

Arquitectura de Software

Estilos de Arquitecturas
Parte I

- “Architects need to think about their software in three ways simultaneously:
 - How it is structured as a set of implementation units
 - How it is structured as a set of elements that have runtime behavior and interactions
 - How it relates to nonsoftware structures in its environment”

“Documenting software Architectures” pag 18

- Las vistas arquitecturales están clasificadas en tres tipos (viewtypes):
 - Module
 - Component and Connector
 - Allocation

□ Module

- Las unidades de descomposición son módulos
- Unidades de código que implementan un conjunto de responsabilidades
 - Clase, Colección de clases, Capas
- La relación “sub modulo” está presente

- Component and Connector
 - Modela los aspectos dinámicos del sistema en ejecución
 - Las unidades son procesos
 - La relación entre los elementos representa la comunicación entre los componentes y los conectores

□ Allocation

- Modela las estrategias de asignación entre elementos
- Los elementos pueden ser software, hardware, o canales de comunicación
- La relación “asignado en” es utilizada entre los elementos

Un **Estilo Arquitectural** expresa una estructura fundamental o un esquema organizacional de un sistema de software [1]

Un **Estilo** define una familia de arquitecturas que satisfacen algunas restricciones [2]

Un estilo define:

- Una familia de sistemas cuya estructura es la misma.
- Un vocabulario para los componentes
- Un vocabulario para los estilos
- Un patrón de arquitectura

- Para cada Estilo Arquitectural es importante tener en cuenta
 - Propósito del estilo
 - Elementos que se pueden utilizar y sus propiedades
 - Relaciones entre los elementos
 - Notaciones apropiadas para dicho estilo (por ejemplo Diagramas UML, etc.)



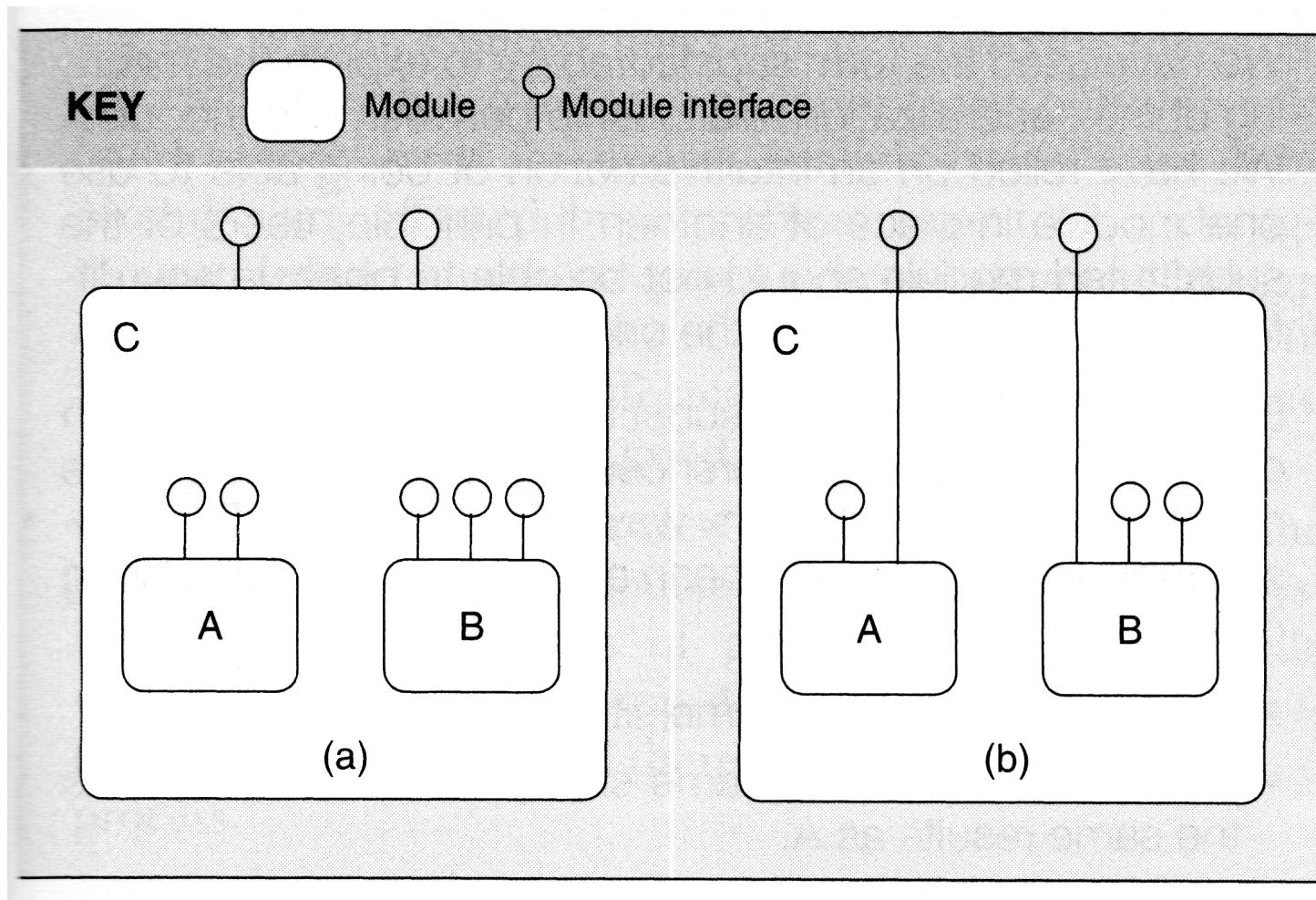
Conceptos Básicos

Module Viewtype

Elementos	Módulo
Relaciones	Is-part-of Is-a Depends-on
Propiedades	Nombre Responsabilidades del módulo
Topología	No tiene restricciones
Usos	<ul style="list-style-type: none">•Provee una maqueta del código fuente•Trazabilidad de requerimientos•Análisis de Impacto•Permite explicar la funcionalidad del sistema a los stakeholders
Notaciones	<ul style="list-style-type: none">•Informales•UML (Clases, paquetes)

