Maestría en Software

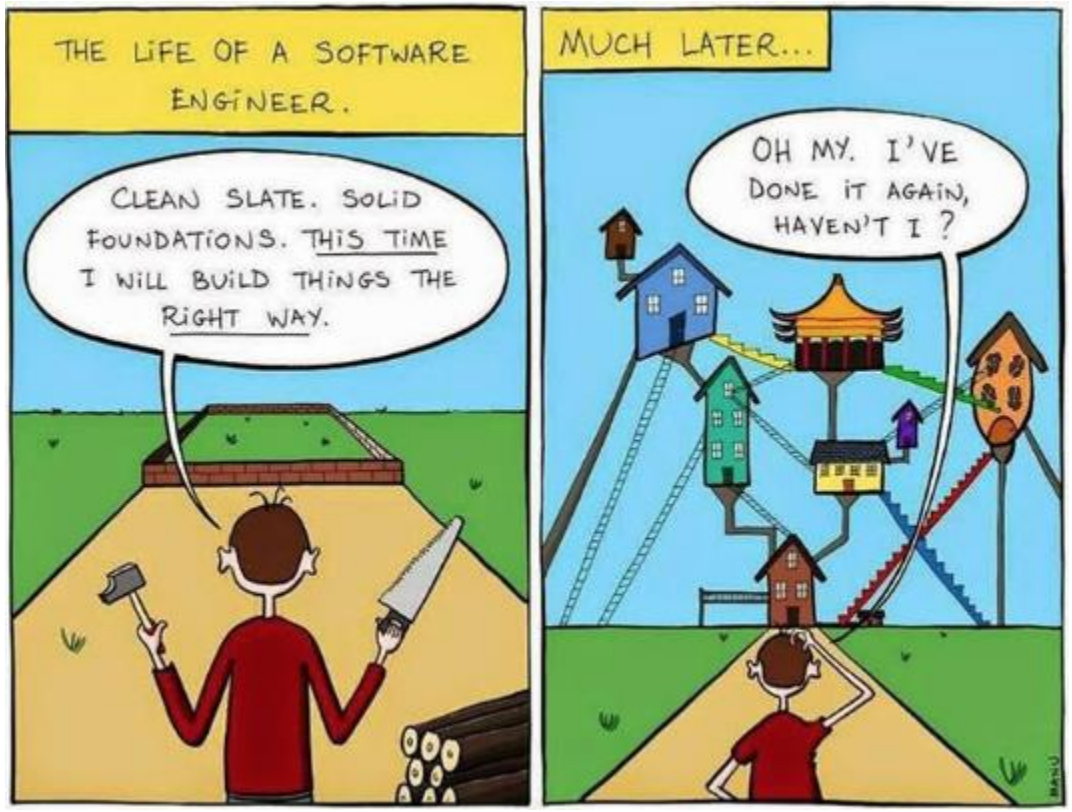
Modelado Arquitectónico en Ambientes Cloud

Nociones arquitectónicas

@aacabrera

Introducción

- Disciplina cuyos primeros trabajos datan de 1970
- Incremento de complejidad en el desarrollo de sistemas complejos y de misión crítica.
- Constructo fundamental de la ingeniería de sistemas y desarrollo de software.



Algo de historia

- En sus inicios la representación arquitectónica utilizo mecanismos:
  - Confusos
  - Inconsistentes
  - Imprecisos
  - Desorganizados
- Representación diagramática y textual la ***estructura*** y ***comportamiento*** de un sistema

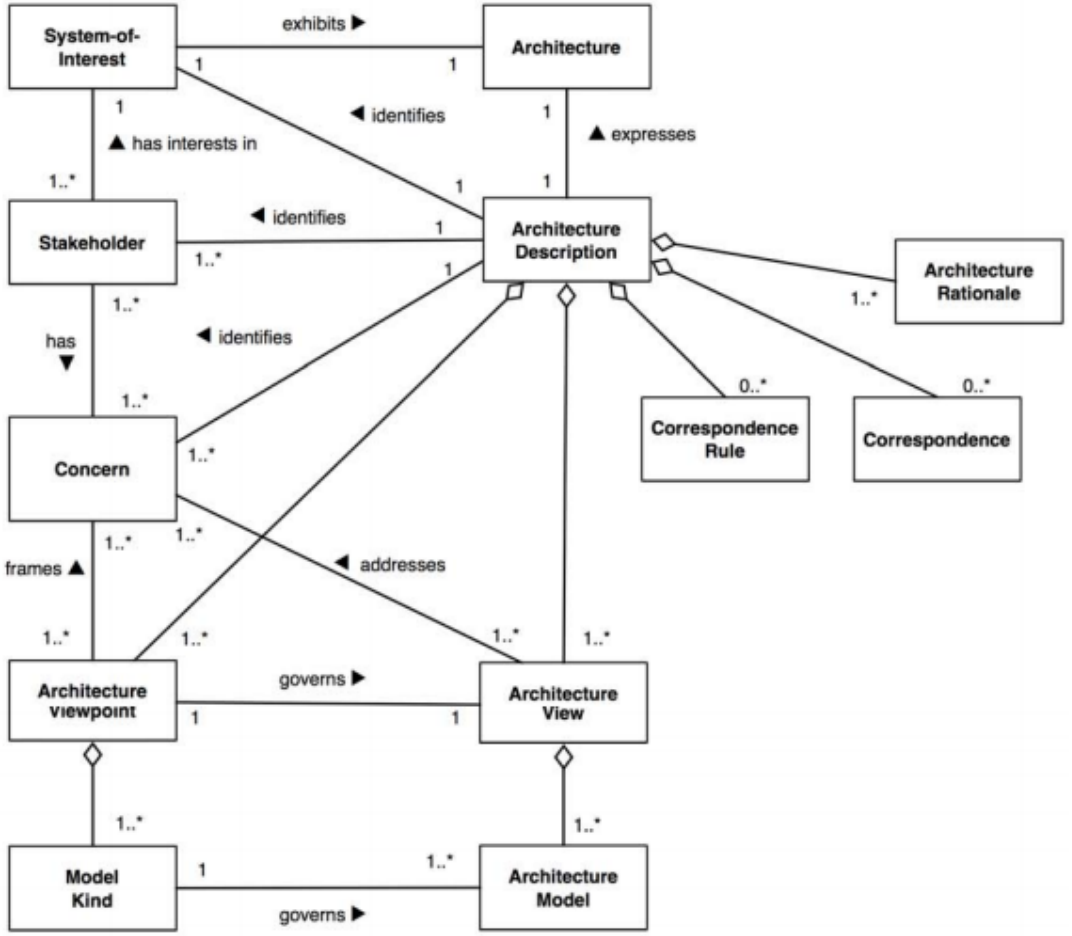
El dilema

Representación Arquitectónica

Tradicional

Deseada

Descripción arquitectónica



ISO/IEC/42010 Systems and software engineering — Architecture description

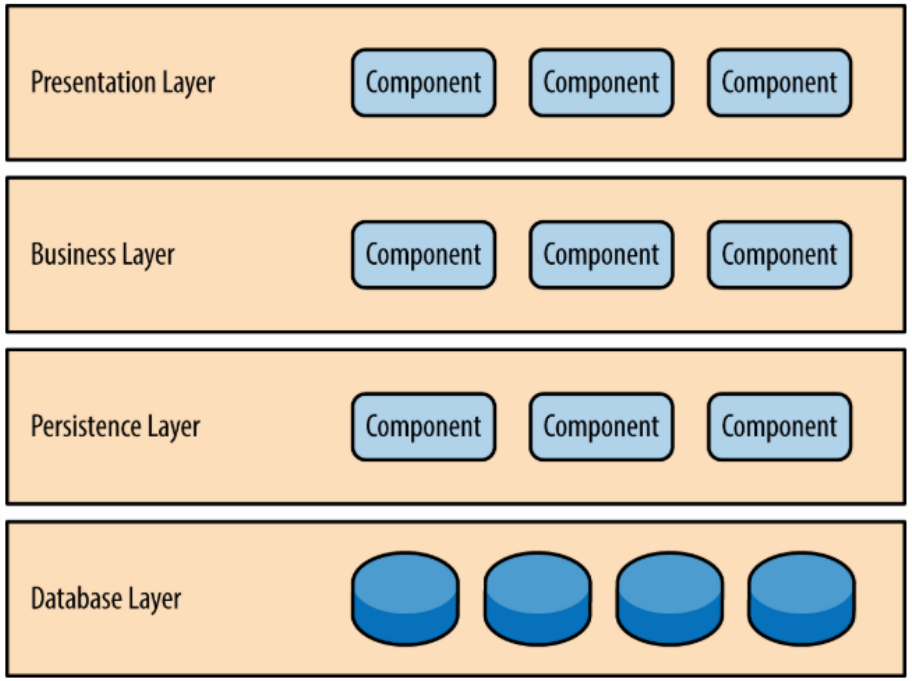
Definición

"La arquitectura de software de un programa o sistema informático es la **estructura o estructuras** del sistema, que comprende ***componentes*** de software, las ***propiedades*** visibles externamente de esos componentes y las ***relaciones*** entre ellos."

Bass, Clements, and Kazman (2012)

Definición

- Compuesta de constructos de grano grueso (coarse-grained) -> ***componentes de software*** -> bloques de construcción de la arquitectura.
- Detalles internos de como cada componente es diseñado e implementado no debe ser una preocupación para el resto del sistema.
- Detalles internos no son expuestos -> cajas negras



Software architecture patterns- O'Reilly

Introducción

“La arquitectura en lo que se refiere a la ingeniería de software consiste en ***descomponer o dividir*** un solo sistema en un conjunto de partes que se pueden construir de forma ***modular, iterativa, incremental e independiente***. Estas partes individuales tienen, como se mencionó anteriormente, ***relaciones explícitas*** entre ellas que, cuando se entrelazan o agrupan, forman el sistema, es decir, la arquitectura de software de la aplicación.”

