer un resultado de valor a un actor

Flujos de trabajo

- Un sistema puede tener diferentes tipos de usuarios:
  - **Cada usuario se representa como un actor**:
    - Utilizan el sistema interactuando con los casos de uso
  - **Un caso de uso es una secuencia de acciones**:
    - El sistema las procesa para ofrecer un resultado de valor a un actor
- El modelo de casos de uso se transforma en un modelo de diseño a través de un modelo de análisis

Flujos de trabajo

- Un sistema puede tener diferentes tipos de usuarios:
  - **Cada usuario se representa como un actor**:
    - Utilizan el sistema interactuando con los casos de uso
  - **Un caso de uso es una secuencia de acciones**:
    - El sistema las procesa para ofrecer un resultado de valor a un actor
- El modelo de casos de uso se transforma en un modelo de diseño a través de un modelo de análisis
  - Los Modelos de Análisis y Diseño son estructuras compuestas por:

Flujos de trabajo

- Un sistema puede tener diferentes tipos de usuarios:
  - **Cada usuario se representa como un actor**:
    - Utilizan el sistema interactuando con los casos de uso
  - **Un caso de uso es una secuencia de acciones**:
    - El sistema las procesa para ofrecer un resultado de valor a un actor
- El modelo de casos de uso se transforma en un modelo de diseño a través de un modelo de análisis
  - Los Modelos de Análisis y Diseño son estructuras compuestas por:
    - Clasificadores

Flujos de trabajo

- Un sistema puede tener diferentes tipos de usuarios:
  - **Cada usuario se representa como un actor**:
    - Utilizan el sistema interactuando con los casos de uso
  - **Un caso de uso es una secuencia de acciones**:
    - El sistema las procesa para ofrecer un resultado de valor a un actor
- El modelo de casos de uso se transforma en un modelo de diseño a través de un modelo de análisis
  - Los Modelos de Análisis y Diseño son estructuras compuestas por:
    - Clasificadores
    - Realizaciones que explican como la estructura resuelve los casos de uso

Flujos de trabajo

- Un sistema puede tener diferentes tipos de usuarios:
  - **Cada usuario se representa como un actor**:
    - Utilizan el sistema interactuando con los casos de uso
  - **Un caso de uso es una secuencia de acciones**:
    - El sistema las procesa para ofrecer un resultado de valor a un actor
- El modelo de casos de uso se transforma en un modelo de diseño a través de un modelo de análisis
  - Los Modelos de Análisis y Diseño son estructuras compuestas por:
    - Clasificadores
    - Realizaciones que explican como la estructura resuelve los casos de uso
  - El comportamiento **profundo** de los clasificadores se puede describir con DTE's (State Transition Diagram)

Modelo de Análisis

El modelo de análisis es:

- La especificación **detallada** de los requisitos

Modelo de Análisis

El modelo de análisis es:

- La especificación **detallada** de los requisitos
- La primera **aproximación** al modelo de diseño

Modelo de Análisis

El modelo de análisis es:

- La especificación **detallada** de los requisitos
- La primera **aproximación** al modelo de diseño
- Un modelo con entidad propia que

Modelo de Análisis

El modelo de análisis es:

- La especificación **detallada** de los requisitos
- La primera **aproximación** al modelo de diseño
- Un modelo con entidad propia que
  - Refina los casos de uso en **colaboraciones** generales entre **clasificadores** conceptuales

Modelo de Análisis

El modelo de análisis es:

- La especificación **detallada** de los requisitos
- La primera **aproximación** al modelo de diseño
- Un modelo con entidad propia que
  - Refina los casos de uso en **colaboraciones** generales entre **clasificadores** conceptuales
  - Puede reutilizar **componentes** robustos

Modelo de Análisis

El modelo de análisis es:

- La especificación **detallada** de los requisitos
- La primera **aproximación** al modelo de diseño
- Un modelo con entidad propia que
  - Refina los casos de uso en **colaboraciones** generales entre **clasificadores** conceptuales
  - Puede reutilizar **componentes** robustos
- Un Modelo de Analisis puede ser **transitorio**, pero dependiendo de la complejidad del sistema, debería **mantenerse** durante toda su vida

Modelo de Análisis

El modelo de análisis es:

- La especificación **detallada** de los requisitos
- La primera **aproximación** al modelo de diseño
- Un modelo con entidad propia que
  - Refina los casos de uso en **colaboraciones** generales entre **clasificadores** conceptuales
  - Puede reutilizar **componentes** robustos
- Un Modelo de Analisis puede ser **transitorio**, pero dependiendo de la complejidad del sistema, debería **mantenerse** durante toda su vida
- Los desarrolladores asignan las **responsabilidades** de los casos de uso a las clases del dominio que los realizan

Modelo de Diseño

El modelo de diseño:

- Es jerárquico y contiene relaciones que atraviesan la jerarquía

Modelo de Diseño

El modelo de diseño:

- Es jerárquico y contiene relaciones que atraviesan la jerarquía
  - **Asociaciones**

Modelo de Diseño

El modelo de diseño:

- Es jerárquico y contiene relaciones que atraviesan la jerarquía
  - **Asociaciones**
  - **Generalizaciones**

Modelo de Diseño

El modelo de diseño:

- Es jerárquico y contiene relaciones que atraviesan la jerarquía
  - **Asociaciones**
  - **Generalizaciones**
  - **Dependencias**

Modelo de Diseño

El modelo de diseño:

- Es jerárquico y contiene relaciones que atraviesan la jerarquía
  - **Asociaciones**
  - **Generalizaciones**
  - **Dependencias**
- Las realizaciones de los casos de uso se explicitan con **estereotipos**

Modelo de Diseño

El modelo de diseño:

- Es jerárquico y contiene relaciones que atraviesan la jerarquía
  - **Asociaciones**
  - **Generalizaciones**
  - **Dependencias**
- Las realizaciones de los casos de uso se explicitan con **estereotipos**
- Se detallan las **colaboraciones**

Modelo de Diseño

El modelo de diseño:

- Es jerárquico y contiene relaciones que atraviesan la jerarquía
  - **Asociaciones**
  - **Generalizaciones**
  - **Dependencias**
- Las realizaciones de los casos de uso se explicitan con **estereotipos**
- Se detallan las **colaboraciones**
  - Representan la **participación** y la de los **clasificadores**

Modelo de Diseño

El modelo de diseño:

- Es jerárquico y contiene relaciones que atraviesan la jerarquía
  - **Asociaciones**
  - **Generalizaciones**
  - **Dependencias**
- Las realizaciones de los casos de uso se explicitan con **estereotipos**
- Se detallan las **colaboraciones**
  - Representan la **participación** y la de los **clasificadores**
  - Describen los roles

Modelo de Diseño

El modelo de diseño:

- Es jerárquico y contiene relaciones que atraviesan la jerarquía
  - **Asociaciones**
  - **Generalizaciones**
  - **Dependencias**
- Las realizaciones de los casos de uso se explicitan con **estereotipos**
- Se detallan las **colaboraciones**
  - Representan la **participación** y la de los **clasificadores**
  - Describen los roles
- Esquematiza inicialmente la futura implementación

Modelo de Diseño

El modelo de diseño:

- Es jerárquico y contiene relaciones que atraviesan la jerarquía
  - **Asociaciones**
  - **Generalizaciones**
  - **Dependencias**
- Las realizaciones de los casos de uso se explicitan con **estereotipos**
- Se detallan las **colaboraciones**
  - Representan la **participación** y la de los **clasificadores**
  - Describen los roles
- Esquematiza inicialmente la futura implementación
  - Correspondencia entre sus **susbsistemas** y los componentes del modelo de **implementación**

Modelo de Diseño

Los desarrolladores:

- Diseñan en detalle las **clases** para obtener el mejor rendimiento de los productos y tecnologías (Object Request Brokers, Frameworks de GUI y DBMS)

Modelo de Diseño

Los desarrolladores:

- Diseñan en detalle las **clases** para obtener el mejor rendimiento de los productos y tecnologías (Object Request Brokers, Frameworks de GUI y DBMS)
- Las agrupan en susbistemas

Modelo de Diseño

Los desarrolladores:

- Diseñan en detalle las **clases** para obtener el mejor rendimiento de los productos y tecnologías (Object Request Brokers, Frameworks de GUI y DBMS)
- Las agrupan en susbistemas
- Definen interfaces entre los **subsistemas**

Modelo de Diseño

Los desarrolladores:

- Diseñan en detalle las **clases** para obtener el mejor rendimiento de los productos y tecnologías (Object Request Brokers, Frameworks de GUI y DBMS)
- Las agrupan en susbistemas
- Definen interfaces entre los **subsistemas**
- Orientan el Modelo de **Despliegue**

Modelo de Diseño

Los desarrolladores:

- Diseñan en detalle las **clases** para obtener el mejor rendimiento de los productos y tecnologías (Object Request Brokers, Frameworks de GUI y DBMS)
- Las agrupan en susbistemas
- Definen interfaces entre los **subsistemas**
- Orientan el Modelo de **Despliegue**
  - Definen la organización **física** en términos de nodos de cómputo

Modelo de Diseño

Los desarrolladores:

- Diseñan en detalle las **clases** para obtener el mejor rendimiento de los productos y tecnologías (Object Request Brokers, Frameworks de GUI y DBMS)
- Las agrupan en susbistemas
- Definen interfaces entre los **subsistemas**
- Orientan el Modelo de **Despliegue**
  - Definen la organización **física** en términos de nodos de cómputo
  - **Verifican** que los Casos de Uso puedan, en forma de **componentes**, implementarse y ejecutarse en esos nodos

Modelo de Diseño

Los desarrolladores:

- Diseñan en detalle las **clases** para obtener el mejor rendimiento de los productos y tecnologías (Object Request Brokers, Frameworks de GUI y DBMS)
- Las agrupan en susbistemas
- Definen interfaces entre los **subsistemas**
- Orientan el Modelo de **Despliegue**
  - Definen la organización **física** en términos de nodos de cómputo
  - **Verifican** que los Casos de Uso puedan, en forma de **componentes**, implementarse y ejecutarse en esos nodos
- Llegado a este **punto** del estado del proyecto, se inicia el desarrollo del Modelo de **Implementación**

Modelo de Implementación

Se realizan las clases:

- Declaración de **atributos**

Modelo de Implementación

Se realizan las clases:

- Declaración de **atributos**
- **Lógica** de los métodos

Modelo de Implementación

Se realizan las clases:

- Declaración de **atributos**
- **Lógica** de los métodos
- Definición de los espacios, **esquemas** y tablas de BBDD

Modelo de Implementación

Se realizan las clases:

- Declaración de **atributos**
- **Lógica** de los métodos
- Definición de los espacios, **esquemas** y tablas de BBDD
- Determinación de **DAOs** (si se decide esta capa)

Modelo de Implementación

Se realizan las clases:

- Declaración de **atributos**
- **Lógica** de los métodos
- Definición de los espacios, **esquemas** y tablas de BBDD
- Determinación de **DAOs** (si se decide esta capa)
- **Codificación** de métodos

Modelo de Implementación

Se realizan las clases:

- Declaración de **atributos**
- **Lógica** de los métodos
- Definición de los espacios, **esquemas** y tablas de BBDD
- Determinación de **DAOs** (si se decide esta capa)
- **Codificación** de métodos

Modelo de Implementación

Se realizan las clases:

- Declaración de **atributos**
- **Lógica** de los métodos
- Definición de los espacios, **esquemas** y tablas de BBDD
- Determinación de **DAOs** (si se decide esta capa)
- **Codificación** de métodos

Los casos de uso orientan a los desarrolladores, en esta pauta, para determinar el **orden** de implementación e integración de los componentes

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:
  - Se comprueba la **utilidad** de los servicios del sistema

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:
  - Se comprueba la **utilidad** de los servicios del sistema
    - **Cobertura** de la funcionalidad descrita en los casos de uso

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:
  - Se comprueba la **utilidad** de los servicios del sistema
    - **Cobertura** de la funcionalidad descrita en los casos de uso
    - **Satisfacción** de los **requisitos** completos del sistema

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:
  - Se comprueba la **utilidad** de los servicios del sistema
    - **Cobertura** de la funcionalidad descrita en los casos de uso
    - **Satisfacción** de los **requisitos** completos del sistema
- Un caso de prueba

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:
  - Se comprueba la **utilidad** de los servicios del sistema
    - **Cobertura** de la funcionalidad descrita en los casos de uso
    - **Satisfacción** de los **requisitos** completos del sistema
- Un caso de prueba
  - Define una colección de **entradas**

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:
  - Se comprueba la **utilidad** de los servicios del sistema
    - **Cobertura** de la funcionalidad descrita en los casos de uso
    - **Satisfacción** de los **requisitos** completos del sistema
- Un caso de prueba
  - Define una colección de **entradas**
    - Interacciones

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:
  - Se comprueba la **utilidad** de los servicios del sistema
    - **Cobertura** de la funcionalidad descrita en los casos de uso
    - **Satisfacción** de los **requisitos** completos del sistema
- Un caso de prueba
  - Define una colección de **entradas**
    - Interacciones
    - Mensajes

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:
  - Se comprueba la **utilidad** de los servicios del sistema
    - **Cobertura** de la funcionalidad descrita en los casos de uso
    - **Satisfacción** de los **requisitos** completos del sistema
- Un caso de prueba
  - Define una colección de **entradas**
    - Interacciones
    - Mensajes
  - Determina condiciones de **ejecución**

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:
  - Se comprueba la **utilidad** de los servicios del sistema
    - **Cobertura** de la funcionalidad descrita en los casos de uso
    - **Satisfacción** de los **requisitos** completos del sistema
- Un caso de prueba
  - Define una colección de **entradas**
    - Interacciones
    - Mensajes
  - Determina condiciones de **ejecución**
    - Escenarios

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:
  - Se comprueba la **utilidad** de los servicios del sistema
    - **Cobertura** de la funcionalidad descrita en los casos de uso
    - **Satisfacción** de los **requisitos** completos del sistema
- Un caso de prueba
  - Define una colección de **entradas**
    - Interacciones
    - Mensajes
  - Determina condiciones de **ejecución**
    - Escenarios
    - Eventos y restricciones

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:
  - Se comprueba la **utilidad** de los servicios del sistema
    - **Cobertura** de la funcionalidad descrita en los casos de uso
    - **Satisfacción** de los **requisitos** completos del sistema
- Un caso de prueba
  - Define una colección de **entradas**
    - Interacciones
    - Mensajes
  - Determina condiciones de **ejecución**
    - Escenarios
    - Eventos y restricciones
  - Describe y tipifica los **resultados**

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:
  - Se comprueba la **utilidad** de los servicios del sistema
    - **Cobertura** de la funcionalidad descrita en los casos de uso
    - **Satisfacción** de los **requisitos** completos del sistema
- Un caso de prueba
  - Define una colección de **entradas**
    - Interacciones
    - Mensajes
  - Determina condiciones de **ejecución**
    - Escenarios
    - Eventos y restricciones
  - Describe y tipifica los **resultados**

Modelo de Prueba

Se realizan las clases:

- Con el modelo de Prueba, compuesto por **Casos de Prueba**:
  - Se comprueba la **utilidad** de los servicios del sistema
    - **Cobertura** de la funcionalidad descrita en los casos de uso
    - **Satisfacción** de los **requisitos** completos del sistema
- Un caso de prueba
  - Define una colección de **entradas**
    - Interacciones
    - Mensajes
  - Determina condiciones de **ejecución**
    - Escenarios
    - Eventos y restricciones
  - Describe y tipifica los **resultados**

Los **casos** de **prueba** se obtienen directamente de los casos de **uso**, con una correspondencia muy directa, existe pues entre ellos una **dependencia** de traza

Casos de Uso: ¿Por qué?

Un Caso de Uso ya es un concepto Universal:

- Proporcionan un medio sistemático de capturar requisitos funcionales

Casos de Uso: ¿Por qué?

Un Caso de Uso ya es un concepto Universal:

- Proporcionan un medio sistemático de capturar requisitos funcionales
  - **Intuitivo**

Casos de Uso: ¿Por qué?

Un Caso de Uso ya es un concepto Universal:

- Proporcionan un medio sistemático de capturar requisitos funcionales
  - **Intuitivo**
  - **Multidisciplinar**

Casos de Uso: ¿Por qué?

Un Caso de Uso ya es un concepto Universal:

- Proporcionan un medio sistemático de capturar requisitos funcionales
  - **Intuitivo**
  - **Multidisciplinar**
- Dirigen el Proceso de Desarrollo

Casos de Uso: ¿Por qué?

Un Caso de Uso ya es un concepto Universal:

- Proporcionan un medio sistemático de capturar requisitos funcionales
  - **Intuitivo**
  - **Multidisciplinar**
- Dirigen el Proceso de Desarrollo
  - El Análisis se desarrolla estudiando su **alcance** y la estructura

Casos de Uso: ¿Por qué?

Un Caso de Uso ya es un concepto Universal:

- Proporcionan un medio sistemático de capturar requisitos funcionales
  - **Intuitivo**
  - **Multidisciplinar**
- Dirigen el Proceso de Desarrollo
  - El Análisis se desarrolla estudiando su **alcance** y la estructura
  - El Diseño se **organiza** realizándolos para su soporte

Casos de Uso: ¿Por qué?

Un Caso de Uso ya es un concepto Universal:

- Proporcionan un medio sistemático de capturar requisitos funcionales
  - **Intuitivo**
  - **Multidisciplinar**
- Dirigen el Proceso de Desarrollo
  - El Análisis se desarrolla estudiando su **alcance** y la estructura
  - El Diseño se **organiza** realizándolos para su soporte
  - La Implementación y Prueba se **planifica** en función de ellos

Casos de Uso: ¿Por qué?

Un Caso de Uso ya es un concepto Universal:

- Proporcionan un medio sistemático de capturar requisitos funcionales
  - **Intuitivo**
  - **Multidisciplinar**
- Dirigen el Proceso de Desarrollo
  - El Análisis se desarrolla estudiando su **alcance** y la estructura
  - El Diseño se **organiza** realizándolos para su soporte
  - La Implementación y Prueba se **planifica** en función de ellos

Casos de Uso: ¿Por qué?

Un Caso de Uso ya es un concepto Universal:

- Proporcionan un medio sistemático de capturar requisitos funcionales
  - **Intuitivo**
  - **Multidisciplinar**
- Dirigen el Proceso de Desarrollo
  - El Análisis se desarrolla estudiando su **alcance** y la estructura
  - El Diseño se **organiza** realizándolos para su soporte
  - La Implementación y Prueba se **planifica** en función de ellos

Estos puntos quedan claramente explicitados cuando tenemos una **arquitectura estable**, esto será después del primer conjunto de iteraciones.

