

**Agent and Object Technology LAB**  
**Dipartimento di Ingegneria dell'Informazione**  
**Facoltà di Ingegneria**  
**Università degli Studi di Parma**



# **SVILUPPO DI UN'INFRASTRUTTURA *PEER-TO-PEER* A SUPPORTO DI UN SISTEMA PER LA CONDIVISIONE DI INFORMAZIONI**

**Relatore**

*Chiar.mo Prof. Ing. A. Poggi*

**Correlatore**

*Dott. Ing. M. Mari*

**Tesi di Laurea Specialistica di**

*Michele Longari*

*Peer-to-peer, MAS e problematiche legate ad essi*

# **INTRODUZIONE AL PROGETTO**

- ◆ **Sistemi *peer-to-peer*: P2P**
  - Assenza di nodi gerarchizzati (*client* o *server*) fissi
  - Vantaggi: scalabilità, *fault tolerance*, sfruttamento efficiente delle risorse di rete inutilizzate, ...
  - Svantaggi: ogni *peer* deve poter processare le richieste di tutti gli altri, necessità di protezione del sistema, ...
- ◆ **Sistemi multi-agente: MAS**
  - Agenti: entità autonome capaci di muoversi in rete, reperire informazioni e di interfacciarsi fra di loro
  - Interagiscono con l'ambiente di esecuzione e operano in maniera asincrona ed autonoma su di esso

- ◆ Possibilità di **aumentare le risorse condivise**
  - ◆ **Principale ostacolo:** assenza di *standard* comuni
- 
- I. **Identificare le caratteristiche comuni** per definire un'architettura *peer-to-peer* “astratta”
  - II. **Stabilire un opportuno set di protocolli comuni** per i vari aspetti dei sistemi *peer-to-peer*
  - III. **Definire un comune linguaggio** per la **comunicazione** tra i vari nodi della rete

*Strumenti e tecnologie sfruttate lungo il corso del progetto*

# **STRUMENTI E TECNOLOGIE UTILIZZATE**

- ◆ **Problema** ricorrente: integrazione dei servizi
- ◆ **Soluzione:** *pattern Inversion of Control (IoC)*
  - Detto anche *Dependency Injection*
  - In antitesi con quello più classico del *Service Locator*
- ◆ **IoC:** descrizione del valore da dare agli oggetti e dei collegamenti esistenti fra di loro
  - E' il *container* che "inietta" le dipendenze tra gli oggetti
  - Non è necessaria una classe di controllo, ma sono i servizi stessi che vengono resi disponibili in un *container*
- ◆ **Implementazione dell'IoC:** *framework* Spring

## ◆ JXTA

- *Suite* di protocolli aperti
- Basati sullo *standard XML*
- Disegnati per lo sviluppo delle applicazioni P2P e delle loro funzionalità

## ◆ Caratteristiche

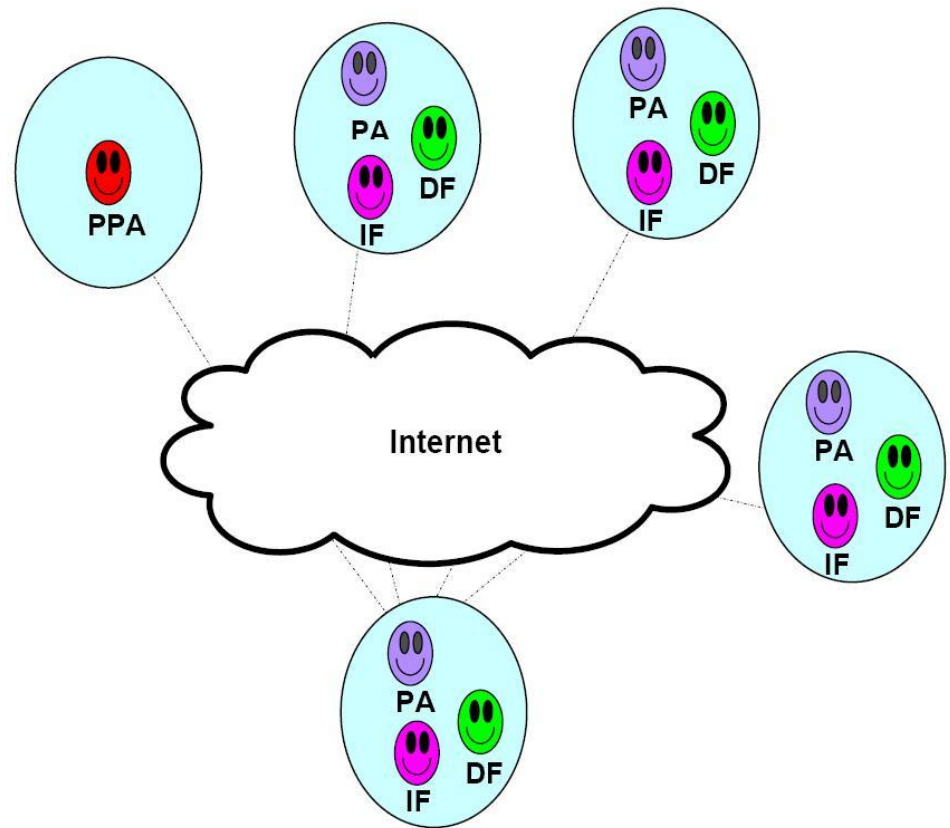
- Definizione di una *overlay network* virtuale
- Interoperabilità fra i *peer*
- Indipendenza dalla piattaforma
- Ubiquità

