UNLAM 2017

Diseño de Sistemas
Construcción II

Introducción a los conceptos de **Ciclo de Vida del Desarrollo de Software**

Incluyendo :

RUP (Rational Unified Process), UML mención ALM

Referencia: Fuentes originales IBM y HPE

Qué es un Proceso?

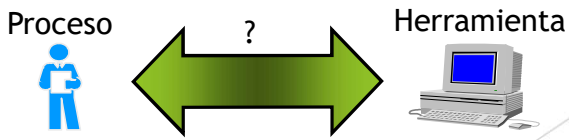
- Un proceso define **Quién** está haciendo **Qué**, **Cuándo** y **Cómo** para lograr un cierto objetivo. En la ingeniería de software el objetivo es construir un producto de software ó mejorar alguno existente.

El Problema

•Si un proceso es utilizado, equipos funcionales diferentes normalmente utilizan procesos y lenguajes de modelación inconsistentes.

•La mayoría de los proyectos de software utilizan procesos que no están bien definidos. En su lugar los miembros del equipo (re)inventan sus propios procesos.

•Los procesos no están apropiadamente relacionados con herramientas, ó no están propiamente automatizados.

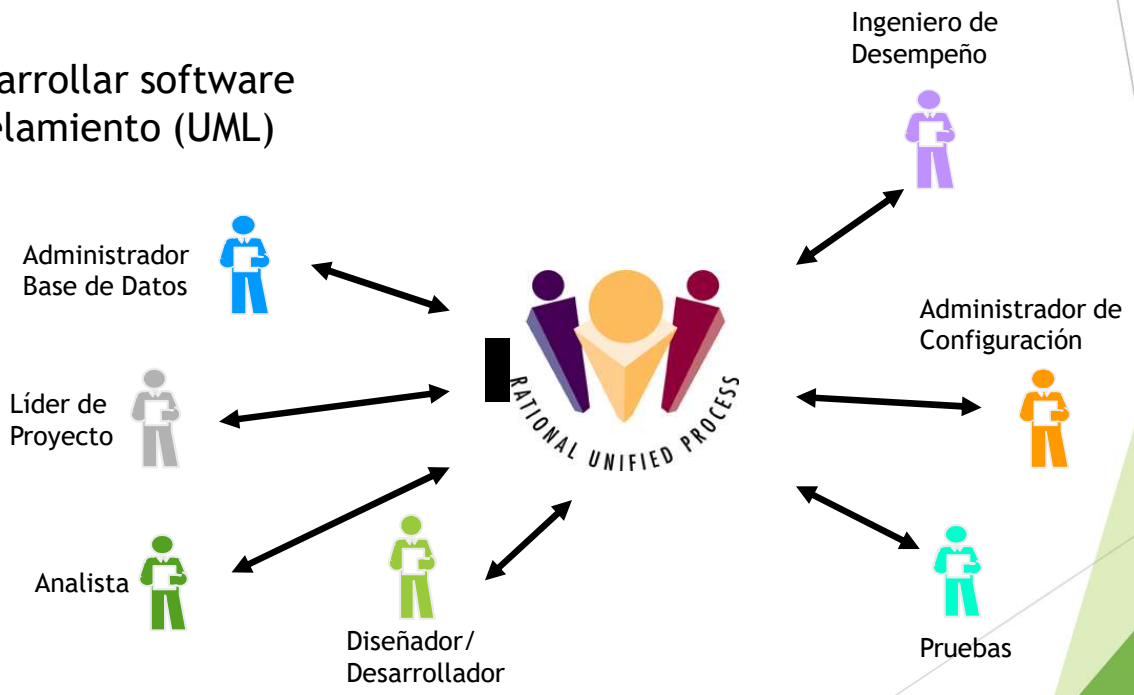


Rational Unified Process (RUP)

- Captura varias de las *mejores prácticas* en el desarrollo moderno de software en una forma que es aplicable para un amplio rango de proyectos y organizaciones.
- Es una guía de cómo utilizar de manera efectiva *UML*.
- Provee a cada miembro de un equipo un fácil acceso a una base de conocimiento con guías, plantillas y herramientas para todas las actividades críticas de desarrollo.
- Crea y mantiene *modelos*, en lugar de enfocarse en la producción de una gran cantidad de papeles de documentación.

Incremento de la Productividad en Equipo

Todos los miembros del equipo comparten
- 1 Base de conocimiento
- 1 Proceso
- 1 Vista de cómo desarrollar software
- 1 Lenguaje de modelamiento (UML)



6 Mejores Prácticas (Best Practices)

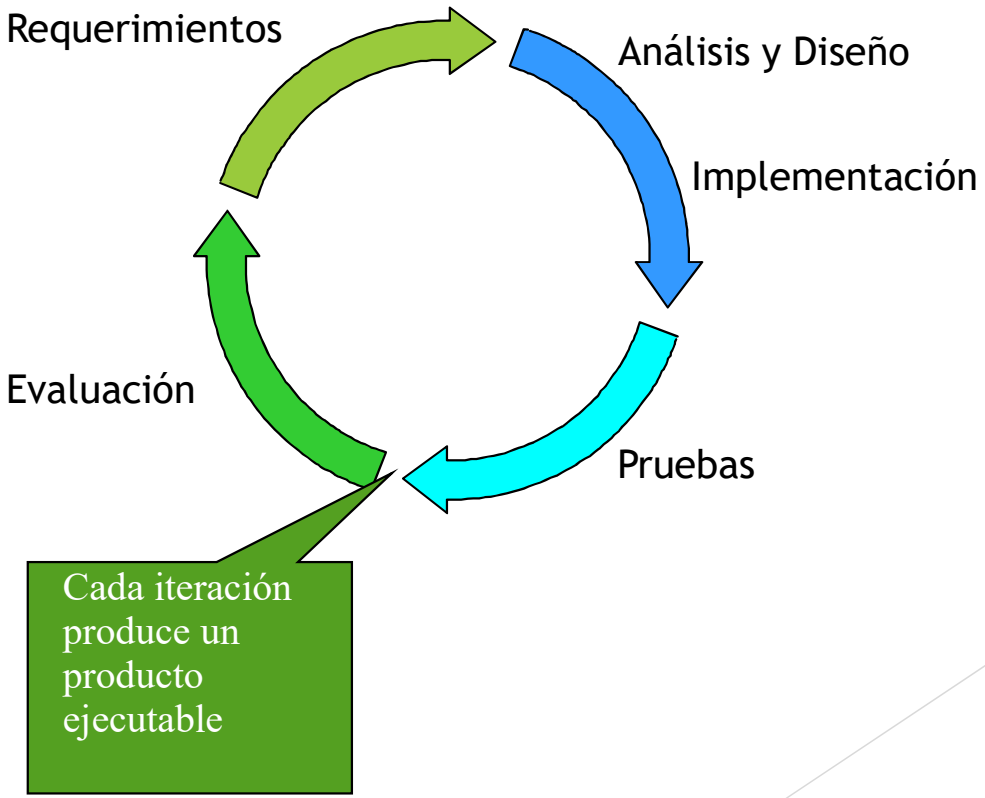
- RUP describe como utilizar de forma efectiva procedimientos comerciales probados en el desarrollo de software para equipos de desarrollo de software, conocidos como “mejores prácticas”.

Desarrollo Iterativo de Software

- Dados los sistemas de software sofisticados de la actualidad, no es posible hacer de manera secuencial la definición completa del problema, diseñar la solución completa, construir el software y por último probarlo.
- El descubrimiento de defectos en fases posteriores de diseño dan como resultado un aumento en los costos y/ó la cancelación del proyecto.

El tiempo y dinero gastados en la implementación de un diseño fallido, son no recuperables

Desarrollo Iterativo



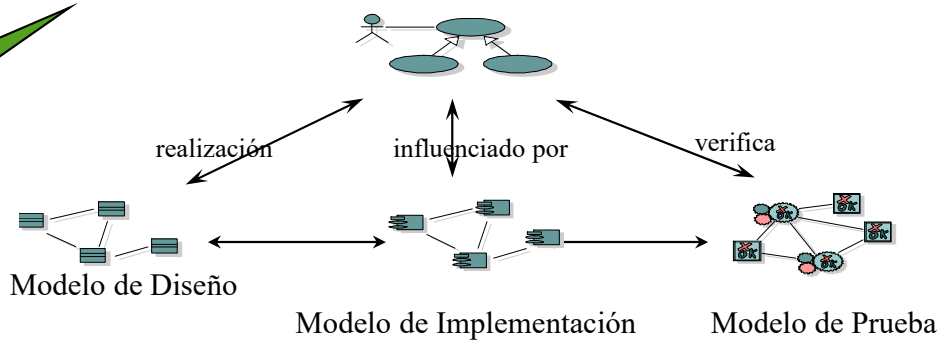
Características del Desarrollo Iterativo

- Permite un entendimiento incremental del problema a través de refinamientos sucesivos.
- Habilita una fácil retroalimentación de usuario.
- Metas específicas permiten que el equipo de desarrollo mantenga su atención en producir resultados.
- El progreso es medido conforme avanzan las implementaciones.

Administración de Requerimientos

- Elicitar, organizar, y documentar la funcionalidad y restricciones requeridas.
- Llevar un registro y documentación de cambios y decisiones.
- Los requerimientos de negocio son fácilmente capturados y comunicados a través de casos de uso.
- Los casos de uso son instrumentos importantes de planeación.

Los casos de uso dirigen el trabajo desde el análisis hasta las pruebas



Arquitectura Basada en Componentes

- Se enfoca en el pronto desarrollo de una arquitectura ejecutable robusta.
  - Resistente al cambio mediante el uso de interfaces bien definidas.
  - Intuitivamente comprensible.
  - Promueve un reuso más efectivo de software.
  - Es derivada a partir de los casos de uso más importantes.

Modelación Visual de Software

- Captura la estructura y comportamiento de arquitecturas y componentes.
- Muestra como encajan de forma conjunta los elementos del sistema.
- Mantiene la consistencia entre un diseño y su implementación.
- Promueve una comunicación no ambigua.

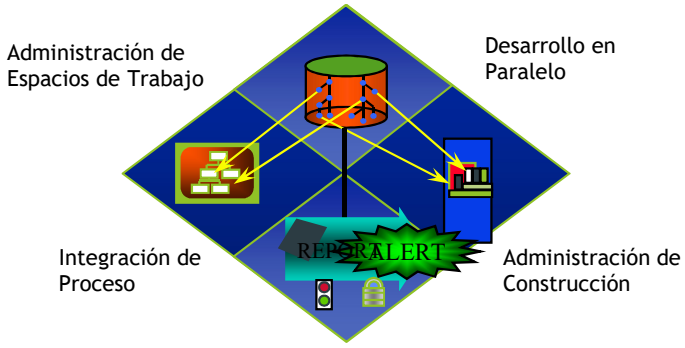
Verificación de la Calidad del Software

- Crea pruebas para cada escenario (casos de uso) para asegurar que todos los requerimientos están propiamente implementados.
- Verifica la calidad del software con respecto a los requerimientos basados en la confiabilidad, funcionalidad, desempeño de la aplicación y del sistema.
- Prueba cada iteración

Los problemas del software son de 100 a 1000 veces mas costosos de encontrar y reparar después del desarrollo

Control de Cambios del Software

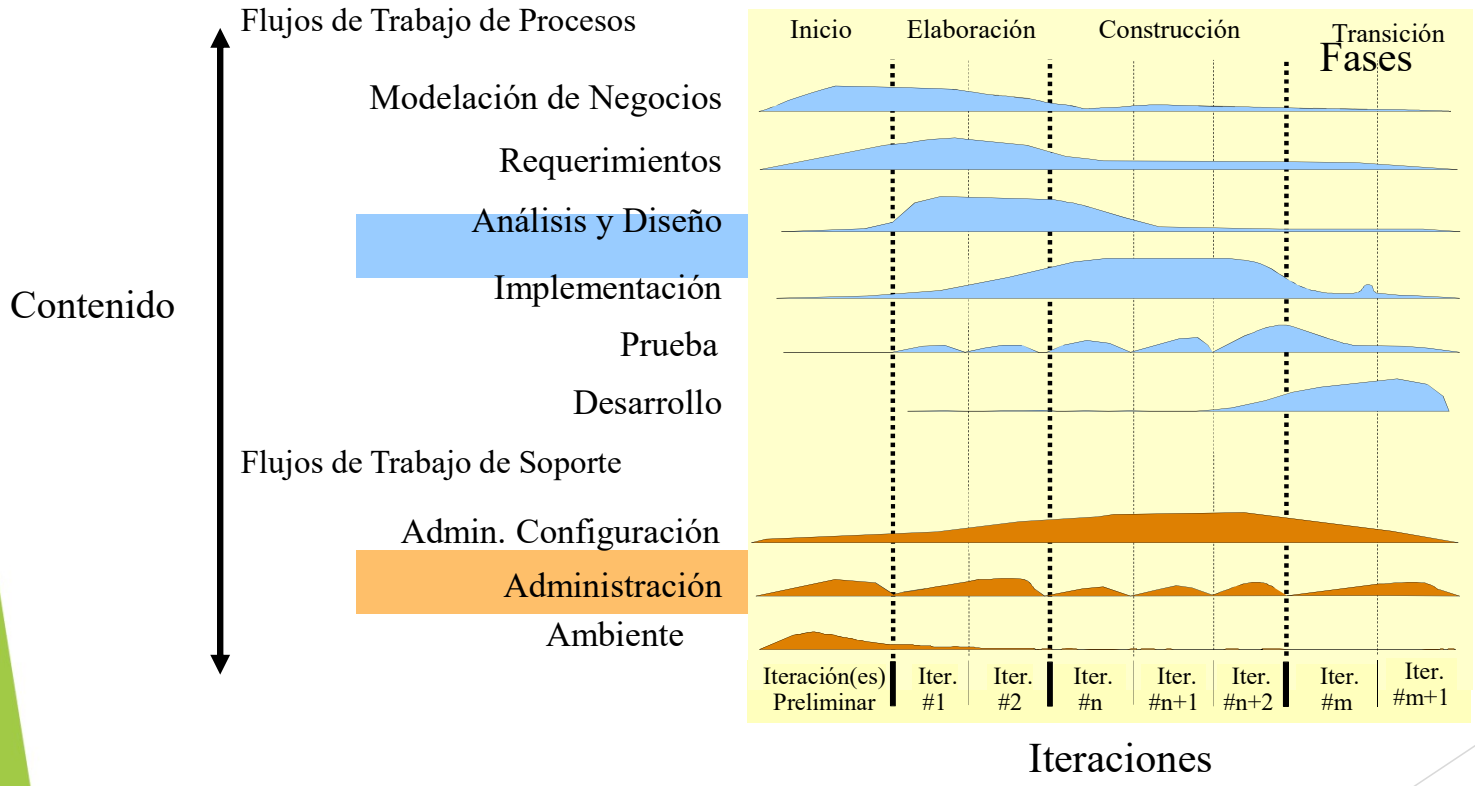
- Controlar, llevar un registro y monitorear cambios para permitir un desarrollo iterativo.
- Establece espacios de trabajo seguros para cada desarrollador
  - Provee aislamiento de cambios hechos en otros espacios de trabajo
  - Controla todos los artefactos de software – modelos, código, documentos, etc…



Estructura de RUP

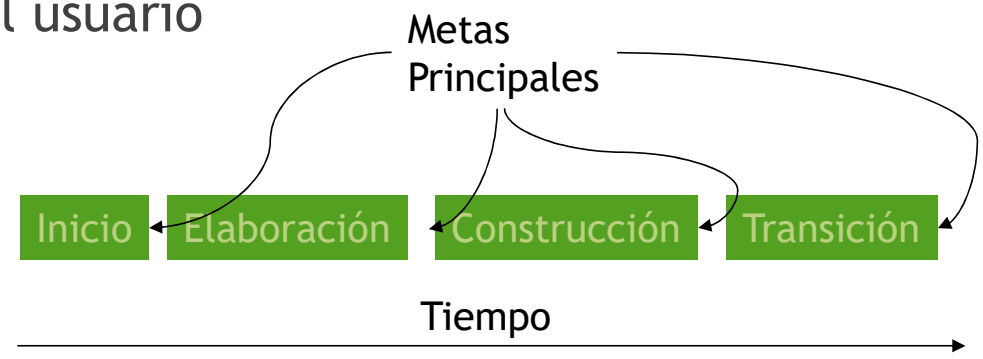
- El proceso puede describirse en dos dimensiones, o a lo largo de dos ejes:
  - El eje horizontal representa *tiempo* y muestra el aspecto dinámico del proceso, expresado en términos de *ciclos*, *fases*, *iteraciones*, y *metas*.
  - El eje vertical representa el aspecto estático del proceso; como está descrito en términos de *actividades*, *artefactos*, *trabajadores* y *flujos de trabajo*.

Estructura de RUP Cont.



