

RPC

Programmazione in Ambienti Distribuiti

marco.torchiano@polito.it

V 1.1 © Marco Torchiano 2006

RPC - Runtime

- Attivazione del server
 - ◆ Registra le procedure
 - ◆ Si mette in ascolto
 - Avvio del client
 - ◆ Crea la connessione
 - ◆ Effettua le chiamate
 - ◆ Chiude la connessione
-

RPC - Client

- Effettua la connessione

```
CLIENT *cl;  
cl = clnt_create("host", DATE_PROG,  
                DATE_VERS, "udp");
```

- Chiama la procedura remota

```
long *lresult;  
lresult = bin_date_1(NULL, cl);
```

- Chiude la connessione

```
clnt_destroy(cl);
```

RPC - Server

- Implementazione della procedura remota

```
long *bin_date_1(  
    void* ignore_it,  
    struct svc_req *rqstp){  
    static long timeval;  
    time(&timeval);  
    return (&timeval);  
}
```

Conversione in RPC

- Identificare le funzioni remote
 - Costruire il file XDR con la dichiarazione delle funzioni e dei tipi di dato
 - Usare rpcgen
 - Etc.
-

Conversione in RPC

- Due tipi di funzioni remote
 - ♦ Mute: non generano output
 - Nessun problema nella conversione
 - ♦ Parlanti: generano output
 - L'output ragionevolmente deve finire al client
 - Occorre aggiungere un parametro di uscita
-

Parametri multipli

- Le funzioni RPC accettano un solo parametro
 - ♦ Per passare più parametri occorre definire una struct contenente i parametri
- Analogamente se si devono restituire più valori

Chiamate a funzioni RPC

- Chiamata diretta
 - ♦ Si modifica il client:
 - ♦ Da: `result = bin_date();`
 - ♦ A: `Iresult = bin_date_1(NULL, cl);`
- Uso di wrapper
 - ♦ Si crea una funzione che richiama l'RPC
 - ♦

```
long bin_date(){
    long* Iresult = bin_date_1(NULL, cl);
    return *Iresult;
}
```
 - ♦ NB: qualcuno deve inizializzare il client handle