El desafío

https://www.comakeit.com/blog/a-primer-on-hybrid-mobile-applications/

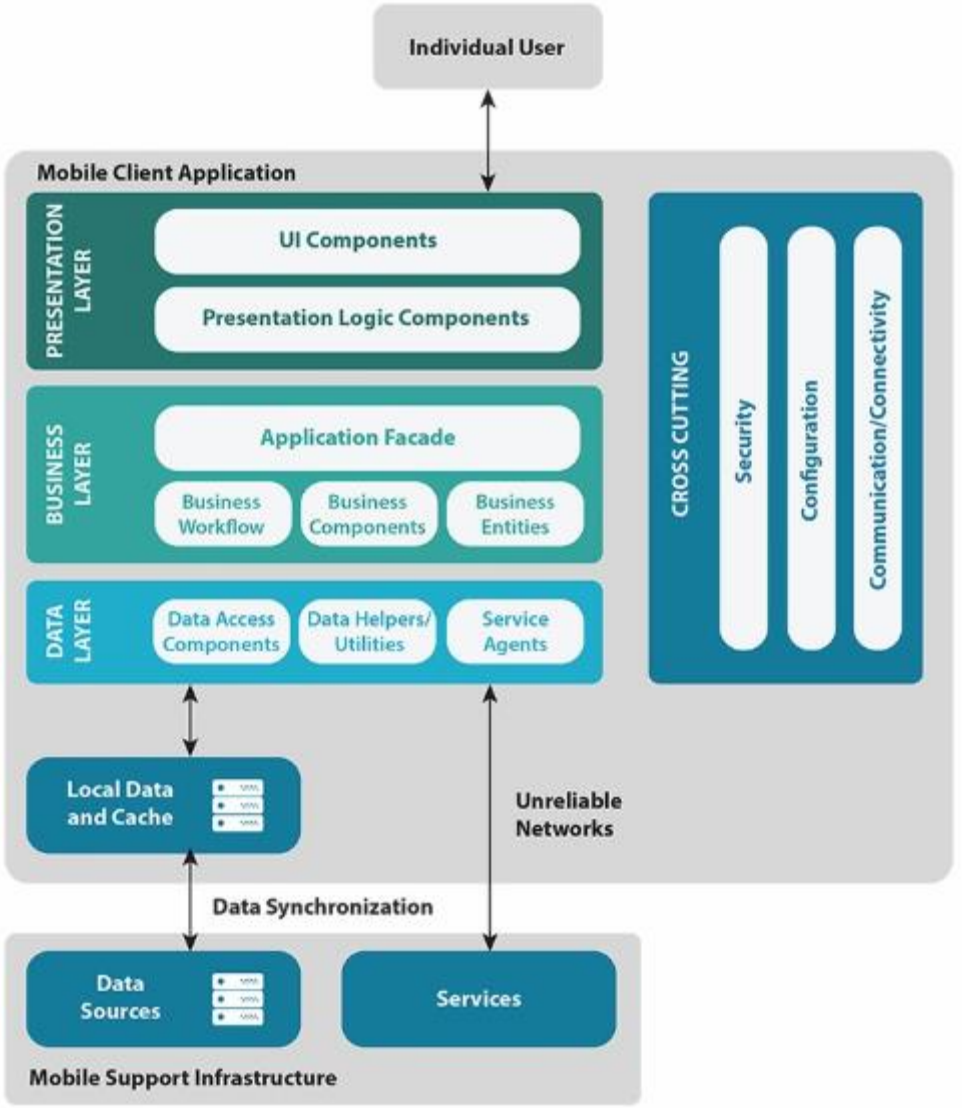


Image source: Microsoft Developers Network

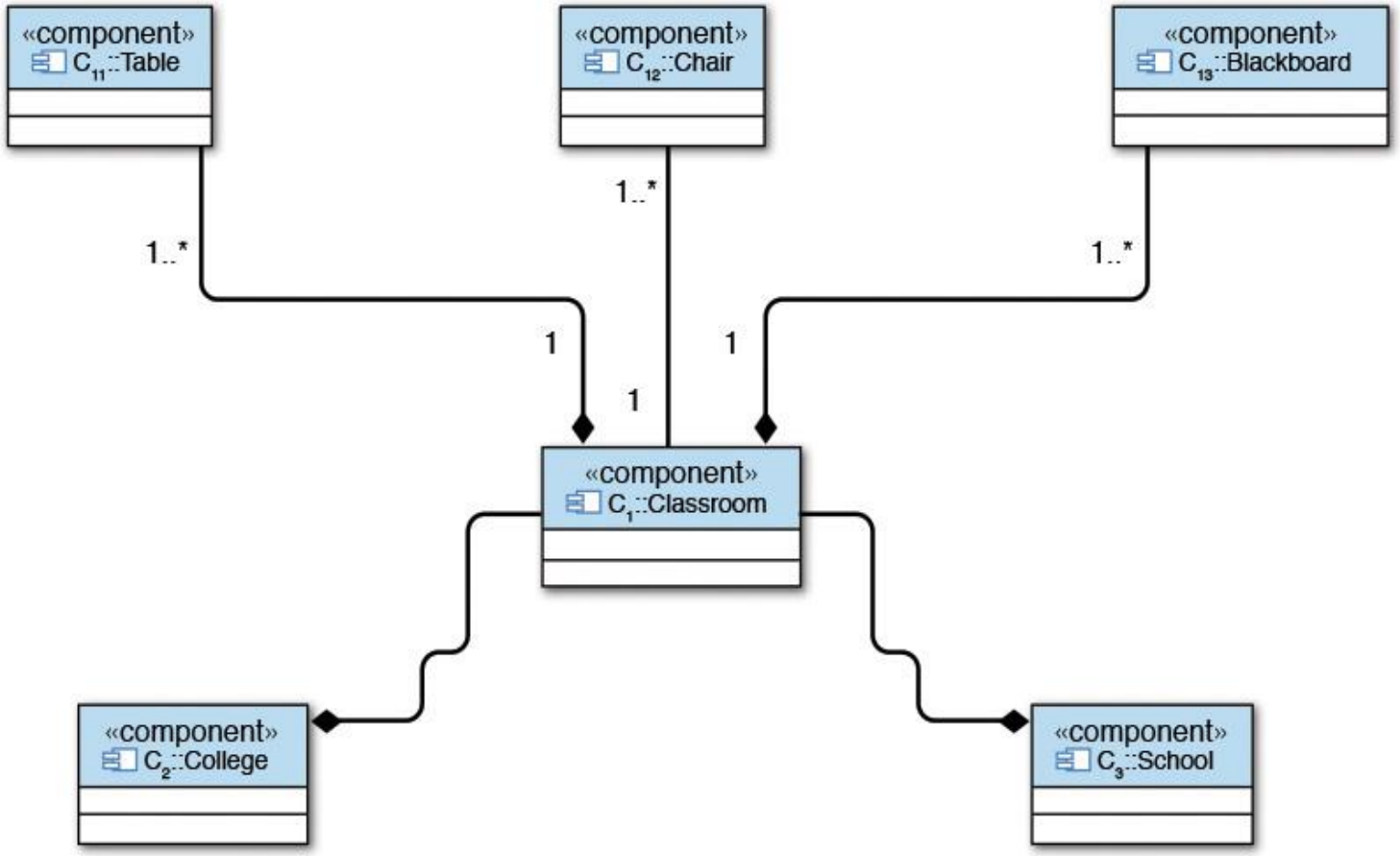
"Todas las ***arquitecturas*** son ***diseños***, pero no todos los ***diseños*** son ***arquitecturas***"

Bass, Clements, and Kazman (2012)

Arquitectura-Diseño

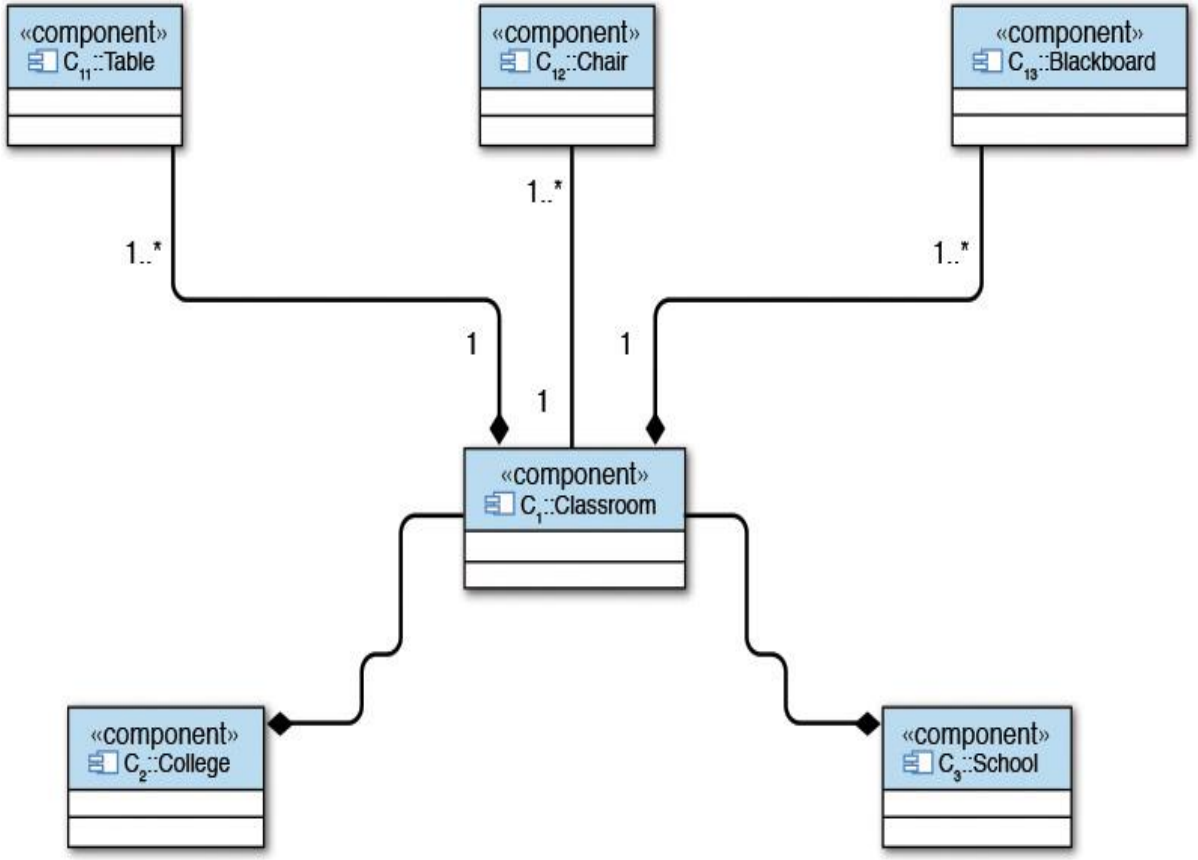
- La arquitectura considera un como una caja negra,
- El diseño trata con la configuración, la personalización y el funcionamiento interno de un componente de software
- La arquitectura confina un componente de software a sus propiedades externas.
- El diseño suele ser mucho más relajado, ya que tiene muchas más opciones con respecto a cómo adherirse a las propiedades externas del componente; considera varias alternativas de cómo se pueden implementar los detalles internos del componente.

