Ingeniería de Software II

Ingeniería de Software II

Segundo Cuatrimestre de 2011

Clase 19: Conectores de Software

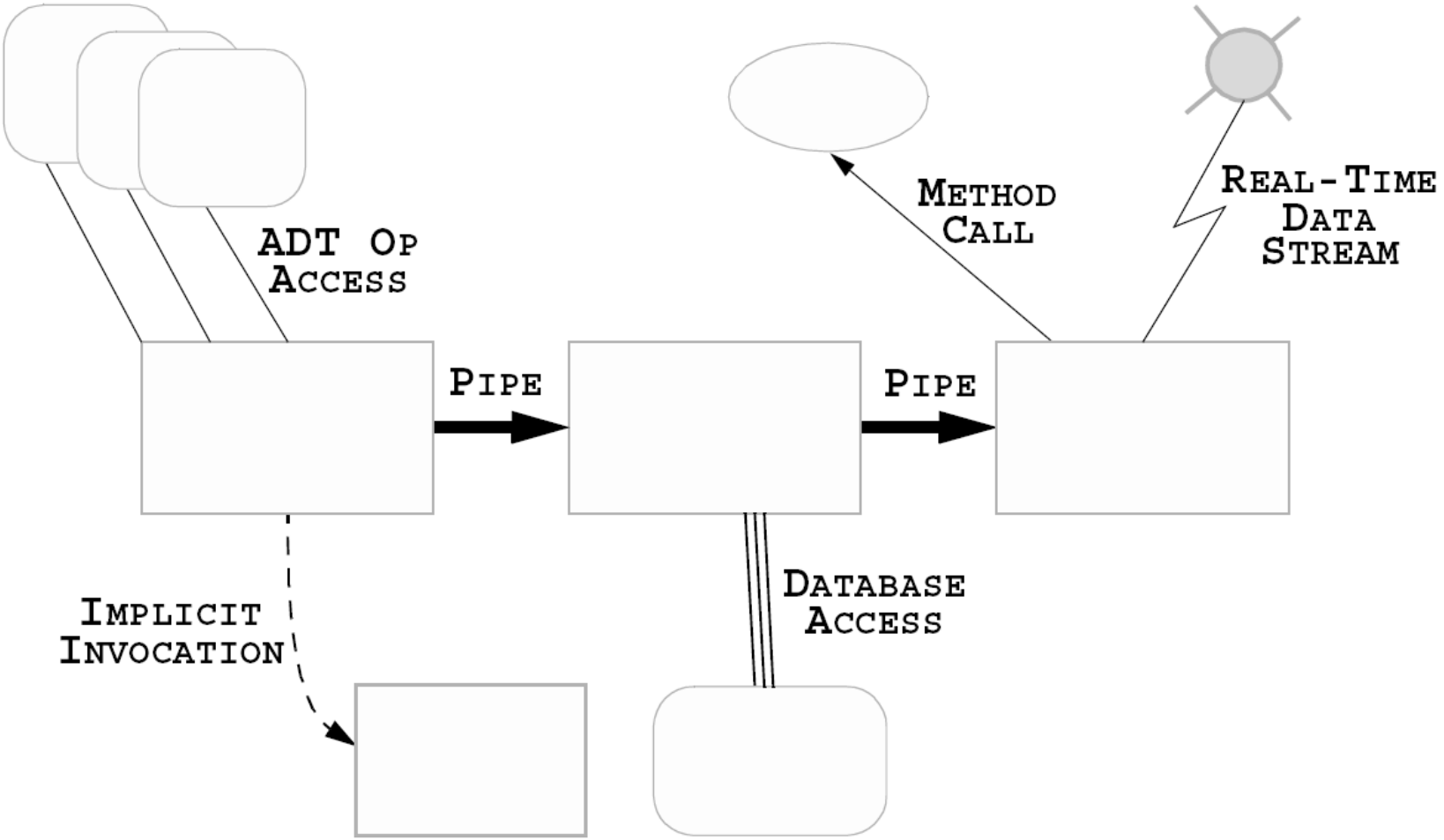
Basada en: *Software Architecture: Foundations, Theory, and Practice*; Richard N. Taylor, Nenad Medvidovic, and Eric M. Dashofy; © 2008 John Wiley & Sons, Inc. Imágenes usadas con permiso.

Buenos Aires, 31 de Octubre de 2011

Definiciones

- Un conector es un elemento de la arquitectura que modela:
  - Interacción entre componentes
  - Reglas sobre esas interacciones
- Cada conector provee
  - “Ductos” para la interacción
  - Transferencia de control y/o datos

¿Dónde están los conectores?



Software Architecture: Foundations, Theory, and Practice; Richard N. Taylor, Nenad Medvidovic, and Eric M. Dashofy; © 2008 John Wiley & Sons, Inc. Reprinted with permission.