Fases en RUP

- **Inicio** – Define el alcance del proyecto
- **Elaboración** – Plan del proyecto, especificación de características, arquitectura base
- **Construcción** – Construir el producto
- **Transición** – Transición del producto a la comunidad del usuario

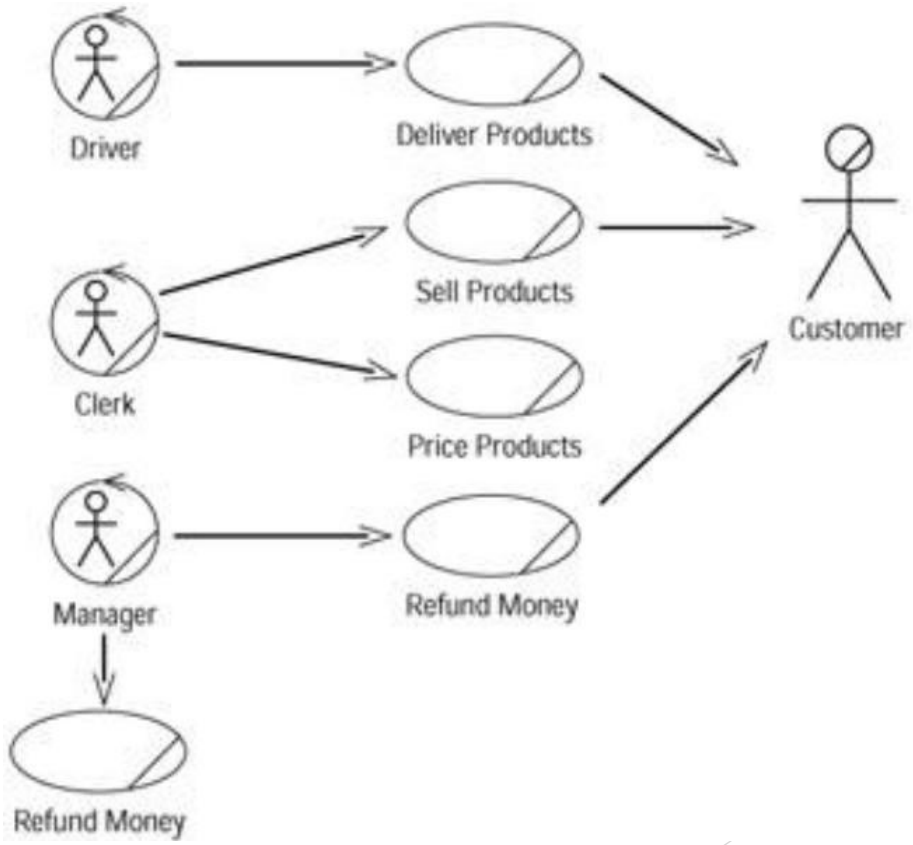


Fase de Inicio

- Propósito
  - Establecer casos de negocios para un nuevo sistema o para alguna actualización importante de un sistema existente
  - Especificar el alcance del proyecto
- Resultado
  - Una visión general de los requerimientos del proyecto, i.e., los requerimientos principales
    - Un modelo inicial de casos de uso y modelo del dominio (10-20%)
  - Un caso de negocios inicial, incluyendo:
    - Evaluación inicial de riesgos
    - Una estimación de los recursos requeridos

Ejemplo de Diagrama de Caso de Uso de Negocios

Caso de Negocios: modelar la empresa (como funciona la empresa a la que se le va a desarrollar el software)



Fase de Elaboración

- Propósito
  - Analizar el dominio del problema
  - Establecer una buena arquitectura
  - Lidiar con los elementos de riesgo más altos del proyecto
  - Desarrollar un plan comprensivo mostrando como el proyecto será completado
- Resultado
  - Un modelo del dominio y de casos de uso 80% completo
  - Requerimientos suplementarios que capturen los requerimientos no funcionales y cualesquiera requerimientos que no estén asociados con un caso de uso específico
  - Una lista de riesgos revisada

Fase de Construcción

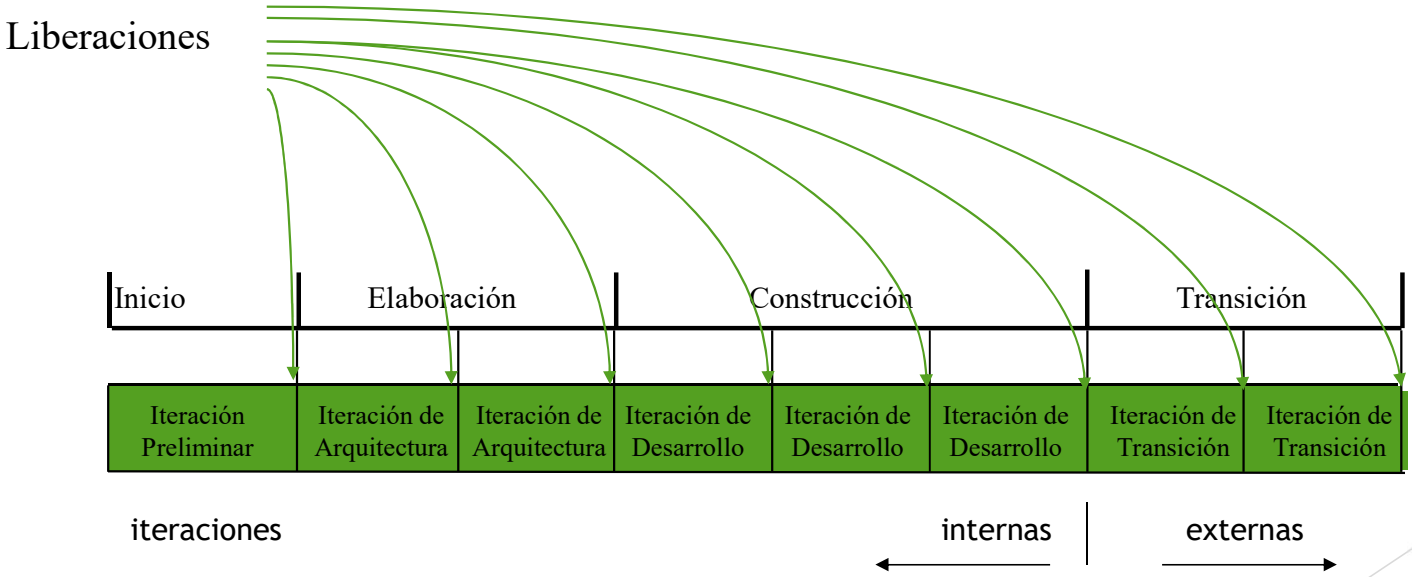
- Propósito
  - Desarrollar incrementalmente producto de software completo el cual estará listo para ser transferido al usuario
- Productos
  - Un modelo completo de diseño y casos de uso
  - Liberaciones de productos ejecutables de funcionalidad incremental
  - Documentación de usuario
  - Una liberación “beta” del producto

Fase de Transición

- Hacer la transición final del producto de software al usuario
- Productos
  - Liberaciones ejecutables de producto
  - “Pruebas beta” para validar el nuevo sistema vs. las expectaciones del usuario
  - Manuales de usuario actualizados
  - Documentación de desarrollo actualizada
- Está el usuario satisfecho?
- Gastos reales de los recursos vs. Gastos previstos ← Aceptables?

Iteraciones

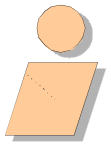
- Cada fase en RUP puede descomponerse en iteraciones. Una *iteración* es un ciclo de desarrollo completo dando como resultado una entrega de producto ejecutable (interna o externa)



Noción de Proceso

Trabajador/Quién?

Rol que puede ser desempeñado por un individuo o conjunto de individuos en la organización de desarrollo



Diseñador

responsable de

Caso de Uso

Paquete de Caso de Uso

Actividad/Cómo?

Diseño de Casos de uso

Describe una unidad de trabajo que puede ser asignada a un trabajador.

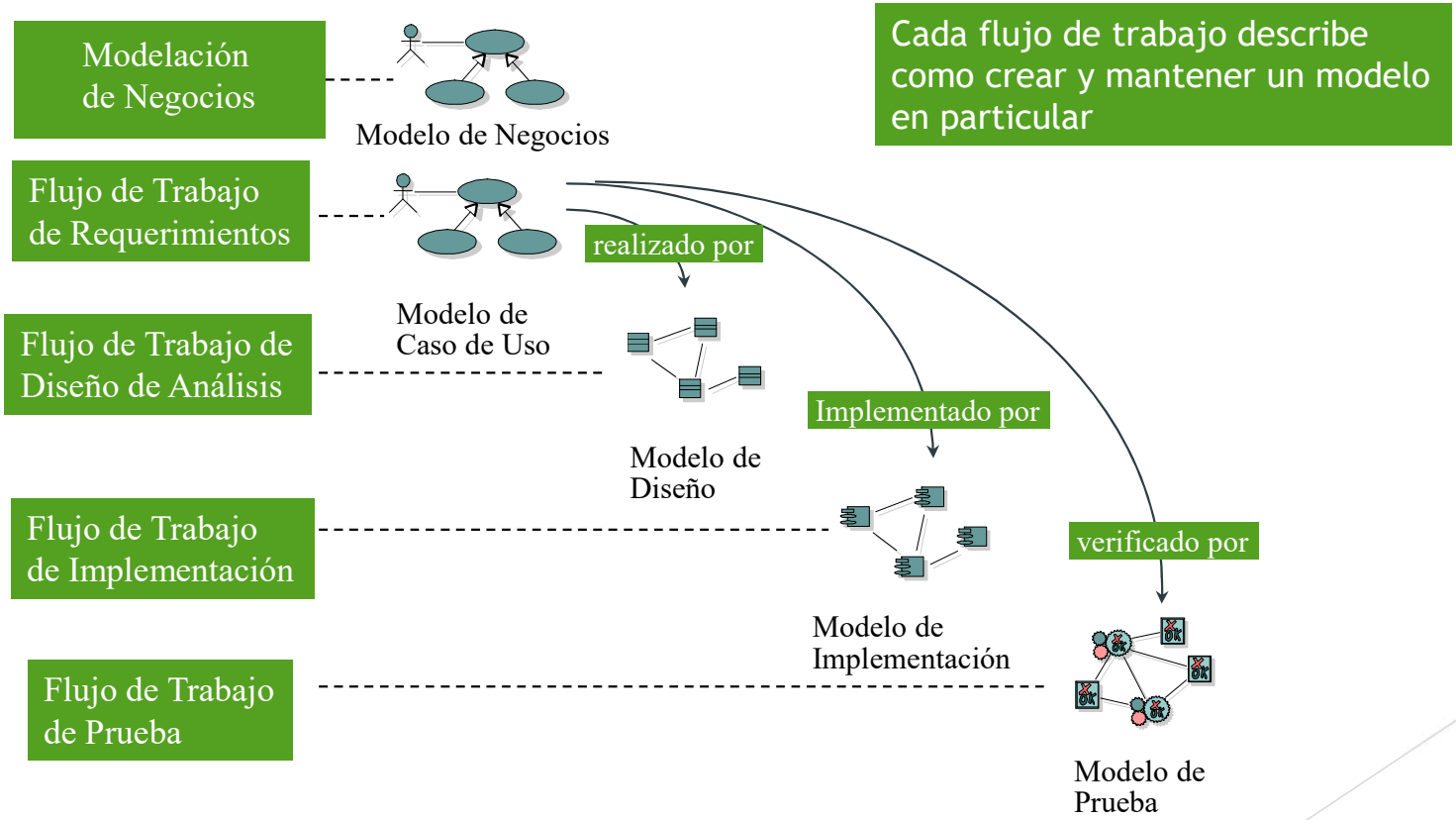
Artefacto/Qué?

Pieza de información que es producida, modificada, ó utilizada por un proceso

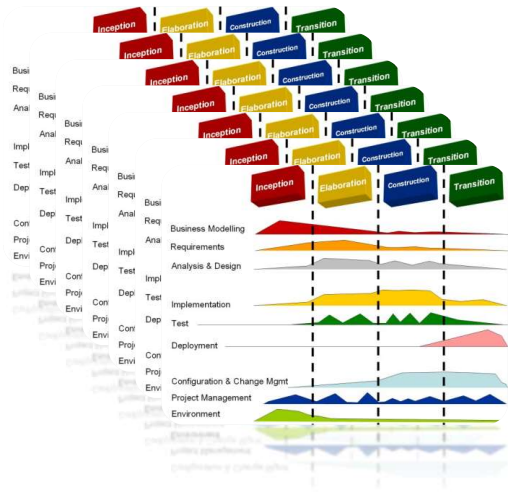
Modelos y Flujos de Trabajo

- Una mera enumeración de todos los trabajadores, actividades y artefactos no constituyen un proceso. Se necesita una forma de describir secuencias significativas que produzcan algún resultado válido, y que muestre la interacción entre trabajadores.
- Un *flujo de trabajo* es una secuencia de actividades que producen un resultado de valor observable.
- En términos de UML pueden ser expresados como un diagrama de secuencia, un diagrama de colaboración, ó como un diagrama de actividad.
- Los grupos de trabajo agrupan actividades en forma lógica

Modelos y Flujos de Trabajo Cont.



Parte II:
Visión Holística y Deep dive al Proceso Unificado



CONCEPTOS FUNDAMENTALES

Proceso:

- Compuesto por actividades de trabajo y actividades de protección.

Producto:

- Resultado del proceso.

CONCEPTOS FUNDAMENTALES

Fase:

Iteración:

CONCEPTOS FUNDAMENTALES

- **CICLO DE VIDA DEL SOFTWARE:**
  - Es el conjunto de fases por las que pasa el software, que abarcan desde su creación u origen, hasta su eliminación o liquidación formal.
- **MODELO DE DESARROLLO:**
  - También denominado *Modelo de Proceso*.
  - Estrategia de desarrollo basada en el ciclo de vida, naturaleza del proyecto y metodología, que determina las características específicas del proceso (Pressman 2001).

EL PROCESO UNIFICADO

El Proceso Unificado:

A. Es un Proceso iterativo.
B. Está centrado en la arquitectura.
C. Está dirigido por los casos de uso.
D. Es un proceso configurable.
E. Soporta las técnicas orientadas a objetos.
F. Impulsa un control de calidad y una gestión del riesgo objetivos y continuos.

EL PROCESO UNIFICADO

- **A. RUP ES UN PROCESO ITERATIVO:**

  - Un enfoque iterativo propone una comprensión incremental del problema.
  - Como parte del enfoque iterativo se encuentra la flexibilidad para acomodarse a nuevos requisitos o a cambios tácticos en los objetivos del negocio.
  - Permite que el proyecto identifique y resuelva los riesgos más bien pronto que tarde.

EL PROCESO UNIFICADO

- **B. ASPECTOS DEL RUP:**
  - El desarrollo bajo el Proceso Unificado está **centrado en la arquitectura.**
  - El proceso se centra en establecer al principio una arquitectura software que guía el desarrollo del sistema:
    - Se facilita el desarrollo en paralelo.
    - Se minimiza la repetición de trabajos.
    - Se incrementa la probabilidad de reutilización de componentes y el mantenimiento posterior del sistema.

EL PROCESO UNIFICADO

- **C. ASPECTOS DEL RUP:**
  - Las actividades de desarrollo bajo el Proceso Unificado están **dirigidas por los casos de uso.**
  - El Proceso Unificado pone un gran énfasis en la construcción de sistemas basada en una amplia comprensión de cómo se utilizará el sistema que se entregue.
  - Las nociones de los casos de uso y los escenarios se utilizan para guiar el flujo de procesos desde la captura de los requisitos hasta las pruebas, y para proporcionar caminos que se pueden reproducir durante el desarrollo del sistema.

EL PROCESO UNIFICADO

- **D. ASPECTOS DEL RUP:**
  - El Proceso Unificado es **un proceso configurable.**
  - Aunque un único proceso no es adecuado para todas las organizaciones de desarrollo de software, el Proceso Unificado es adaptable y puede configurarse para cubrir las necesidades de proyectos que van desde pequeños equipos de desarrollo de software hasta grandes empresas de desarrollo.
  - También se basa en una arquitectura de proceso simple y clara, que proporciona un marco común a toda una familia de procesos y que, además, puede variarse para acomodarse a distintas situaciones.

EL PROCESO UNIFICADO

- **E. ASPECTOS DEL RUP:**
  - El Proceso Unificado **soporta las técnicas orientadas a objetos.**
  - Los modelos del Proceso Unificado se basan en los conceptos de objeto y clase y las relaciones entre ellos, y utilizan UML como la notación común.

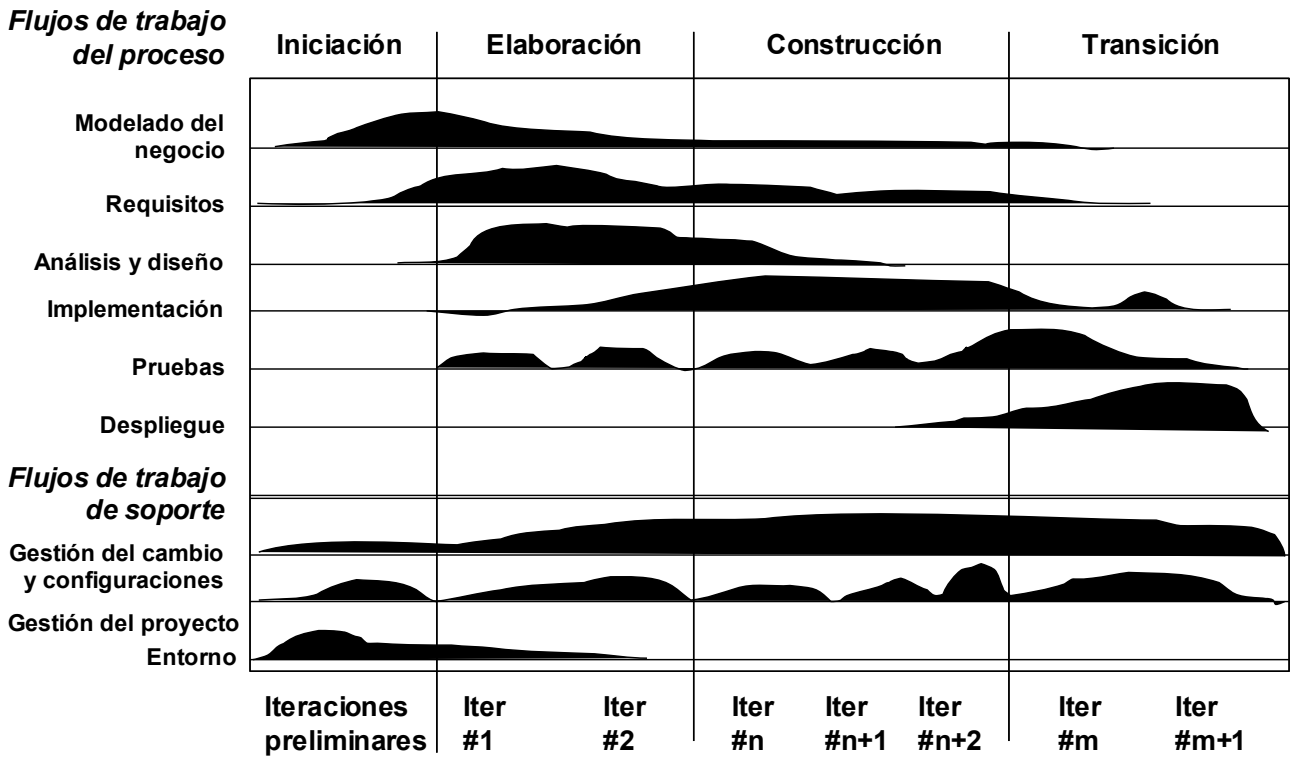
EL PROCESO UNIFICADO

- **F. ASPECTOS DEL RUP:**
  - El Proceso Unificado es **impulsa un control de calidad y una gestión del riesgo objetivos y continuos.**
  - La evaluación de la calidad va contenida en el proceso, en todas las actividades, e implicando a todos los participantes, mediante medidas y criterios objetivos. No se trata como algo a posteriori o una actividad separada.
  - La gestión del riesgo va contenida en el proceso, de manera que los riesgos para el éxito del proyecto se identifican y se acometen al principio del proceso de desarrollo, cuando todavía hay tiempo de reaccionar.

