

**Progetto**  
**Laboratorio Programmazione di rete**  
**Mini-KaZaA**

Andrea Di Grazia, Massimiliano Giovine

Anno Accademico 2008 - 2009

# Indice

<b>1</b>	<b>Introduzione</b>	<b>2</b>
1.1	Una panoramica generale.	2
1.2	La rete Mini-KaZaA.	2
<b>2</b>	<b>Bootstrap Server</b>	<b>4</b>
2.1	Il Bootstrap server in generale	4
2.2	Entriamo nel dettaglio	4
2.3	La classe NodeInfo	5
2.4	L'interfaccia grafica.	6
<b>3</b>	<b>Mini-KaZaA Client</b>	<b>7</b>
3.1	Mini-KaZaA Client in generale	7
<b>4</b>	<b>Ordinary Node</b>	<b>8</b>
4.1	Scelta del SN al quale connettersi	8

# Capitolo 1

## Introduzione

### 1.1 Una panoramica generale.

Il progetto Mini-KaZaA mira allo sviluppo di un sistema p2p per lo scambio di file su WAN, ispirato alla più famosa rete p2p KaZaA. Ogni peer<sup>1</sup> partecipante alla rete Mini-KaZaA condivide un insieme di files con gli altri peer connessi e può ricercare file all'interno della rete e effettuarne il download.

Mini-KaZaA prevede due tipi diversi di peer:

- **Super Nodes (SN):** i SN hanno il compito di gestire le comunicazioni all'interno della rete;
- **Ordinary Nodes (ON):** gli ON hanno responsabilità più limitate, condividono e cercano file nella rete.

Nella rete Mini-KaZaA è prevista anche un'altra entità chiamata **Bootstrap Servers** che contiene la lista di tutti i peer connessi alla rete e dalla quale ogni nodo che desidera entrare a far parte della rete può scaricare la lista aggiornata di tutti i SN presenti.

La rete si costruisce automaticamente dai vari peer secondo un preciso schema e si mantiene stabile grazie a processi automatizzati che lavorano in background, completamente trasparenti all'utente.

### 1.2 La rete Mini-KaZaA.

Ogni peer della rete Mini-KaZaA viene configurato esplicitamente dall'utente al primo avvio come SN o come ON. Successivamente non sarà possibile cambiare tale configurazione.

---

<sup>1</sup>Ogni nodo della rete è un pari all'interno del network poichè funziona sia da client, per ciò che concerne la ricerca e il download dei file, sia da server per la condivisione dei file o lo smistamento delle ricerche nella rete p2p.

Al momento della connessione alla rete ogni peer, SN o ON, contatta un Bootstrap server che gli fornisce la lista aggiornata di SN presenti in quel momento all'interno della rete.

Un ON sceglie il migliore SN per lui e si connette ad esso. Un SN mantiene in memoria la lista di riferimenti a SN che gli servirà, in un secondo momento, per smistare le interrogazioni. Gli SN, inoltre, avendo un sistema dinamico di connessione ai pari SN esplorano a ogni interrogazione porzioni nuove della rete in modo tale che vi siano il meno possibile porzioni isolate della rete.

## Capitolo 2

# Bootstrap Server

### 2.1 Il Bootstrap server in generale

Il Bootstrap server ha il compito di tenere un indice di tutti i SN presenti nella rete che abbiano una certa affidabilità. Per poter fare questo fornisce un servizio di RMI<sup>1</sup> tramite il quale i SN si possono iscrivere alla rete Mini-KaZaA e richiedere liste aggiornate.

Gli aggiornamenti vengono spediti a ogni SN presente nella lista del Bootstrap server tramite un sistema di *callbacks*<sup>2</sup>

Il Bootstrap server deve fornire questo servizio anche agli ON che vogliono entrare nella rete per poter individuare il “miglior”<sup>3</sup> SN al quale potersi connettere. Per questa ragione si è reso necessario indicizzare anche gli ordinary node all’interno del bootstrap server.

### 2.2 Entriamo nel dettaglio

Il bootstrap server si avvia dal main situato all’interno del file `BootstrapService.java` e subito crea il servizio RMI sulla porta 2008 da mettere a disposizione per i vari nodi della rete con le seguenti istruzioni:

```
1 Registry registry = LocateRegistry.createRegistry(2008);
2 BootstrapServer bss = new BootstrapServer(g, sn_list);
3
4 BootstrapServerInterface stub =
5     (BootstrapServerInterface)
6     UnicastRemoteObject.exportObject(bss, 2008);
```

---

<sup>1</sup>Remote Method Invocation, tramite questo servizio è possibile invocare metodi che si trovano su una macchina diversa da quella in cui si trova la chiamata a procedura.

<sup>2</sup>Ogni nodo mette a disposizione del Bootstrap server alcune chiamate di procedura che, richiamate, consentono di inviare aggiornamenti.

