

Introduzione al Middleware

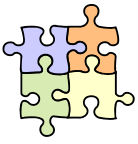
Dispensa MW 1

ottobre 2008



- Fonti

- [Bakken] Middleware (da Encyclopedia of Distributed Computing)
- [Gorton] Essential Software Architecture, Chapter 4, A Guide to Middleware Architectures and Technologies
- [Mühl et al.] Distributed Event-Based Systems, Chapter 2, Basics
- [Bernstein] Middleware, Communications of the ACM, 1996



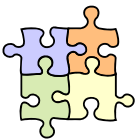
- Obiettivi e argomenti

□ Obiettivi

- comprendere gli aspetti di base della comunicazione tra componenti nei sistemi distribuiti
- definire il middleware
- fornire una classificazione del middleware

□ Argomenti

- comunicazione tra componenti
- middleware
- modelli di interazione
- classificazione del middleware
- che cosa vedremo



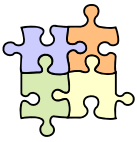
* Comunicazione tra componenti

□ Punto di vista delle architetture software

- un sistema software è costituito da componenti e da relazioni tra questi componenti
 - è necessario ragionare sulle modalità di interazione/comunicazione tra i vari componenti
 - questi componenti, talvolta eterogenei, devono comunicare scambiandosi dati e informazioni di controllo, in modo sicuro ed affidabile

□ In particolare, nei sistemi distribuiti, i componenti runtime sono solitamente distribuiti

- ovvero, in esecuzioni su processi distinti e calcolatori distinti
- la comunicazione è una **comunicazione inter-processo** – **IPC**, interprocess communication – spesso di tipo remoto



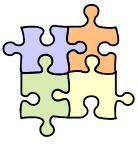
Comunicazione e pattern architetturali

- Alcune modalità di comunicazione tra componenti sono state usate più volte e con successo
 - i **pattern** (o **stili**) **architetturali** descrivono delle soluzioni architetturali usate con successo
 - i pattern architetturali sono soluzioni generali, descritte in modo indipendente dalle particolari tecnologie esistenti
 - sono rappresentazioni astratte di un'architettura, che può essere realizzata in molte forme concrete
 - ciascun pattern architetturale solitamente fa riferimento a una modalità di interazione/comunicazione prevalente tra i componenti identificati
- E' chiaramente utile conoscere e comprendere queste modalità di comunicazioni prevalenti tra componenti distribuiti



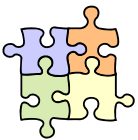
Modalità di comunicazione e middleware

- Le modalità di comunicazione “astratte” sono spesso supportate da tecnologie e strumenti concreti
 - in alcuni casi, da servizi del sistema operativo
 - in particolare, da pipe e socket
 - in altri casi, da strumenti di middleware



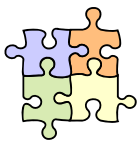
* Middleware

- Per **middleware** si intende
 - una classe di tecnologie software sviluppate per aiutare gli sviluppatori nella gestione della complessità e della eterogeneità presenti nei sistemi distribuiti
 - uno strato software “in mezzo”
 - sopra al sistema operativo, ma sotto i programmi applicativi
 - fornisce un’astrazione di programmazione distribuita – un modello computazionale uniforme
 - per mascherare le eterogeneità di elementi sottostanti – reti, hardware, sistemi operativi, linguaggi di programmazione, ...
- Il middleware ha lo scopo di fornire dei blocchi utili per costruire componenti software che possano lavorare insieme ad altri in un sistema distribuito



Middleware - esempi

- Alcuni esempi di (famiglie di) prodotti di middleware
 - RPC,
 - Java RMI,
 - Messaging,
 - SQL Stored Procedures,
 - ORB,
 - TP monitor,
 - web services,
 - ...