## ◆ Sistemi P2P

- Molto efficienti per contenuti multimediali
- Poco efficienti per *file* e documenti testuali

## ◆ RAIS

- ◆ Condivisione P2P + ricerca *Desktop Search*
- Ricerca simile ai motori di ricerca del Web, però preservando la confidenzialità dei dati



*L'integrazione di jade-jxta con il sistema P2P ad agenti RAIS*

# **PARTE 1: L'INTEGRAZIONE JADE-JXTA- RAIS**

## 1. *jade-jxta*, Agosti-Scalise

- Partendo dai sorgenti di JADE e di JXTA, ha portato ad ottenere un'integrazione immediata e trasparente della piattaforma JADE all'interno di una rete P2P JXTA
- Perché?
  - ...per fornire ai MAS la potenza comunicativa delle reti *peer-to-peer*

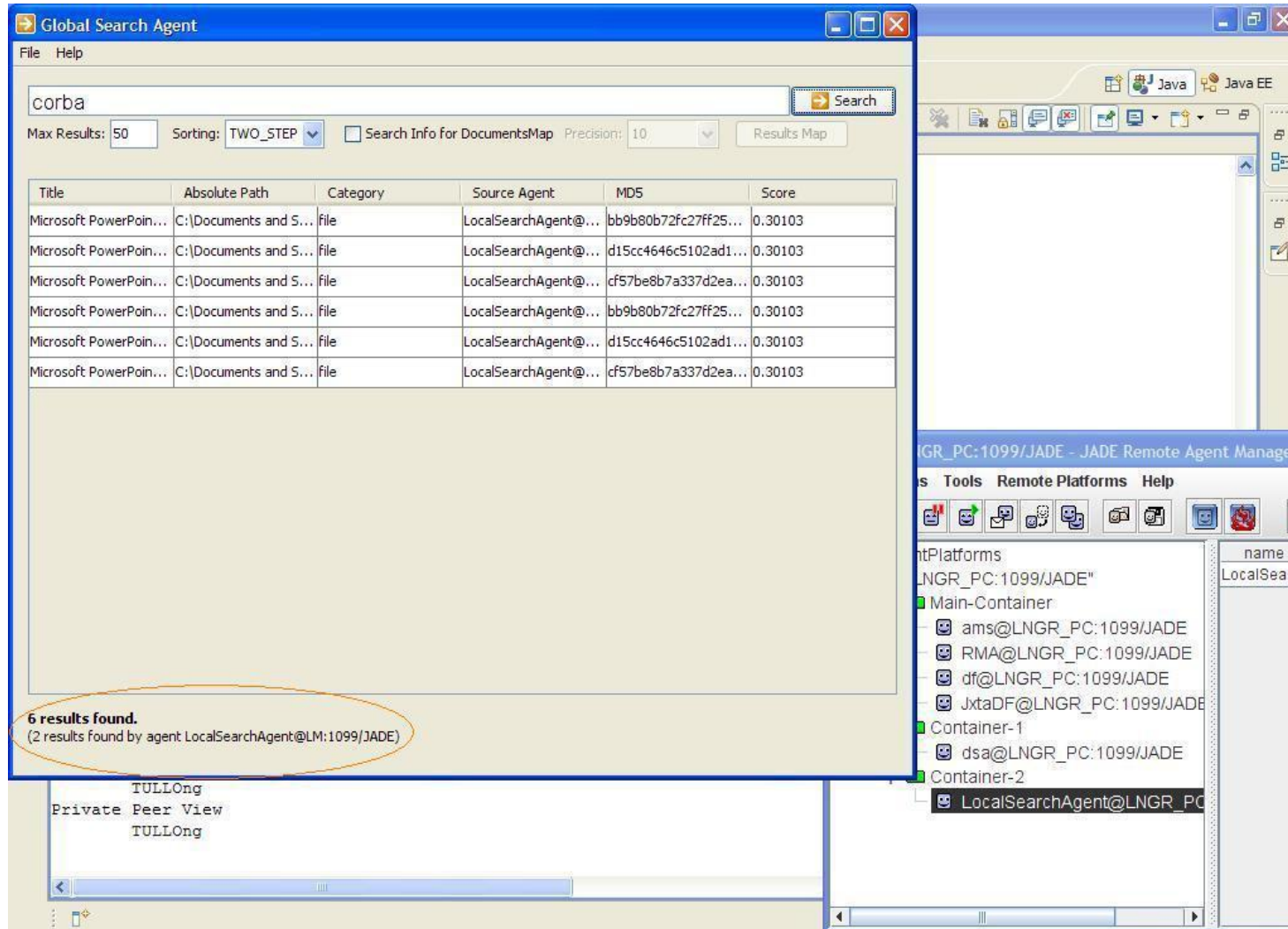
## 2. RAIS, Minari-Simonazzi

- Realizzazione di un sistema P2P per il *content-sharing*
- Perché?
  - ...per permettere una condivisione di risorse in modo veloce ed efficiente

- ◆ *jade-jxta*
  - Fornitura di servizi di base
  - Ulteriori funzionalità → modifiche al nucleo di *jade-jxta*: JxtaDF
- ◆ JxtaDF
  - Implementa alcuni servizi offerti solitamente dal DF
    - Si registra presso il DF e mantiene un riferimento al nodo JXTA
    - Permette ai *peer* di compiere le operazioni di *Publish/Subscribe*
- ◆ Modifiche al JxtaDF
  - Necessità di gestione di un maggior numero di agenti

- ◆ *DesktopSearchAgent (PA)*: interazione con l'utente
  - Corretta visualizzazione dei risultati
  - Migliorati i messaggi e gli eventi d'avviso
- ◆ *LocalSearchAgent (IF)*: agente di *Desktop Search*
  - Implementata la registrazione dell'agente sul JxtaDF