Arquitectura-Diseño



Arquitectura-Diseño

- C1 Classroom
  - C11 Table object
  - C12 Chair object
  - C13 Blackboard object



"Dividir un sistema complejo en partes constituyentes pequeñas y luego concentrarse en cada parte para lograr mayor entendimiento y elaboración"

Views & Viewpoint

- ***Viewpoint:***

“Una especificación de las convenciones para construir y usar una vista.

"Un patrón o plantilla desde la cual desarrollar vistas individuales mediante el establecimiento de los propósitos y la audiencia para una vista y las técnicas para su creación y análisis".

- ***View:***

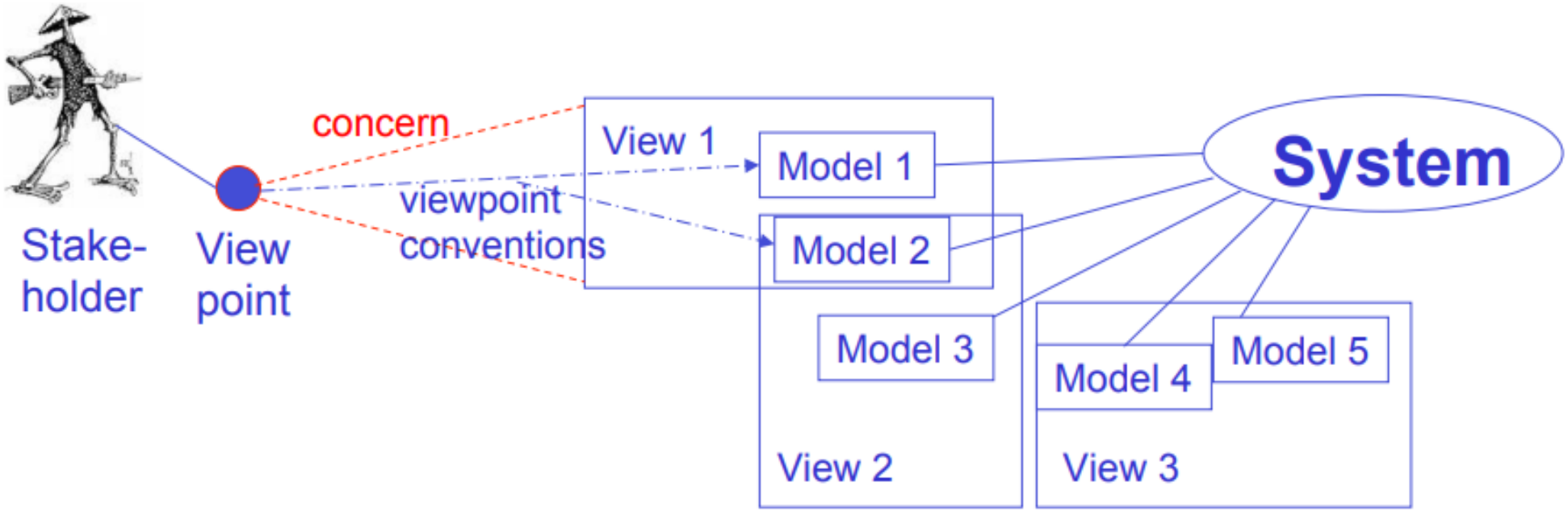
“Una representación de todo un sistema desde la perspectiva de un conjunto de inquietudes relacionadas”.

Views & Viewpoint

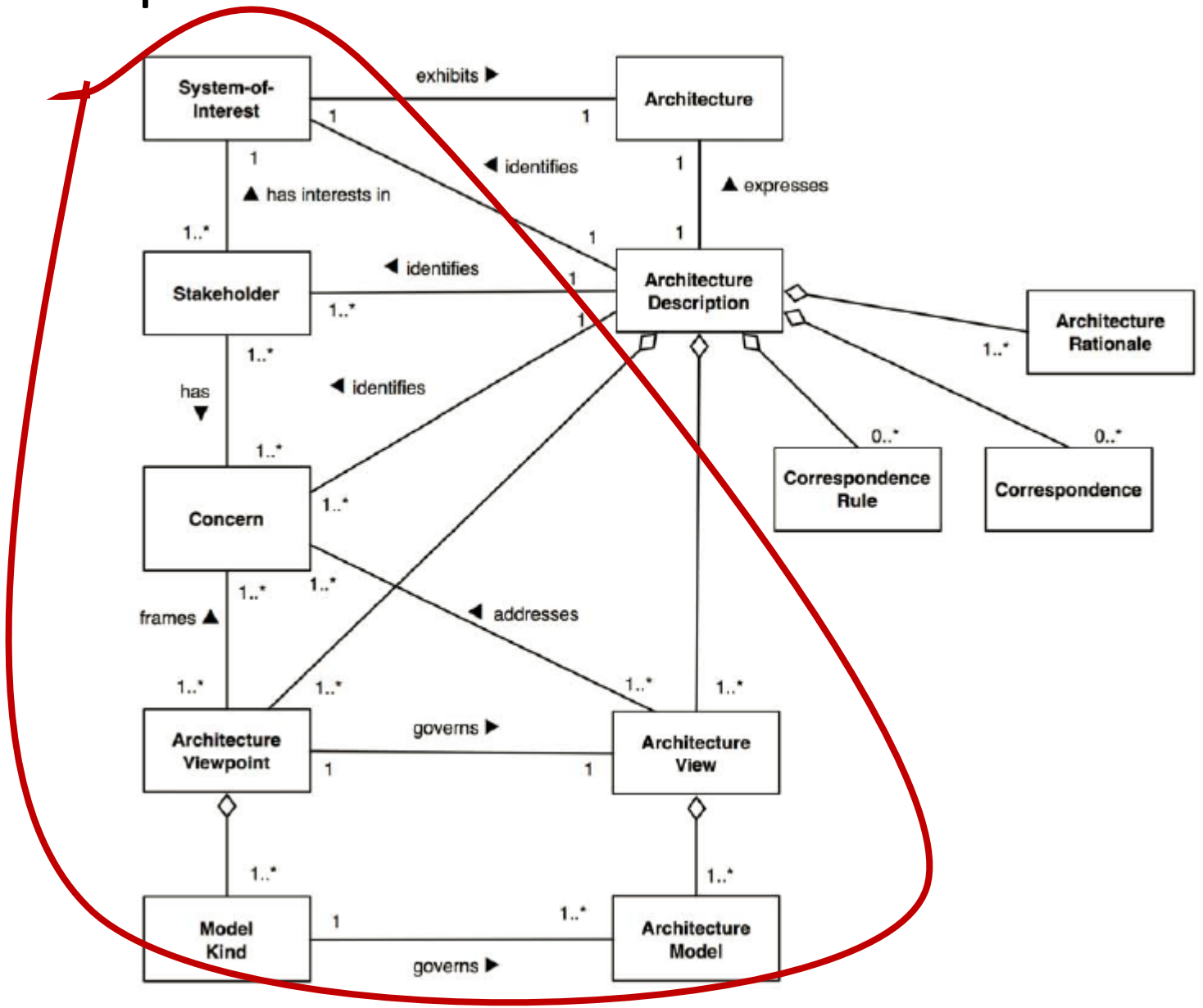
- El punto de vista debe gobernar la vista
- Relación similar a: tipo/instancia, gramática/lenguaje, ...
- Los puntos de vista son genéricos y pueden almacenarse en bibliotecas para su reutilización.

- Una vista siempre es específica de la arquitectura para la que está creada.

Views & Viewpoint



Views & Viewpoint

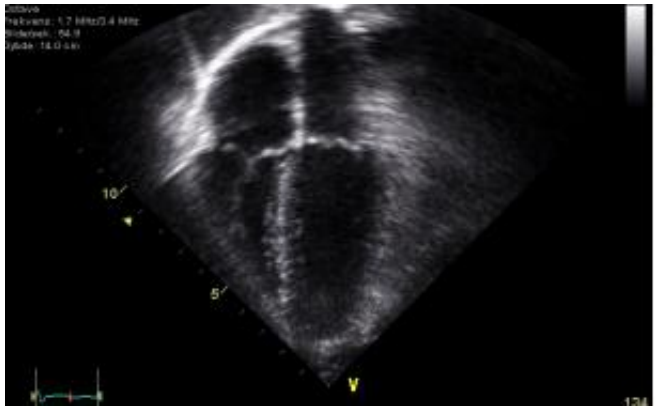


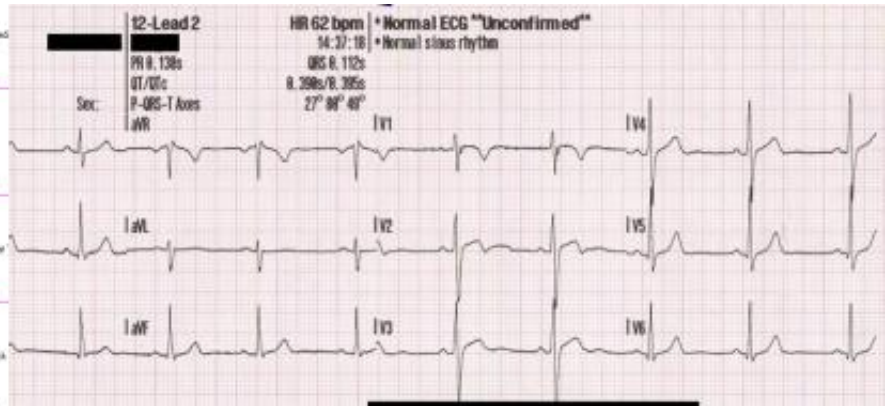
Views & Viewpoint

- **Sistema:** corazón humano
- **Stakeholder:** cardiólogo
- **Preocupaciones:** identificar y tratar desordenes de corazón
- **Viewpoint:**
  - Modelos usados: elctrocardiograma, ecocardiograma, ballistocardiograma
  - Tecnicas de Analisis: Interpretación ECG
- **Vista**

