




Análisis y Diseño de Sistemas 2

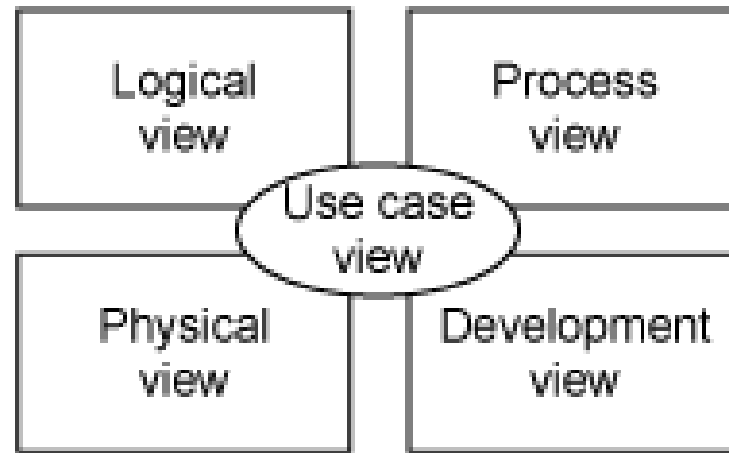
2015

Ing. Luis Alberto Arias Solórzano

Unidad 3

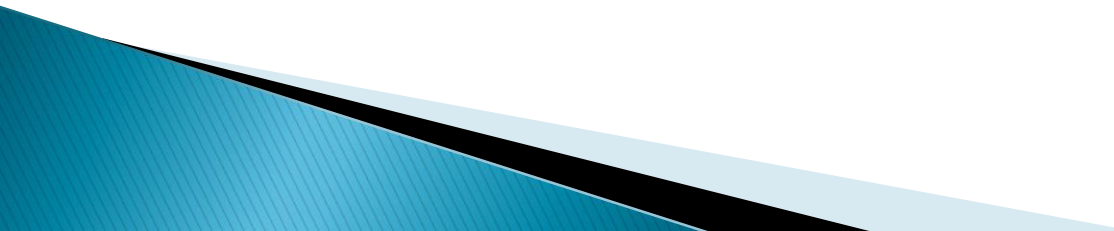
Model 4+1 Vistas

- ▶ Herramienta de arquitectura para realizar vistas y documentar una aplicación de software
–IBM Architectural Manifesto–
 - ▶ Introducido en 1995, describe la arquitectura de software utilizando cinco vistas concurrentes, cada una de las vistas se refiere a un conjunto específico de problemas/preocupaciones (concerns)
 - ▶ Cada una de las vistas se enfoca en algún elemento específico del sistema. El modelo 4+1 vistas es una excelente forma de aprender acerca de la arquitectura del sistema, tanto de los arquitectos como del resto del equipo. Los arquitectos lo utilizan para entender y documentar las muchas capas que una aplicación tiene, de manera sistemática y estandarizada.
- 

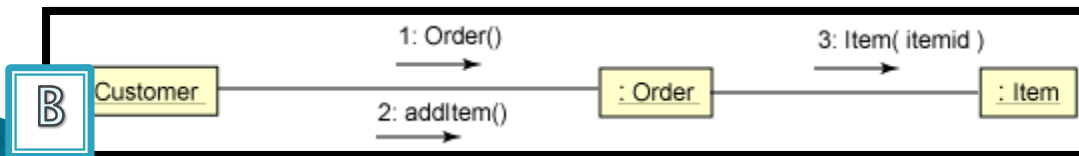
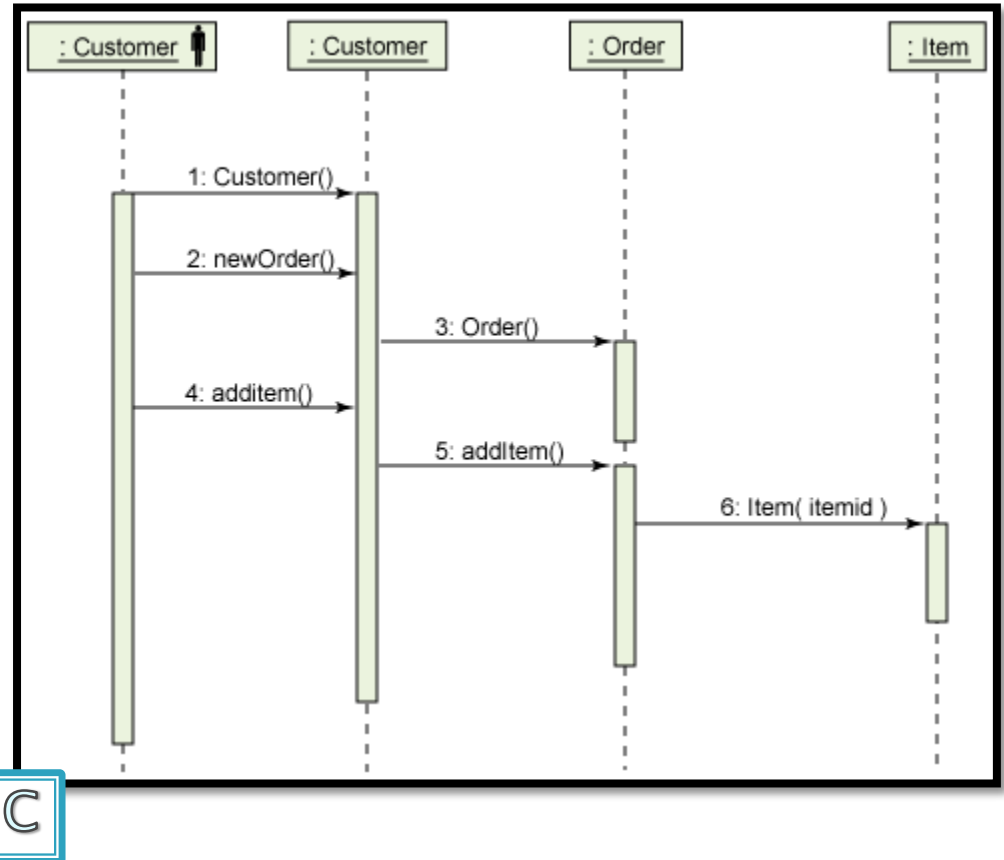
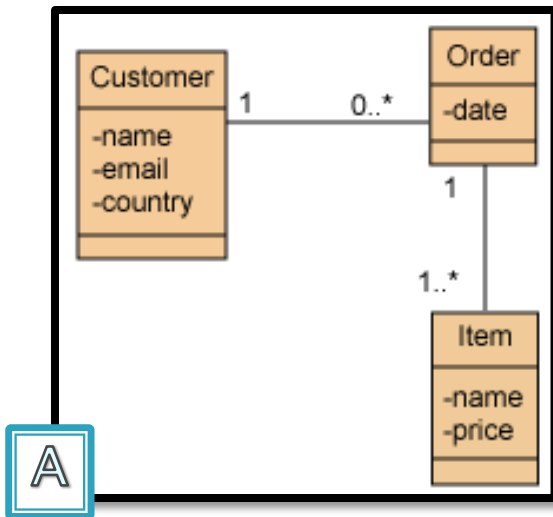