Conceptos Básicos

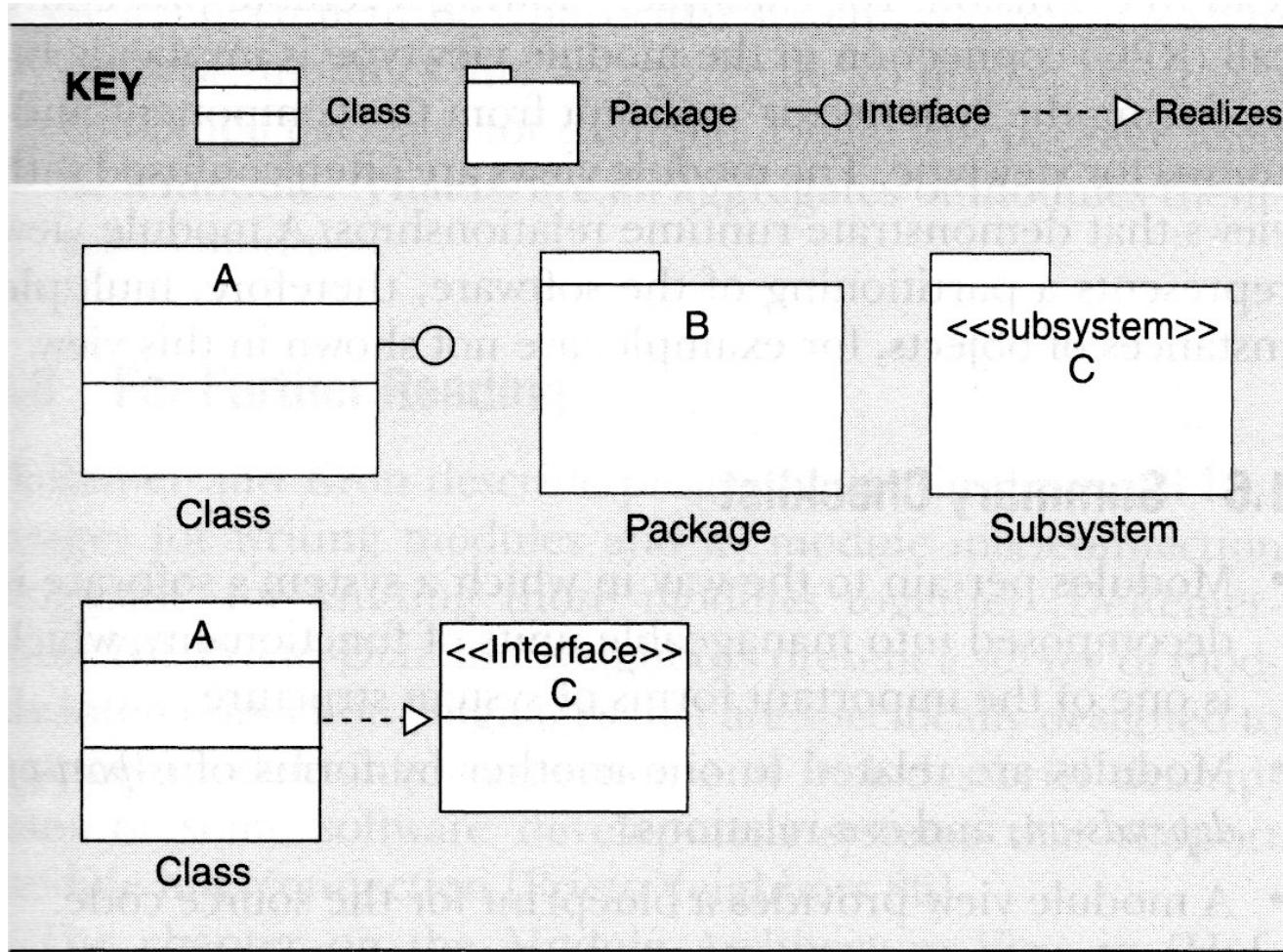
Ejemplos de notaciones para el tipo de vista *Module*