Casos de Uso: ¿Por qué?

> *“La perspectiva que proporcionan los **casos de uso** refuerza el objetivo último de la Ingeniería del Software: la creación de **productos** que permitan a los clientes realizar un **trabajo útil**”*
>
> *Karl Wiegers*

Casos de Uso: ¿Por qué?

- Un caso de uso **identifica** las facilidades del software que hay que lograr para cumplir los **objetivos** de la **organización** en general y en particular de los **usuarios**

Casos de Uso: ¿Por qué?

- Un caso de uso **identifica** las facilidades del software que hay que lograr para cumplir los **objetivos** de la **organización** en general y en particular de los **usuarios**
- Si un sistema se define pensando en **funcionalidades**, sin tener en cuenta los casos de uso, será **difícil**

Casos de Uso: ¿Por qué?

- Un caso de uso **identifica** las facilidades del software que hay que lograr para cumplir los **objetivos** de la **organización** en general y en particular de los **usuarios**
- Si un sistema se define pensando en **funcionalidades**, sin tener en cuenta los casos de uso, será **difícil**
  - Determinar si las funciones son **importantes** o incluso convenientes

Casos de Uso: ¿Por qué?

- Un caso de uso **identifica** las facilidades del software que hay que lograr para cumplir los **objetivos** de la **organización** en general y en particular de los **usuarios**
- Si un sistema se define pensando en **funcionalidades**, sin tener en cuenta los casos de uso, será **difícil**
  - Determinar si las funciones son **importantes** o incluso convenientes
  - Conocer a quien **ayudan** y que responsabilidades sustentan

Casos de Uso: ¿Por qué?

- Un caso de uso **identifica** las facilidades del software que hay que lograr para cumplir los **objetivos** de la **organización** en general y en particular de los **usuarios**
- Si un sistema se define pensando en **funcionalidades**, sin tener en cuenta los casos de uso, será **difícil**
  - Determinar si las funciones son **importantes** o incluso convenientes
  - Conocer a quien **ayudan** y que responsabilidades sustentan
  - Concretar que necesidades del negocio **satisfacen**

Casos de Uso: ¿Por qué?

- Un caso de uso **identifica** las facilidades del software que hay que lograr para cumplir los **objetivos** de la **organización** en general y en particular de los **usuarios**
- Si un sistema se define pensando en **funcionalidades**, sin tener en cuenta los casos de uso, será **difícil**
  - Determinar si las funciones son **importantes** o incluso convenientes
  - Conocer a quien **ayudan** y que responsabilidades sustentan
  - Concretar que necesidades del negocio **satisfacen**
  - Conocer si potencian el negocio y cuanto **valor** aportan

Casos de Uso: ¿Por qué?

- La experiencia ha permitido llegar a la conclusión de que trabajar *elucubrando* sobre lo que **podría ofrecer** un sistema no ayuda a obtener ni respuestas ni definiciones correctas

Casos de Uso: ¿Por qué?

- La experiencia ha permitido llegar a la conclusión de que trabajar *elucubrando* sobre lo que **podría ofrecer** un sistema no ayuda a obtener ni respuestas ni definiciones correctas
  - Se obtienen listas de **funciones** aparentemente **valiosas**, pero...

Casos de Uso: ¿Por qué?

- La experiencia ha permitido llegar a la conclusión de que trabajar *elucubrando* sobre lo que **podría ofrecer** un sistema no ayuda a obtener ni respuestas ni definiciones correctas
  - Se obtienen listas de **funciones** aparentemente **valiosas**, pero...
  - ... cuando se examinan en profundidad se ve que no están **ajustadas** al **comportamiento** eficiente y racionalizado de la **organización** y a las **necesidades** específicas de los diferentes perfiles de **usuario**

Casos de Uso: ¿Por qué?

- La experiencia ha permitido llegar a la conclusión de que trabajar *elucubrando* sobre lo que **podría ofrecer** un sistema no ayuda a obtener ni respuestas ni definiciones correctas
  - Se obtienen listas de **funciones** aparentemente **valiosas**, pero...
  - ... cuando se examinan en profundidad se ve que no están **ajustadas** al **comportamiento** eficiente y racionalizado de la **organización** y a las **necesidades** específicas de los diferentes perfiles de **usuario**
- Sistemas con **funcionalidades sin sentido** o justificación práctica son muchas veces **perjudiciales** para el trabajo eficiente de la organización.

Casos de Uso: ¿Por qué?

- Los casos de uso son **intuitivos**:

Casos de Uso: ¿Por qué?

- Los casos de uso son **intuitivos**:
  - No requieren para su **explicitación** de conceptos y notaciones complejas

Casos de Uso: ¿Por qué?

- Los casos de uso son **intuitivos**:
  - No requieren para su **explicitación** de conceptos y notaciones complejas
  - Se pueden redactar en **lenguaje natural**

Casos de Uso: ¿Por qué?

- Los casos de uso son **intuitivos**:
  - No requieren para su **explicitación** de conceptos y notaciones complejas
  - Se pueden redactar en **lenguaje natural**
  - Facilitan la **lectura** de la estructura **funcional**, la **evaluación** de la **oportunidad** y la determinación del **alcance** y propuesta de cambios

Casos de Uso: ¿Por qué?

- Los casos de uso son **intuitivos**:
  - No requieren para su **explicitación** de conceptos y notaciones complejas
  - Se pueden redactar en **lenguaje natural**
  - Facilitan la **lectura** de la estructura **funcional**, la **evaluación** de la **oportunidad** y la determinación del **alcance** y propuesta de cambios
- La captura de los casos de uso **implica** a:

Casos de Uso: ¿Por qué?

- Los casos de uso son **intuitivos**:
  - No requieren para su **explicitación** de conceptos y notaciones complejas
  - Se pueden redactar en **lenguaje natural**
  - Facilitan la **lectura** de la estructura **funcional**, la **evaluación** de la **oportunidad** y la determinación del **alcance** y propuesta de cambios
- La captura de los casos de uso **implica** a:
  - **Clientes** y usuarios en el rol de expertos en requerimientos y requisitos

Casos de Uso: ¿Por qué?

- Los casos de uso son **intuitivos**:
  - No requieren para su **explicitación** de conceptos y notaciones complejas
  - Se pueden redactar en **lenguaje natural**
  - Facilitan la **lectura** de la estructura **funcional**, la **evaluación** de la **oportunidad** y la determinación del **alcance** y propuesta de cambios
- La captura de los casos de uso **implica** a:
  - **Clientes** y usuarios en el rol de expertos en requerimientos y requisitos
  - **Analistas**, para ayudar a los usuarios y clientes a comunicar sus responsabilidades y ponderarlas en el contexto global del alcance del sistema

Casos de Uso: ¿Por qué?

- Los casos de uso son **intuitivos**:
  - No requieren para su **explicitación** de conceptos y notaciones complejas
  - Se pueden redactar en **lenguaje natural**
  - Facilitan la **lectura** de la estructura **funcional**, la **evaluación** de la **oportunidad** y la determinación del **alcance** y propuesta de cambios
- La captura de los casos de uso **implica** a:
  - **Clientes** y usuarios en el rol de expertos en requerimientos y requisitos
  - **Analistas**, para ayudar a los usuarios y clientes a comunicar sus responsabilidades y ponderarlas en el contexto global del alcance del sistema
  - **Desarrolladores** en general, para organizar el sistema y la funcionalidad y facilitar su discusión

Casos de Uso: ¿Por qué?

> *“Un modelo de casos de uso debe ser la base del «contrato» de desarrollo con un cliente”*

Casos de Uso: ¿Por qué?

Dirigen el Proceso:

- Los flujos de trabajo que regulan el progreso de un proyecto, se inician en los casos de uso

Casos de Uso: ¿Por qué?

Dirigen el Proceso:

- Los flujos de trabajo que regulan el progreso de un proyecto, se inician en los casos de uso
  - Permiten encontrar las clases **colaboradoras**

Casos de Uso: ¿Por qué?

Dirigen el Proceso:

- Los flujos de trabajo que regulan el progreso de un proyecto, se inician en los casos de uso
  - Permiten encontrar las clases **colaboradoras**
  - Facilitan el desarrollo de los **interfaces** de usuario

Casos de Uso: ¿Por qué?

Dirigen el Proceso:

- Los flujos de trabajo que regulan el progreso de un proyecto, se inician en los casos de uso
  - Permiten encontrar las clases **colaboradoras**
  - Facilitan el desarrollo de los **interfaces** de usuario
- Los casos de uso ayudan a los jefes de Proyecto a **planificar** y **controlar** tareas:

Casos de Uso: ¿Por qué?

Dirigen el Proceso:

- Los flujos de trabajo que regulan el progreso de un proyecto, se inician en los casos de uso
  - Permiten encontrar las clases **colaboradoras**
  - Facilitan el desarrollo de los **interfaces** de usuario
- Los casos de uso ayudan a los jefes de Proyecto a **planificar** y **controlar** tareas:
  - Cada caso de uso puede considerarse una **tarea** de desarrollo y su desglose estructurado, un desglose estructurado de tareas

Casos de Uso: ¿Por qué?

Dirigen el Proceso:

- Los flujos de trabajo que regulan el progreso de un proyecto, se inician en los casos de uso
  - Permiten encontrar las clases **colaboradoras**
  - Facilitan el desarrollo de los **interfaces** de usuario
- Los casos de uso ayudan a los jefes de Proyecto a **planificar** y **controlar** tareas:
  - Cada caso de uso puede considerarse una **tarea** de desarrollo y su desglose estructurado, un desglose estructurado de tareas
  - Los casos de uso permiten la **estimación** de esfuerzos, medios y recursos necesarios para realizarlos

Casos de Uso: ¿Por qué?

Dirigen el Proceso:

- Los flujos de trabajo que regulan el progreso de un proyecto, se inician en los casos de uso
  - Permiten encontrar las clases **colaboradoras**
  - Facilitan el desarrollo de los **interfaces** de usuario
- Los casos de uso ayudan a los jefes de Proyecto a **planificar** y **controlar** tareas:
  - Cada caso de uso puede considerarse una **tarea** de desarrollo y su desglose estructurado, un desglose estructurado de tareas
  - Los casos de uso permiten la **estimación** de esfuerzos, medios y recursos necesarios para realizarlos
  - Los casos de uso permiten asignar **responsables**

Casos de Uso: ¿Por qué?

Dirigen el Proceso:

- Los flujos de trabajo que regulan el progreso de un proyecto, se inician en los casos de uso
  - Permiten encontrar las clases **colaboradoras**
  - Facilitan el desarrollo de los **interfaces** de usuario
- Los casos de uso ayudan a los jefes de Proyecto a **planificar** y **controlar** tareas:
  - Cada caso de uso puede considerarse una **tarea** de desarrollo y su desglose estructurado, un desglose estructurado de tareas
  - Los casos de uso permiten la **estimación** de esfuerzos, medios y recursos necesarios para realizarlos
  - Los casos de uso permiten asignar **responsables**

Casos de Uso: ¿Por qué?

Dirigen el Proceso:

- Los flujos de trabajo que regulan el progreso de un proyecto, se inician en los casos de uso
  - Permiten encontrar las clases **colaboradoras**
  - Facilitan el desarrollo de los **interfaces** de usuario
- Los casos de uso ayudan a los jefes de Proyecto a **planificar** y **controlar** tareas:
  - Cada caso de uso puede considerarse una **tarea** de desarrollo y su desglose estructurado, un desglose estructurado de tareas
  - Los casos de uso permiten la **estimación** de esfuerzos, medios y recursos necesarios para realizarlos
  - Los casos de uso permiten asignar **responsables**

Dan pues **trazabilidad** entre los distintos **modelos** lo que permite **mantener** el sistema ante un **cambio** de requisitos

Casos de Uso: ¿Por qué?

Para enfocar, explicar y valorar la arquitectura:

- Una arquitectura estable se logra en las primeras iteraciones persiguiendo la utilidad manifiesta del sistema

Casos de Uso: ¿Por qué?

Para enfocar, explicar y valorar la arquitectura:

- Una arquitectura estable se logra en las primeras iteraciones persiguiendo la utilidad manifiesta del sistema
- De este manera se otorga:

Casos de Uso: ¿Por qué?

Para enfocar, explicar y valorar la arquitectura:

- Una arquitectura estable se logra en las primeras iteraciones persiguiendo la utilidad manifiesta del sistema
- De este manera se otorga:
  - **Significado** *arquitectónico* a los casos de uso

Casos de Uso: ¿Por qué?

Para enfocar, explicar y valorar la arquitectura:

- Una arquitectura estable se logra en las primeras iteraciones persiguiendo la utilidad manifiesta del sistema
- De este manera se otorga:
  - **Significado** *arquitectónico* a los casos de uso
  - Fundamento a los **ciclos posteriores** del desarrollo

Casos de Uso: ¿Por qué?

Para enfocar, explicar y valorar la arquitectura:

- Una arquitectura estable se logra en las primeras iteraciones persiguiendo la utilidad manifiesta del sistema
- De este manera se otorga:
  - **Significado** *arquitectónico* a los casos de uso
  - Fundamento a los **ciclos posteriores** del desarrollo
- El punto de partida de la **documentación** de usuario son los casos de uso

Casos de Uso: ¿Por qué?

Para enfocar, explicar y valorar la arquitectura:

- Una arquitectura estable se logra en las primeras iteraciones persiguiendo la utilidad manifiesta del sistema
- De este manera se otorga:
  - **Significado** *arquitectónico* a los casos de uso
  - Fundamento a los **ciclos posteriores** del desarrollo
- El punto de partida de la **documentación** de usuario son los casos de uso
  - Organizan el **manual** de explotación

Casos de Uso: ¿Por qué?

Para enfocar, explicar y valorar la arquitectura:

- Una arquitectura estable se logra en las primeras iteraciones persiguiendo la utilidad manifiesta del sistema
- De este manera se otorga:
  - **Significado** *arquitectónico* a los casos de uso
  - Fundamento a los **ciclos posteriores** del desarrollo
- El punto de partida de la **documentación** de usuario son los casos de uso
  - Organizan el **manual** de explotación
  - Permiten describir las **instrucciones de uso**

Casos de Uso: Captura

- La primera parte del Flujo de Trabajo captura los requisitos **funcionales**

Casos de Uso: Captura

- La primera parte del Flujo de Trabajo captura los requisitos **funcionales**
  - Se expresan como casos de uso

Casos de Uso: Captura

- La primera parte del Flujo de Trabajo captura los requisitos **funcionales**
  - Se expresan como casos de uso
- También es necesario determinar algún **otro tipo** de requisitos

Casos de Uso: Captura

- La primera parte del Flujo de Trabajo captura los requisitos **funcionales**
  - Se expresan como casos de uso
- También es necesario determinar algún **otro tipo** de requisitos
  - **Orientan** la realización de los funcionales

Casos de Uso: Captura

- La primera parte del Flujo de Trabajo captura los requisitos **funcionales**
  - Se expresan como casos de uso
- También es necesario determinar algún **otro tipo** de requisitos
  - **Orientan** la realización de los funcionales
  - **Contextualizan** tecnológicamente el problema

Casos de Uso: Captura

- La primera parte del Flujo de Trabajo captura los requisitos **funcionales**
  - Se expresan como casos de uso
- También es necesario determinar algún **otro tipo** de requisitos
  - **Orientan** la realización de los funcionales
  - **Contextualizan** tecnológicamente el problema
  - **Enfocan** la integración y el sincronismo con otros sistemas, si la hubiere

Casos de Uso: Captura

- La primera parte del Flujo de Trabajo captura los requisitos **funcionales**
  - Se expresan como casos de uso
- También es necesario determinar algún **otro tipo** de requisitos
  - **Orientan** la realización de los funcionales
  - **Contextualizan** tecnológicamente el problema
  - **Enfocan** la integración y el sincronismo con otros sistemas, si la hubiere
- Los requisitos **no funcionales** o bien se adjuntan a los casos de uso funcionales a los que afecta, o bien se refieren o describen aparte.

Casos de Uso: Captura

- Los usuarios de un sistema se representan mediante **actores**

Casos de Uso: Captura

- Los usuarios de un sistema se representan mediante **actores**
  - Hay muchos **tipos** de usuario

Casos de Uso: Captura

- Los usuarios de un sistema se representan mediante **actores**
  - Hay muchos **tipos** de usuario
  - Conviene estructurarlos en **jerarquías**

Casos de Uso: Captura

- Los usuarios de un sistema se representan mediante **actores**
  - Hay muchos **tipos** de usuario
  - Conviene estructurarlos en **jerarquías**
- Un diagrama de casos de Uso

Casos de Uso: Captura

- Los usuarios de un sistema se representan mediante **actores**
  - Hay muchos **tipos** de usuario
  - Conviene estructurarlos en **jerarquías**
- Un diagrama de casos de Uso
  - Muestra la **estructura** de todos los actores y casos de uso

Casos de Uso: Captura

- Los usuarios de un sistema se representan mediante **actores**
  - Hay muchos **tipos** de usuario
  - Conviene estructurarlos en **jerarquías**
- Un diagrama de casos de Uso
  - Muestra la **estructura** de todos los actores y casos de uso
  - Asocia pares actor/caso de uso con **interacciones**

Casos de Uso: Captura

- Hay varios tipos de **actores**

Casos de Uso: Captura

- Hay varios tipos de **actores**
  - **Personas**.