Model Kind

Archiitecture Model





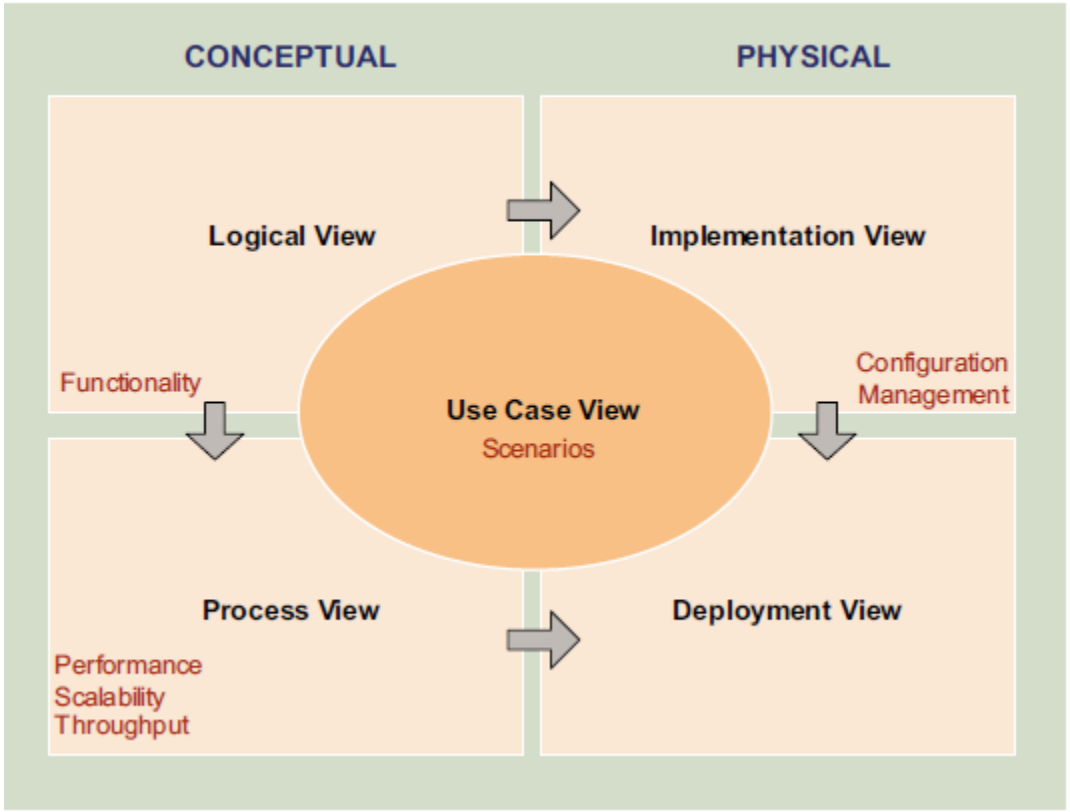
Views & Viewpoint

- Philippe Kruchten (1995, noviembre) fue el pionero que postuló el uso vistas y puntos de vista para abordar las diversas inquietudes de cualquier arquitectura de software.

- Kruchten formo parte del organismo de estándares IEEE 1471, que estandarizó las definiciones de vista e introdujo el concepto de punto de vista que, tal como se publicó en su artículo – Ver Dropbox

Vistas 4+1

- El enfoque de vista 4 + 1 es una descripción arquitectónica
- Organiza las representaciones de la arquitectura de una aplicación en vistas
- Satisface las necesidades de interesados individuales.



Vistas 4+1 – Vista Lógica (Object Oriented Decomposition)

- Realizar la funcionalidad de una aplicación en términos de elementos estructurales, abstracciones y mecanismos clave, separación de preocupaciones y distribución de responsabilidades.
- Los arquitectos utilizan esta vista para el análisis funcional.
- Representada en diferentes niveles de abstracción y evoluciona progresivamente en iteraciones.

Vistas 4+1 – Vista Lógica (Object Oriented Decomposition)

- Representada en diferentes niveles de abstracción que evoluciona progresivamente en iteraciones.

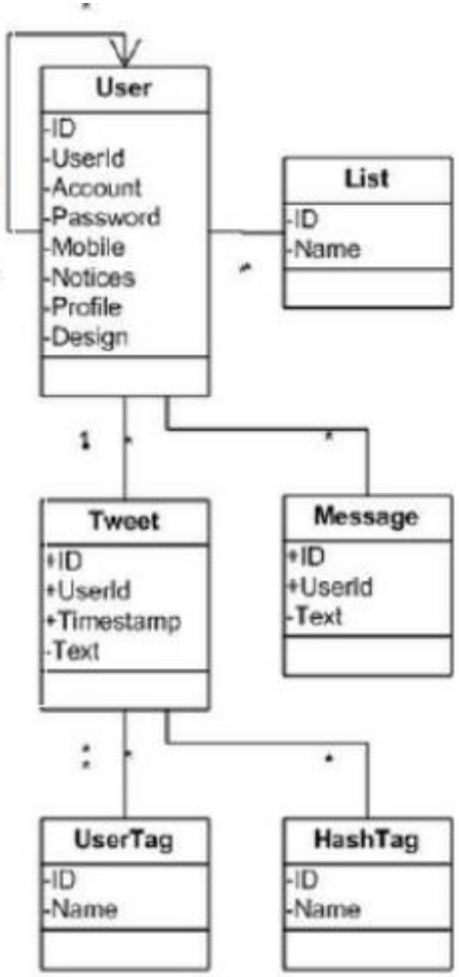
1. Divisiones verticales y horizontales de una aplicación
   - Se puede dividir verticalmente en áreas funcionales significativas (es decir, subsistemas de captura de ordenes, subsistemas de procesamiento de ordenes).
   - O bien, puede dividirse horizontalmente en una arquitectura en capas que distribuye las responsabilidades entre estas capas (es decir, capas de presentación, capas de servicios, capas de lógica de negocios y capas de acceso a datos).
2. Representación de elementos estructurales como clases u objetos y sus relaciones.

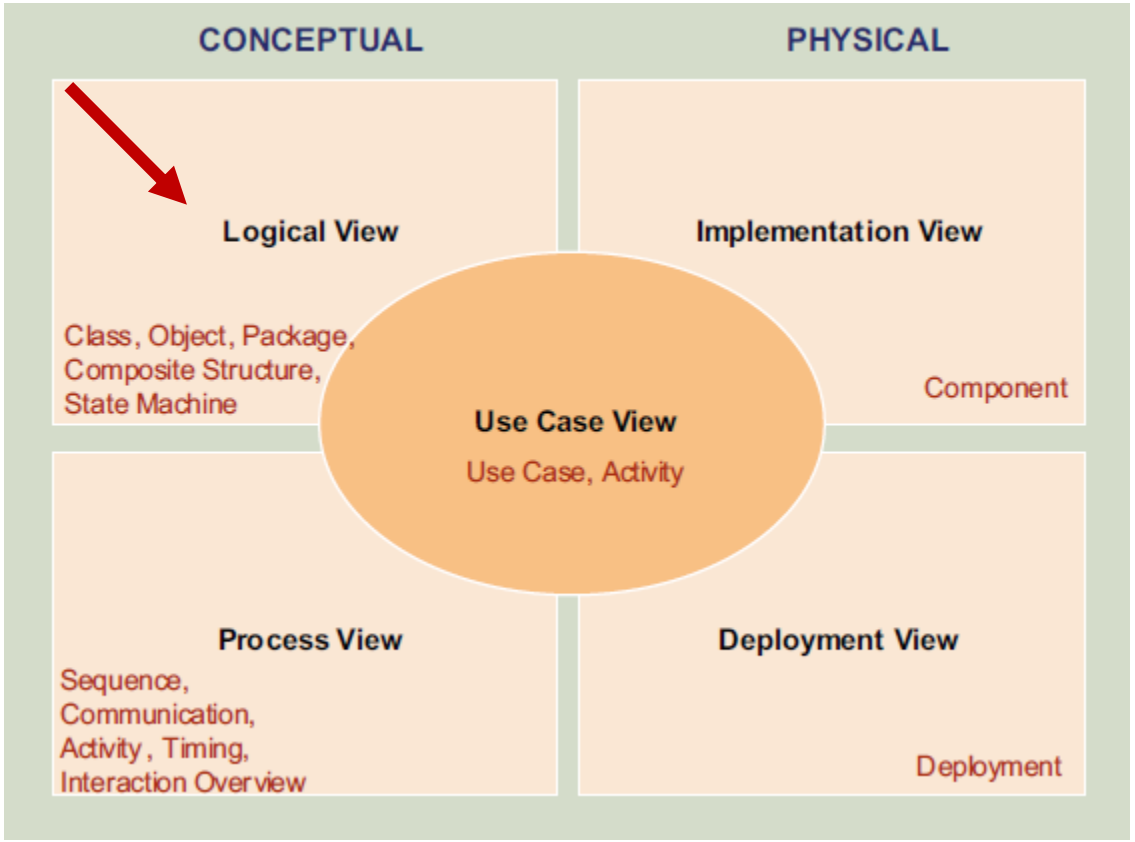
Vistas 4+1 – Vista Lógica (Object Oriented Decomposition)

1. **Diagramas de clase / diagramas estructurales**: definen los bloques de construcción básicos de un modelo. Se enfocan en cada clase individual, operaciones principales y las relaciones con las asociaciones, el uso, la composición, la herencia, etc. de otras clases.
2. **Diagramas de objetos:** muestran cómo se relacionan las instancias de los elementos estructurales. Ayudan a entender los diagramas de clase cuando las relaciones son complejas.
3. **Diagramas de paquetes**: se utilizan para dividir el modelo en contenedores lógicos o 'paquetes'. Se pueden utilizar para representar divisiones verticales y horizontales como paquetes.
4. **Diagramas de estructura compuesta**: ayudan a modelar las partes que contiene una clase y las relaciones entre las partes.
5. **Diagramas de máquina de estado**: necesarios para comprender los estados instantáneos de un objeto definido por una clase. Se usan opcionalmente cuando es necesario comprender los posibles estados de una clase.