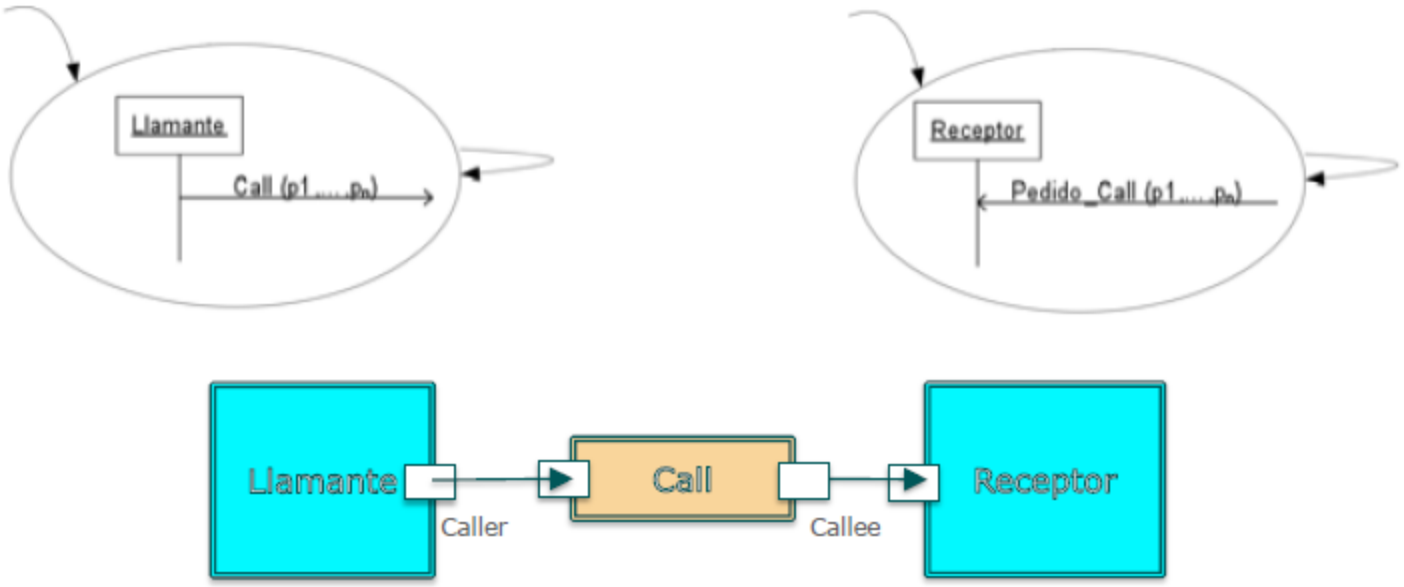
Ambigüedad

- C1 invoca a C2? o C2 invoca a C1?
- Si C1 llama a C2:
  - C1 le pasa datos a C2?, C1 obtiene un resultado de C2?, C1 causa que C2 se "cargue"?, C2 no puede ejecutar hasta que C1 ejecute?, C1 tiene que esperar que C2 termine?
- Si tengo que mostrar flujo de datos: uso dos flechas?, uso una flecha con dos cabezas?, y quién empieza la interacción?, y si son pares?
- Y si la situación es más complicada: protocolo de invocación com múltiples llamados, timeouts, callbacks?, etc.

Conectores

- Un conector representa una interacción en tiempo de ejecución entre dos o más componentes
- El tipo de conector indica la cardinalidad (cantidad de componentes en la interacción), las interfaces que soporta y las propiedades requeridas
- El tipo de conector se hereda generalmente del estilo
- El conector asume un conjunto de roles dentro de la arquitectura
- Un componente puede tener varios **puertos**

Ejemplo: conector call



Pipe and filter

El sensor va enviando las señales que se van encolando hasta que el controlador las va tomando.

-Conector debe definir:

- -Protocolo de Cola
- -Tamaño de la Cola
- -Que pasa cuando se llena
  - -Ignora paquetes?
  - -Descarta los mas viejos?

Conectores en arquitectura vs. en implementación

- En arquitectura:
  - Entidades de "primera clase"
  - Tienen identidad
  - Describen toda la interacción
  - Tienen especificaciones y abstracciones

- En implementaciones
  - Generalmente sin código dedicado
  - Generalmente sin identidad
  - Típicamente no son unidades de compilación

Beneficios de conectores de "primera clase"

- Componentes <> Conectores
  - Funcionalidad específica de una aplicación vs. mecanismos de interacción independientes

- Separar computación de interacción
- Minimizar interdependencias
- Facilitar evolución
- Facilitar heterogeneidad
- Facilitar el testing

Algunos puntos clave (en mi experiencia)

- El bloqueo del que invoca esperando la respuesta del invocado, es decir la necesidad de concurrencia
- La conveniencia de hacer esa comunicación implícita o explícita por razones de flexibilidad
- Las necesidades de "anticipación al cambio" en cuanto a la comunicación (haciendo que usemos, por ejemplo, adaptadores)

Roles de conectores

- Especificación del protocolo que define sus propiedades
  - Tipos de interfaces
  - Certezas sobre propiedades de la interacción
  - Reglas sobre órdenes de invocación
- Roles
  - Comunicación
  - Coordinación
  - Conversión
  - Facilitación

La Tabla Periódica de Conectores

Uchitel, Hirsch, Yankelevich

Knows Target: The source component must explicitly know the target comp.
Request/Reply: The connector guarantees that for each service request exactly one service response is received.
Synchronous: For a comm to take place both source and target must be available.
Flow Control: Connector allows components to stop, start and pause comm.
One Way: Communication can only be done in one way.
Broadcast: Connector allows multiple recipients.
Streamed :Communication is based on continuous flow of data.
Connection Oriented Components must explicitly manage communication sessions.
Reliable: The connector never loses information. Either the target receives transmitted data or the source is informed that the target did not receive it.
Encryption: Connector has security capabilities for encryption.
Authentication: Connector has security capabilities for authentication.
Compression: Connector can transparently compress data for transmission.
Monitoring: Connector can recognize and transmit certain classes of communication events to a monitoring component.
Typed: Connector requires source and target to agree on type of information to be transmitted.

Clasificación de conectores

Property	Pipe	Remote Procedure Call (RPC)	Event Broadcast	Procedure Call (PC)	Shared Data
Knows Target	No	Yes	No	Yes	No
Request/Reply	No	Yes	No	Yes	No
Synchronous	No	Yes	No	Yes	No
Flow Control	Yes	No	No	No	No
One Way	Yes	No	No	No	No
Broadcast	No	No	Yes	No	Yes
Streamed	Yes	No	No	No	No
Connection Oriented	No	No	No	No	No
Reliable	Yes	Yes	Yes	Yes	No
Typed	No	Yes	Yes	Yes	Yes
Encryption	No	No	No	No	No
Authentication	No	No	No	No	No
Compression	No	No	No	No	No
Monitoring	No	No	No	No	No