EL PROCESO UNIFICADO

- El Proceso Unificado tiene una estructura matricial donde se relacionan esfuerzos y tiempos:
  - Los tiempos están definidos por las fases y las iteraciones.
  - Los esfuerzos están definidos por los flujos de trabajo del proceso y de soporte.
  - La representación gráfica se denomina en la jerga el *Diagrama de Montañas*.

El ciclo de vida del desarrollo del software



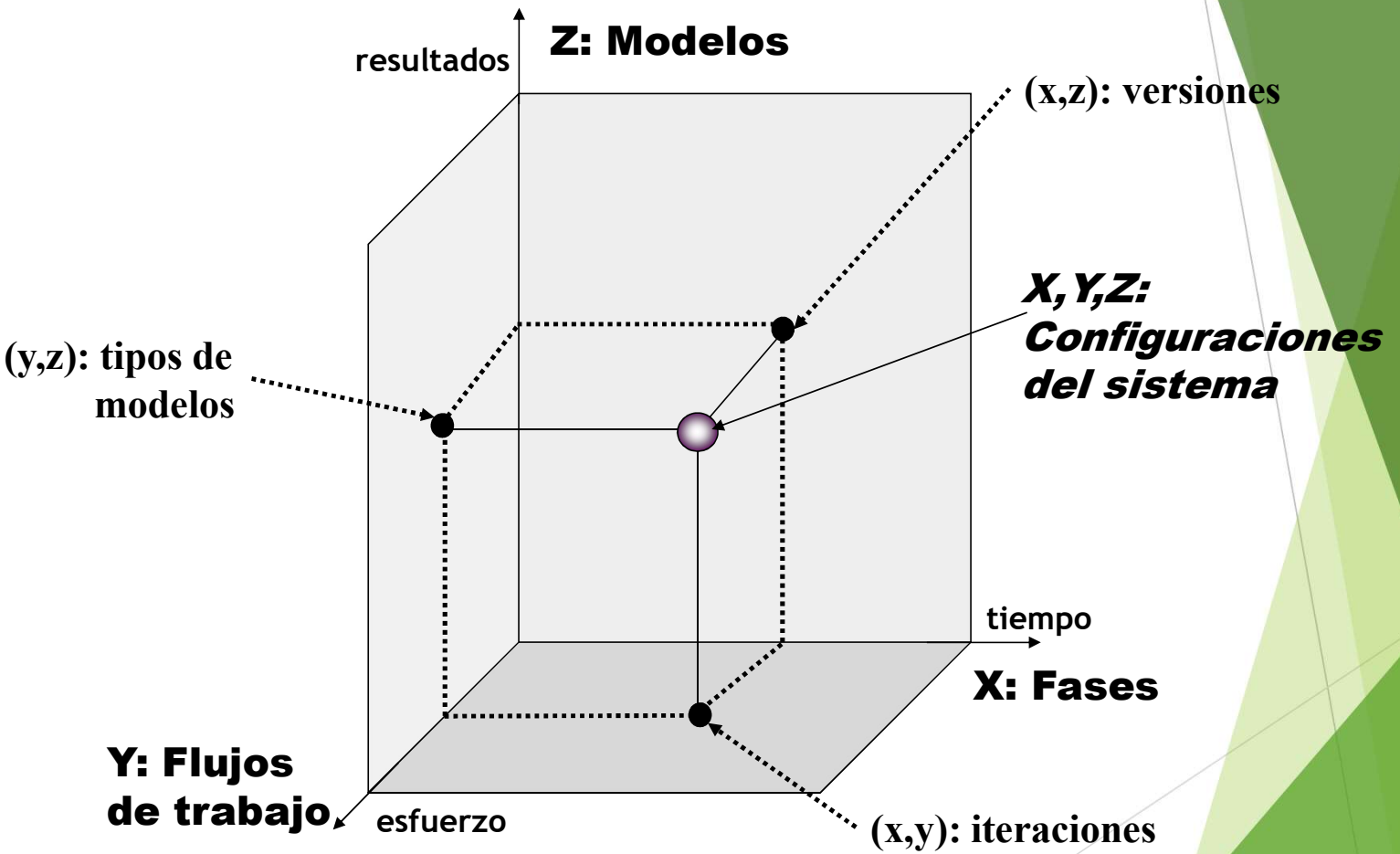
Fuente: Jacobson et al., 2000

EL PROCESO UNIFICADO

- En esta estructura matricial se puede deducir que:
  - Los ***resultados*** de los flujos de trabajo de proceso son los **MODELOS.**
  - La conjunción de tiempo (fases) y esfuerzos (flujos de trabajo) da lugar a las ***iteraciones***.
  - La conjunción de resultados (modelos) y esfuerzos (flujos de trabajo) da lugar a los ***tipos de modelos***.
  - La conjunción de tiempo (fases) y resultados (modelos) da lugar a las ***versiones***.

EL PROCESO UNIFICADO

- Se puede representar esta estructura conceptual (metamodelo) mediante una figura tridimensional donde:
  - ***Eje X: Fases → tiempo***
  - ***Eje Y: Flujos de trabajo → esfuerzos***
  - ***Eje Z: Modelos → resultados***


Z: Modelos
resultados
(x,z): versiones
X,Y,Z:
Configuraciones
del sistema
(y,z): tipos de
modelos
tiempo
X: Fases
Y: Flujos
de trabajo
esfuerzo
(x,y): iteraciones

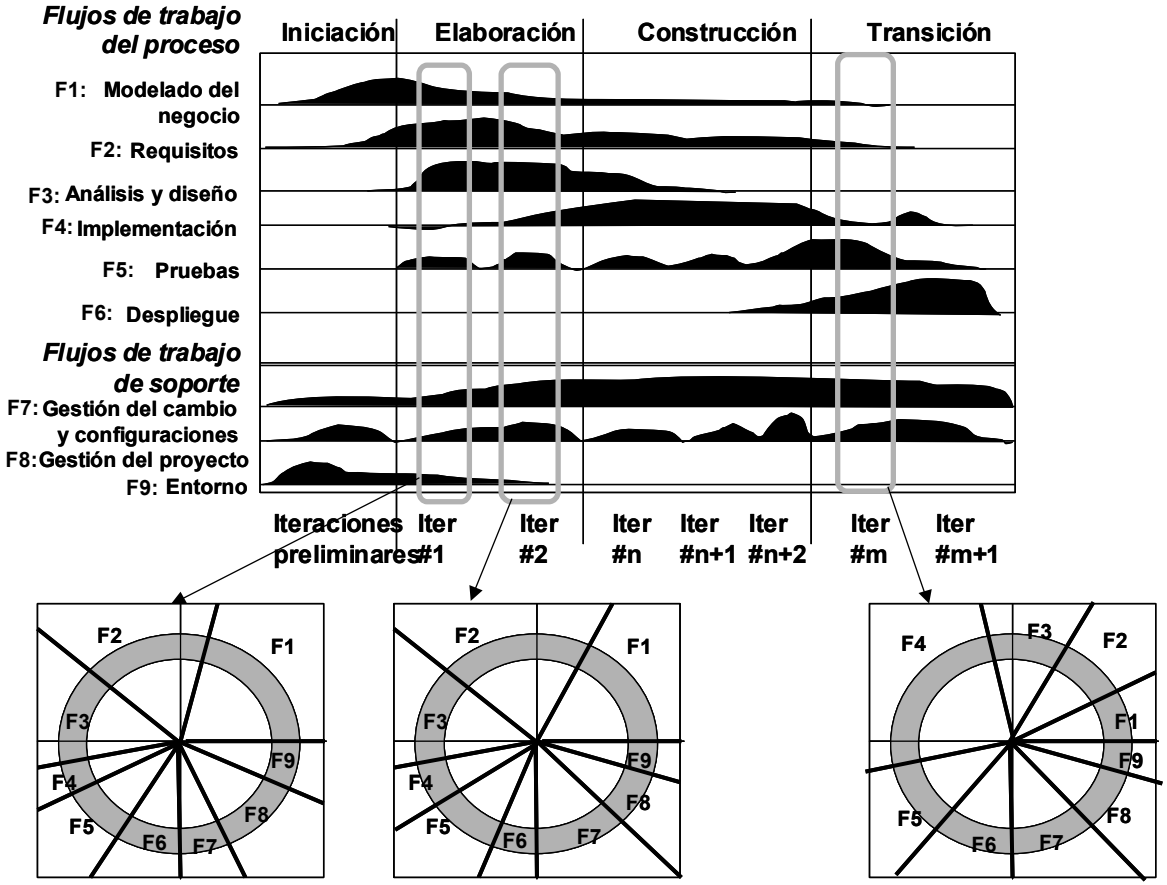
Fases del ciclo

- **Fase**: es el intervalo de tiempo entre dos hitos importantes del proceso durante el que se cumple un conjunto bien definido de objetivos, se completan artefactos y se toman decisiones sobre si pasar o no a la siguiente fase.
- Dentro de cada fase hay varias iteraciones
  - **Iteración**: representa un ciclo de desarrollo completo, desde la captura de requisitos en el análisis hasta la implementación y pruebas, que produce como resultado la entrega al cliente o la salida al mercado de un proyecto ejecutable.

Fases del ciclo

- ***Iniciación***.
  - Se establece la planificación del proyecto y se delimita su alcance.
- ***Elaboración***.
  - Se analiza el dominio del problema, se establece una base arquitectónica sólida, se desarrolla el plan del proyecto y se eliminan los elementos de más alto riesgo del proyecto.
- ***Construcción***.
  - Se desarrolla de forma iterativa e incremental un producto completo que está preparado para la transición hacia la comunidad de usuarios.
- ***Transición***.
  - El software se despliega en la comunidad de usuarios.

Las iteraciones son distintas en el ciclo de vida



FASES DEL CICLO

- Cada iteración pasa a través de varios flujos de trabajo del proceso, aunque con un énfasis diferente en cada uno de ellos, dependiendo de la fase en que se encuentre:
  - Durante la *iniciación*, el interés se orienta hacia el análisis y el diseño.
  - También durante la *elaboración*.
  - Durante la *construcción*, la actividad central es la implementación.
  - La *transición* se centra en despliegue.

Esfuerzos

- Los esfuerzos aplicados en el ciclo de vida de desarrollo son de dos tipos:
- ***Flujos de trabajo del proceso***:
  - Conjunto de actividades fundamentalmente técnicas.
- ***Flujos de trabajo de soporte***:
  - Conjunto de actividades fundamentalmente de gestión.

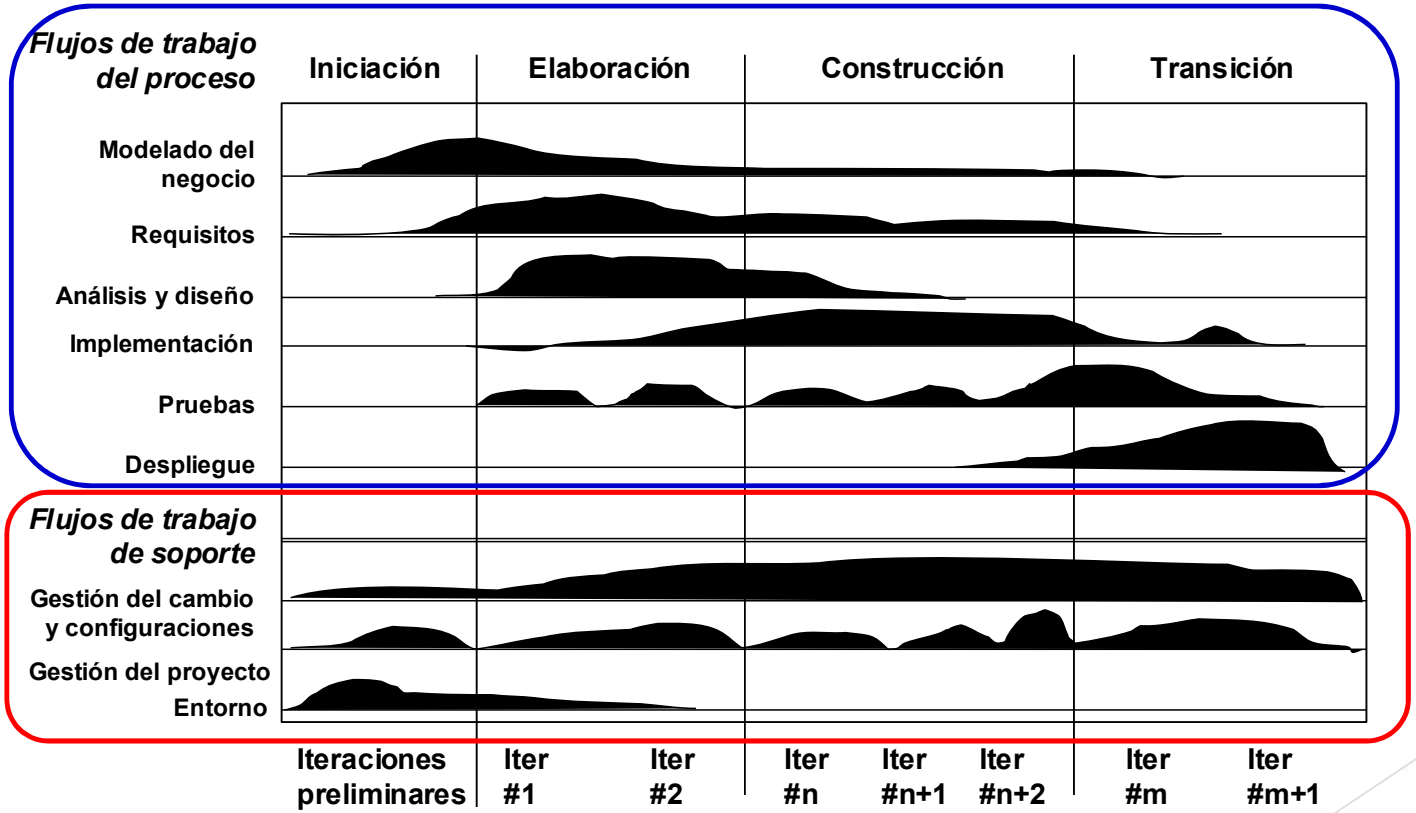
FLUJOS DE TRABAJO del proceso

1. ***Modelado del negocio***: describe la estructura y la dinámica de la organización.
2. ***Requisitos***: describe el método basado en casos de uso para extraer los requisitos.
3. ***Análisis y diseño***: describe las diferentes vistas arquitectónicas.
4. ***Implementación***: tiene en cuenta el desarrollo de software, la prueba de unidades y la integración.
5. ***Pruebas***: describe los casos de pruebas, los procedimientos y las métricas para evaluación de defectos.
6. ***Despliegue***: cubre la configuración del sistema entregable.

Flujos de trabajo de soporte:

1. ***Gestión de configuraciones***: controla los cambios y mantiene la integridad de los artefactos de un proyecto.
2. ***Gestión del Proyecto***: describe varias estrategias de trabajo en un proceso iterativo.
3. ***Entorno***: cubre la infraestructura necesaria para desarrollar un sistema.

El ciclo de vida del desarrollo del software: Flujos



TIPOS DE RESULTADOS

- Un modelo es una abstracción de la realidad o de un sistema real tomando los elementos más representativos con un propósito determinado.

- De un mismo sistema puede haber más de un modelo, porque, según el propósito del mismo, los elementos representativos pueden ser distintos.

- Los elementos a considerar en la construcción de modelos son: supuestos, simplificaciones, limitaciones o restricciones y preferencias

Tipos de resultados

- Los **supuestos:**
  - Son elementos para la construcción de modelos que reducen el número de permutaciones y variaciones posibles, permitiendo al modelo reflejar el problema de manera razonable.
- Las **simplificaciones:**
  - Son elementos para la construcción de modelos que permiten crear el modelo a tiempo.
- Las **limitaciones** o **restricciones:**
  - Son elementos para la construcción de modelos que ayudan a delimitar el problema.
- Las **preferencias:**
  - Son elementos para la construcción de modelos que indican la arquitectura preferida para toda la información, funciones y tecnología.
  - Pueden tener conflictos con otros factores restrictivos.
  - Es recomendable tenerlas en cuenta para obtener un resultado aceptado, además de correcto.