Vistas 4+1 – Vista Lógica (Object Oriented Decomposition)

Twitter





Vistas 4+1 – Vista de Procesos (Process Decomposition)

- Considera aspectos no funcionales, como la escalabilidad y el rendimiento.
- Aborda los problemas de concurrencia, distribución y tolerancia a fallos.
- Muestra las abstracciones principales de la Vista lógica ejecutándose sobre un hilo como una operación.
- Un proceso es un grupo de tareas que forman una unidad ejecutable; un sistema de software se divide en conjuntos de tareas.
- Cada tarea es un hilo de control que se ejecuta con la colaboración entre diferentes elementos estructurales (desde la Vista lógica).
- La vista de proceso también abarca patrones de interacción reutilizables para resolver problemas recurrentes y cumplir con niveles de servicio no funcionales.

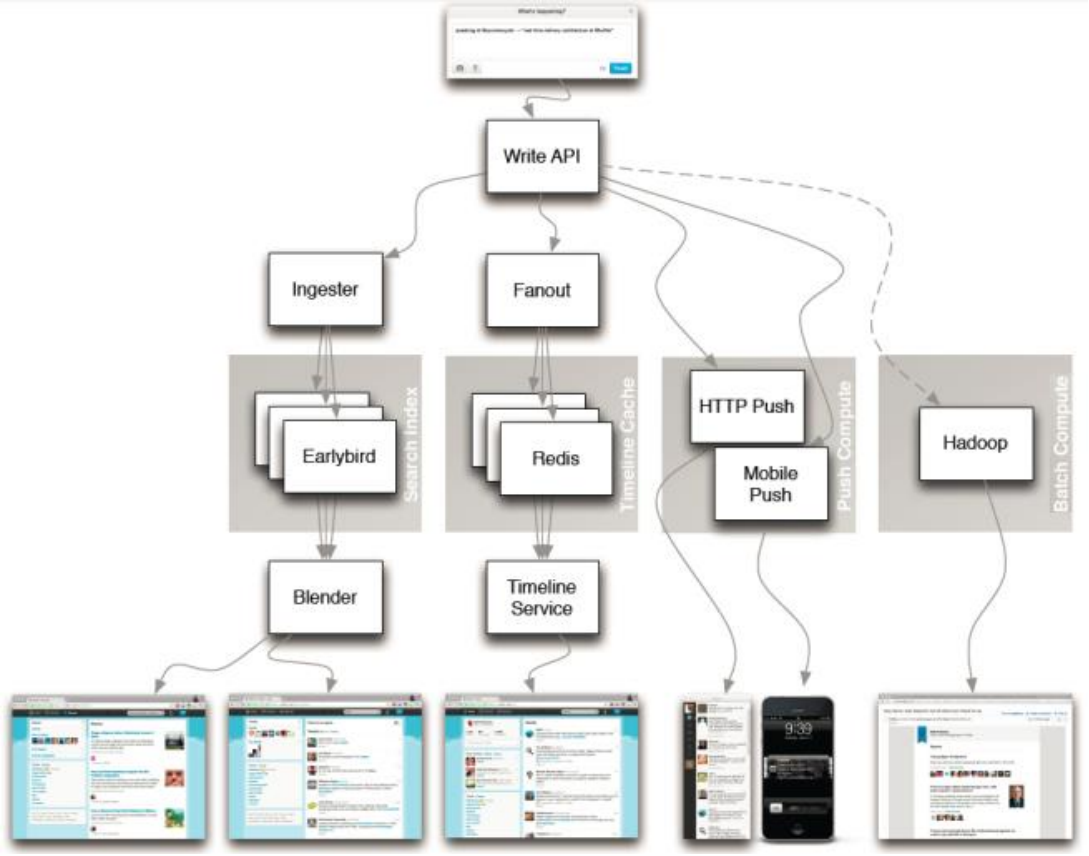
Vistas 4+1 – Vista de Procesos (Process Decomposition)

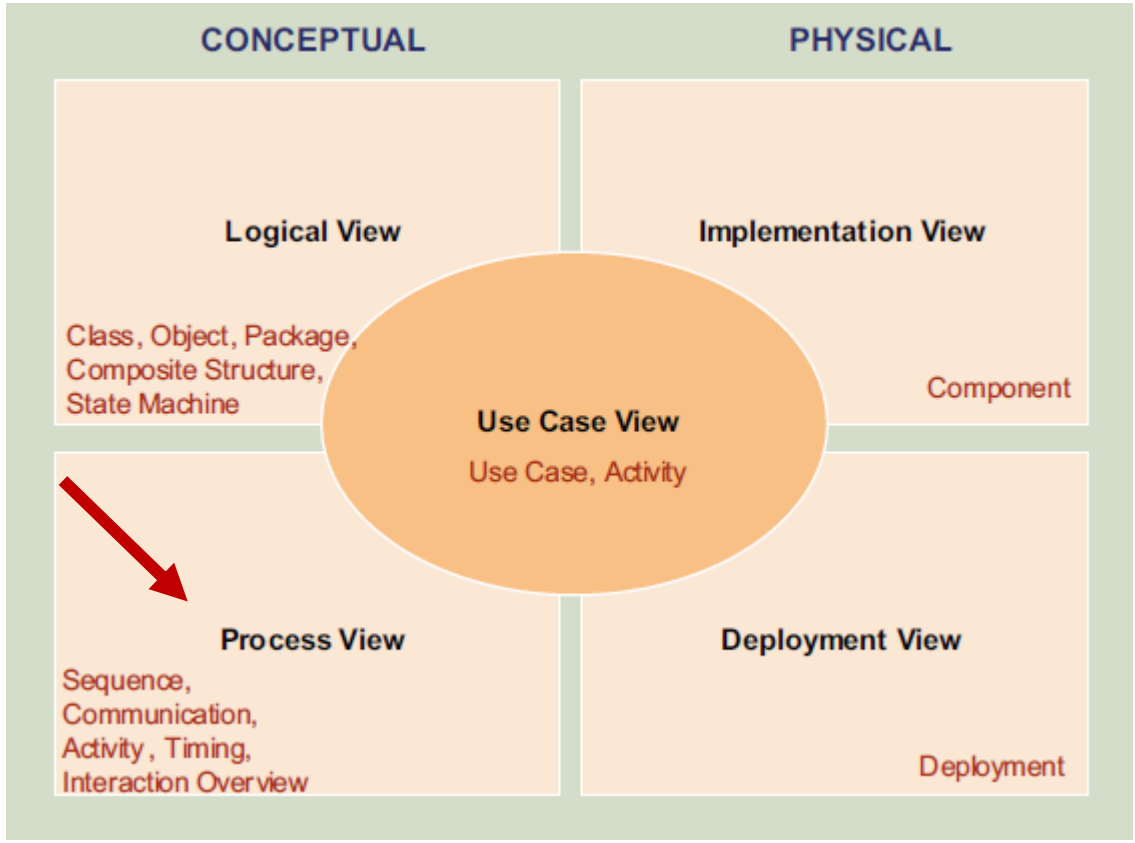
La vista de procesos se puede representar en varios niveles de abstracción, como las interacciones entre sistemas, subsistemas y objetos, según la necesidad.

La vista de proceso se puede representar mediante los siguientes diagramas:

1. **Diagramas de secuencia**: muestran la secuencia de mensajes pasados entre los objetos en una línea de tiempo vertical.
2. **Diagramas de comunicación:** muestran las comunicaciones entre objetos en tiempo de ejecución durante una instancia de colaboración. Estos diagramas están estrechamente relacionados con los diagramas de secuencia.
3. **Diagramas de actividad:** son similares a los diagramas de flujo y tienen una amplia gama de usos en diferentes puntos de vista. En la vista de proceso, se pueden usar para representar los flujos del programa y la lógica del negocio con acciones, puntos de decisión, ramificación
4. **Diagramas de temporización:** abordan específicamente el modelado para el rendimiento. Representan la cantidad de tiempo permitido para que un participante reciba eventos y cambie de estado, y durante cuánto tiempo un participante puede permanecer en un estado específico.
5. **Diagramas de interacción:** estos diagramas proporcionan una descripción general de cómo varias interacciones trabajan juntas para implementar un problema del sistema. Son una fusión de diagramas de actividad, secuencia y temporización.

Vistas 4+1 – Vista de Procesos (Process Decomposition)

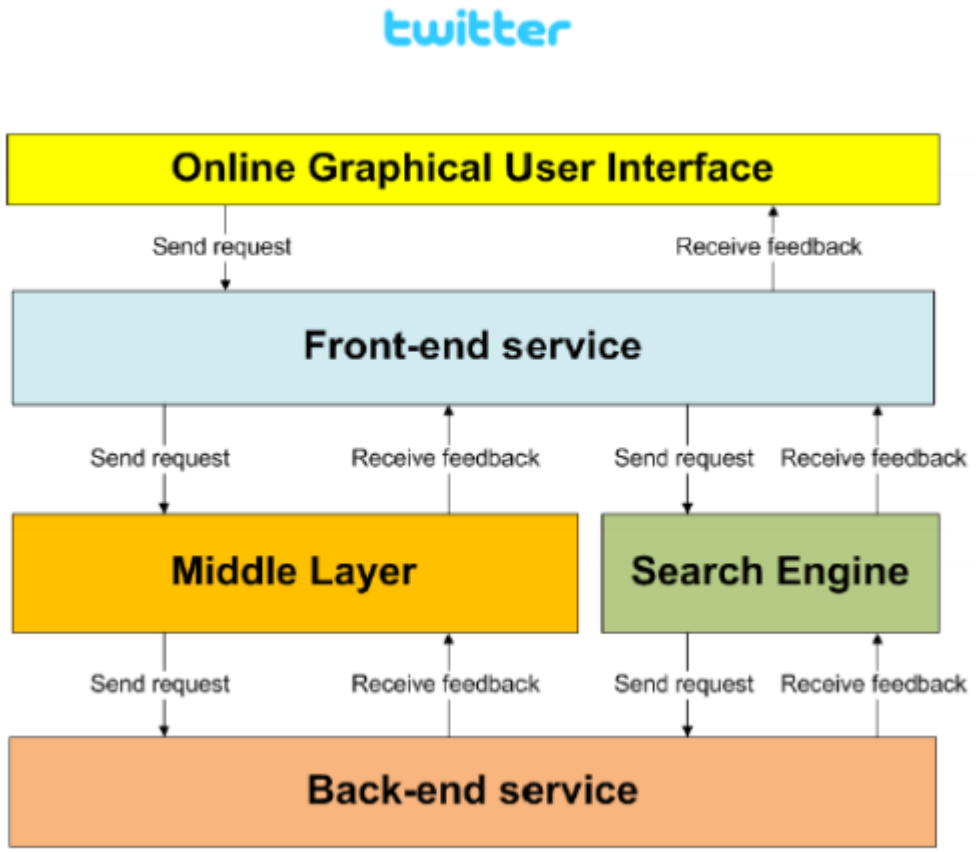




Vistas 4+1 – Vista de Implementación o Desarrollo (Subsystem Decomposition)