Tomado de “Documenting Software Architectures” pag 45

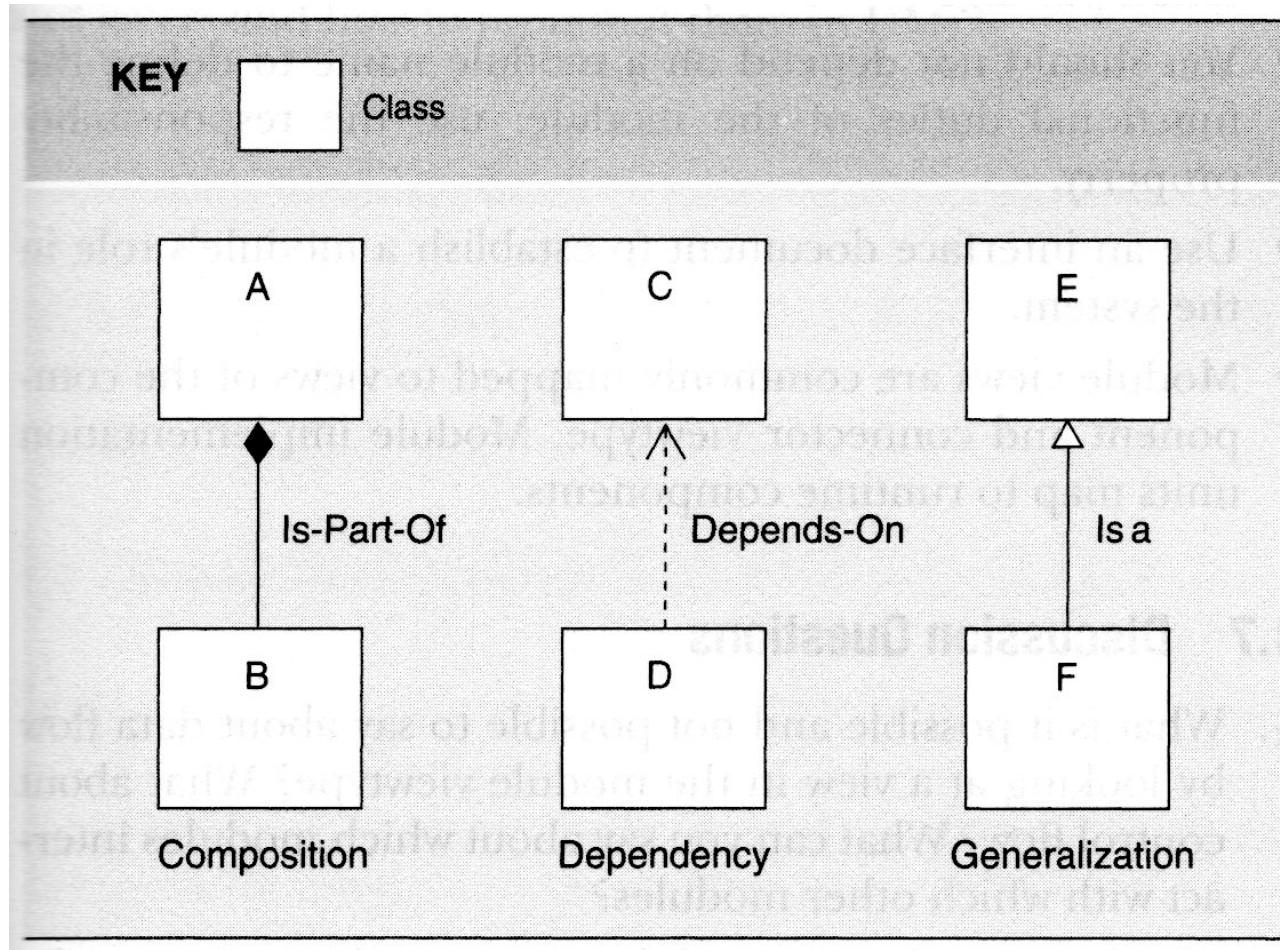
Conceptos Básicos

Ejemplos de notaciones para el tipo de vista *Module*

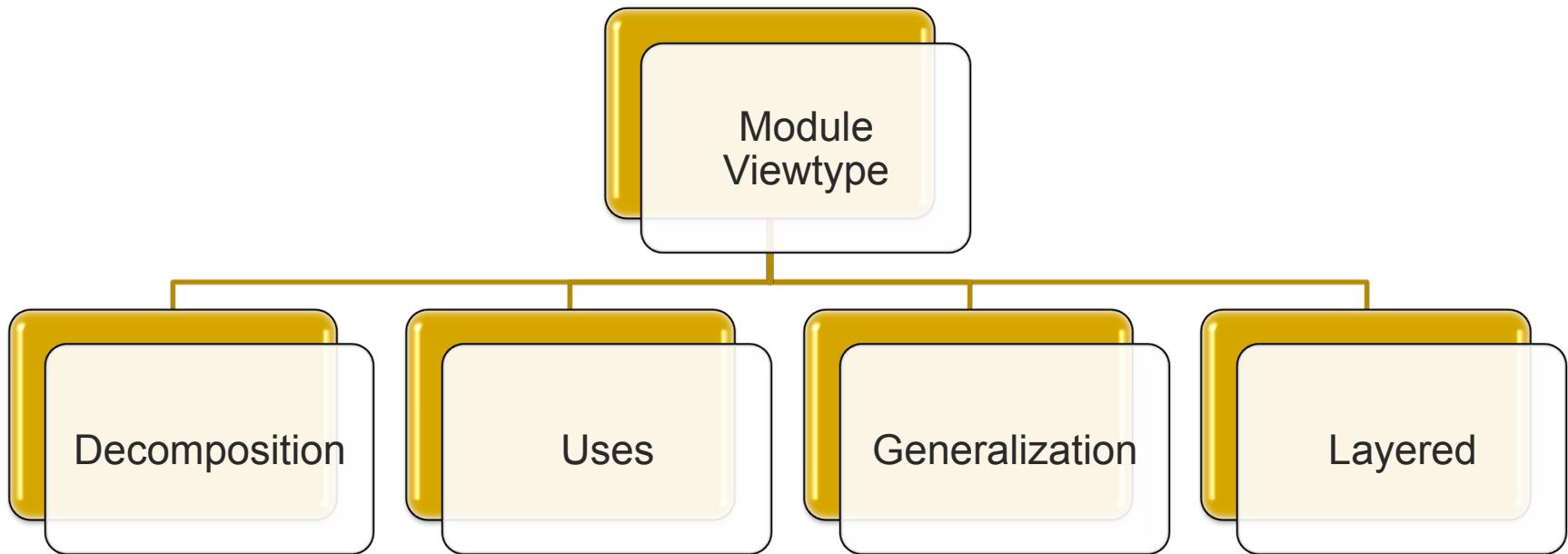


Conceptos Básicos

Ejemplos de notaciones para el tipo de vista *Module*



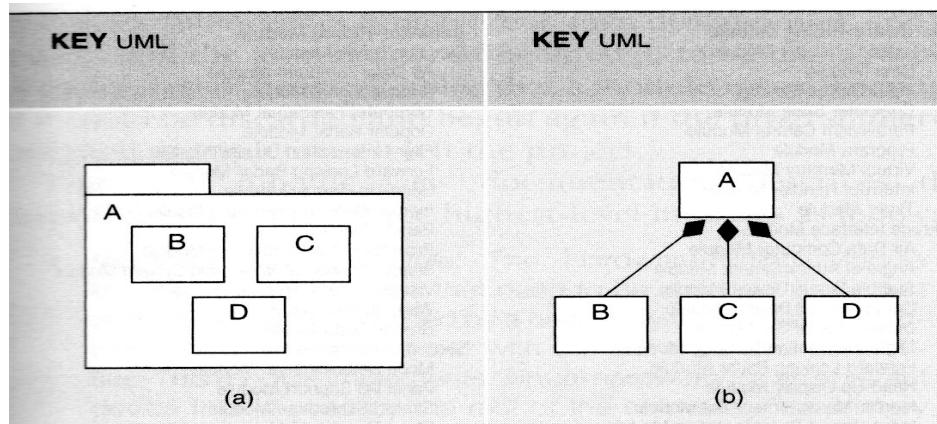
Conceptos Básicos



Conceptos Básicos

Decomposition Style

Elementos	Módulos
Relaciones	is-part-of
Propiedades	
Propiedades de las relaciones	Visibilidad
Topología	No se permiten ciclos Un módulo solo es parte de otro módulo
Notación	UML



Tomado de "Documenting Software Architectures" pag 57

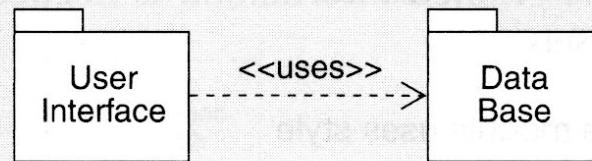
Conceptos Básicos

Uses Style

Elementos	Módulos
Relaciones	depends-on
Propiedades	
Propiedades de las relaciones	
Topología	
Notación	UML

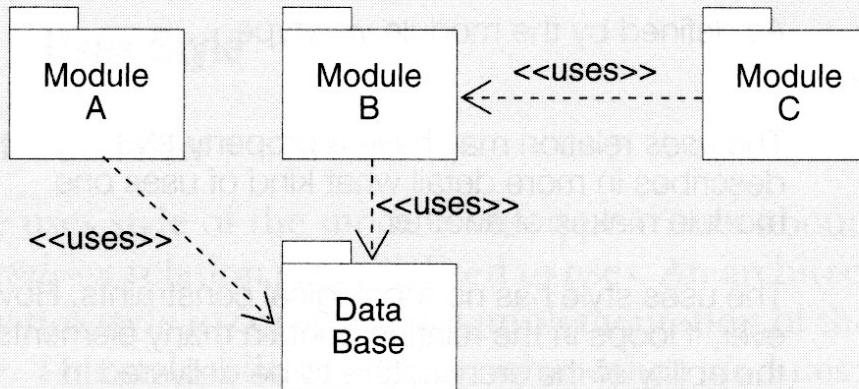
Conceptos Básicos

KEY UML



(a)

KEY UML

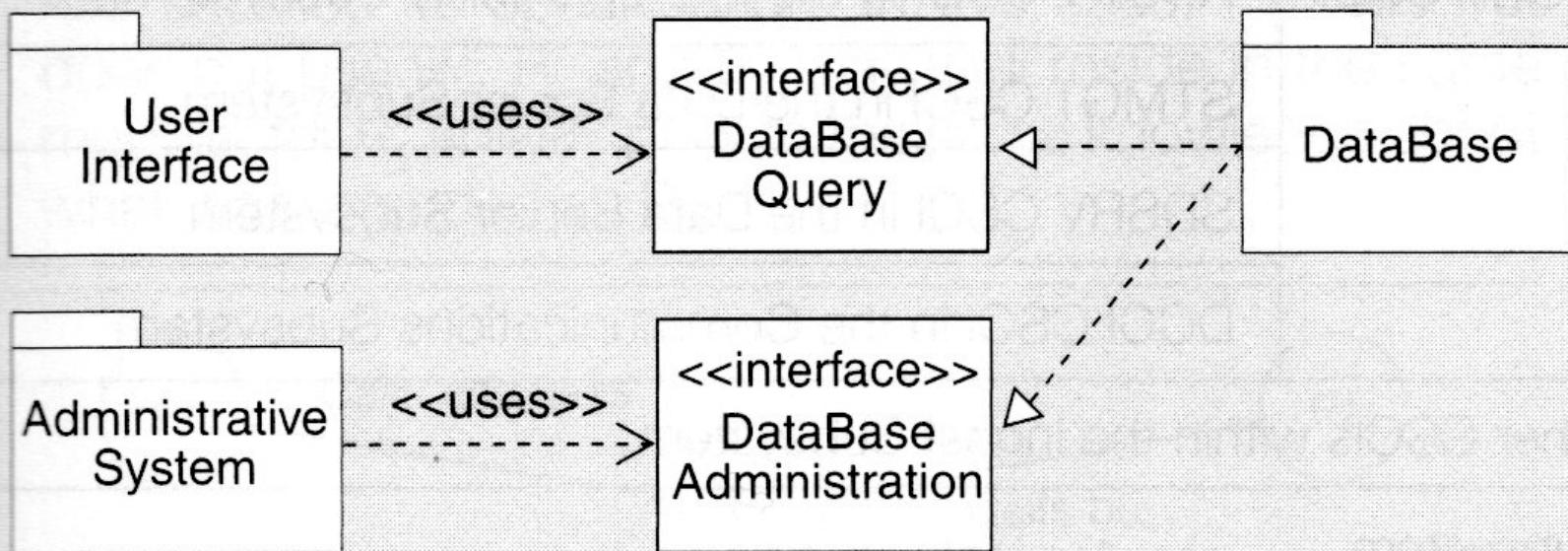


(b)