TIPOS DE RESULTADOS

- Un modelo de objetos o modelo orientado a objetos es una abstracción de un sistema informático orientado a objetos real que tiene un propósito determinado.
- Según el propósito final, el mismo sistema puede tener distintos modelos.
- Sin embargo, cualquiera de los modelos se construye con el mismo conjunto de elementos para representar las *propiedades estáticas* (estructura) y *dinámicas* (comportamiento) tanto del sistema como de las entidades que lo componen.

TIPOS DE RESULTADOS

- Cada actividad del Proceso Unificado lleva algunos artefactos asociados.
- Algunos artefactos:
  - Se utilizan como entradas directas en las actividades siguientes.
  - Se mantienen como recursos de referencia en el proyecto.
  - Se generan en algún formato específico, en forma de entregas definidas en el contrato.
- Estos artefactos son adicionales a los que proporciona el propio UML:
  - Los modelos y los conjuntos.

Tipos de resultados

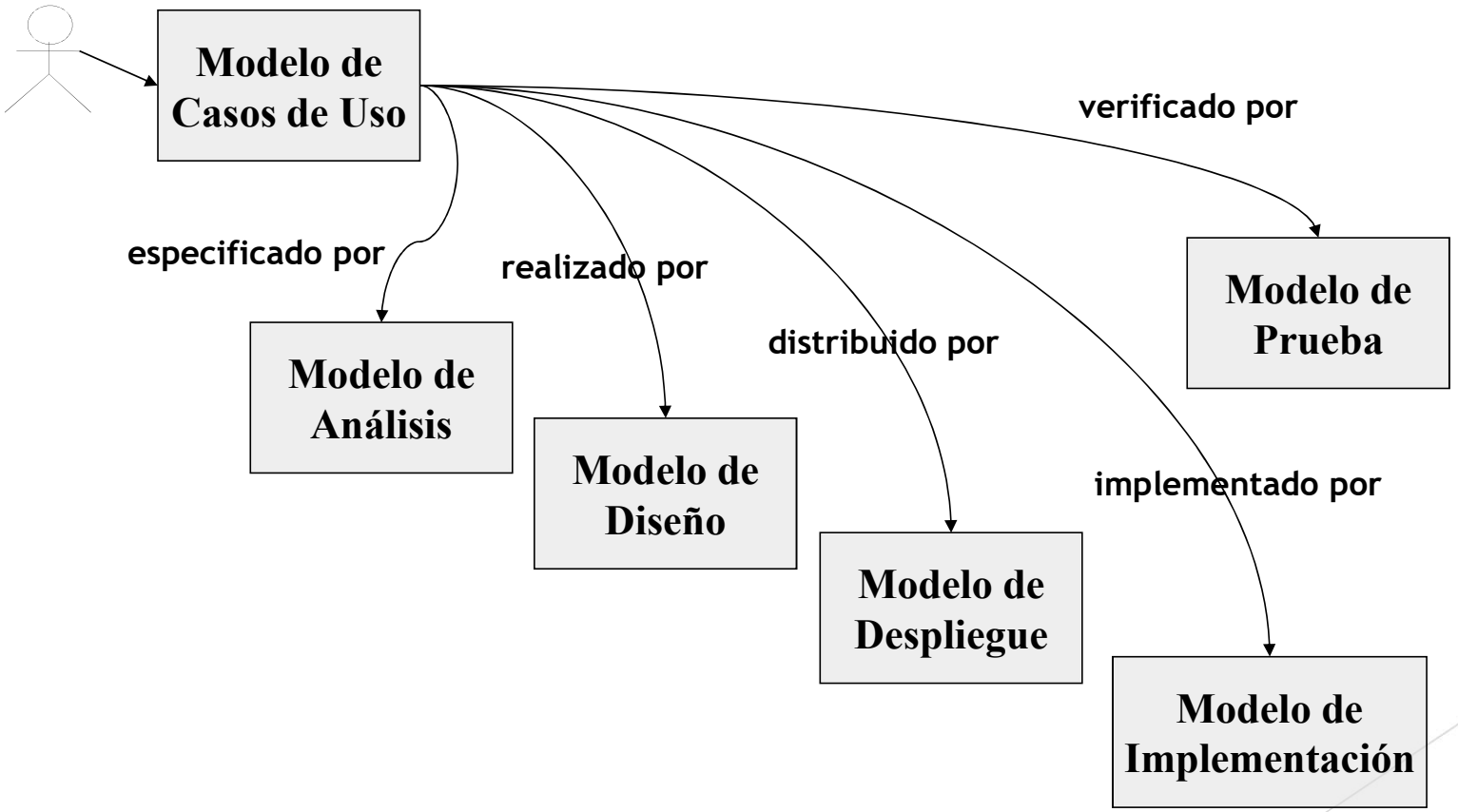
- Los modelos son el tipo de artefacto más importante en el Proceso Unificado.
- Constituyen el tercer eje del metamodelo 3-D:
  - Los tipos de resultados obtenidos con los distintos esfuerzos a lo largo de las fases del ciclo.
- Hay **nueve** modelos que en conjunto cubren todas las decisiones importantes implicadas en la visualización, especificación, construcción y documentación de un sistema con gran cantidad de software.

TIPOS DE RESULTADOS

Modelos del Proceso Unificado:

1. ***Modelo del negocio***: establece una abstracción de la organización.
2. ***Modelo del dominio***: establece el contexto del sistema.
3. ***Modelo de casos de uso***: establece los requisitos funcionales del sistema.
4. ***Modelo de análisis*** (opcional): establece un diseño de las ideas.
5. ***Modelo de diseño***: establece el vocabulario del problema y su solución.
6. ***Modelo del proceso*** (opcional): establece los mecanismos de concurrencia y sincronización del sistema.
7. ***Modelo de despliegue***: establece la topología hardware sobre la cual se ejecutará el sistema.
8. ***Modelo de implementación***: establece las partes que se utilizarán para ensamblar y hacer disponible el sistema físico.
9. ***Modelo de pruebas***: establece las formas de validar y verificar el sistema.

Relaciones lógicas entre los modelos :



Modelos y flujos de trabajo del Proceso Unificado

	Modelado del Negocio	Requisitos	Análisis	Diseño	Implementa-ción	Prueba	Despliegue
Modelo del Negocio	X						
Modelo del Dominio	X	X					
Modelo de Casos de Uso		X					
Modelo de Análisis			X				
Modelo de Diseño				X			
Modelo de Procesos				X			
Modelo de Despliegue				X			X
Modelo de Implementación					X		X
Modelo de Prueba						X	X

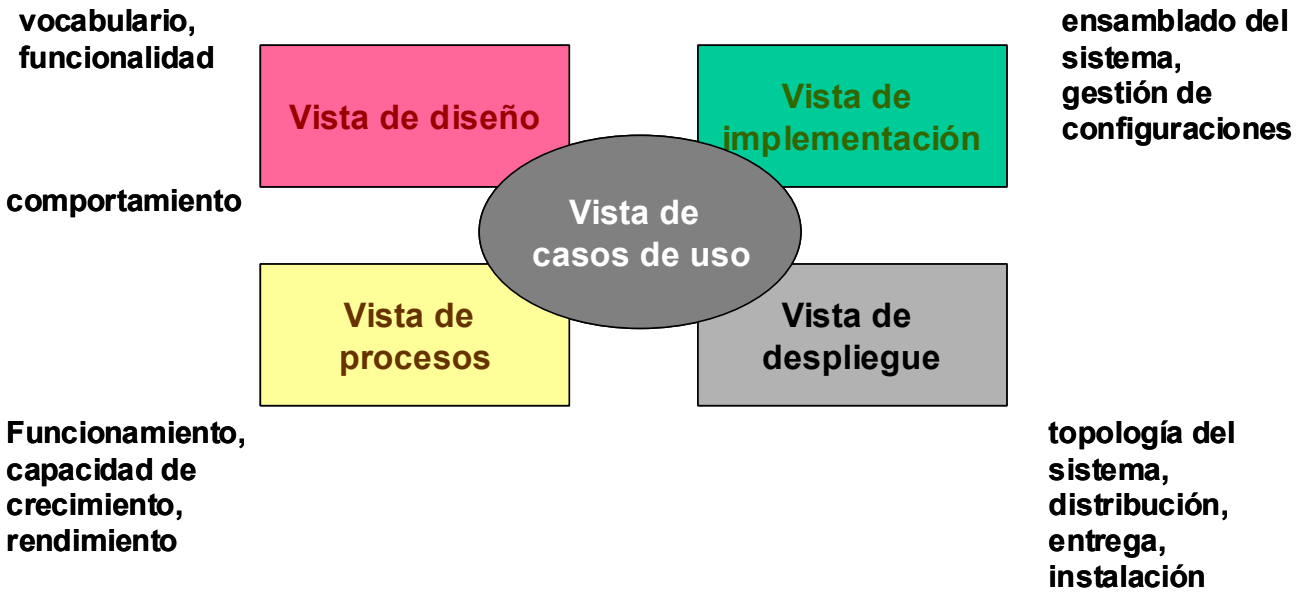
MODELOS Y DIAGRAMAS EN EL RUP

	Modelo del Negocio		Modelo del Dominio		Modelo de Casos de Uso		Modelo de Análisis		Modelo de Diseño		Modelo de Procesos		Modelo de Despliegue		Modelo Implemen-tación		Modelo de Prueba	
	Est.	Din.	Est.	Din.	Est.	Din.	Est.	Din.	Est.	Din.	Est.	Din.	Est.	Din.	Est.	Din.	Est.	Din.
Diagrama de Casos de Uso	X				X												X	
Diagrama de Interacción-Secuencia		X		X		X		X		X		X				X		X
Diagrama de Interacción-Colaboración		X		X		X		X		X		X				X		X
Diagrama de Clases de Análisis							X											
Diagrama de Objetos de Análisis							X											
Diagrama de Clases de Diseño									X		X							
Diagrama de Objetos de Diseño									X		X							
Diagrama de Estados						X		X		X		X		X		X		
Diagrama de Actividades								X		X		X		X		X		
Diagrama de Componentes															X			
Diagrama de Despliegue													X					

TIPOS DE RESULTADOS

- El Proceso Unificado recupera el concepto de vista de UML.
- Para el Proceso Unificado una **vista** es:
  - Una proyección de un modelo.
  - Una proyección de la organización y la estructura del sistema que se centra en un aspecto particular del sistema.
- La arquitectura de un sistema se captura en forma de cinco vistas que interactúan entre sí:
  - La vista de casos de uso.
  - La vista de diseño.
  - La vista de procesos.
  - La vista de despliegue.
  - La vista de implementación.

Vistas de la arquitectura de un sistema



TIPOS DE RESULTADOS

- Cada una de las vistas presenta:
  - **Aspectos estáticos**: mediante los diagramas estructurales de UML.
  - **Aspectos dinámicos**: mediante diagramas dinámicos de UML.
- Ejemplo: se puede trabajar con la vista de casos de uso estática y la vista de casos de uso dinámica, la vista de diseño estática y la vista de diseño dinámica, y así sucesivamente.
- En el RUP se da más importancia a los modelos que a las vistas. Aunque se siguen manteniendo para determinados propósitos de modelado.

TIPOS DE RESULTADOS

Nombre	Descripción	Aspectos Estáticos	Aspectos Dinámicos
Vista de casos de uso	Proyecta el comportamiento del sistema tal y como es percibido por los: usuarios finales, analistas y encargados de las pruebas. Especifica las fuerzas que configuran la arquitectura del sistema.	Diagramas de casos de uso	Diagramas de interacción Diagramas de estados
Vista de diseño	Soporta los requisitos funcionales del sistema: servicios proporcionados a los usuarios finales. Vocabulario del problema y su solución: clases, interfaces y colaboraciones.	Diagramas de clases Diagramas de objetos	Diagramas de interacción Diagramas de estados Diagramas de actividades
Vista de procesos	Cubre el funcionamiento, capacidad de crecimiento y rendimiento del sistema. Mecanismos de sincronización y concurrencia del sistema: hilos y procesos.	Diagramas de clases (activas) Diagramas de objetos	Diagramas de interacción Diagramas de estados Diagramas de actividades
Vista de implementación	Cubre la gestión de configuraciones de las distintas versiones de un sistema a partir de componentes y archivos quasi-independientes. Ensamblado y disponibilidad del sistema: componentes y archivos.	Diagramas de componentes	Diagramas de interacción Diagramas de estados Diagramas de actividades
Vista de despliegue	Contiene los nodos que forman la arquitectura (topología) hardware sobre la que se ejecuta el sistema a través de sus componentes. Está destinada a representar la distribución, entrega e instalación de las partes que forman el sistema informático físico.	Diagramas de despliegue	Diagramas de interacción Diagramas de estados Diagramas de actividades

VISTAS Y DIAGRAMAS EN UML

		Diagrama de Casos de Uso	Diagrama de Interacción-Secuencia	Diagrama de Interacción-Colaboración	Diagrama de Clases	Diagrama de Objetos	Diagrama de Estados	Diagrama de Actividades	Diagrama de Componentes	Diagrama de Despliegue
Vista de Casos de Uso	Estática									
	Dinámica									
Vista de Diseño	Estática									
	Dinámica									
Vista de Procesos	Estática									
	Dinámica									
Vista de Implementación	Estática									
	Dinámica									
Vista de Despliegue	Estática									
	Dinámica									

TIPOS DE RESULTADOS

- Los artefactos **conjunto** del RUP son los siguientes:
  1. Conjunto de requisitos.
  2. Conjunto de diseño.
  3. Conjunto de implementación.
  4. Conjunto de despliegue.

TIPOS DE RESULTADOS

1. Conjunto de requisitos:
   - Agrupa toda la información que describe lo que debe hacer el sistema.
   - Puede comprender un modelo de casos de uso, un modelo de requisitos no funcionales, un modelo del dominio, un modelo de análisis y otras formas de expresión de las necesidades del usuario, incluyendo pero no limitándose a maquetas, prototipos de la interfaz, restricciones legales, etc.

TIPOS DE RESULTADOS

2. Conjunto de diseño:
   - Agrupa información que describe cómo se va a construir el sistema y captura las decisiones acerca de cómo se va realizar, teniendo en cuenta las restricciones de tiempo, presupuesto, aplicaciones existentes, reutilización, objetivos de calidad y demás consideraciones.
   - Puede implicar un modelo de diseño, un modelo de pruebas y otras formas de expresión de la naturaleza del sistema, incluyendo, pero no limitándose, a prototipos y arquitecturas ejecutables.

TIPOS DE RESULTADOS

3. Conjunto de implementación:
   - Agrupa toda la información acerca de los elementos software que comprende el sistema, incluyendo, pero no limitándose, a código fuente en varios lenguajes de programación, archivos de configuración, archivos de datos, componentes software, etc., junto con la información que describe cómo ensamblar el sistema.

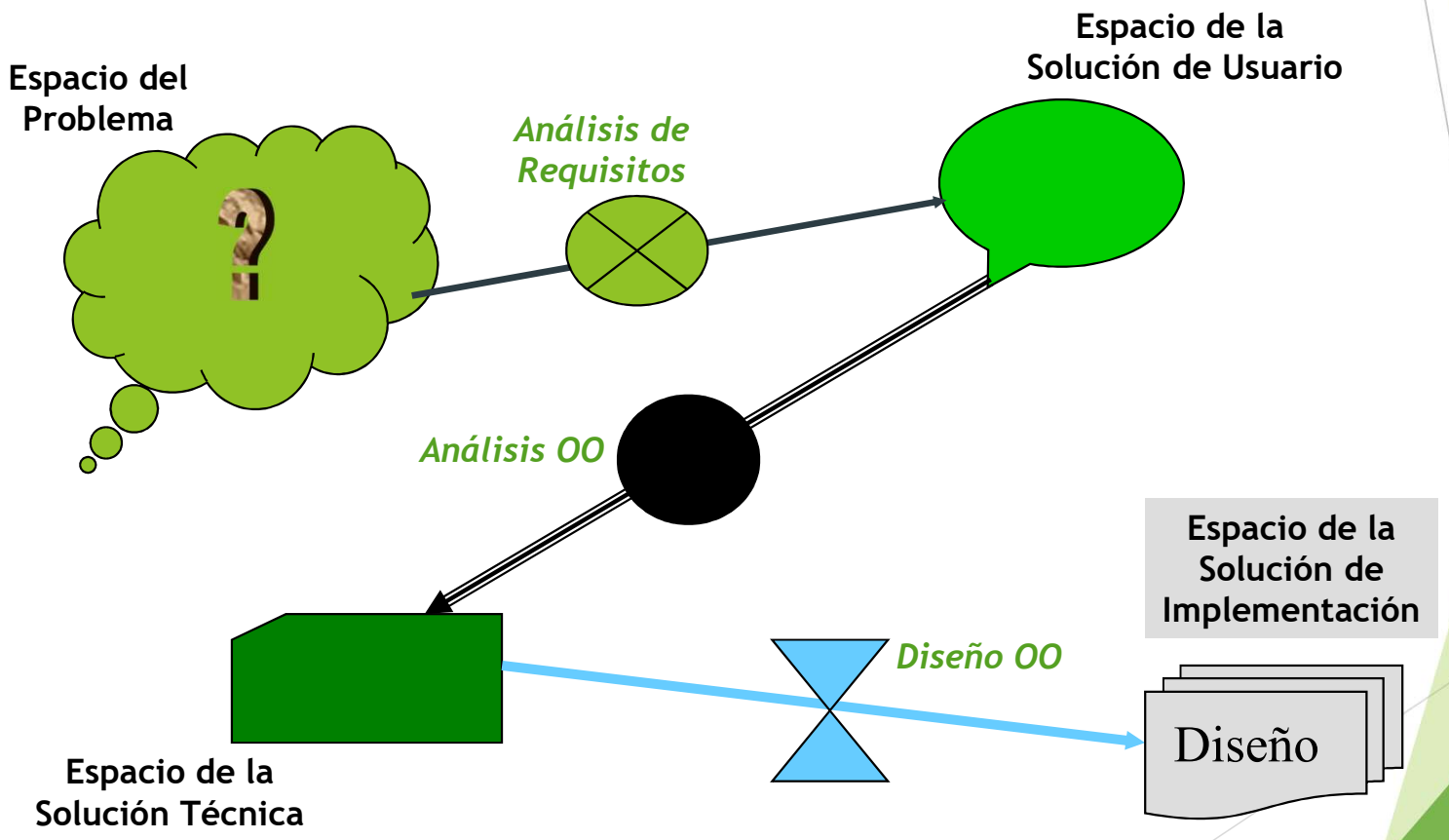
TIPOS DE RESULTADOS

4. Conjunto de despliegue:
   - Agrupa toda la información acerca de la forma en que se empaqueta actualmente el software, se distribuye, se instala y se ejecuta en el entorno destino.