- Abarca los componentes utilizados para ensamblar y liberar un sistema físico.
- Se centra en la gestión de la configuración y la organización del módulo de software real en el entorno de desarrollo.
- El software está realmente empaquetado en componentes que pueden ser desarrollados y probados por el equipo de desarrollo. Mientras que la vista lógica está en el nivel conceptual, los diagramas en esta vista representan los artefactos de nivel físico que el equipo construye.
- Los diagramas de componentes:
  - se utilizan para representar la vista de implementación.
  - muestran diferentes componentes, los puertos disponibles y las dependencias del entorno en términos de interfaces proporcionadas y requeridas. Los componentes se pueden vincular a las clases y estructuras compuestas que realizan estos componentes.

Vistas 4+1 – Vista de Implementación o Desarrollo (Subsystem Decomposition)



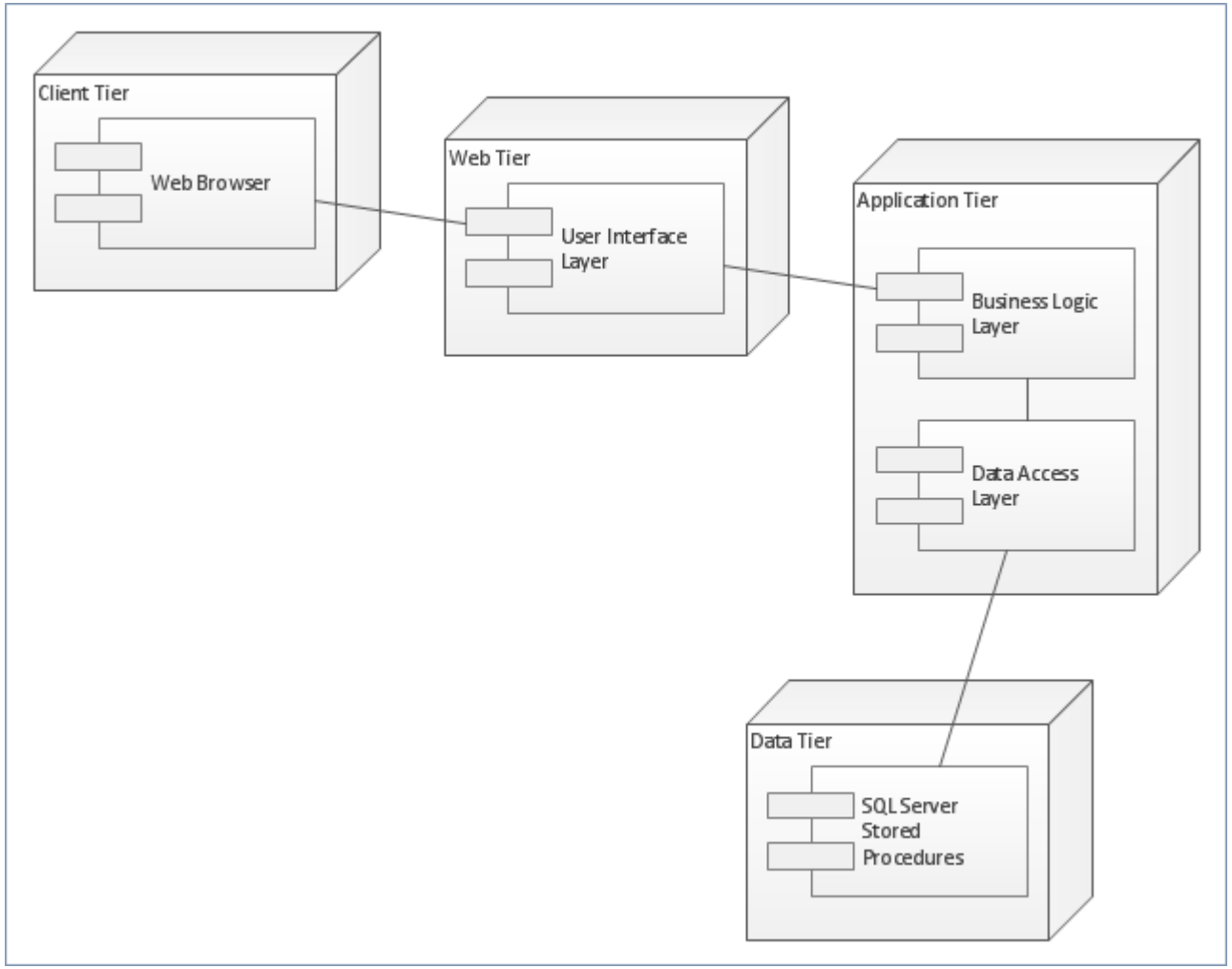
Vistas 4+1 – Vista de Física o de Despliegue (Mapping Software to Hardware)

- Esta vista abarca los nodos que forman la topología de hardware del sistema en la que se ejecuta el sistema; se centra en la distribución, comunicación y aprovisionamiento.
- El software se ejecuta en una red de computadoras, o de procesamiento de nodos. Los diversos elementos, como los procesos, las tareas y los objetos, deben asignarse a los nodos en los que se ejecutan.
- Estas configuraciones físicas pueden diferir entre los entornos de producción, desarrollo y prueba. El software debe estar diseñado para ser flexible para escalar a través de estos cambios de hardware.

Vistas 4+1 – Vista de Física o de Despliegue (Mapping Software to Hardware)

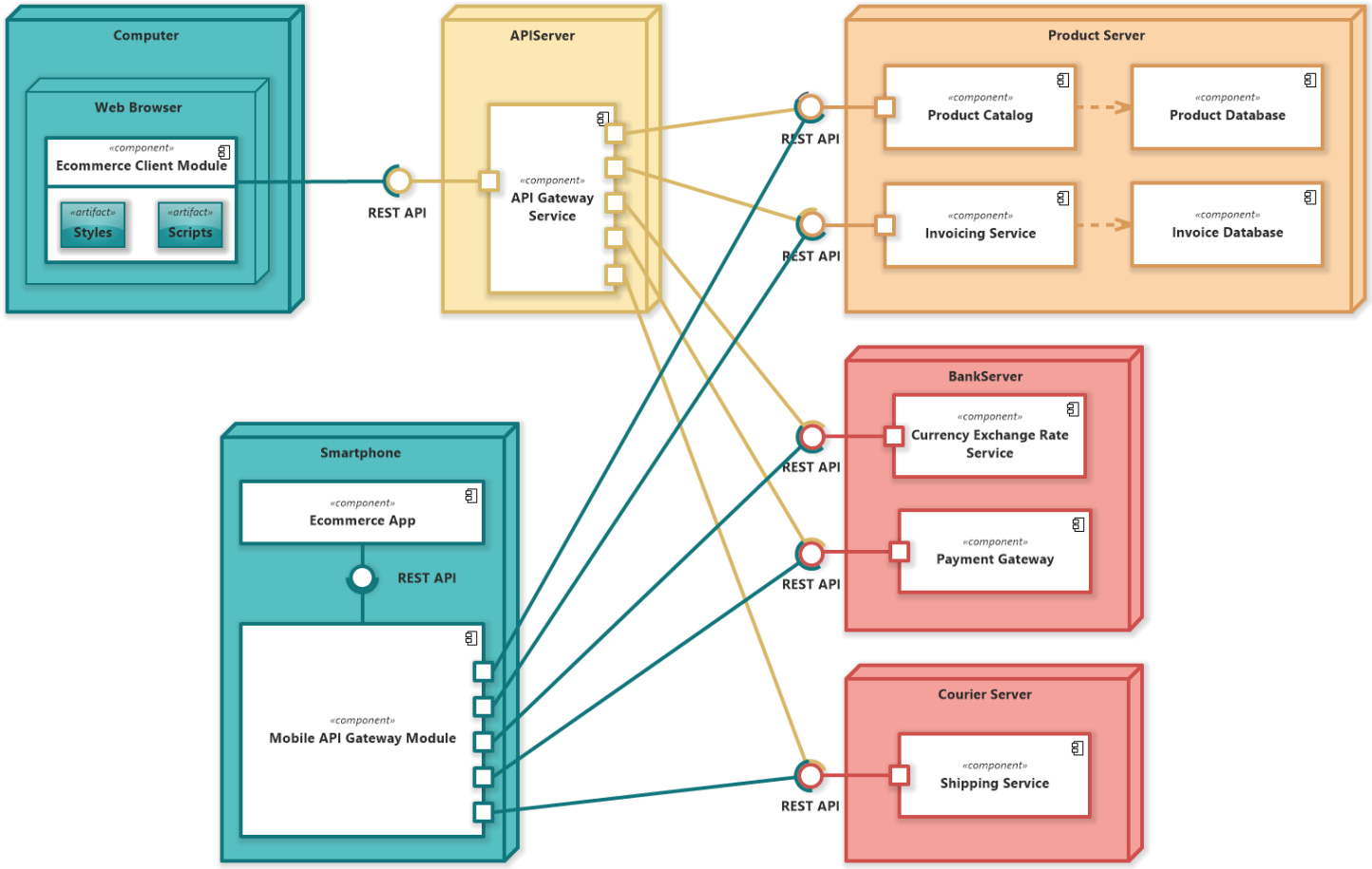
- Esta vista se adapta a los requisitos no funcionales, tales como disponibilidad, confiabilidad, rendimiento, rendimiento y escalabilidad.
- Esta vista proporciona todas las configuraciones de hardware posibles y asigna los componentes de la Vista de implementación a estas configuraciones.
- Los diagramas de implementación muestran la disposición física de los artefactos en la configuración del mundo real.