<sup>3</sup>Spiegheremo nella sezione 4.1 i parametri secondo i quali ogni ON sceglie un SN al quale connettersi.

```

7
8
9 SupernodeCallbacksImpl client_impl =
10     new SupernodeCallbacksImpl(
11         new SupernodeList(),
12         new NodeConfig());
13
14 SupernodeCallbacksInterface client_stub =
15     (SupernodeCallbacksInterface)
16     UnicastRemoteObject.exportObject( client_impl, 2008);

```

Con le prime istruzioni il Bootstrap server mette a disposizione tutti i metodi della classe `BootStrapServerInterface` che sono i seguenti:

```

1 public boolean addSuperNode(NodeInfo new_node) throws RemoteException;
2 public boolean removeSuperNode(NodeInfo new_node) throws RemoteException;
3 public boolean addOrdinaryNode(NodeInfo new_node) throws RemoteException;
4 public boolean removeOrdinaryNode(NodeInfo new_node) throws RemoteException;
5 public ArrayList<NodeInfo> getSuperNodeList() throws RemoteException;

```

Con le istruzioni alla riga 9 e 14 registra l'interfaccia di callback `SupernodeCallbacksInterface` che ha i seguenti metodi:

```

1 public void notifyMeAdd(NodeInfo new_node) throws RemoteException;
2 public void notifyMeRemove(NodeInfo new_node) throws RemoteException;

```

Per poter trasmettere e ricevere le informazioni riguardanti i vari nodi della rete, il client Mini-KaZaA e il Bootstrap server usano una classe serializzabile che si chiama `NodeInfo` che analizziamo nella sezione 2.3

## 2.3 La classe NodeInfo

La classe `NodeInfo` si trova nel package `lpr.minikazaa.bootstrap` ma viene utilizzata da tutto il pacchetto Mini-KaZaA per poter inviare nella rete le informazioni relative ai nodi.

Questa classe rappresenta le informazioni utili di un nodo con le sue variabili private.

```

1 private InetAddress ia_node;
2 private int door;
3 private String id_node;
4 private String username;
5 private SupernodeCallbacksInterface stub;
6 private long ping;
7 private boolean is_sn;

```

Le prime tre variabili private riguardano tutte le informazioni di rete dei nodi, ovvero l'indirizzo IP, la porta di connessione e un id univoco ottenuto facendo la concatenazione della rappresentazione decimale dell'indirizzo IP.

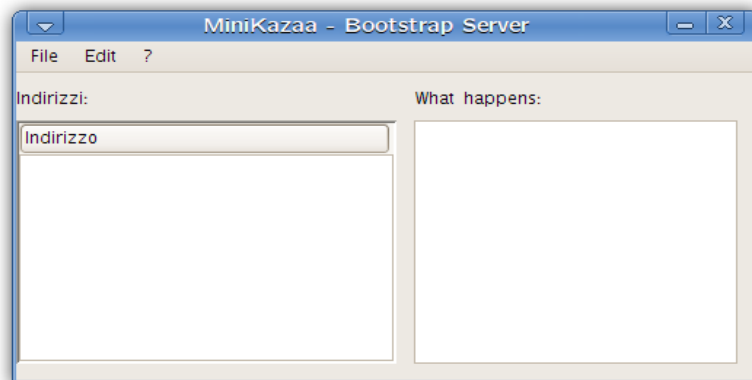


Figura 2.1: Interfaccia grafica del Bootstrap server

La variabile privata `private SupernodeCallbacksInterface stub` rappresenta l'interfaccia per le callbacks che viene messa a disposizione dal nodo. In questo modo il bootstrap server può richiamare direttamente l'interfaccia delle callback di ogni nodo interessato all'aggiornamento.

L'ultima variabile, `private boolean is_sn` indica se il nodo al quale si riferiscono le informazioni, all'interno della rete ricopre il ruolo di SN o di ON.

La classe contiene tutti i metodi `set` e `get` per poter assegnare valori alle variabili private e per poterne ricavare il contenuto in qualsiasi momento.

## 2.4 L'interfaccia grafica.

L'interfaccia grafica fornisce le informazioni riguardo a ciò che avviene all'interno della rete.

Uno screenshot dell'interfaccia grafica principale del Bootstrap server si può vedere in Figura 2.1.

Nella parte a sinistra dell'interfaccia vengono inseriti gli *id* di tutti i nodi che si connettono alla rete.

Nella parte destra, invece, vengono visualizzati dei messaggi che spiegano cosa avviene all'interno della rete.

## Capitolo 3

# Mini-KaZaA Client

### 3.1 Mini-KaZaA Client in generale



## Capitolo 4

# Ordinary Node

### 4.1 Scelta del SN al quale connettersi