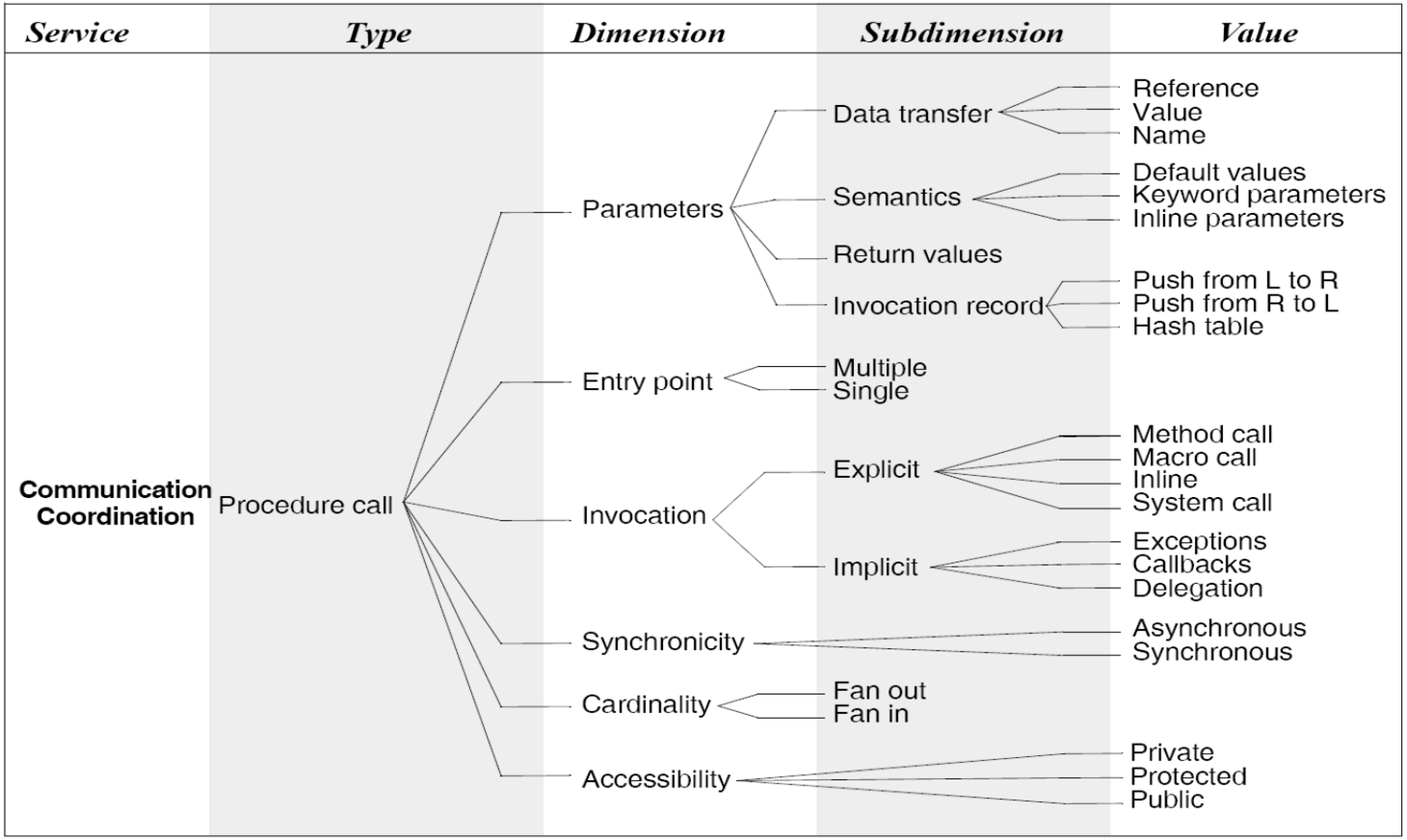
Roles de conectores

- Especificación del protocolo que define sus propiedades
  - Tipos de interfaces
  - Certezas sobre propiedades de la interacción
  - Reglas sobre órdenes de invocación
- Roles
  - Comunicación
  - Coordinación
  - Conversión
  - Facilitación

Tipos de conectores

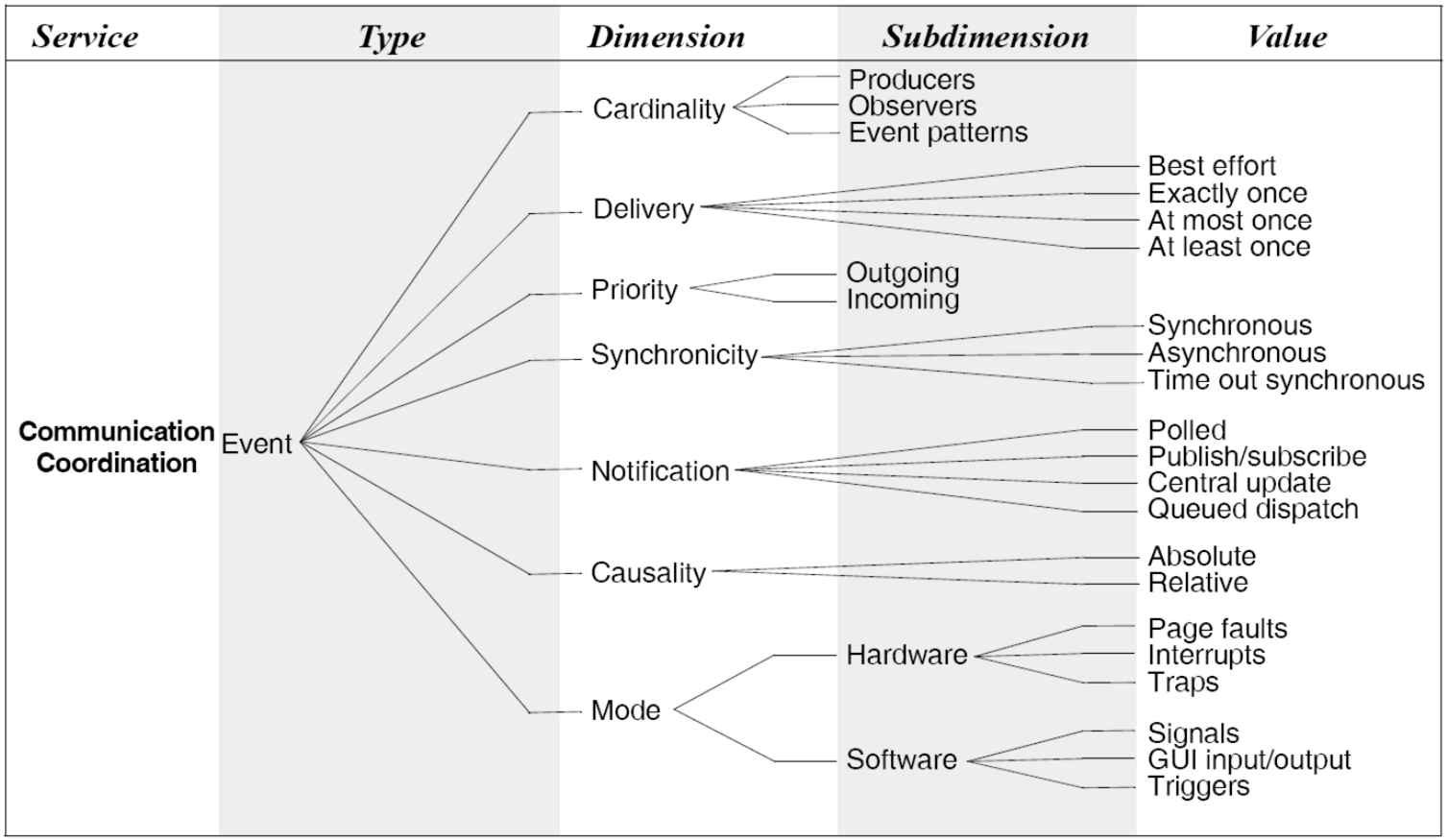
- Llamada a procedimientos
- Datos compartidos
- Acceso a datos
- Evento
- Stream
- Distributor
- Arbitrator
- Adaptor