Vistas 4+1 – Vista de Física o de Despliegue (Mapping Software to Hardware)



https://conceptdraw.com/a661c3/preview--UML%20deployment%20diagram%20template

Vistas 4+1 – Vista de Física o de Despliegue (Mapping Software to Hardware)

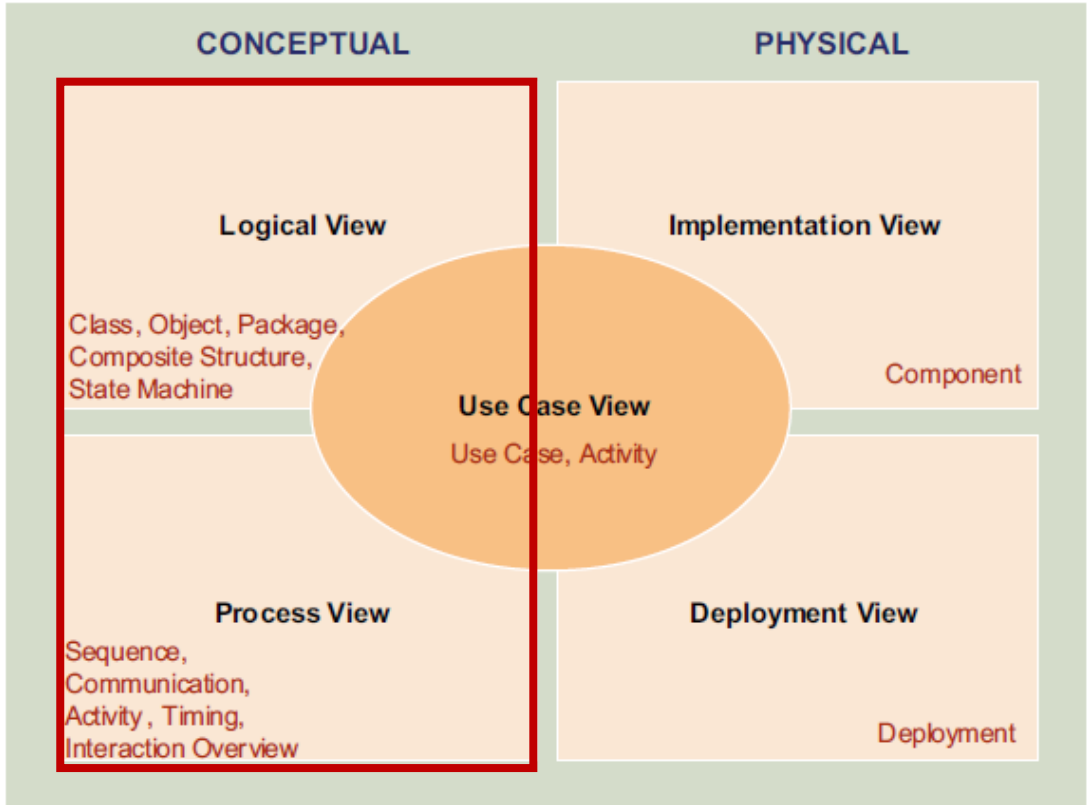


Vistas 4 + 1 – Vista de Casos de Uso (Mapping Software to Hardware)

- Además de las cuatro vistas explicadas anteriormente, esta es la vista central para capturar escenarios.
- La vista de casos de uso abarca los casos de uso que describen el comportamiento del sistema como lo ven sus usuarios finales y otras partes interesadas.
- Aunque tradicionalmente se analiza como la última vista, esta es la primera vista creada en el ciclo de vida del desarrollo del sistema.
- Esta vista representa los escenarios que unen las cuatro vistas y constituye la razón por la que existen todas las demás vistas. Con todas las demás vistas en su lugar, esta vista parece redundante (por lo tanto, +1).
- Sin embargo, representa los requisitos arquitectónicamente significativos en forma de escenarios. También ayuda a verificar que se cumplan todos los escenarios requeridos.

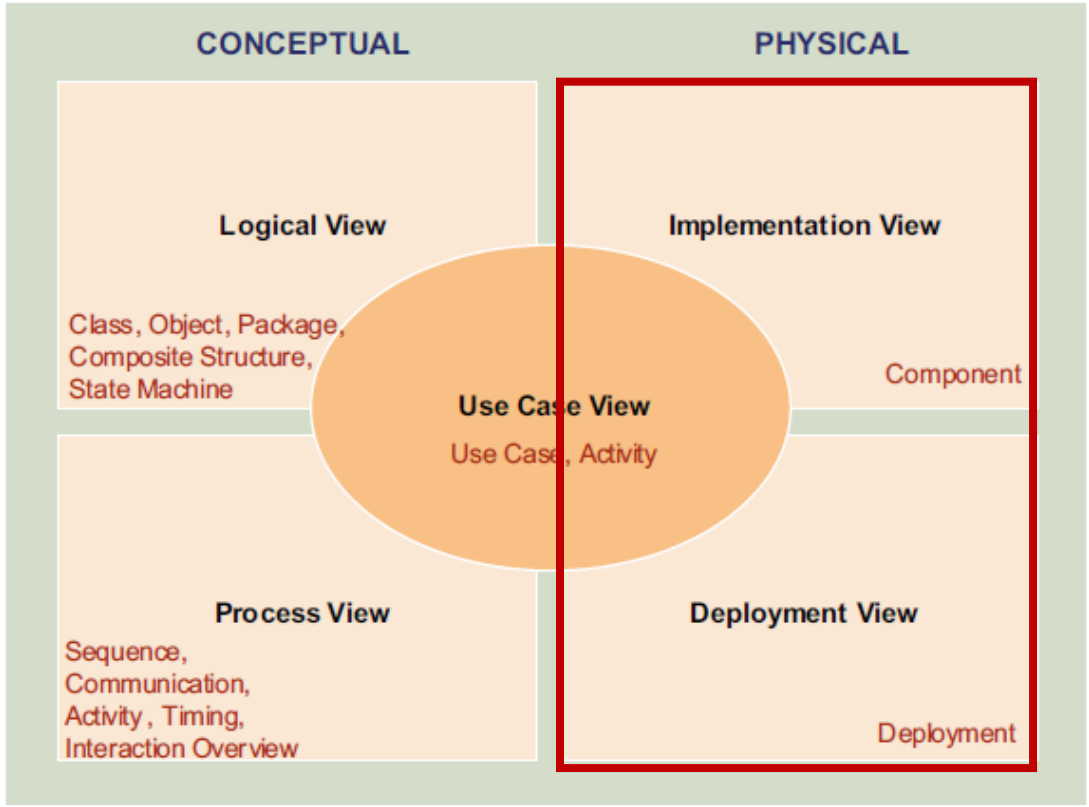
Vistas 4+1 – Relación entre vistas

- La vista lógica y la vista de proceso se encuentran en un nivel conceptual y se utilizan desde el análisis hasta el diseño.



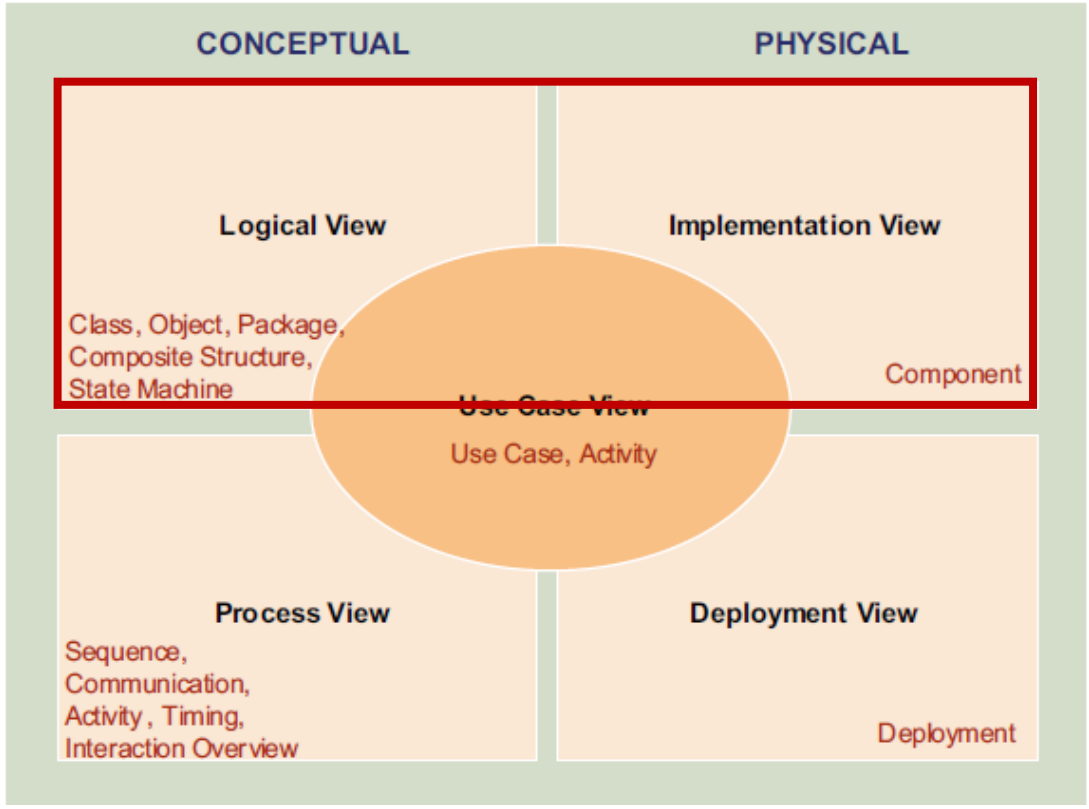
Vistas 4+1 – Relación entre vistas

- La vista de implementación y la vista de despliegue se encuentran en el nivel físico y representan los componentes reales de la aplicación creados y desplegados.