Conceptos Básicos

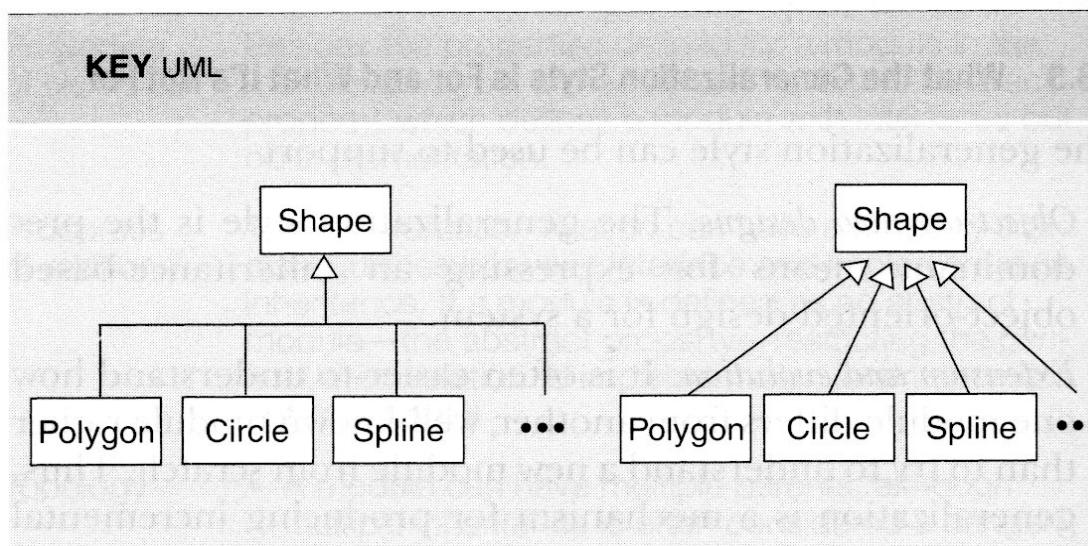
KEY UML



Conceptos Básicos

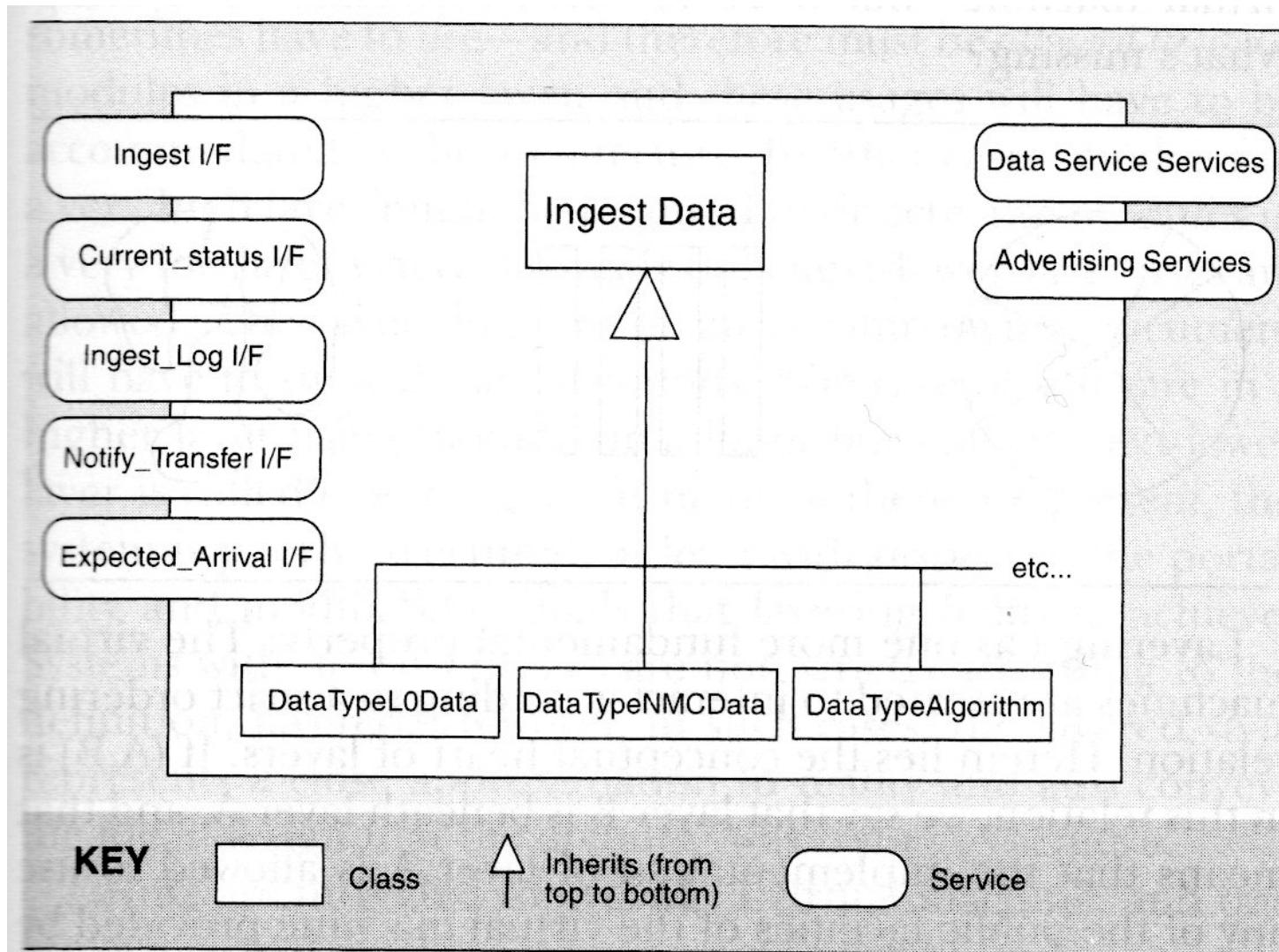
Generalization Style

Elementos	Módulos
Relaciones	Generalization
Propiedades	
Propiedades de las relaciones	Diferenciar entre interfaces e implementación
Topología	Se permite herencia múltiple
Notación	UML



Tomado de "Documenting Software Architectures" pag 74

Conceptos Básicos



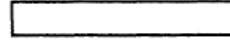
Conceptos Básicos

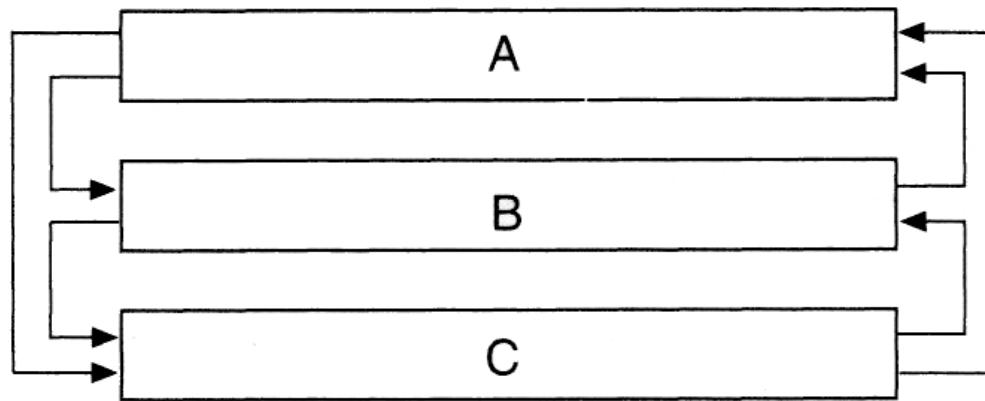
Layered Style

Elementos	Capas
Relaciones	Autorizado para usar
Propiedades	Nombre de la capa
Propiedades de las relaciones	Diferenciar entre interfaces e implementación
Topología	No se permite el intercambio de posiciones en la jerarquía
Notación	Informales Segmentadas Anillos UML (mediante paquetes)

Conceptos Básicos

Ejemplo de notación informal para representar un estilo por capas

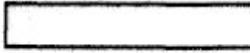
KEY (informal notation)  Layer $x \rightarrow y$ x is allowed to use y

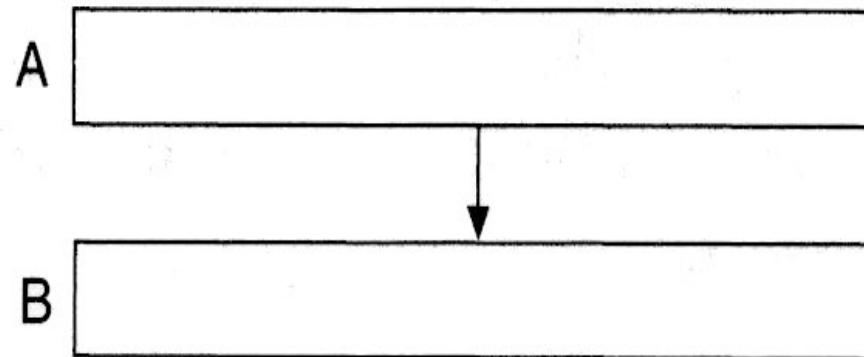


Nota algo raro en este ejemplo?

