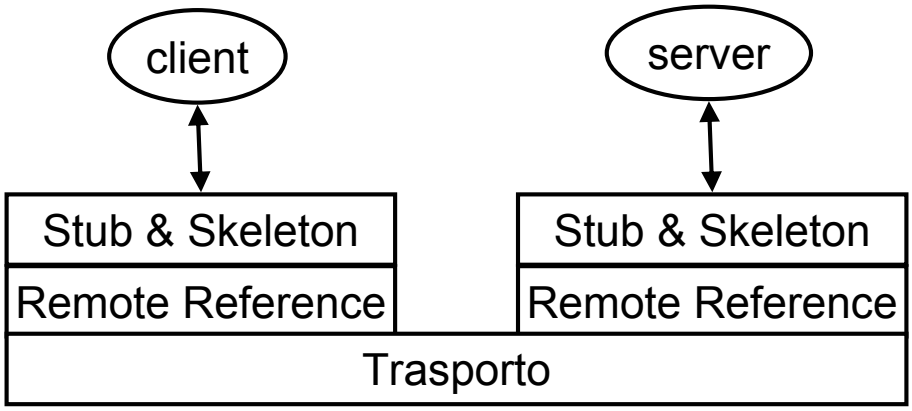
Chiamata remota di metodi

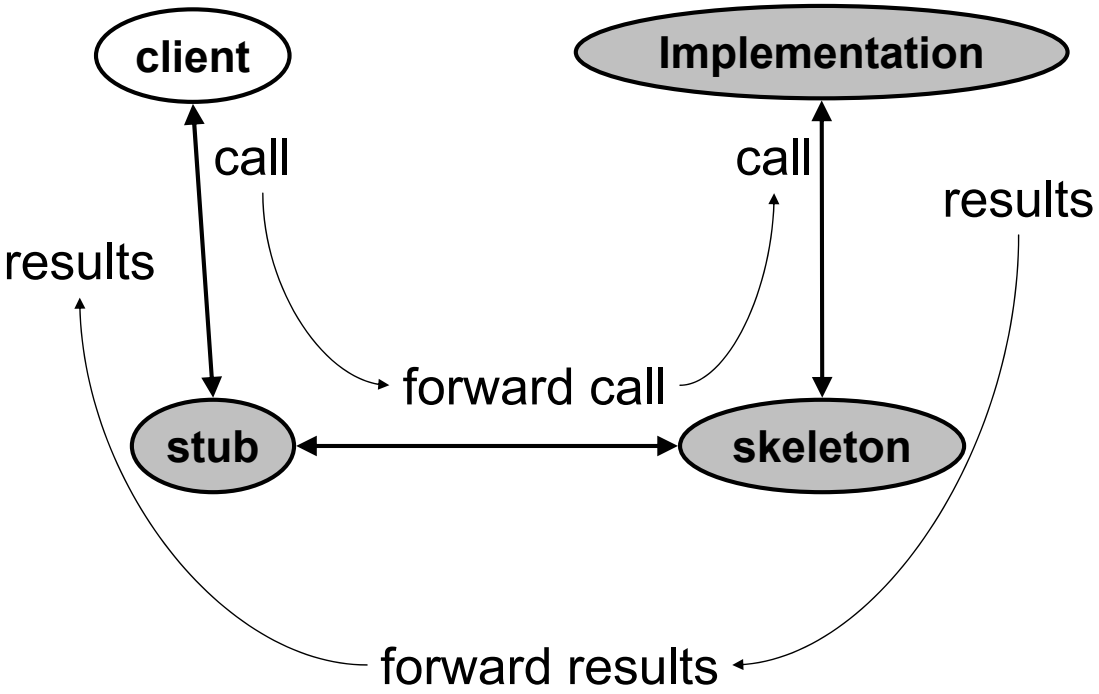
- Architettura di Java RMI
- Esecuzione di una Java RMI