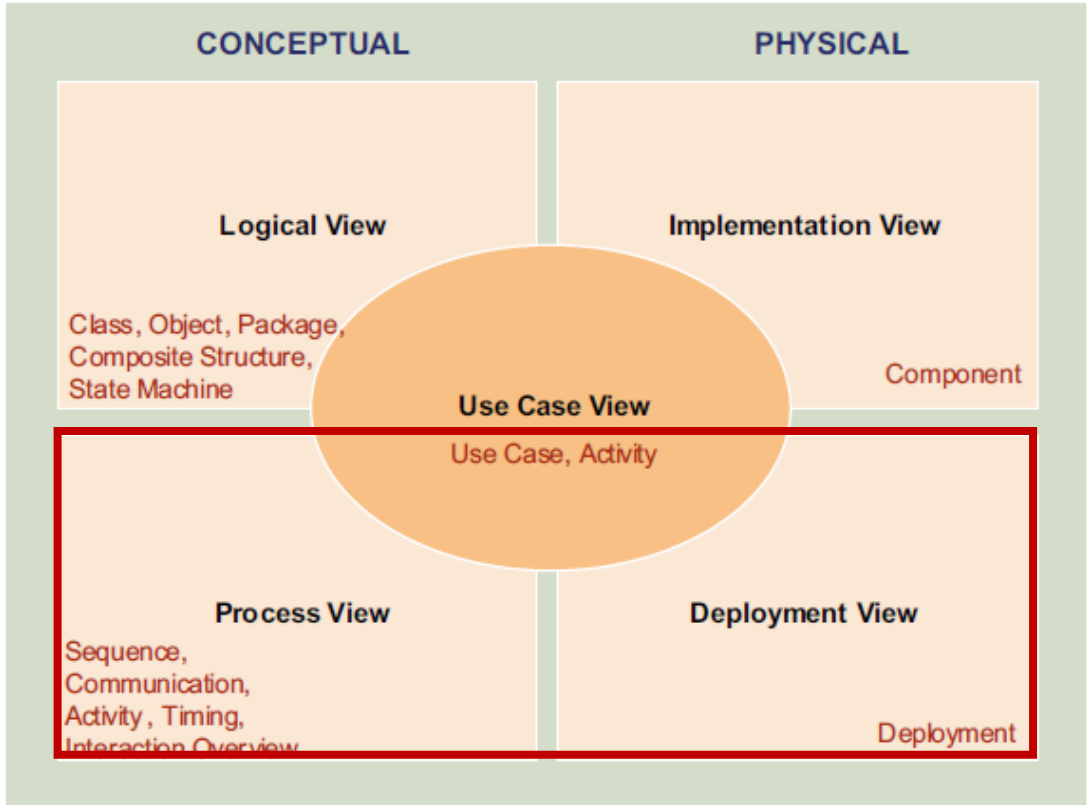
Vistas 4+1 – Relación entre vistas

- La Vista lógica y la Vista de implementación están más vinculadas a la funcionalidad, representan cómo se modela e implementa la funcionalidad.



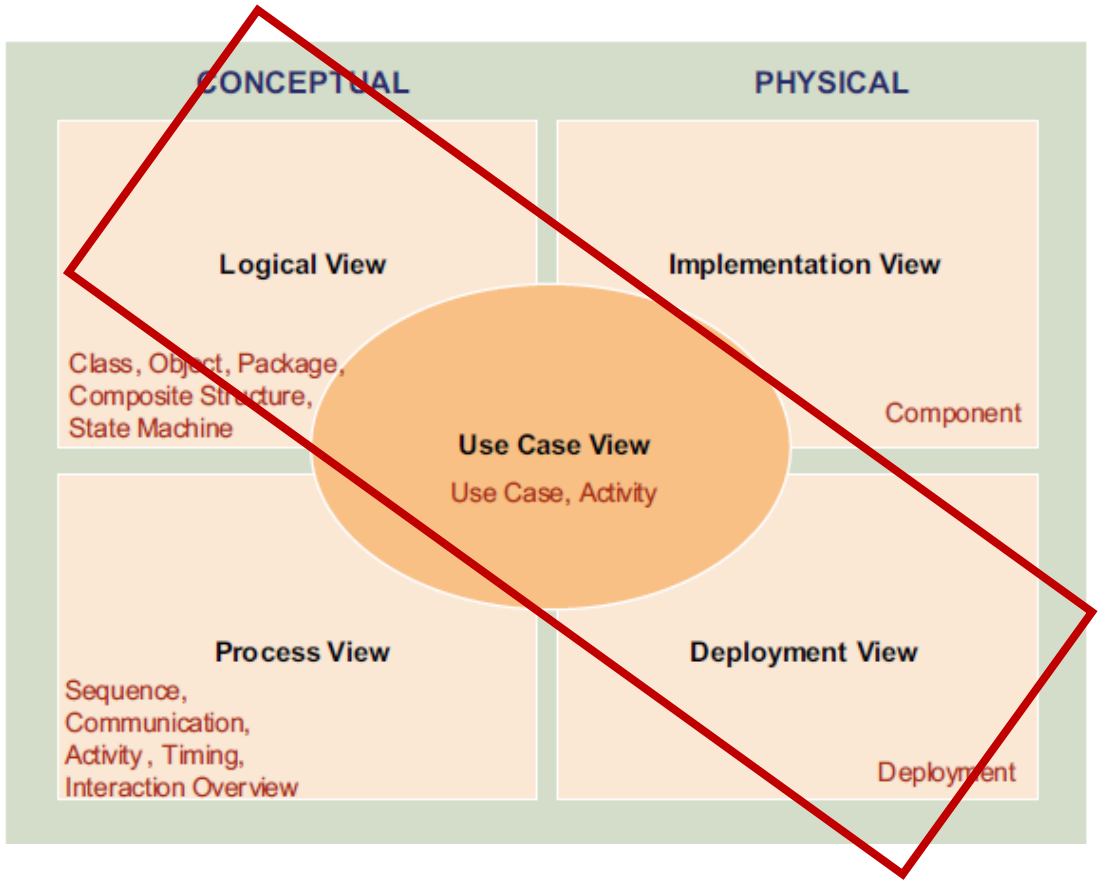
Vistas 4+1 – Relación entre vistas

- La vista de proceso y la vista de despliegue describen los aspectos no funcionales mediante el modelado físico y de comportamiento.



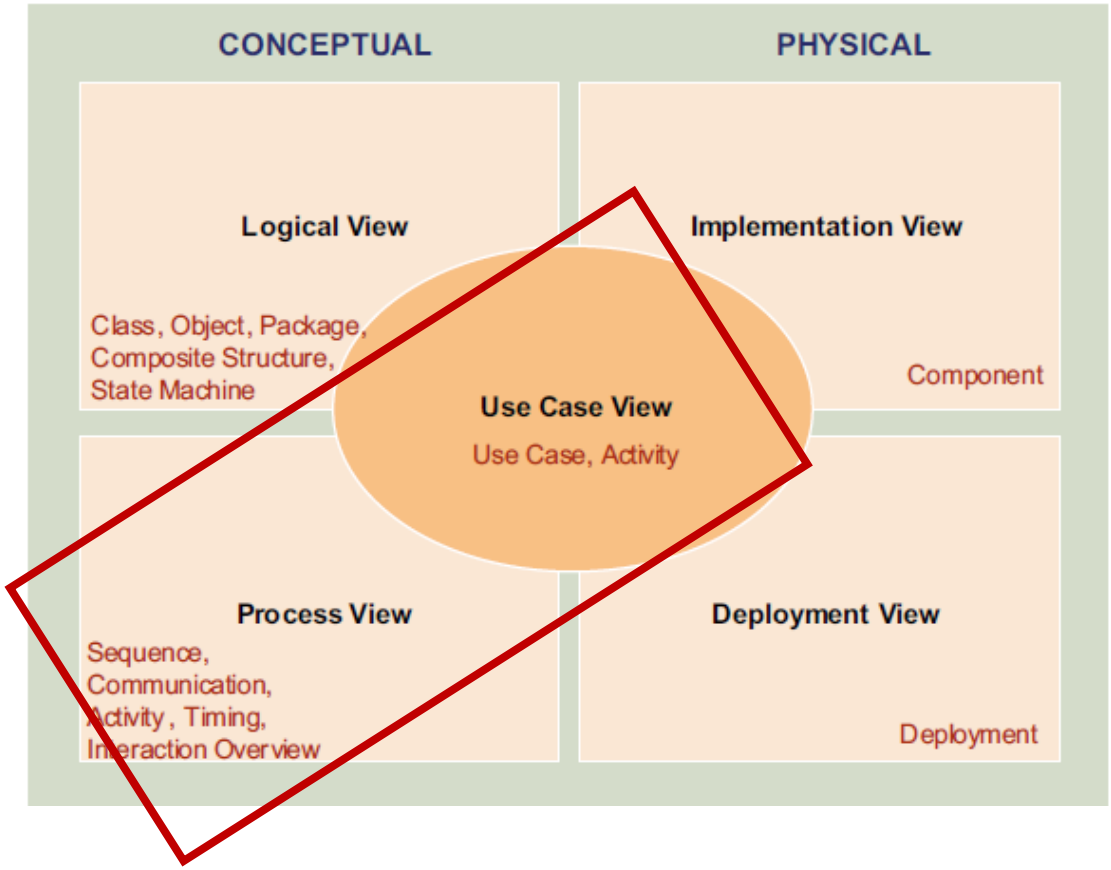
Vistas 4+1 – Relación entre vistas

- La vista de casos de uso describe elementos estructurales que se analizan en la vista lógica y se implementan en la vista de desarrollo.



Vistas 4+1 – Relación entre vistas

- Los escenarios en la Vista de casos de uso se realizan en la Vista de proceso y se implementan en la Vista física.



TRABAJO EN GRUPO

- Analizar el caso propuesto
- Definir una estrategia de desarrollo
  - Identificar fases
  - Identificar arquitectura propuesta.

Clase Práctica

- De acuerdo al caso entregado, desarrolle un esquema de descripción 4 vistas + 1