- ▶ **Lógica:** Modelo objeto del diseño (cuando se utiliza un diseño orientado a objetos)
- ▶ **Desarrollo:** Describe la organización estática del software y su ambiente de desarrollo
- ▶ **Proceso:** Captura aspectos de concurrencia y sincronización del diseño
- ▶ **Física:** Describe el mapeo del software en el hardware y refleja los aspectos distribuidos del sistema
- ▶ La arquitectura es descrita usando estas cuatro vistas y es ilustrada por medio de escenarios, la quinta vista **Casos de Uso** la cual intercepta todas las vistas

Vista lógica

- ▶ Soporta requerimientos de comportamiento y muestra como el sistema esta distribuido en un conjunto de abstracciones.
 - ▶ La forma de representar la vista lógica es principalmente haciendo uso de clases y objetos, se puede hacer uso de diagramas de clase, de colaboración y de secuencia, entre otros, la idea es mostrar la relación de los elementos desde un punto de vista lógico.
- 

Representando la vista lógica



- A: Diagrama de clases**
- B: Diagrama de colaboración**
- C: Diagrama de secuencia**

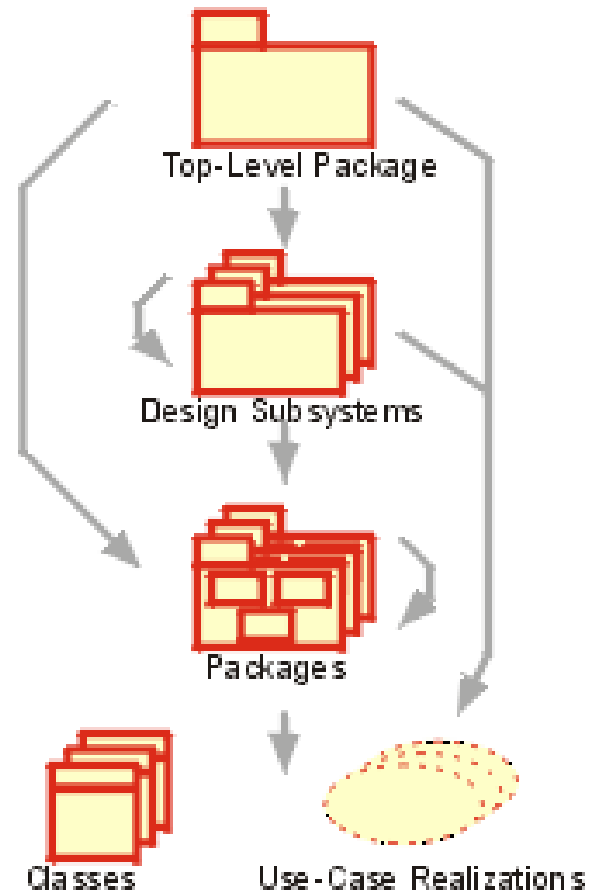
Vista l3gica

RUP

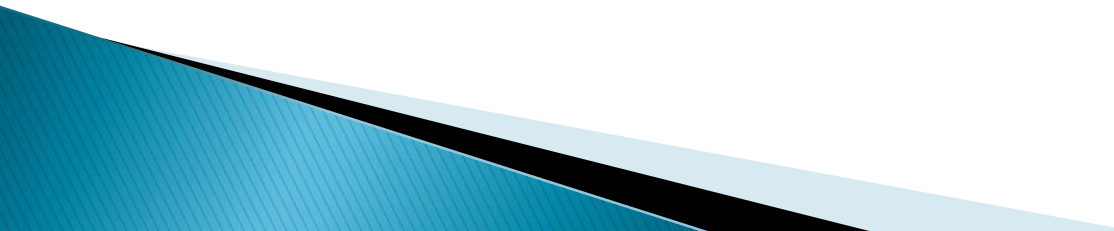
The Logical View

Shows an architecturally
significant subset of

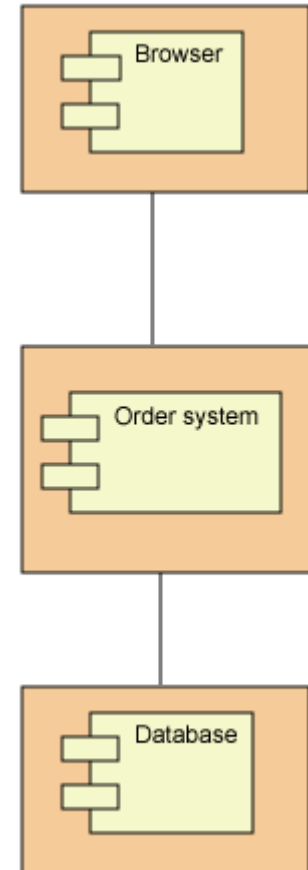
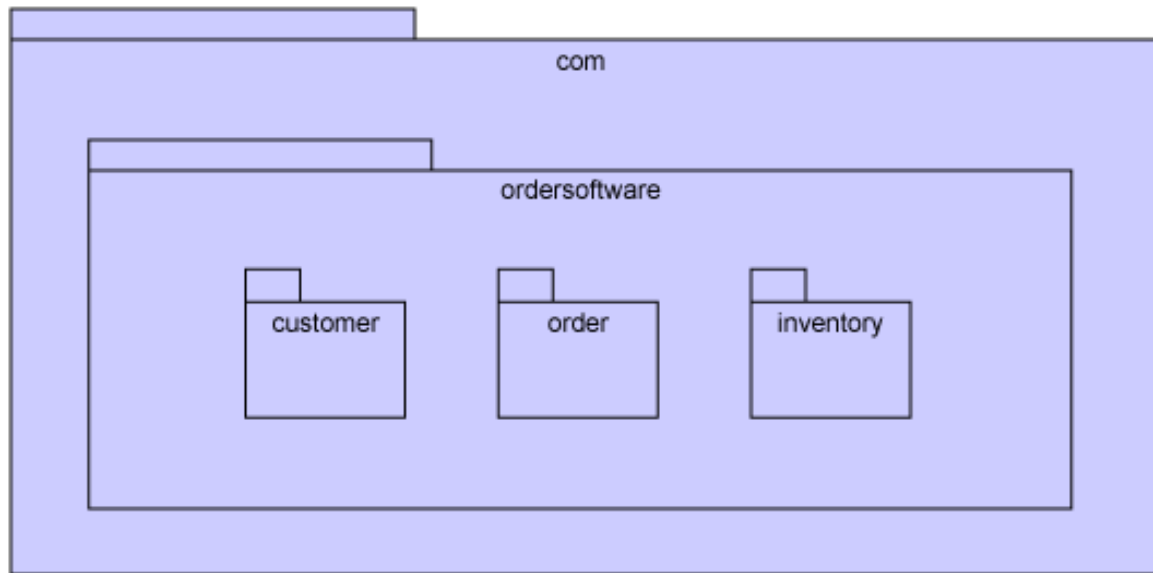
The Design Model



Vista de desarrollo

- ▶ Utilizada para describir los **módulos** del sistema. Los módulos son bloques de construcción más grandes que las clases y los objetos, además varían de acuerdo al ambiente de desarrollo.
 - ▶ Paquetes, subsistemas y librerías son todos considerados módulos
 - ▶ Se utiliza la vista de desarrollo para estudiar en donde se van a ubicar los archivos en el sistema y ambiente de desarrollo, es una buena forma de ver las capas del sistema en un sistema N-Capas: UI Layer, Presentation Layer, Application Logic Layer, Persistence Layer
- 