Casos de Uso: Captura

- Hay varios tipos de **actores**
  - **Personas**.
  - **Sistemas** y subsistemas

Casos de Uso: Captura

- Hay varios tipos de **actores**
  - **Personas**.
  - **Sistemas** y subsistemas
  - **Dispositivos**, redes y hardware externo que interactuará con el sistema

Casos de Uso: Captura

- Hay varios tipos de **actores**
  - **Personas**.
  - **Sistemas** y subsistemas
  - **Dispositivos**, redes y hardware externo que interactuará con el sistema
- Los actores asumen **roles**

Casos de Uso: Captura

- Hay varios tipos de **actores**
  - **Personas**.
  - **Sistemas** y subsistemas
  - **Dispositivos**, redes y hardware externo que interactuará con el sistema
- Los actores asumen **roles**
  - Cada actor puede asumir un **conjunto** coherente de roles

Casos de Uso: Captura

- Hay varios tipos de **actores**
  - **Personas**.
  - **Sistemas** y subsistemas
  - **Dispositivos**, redes y hardware externo que interactuará con el sistema
- Los actores asumen **roles**
  - Cada actor puede asumir un **conjunto** coherente de roles
  - Un usuario físico puede actuar como uno o **varios** actores en distintos **roles**

Casos de Uso: Captura

- Hay varios tipos de **actores**
  - **Personas**.
  - **Sistemas** y subsistemas
  - **Dispositivos**, redes y hardware externo que interactuará con el sistema
- Los actores asumen **roles**
  - Cada actor puede asumir un **conjunto** coherente de roles
  - Un usuario físico puede actuar como uno o **varios** actores en distintos **roles**
  - **Varios** usuarios concretos pueden interactuar como diferentes instancias u ocurrencias del **mismo** actor

Casos de Uso: Captura

- Hay varios tipos de **actores**
  - **Personas**.
  - **Sistemas** y subsistemas
  - **Dispositivos**, redes y hardware externo que interactuará con el sistema
- Los actores asumen **roles**
  - Cada actor puede asumir un **conjunto** coherente de roles
  - Un usuario físico puede actuar como uno o **varios** actores en distintos **roles**
  - **Varios** usuarios concretos pueden interactuar como diferentes instancias u ocurrencias del **mismo** actor
- Los actores se **comunican** con el sistema mediante mensajes

Casos de Uso: Captura

- Hay varios tipos de **actores**
  - **Personas**.
  - **Sistemas** y subsistemas
  - **Dispositivos**, redes y hardware externo que interactuará con el sistema
- Los actores asumen **roles**
  - Cada actor puede asumir un **conjunto** coherente de roles
  - Un usuario físico puede actuar como uno o **varios** actores en distintos **roles**
  - **Varios** usuarios concretos pueden interactuar como diferentes instancias u ocurrencias del **mismo** actor
- Los actores se **comunican** con el sistema mediante mensajes
  - La **separación** de la actividad de los actores y del sistema ayudará a definir las **responsabilidades** de ambos

Casos de Uso: Captura

- Hay varios tipos de **actores**
  - **Personas**.
  - **Sistemas** y subsistemas
  - **Dispositivos**, redes y hardware externo que interactuará con el sistema
- Los actores asumen **roles**
  - Cada actor puede asumir un **conjunto** coherente de roles
  - Un usuario físico puede actuar como uno o **varios** actores en distintos **roles**
  - **Varios** usuarios concretos pueden interactuar como diferentes instancias u ocurrencias del **mismo** actor
- Los actores se **comunican** con el sistema mediante mensajes
  - La **separación** de la actividad de los actores y del sistema ayudará a definir las **responsabilidades** de ambos
  - La **división** de las **responsabilidades** permitirá delimitar el **alcance** del sistema

Casos de Uso: Captura

> *Un caso de uso especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto*

- Un caso de uso es el modo en el que el **usuario** necesita **utilizar** el **sistema** para realizar su trabajo

Casos de Uso: Captura

> *Un caso de uso especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto*

- Un caso de uso es el modo en el que el **usuario** necesita **utilizar** el **sistema** para realizar su trabajo
- Esto lleva a una **estructura** de casos de uso basada en **objetivos**

Casos de Uso: Captura

> *Un caso de uso especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto*

- Un caso de uso es el modo en el que el **usuario** necesita **utilizar** el **sistema** para realizar su trabajo
- Esto lleva a una **estructura** de casos de uso basada en **objetivos**
- Cada modo de emplear el sistema es un **caso de uso candidato**, que

Casos de Uso: Captura

> *Un caso de uso especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto*

- Un caso de uso es el modo en el que el **usuario** necesita **utilizar** el **sistema** para realizar su trabajo
- Esto lleva a una **estructura** de casos de uso basada en **objetivos**
- Cada modo de emplear el sistema es un **caso de uso candidato**, que
  - Pueden **ampliarse**

Casos de Uso: Captura

> *“Un caso de uso especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto”*

- Un caso de uso es el modo en el que el **usuario** necesita **utilizar** el **sistema** para realizar su trabajo
- Esto lleva a una **estructura** de casos de uso basada en **objetivos**
- Cada modo de emplear el sistema es un **caso de uso candidato**, que
  - Pueden **ampliarse**
  - Pueden **cambiarse**

Casos de Uso: Captura

> *Un caso de uso especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto*

- Un caso de uso es el modo en el que el **usuario** necesita **utilizar** el **sistema** para realizar su trabajo
- Esto lleva a una **estructura** de casos de uso basada en **objetivos**
- Cada modo de emplear el sistema es un **caso de uso candidato**, que
  - Pueden **ampliarse**
  - Pueden **cambiarse**
  - Pueden **desglosarse** en casos de uso más pequeños

Casos de Uso: Captura

> *Un caso de uso especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto*

- Un caso de uso es el modo en el que el **usuario** necesita **utilizar** el **sistema** para realizar su trabajo
- Esto lleva a una **estructura** de casos de uso basada en **objetivos**
- Cada modo de emplear el sistema es un **caso de uso candidato**, que
  - Pueden **ampliarse**
  - Pueden **cambiarse**
  - Pueden **desglosarse** en casos de uso más pequeños
  - Pueden **integrarse**, por inclusión o extensión, en casos de uso más completos

Casos de Uso: Captura

- Una **instancia** de un caso de uso es una **ejecución** específica del mismo

Casos de Uso: Captura

- Una **instancia** de un caso de uso es una **ejecución** específica del mismo
  - Es una **secuencia** de **acciones** que determinan un camino

Casos de Uso: Captura

- Una **instancia** de un caso de uso es una **ejecución** específica del mismo
  - Es una **secuencia** de **acciones** que determinan un camino
  - Pueden darse varios caminos o **variantes**

Casos de Uso: Captura

- Una **instancia** de un caso de uso es una **ejecución** específica del mismo
  - Es una **secuencia** de **acciones** que determinan un camino
  - Pueden darse varios caminos o **variantes**
  - Se pueden **agrupar** variantes similares en un sólo caso de uso, con el fin práctico de lograr una explicación más **comprensible**

Casos de Uso: Captura

- Una **instancia** de un caso de uso es una **ejecución** específica del mismo
  - Es una **secuencia** de **acciones** que determinan un camino
  - Pueden darse varios caminos o **variantes**
  - Se pueden **agrupar** variantes similares en un sólo caso de uso, con el fin práctico de lograr una explicación más **comprensible**
- Los casos de uso pueden ser **contenedores** de requisitos no funcionales

Casos de Uso: Captura

- Una **instancia** de un caso de uso es una **ejecución** específica del mismo
  - Es una **secuencia** de **acciones** que determinan un camino
  - Pueden darse varios caminos o **variantes**
  - Se pueden **agrupar** variantes similares en un sólo caso de uso, con el fin práctico de lograr una explicación más **comprensible**
- Los casos de uso pueden ser **contenedores** de requisitos no funcionales
  - Requisitos de rendimiento

Casos de Uso: Captura

- Una **instancia** de un caso de uso es una **ejecución** específica del mismo
  - Es una **secuencia** de **acciones** que determinan un camino
  - Pueden darse varios caminos o **variantes**
  - Se pueden **agrupar** variantes similares en un sólo caso de uso, con el fin práctico de lograr una explicación más **comprensible**
- Los casos de uso pueden ser **contenedores** de requisitos no funcionales
  - Requisitos de rendimiento
  - Requisitos de disponibilidad

Casos de Uso: Captura

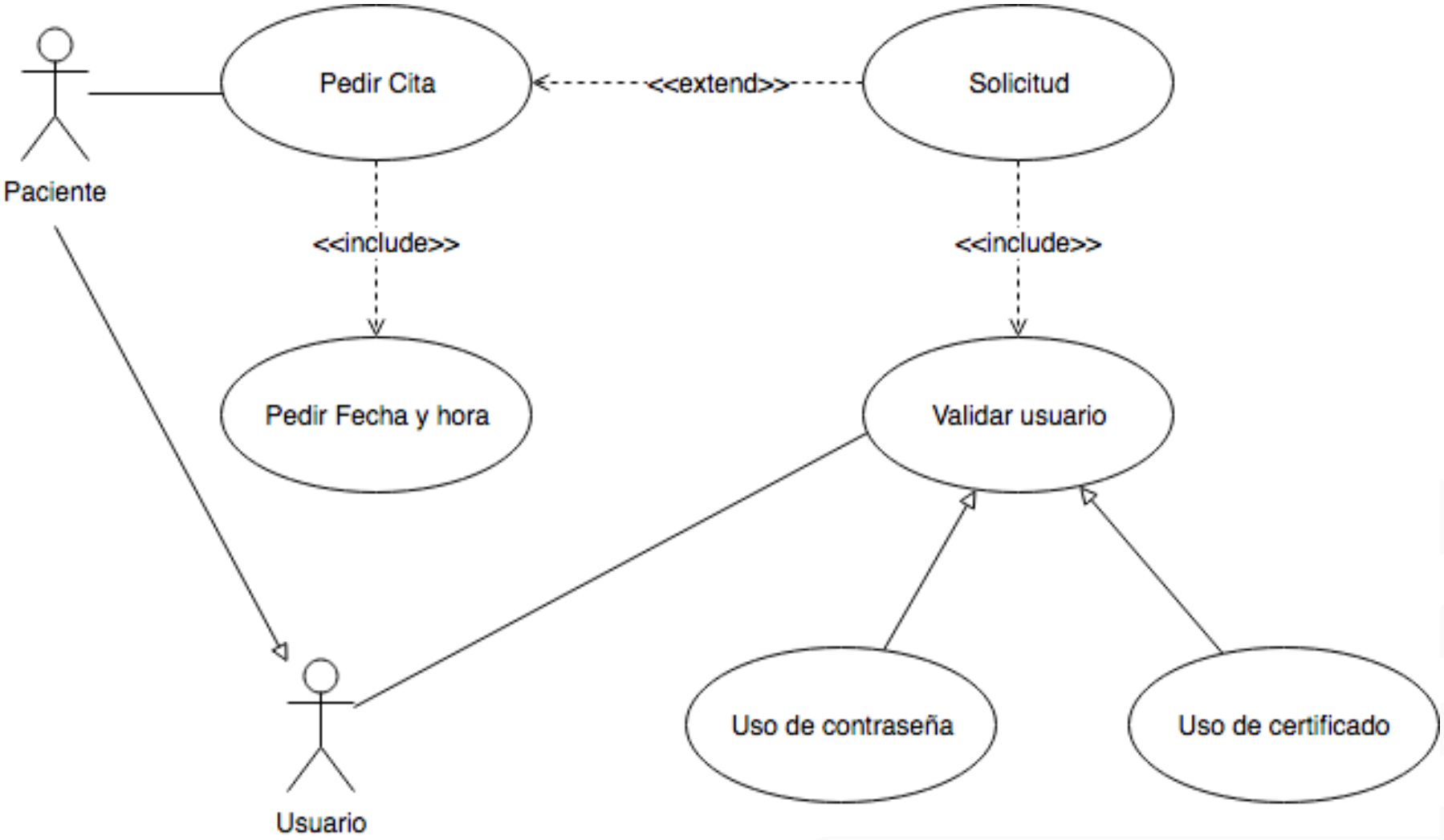
- Una **instancia** de un caso de uso es una **ejecución** específica del mismo
  - Es una **secuencia** de **acciones** que determinan un camino
  - Pueden darse varios caminos o **variantes**
  - Se pueden **agrupar** variantes similares en un sólo caso de uso, con el fin práctico de lograr una explicación más **comprensible**
- Los casos de uso pueden ser **contenedores** de requisitos no funcionales
  - Requisitos de rendimiento
  - Requisitos de disponibilidad
  - Requisitos de exactitud.

Casos de Uso: Captura

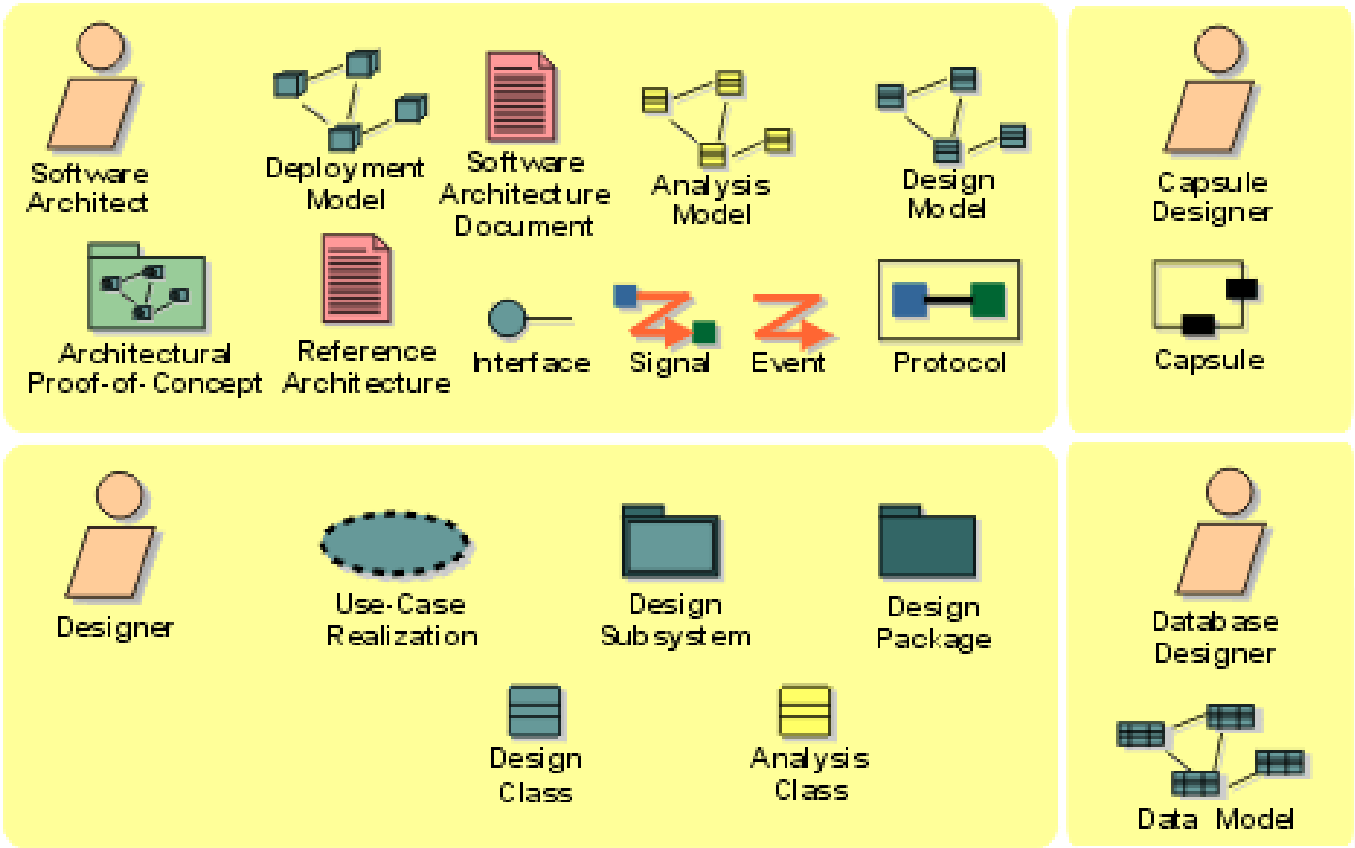
- Una **instancia** de un caso de uso es una **ejecución** específica del mismo
  - Es una **secuencia** de **acciones** que determinan un camino
  - Pueden darse varios caminos o **variantes**
  - Se pueden **agrupar** variantes similares en un sólo caso de uso, con el fin práctico de lograr una explicación más **comprensible**
- Los casos de uso pueden ser **contenedores** de requisitos no funcionales
  - Requisitos de rendimiento
  - Requisitos de disponibilidad
  - Requisitos de exactitud.
  - Requisitos de seguridad

Casos de Uso: Captura

Un ejemplo (parcial) de solicitud de cita médica



Análisis


Software Architect
Deployment Model
Software Architecture Document
Analysis Model
Design Model
Capsule Designer
Architectural Proof-of-Concept
Reference Architecture
Interface
Signal
Event
Protocol
Capsule
Designer
Use-Case Realization
Design Subsystem
Design Package
Database Designer
Design Class
Analysis Class
Data Model

Creación

- El modelo de análisis crece incrementalmente según se añaden casos de uso:

Creación

- El modelo de análisis crece incrementalmente según se añaden casos de uso:
  - Cada **iteración** trata y refleja en el análisis **un conjunto de casos de uso**

Creación

- El modelo de análisis crece incrementalmente según se añaden casos de uso:
  - Cada **iteración** trata y refleja en el análisis **un conjunto de casos de uso**
  - Un sistema es una estructura de ***clases de análisis*** y relaciones de **especialización**, **inclusión** o **extensión**

Creación

- El modelo de análisis crece incrementalmente según se añaden casos de uso:
  - Cada **iteración** trata y refleja en el análisis **un conjunto de casos de uso**
  - Un sistema es una estructura de ***clases de análisis*** y relaciones de **especialización**, **inclusión** o **extensión**
  - También se describen las **colaboraciones**, como realizaciones del caso de uso (clases **colaboradoras**, **estados** y **asociaciones**)

Creación

- El procedimiento práctico de trabajo es el siguiente:

Creación

- El procedimiento práctico de trabajo es el siguiente:
  - Identificar y **describir** los casos de uso de una iteración

Creación

- El procedimiento práctico de trabajo es el siguiente:
  - Identificar y **describir** los casos de uso de una iteración
  - Proponer **clasificadores** y **asociaciones**

Creación

- El procedimiento práctico de trabajo es el siguiente:
  - Identificar y **describir** los casos de uso de una iteración
  - Proponer **clasificadores** y **asociaciones**
  - Abordar coordinadamente **todos** los casos de uso de la iteración

Creación

- El procedimiento práctico de trabajo es el siguiente:
  - Identificar y **describir** los casos de uso de una iteración
  - Proponer **clasificadores** y **asociaciones**
  - Abordar coordinadamente **todos** los casos de uso de la iteración
- Cada clasificador desempeña uno o varios roles en la realización del caso de uso:

Creación

- El procedimiento práctico de trabajo es el siguiente:
  - Identificar y **describir** los casos de uso de una iteración
  - Proponer **clasificadores** y **asociaciones**
  - Abordar coordinadamente **todos** los casos de uso de la iteración
- Cada clasificador desempeña uno o varios roles en la realización del caso de uso:
  - Un rol especifica **responsabilidades**, **atributos** y **operaciones**.