Conceptos Básicos

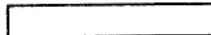
Ejemplo de notación informal para representar un estilo por capas

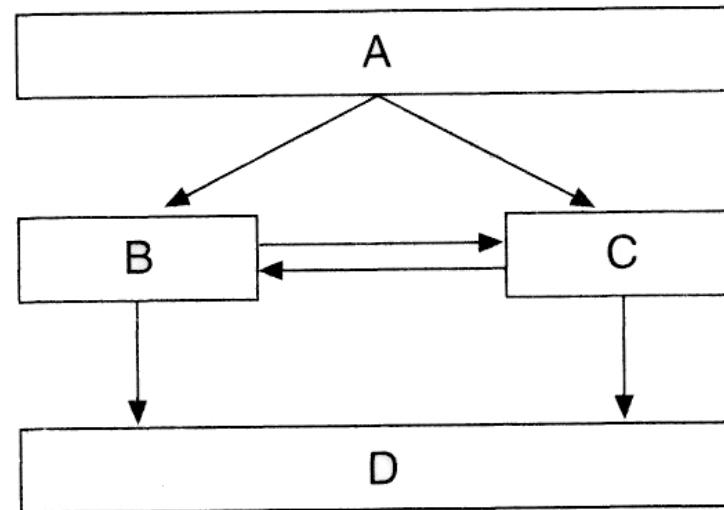
KEY (informal notation)  Layer $x \rightarrow y$ x is allowed to use y



Conceptos Básicos

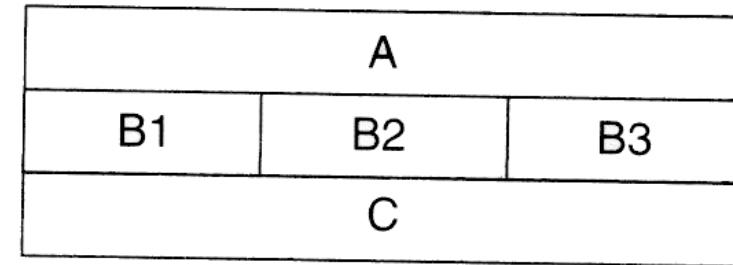
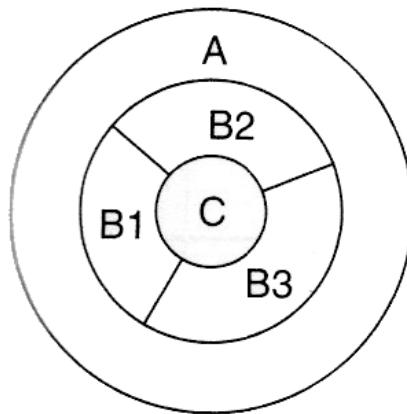
Ejemplo de notación informal para representar un estilo por capas