Conectores de llamada a procedimientos



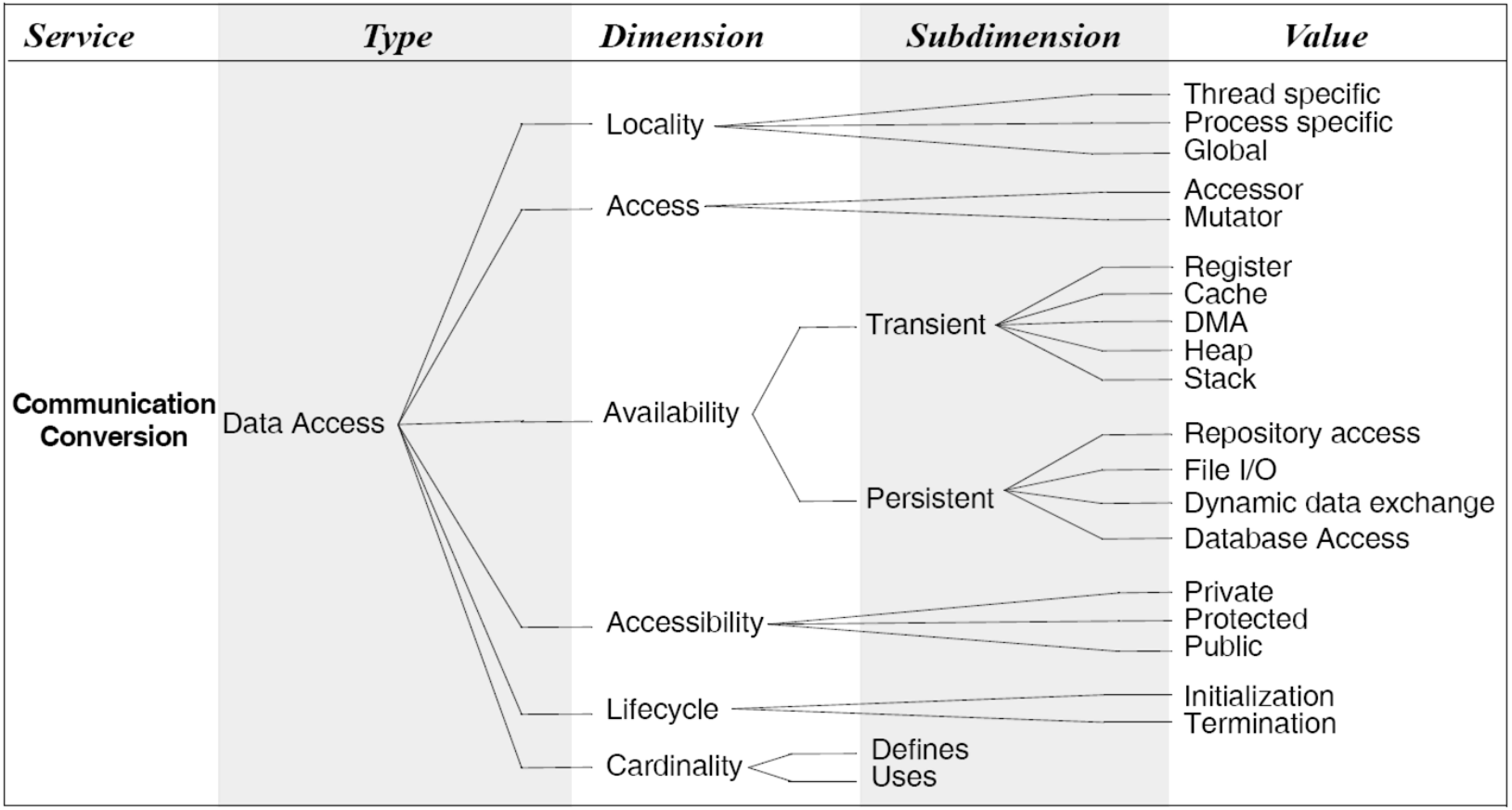
Software Architecture: Foundations, Theory, and Practice; Richard N. Taylor, Nenad Medvidovic, and Eric M. Dashofy; © 2008 John Wiley & Sons, Inc. Reprinted with permission.