Creación

- El procedimiento práctico de trabajo es el siguiente:
  - Identificar y **describir** los casos de uso de una iteración
  - Proponer **clasificadores** y **asociaciones**
  - Abordar coordinadamente **todos** los casos de uso de la iteración
- Cada clasificador desempeña uno o varios roles en la realización del caso de uso:
  - Un rol especifica **responsabilidades**, **atributos** y **operaciones**.
  - Un rol da **significado** a los *links* con otros clasificadores.

Creación

- El procedimiento práctico de trabajo es el siguiente:
  - Identificar y **describir** los casos de uso de una iteración
  - Proponer **clasificadores** y **asociaciones**
  - Abordar coordinadamente **todos** los casos de uso de la iteración
- Cada clasificador desempeña uno o varios roles en la realización del caso de uso:
  - Un rol especifica **responsabilidades**, **atributos** y **operaciones**.
  - Un rol da **significado** a los *links* con otros clasificadores.
  - Un rol es un ***gen*** de un clasificador en un estado *embrionario*

Creación

- Un rol, en UML, **conceptualmente** es un clasificador:

Creación

- Un rol, en UML, **conceptualmente** es un clasificador:
  - Puede entenderse un rol de una clase como una **vista** de la clase

Creación

- Un rol, en UML, **conceptualmente** es un clasificador:
  - Puede entenderse un rol de una clase como una **vista** de la clase
  - Una vista incluye sólo las **propiedades** significativas y necesarias para **ejercer** el rol

Creación

- Un rol, en UML, **conceptualmente** es un clasificador:
  - Puede entenderse un rol de una clase como una **vista** de la clase
  - Una vista incluye sólo las **propiedades** significativas y necesarias para **ejercer** el rol
  - Esta **encapsulación** de propiedades se establece para ejercer el rol en la **realización** del caso de uso en concreto

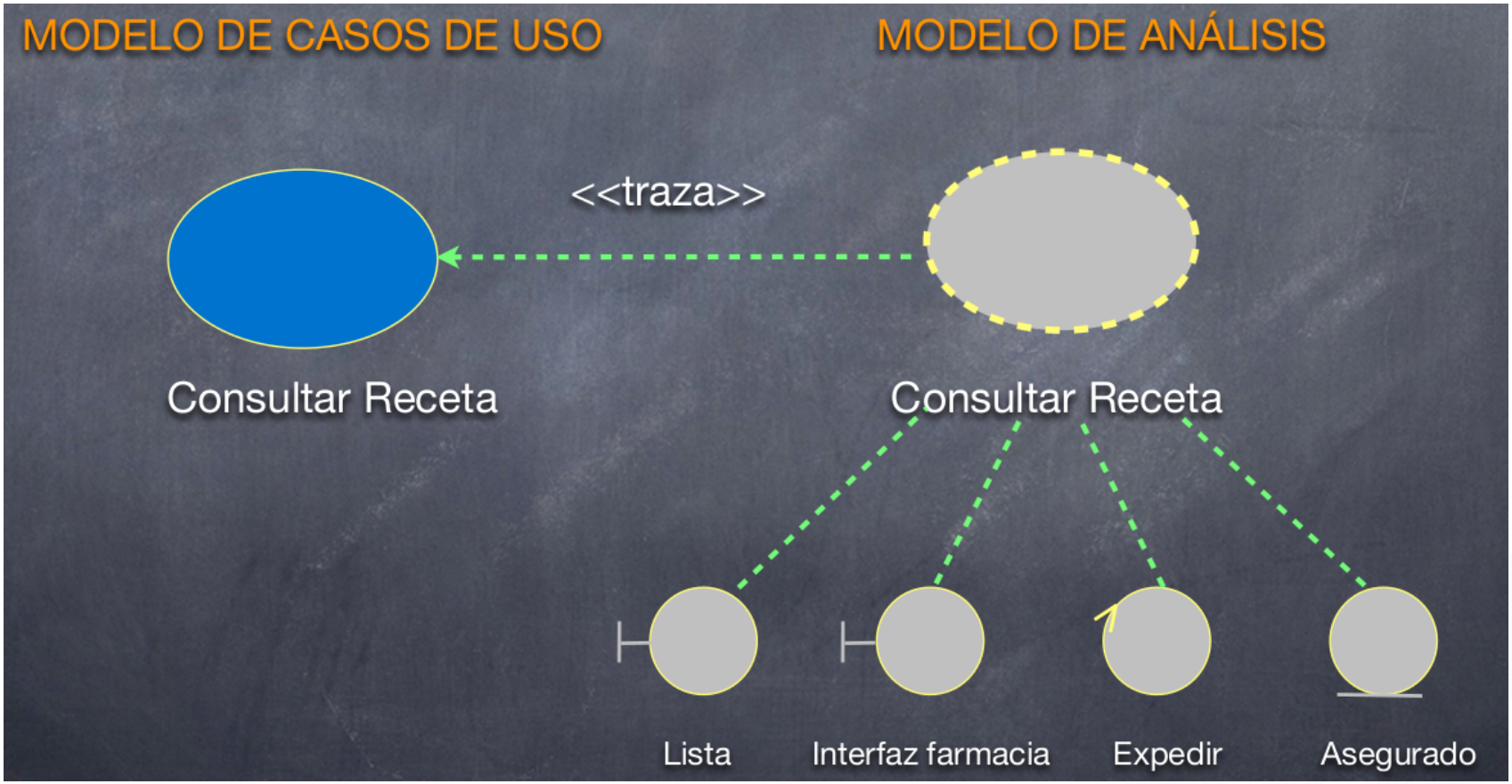
Creación

- Un rol, en UML, **conceptualmente** es un clasificador:
  - Puede entenderse un rol de una clase como una **vista** de la clase
  - Una vista incluye sólo las **propiedades** significativas y necesarias para **ejercer** el rol
  - Esta **encapsulación** de propiedades se establece para ejercer el rol en la **realización** del caso de uso en concreto
- Un rol, también puede entenderse como el resultado de la aplicación de un **filtro** a un clasificador:

Creación

- Un rol, en UML, **conceptualmente** es un clasificador:
  - Puede entenderse un rol de una clase como una **vista** de la clase
  - Una vista incluye sólo las **propiedades** significativas y necesarias para **ejercer** el rol
  - Esta **encapsulación** de propiedades se establece para ejercer el rol en la **realización** del caso de uso en concreto
- Un rol, también puede entenderse como el resultado de la aplicación de un **filtro** a un clasificador:
  - Los filtros excluyen los roles de la clase que **no** comparten **responsabilidades**

Creación


MODELO DE CASOS DE USO
MODELO DE ANÁLISIS
<<traza>>
Consultar Receta
Consultar Receta
Lista
Interfaz farmacia
Expedir
Asegurado

Creación

- Normalmente se utilizan 3 estereotipos de UML

Creación

- Normalmente se utilizan 3 estereotipos de UML
  - **Boundary**: normalmente representa un punto de interacción con el usuario

Creación

- Normalmente se utilizan 3 estereotipos de UML
  - **Boundary**: normalmente representa un punto de interacción con el usuario

  - **Control**: para aquellos puntos donde se implementa la lógica de negocio entre el interfaz y la persistencia

Creación

- Normalmente se utilizan 3 estereotipos de UML
  - **Boundary**: normalmente representa un punto de interacción con el usuario

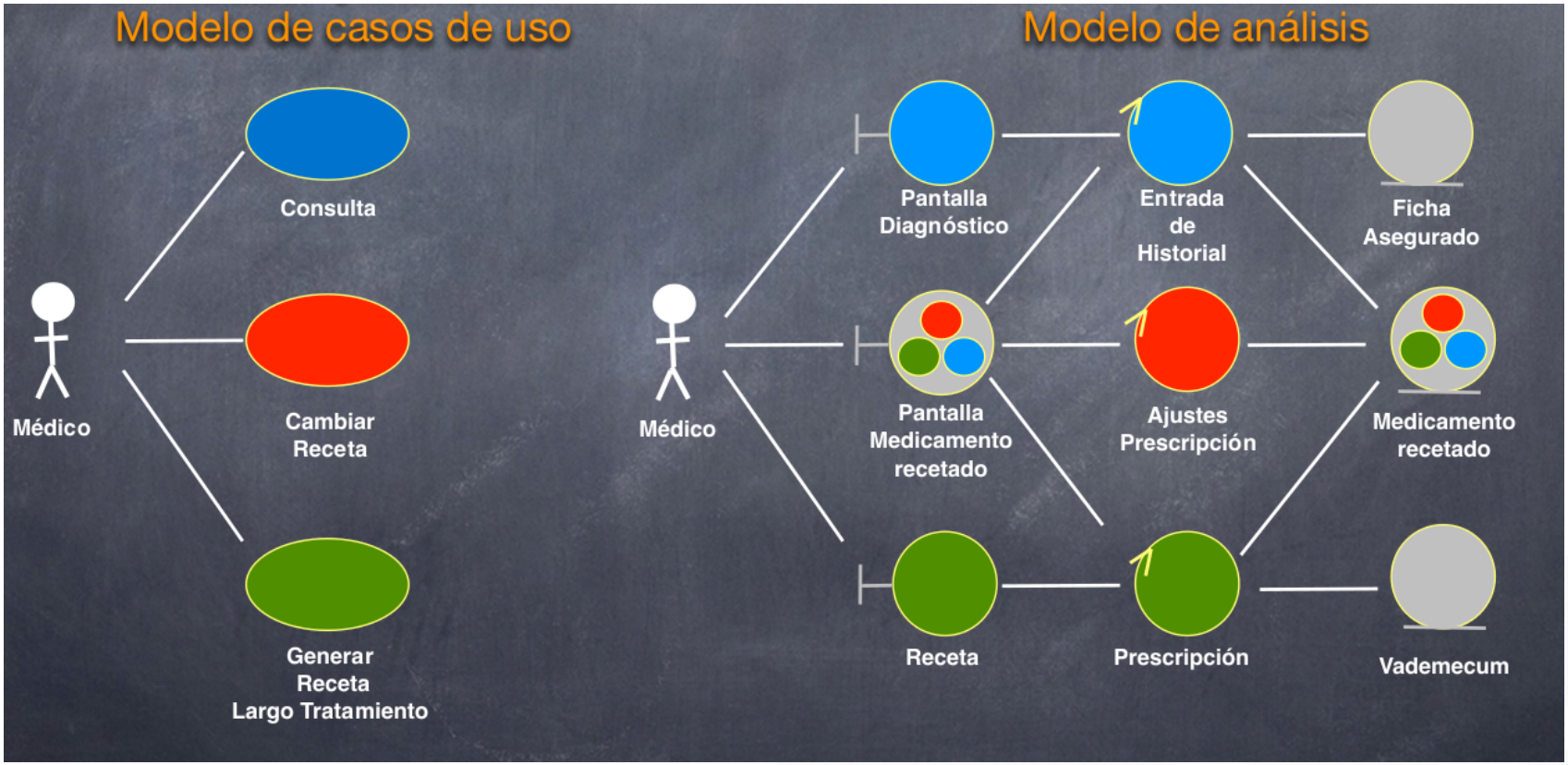
  - **Control**: para aquellos puntos donde se implementa la lógica de negocio entre el interfaz y la persistencia

  - **Entity**: que representa aspectos persistentes de la aplicación

Creación: Ejemplo



Creación: Ejemplo

- La realización de los tres casos de uso (Consulta, Cambiar posología receta, y generar receta largo tratamiento), implica a:

Creación: Ejemplo

- La realización de los tres casos de uso (Consulta, Cambiar posología receta, y generar receta largo tratamiento), implica a:
  - La clase interfaz *Pantalla medicamento Recetado*.

Creación: Ejemplo

- La realización de los tres casos de uso (Consulta, Cambiar posología receta, y generar receta largo tratamiento), implica a:
  - La clase interfaz *Pantalla medicamento Recetado*.
  - La clase entidad *Ficha de Asegurado*

Creación: Ejemplo

- La realización de los tres casos de uso (Consulta, Cambiar posología receta, y generar receta largo tratamiento), implica a:
  - La clase interfaz *Pantalla medicamento Recetado*.
  - La clase entidad *Ficha de Asegurado*
- El Objeto *Ajustes Prescripción*:

Creación: Ejemplo

- La realización de los tres casos de uso (Consulta, Cambiar posología receta, y generar receta largo tratamiento), implica a:
  - La clase interfaz *Pantalla medicamento Recetado*.
  - La clase entidad *Ficha de Asegurado*
- El Objeto *Ajustes Prescripción*:
  - Pide a los dos objetos instancia *Medicamento Recetado*, relacionados con el asegurado, que actualicen su estado en la lista de medicamentos prescritos

Creación: Ejemplo

- La realización de los tres casos de uso (Consulta, Cambiar posología receta, y generar receta largo tratamiento), implica a:
  - La clase interfaz *Pantalla medicamento Recetado*.
  - La clase entidad *Ficha de Asegurado*
- El Objeto *Ajustes Prescripción*:
  - Pide a los dos objetos instancia *Medicamento Recetado*, relacionados con el asegurado, que actualicen su estado en la lista de medicamentos prescritos
- El Objeto *Entrada historial*:

Creación: Ejemplo

- La realización de los tres casos de uso (Consulta, Cambiar posología receta, y generar receta largo tratamiento), implica a:
  - La clase interfaz *Pantalla medicamento Recetado*.
  - La clase entidad *Ficha de Asegurado*
- El Objeto *Ajustes Prescripción*:
  - Pide a los dos objetos instancia *Medicamento Recetado*, relacionados con el asegurado, que actualicen su estado en la lista de medicamentos prescritos
- El Objeto *Entrada historial*:
  - Refleja las variaciones de diagnóstico mostradas en el objeto interfaz *pantalla diagnostico* y actualiza las clases entidad *ficha asegurado* y *medicamento recetado*

Creación: Ejemplo

- La realización de los tres casos de uso (Consulta, Cambiar posología receta, y generar receta largo tratamiento), implica a:
  - La clase interfaz *Pantalla medicamento Recetado*.
  - La clase entidad *Ficha de Asegurado*
- El Objeto *Ajustes Prescripción*:
  - Pide a los dos objetos instancia *Medicamento Recetado*, relacionados con el asegurado, que actualicen su estado en la lista de medicamentos prescritos
- El Objeto *Entrada historial*:
  - Refleja las variaciones de diagnóstico mostradas en el objeto interfaz *pantalla diagnostico* y actualiza las clases entidad *ficha asegurado* y *medicamento recetado*
- El Objeto *Prescripción*:

Creación: Ejemplo

- La realización de los tres casos de uso (Consulta, Cambiar posología receta, y generar receta largo tratamiento), implica a:
  - La clase interfaz *Pantalla medicamento Recetado*.
  - La clase entidad *Ficha de Asegurado*
- El Objeto *Ajustes Prescripción*:
  - Pide a los dos objetos instancia *Medicamento Recetado*, relacionados con el asegurado, que actualicen su estado en la lista de medicamentos prescritos
- El Objeto *Entrada historial*:
  - Refleja las variaciones de diagnóstico mostradas en el objeto interfaz *pantalla diagnostico* y actualiza las clases entidad *ficha asegurado* y *medicamento recetado*
- El Objeto *Prescripción*:
  - Muestra la receta en la clase interfaz *receta* y accede a las clases entidad *Vademecum* (en lectura) y *Medicamento recetado* (en creación)

Creación: Ejemplo

- Obtenida una **estructura** estable del sistema para la iteración en curso

Creación: Ejemplo

- Obtenida una **estructura** estable del sistema para la iteración en curso
  - Con las **responsabilidades** de los clasificadores participantes identificadas

Creación: Ejemplo

- Obtenida una **estructura** estable del sistema para la iteración en curso
  - Con las **responsabilidades** de los clasificadores participantes identificadas
  - Queda establecida la **relación** entre entre estos clasificadores

Creación: Ejemplo

- Obtenida una **estructura** estable del sistema para la iteración en curso
  - Con las **responsabilidades** de los clasificadores participantes identificadas
  - Queda establecida la **relación** entre entre estos clasificadores
  - Describiendo como se **instancia** una realización del caso de uso

Creación: Ejemplo

- Obtenida una **estructura** estable del sistema para la iteración en curso
  - Con las **responsabilidades** de los clasificadores participantes identificadas
  - Queda establecida la **relación** entre entre estos clasificadores
  - Describiendo como se **instancia** una realización del caso de uso
- El paso siguiente consiste en averiguar y explicitar en detalle la interacción

