



Università degli Studi di Bologna

Facoltà di Ingegneria

Sistemi Distribuiti M

A.A. 2013 – 2014

Esercitazione JBI

Ing. Stefano Monti

stefano.monti@epocaricerca.it

Un tipico scenario di integrazione

Si supponga di dover integrare due applicazioni che attualmente “dialogano” tramite scambio di file:

- applicazione A **scrive file** su cartella *homeA*
- applicazione B **legge file** da cartella *homeB*
- attualmente, un **processo di sistema**
 - monitora la cartella *homeA*
 - sposta file da *homeA* a *homeB* appena disponibili

Scenario intrinsecamente asincrono e basato su scambio di messaggi

- introdurre un ESB JBI-compliant che **sostituisca** il processo di sistema
- approccio incrementale
 - reimplementare lo scenario basilare
 - estenderlo con funzionalità avanzate
 - interfacce WS
 - messaggistica JMS

Pattern di realizzazione in JBI

- 1) identificare i componenti necessari alla realizzazione della soluzione
 - Binding Component (BC) e Service Engine opportuni (SE)
 - es. interazione filesystem, trasformazione di formato XSLT, ecc...

 - 1) per ciascun componente scelto, creare una o più **Service Unit (SU)** secondo le specifiche **JBI**, cioè configurazioni specifiche del componente
 - es. polling FS su cartella X, scrittura su cartella Y

 - 1) una volta create tutte le **SU** necessarie, creare un **Service Assembly (SA) JBI**
 - aggregato di service unit che realizza un determinato scenario di integrazione

 - 1) caricare (deploy) la SA su container ESB
 - es. Apache ServiceMix mette a disposizione una cartella *hotdeploy* per deploy a container attivo
-

Strumenti suggeriti

Si suggerisce l'adozione di ESB Apache ServiceMix (v. 3.3.x)

<http://servicemix.apache.org>

Lista dei componenti (sia *Binding Component* che *Service Engine*)

<http://servicemix.apache.org/components-list.html>

ServiceMix caldeggia l'uso di Maven per

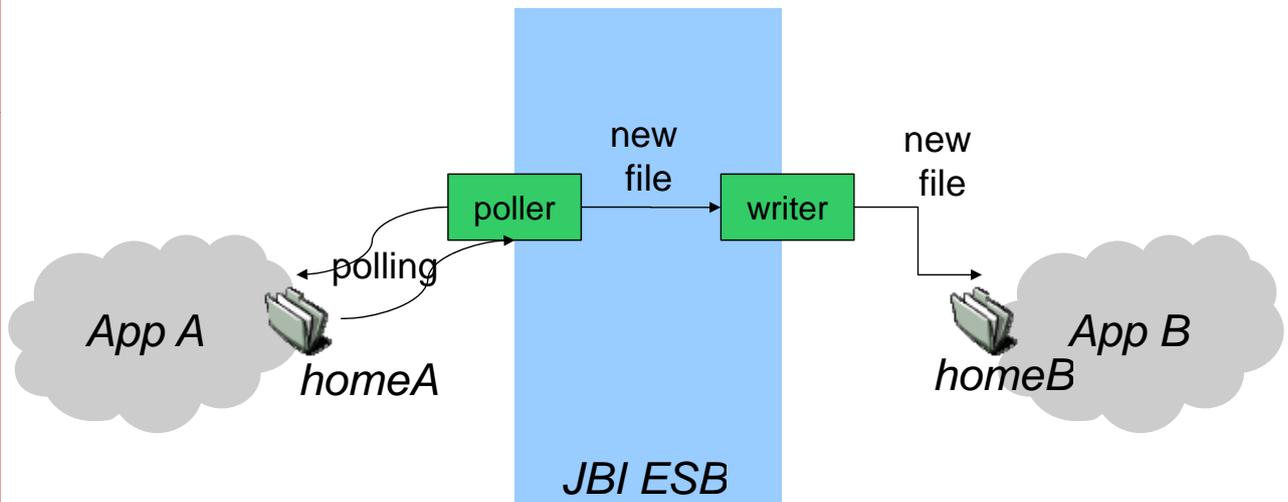
- creazione di progetti JBI
 - archetipi Maven per la creazione di SU e SA
- compilazione e packaging
 - plugin Maven per la creazione di .jar *JBI compliant* per SU ed SA

Sono disponibili esempi nel tutorial di ServiceMix

<http://servicemix.apache.org/tutorials.html>

Obiettivo 1

Realizzare spostamento di file da cartella *homeA* a *homeB* mediante ESB JBI-compliant