Conectores de eventos



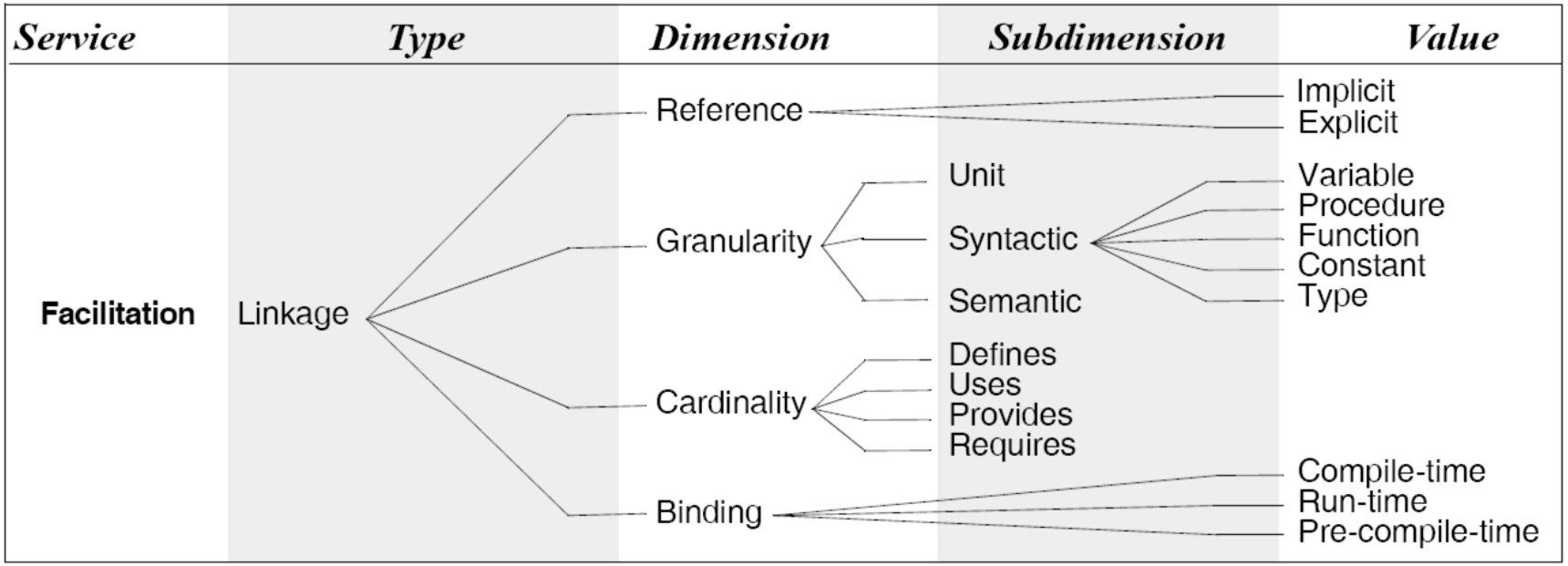
Software Architecture: Foundations, Theory, and Practice; Richard N. Taylor, Nenad Medvidovic, and Eric M. Dashofy; © 2008 John Wiley & Sons, Inc. Reprinted with permission.

Conectores de acceso a datos



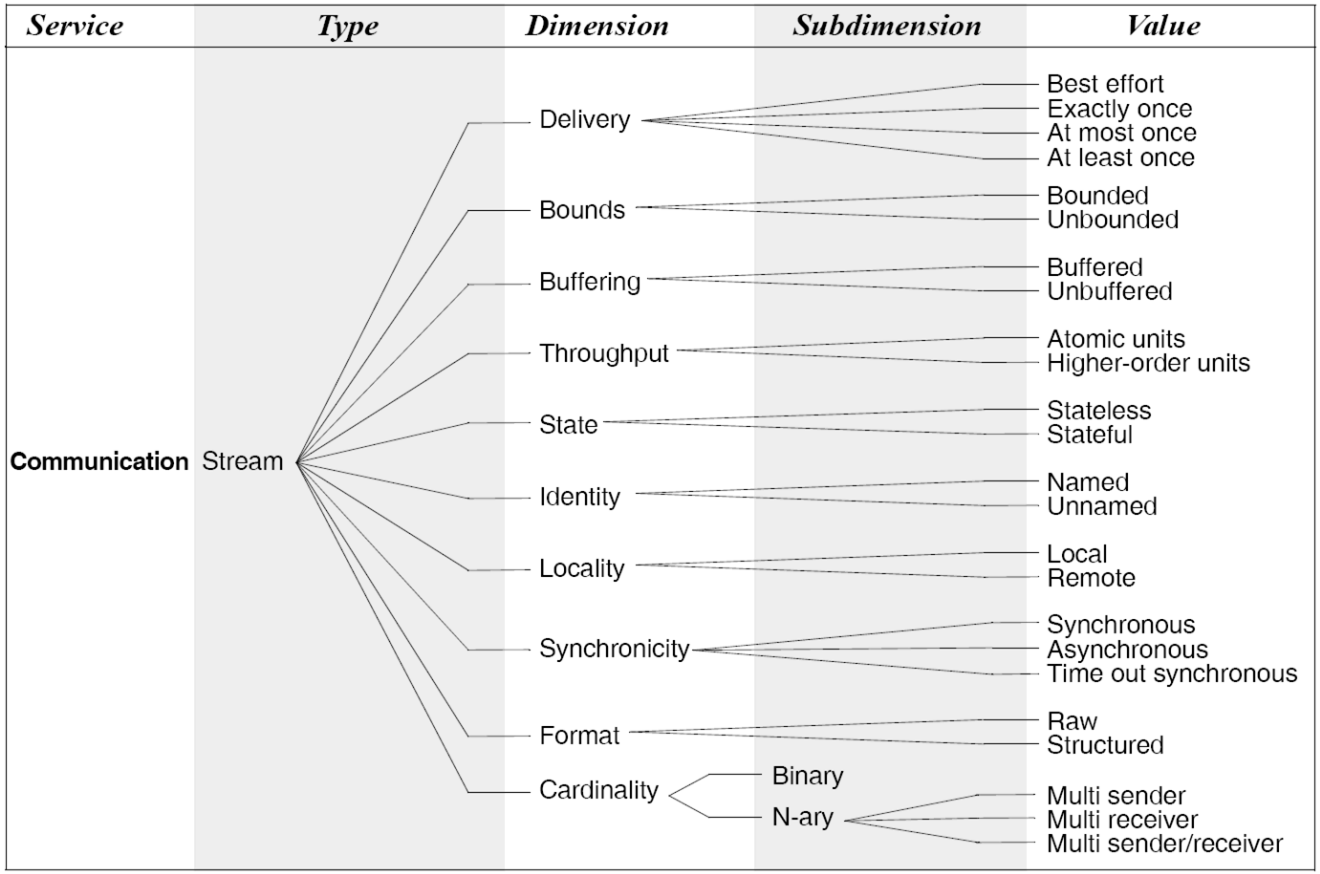
Software Architecture: Foundations, Theory, and Practice; Richard N. Taylor, Nenad Medvidovic, and Eric M. Dashofy; © 2008 John Wiley & Sons, Inc. Reprinted with permission.