Vista de desarrollo

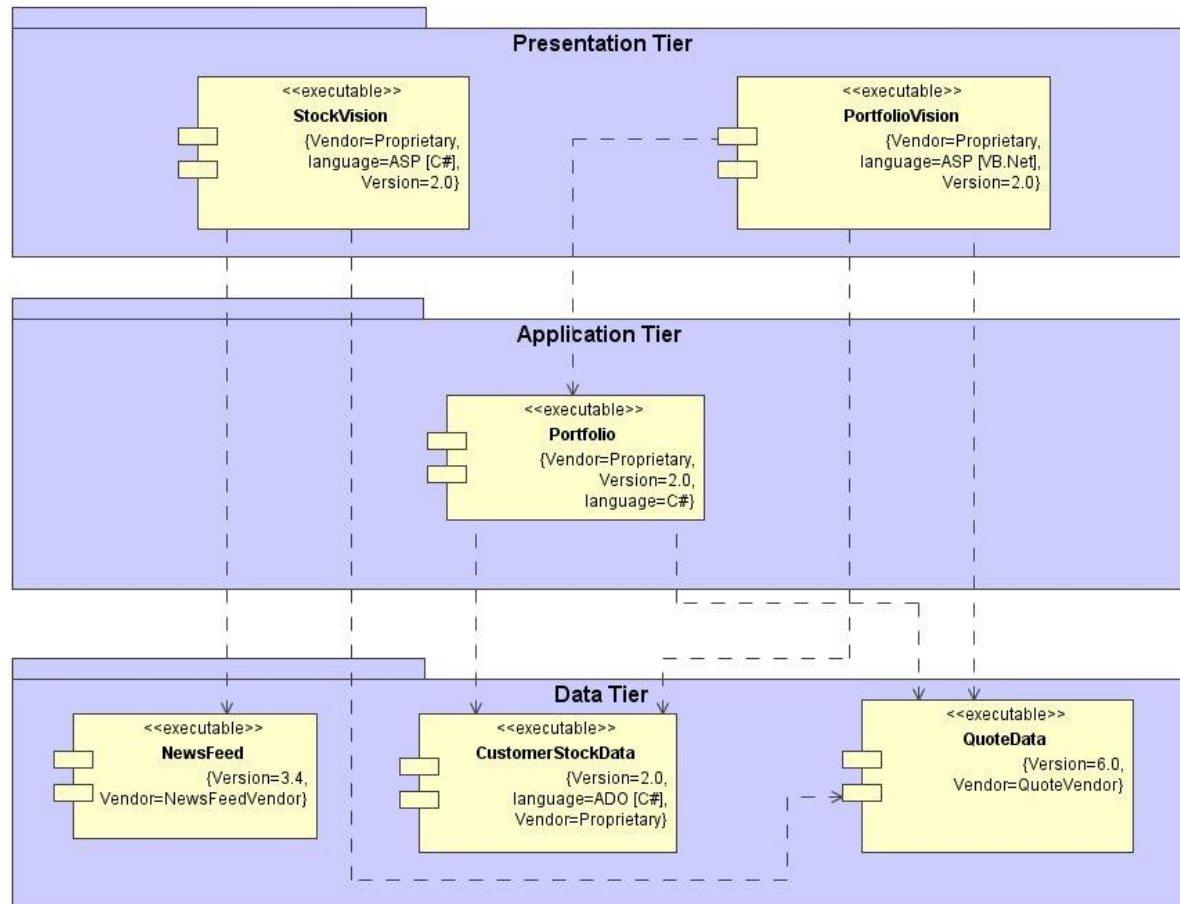


Vista de desarrollo

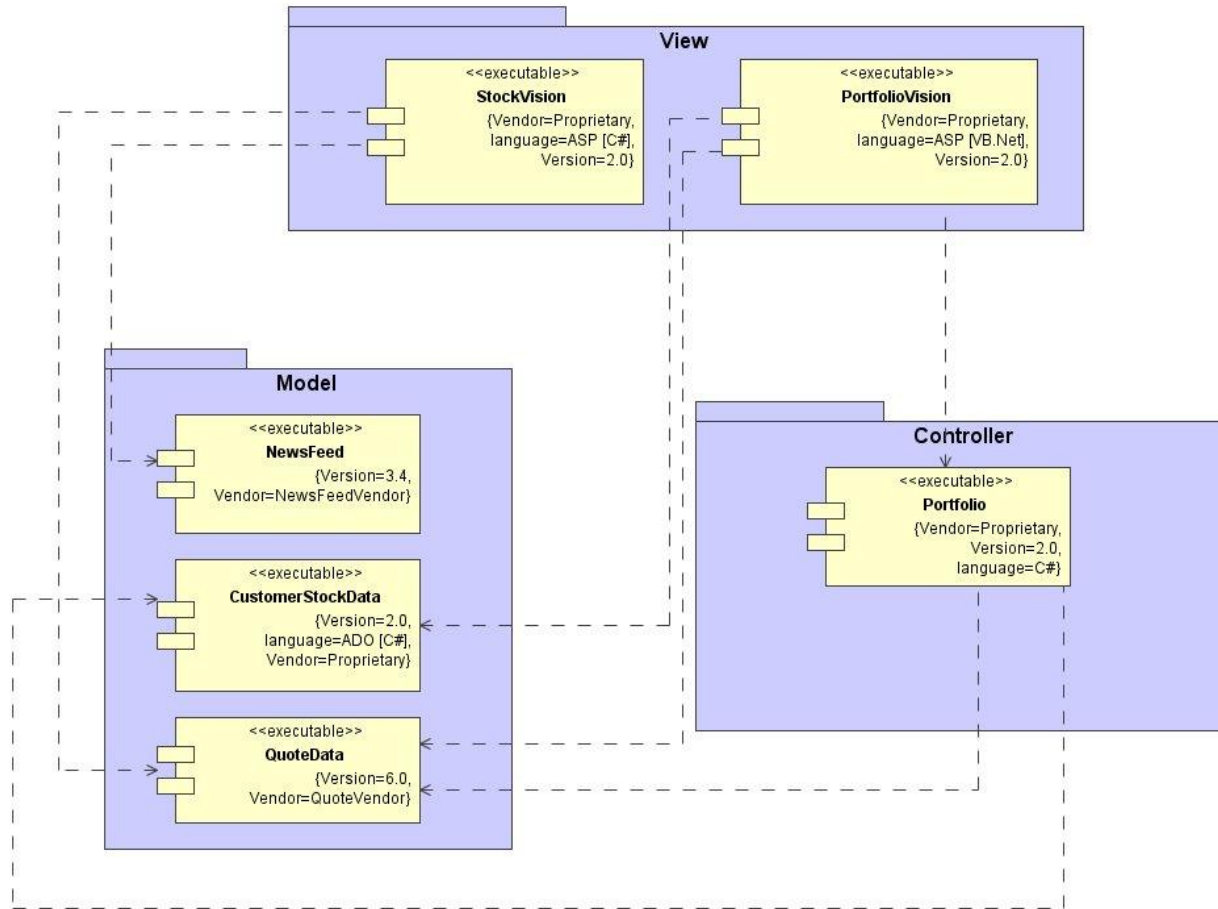
RUP (Implementation View)

- ▶ El propósito es capturar las decisiones de arquitectura en cuanto a:
 - Subsistemas
 - Diagramas de componentes que ilustran como los subsistemas están organizados en capas y jerarquías
 - Dependencias entre subsistemas (imports)
- ▶ Utilizada para:
 - Asignar tareas de implementación (desarrollo) a equipos individuales de desarrollo
 - Evaluar la cantidad de código a ser desarrollada, modificada o eliminada
 - Razonar la reutilización, a gran escala
 - Considerar estrategias de entregas (release)