Creación: Ejemplo

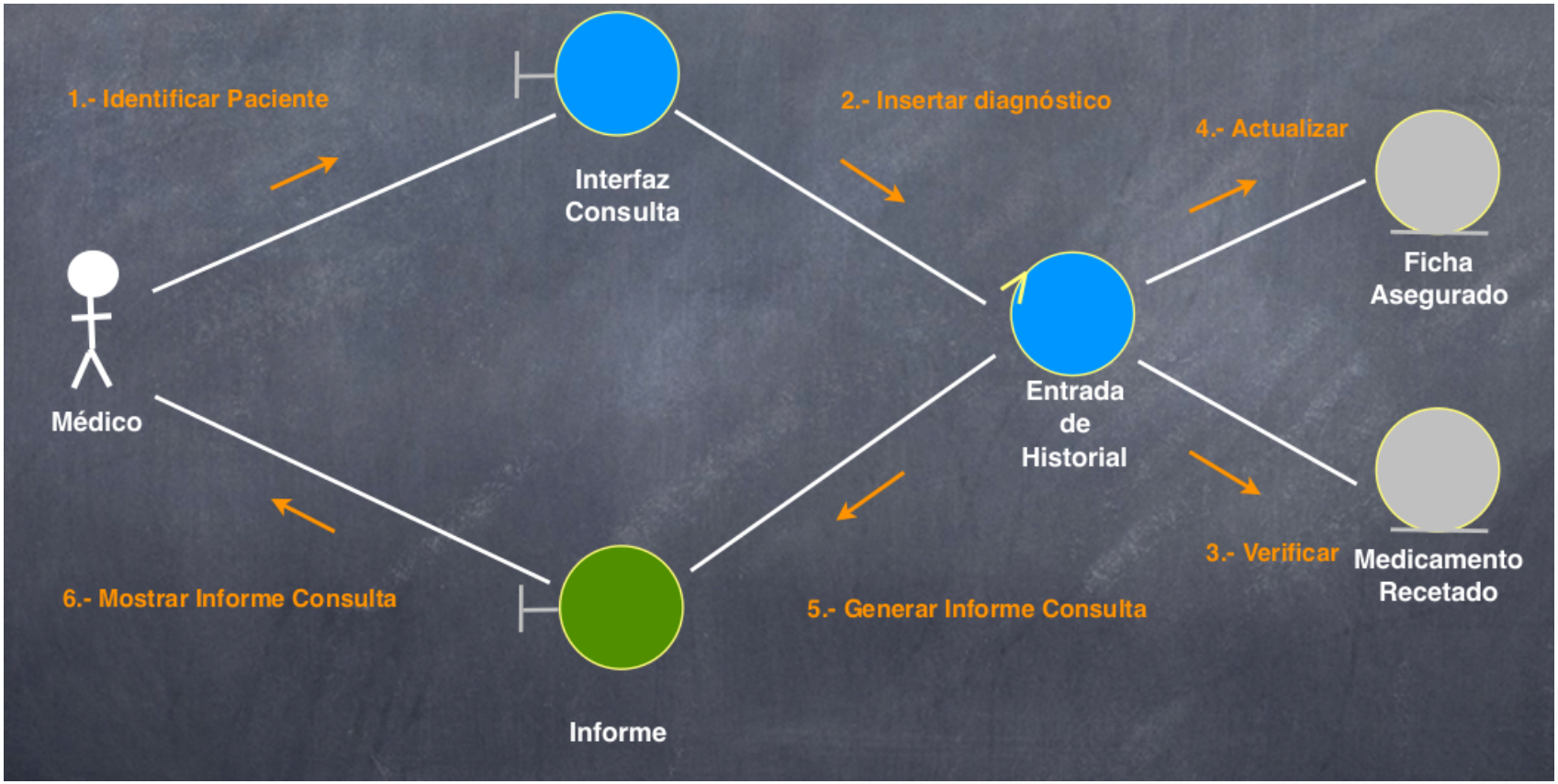
- Obtenida una **estructura** estable del sistema para la iteración en curso
  - Con las **responsabilidades** de los clasificadores participantes identificadas
  - Queda establecida la **relación** entre entre estos clasificadores
  - Describiendo como se **instancia** una realización del caso de uso
- El paso siguiente consiste en averiguar y explicitar en detalle la interacción
  - Se elaboran los **diagramas de colaboración** que son una vista de la estructura de clases, orientadas a la comunicación

Creación: Ejemplo

- Obtenida una **estructura** estable del sistema para la iteración en curso
  - Con las **responsabilidades** de los clasificadores participantes identificadas
  - Queda establecida la **relación** entre entre estos clasificadores
  - Describiendo como se **instancia** una realización del caso de uso
- El paso siguiente consiste en averiguar y explicitar en detalle la interacción
  - Se elaboran los **diagramas de colaboración** que son una vista de la estructura de clases, orientadas a la comunicación
  - Se elaboran, en algunos casos, los **diagramas de secuencia** que son una traza de eventos y mensajes entre las lineas de vida de los clasificadores que colaboran

Creación: Ejemplo

Una diagrama de colaboración **realiza** un caso de uso



Creación: Ejemplo

- El médico atiende al paciente, identificándolo en su clase Interfaz *Consulta*

Creación: Ejemplo

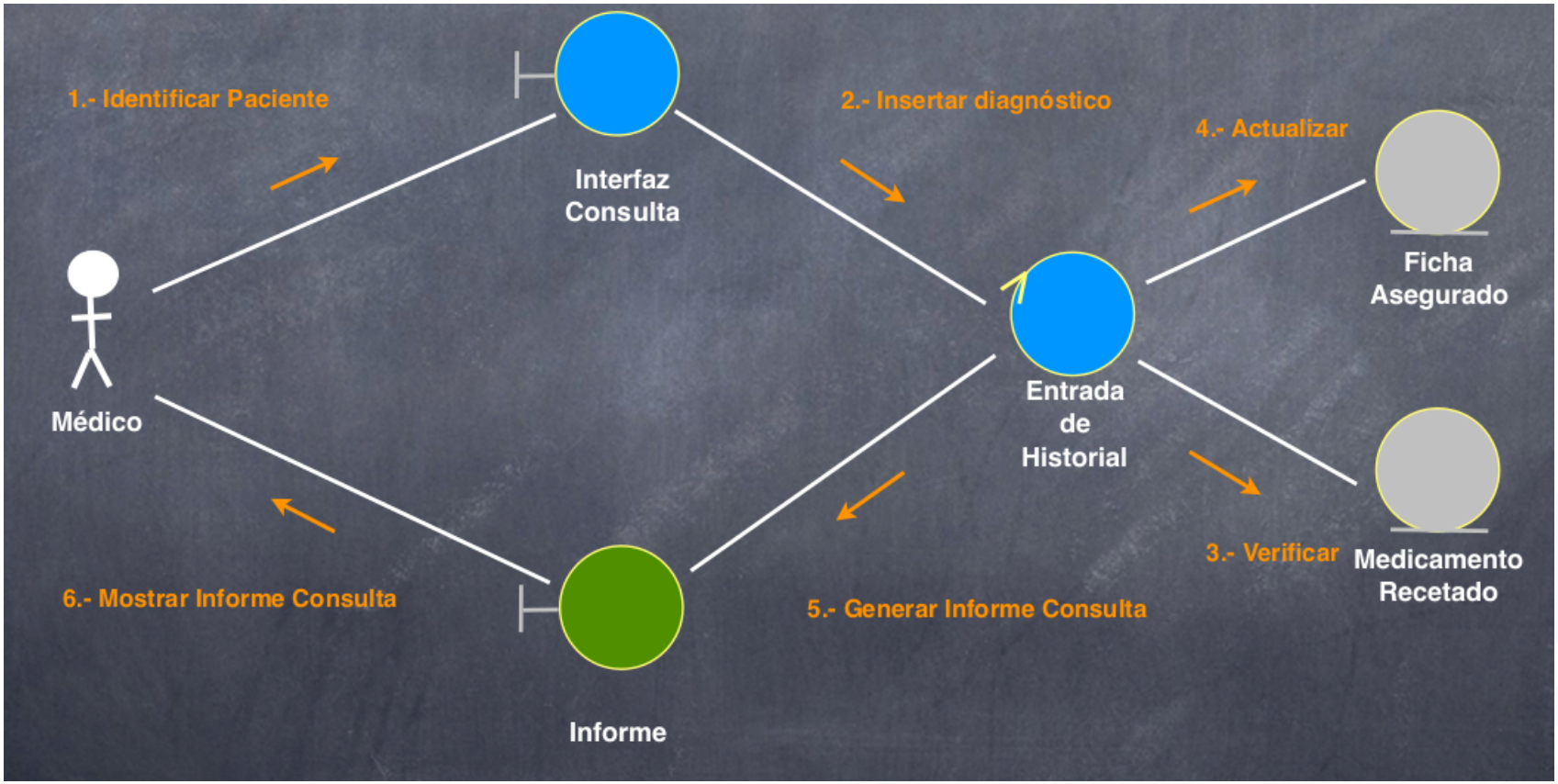
- El médico atiende al paciente, identificándolo en su clase Interfaz *Consulta*
- El médico diagnostica y verifica la medicación en la clase entidad *Medicamento Recetado* y actualiza la clase entidad *Ficha de Asegurado*, todo ello mediante la clase de control *Entrada de Historial*, que incrementa este

Creación: Ejemplo

- El médico atiende al paciente, identificándolo en su clase Interfaz *Consulta*
- El médico diagnostica y verifica la medicación en la clase entidad *Medicamento Recetado* y actualiza la clase entidad *Ficha de Asegurado*, todo ello mediante la clase de control *Entrada de Historial*, que incrementa este
- La clase *Entrada de Historial* genera un informe de la consulta, que se muestra en la clase interfaz *Informe*, y que el médico leerá e imprimirá para su firma y entrega al paciente

Creación: Ejemplo

Una diagrama de colaboración **realiza** un caso de uso



Creación: Ejemplo

- El diagrama de colaboración refleja

Creación: Ejemplo

- El diagrama de colaboración refleja
  - Como el **control** se **transfiere** de un objeto a otro a medida que se ejecuta el caso de uso

Creación: Ejemplo

- El diagrama de colaboración refleja
  - Como el **control** se **transfiere** de un objeto a otro a medida que se ejecuta el caso de uso
  - Que **mensajes** (y atributos) se **intercambian** entre los objetos

Creación: Ejemplo

- El diagrama de colaboración refleja
  - Como el **control** se **transfiere** de un objeto a otro a medida que se ejecuta el caso de uso
  - Que **mensajes** (y atributos) se **intercambian** entre los objetos
- Un **mensaje**

Creación: Ejemplo

- El diagrama de colaboración refleja
  - Como el **control** se **transfiere** de un objeto a otro a medida que se ejecuta el caso de uso
  - Que **mensajes** (y atributos) se **intercambian** entre los objetos
- Un **mensaje**
  - **Dispara al objeto receptor** para que tome el control y lleve a cabo una de las responsabilidades de su clase (ejecutando operaciones/métodos)

Creación: Ejemplo

- El diagrama de colaboración refleja
  - Como el **control** se **transfiere** de un objeto a otro a medida que se ejecuta el caso de uso
  - Que **mensajes** (y atributos) se **intercambian** entre los objetos
- Un **mensaje**
  - **Dispara al objeto receptor** para que tome el control y lleve a cabo una de las responsabilidades de su clase (ejecutando operaciones/métodos)
  - Su **nombre** indica el **motivo** del objeto emisor que realiza la llamada o emite la señal, en su interacción con el objeto invocado

Creación: Ejemplo

- El diagrama de colaboración refleja
  - Como el **control** se **transfiere** de un objeto a otro a medida que se ejecuta el caso de uso
  - Que **mensajes** (y atributos) se **intercambian** entre los objetos
- Un **mensaje**
  - **Dispara al objeto receptor** para que tome el control y lleve a cabo una de las responsabilidades de su clase (ejecutando operaciones/métodos)
  - Su **nombre** indica el **motivo** del objeto emisor que realiza la llamada o emite la señal, en su interacción con el objeto invocado
- Estos mensajes del modelo de análisis tendrán que reflejarse necesariamente en el modelo de diseño como operaciones sobre atributos en clases.

Creación: Ejemplo

Una vez analizados los casos de uso, e identificados todas las clases rol que colaboran en su realización, se continúa con el procedimiento de trabajo, **analizando cada clase**

- Las **responsabilidades** de una clase son la recopilación de sus roles

Creación: Ejemplo

Una vez analizados los casos de uso, e identificados todas las clases rol que colaboran en su realización, se continúa con el procedimiento de trabajo, **analizando cada clase**

- Las **responsabilidades** de una clase son la recopilación de sus roles
  - Se depuran repeticiones y roles espúreos

Creación: Ejemplo

Una vez analizados los casos de uso, e identificados todas las clases rol que colaboran en su realización, se continúa con el procedimiento de trabajo, **analizando cada clase**

- Las **responsabilidades** de una clase son la recopilación de sus roles
  - Se depuran repeticiones y roles espúreos
  - Se orientan las responsabilidades/operaciones finales

Creación: Ejemplo

Una vez analizados los casos de uso, e identificados todas las clases rol que colaboran en su realización, se continúa con el procedimiento de trabajo, **analizando cada clase**

- Las **responsabilidades** de una clase son la recopilación de sus roles
  - Se depuran repeticiones y roles espúreos
  - Se orientan las responsabilidades/operaciones finales
  - Se esbozan los atributos de la clase

Creación: Ejemplo

Una vez analizados los casos de uso, e identificados todas las clases rol que colaboran en su realización, se continúa con el procedimiento de trabajo, **analizando cada clase**

- Las **responsabilidades** de una clase son la recopilación de sus roles
  - Se depuran repeticiones y roles espúreos
  - Se orientan las responsabilidades/operaciones finales
  - Se esbozan los atributos de la clase
- Si se **cambia** una **clase**

Creación: Ejemplo

Una vez analizados los casos de uso, e identificados todas las clases rol que colaboran en su realización, se continúa con el procedimiento de trabajo, **analizando cada clase**

- Las **responsabilidades** de una clase son la recopilación de sus roles
  - Se depuran repeticiones y roles espúreos
  - Se orientan las responsabilidades/operaciones finales
  - Se esbozan los atributos de la clase
- Si se **cambia** una **clase**
  - Se debe de garantizar que sigue cumpliendo sus roles en la realización de los casos de uso en la que está implicada

Creación: Ejemplo

Una vez analizados los casos de uso, e identificados todas las clases rol que colaboran en su realización, se continúa con el procedimiento de trabajo, **analizando cada clase**

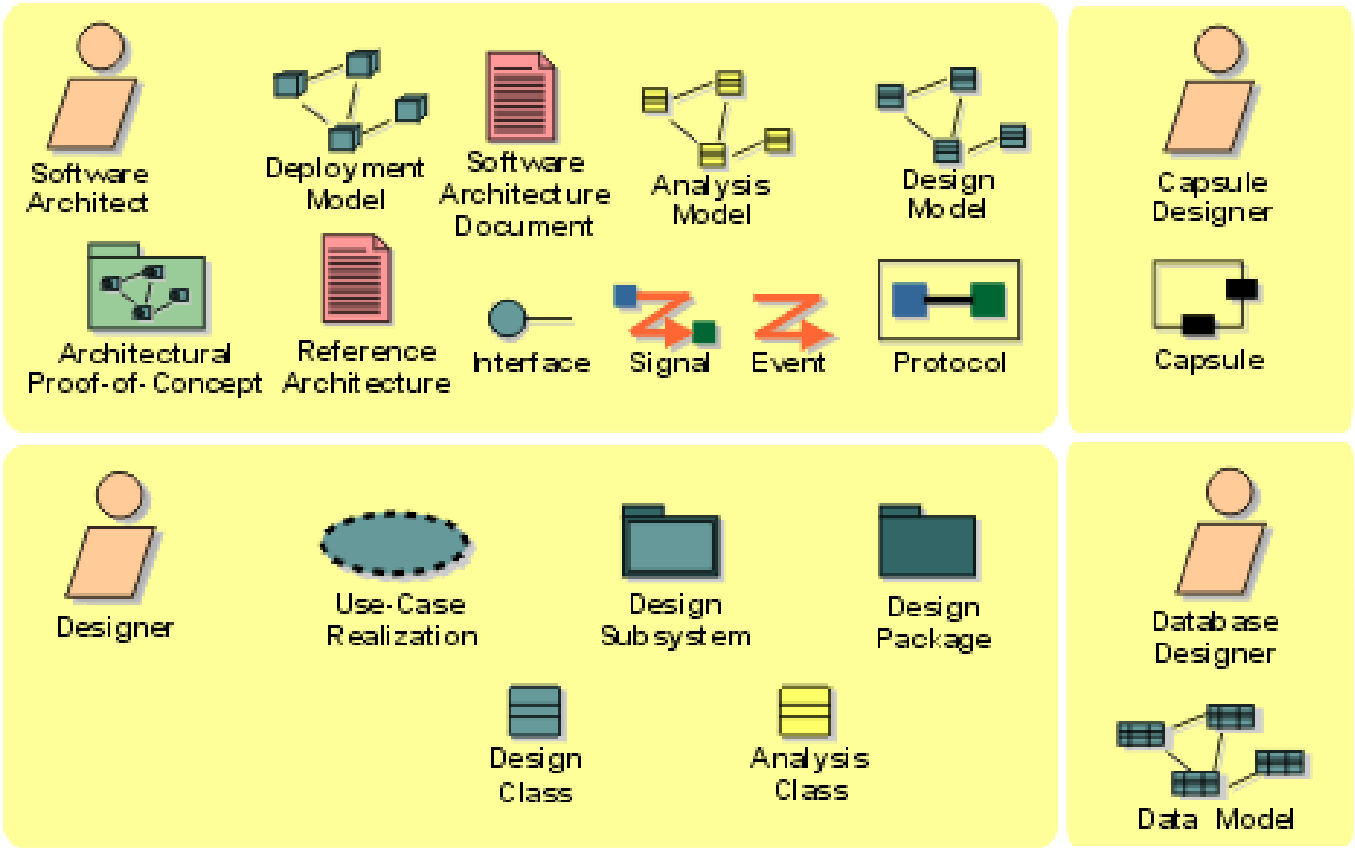
- Las **responsabilidades** de una clase son la recopilación de sus roles
  - Se depuran repeticiones y roles espúreos
  - Se orientan las responsabilidades/operaciones finales
  - Se esbozan los atributos de la clase
- Si se **cambia** una **clase**
  - Se debe de garantizar que sigue cumpliendo sus roles en la realización de los casos de uso en la que está implicada
- Si se cambia un **rol**

Creación: Ejemplo

Una vez analizados los casos de uso, e identificados todas las clases rol que colaboran en su realización, se continúa con el procedimiento de trabajo, **analizando cada clase**

- Las **responsabilidades** de una clase son la recopilación de sus roles
  - Se depuran repeticiones y roles espúreos
  - Se orientan las responsabilidades/operaciones finales
  - Se esbozan los atributos de la clase
- Si se **cambia** una **clase**
  - Se debe de garantizar que sigue cumpliendo sus roles en la realización de los casos de uso en la que está implicada
- Si se cambia un **rol**
  - El desarrollador del caso de uso debe comunicar el cambio al desarrollador/administrador de la clase

Diseño


Software Architect
Deployment Model
Software Architecture Document
Analysis Model
Design Model
Capsule Designer
Architectural Proof-of-Concept
Reference Architecture
Interface
Signal
Event
Protocol
Capsule
Designer
Use-Case Realization
Design Subsystem
Design Package
Database Designer
Design Class
Analysis Class
Data Model

Creación

- El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal:

Creación

- El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal:
  - Se adapta al entorno de **implementación** elegido

Creación

- El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal:
  - Se adapta al entorno de **implementación** elegido
  - Se adapta al ***legacy***

Creación

- El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal:
  - Se adapta al entorno de **implementación** elegido
  - Se adapta al ***legacy***
    - Sistemas y aplicaciones heredadas

Creación

- El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal:
  - Se adapta al entorno de **implementación** elegido
  - Se adapta al ***legacy***
    - Sistemas y aplicaciones heredadas
    - Marcos de trabajo de desarrollo específico para el proyecto

Creación

- El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal:
  - Se adapta al entorno de **implementación** elegido
  - Se adapta al ***legacy***
    - Sistemas y aplicaciones heredadas
    - Marcos de trabajo de desarrollo específico para el proyecto
    - Integradores.