Architettura di RMI



Ciascun livello può essere sostituito o esteso senza modificare gli altri

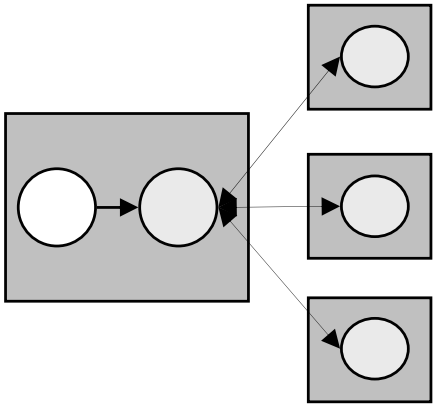
Il livello Stub & Skeleton



Il Livello Remote Reference

Il livello Remote Reference implementa la semantica di Java RMI

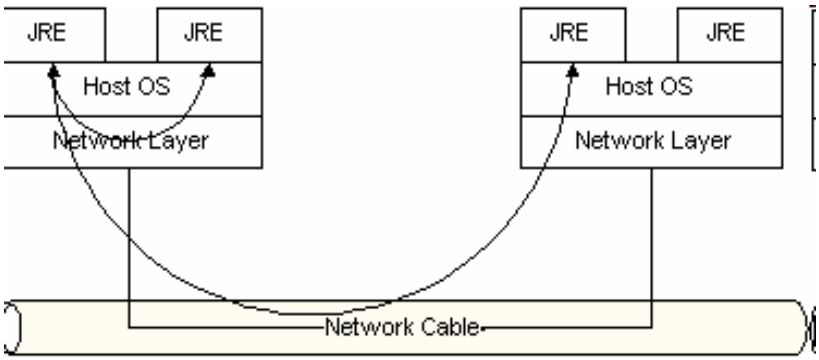
- Oggetti unicast, volatili, at-most-once (JDK 1.1)
- Oggetti attivabili (JDK 2)
- Altre semantiche possibili: multicast



Il proxy

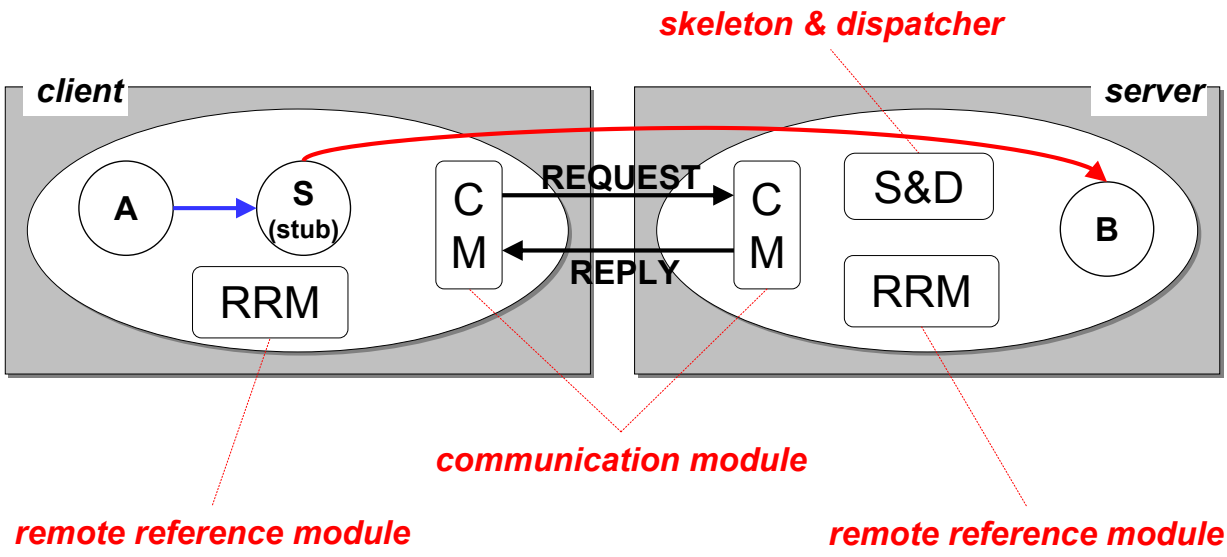
- inoltra simultaneamente la richiesta ad un insieme di implementazioni ed
- accetta la prima risposta

Il livello di trasporto



- Tutte le connessioni sono TCP/IP
- Su TCP/IP, Java RMI implementa Java Remote Method Protocol (JRMP)