3-Tier



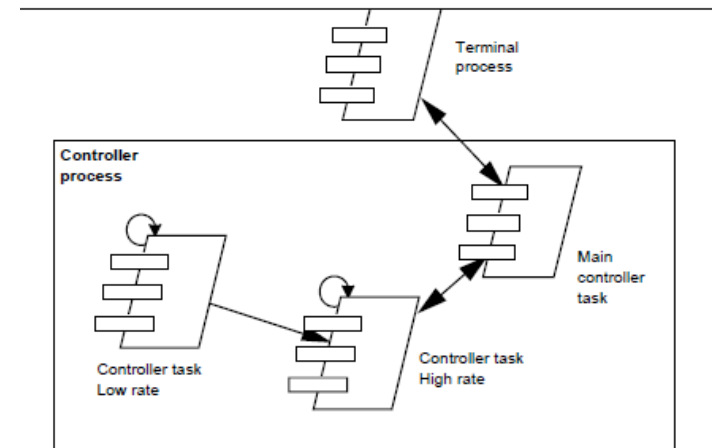
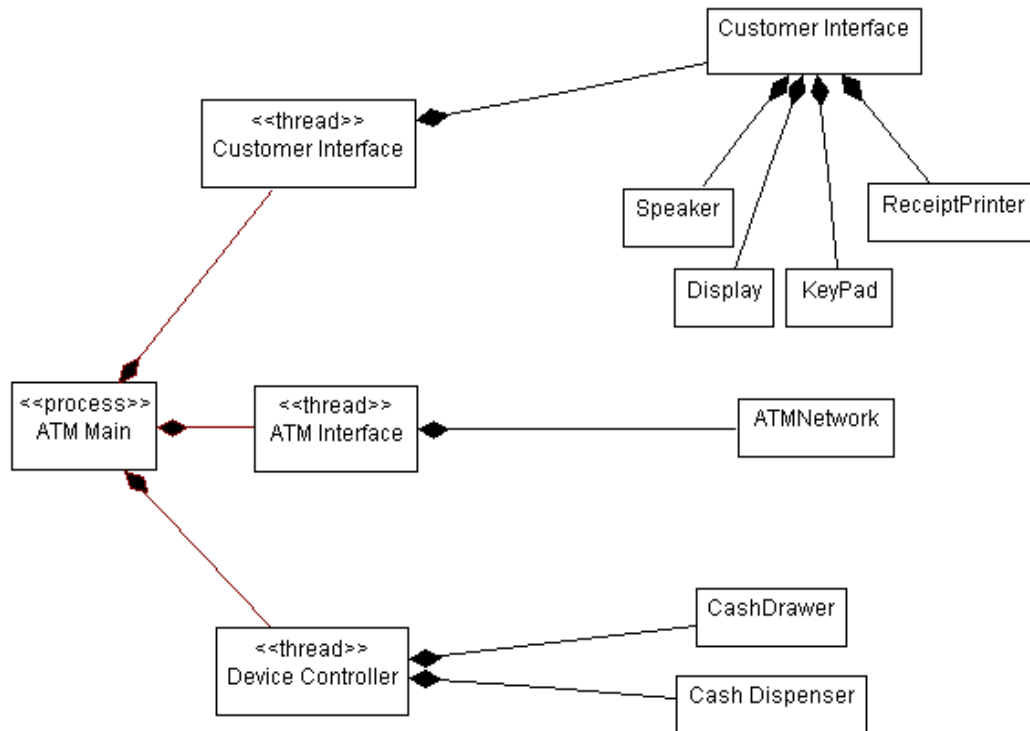
MVC



Vista de proceso

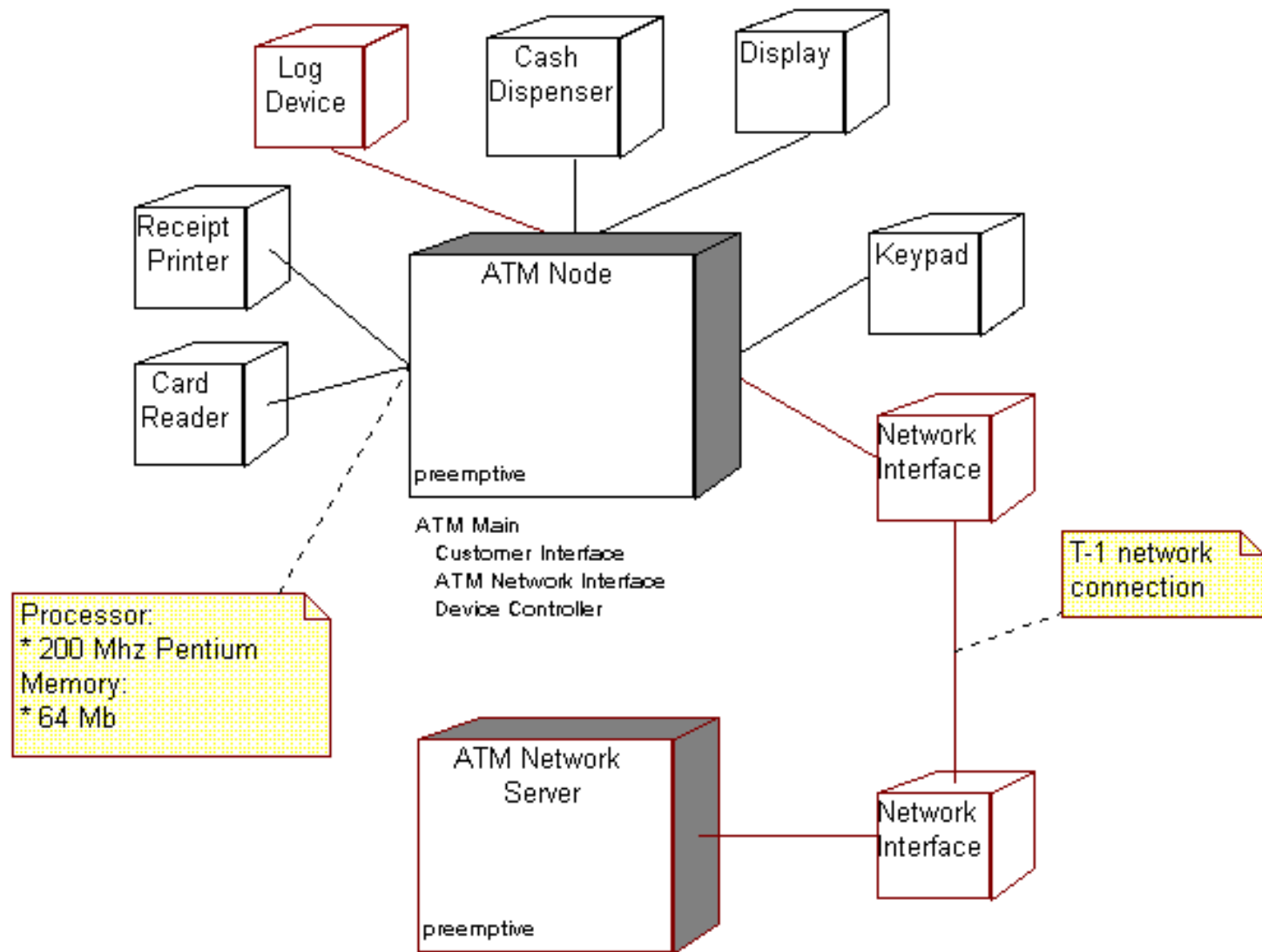
- ▶ La vista de proceso permite describir y estudiar los procesos del sistema y como estos se comunican, ésta vista es muy útil cuando se tiene múltiples, simultáneos procesos o hilos en el software.
- ▶ Hay solo una vista de proceso en el sistema, ésta ilustra la descomposición de los procesos del sistema, incluyendo el mapeo de clases y subsistemas en procesos e hilos.
- ▶ La vista de procesos nos permite identificar problemas de concurrencia, tiempo de respuesta, deadlocks, throughput, tolerancia a fallas y escalabilidad