CAPTURA Y MODELADO DE REQUISITOS

- El Análisis de Requisitos tiene por misión convertir el problema, expresado en términos del dominio del negocio, a soluciones descritas en en lenguaje del dominio de la Tecnología de Información.
- El problema y su planteamiento pertenecen al Espacio del Problema:
  - Descripción concreta del negocio.
  - Dominio de los Objetos de Negocio (DON).
- Las soluciones pertenecen al Espacio de la Solución:
  - Descripción concreta del sistema de información.
  - Dominio de los Objetos de Negocio.
  - Dominio de los Objetos de Infraestructura (DOI):
    - Subdominio de Objetos de Bases de Datos (SDOBD).
    - Subdominio de Objetos de Interfaz (SDOIZ).

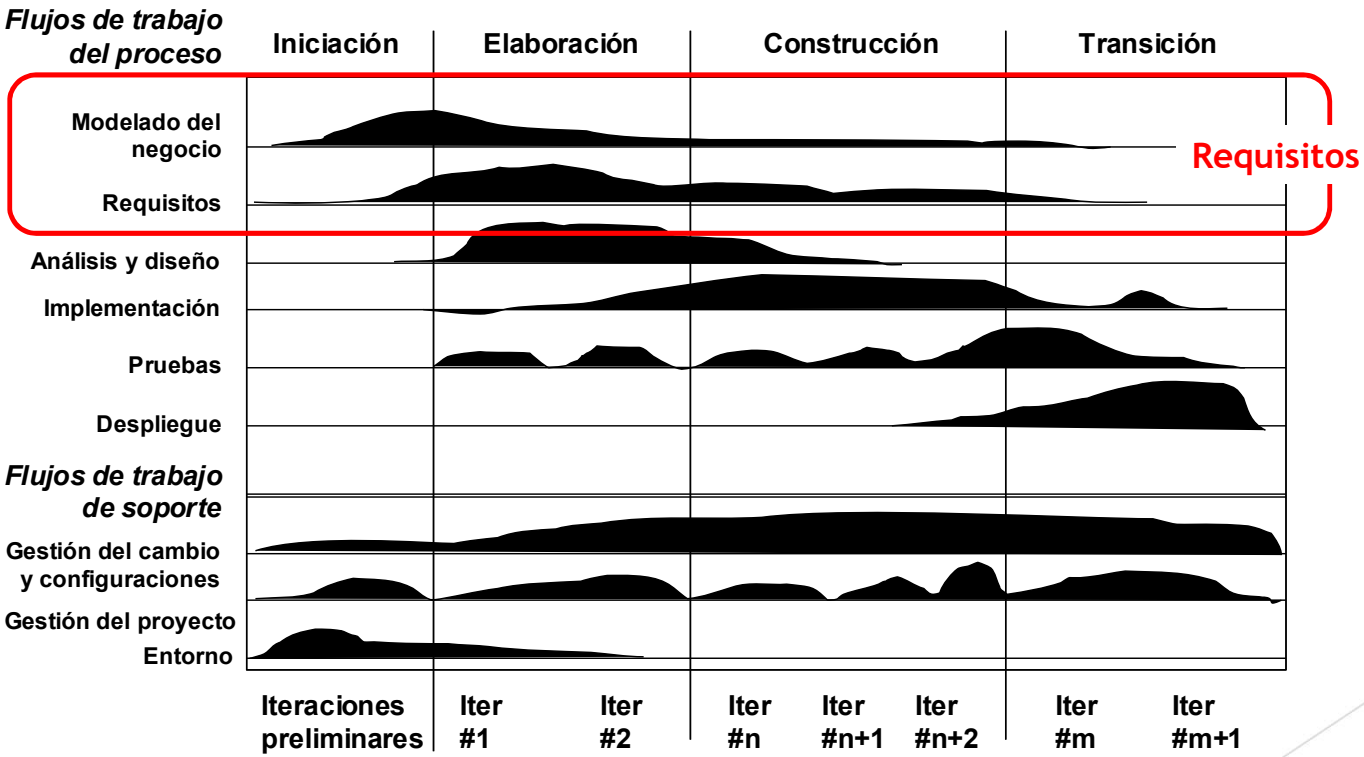
Captura y Modelado de Requisitos



Captura y Modelado de Requisitos

- El Análisis de Requisitos en el RUP se realiza por medio de los flujos de trabajo:
  - Modelado del negocio.
  - Requisitos.
- El resultado del Análisis de Requisitos es el siguiente:
  - Modelo del Negocio.
  - Modelo del Dominio.
  - Modelo de Casos de Uso.
  - Documento de Especificaciones Técnicas del Sistema (según norma IEEE-830/1999).

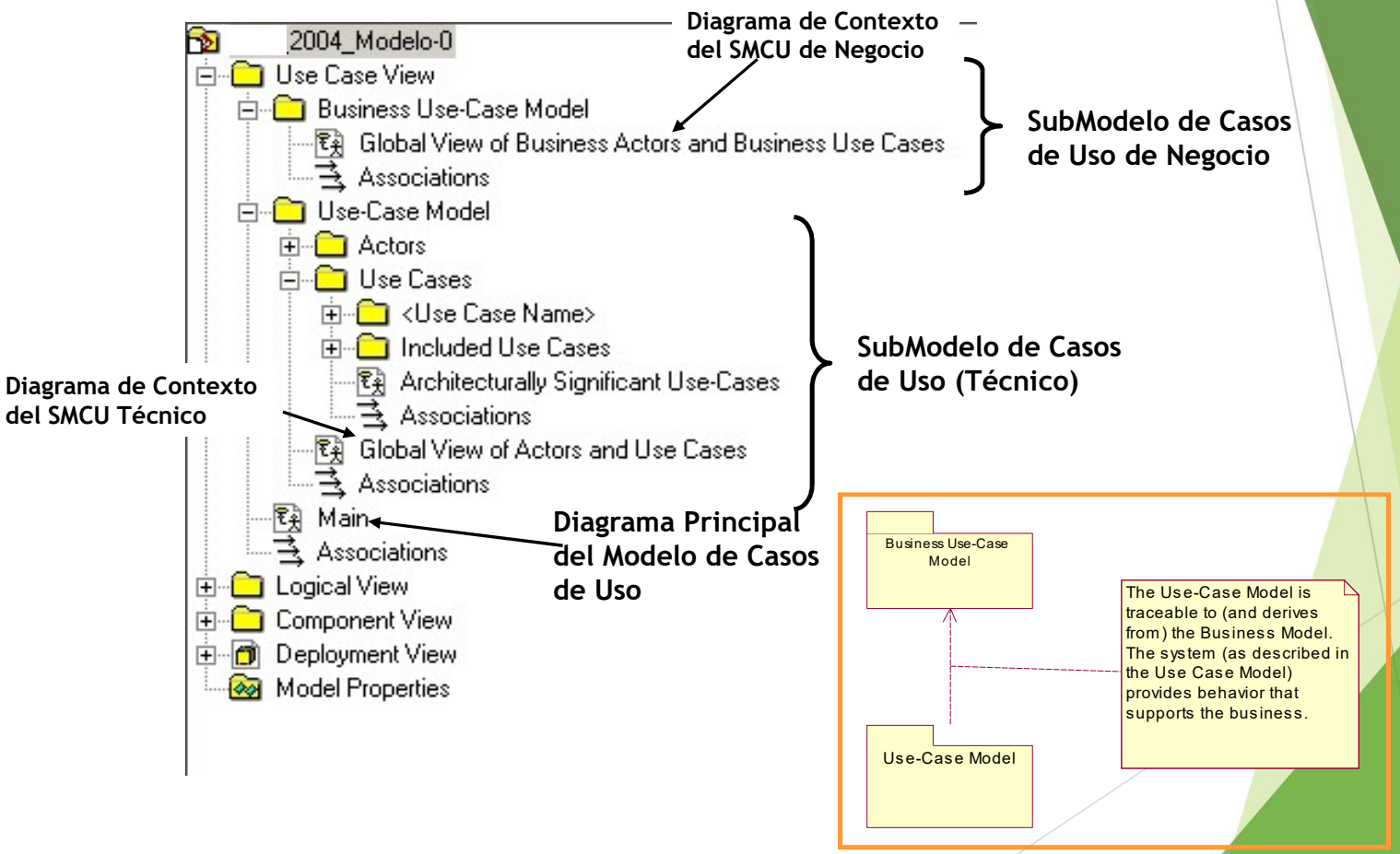
Captura y Modelado de Requisitos



CAPTURA Y MODELADO DE REQUISITOS

- El Modelo de Casos de Uso (MCU) establece los requisitos funcionales del sistema de información.
- En el MCU se recoge la descripción externa y observable de cómo se utiliza el sistema de información:
  - Descripción de CÓMO se utiliza el sistema:
    - Funciones, Servicios y Procesos.
  - Descripción EXTERNA del uso del sistema:
    - Se identifican y describen funciones/servicios/procesos del negocio que un usuario puede hacer con el soporte del sistema de información.
  - Descripción OBSERVABLE del uso del sistema:
    - Es como si hubiera un observador externo que va anotando lo que hace el usuario con el sistema y lo que el sistema responde al usuario.

CAPTURA Y MODELADO DE REQUISITOS



Captura y Modelado de Requisitos

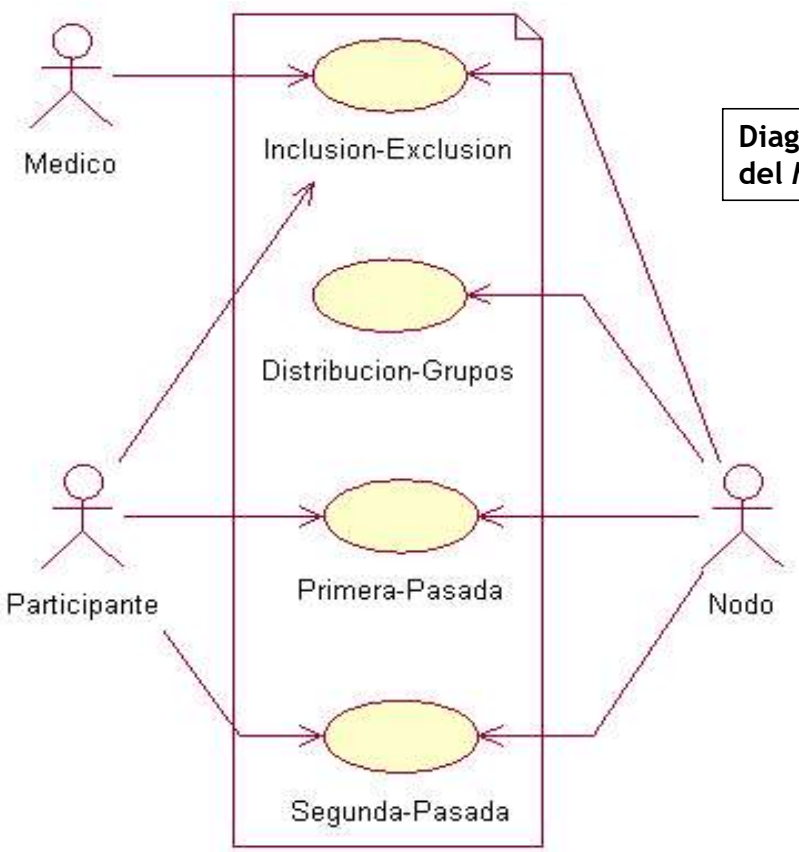
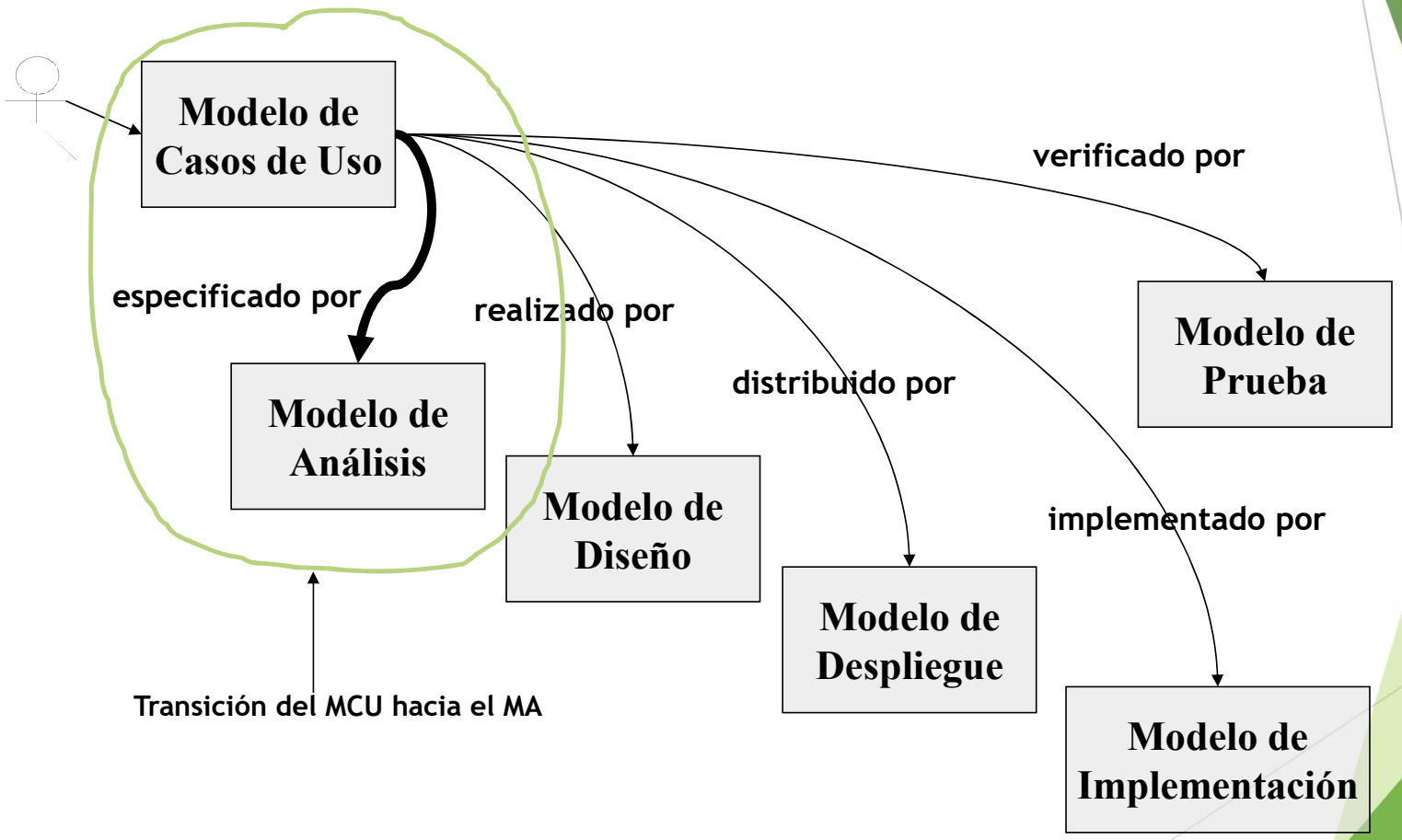


Diagrama de Contexto del MCU

Modelado de Análisis

- Una vez completado el modelo de casos de uso (CU) se ha llegado a obtener diagramas de casos de uso en determinados niveles que ya no se pueden explotar más.
- Si se intentara explotar los CU, se pasaría a describir el comportamiento interno de las funciones con artefactos inadecuados.
- Los casos de uso contenidos en estos diagramas se denominan casos de uso elementales.
- Esta situación límite indica que se debe pasar a trabajar con otros artefactos, que son los del modelo de análisis:
  - Clases de análisis.
  - Asociaciones.
  - Diagramas de clases.
  - Diagramas de colaboración asociados a los diagramas de clases.