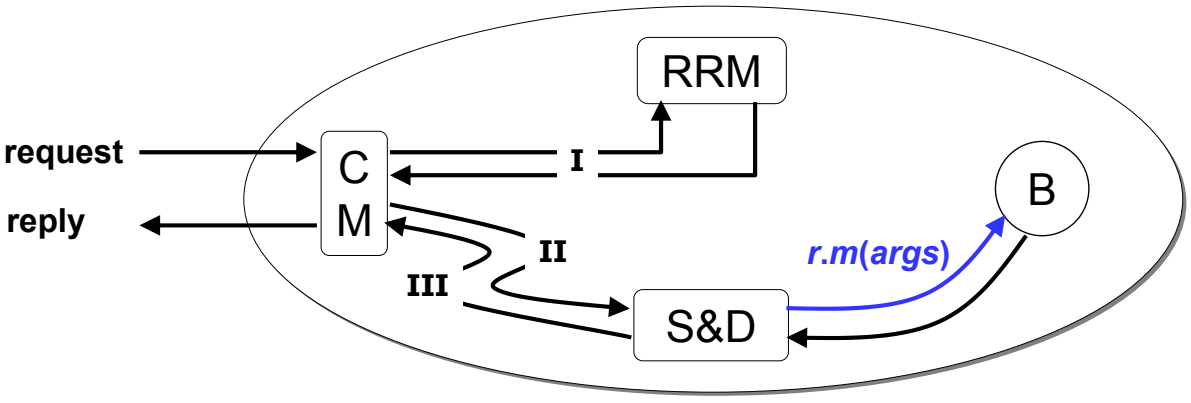
Implementazione di RMI



RMI processing

- Siano:
  - **B** un oggetto remoto
  - **m** un metodo della sua interfaccia remota
  - **R** il riferimento remoto a **B**
  - **S** uno stub che incapsula **R**
  - **r** il riferimento locale al server dell'oggetto remoto **B**
- RMI processing deve affrontare due aspetti:
  - l'invocazione (attraverso **S**) del metodo remoto **m** di **B**
  - l'oggetto remoto **B** viene trasmesso come argomento o valore di ritorno

RMI Processing



I. Alla ricezione del **request message**, CM invia il riferimento remoto **R** ad RRM, il quale reperisce in ROT il corrispondente riferimento locale **r** all'oggetto B e lo ritorna a CM

II. CM determina il modulo S&D sulla base del valore del campo Remote Class/Interface del riferimento remoto **R** e passa a tale modulo la coppia (**request message, r**)

(*continua*)

RMI processing

III. Alla ricezione della coppia (**request message, r**), il modulo S&D esegue le seguenti azioni:

   a. determina il metodo **m** da invocare sulla base del valore del campo **Method Identifier** del **Request Message**

   b. esegue l'unmarshaling degli argomenti **args** contenuti nel campo **Arguments** del **Request Message**

   c. invoca il metodo **results = r.m(args)**

   d. esegue il marshalling dei valori di ritorno **results**

   e. prepara il **Reply Message** e lo passa a CM

- **N.B.: l'elaborazione di una chiamata remota assume che il mapping R→r sia già disponibile nella tabella ROT di RRM**

RMI processing

- **Un oggetto remoto può essere trasmesso come argomento o valore di ritorno**
  - **L'oggetto viene trasmesso come argomento (request message)**
    - Lo stub trasmette al modulo RRM il riferimento locale **r** all'oggetto il quale ritorna il riferimento remoto **R** da inserire nel messaggio;
  - **L'oggetto viene trasmesso come valore di ritorno (reply message)**
    - Il S&D trasmette al modulo RRM il riferimento locale **r** all'oggetto il quale ritorna il riferimento remoto **R** da inserire nel messaggio;
    - Se è la prima volta che l'oggetto viene trasmesso, il riferimento **R** non è presente nella tabella ROT, il modulo RRM perciò crea un riferimento remoto **R** per l'oggetto ed inserisce la coppia (**R, r**) nella tabella ROT

(*continua*)

RMI modules

- **Proxy/stub**
  - Il proxy/stub rende la RMI trasparente all'oggetto cliente
  - Il proxy/stub ha la stessa interfaccia dell'oggetto remoto ed inoltra a questi l'invocazione di metodo
  - Il proxy/stub nasconde il riferimento remoto ed esegue marshalling ed unmarshalling (lato client)
  - C'è un proxy/stub per ogni oggetto remoto di cui il processo ha un riferimento remoto
- **Il Communication Module (CM)**
  - esegue il protocollo request-reply e realizza la semantica (at-most-once, at-least-once, …) della RMI

RMI modules

- **Il Remote Reference Module (RRM)**
  - RRM ha il compito di creare riferimenti remoti
  - RRM ha anche il compito di tradurre un riferimento remoto in un riferimento locale (ad uno stub o ad un oggetto remoto) e viceversa
  - A questo scopo, RRM mantiene la **Remote Object Table** (**ROT**)

Remote object table

riferimento remoto	riferimento locale
R	r

riferimento a:
- oggetto remoto (server)
- stub (client)

RMI modules

- **Il modulo skeleton & dispatcher (S&D)**
  - Un modulo per ciascuna classe remota
  - Il modulo skeleton & dispatcher ha il compito di:
    - determinare il metodo da invocare;
    - eseguire l'unmarshalling degli argomenti;
    - eseguire l'invocazione locale del metodo;
    - eseguire il marshalling dei risultati;
    - preparare il messaggio di reply
  - Il modulo skeleton & dispatcher è generato dinamicamente a partire dall'interfaccia remota

RMI modules

- Il binder
  - il binder è un servizio (directory service) separato che mantiene la corrispondenza tra nomi (stringhe) e riferimenti remoti
  - un server registra un proprio oggetto remoto sotto un certo nome
  - I client ottengono i riferimenti remoti a tale oggetto facendo una ricerca per nome