Conectores de "linkage"



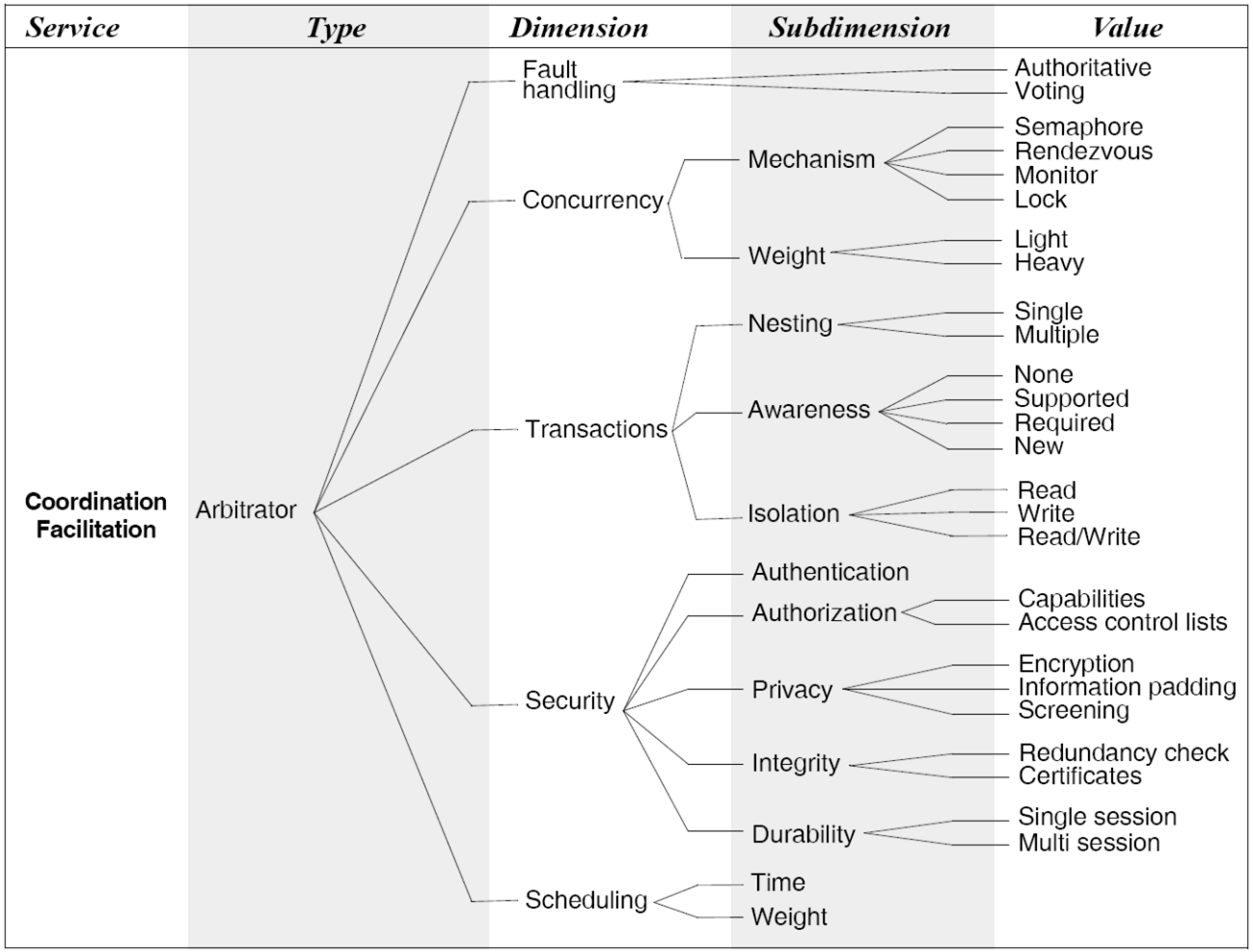
Software Architecture: Foundations, Theory, and Practice; Richard N. Taylor, Nenad Medvidovic, and Eric M. Dashofy; © 2008 John Wiley & Sons, Inc. Reprinted with permission.

Conectores de Stream



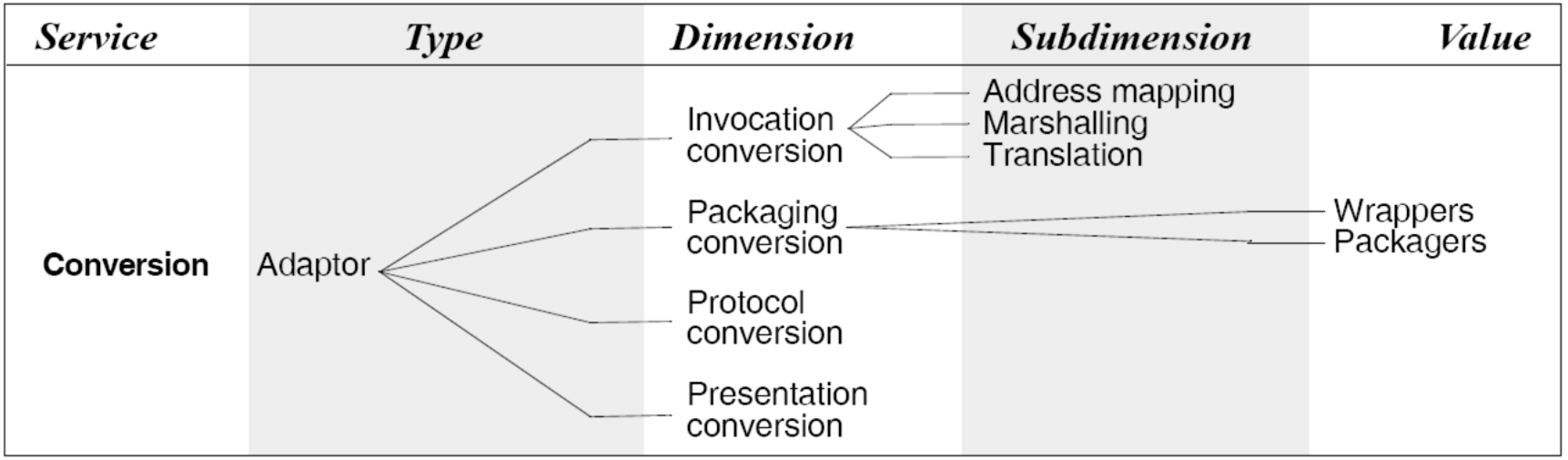
Software Architecture: Foundations, Theory, and Practice; Richard N. Taylor, Nenad Medvidovic, and Eric M. Dashofy; © 2008 John Wiley & Sons, Inc. Reprinted with permission.