KEY (informal notation)  Layer $x \rightarrow y$ x is allowed to use y



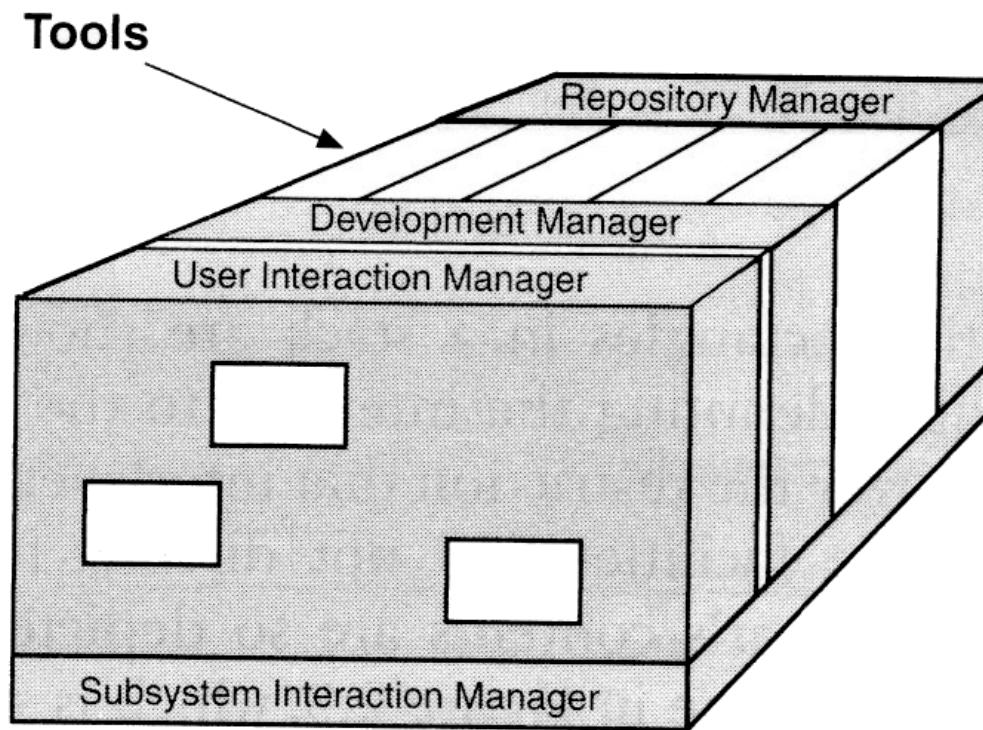
Conceptos Básicos

Ejemplo de notación informal para representar un estilo por capas



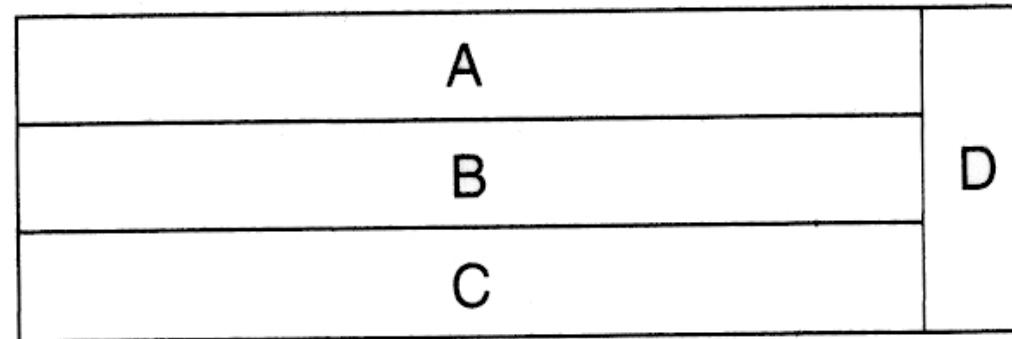
Conceptos Básicos

Ejemplo de notación informal para representar un estilo por capas



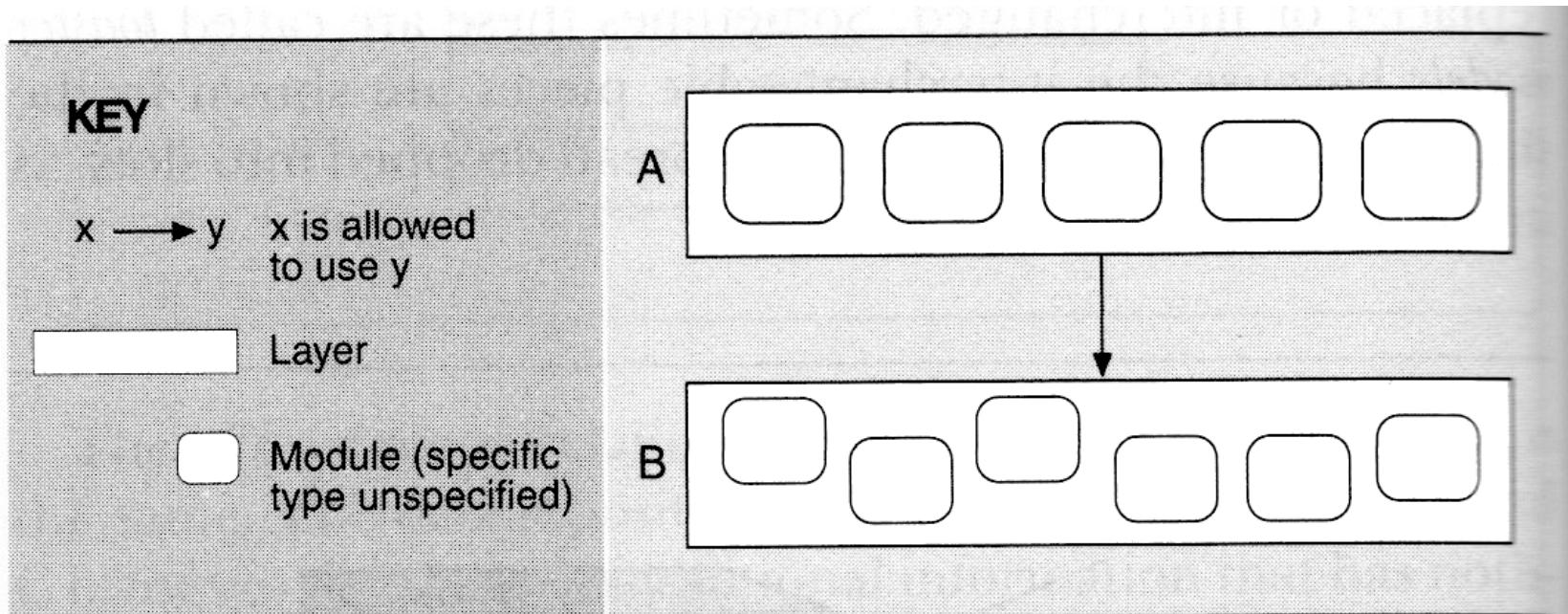
Conceptos Básicos

Ejemplo de notación informal para representar un estilo por capas



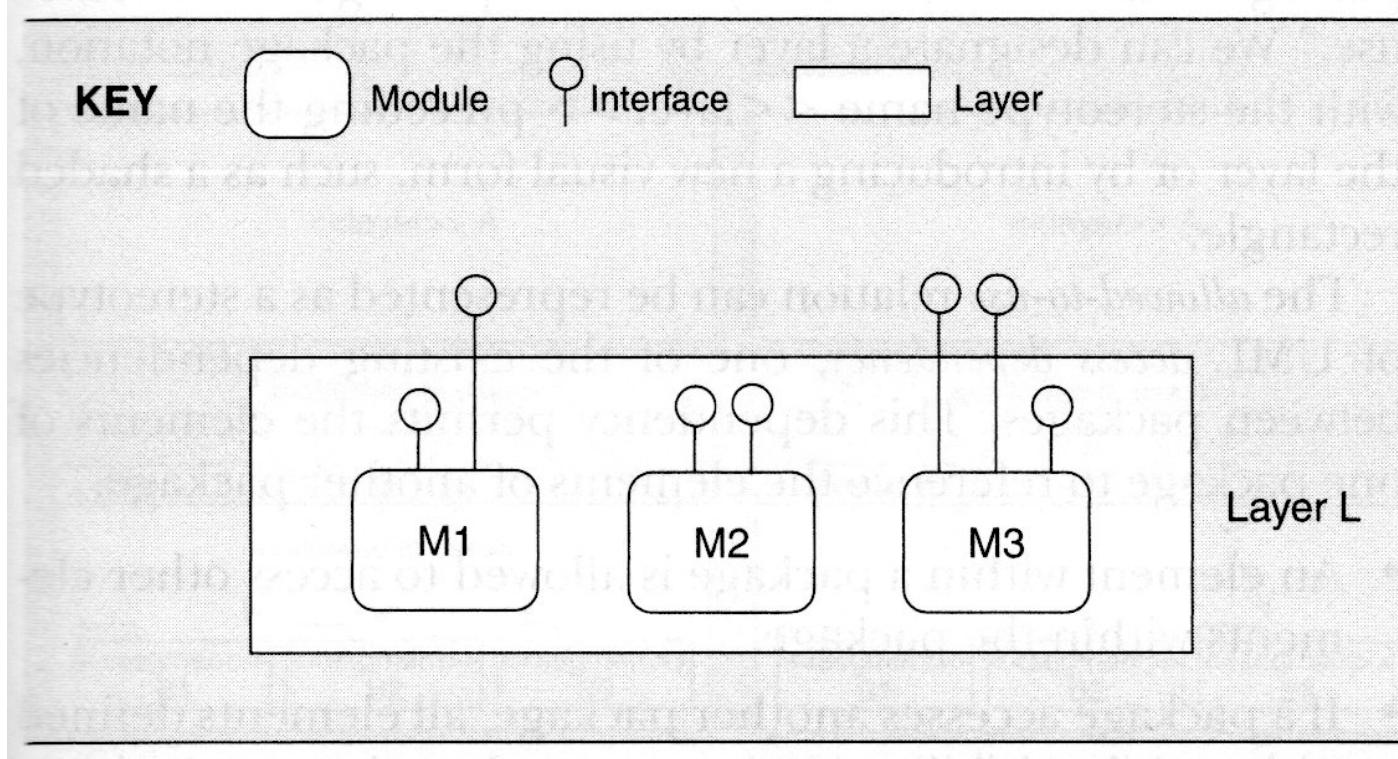