Creación

- El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal:
  - Se adapta al entorno de **implementación** elegido
  - Se adapta al ***legacy***
    - Sistemas y aplicaciones heredadas
    - Marcos de trabajo de desarrollo específico para el proyecto
    - Integradores.
- Funciona como un **esquema** para la implementación

Creación

El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal

- Se continúa la **definición**, como *contrapartidas de diseño*, para la implementación de los **elementos del análisis**

Creación

El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal

- Se continúa la **definición**, como *contrapartidas de diseño*, para la implementación de los **elementos del análisis**
  - Clasificadores (clases, subsistemas e interfaces)

Creación

El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal

- Se continúa la **definición**, como *contrapartidas de diseño*, para la implementación de los **elementos del análisis**
  - Clasificadores (clases, subsistemas e interfaces)
  - Relaciones

Creación

El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal

- Se continúa la **definición**, como *contrapartidas de diseño*, para la implementación de los **elementos del análisis**
  - Clasificadores (clases, subsistemas e interfaces)
  - Relaciones
  - Colaboraciones de los casos de uso

Creación

El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal

- Se continúa la **definición**, como *contrapartidas de diseño*, para la implementación de los **elementos del análisis**
  - Clasificadores (clases, subsistemas e interfaces)
  - Relaciones
  - Colaboraciones de los casos de uso
- La definición de los elementos del sistema es de diferente naturaleza

Creación

El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal

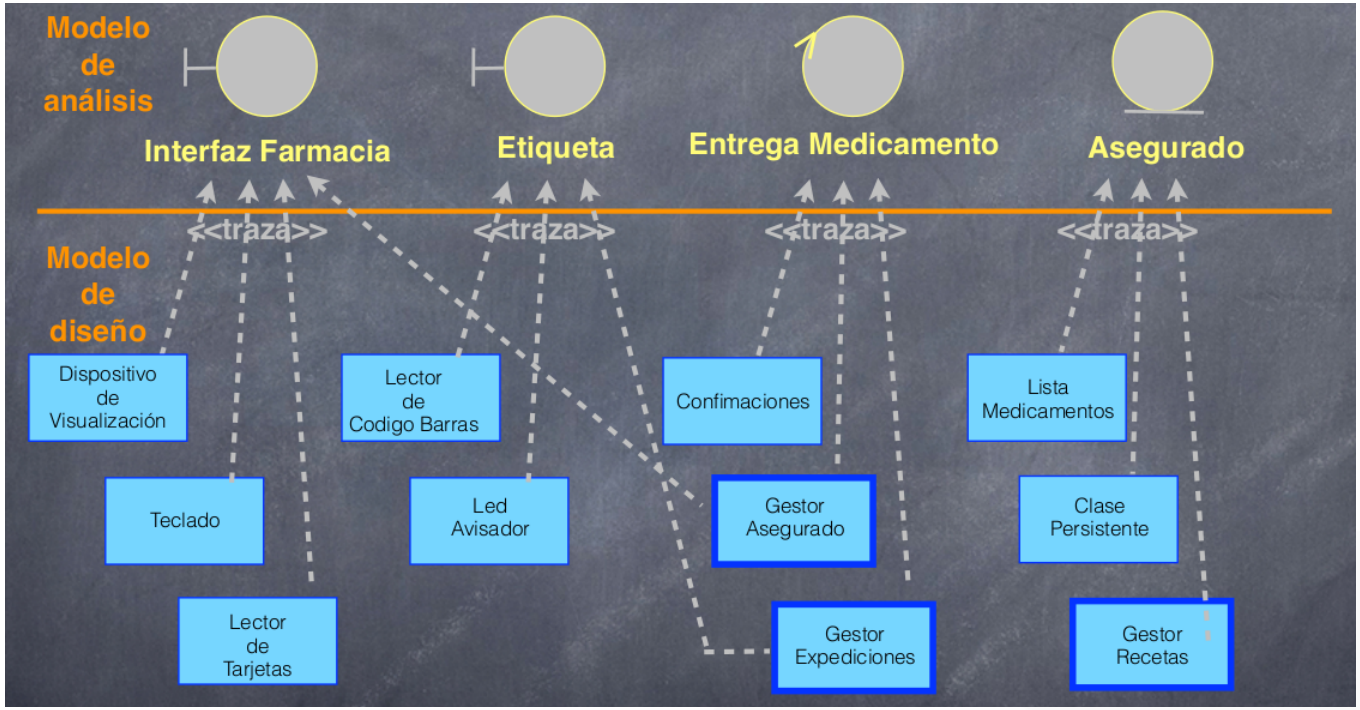
- Se continúa la **definición**, como *contrapartidas de diseño*, para la implementación de los **elementos del análisis**
  - Clasificadores (clases, subsistemas e interfaces)
  - Relaciones
  - Colaboraciones de los casos de uso
- La definición de los elementos del sistema es de diferente naturaleza
  - En el Modelo de Diseño es más fisica o tecnológica

Creación

El Modelo de Diseño se crea tomando el Modelo de Análisis como entrada principal

- Se continúa la **definición**, como *contrapartidas de diseño*, para la implementación de los **elementos del análisis**
  - Clasificadores (clases, subsistemas e interfaces)
  - Relaciones
  - Colaboraciones de los casos de uso
- La definición de los elementos del sistema es de diferente naturaleza
  - En el Modelo de Diseño es más fisica o tecnológica
  - En el Modelo de Análisis es conceptual

Creación: Ejemplo



Creación: Ejemplo

La clase de análisis *Interfaz Farmacia* se realiza con las clases de diseño

- **No Activas** (además en este caso, tendrán implicaciones en el modelo de despliegue)

Creación: Ejemplo

La clase de análisis *Interfaz Farmacia* se realiza con las clases de diseño

- **No Activas** (además en este caso, tendrán implicaciones en el modelo de despliegue)
  - Dispositivo de Visualización; Teclado; Lector de Tarjetas

Creación: Ejemplo

La clase de análisis *Interfaz Farmacia* se realiza con las clases de diseño

- **No Activas** (además en este caso, tendrán implicaciones en el modelo de despliegue)
  - Dispositivo de Visualización; Teclado; Lector de Tarjetas
- **Activas** (representan propiedades que organizan el trabajo de otras clases)

Creación: Ejemplo

La clase de análisis *Interfaz Farmacia* se realiza con las clases de diseño

- **No Activas** (además en este caso, tendrán implicaciones en el modelo de despliegue)
  - Dispositivo de Visualización; Teclado; Lector de Tarjetas
- **Activas** (representan propiedades que organizan el trabajo de otras clases)
  - Gestor de asegurados

Creación: Ejemplo

La clase de análisis *Interfaz Farmacia* se realiza con las clases de diseño

- **No Activas** (además en este caso, tendrán implicaciones en el modelo de despliegue)
  - Dispositivo de Visualización; Teclado; Lector de Tarjetas
- **Activas** (representan propiedades que organizan el trabajo de otras clases)
  - Gestor de asegurados
- Muchas clases de diseño muestran una **única traza** a las correspondientes de análisis

Creación: Ejemplo

La clase de análisis *Interfaz Farmacia* se realiza con las clases de diseño

- **No Activas** (además en este caso, tendrán implicaciones en el modelo de despliegue)
  - Dispositivo de Visualización; Teclado; Lector de Tarjetas
- **Activas** (representan propiedades que organizan el trabajo de otras clases)
  - Gestor de asegurados
- Muchas clases de diseño muestran una **única traza** a las correspondientes de análisis
  - Habitual en las clases **específicas** del dominio y la aplicación

Creación: Ejemplo

La clase de análisis *Interfaz Farmacia* se realiza con las clases de diseño

- **No Activas** (además en este caso, tendrán implicaciones en el modelo de despliegue)
  - Dispositivo de Visualización; Teclado; Lector de Tarjetas
- **Activas** (representan propiedades que organizan el trabajo de otras clases)
  - Gestor de asegurados
- Muchas clases de diseño muestran una **única traza** a las correspondientes de análisis
  - Habitual en las clases **específicas** del dominio y la aplicación
  - La estructura del sistema definida en el análisis se conserva de forma natural durante el diseño

Creación: Ejemplo

La clase de análisis *Interfaz Farmacia* se realiza con las clases de diseño

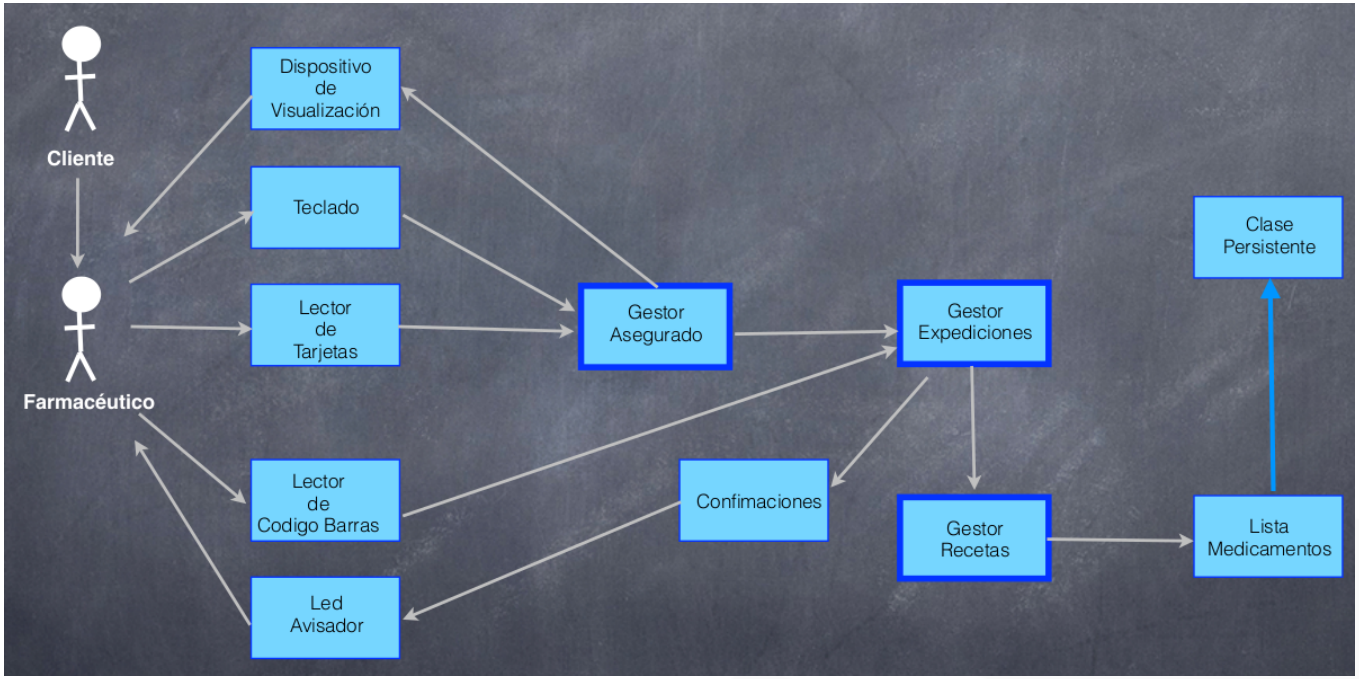
- **No Activas** (además en este caso, tendrán implicaciones en el modelo de despliegue)
  - Dispositivo de Visualización; Teclado; Lector de Tarjetas
- **Activas** (representan propiedades que organizan el trabajo de otras clases)
  - Gestor de asegurados
- Muchas clases de diseño muestran una **única traza** a las correspondientes de análisis
  - Habitual en las clases **específicas** del dominio y la aplicación
  - La estructura del sistema definida en el análisis se conserva de forma natural durante el diseño
- Pueden, no obstante, ser necesarios algunos **cambios** en la estructura que amplía la trazabilidad

Creación: Ejemplo

La clase de análisis *Interfaz Farmacia* se realiza con las clases de diseño

- **No Activas** (además en este caso, tendrán implicaciones en el modelo de despliegue)
  - Dispositivo de Visualización; Teclado; Lector de Tarjetas
- **Activas** (representan propiedades que organizan el trabajo de otras clases)
  - Gestor de asegurados
- Muchas clases de diseño muestran una **única traza** a las correspondientes de análisis
  - Habitual en las clases **específicas** del dominio y la aplicación
  - La estructura del sistema definida en el análisis se conserva de forma natural durante el diseño
- Pueden, no obstante, ser necesarios algunos **cambios** en la estructura que amplía la trazabilidad
  - Por ejemplo, *Gestor de Asegurados* participa en el diseño de las clases de análisis *Interfaz Farmacia* y *Entrega de Medicamentos*

Creación: Ejemplo



Creación: Ejemplo

- Se debe identificar la interacción **detallada** entre los objetos de diseño

Creación: Ejemplo

- Se debe identificar la interacción **detallada** entre los objetos de diseño
  - Se Utilizan los diagramas de **secuencia**

Creación: Ejemplo

- Se debe identificar la interacción **detallada** entre los objetos de diseño
  - Se Utilizan los diagramas de **secuencia**
    - Clasificadores

Creación: Ejemplo

- Se debe identificar la interacción **detallada** entre los objetos de diseño
  - Se Utilizan los diagramas de **secuencia**
    - Clasificadores
    - Líneas de vida

Creación: Ejemplo

- Se debe identificar la interacción **detallada** entre los objetos de diseño
  - Se Utilizan los diagramas de **secuencia**
    - Clasificadores
    - Líneas de vida
    - Mensajes

Creación: Ejemplo

- Se debe identificar la interacción **detallada** entre los objetos de diseño
  - Se Utilizan los diagramas de **secuencia**
    - Clasificadores
    - Líneas de vida
    - Mensajes
    - Activaciones

Creación: Ejemplo

- Se debe identificar la interacción **detallada** entre los objetos de diseño
  - Se Utilizan los diagramas de **secuencia**
    - Clasificadores
    - Líneas de vida
    - Mensajes
    - Activaciones
  - Un diagrama de secuencia muestra como el control pasa de un objeto a otro ante el **disparo** de **mensajes**

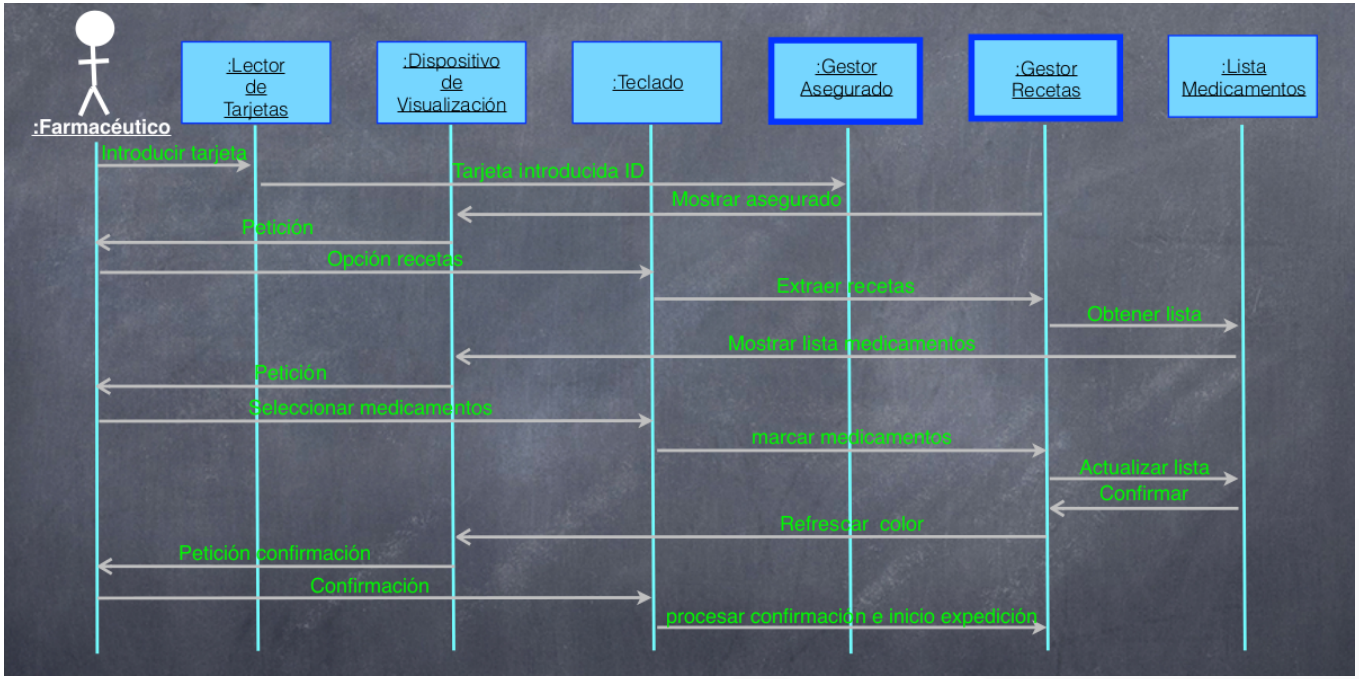
Creación: Ejemplo

- Se debe identificar la interacción **detallada** entre los objetos de diseño
  - Se Utilizan los diagramas de **secuencia**
    - Clasificadores
    - Líneas de vida
    - Mensajes
    - Activaciones
  - Un diagrama de secuencia muestra como el control pasa de un objeto a otro ante el **disparo** de **mensajes**
    - Síncronos (llamadas, estereotipo «call»)

Creación: Ejemplo

- Se debe identificar la interacción **detallada** entre los objetos de diseño
  - Se Utilizan los diagramas de **secuencia**
    - Clasificadores
    - Líneas de vida
    - Mensajes
    - Activaciones
  - Un diagrama de secuencia muestra como el control pasa de un objeto a otro ante el **disparo** de **mensajes**
    - Síncronos (llamadas, estereotipo «call»)
    - Asíncronos (señales, estereotipo «signal»)

Creación: Ejemplo



Subsistemas

- El modelo de diseño aumenta la complejidad y contiene muchas clases de muchos tipos

Subsistemas

- El modelo de diseño aumenta la complejidad y contiene muchas clases de muchos tipos
- Para organizarlas se agrupan o *empaquetan* en **subsistema**

Subsistemas

- El modelo de diseño aumenta la complejidad y contiene muchas clases de muchos tipos
- Para organizarlas se agrupan o *empaquetan* en **subsistema**
  - Un susbsistema es un **agrupamiento**, semánticamente útil, de clases o de otros subsistemas

Subsistemas

- El modelo de diseño aumenta la complejidad y contiene muchas clases de muchos tipos
- Para organizarlas se agrupan o *empaquetan* en **subsistema**
  - Un susbsistema es un **agrupamiento**, semánticamente útil, de clases o de otros subsistemas
    - Posee un conjunto de interfaces que ofrece al usuario

Subsistemas

- El modelo de diseño aumenta la complejidad y contiene muchas clases de muchos tipos
- Para organizarlas se agrupan o *empaquetan* en **subsistema**
  - Un susbsistema es un **agrupamiento**, semánticamente útil, de clases o de otros subsistemas
    - Posee un conjunto de interfaces que ofrece al usuario
    - Las interfaces definen el contexto del subsistema

Subsistemas

- El modelo de diseño aumenta la complejidad y contiene muchas clases de muchos tipos
- Para organizarlas se agrupan o *empaquetan* en **subsistema**
  - Un susbsistema es un **agrupamiento**, semánticamente útil, de clases o de otros subsistemas
    - Posee un conjunto de interfaces que ofrece al usuario
    - Las interfaces definen el contexto del subsistema
  - Los subsistemas de bajo nivel se denominan **Susbsistemas de Servicio**

Subsistemas

- El modelo de diseño aumenta la complejidad y contiene muchas clases de muchos tipos
- Para organizarlas se agrupan o *empaquetan* en **subsistema**
  - Un susbsistema es un **agrupamiento**, semánticamente útil, de clases o de otros subsistemas
    - Posee un conjunto de interfaces que ofrece al usuario
    - Las interfaces definen el contexto del subsistema
  - Los subsistemas de bajo nivel se denominan **Susbsistemas de Servicio**
    - Constituyen una unidad manejable de funcionalidad opcional

Subsistemas

- El modelo de diseño aumenta la complejidad y contiene muchas clases de muchos tipos
- Para organizarlas se agrupan o *empaquetan* en **subsistema**
  - Un susbsistema es un **agrupamiento**, semánticamente útil, de clases o de otros subsistemas
    - Posee un conjunto de interfaces que ofrece al usuario
    - Las interfaces definen el contexto del subsistema
  - Los subsistemas de bajo nivel se denominan **Susbsistemas de Servicio**
    - Constituyen una unidad manejable de funcionalidad opcional
    - Sólo es factible instalar un subsistema en un sistema cliente si se hace en su totalidad

Subsistemas

- El modelo de diseño aumenta la complejidad y contiene muchas clases de muchos tipos
- Para organizarlas se agrupan o *empaquetan* en **subsistema**
  - Un susbsistema es un **agrupamiento**, semánticamente útil, de clases o de otros subsistemas
    - Posee un conjunto de interfaces que ofrece al usuario
    - Las interfaces definen el contexto del subsistema
  - Los subsistemas de bajo nivel se denominan **Susbsistemas de Servicio**
    - Constituyen una unidad manejable de funcionalidad opcional
    - Sólo es factible instalar un subsistema en un sistema cliente si se hace en su totalidad
    - Sirven también para modelar grupos de clase que tienden a cambiar juntas.