Conectores tipo “arbitrators”



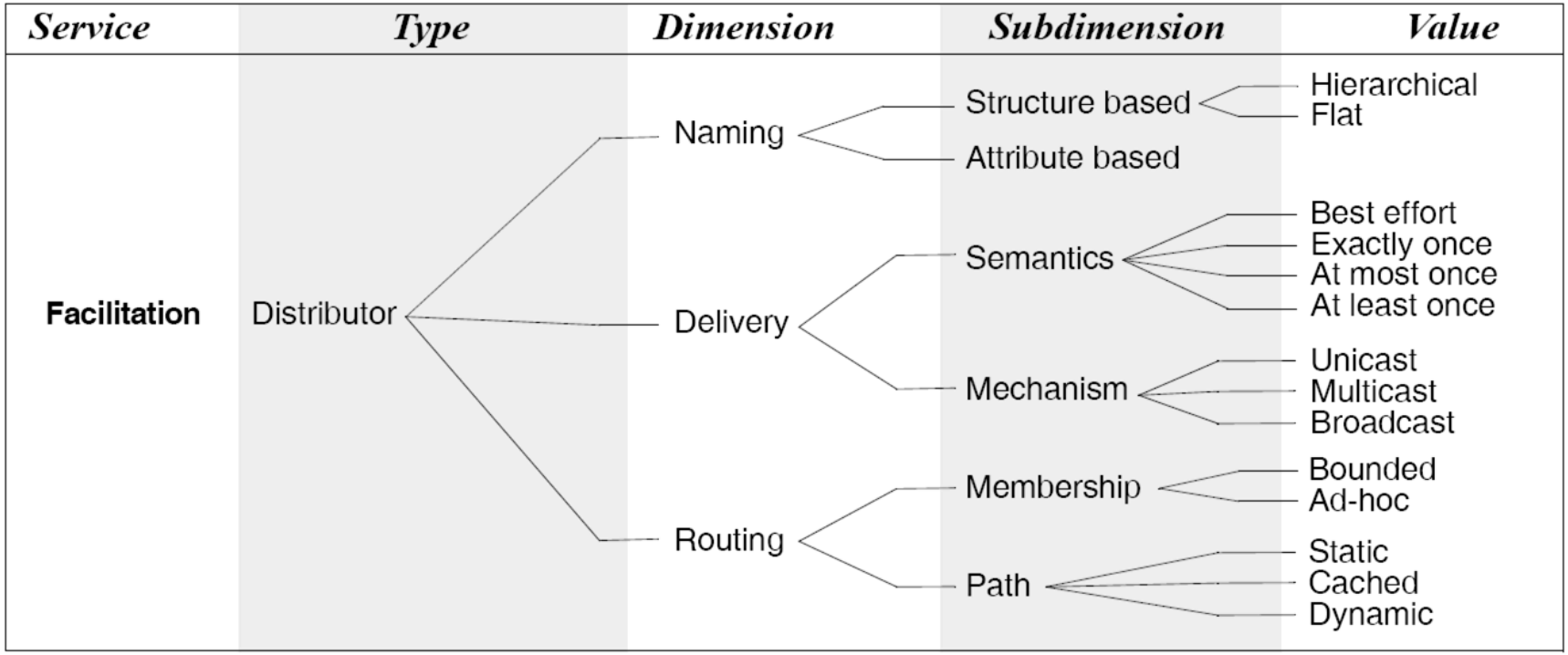
Software Architecture: Foundations, Theory, and Practice; Richard N. Taylor, Nenad Medvidovic, and Eric M. Dashofy; © 2008 John Wiley & Sons, Inc. Reprinted with permission.

Conectores tipo “adaptors”



Software Architecture: Foundations, Theory, and Practice; Richard N. Taylor, Nenad Medvidovic, and Eric M. Dashofy; © 2008 John Wiley & Sons, Inc. Reprinted with permission.

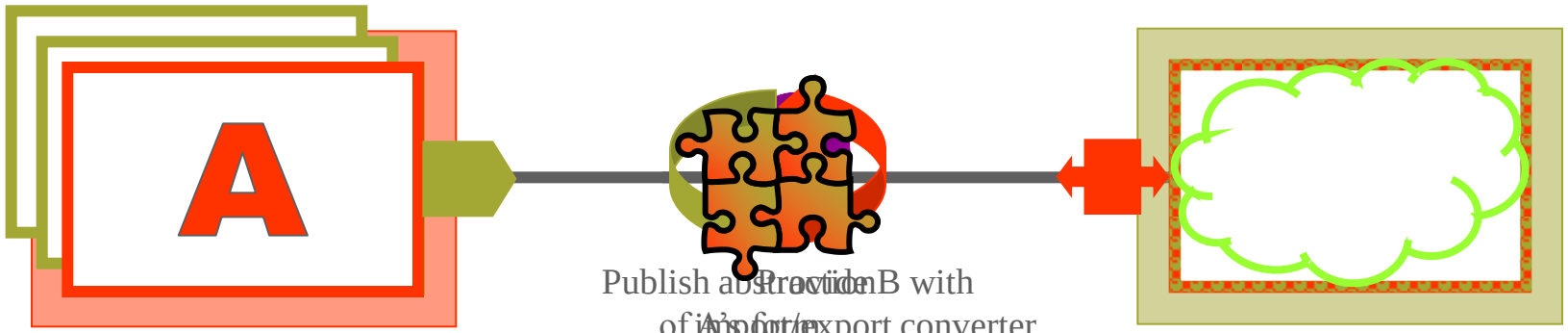
Conectores tipo “distributor”



Software Architecture: Foundations, Theory, and Practice; Richard N. Taylor, Nenad Medvidovic, and Eric M. Dashofy; © 2008 John Wiley & Sons, Inc. Reprinted with permission.

El desafío de los conectores

How do we enable components A and B to interact?