Vista de proceso

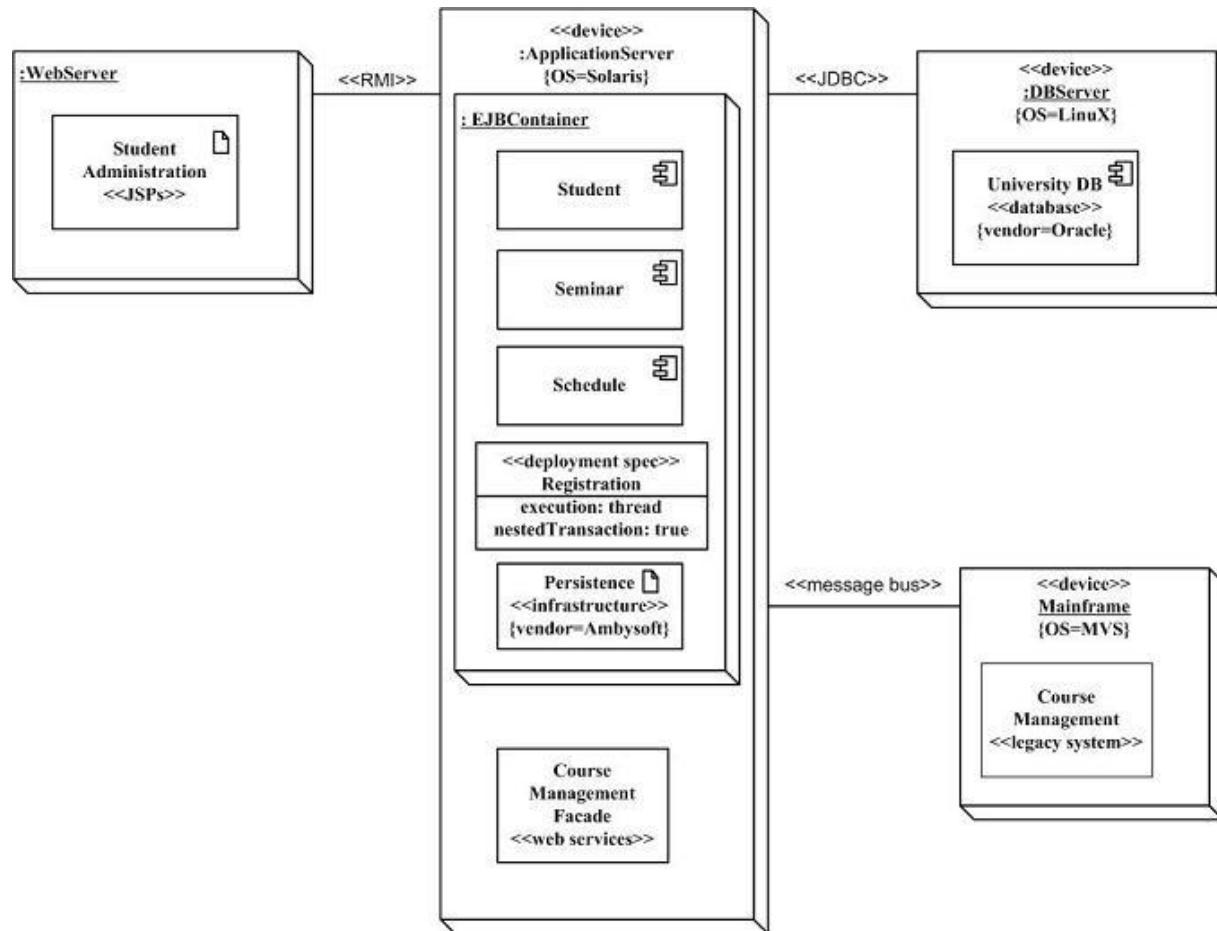


Vista física

- ▶ Provee bases para entender la distribución física del sistema en un conjunto de nodos del sistema, usualmente ilustrada por medio del diagrama de deployment
- ▶ Representa como la aplicación es instalada y como se ejecuta en una red de computadoras, esta vista toma en cuenta requerimientos no funcionales como, disponibilidad, confiabilidad, performance y escalabilidad