Subsistemas

- Los subsistemas pueden diseñarse de modo **ascendente** o **descendente**

Subsistemas

- Los subsistemas pueden diseñarse de modo **ascendente** o **descendente**
  - Ascendente

Subsistemas

- Los subsistemas pueden diseñarse de modo **ascendente** o **descendente**
  - Ascendente
    - Los desarrolladores proponen subsistemas basados en clases ya identificadas

Subsistemas

- Los subsistemas pueden diseñarse de modo **ascendente** o **descendente**
  - Ascendente
    - Los desarrolladores proponen subsistemas basados en clases ya identificadas
    - Empaquetan estas clases en unidades con funciones bien definidas

Subsistemas

- Los subsistemas pueden diseñarse de modo **ascendente** o **descendente**
  - Ascendente
    - Los desarrolladores proponen subsistemas basados en clases ya identificadas
    - Empaquetan estas clases en unidades con funciones bien definidas
  - Descendentes

Subsistemas

- Los subsistemas pueden diseñarse de modo **ascendente** o **descendente**
  - Ascendente
    - Los desarrolladores proponen subsistemas basados en clases ya identificadas
    - Empaquetan estas clases en unidades con funciones bien definidas
  - Descendentes
    - El arquitecto identifica subsistemas de más alto nivel y sus interfaces antes de identificar las clases

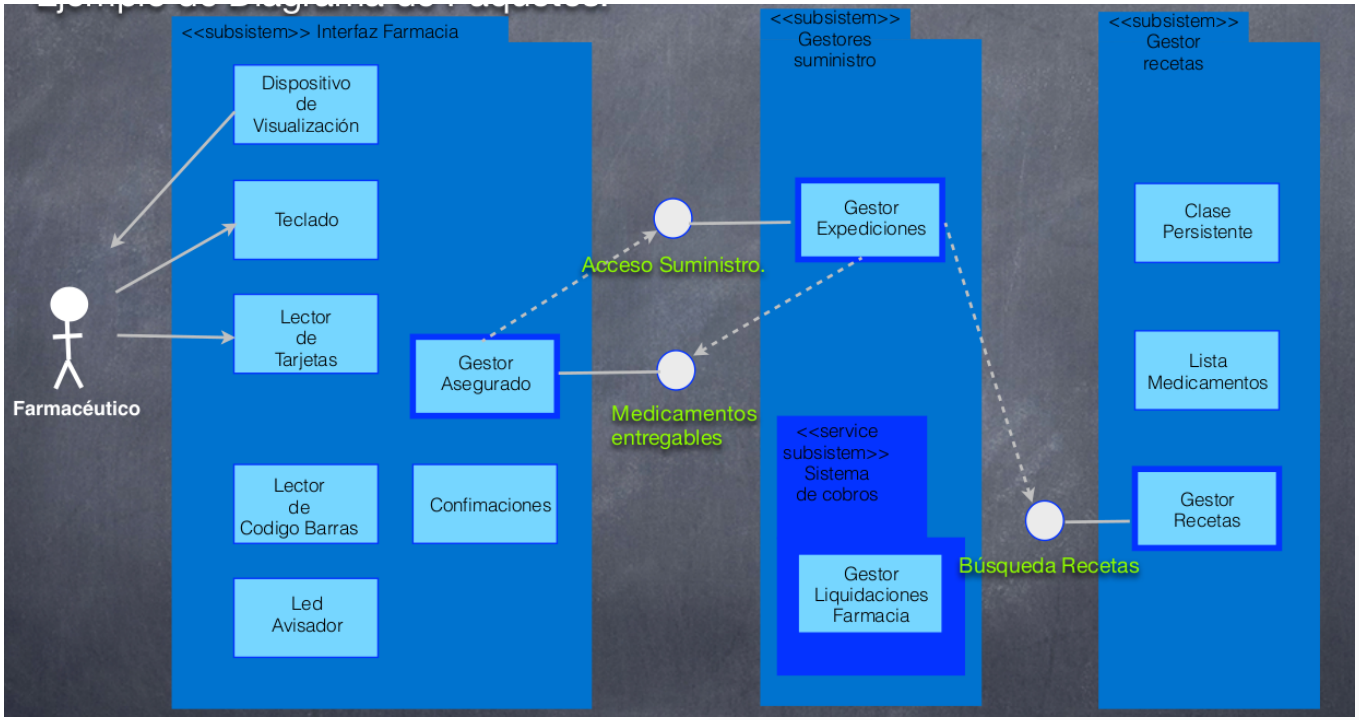
Subsistemas

- Los subsistemas pueden diseñarse de modo **ascendente** o **descendente**
  - Ascendente
    - Los desarrolladores proponen subsistemas basados en clases ya identificadas
    - Empaquetan estas clases en unidades con funciones bien definidas
  - Descendentes
    - El arquitecto identifica subsistemas de más alto nivel y sus interfaces antes de identificar las clases
    - Los desarrolladores trabajan con los subsistemas para identificar y diseñar las clases juntas.

Subsistemas

- Los subsistemas pueden diseñarse de modo **ascendente** o **descendente**
  - Ascendente
    - Los desarrolladores proponen subsistemas basados en clases ya identificadas
    - Empaquetan estas clases en unidades con funciones bien definidas
  - Descendentes
    - El arquitecto identifica subsistemas de más alto nivel y sus interfaces antes de identificar las clases
    - Los desarrolladores trabajan con los subsistemas para identificar y diseñar las clases juntas.
- Los susbstitemas desde un punto de vista de gestión sirven para desglosar y **reducir la complejidad** del modelo

Subsistemas: Ejemplo



Subsistemas: Ejemplo

- Se muestran tres interfaces

Subsistemas: Ejemplo

- Se muestran tres interfaces
  - Acceso Suministro

Subsistemas: Ejemplo

- Se muestran tres interfaces
  - Acceso Suministro
  - Medicamentos Entregables

Subsistemas: Ejemplo

- Se muestran tres interfaces
  - Acceso Suministro
  - Medicamentos Entregables
  - Búsqueda Recetas

Subsistemas: Ejemplo

- Se muestran tres interfaces
  - Acceso Suministro
  - Medicamentos Entregables
  - Búsqueda Recetas
- Estos interfaces permiten

Subsistemas: Ejemplo

- Se muestran tres interfaces
  - Acceso Suministro
  - Medicamentos Entregables
  - Búsqueda Recetas
- Estos interfaces permiten
  - **Acceso Suministro**: Definir las operaciones para gestionar los medicamentos las prescripciones de los Asegurados

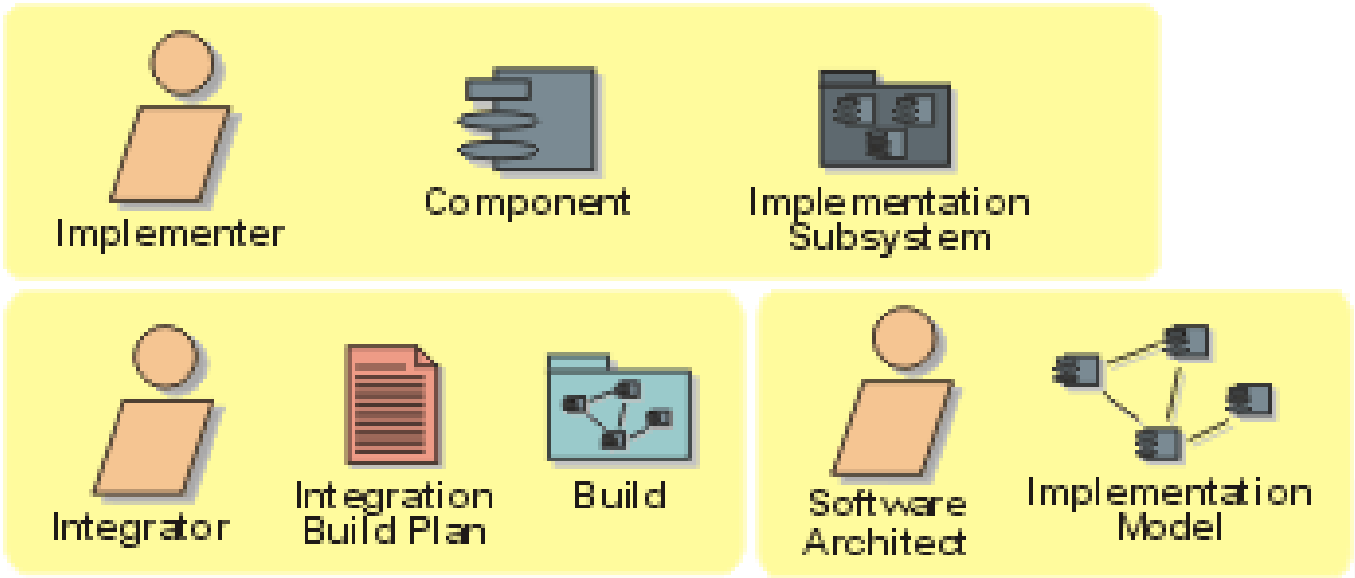
Subsistemas: Ejemplo

- Se muestran tres interfaces
  - Acceso Suministro
  - Medicamentos Entregables
  - Búsqueda Recetas
- Estos interfaces permiten
  - **Acceso Suministro**: Definir las operaciones para gestionar los medicamentos las prescripciones de los Asegurados
  - **Medicamentos Entregables**: Definir operaciones para solicitar la lista de medicamentos en fechas de entrega

Subsistemas: Ejemplo

- Se muestran tres interfaces
  - Acceso Suministro
  - Medicamentos Entregables
  - Búsqueda Recetas
- Estos interfaces permiten
  - **Acceso Suministro**: Definir las operaciones para gestionar los medicamentos las prescripciones de los Asegurados
  - **Medicamentos Entregables**: Definir operaciones para solicitar la lista de medicamentos en fechas de entrega
  - **Búsqueda Recetas**: Definir operaciones para gestionar la actualización del estado de las recetas

Implementación



Creación

Durante el flujo de Implementación se desarrolla todo lo necesario para obtener un sistema ejecutable

- Componentes:

Creación

Durante el flujo de Implementación se desarrolla todo lo necesario para obtener un sistema ejecutable

- Componentes:
  - Ejecutables.

Creación

Durante el flujo de Implementación se desarrolla todo lo necesario para obtener un sistema ejecutable

- Componentes:
  - Ejecutables.
  - De fichero (código fuente, scripts, ficheros de configuración).

Creación

Durante el flujo de Implementación se desarrolla todo lo necesario para obtener un sistema ejecutable

- Componentes:
  - Ejecutables.
  - De fichero (código fuente, scripts, ficheros de configuración).
  - De tabla (elementos de la persistencia, DAO,s etc.)

Creación

Durante el flujo de Implementación se desarrolla todo lo necesario para obtener un sistema ejecutable

- Componentes:
  - Ejecutables.
  - De fichero (código fuente, scripts, ficheros de configuración).
  - De tabla (elementos de la persistencia, DAO,s etc.)
- Un **componente** es una parte reemplazable del sistema que cumple y proporciona un conjunto de interfaces definidas.

Creación

Durante el flujo de Implementación se desarrolla todo lo necesario para obtener un sistema ejecutable

- Un componente se puede sustituir por otro, si proporciona y requiere las mismas interfaces que el anterior, y que lo enmarcaban operativamente en su contexto

Creación

Durante el flujo de Implementación se desarrolla todo lo necesario para obtener un sistema ejecutable

- Un componente se puede sustituir por otro, si proporciona y requiere las mismas interfaces que el anterior, y que lo enmarcaban operativamente en su contexto
  - Se prepara para que pueda **asignarse** a nodos del despliegue.

Creación

Durante el flujo de Implementación se desarrolla todo lo necesario para obtener un sistema ejecutable

- Un componente se puede sustituir por otro, si proporciona y requiere las mismas interfaces que el anterior, y que lo enmarcaban operativamente en su contexto
  - Se prepara para que pueda **asignarse** a nodos del despliegue.
  - Resuelven subsistemas de servicio del modelo de diseño

Creación

Durante el flujo de Implementación se desarrolla todo lo necesario para obtener un sistema ejecutable

- Un componente se puede sustituir por otro, si proporciona y requiere las mismas interfaces que el anterior, y que lo enmarcaban operativamente en su contexto
  - Se prepara para que pueda **asignarse** a nodos del despliegue.
  - Resuelven subsistemas de servicio del modelo de diseño
- **Subsistema** de Servicio:

Creación

Durante el flujo de Implementación se desarrolla todo lo necesario para obtener un sistema ejecutable

- Un componente se puede sustituir por otro, si proporciona y requiere las mismas interfaces que el anterior, y que lo enmarcaban operativamente en su contexto
  - Se prepara para que pueda **asignarse** a nodos del despliegue.
  - Resuelven subsistemas de servicio del modelo de diseño
- **Subsistema** de Servicio:
  - Se implementa mediante un componente si siempre se soporta en el mismo tipo de nodo

Creación

Durante el flujo de Implementación se desarrolla todo lo necesario para obtener un sistema ejecutable

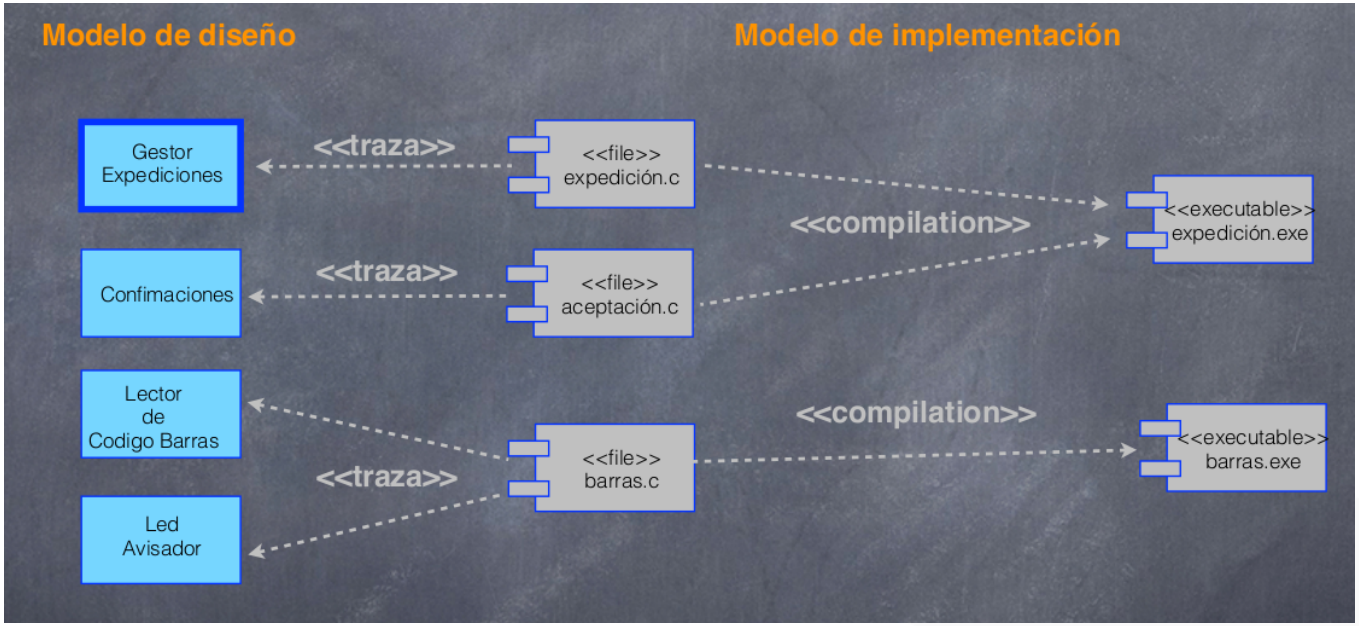
- Un componente se puede sustituir por otro, si proporciona y requiere las mismas interfaces que el anterior, y que lo enmarcaban operativamente en su contexto
  - Se prepara para que pueda **asignarse** a nodos del despliegue.
  - Resuelven subsistemas de servicio del modelo de diseño
- **Subsistema** de Servicio:
  - Se implementa mediante un componente si siempre se soporta en el mismo tipo de nodo
  - Se divide en partes si se sustenta sobre distintos tipos de nodo