Conceptos Básicos

Ejemplo de notación informal para representar un estilo por capas



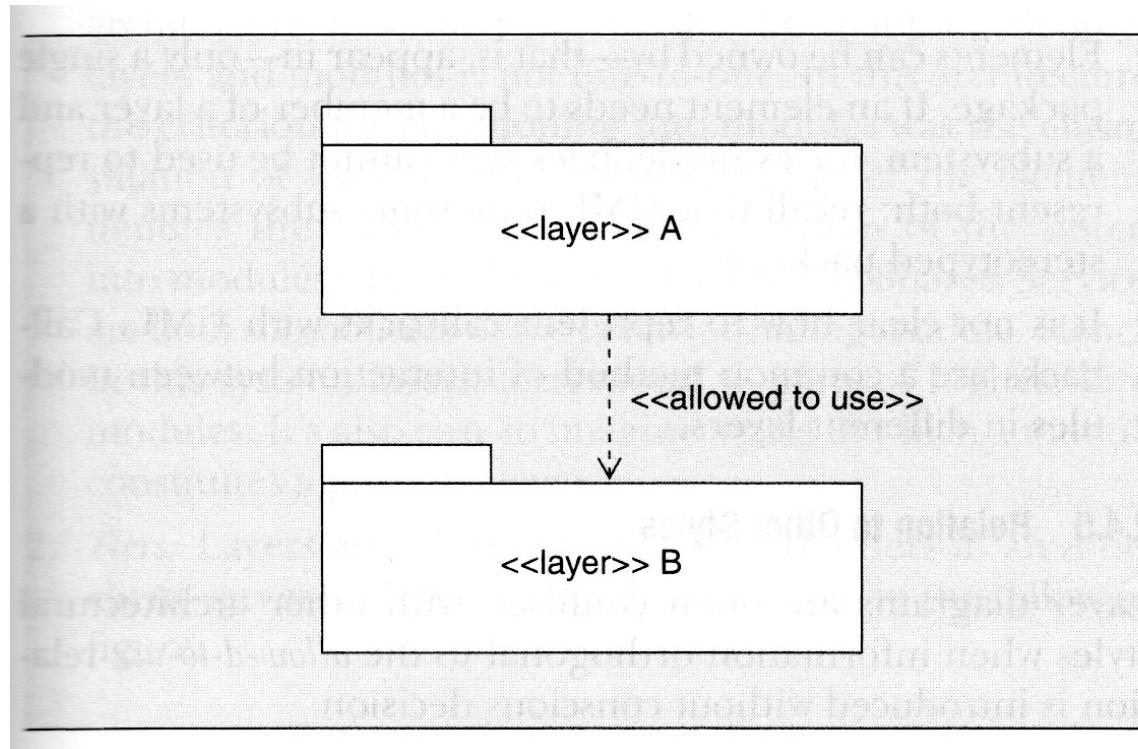
Conceptos Básicos

Ejemplo de notación informal para representar un estilo por capas



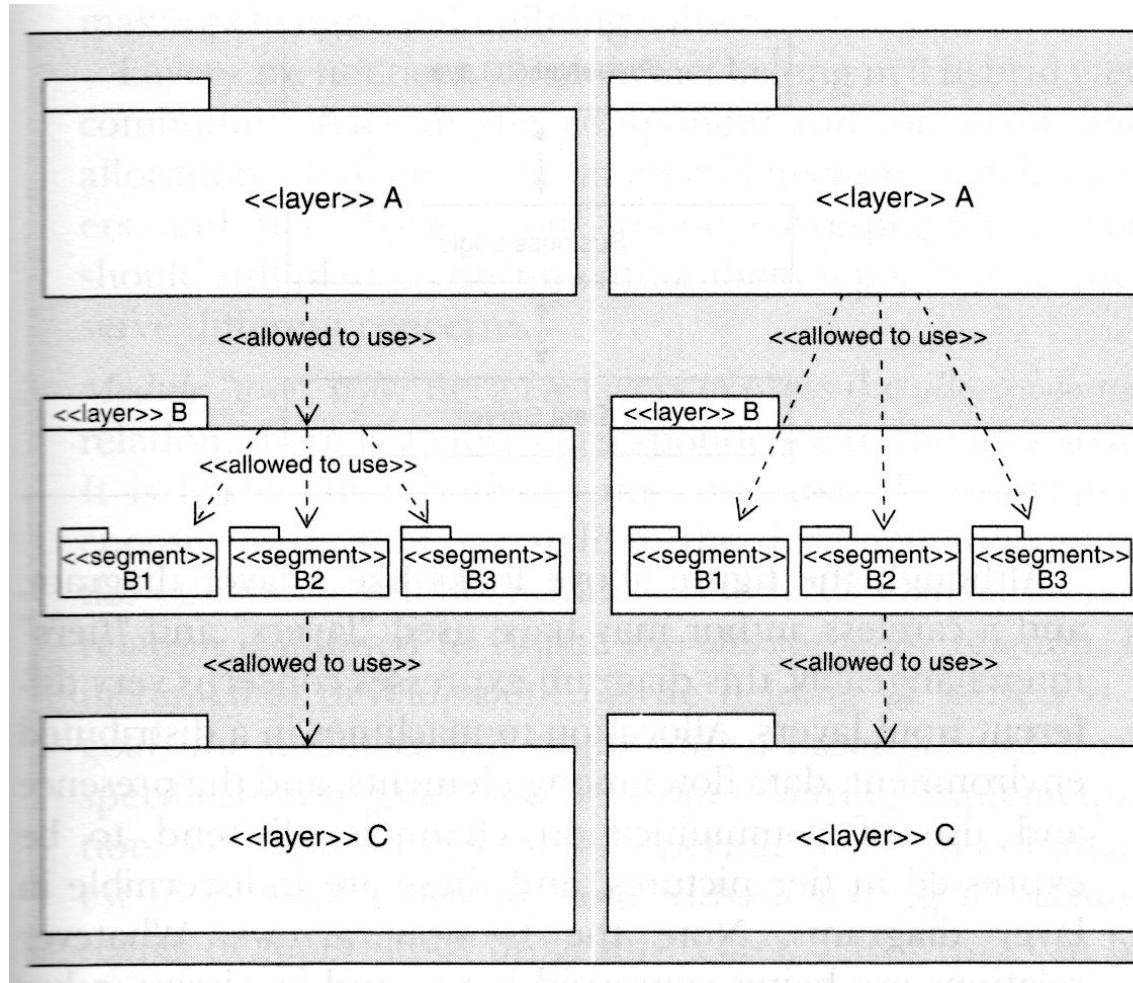
Conceptos Básicos

Ejemplo de notación UML para representar un estilo por capas



Conceptos Básicos

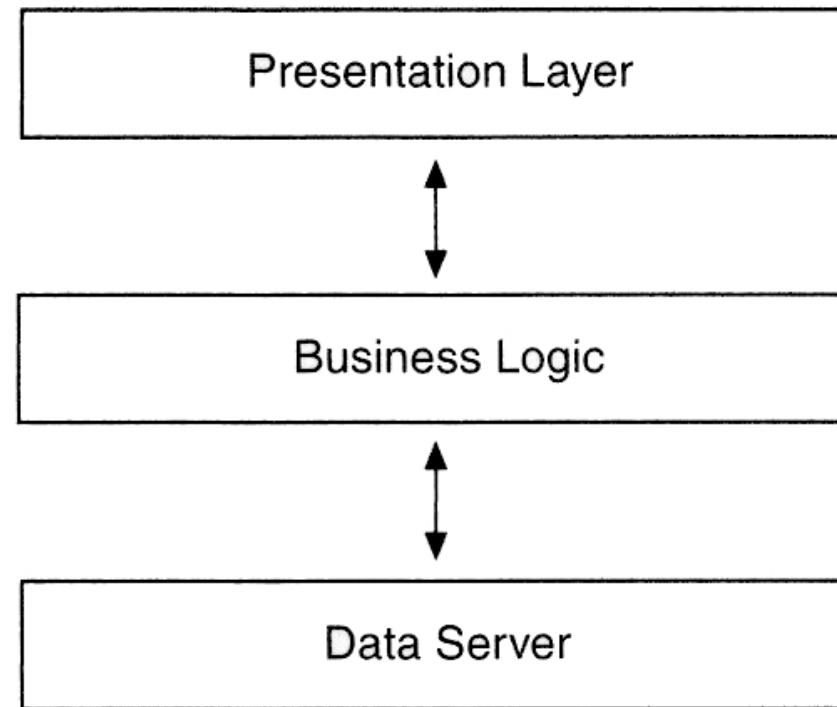
Ejemplo de notación UML para representar un estilo por capas