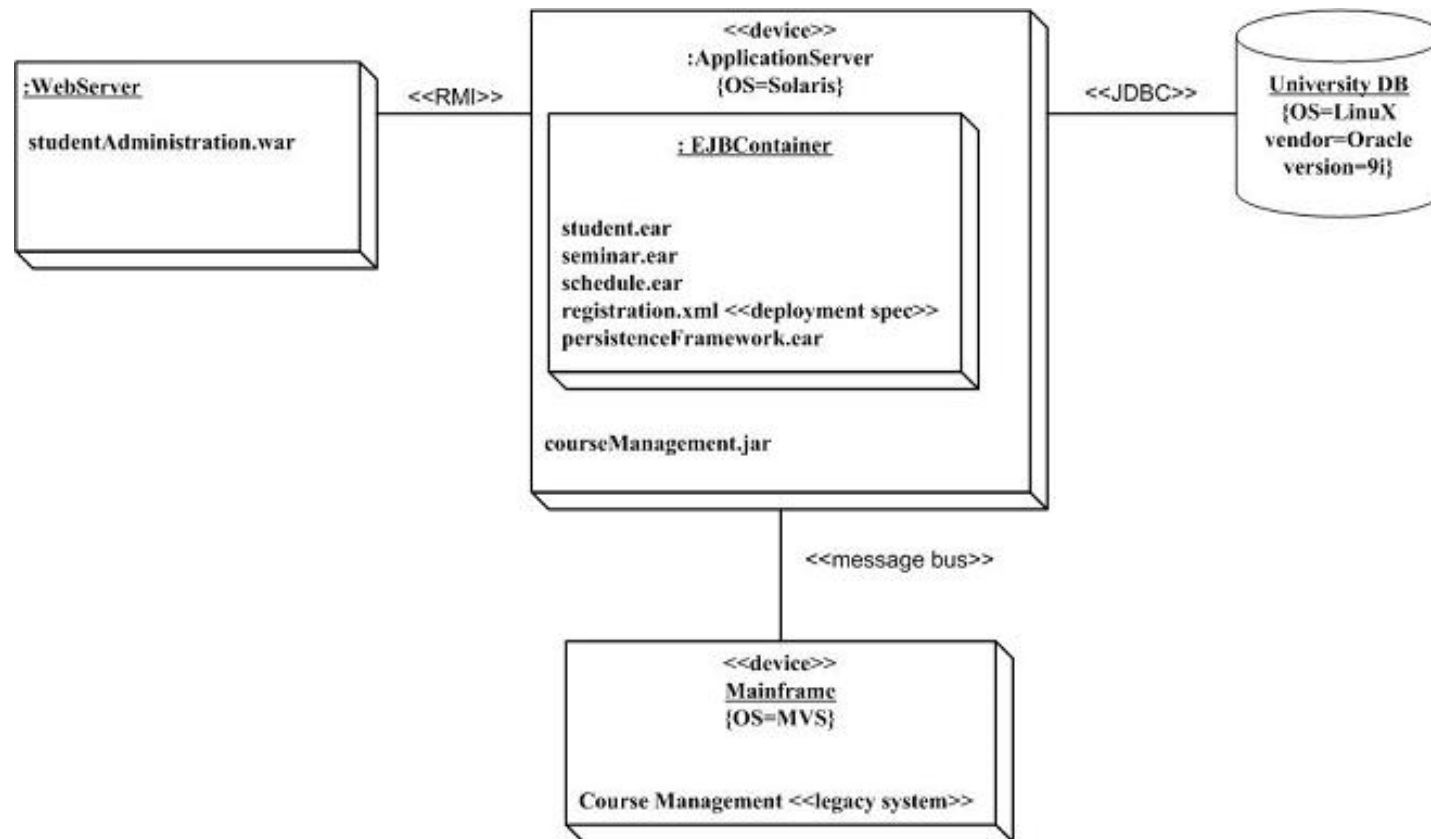


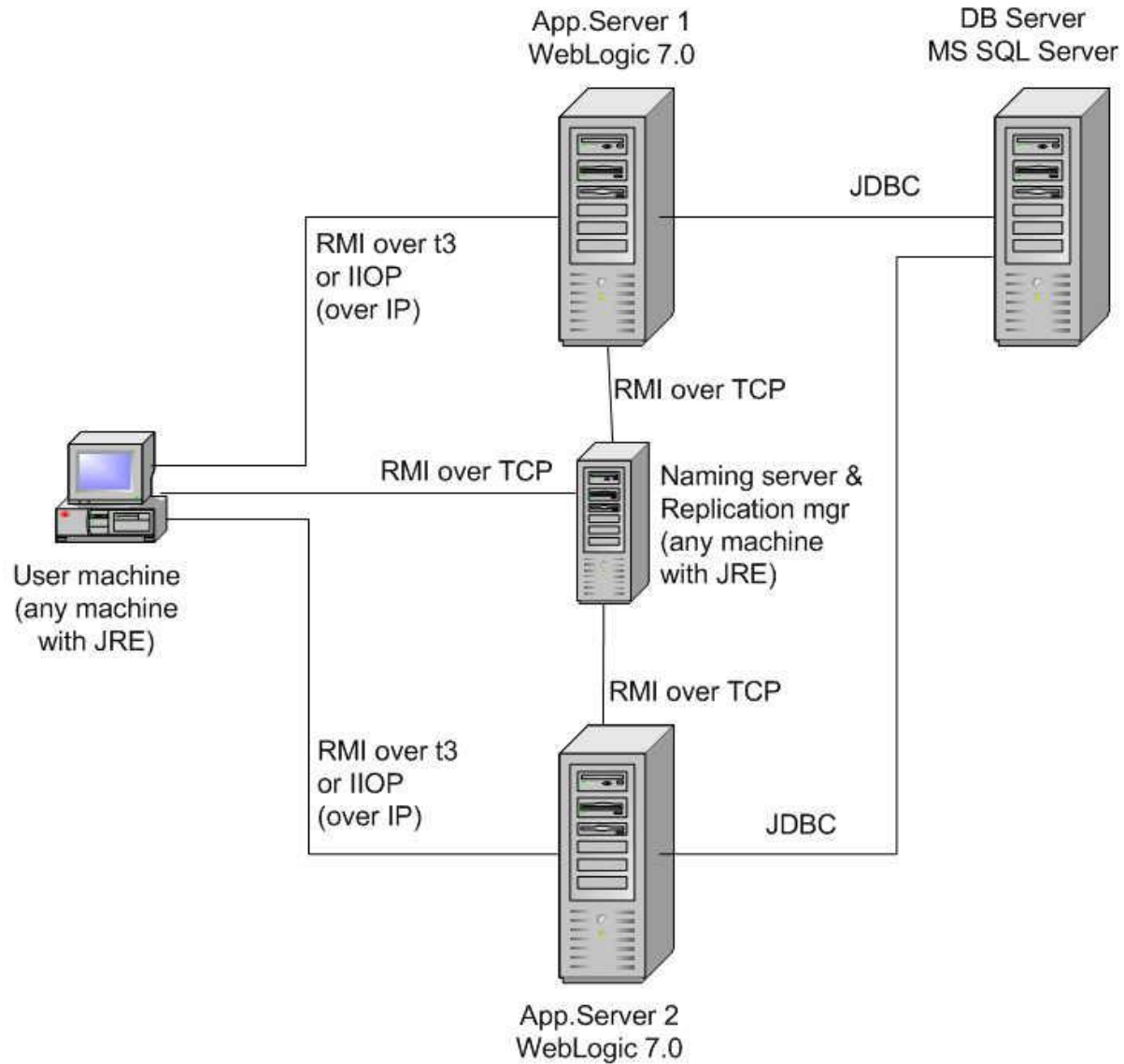
UML 2 Deployment Diagram



UML 2

Deployment diagram (Concise)





Architectural Blueprints

The “4+1” View Model of Software Architecture

– *Philippe Krutchen* –

- ▶ <http://www.cs.ubc.ca/~gregor/teaching/papers/4+1view-architecture.pdf>

Lectura obligatoria



Enlaces

UML Modeling

- ▶ **Deployment Diagram:**