Creación

Durante el flujo de Implementación se desarrolla todo lo necesario para obtener un sistema ejecutable

- Un componente se puede sustituir por otro, si proporciona y requiere las mismas interfaces que el anterior, y que lo enmarcaban operativamente en su contexto
  - Se prepara para que pueda **asignarse** a nodos del despliegue.
  - Resuelven subsistemas de servicio del modelo de diseño
- **Subsistema** de Servicio:
  - Se implementa mediante un componente si siempre se soporta en el mismo tipo de nodo
  - Se divide en partes si se sustenta sobre distintos tipos de nodo
  - Si un subsistema se divide, cada parte se implementará como un **componente**

Creación: Ejemplo



Creación: Ejemplo

- Los fuentes:

Creación: Ejemplo

- Los fuentes:
  - expedicion.c, que contiene el código fuente y documentación de la clase *Gestor expediciones*

Creación: Ejemplo

- Los fuentes:
  - expedicion.c, que contiene el código fuente y documentación de la clase *Gestor expediciones*
  - aceptación.c, que contiene el código fuente y documentación de la clase *Confirmaciones*

Creación: Ejemplo

- Los fuentes:
  - expedicion.c, que contiene el código fuente y documentación de la clase *Gestor expediciones*
  - aceptación.c, que contiene el código fuente y documentación de la clase *Confirmaciones*
  - barras.c, que contiene el código fuente y documentación de dos clases *Lector de Código de Barras* y *Avisador Led*

Creación: Ejemplo

- Los fuentes:
  - expedicion.c, que contiene el código fuente y documentación de la clase *Gestor expediciones*
  - aceptación.c, que contiene el código fuente y documentación de la clase *Confirmaciones*
  - barras.c, que contiene el código fuente y documentación de dos clases *Lector de Código de Barras* y *Avisador Led*
- Los ejecutables

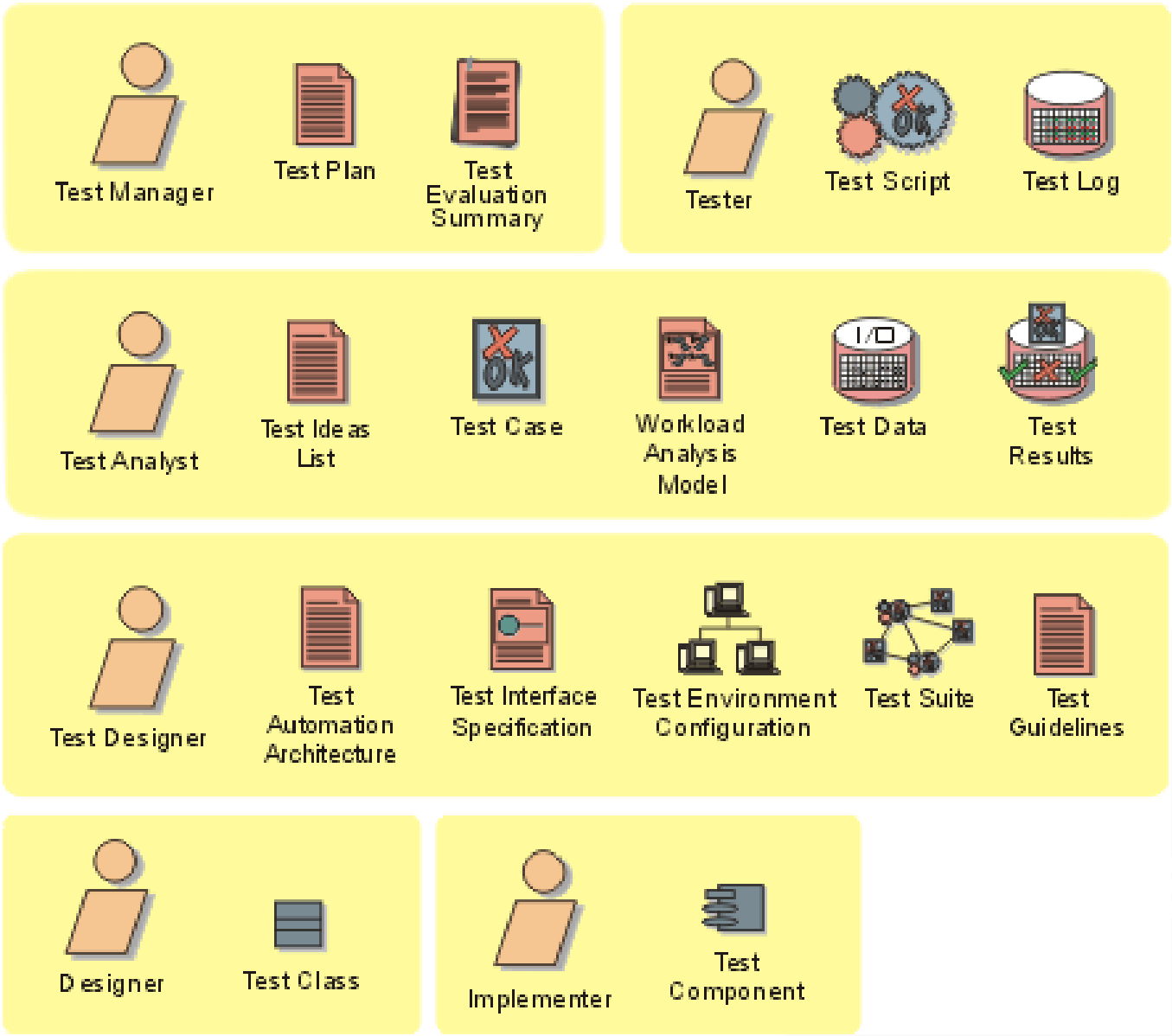
Creación: Ejemplo

- Los fuentes:
  - expedicion.c, que contiene el código fuente y documentación de la clase *Gestor expediciones*
  - aceptación.c, que contiene el código fuente y documentación de la clase *Confirmaciones*
  - barras.c, que contiene el código fuente y documentación de dos clases *Lector de Código de Barras* y *Avisador Led*
- Los ejecutables
  - expedición.exe, que es el resultado de la compilación de expedición.c y aceptación.c, que se enlazan

Creación: Ejemplo

- Los fuentes:
  - expedicion.c, que contiene el código fuente y documentación de la clase *Gestor expediciones*
  - aceptación.c, que contiene el código fuente y documentación de la clase *Confirmaciones*
  - barras.c, que contiene el código fuente y documentación de dos clases *Lector de Código de Barras* y *Avisador Led*
- Los ejecutables
  - expedición.exe, que es el resultado de la compilación de expedición.c y aceptación.c, que se enlazan
  - barras.exe, que es el resultado de la compilación de barras.c

Validación y prueba



- La disciplina de prueba actúa en muchos aspectos como un proveedor de servicios a las otras disciplinas

- La disciplina de prueba actúa en muchos aspectos como un proveedor de servicios a las otras disciplinas
  - Encontrar y documentar defectos en la calidad del software

- La disciplina de prueba actúa en muchos aspectos como un proveedor de servicios a las otras disciplinas
  - Encontrar y documentar defectos en la calidad del software
  - Asesorar sobre la percepción de calidad de software

- La disciplina de prueba actúa en muchos aspectos como un proveedor de servicios a las otras disciplinas
  - Encontrar y documentar defectos en la calidad del software
  - Asesorar sobre la percepción de calidad de software
  - Demostración de la validez de las suposiciones hechas en el diseño y especificaciones de requisitos a través de la demostración

- La disciplina de prueba actúa en muchos aspectos como un proveedor de servicios a las otras disciplinas
  - Encontrar y documentar defectos en la calidad del software
  - Asesorar sobre la percepción de calidad de software
  - Demostración de la validez de las suposiciones hechas en el diseño y especificaciones de requisitos a través de la demostración
  - Validación de las funciones del producto tal como fue diseñado

- La disciplina de prueba actúa en muchos aspectos como un proveedor de servicios a las otras disciplinas
  - Encontrar y documentar defectos en la calidad del software
  - Asesorar sobre la percepción de calidad de software
  - Demostración de la validez de las suposiciones hechas en el diseño y especificaciones de requisitos a través de la demostración
  - Validación de las funciones del producto tal como fue diseñado
  - Validar que los requisitos se han aplicado adecuadamente

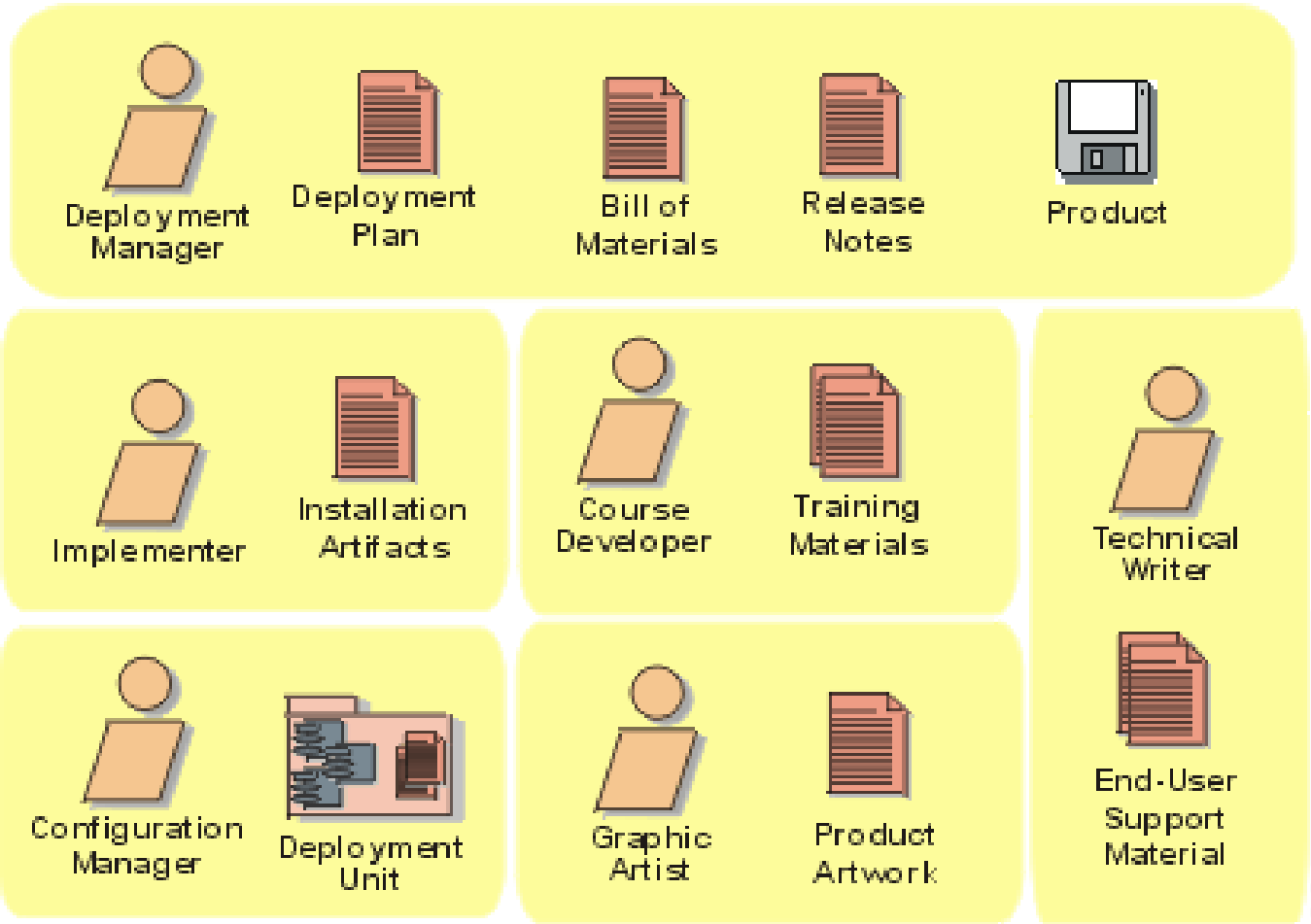
- La disciplina de prueba actúa en muchos aspectos como un proveedor de servicios a las otras disciplinas
  - Encontrar y documentar defectos en la calidad del software
  - Asesorar sobre la percepción de calidad de software
  - Demostración de la validez de las suposiciones hechas en el diseño y especificaciones de requisitos a través de la demostración
  - Validación de las funciones del producto tal como fue diseñado
  - Validar que los requisitos se han aplicado adecuadamente
- Existen distintos tipos de testing y multiples herramientas

- La disciplina de prueba actúa en muchos aspectos como un proveedor de servicios a las otras disciplinas
  - Encontrar y documentar defectos en la calidad del software
  - Asesorar sobre la percepción de calidad de software
  - Demostración de la validez de las suposiciones hechas en el diseño y especificaciones de requisitos a través de la demostración
  - Validación de las funciones del producto tal como fue diseñado
  - Validar que los requisitos se han aplicado adecuadamente
- Existen distintos tipos de testing y multiples herramientas

- La disciplina de prueba actúa en muchos aspectos como un proveedor de servicios a las otras disciplinas
  - Encontrar y documentar defectos en la calidad del software
  - Asesorar sobre la percepción de calidad de software
  - Demostración de la validez de las suposiciones hechas en el diseño y especificaciones de requisitos a través de la demostración
  - Validación de las funciones del producto tal como fue diseñado
  - Validar que los requisitos se han aplicado adecuadamente
- Existen distintos tipos de testing y multiples herramientas

> *Exploratory software testing is a powerful and fun approach to testing. In some situations, it can be orders of magnitude more productive than scripted testing.[...]. Yet few of us study this approach, and it doesn't get much respect in our field. It's high time we stop the denial, and publicly recognize the exploratory approach for what it is: scientific thinking in real-time*
>
> James Bach

Despliegue


Deployment Manager
Deployment Plan
Bill of Materials
Release Notes
Product
Implementer
Installation Artifacts
Course Developer
Training Materials
Technical Writer
Configuration Manager
Deployment Unit
Graphic Artist
Product Artwork
End-User Support Material

- Trata de hacer el **producto** de software **disponible** para el usuario final, y es la culminación de los esfuerzos de desarrollo

- Trata de hacer el **producto** de software **disponible** para el usuario final, y es la culminación de los esfuerzos de desarrollo
- Consta de:

- Trata de hacer el **producto** de software **disponible** para el usuario final, y es la culminación de los esfuerzos de desarrollo
- Consta de:
  - **Plan de Despliegue**: comienza temprano en el ciclo de vida y se dirige no sólo la producción del entregable, sino también el desarrollo de material de formación y material de soporte del sistema para garantizar que el usuario final puede utilizar el producto

- Trata de hacer el **producto** de software **disponible** para el usuario final, y es la culminación de los esfuerzos de desarrollo
- Consta de:
  - **Plan de Despliegue**: comienza temprano en el ciclo de vida y se dirige no sólo la producción del entregable, sino también el desarrollo de material de formación y material de soporte del sistema para garantizar que el usuario final puede utilizar el producto
  - **Material de apoyo**: cubre toda la gama de información que le sea requerida por el usuario final para instalar, operar, utilizar y mantener el sistema de entrega

- Trata de hacer el **producto** de software **disponible** para el usuario final, y es la culminación de los esfuerzos de desarrollo
- Consta de:
  - **Plan de Despliegue**: comienza temprano en el ciclo de vida y se dirige no sólo la producción del entregable, sino también el desarrollo de material de formación y material de soporte del sistema para garantizar que el usuario final puede utilizar el producto
  - **Material de apoyo**: cubre toda la gama de información que le sea requerida por el usuario final para instalar, operar, utilizar y mantener el sistema de entrega
  - **Unidad de Despliegue**: creación de una versión del producto que consiste en el software, y los artefactos de acompañamiento requeridos para instalar y utilizarlo de manera eficaz

- Trata de hacer el **producto** de software **disponible** para el usuario final, y es la culminación de los esfuerzos de desarrollo
- Consta de:
  - **Plan de Despliegue**: comienza temprano en el ciclo de vida y se dirige no sólo la producción del entregable, sino también el desarrollo de material de formación y material de soporte del sistema para garantizar que el usuario final puede utilizar el producto
  - **Material de apoyo**: cubre toda la gama de información que le sea requerida por el usuario final para instalar, operar, utilizar y mantener el sistema de entrega
  - **Unidad de Despliegue**: creación de una versión del producto que consiste en el software, y los artefactos de acompañamiento requeridos para instalar y utilizarlo de manera eficaz
- Es importante asegurar que el producto está bien probado antes de su lanzamiento

- Trata de hacer el **producto** de software **disponible** para el usuario final, y es la culminación de los esfuerzos de desarrollo
- Consta de:
  - **Plan de Despliegue**: comienza temprano en el ciclo de vida y se dirige no sólo la producción del entregable, sino también el desarrollo de material de formación y material de soporte del sistema para garantizar que el usuario final puede utilizar el producto
  - **Material de apoyo**: cubre toda la gama de información que le sea requerida por el usuario final para instalar, operar, utilizar y mantener el sistema de entrega
  - **Unidad de Despliegue**: creación de una versión del producto que consiste en el software, y los artefactos de acompañamiento requeridos para instalar y utilizarlo de manera eficaz
- Es importante asegurar que el producto está bien probado antes de su lanzamiento
  - Debe estar suficientemente probado en el entorno de prueba de desarrollo, y luego volver a probar en el sitio objetivo

- Trata de hacer el **producto** de software **disponible** para el usuario final, y es la culminación de los esfuerzos de desarrollo
- Consta de:
  - **Plan de Despliegue**: comienza temprano en el ciclo de vida y se dirige no sólo la producción del entregable, sino también el desarrollo de material de formación y material de soporte del sistema para garantizar que el usuario final puede utilizar el producto
  - **Material de apoyo**: cubre toda la gama de información que le sea requerida por el usuario final para instalar, operar, utilizar y mantener el sistema de entrega
  - **Unidad de Despliegue**: creación de una versión del producto que consiste en el software, y los artefactos de acompañamiento requeridos para instalar y utilizarlo de manera eficaz
- Es importante asegurar que el producto está bien probado antes de su lanzamiento
  - Debe estar suficientemente probado en el entorno de prueba de desarrollo, y luego volver a probar en el sitio objetivo