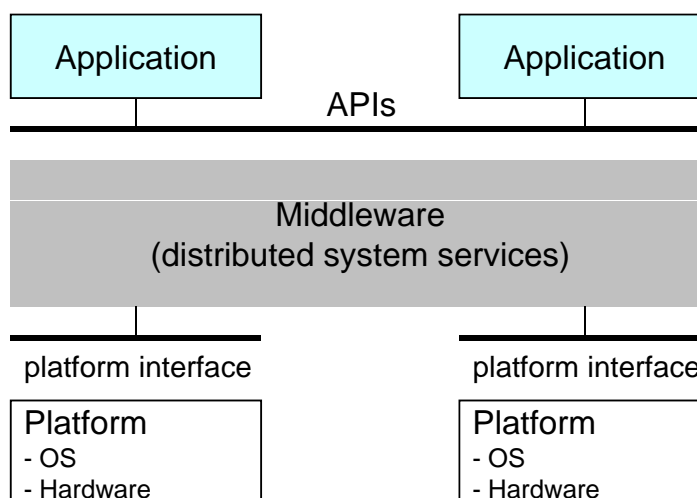
Middleware

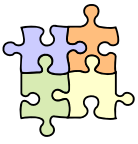
- Chiaramente, così come
 - un calcolatore può essere programmato usando direttamente il linguaggio macchina
- è anche vero che
 - un sistema distribuito può essere programmato usando direttamente le socket
- ma, in pratica, non è esattamente la stessa cosa ...
- Se è possibile dire che
 - i linguaggi di programmazione sono il software che rendono facilmente programmabili i calcolatori
- allora si può anche dire che
 - il middleware è il software che rende facilmente programmabile un sistema distribuito



Middleware

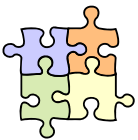
- Un **servizio di middleware** è un servizio general-purpose che si colloca tra piattaforme e applicazioni [Bernstein]





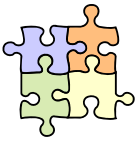
Middleware

- Il middleware supporta il collegamento (plumbing/piping/wiring) tra componenti software
 - fornisce meccanismi per connettere componenti software
 - affinché possano scambiare informazioni
 - sulla base di meccanismi di programmazione relativamente semplici
 - in modo provato
 - le connessioni possono avvenire sulla base di topologie utili e ben comprese
 - sono possibili connessioni uno-a-uno, uno-a-molti, multi-a-molti
 - il middleware è un'infrastruttura trasparente agli utenti del sistema



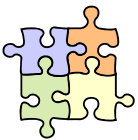
Middleware

- Ciascuno strumento di middleware ha lo scopo di mascherare qualche tipo di eterogeneità comunemente presente in un sistema distribuito
 - il middleware maschera sempre l'eterogeneità delle reti e dell'hardware
 - alcuni strumenti di middleware mascherano eterogeneità nel sistema operativo e/o nei linguaggi di programmazione
 - alcuni strumenti di middleware mascherano eterogeneità nelle diverse implementazioni di uno stesso standard di middleware
 - ad es., alcune implementazione di Corba o Java EE
 - le astrazioni di programmazione offerte dal middleware possono fornire trasparenza rispetto ai seguenti aspetti
 - posizione, concorrenza, replicazione, fallimenti, mobilità
 - in alternativa, il programmatore dovrebbe farsi esplicitamente carico di queste eterogeneità e di questi aspetti



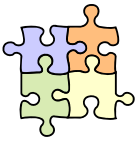
* Modelli di interazione

- Supponiamo che un componente (e.g., un processo) consumatore (**consumer**) sia interessato ad una risorsa (servizio/informazione) posseduta da un componente fornitore (**provider**)
 - quali i possibili modelli di interazione?
- Ecco una possibile classificazione, basata su
 - chi inizia l'interazione
 - consumer
 - provider
 - modalità di indirizzamento
 - diretto – i componenti si conoscono esplicitamente
 - indiretto – la cooperazione avviene mediante un intermediario (e.g., un broker o una coda)



Tassonomia dei modelli di interazione

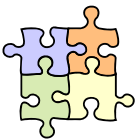
		iniziatore	
		consumer	provider
indirizzamento	diretto	<i>Request/Replay</i>	<i>Callback</i>
	indiretto	<i>Anonymous Request/Replay</i>	<i>Event-based</i>



Modelli di interazione

- Request-Replay
 - interazione iniziata dal consumatore, indirizzamento diretto
 - interazione di tipo client-server – ad es., RPC

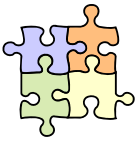
- Anonymous Request-Reply
 - interazione iniziata dal consumatore, indirizzamento indiretto
 - c'è un livello di indirectione – ad es., broker, application server, un sistema di load balancing – per ridurre l'accoppiamento tra i componenti