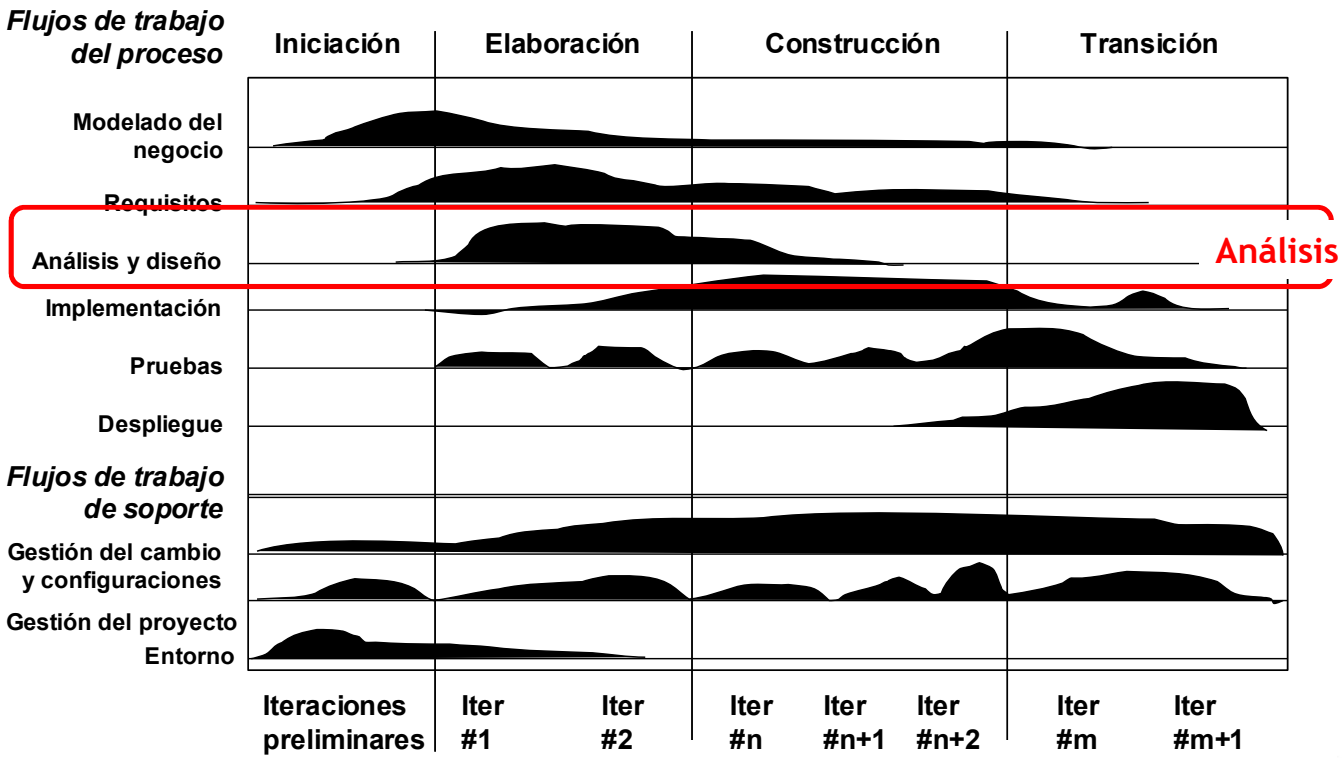
Modelado de Análisis



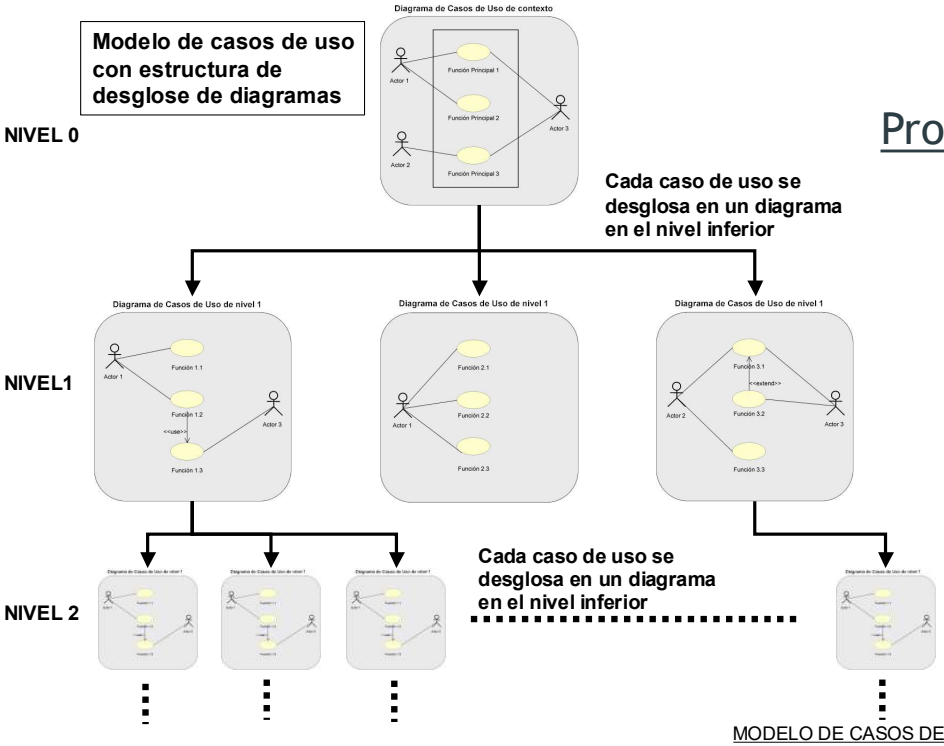
Modelado de Análisis

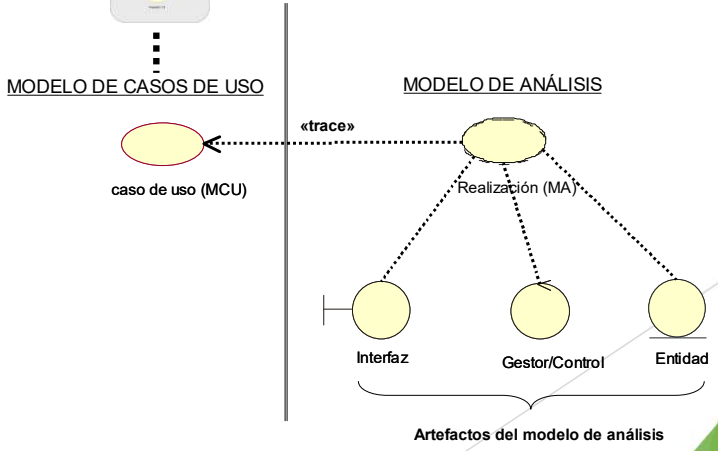
- El Análisis en el RUP se realiza por medio de los flujos de trabajo:
  - Análisis y diseño.
- El resultado del Análisis es el siguiente:
  - Modelo de Análisis.
- El Modelo de Análisis contiene:
  - La Vista de Diseño de UML.
  - La Vista de Procesos de UML.

Modelado de Análisis

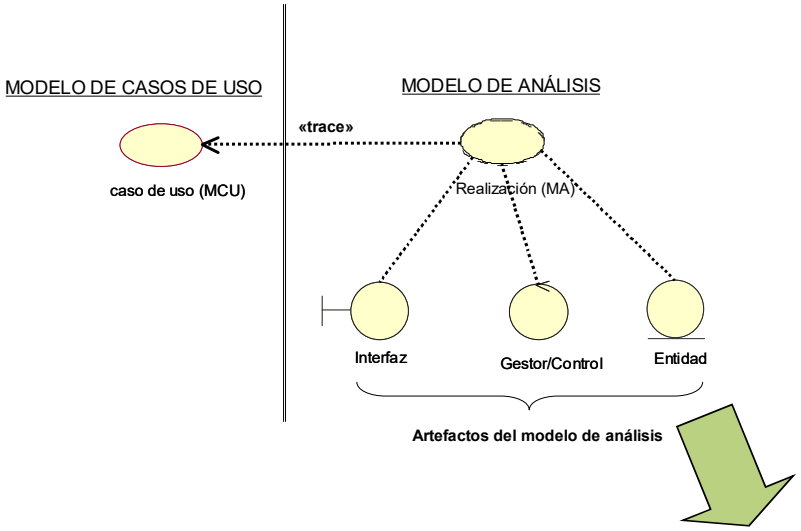


Proceso de Conversión: Casos de Uso → Análisis



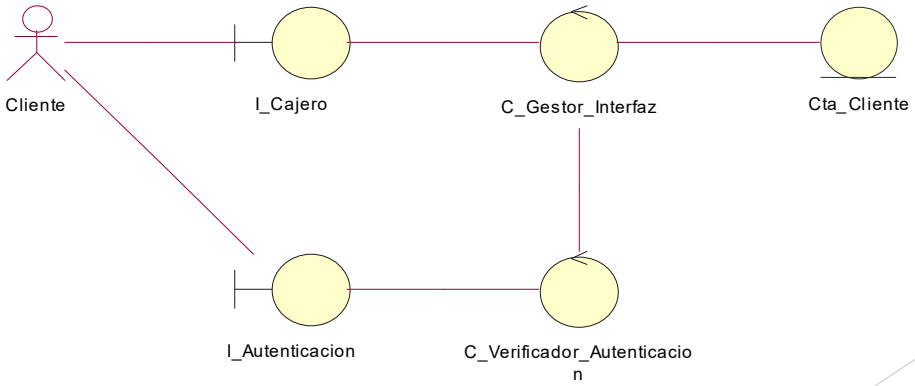


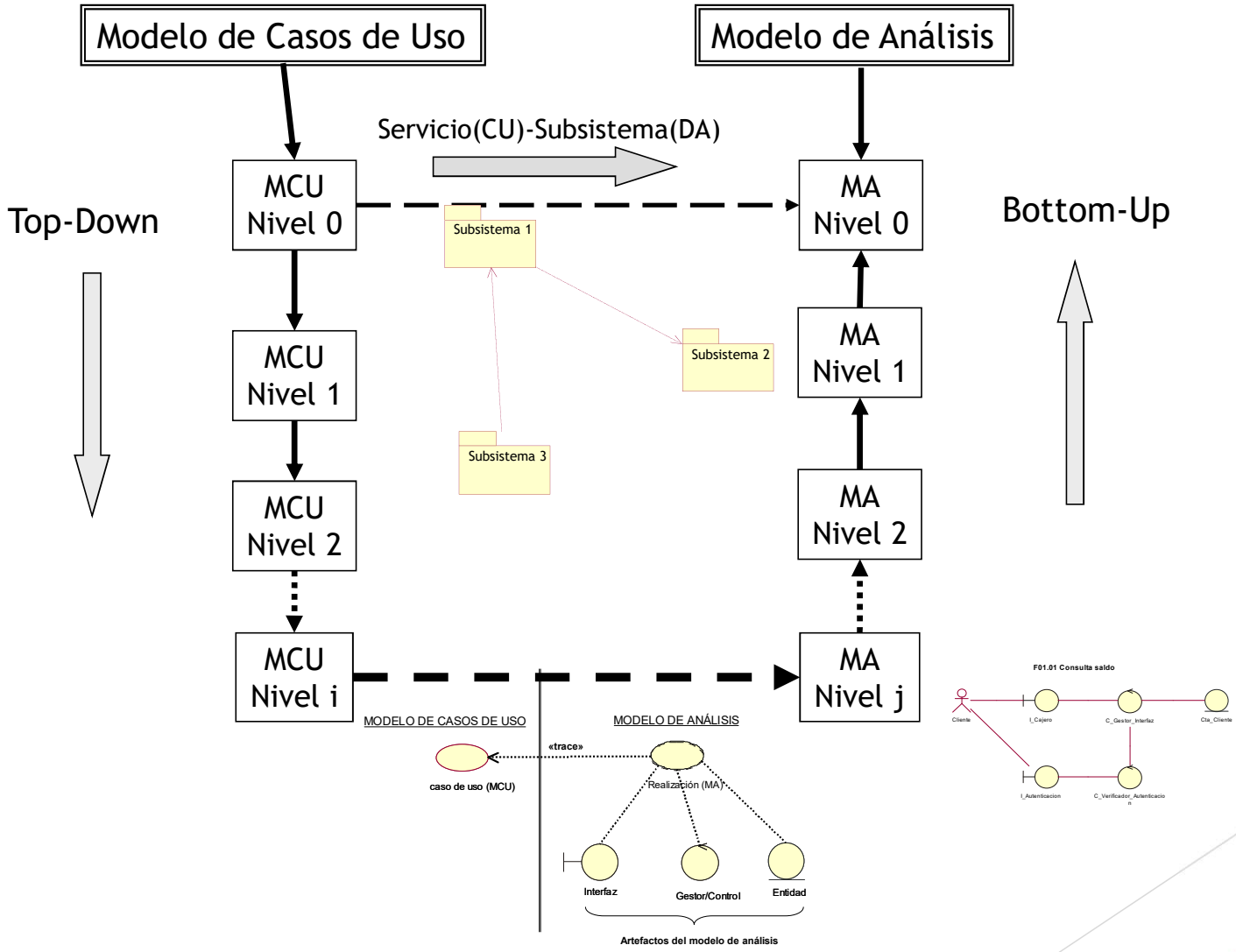
Proceso de Conversión: Casos de Uso → Análisis



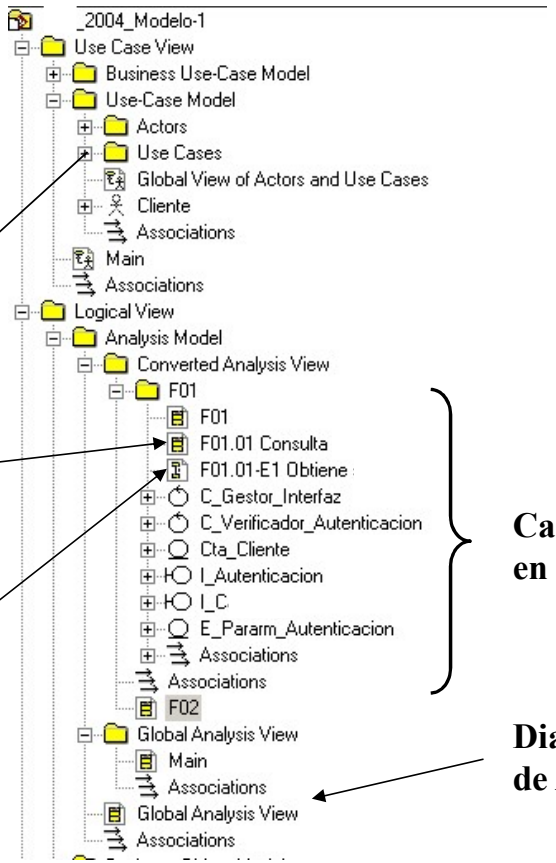
F01.01 Consulta

Diagrama de Clases de Análisis Atómico




Modelo de Casos de Uso
Modelo de Análisis
Servicio(CU)-Subsistema(DA)
Top-Down
Bottom-Up
MCU Nivel 0
MCU Nivel 1
MCU Nivel 2
MCU Nivel i
MA Nivel 0
MA Nivel 1
MA Nivel 2
MA Nivel j
Subsistema 1
Subsistema 2
Subsistema 3
MODELO DE CASOS DE USO
MODELO DE ANÁLISIS
«trace»
caso de uso (MCU)
Realización (MA)
Interfaz
Gestor/Control
Entidad
Artefactos del modelo de análisis
F01.01 Consulta saldo

La estructura del modelo en Rose:



D. Clases Análisis Atómico para el Caso de Uso F01.01 <*Nombre función*>

Diagrama de Colaboración para DCAA F01.01

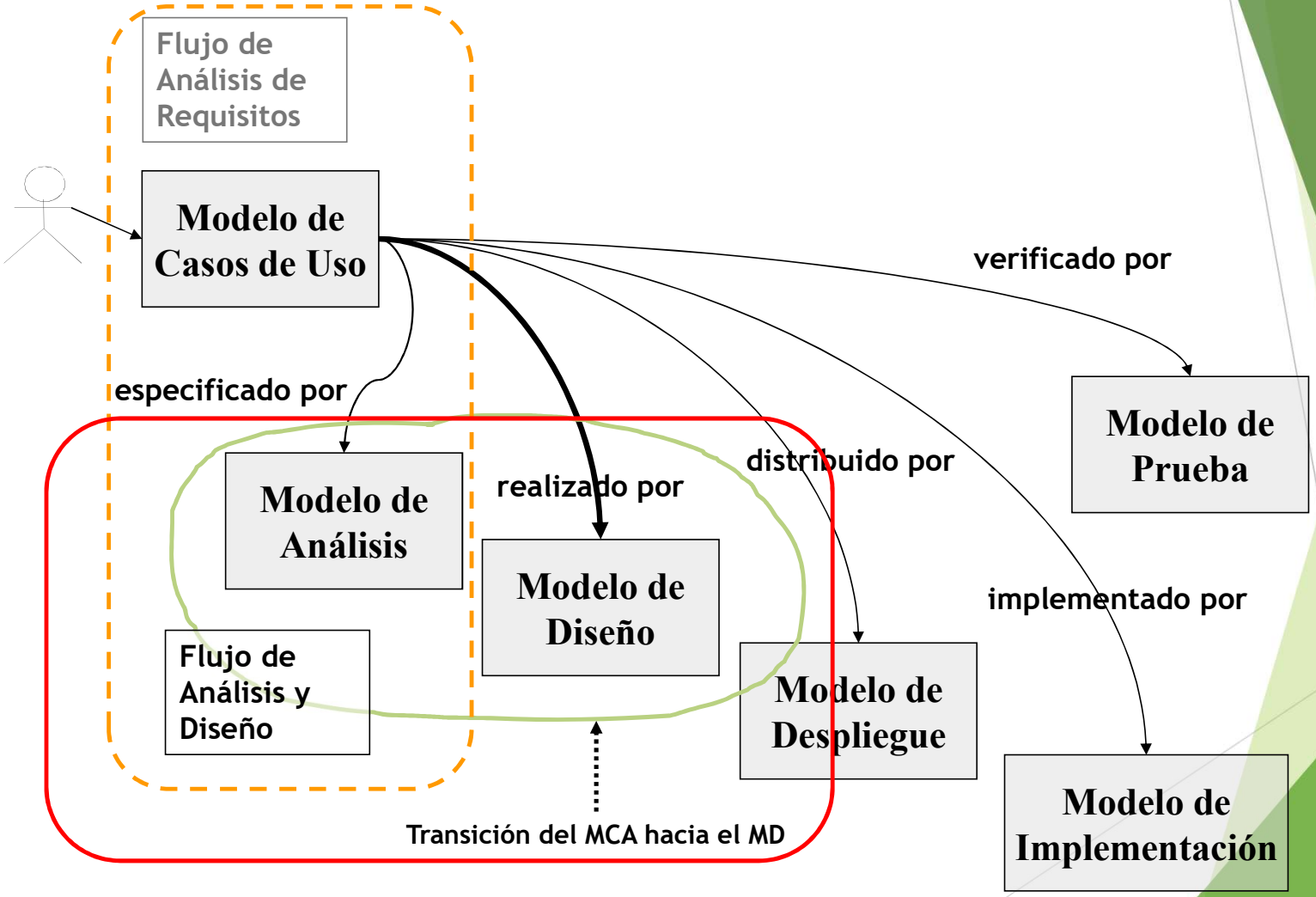
Carpeta de trabajo en la conversión

Diagrama de Clases de Análisis de Contexto

Modelado de Diseño

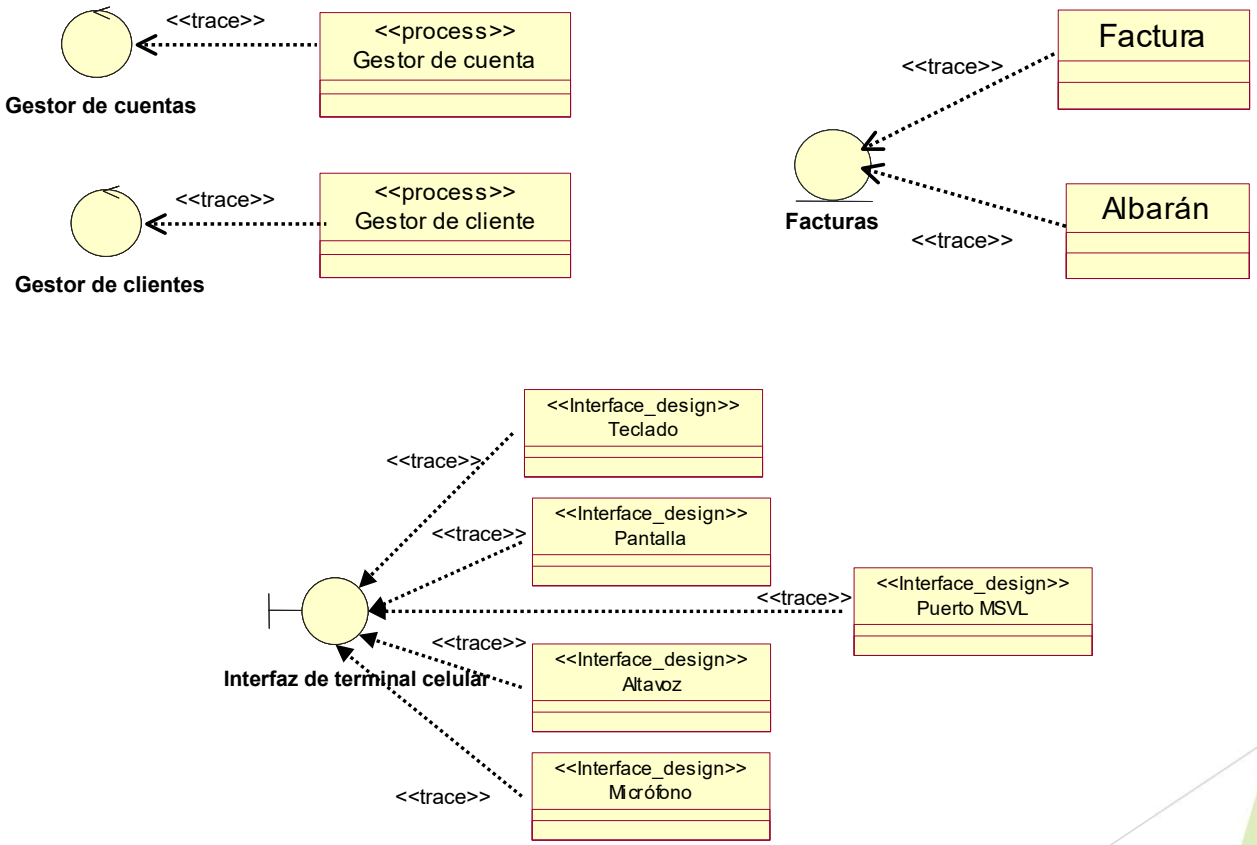
- En el flujo de requisitos se construye un modelo que representa el comportamiento observable o externo del sistema que se quiere obtener.
- En los flujos de análisis, **diseño** e implementación, se representa la estructura y el comportamiento internos del sistema a realizar.
- Característica común de los tres flujos frente al flujo de requisitos:
  - En los tres flujos se trabaja a diferentes niveles de abstracción, desde el más elevado en el análisis, hasta el más bajo en la implementación.

Modelado de Diseño



Modelado de Diseño

- La técnica de modelado consiste en identificar, a través de las especificaciones de las clases de análisis las clases de diseño correspondientes.
- Para cada clase de análisis se puede derivar **una** o **más** clases de diseño:
  - *Clase de control* ➔ clase activa (>= 1)
  - *Clase de entidad* ➔ clase de entidad (>= 1)
  - *Clase de interfaz* ➔ clase de interfaz (>= 1)


<<trace>>
<<process>>
Gestor de cuenta
Gestor de cuentas
<<trace>>
<<process>>
Gestor de cliente
Gestor de clientes
Factura
<<trace>>
Facturas
Albarán
<<trace>>
<<Interface_design>>
Teclado
<<trace>>
<<Interface_design>>
Pantalla
<<trace>>
<<Interface_design>>
Puerto MSVL
<<trace>>
<<Interface_design>>
Altavoz
Interfaz de terminal celular
<<trace>>
<<Interface_design>>
Micrófono
<<trace>>

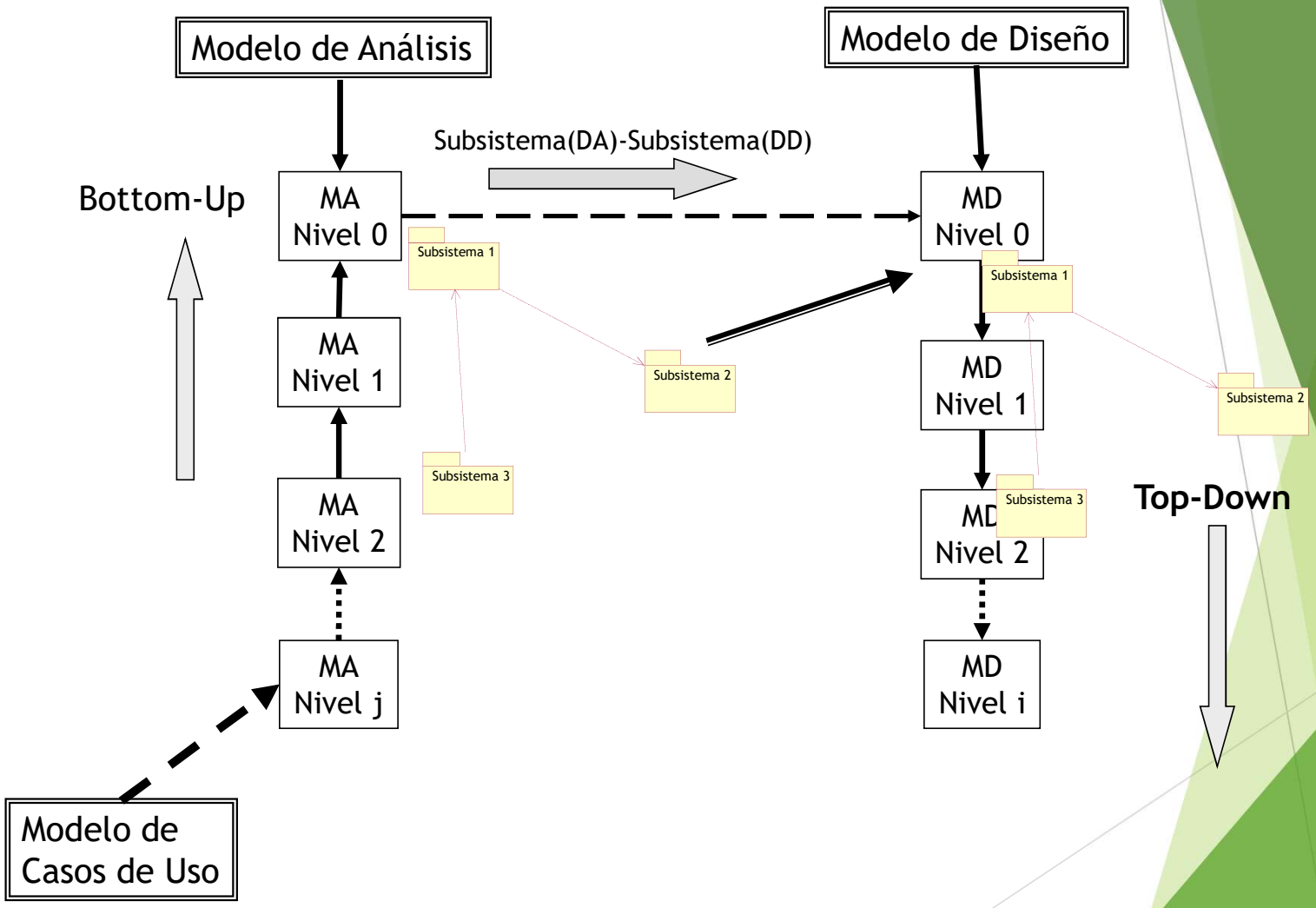
Modelado de Diseño

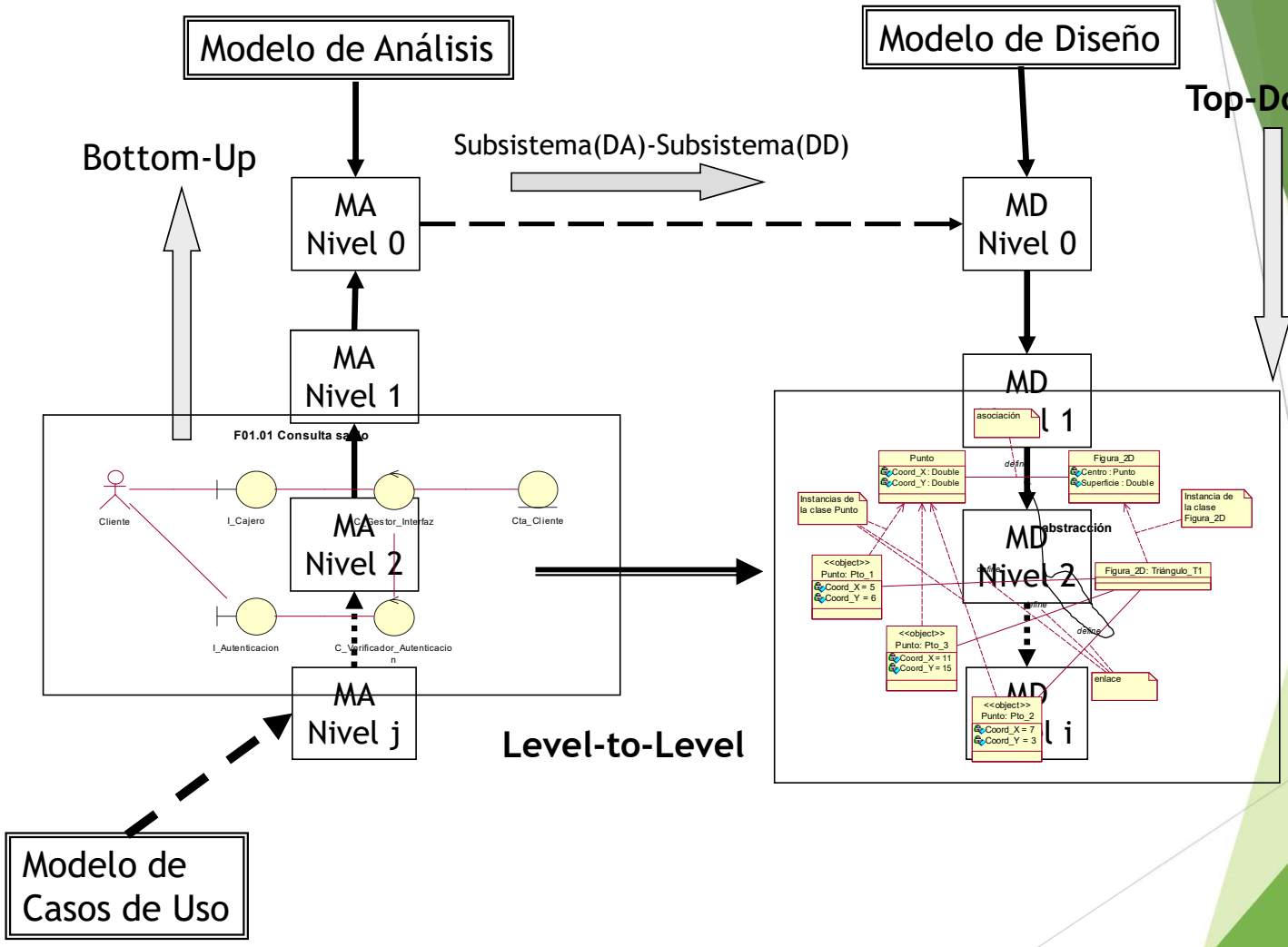
- En el proceso de conversión del Modelo de Análisis (MA) al Modelo de Diseño (MD), la estrategia adoptada es mixta:

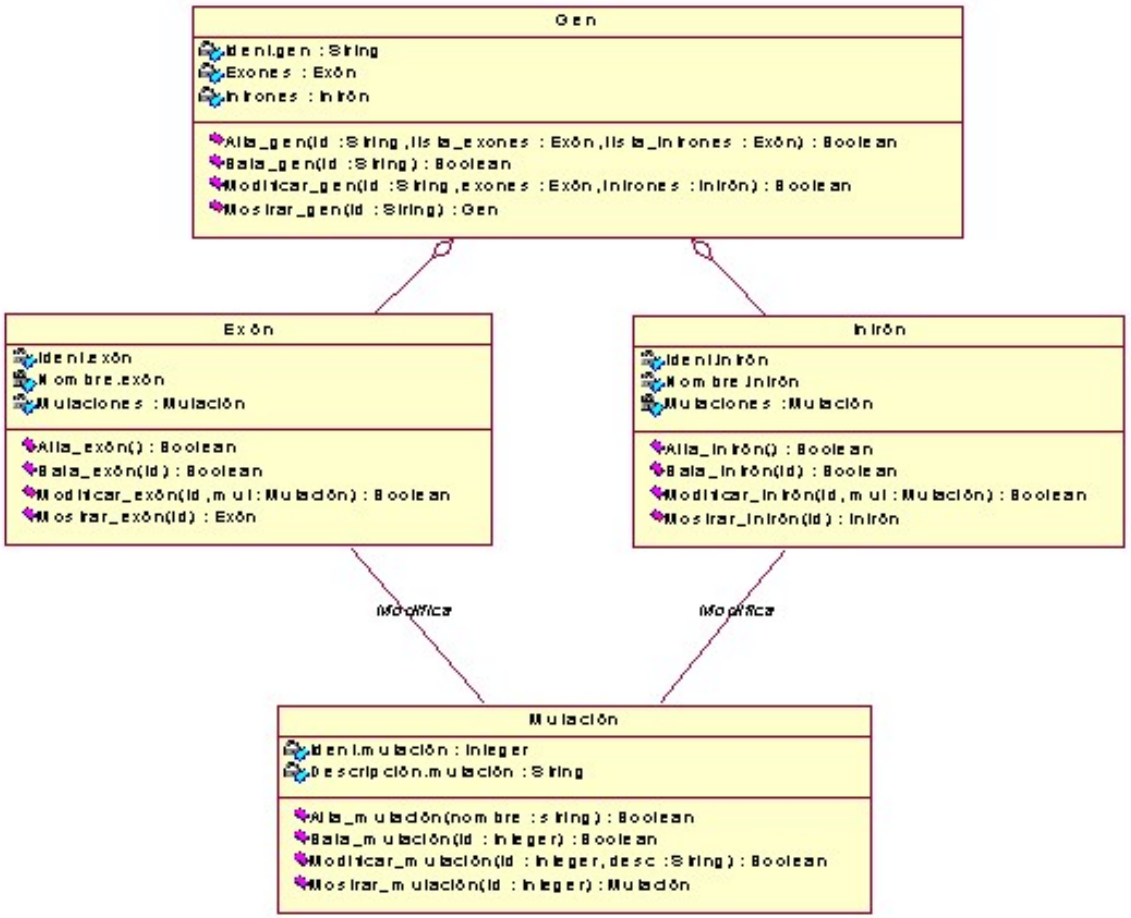
Top-Down

+

Level-to-Level


Modelo de Análisis
Modelo de Diseño
Subsistema(DA)-Subsistema(DD)
Bottom-Up
MA
Nivel 0
MA
Nivel 1
MA
Nivel 2
MA
Nivel j
MD
Nivel 0
MD
Nivel 1
MD
Nivel 2
MD
Nivel i
Subsistema 1
Subsistema 2
Subsistema 3
Subsistema 1
Subsistema 2
Subsistema 3
Top-Down
Modelo de
Casos de Uso