- ◆ *Behaviours*: definiscono le operazioni che gli agenti sono in grado di compiere e le loro modalità
  - Migliorata la gestione delle interazioni tra gli agenti
  - Implementato il controllo finale: attesa *peer* + *TimerBehaviour*

- ◆ Sistema **funzionante ed operativo**
- ◆ Creazione di un'unica libreria (*jade-jxta.jar*) contenente il nucleo operativo del nuovo substrato *peer-to-peer*
- ◆ *Startup* del sistema: **piattaforma JADE** registrata ad uno specifico **nodo** della rete **JXTA**



The image shows two overlapping windows from a JADE-based environment. The left window is the 'Global Search Agent' interface, and the right window is the 'JADE Remote Agent Manager'.

**Global Search Agent Window:**

- Search term: `corba`
- Max Results: 50
- Sorting: TWO\_STEP
- Search Info for DocumentsMap:
- Precision: 10
- Results Map:

Title	Absolute Path	Category	Source Agent	MD5	Score
Microsoft PowerPoin...	C:\Documents and S...	file	LocalSearchAgent@...	bb9b80b72fc27ff25...	0.30103
Microsoft PowerPoin...	C:\Documents and S...	file	LocalSearchAgent@...	d15cc4646c5102ad1...	0.30103
Microsoft PowerPoin...	C:\Documents and S...	file	LocalSearchAgent@...	cf57be8b7a337d2ea...	0.30103
Microsoft PowerPoin...	C:\Documents and S...	file	LocalSearchAgent@...	bb9b80b72fc27ff25...	0.30103
Microsoft PowerPoin...	C:\Documents and S...	file	LocalSearchAgent@...	d15cc4646c5102ad1...	0.30103
Microsoft PowerPoin...	C:\Documents and S...	file	LocalSearchAgent@...	cf57be8b7a337d2ea...	0.30103

6 results found.  
(2 results found by agent LocalSearchAgent@LM:1099/JADE)