Modelli di interazione

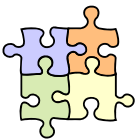
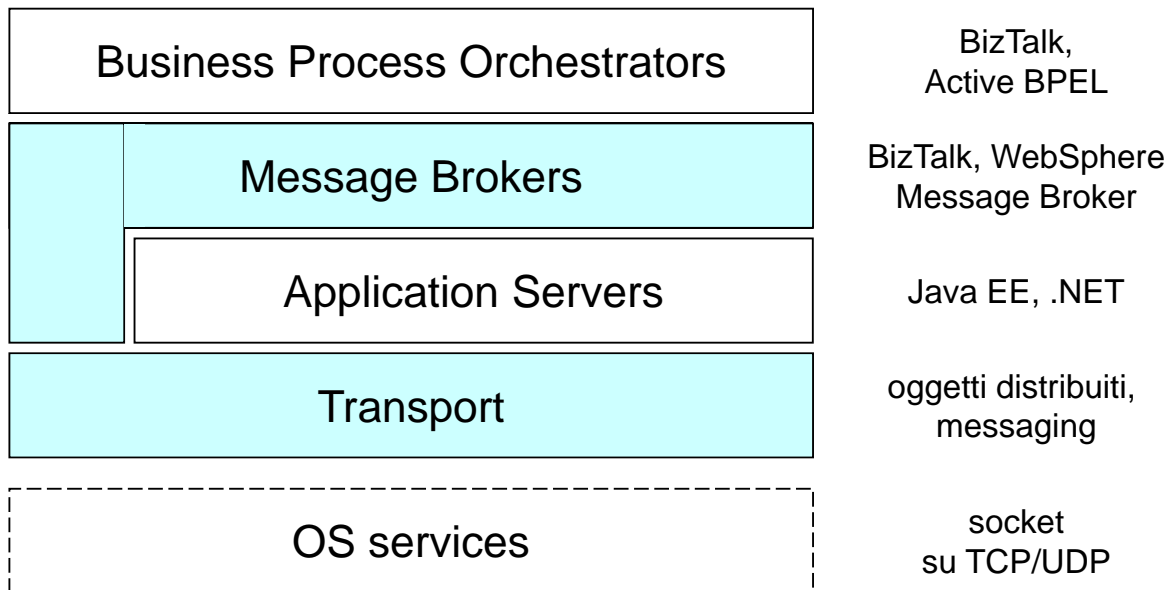
- Callback
 - interazione iniziata dal fornitore, indirizzamento diretto
 - basato sul pattern Observer [GoF/POSA]
 - implementazioni sofisticate portano al modello Event-Based

- Event-Based
 - interazione iniziata dal fornitore, indirizzamento indiretto
 - il fornitore (provider) comunica eventi significativi ai suoi consumatori (subscriber), sotto forma di messaggi/notifiche
 - il provider è disaccoppiato dai consumatori
 - vengono inviati messaggi di notifica, non invocate operazioni remote
 - i consumer implementano un'operazione generica per consumare notifiche – ciascun componente consumer può reagire diversamente a ciascun tipo di notifica



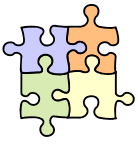
* Classificazione del middleware

- Una possibile classificazione delle tecnologie di middleware



* Che cosa vedremo

- Che cosa vedremo in questo corso
 - socket
 - meccanismo di base della IPC
 - non è uno strumento di middleware
 - per intuire alcune problematiche affrontate dal middleware
 - oggetti distribuiti
 - comunicazione request-replay e callback, sincrona
 - messaging
 - comunicazione event-based, asincrona
 - application server
 - componenti e contenitori
 - web services
 - servizi interoperabili, sincroni e asincroni



Che cosa non vedremo

- Che cosa vedremo e non vedremo in questo corso
 - delle varie tecnologie, vedremo solo esempi di uso piuttosto semplici
 - al fine di introdurre le tecnologie, alcune finalità, alcuni problemi affrontati e risolti, alcuni problemi non risolti che devono dunque essere presi in considerazione dal programmatore
 - faremo alcune considerazioni metodologiche, di natura generale, circa l'uso dei vari paradigmi di comunicazione
 - tuttavia non vedremo aspetti metodologici o pattern specifici per le varie tecnologie
 - non vedremo soluzioni tecnologiche complete circa l'applicazione delle varie metodologie generali