Modelo de Análisis
Modelo de Diseño
Top-Down
Bottom-Up
Subsistema(DA)-Subsistema(DD)
MA
Nivel 0
MD
Nivel 0
MA
Nivel 1
MD
l 1
F01.01 Consulta saldo
Cliente
I_Cajero
C_Gestor_Interfaz
Cta_Cliente
MA
Nivel 2
I_Autenticacion
C_Verificador_Autenticacio
n
MA
Nivel j
Level-to-Level
Punto
Coord_X : Double
Coord_Y : Double
Figura_2D
Centro : Punto
Superficie : Double
asociación
Instancias de
la clase Punto
Instancia de
la clase
Figura_2D
abstracción
MD
Nivel 2
<<object>>
Punto: Pto_1
Coord_X = 5
Coord_Y = 6
Figura_2D: Triángulo_T1
<<object>>
Punto: Pto_3
Coord_X = 11
Coord_Y = 15
enlace
<<object>>
Punto: Pto_2
Coord_X = 7
Coord_Y = 3
l i
Modelo de
Casos de Uso


Gen
Idenigen : String
Exones : Exón
Inirones : Inirón
Alta_gen(id :String,lista_exones : Exón,lista_inirones : Exón) : Boolean
Baja_gen(id :String) : Boolean
Modificar_gen(id :String,exones : Exón,inirones : Inirón) : Boolean
Mostrar_gen(id : String) : Gen
Exón
Idenlexón
Nombre.exón
Mutaciones : Mutación
Alta_exón() : Boolean
Baja_exón(id) : Boolean
Modificar_exón(id,mut : Mutación) : Boolean
Mostrar_exón(id) : Exón
Inirón
Idenlinirón
Nombre.inirón
Mutaciones :Mutación
Alta_inirón() : Boolean
Baja_inirón(id) : Boolean
Modificar_inirón(id,mut : Mutación) : Boolean
Mostrar_inirón(id) : Inirón
Modifica
Modifica
Mutación
Idenl.mutación : Integer
Descripción.mutación : String
Alta_mutación(nombre : string) : Boolean
Baja_mutación(id : Integer) : Boolean
Modificar_mutación(id : Integer,desc :String) : Boolean
Mostrar_mutación(id : Integer) : Mutación

La estructura del modelo en Rose:

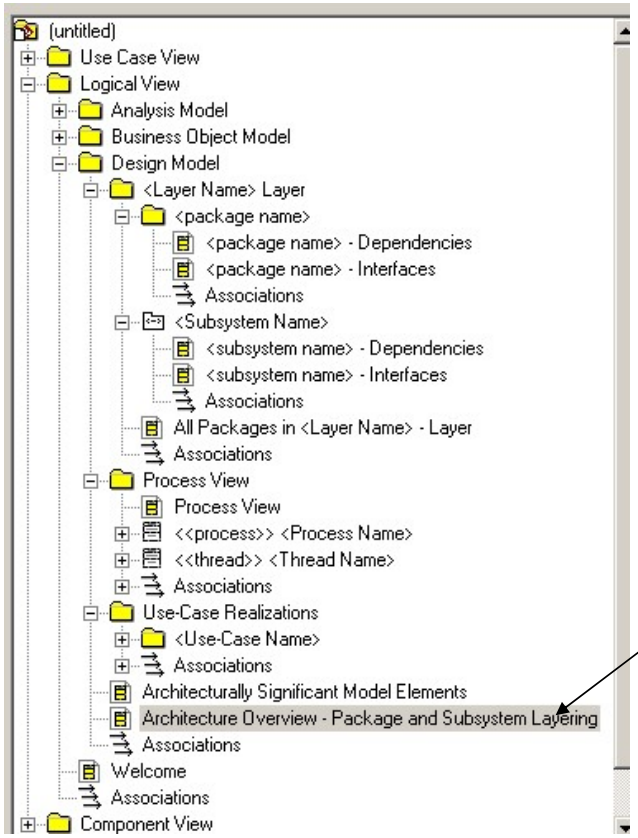
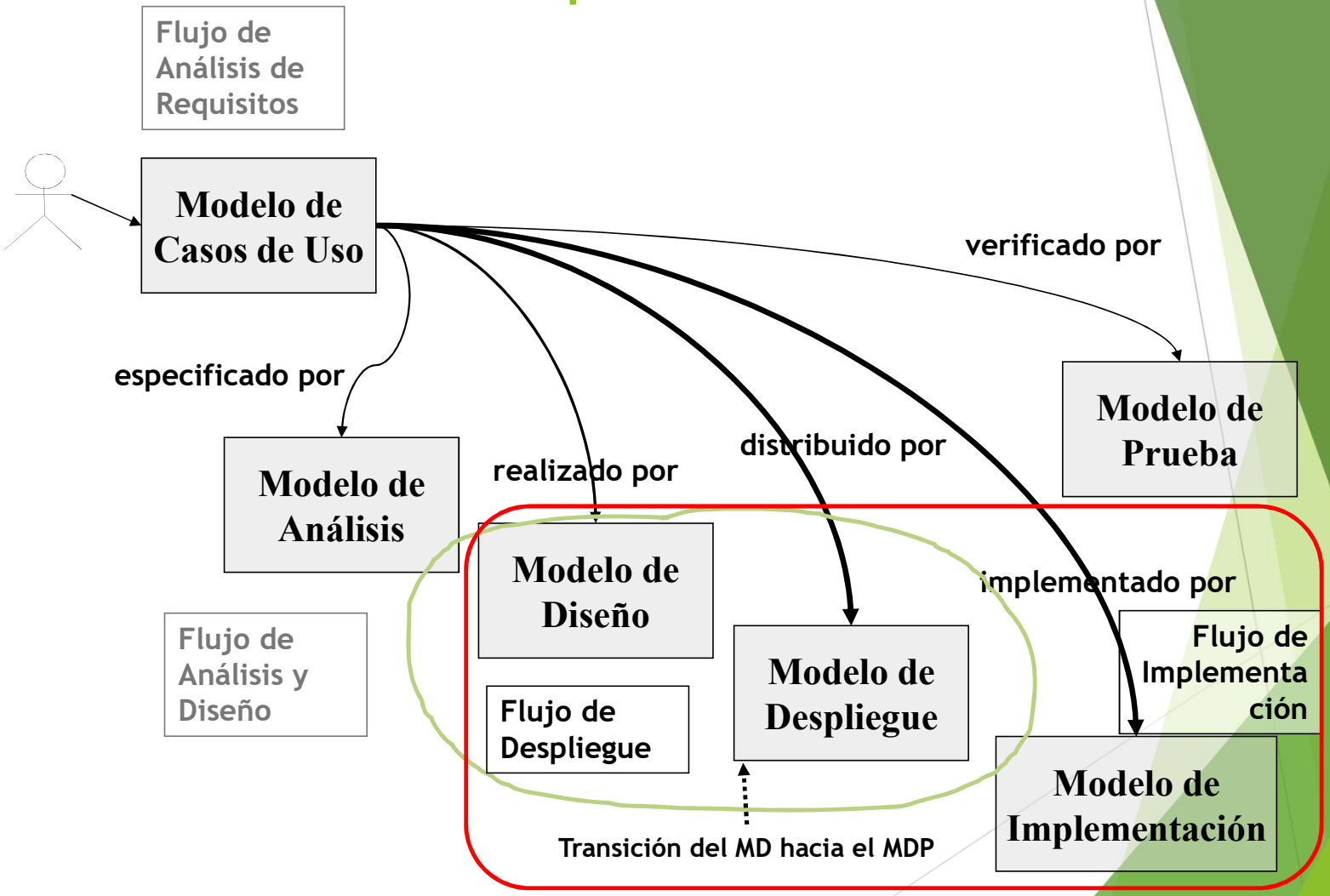


Diagrama de Clases de Diseño de Contexto

Modelado de Implementación

- El modelado de implementación se realiza para obtener:
  - La implementación del sistema en términos de lenguajes y elementos de programación.
  - La distribución de los módulo software en los elementos hardware del sistema.
- En el flujo de **implementación** se construye un modelo que representa la estructura y el comportamiento internos del sistema en cuanto a:
  - Componentes y módulos.
  - Arquitectura software del sistema.
- En el flujo de **despliegue** se construye un modelo que representa la estructura y el comportamiento internos del sistema en cuanto a:
  - Arquitectura hardware del sistema.

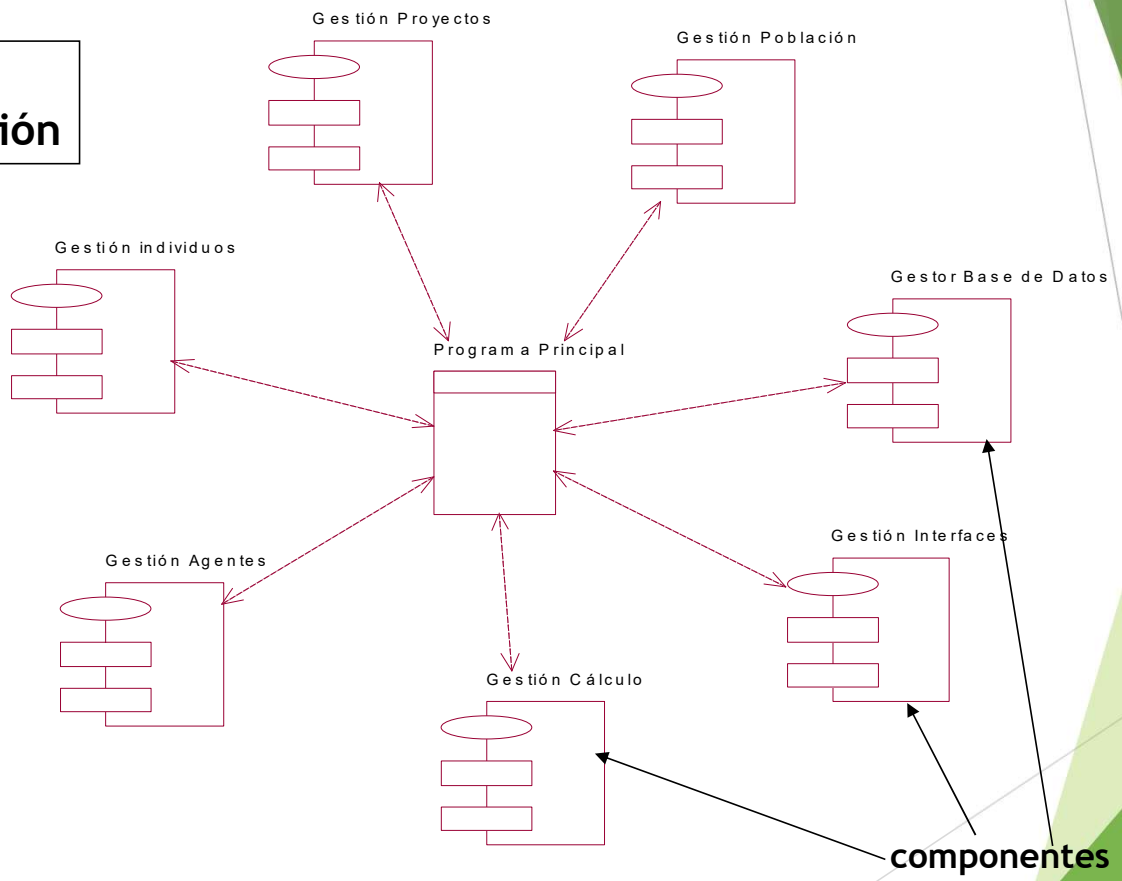
Modelado de Implementación



Modelado de Implementación

Modelo de Implementación

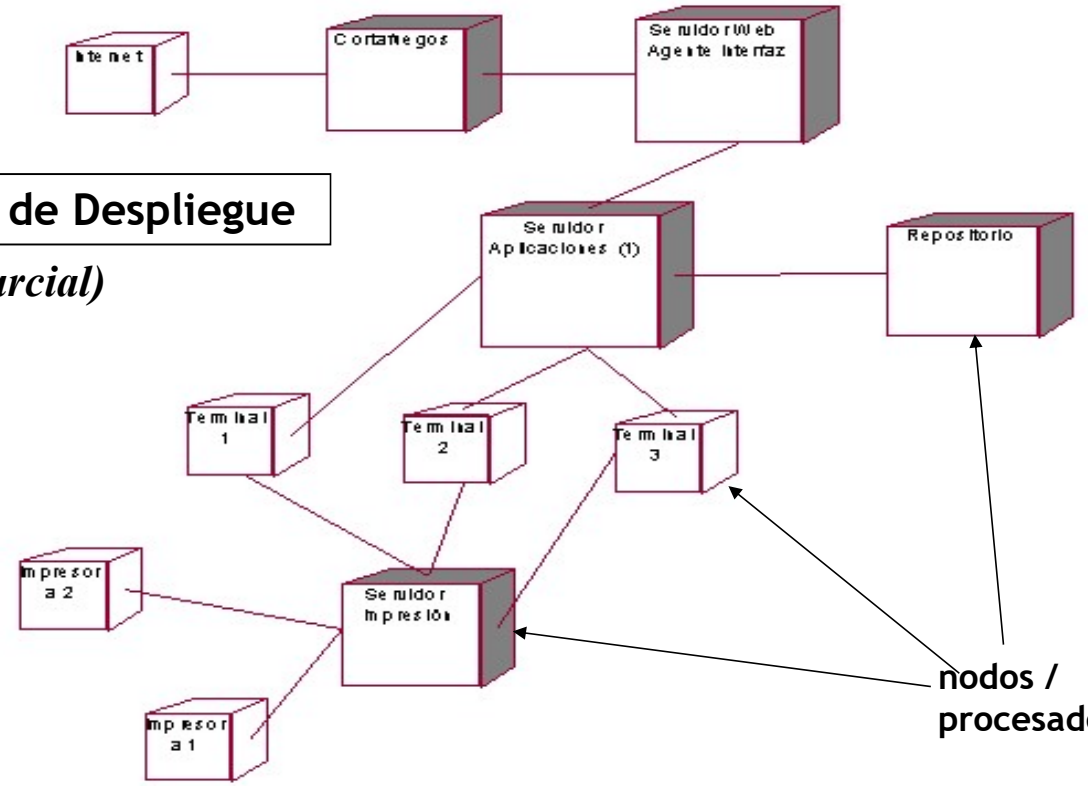
(Vista parcial)



Modelado de Implementación

Modelo de Despliegue

(Vista parcial)



Resumen

- El Proceso Unificado es una metodología creada principalmente para el desarrollo de software orientado a objetos.
- Utiliza el soporte de modelado de UML, pero es independiente de UML.
- El Proceso Unificado:
  - Es un Proceso iterativo.
  - Está centrado en la arquitectura.
  - Está dirigido por los casos de uso.
  - Es un proceso configurable.
  - Soporta las técnicas orientadas a objetos.
  - Impulsa un control de calidad y una gestión del riesgo objetivos y continuos.

Resumen

- La aplicación formal del Proceso Unificado supone:
  - Desventajas:
    - Grandes esfuerzos en la construcción de modelos.
    - Necesidad del soporte de herramientas informáticas.
  - Ventajas:
    - Disminuye el riesgo del error de análisis / diseño acumulado.
    - Aligera el esfuerzo en implementación.
    - Proporciona la documentación del ciclo de vida en el mismo proceso.

Resumen

- El Proceso Unificado es flexible y se puede adaptar al grado de complejidad del modelo de proceso de desarrollo (descarte de algunos modelos o flujos).
- El Proceso Unificado es abierto y permite la incorporación de enfoques y artefactos complementarios:
  - Patrones de diseño.
  - Patrones de implementación.
  - Marcos de diseño.
  - Combinación de varios modelos de proceso.
  - Arquitecturas Dirigidas por Modelos (*Model Driven Architectures*).
  - Ejecutabilidad de modelos: UML 2, validación y verificación formales.

Bibliografía

1. Booch G., Rumbaugh J., Jacobson I. El Lenguaje Unificado de Modelado, Addison-Wesley, Madrid, 1999.
2. Bruegge B., Dutoit A.H. Ingeniería de Software Orientado a Objetos, Prentice Hall– Pearson educación, México, 2002.
3. Jacobson I., Booch G., Rumbaugh J. El Proceso Unificado de Desarrollo de Software, Addison-Wesley, Madrid, 2000.
4. Pressman R.S. Ingeniería del Software. Un enfoque práctico (5ª ed.) Mc Graw-Hill; New York , 2001.
5. Rumbaugh J., Jacobson I., Booch G. El Lenguaje Unificado de Modelado. Manual de Referencia, Addison-Wesley, Madrid, 2000.
6. Sommerville I. Ingeniería de software, 6ª edición, Prentice Hall – Pearson educación, México, 2002.
7. Stevens P., Pooley R. Utilización de UML en Ingeniería del Software con Objetos y Componentes, Addison-Wesley, Madrid, 2002.
8. http://www.omg.org
9. http://www.uml.org

Bibliografía (cont.)

- **A Simplified Approach to RUP**
  Gary K. Evans
  President, Evanetics, Inc.
  http://www.therationaledge.com/content/jan_01/t_rup_ge.html

- **UML y Patrones, Introducción al Análisis y Diseño Orientado a Objetos**
  Craig Larman
  Prentice-Hall

- **Rational Unified Process, Best Practices for Software Development Teams**
  A Rational Software Corporation White Paper