<http://www.agilemodeling.com/artifacts/deploymentDiagram.htm>

- ▶ **Class Diagram:**

<http://www.agilemodeling.com/artifacts/classDiagram.htm>

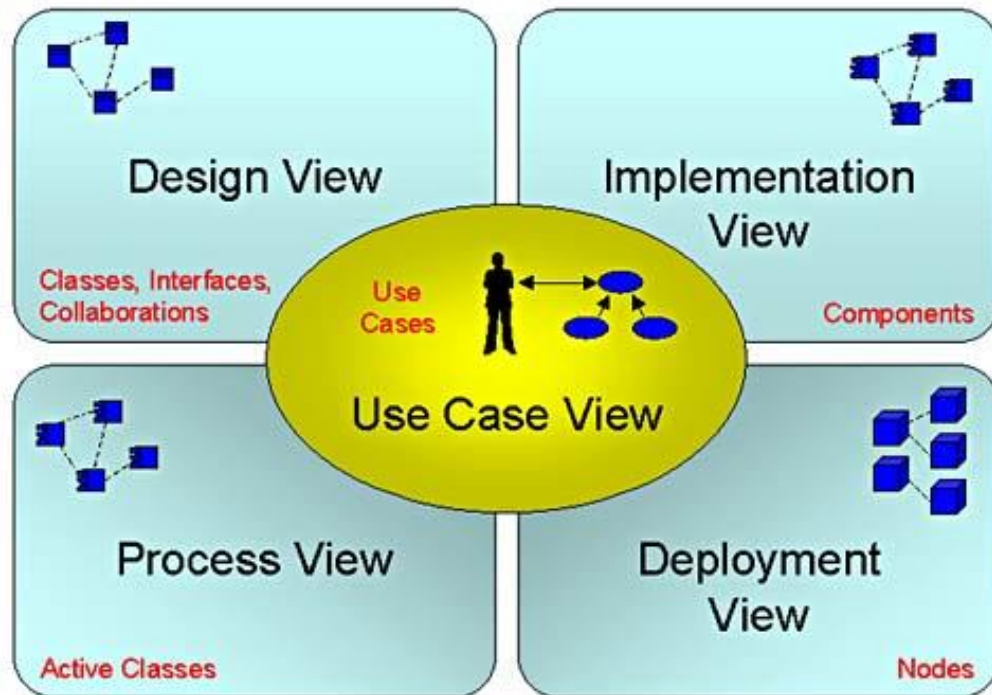
<http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/bell/index.html>

- ▶ **Sequence Diagram:**

<http://www.agilemodeling.com/artifacts/sequenceDiagram.htm>

<http://www.ibm.com/developerworks/rational/library/3101.html>

¿Dudas?



Gracias