Tomado de “Documenting Software Architectures” pag 89

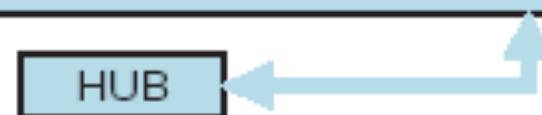
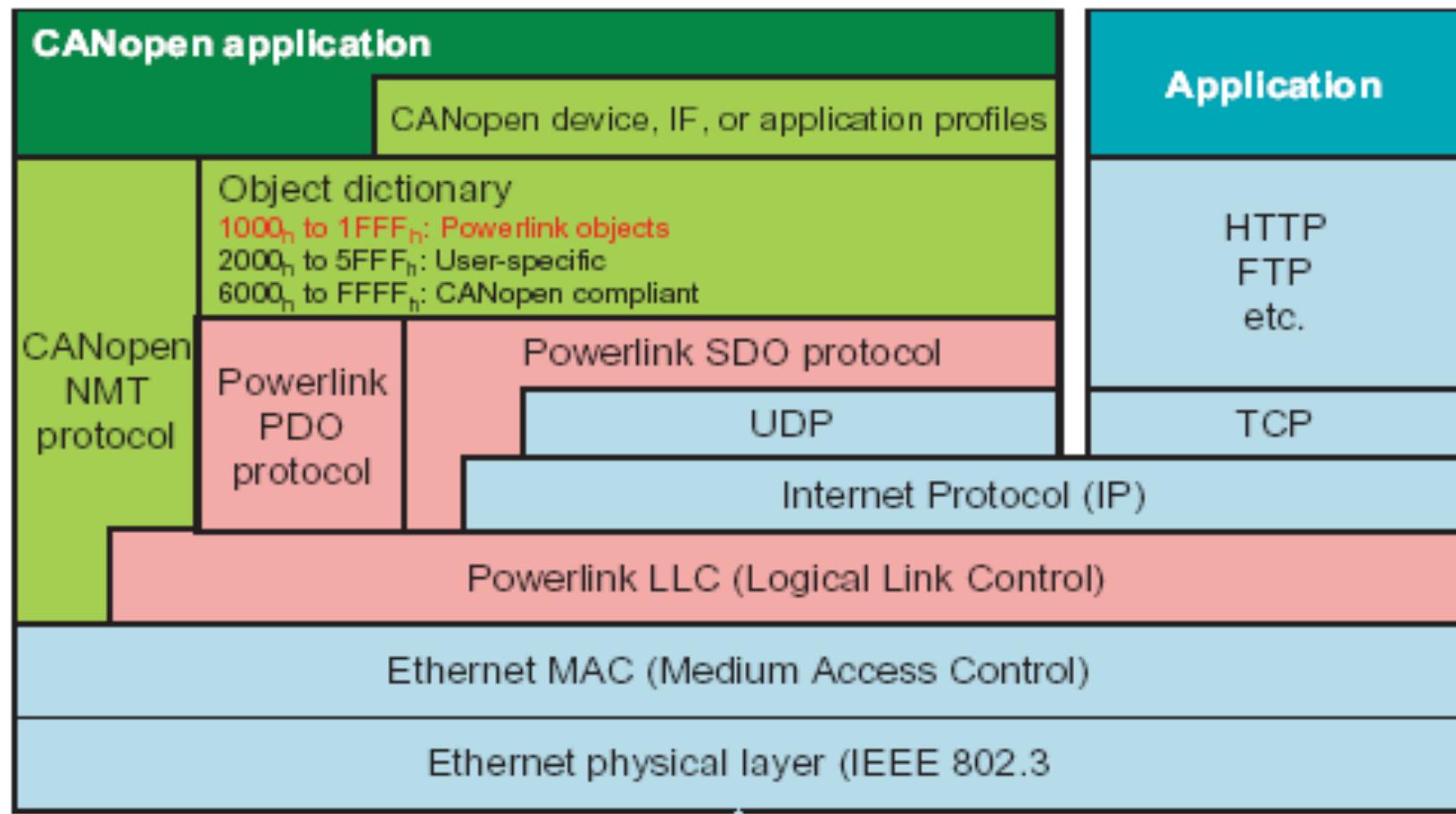
Conceptos Básicos

Qué opina sobre este ejemplo de un estilo arquitectural por capas?



Conceptos Básicos

Ejemplo de notación informal para representar un estadio por capas

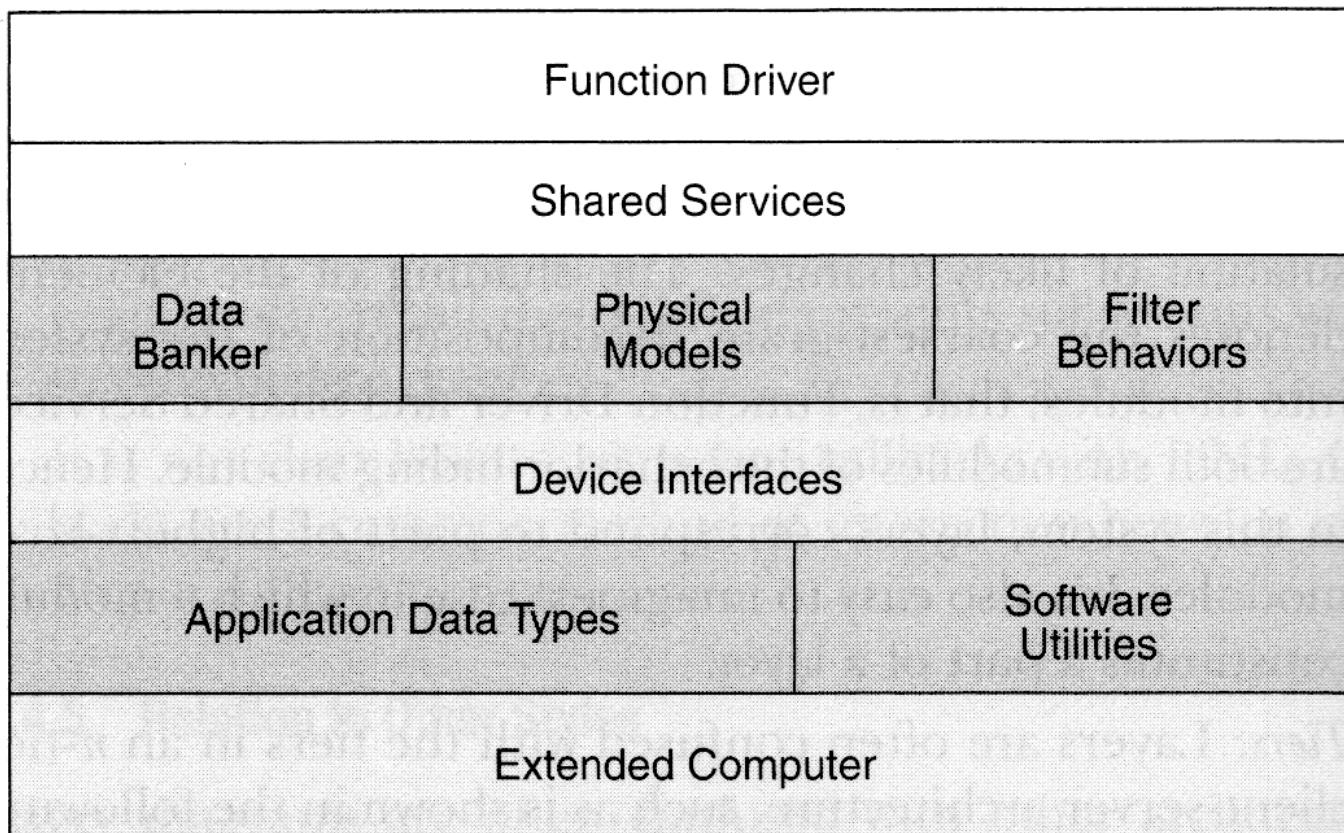


Conceptos Básicos

Ejemplo de notación informal para representar un estilo por capas

KEY

<input type="checkbox"/>	Behavior-hiding module
<input checked="" type="checkbox"/>	Software decision-hiding module
<input type="checkbox"/>	Hardware-hiding module



- [1] Rozanski N, Woods E. “Software Systems Architecture” Addison-Wesley. 2005
- [2] Clements, Paul et al. “Documenting Software Architectures: views and Beyond”. Addison-Wesley. 2002
- [3] Len Bass et Al. “Software Architecture in practice” Second Edition, Addison Wesley, 2007