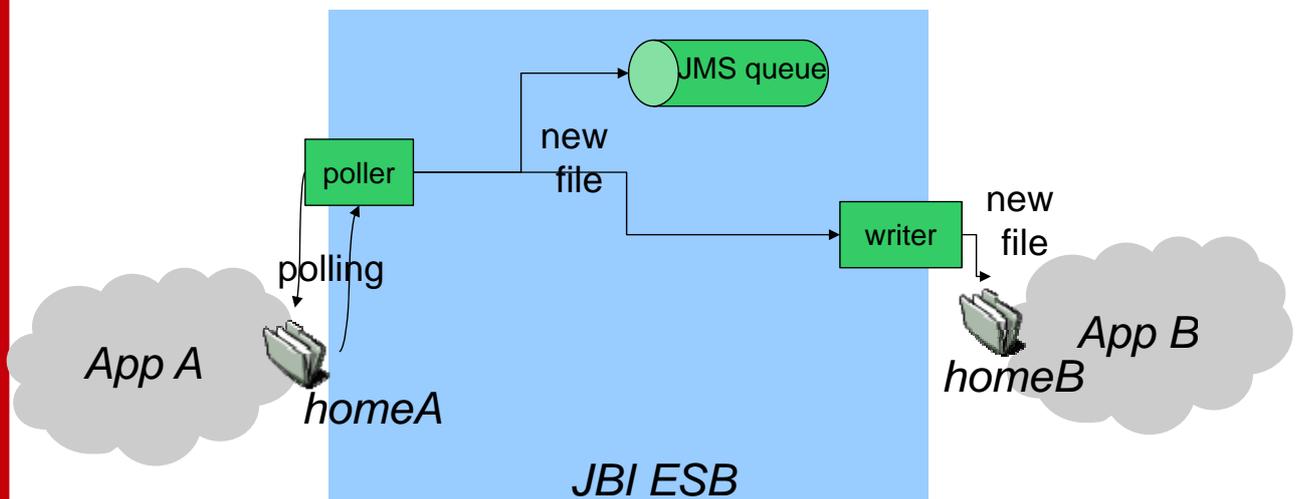
Obiettivo 2

Aggiungere messaggistica tra *file poller* e *file writer*

- store di messaggi scambiati
- uso di una coda JMS

Poller invia ciascun file sia a *writer* che alla coda JMS

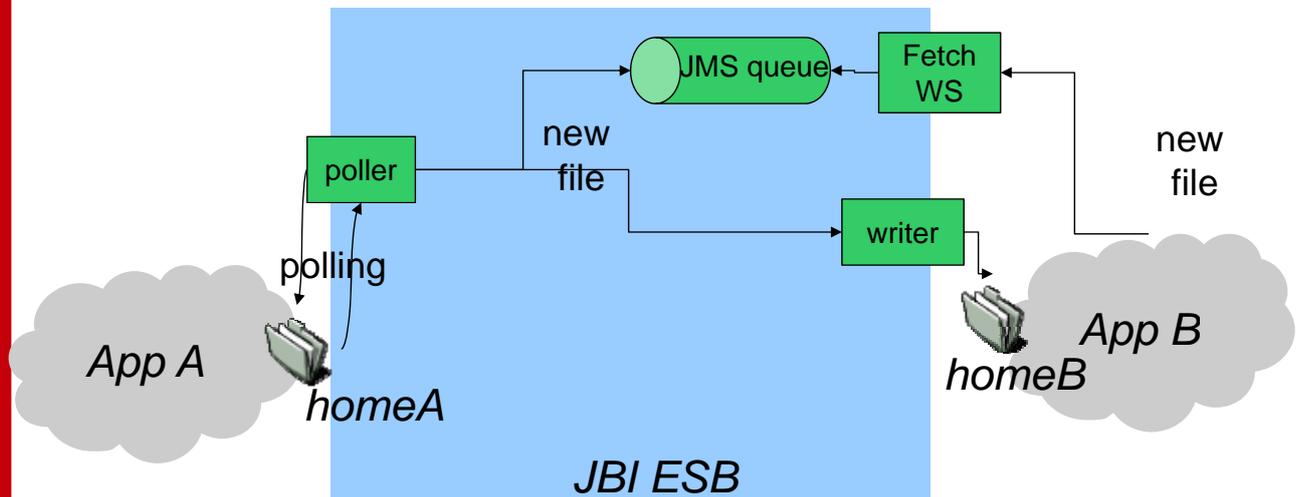
- può farlo *poller* direttamente? È necessario qualche altro componente?



Obiettivo 3

App B preleva messaggi dalla coda mediante Web Services (eliminiamo l'integrazione *old-style* a scambio di file!)

- *Fetch_WS* mette a disposizione un metodo per prelevare il primo messaggio dalla coda e restituirlo ad App B
- App B invoca *Fetch_WS*



Note all'obiettivo 3 – 1/2

- Per la realizzazione di *Fetch_WS*, si suggerisce l'utilizzo dei componenti ServiceMix relativi ad Apache CXF
 - componenti *servicemix-cxf-** in ServiceMix
 - <http://cxf.apache.org>
- Quanti e quali componenti JBI sono necessari?
 - logica vera e propria di prelievo (implementazione WS)
 - interazione via protocollo SOAP
- Suggerimenti sui componenti *servicemix-cxf*
 - richiedono particolare attenzione alla consistenza dei *namespace* dichiarati per i WS
 - richiedono di dichiarare e impacchettare i WSDL corretti dei WS all'interno delle SU che li usano
 - Apache CXF mette a disposizione un tool per la generazione di un WSDL a partire dalla classe Java che lo implementa (annotazioni standard *javax.jws.WebService*)

Note all'obiettivo 3 – 2/2

- Il WS deve interagire con la coda JMS secondo usuale pattern di reperimento messaggi da JMS
 - Necessità di reperire la ConnectionFactory JMS (implem. ActiveMQ) usata in ServiceMix
 - esiste un file di conf. in ServiceMix che contiene la configurazione iniziale del registry JNDI...
 - Necessità di reperire la coda: ServiceMix a default **non pubblica** i nomi delle code su registro JNDI; due opportunità
 - pubblicare la coda su JNDI
 - Utilizzare API esplicite di ActiveMQConnectionFactory per reperire la coda
-

Tips & Tricks

In caso si utilizzino i plugin Maven per ServiceMix

- alcune recenti versioni dei plugin riferiscono delle librerie che al momento risultano corrotte (jar file corrotti)
 - in particolare, *spring-support*, *spring-dao*, *xerces*
 - soluzione
 - scaricare le versioni opportune dei pacchetti che risultano corrotte (vd. output di build Maven)
 - dai relativi siti
 - oppure da repository di pom/jar (es. www.jarvana.com, www.findjar.com)
 - sostituire i jar funzionanti a quelli corrotti nel repository maven locale
 - es. `/home/<user>/m2/repository/...../spring-support.jar`
-