Private Peer View  
TULLong

**JADE Remote Agent Manager Window:**

- Platform: LNGR\_PC:1099/JADE
- Main-Container
  - ams@LNGR\_PC:1099/JADE
  - RMA@LNGR\_PC:1099/JADE
  - df@LNGR\_PC:1099/JADE
  - JxtaDF@LNGR\_PC:1099/JADE
- Container-1
  - dsa@LNGR\_PC:1099/JADE
- Container-2
  - LocalSearchAgent@LNGR\_PC:1099/JADE

*Realizzazione dell'architettura astratta per la nuova versione di RAIS*

# **PARTE 2: L'ARCHITETTURA RAIS**

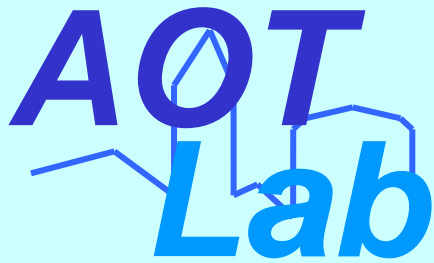
- ◆ “*Sviluppo di una architettura astratta per la realizzazione di applicazioni peer-to-peer*”, L. Gatti
- ◆ Scopo: sviluppare un’**architettura di alto livello**
  - ...per semplificare la realizzazione di applicazioni *peer-to-peer*
- ◆ Aspetto fondamentale: **astrazione** dell’architettura
  - ...per rendere possibili diverse implementazioni e modificarle facilmente
  - ...per poter utilizzare l’architettura su dispositivi *hardware* eterogenei

- ◆ Particolare sforzo di **ricerca** e di **studio**
  - Architettura composta da circa 200 classi ed interfacce Java
- ◆ **Riordinare e riorganizzare** i *package* del progetto secondo le esigenze specifiche di RAIS
- ◆ Creato un nuovo *root package* chiamato *main*
  - *MainUtils*: legge il file di configurazione
  - *P2PBeanFactory*: fa partire *container* e *beans* di Spring
  - *RAIS*: il vero e proprio *startup* del sistema
  - *RAISAction*: il *setting* iniziale del *peer* appena registrato

- ◆ Implementa la **configurazione** della piattaforma **JADE**
  - Acquisisce dal file di configurazione XML i *beans*
- ◆ **Proprietà** acquisite tramite *RAIS.xml*:
  - 1) *agents*: definisce quali agenti far partire al *boot*
  - 2) *dump*: visualizzazione delle opzioni durante il *boot*
  - 3) *localhost*: *setting* del *localhost*
  - 4) *localPort*: *setting* della *localPort*
  - 5) *main*: definisce se il *container* in questione è il *main*
  - 6) *MTPs*: definisce quale MTP utilizzare per JADE
  - 7) *platformID*: *setting* dell'ID univoco della piattaforma

- ◆ *Framework Spring*
  - **Configurazione** del sistema **semplice, versatile** ed **avanzata**
  - **Startup immediato** da riga di comando
    - ...lanciando il JAR appena creato e specificando il percorso del *file* di configurazione XML, è possibile far partire il sistema RAIS senza ulteriori supporti *software*
- ◆ Crea un'altra libreria (***RAIS-arch.jar***)
  - Sorgenti dell'**architettura** implementata per RAIS
  - **Sub-strato comunicativo** di *jade-jxta.jar*

- ◆ Raggiunti gli **obiettivi sperimentali** prefissati
  - Prima parte: **l'integrazione JADE-JXTA-RAIS**
    - ...integrate le caratteristiche di intelligenza, autonomia e collaborazione dei MAS, con quelle di condivisione ed efficienza comunicativa delle reti *peer-to-peer*
  - Seconda parte: **l'architettura RAIS**
    - ...resa semplice ed immediata la configurazione e l'implementazione dell'applicazione tramite un *file* scritto in linguaggio XML
- ◆ **Sviluppi futuri**
  - Ampliare il numero di servizi offerti dal sistema
  - Pubblicazione ed invocazione dei servizi dei *peer*, ispirandosi agli *standard* definiti dai *Web Services*



**Agent and Object Technology LAB**  
**Dipartimento di Ingegneria dell'Informazione**  
**Facoltà di Ingegneria**  
**Università degli Studi di Parma**



**SVILUPPO DI UN'INFRASTRUTTURA  
*PEER-TO-PEER* A SUPPORTO DI UN  
SISTEMA PER LA CONDIVISIONE DI  
INFORMAZIONI**

**Grazie per l'attenzione!**

**FINE**

**Relatore**

*Chiar.mo Prof. Ing. A. Poggi*

**Correlatore**

*Dott. Ing. M. Mari*

**Tesi di Laurea Specialistica di**

*Michele Longari*