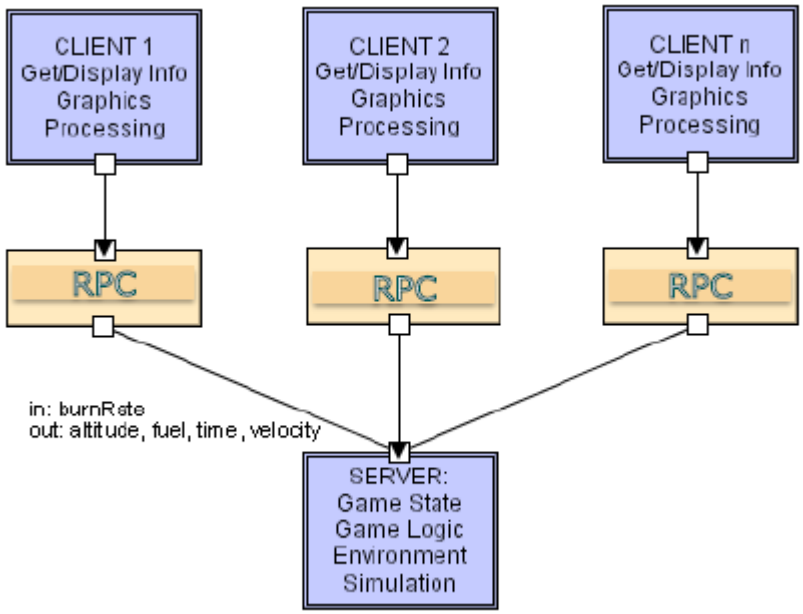
What is the right answer?

¿Cómo se elige un conector?

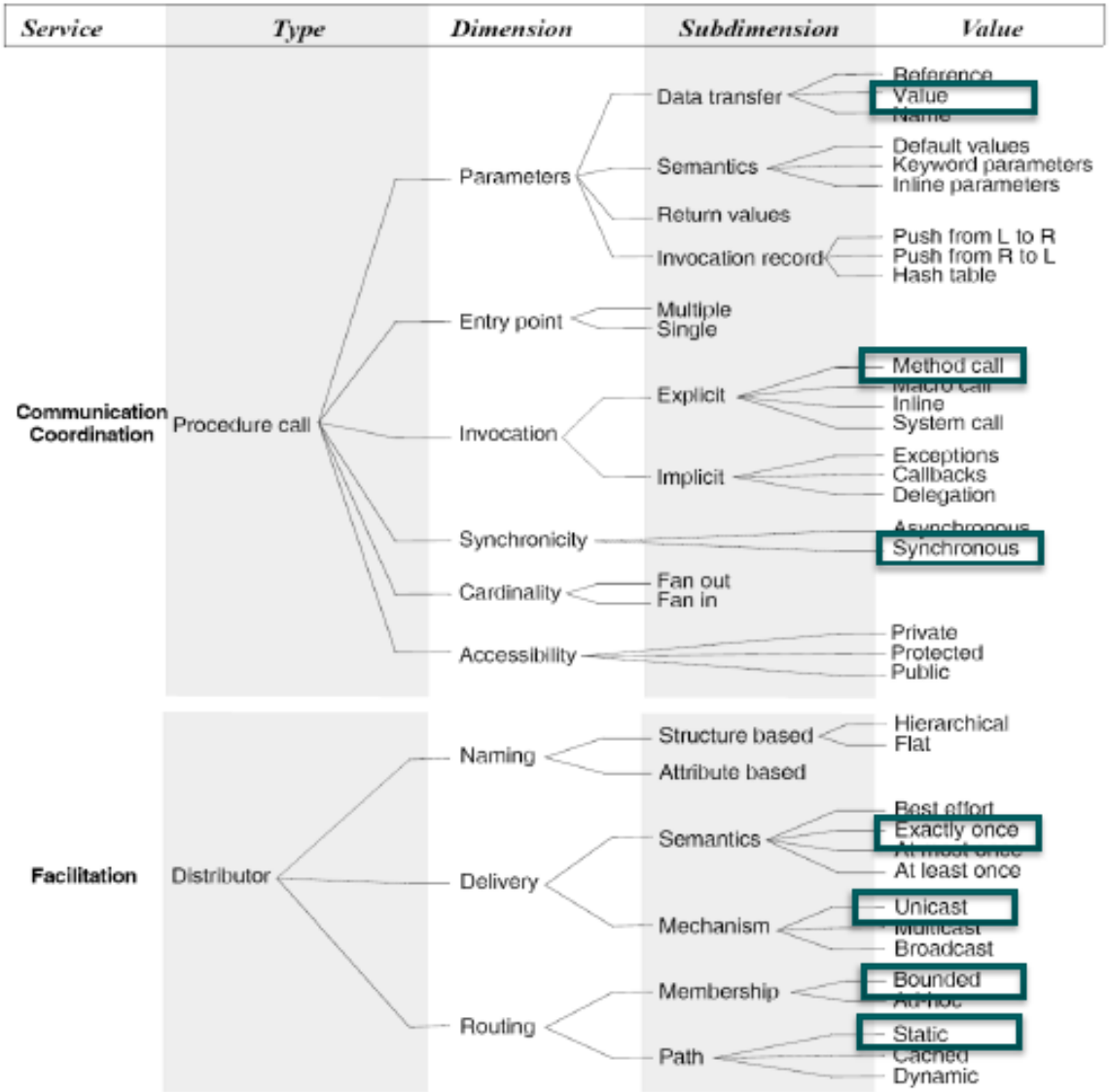
- Determinar las necesidades de interconexión e interacción
- Determinar roles a ser cumplidos por los conectores de software
  - Comunicación, coordinación, conversión, facilitación
- Para cada conector
  - Determinar su tipo apropiado
  - Determinar las dimensiones de interés
  - Seleccionar valores para cada dimensión
- Para conectores “compuestos”
  - Determinar las compatibilidades de conectores atómicos

Composición de componentes básicos

- En muchos sistemas se requieren conectores que “unan” varios conectores indivuduales
- No todos los conectores pueden ser compuestos
  - Algunos no pueden operar o son incompatibles
  - Todos implican trade-offs



RPC – Combinando Call y Distributor



Relaciones entre dimensiones de conectores

- Requiere – &
  - Elegir una dimensión implica elegir otra
- Prohibe – ☒
  - Dos dimensiones no pueden ser compuestas en un sólo conector
- Restringe – ⊘
  - Las dimensiones no siempre son requeridas para ser usadas juntas
  - Algunas combinaciones puede ser inválidas
- Cuidados - &
  - Combinaciones puede resultar en conectores poco confiables o inestables

Inter relaciones entre dependencias