RPC: REMOTE PROCEDURE CALL

DR. ACHRAF EL ALLALI

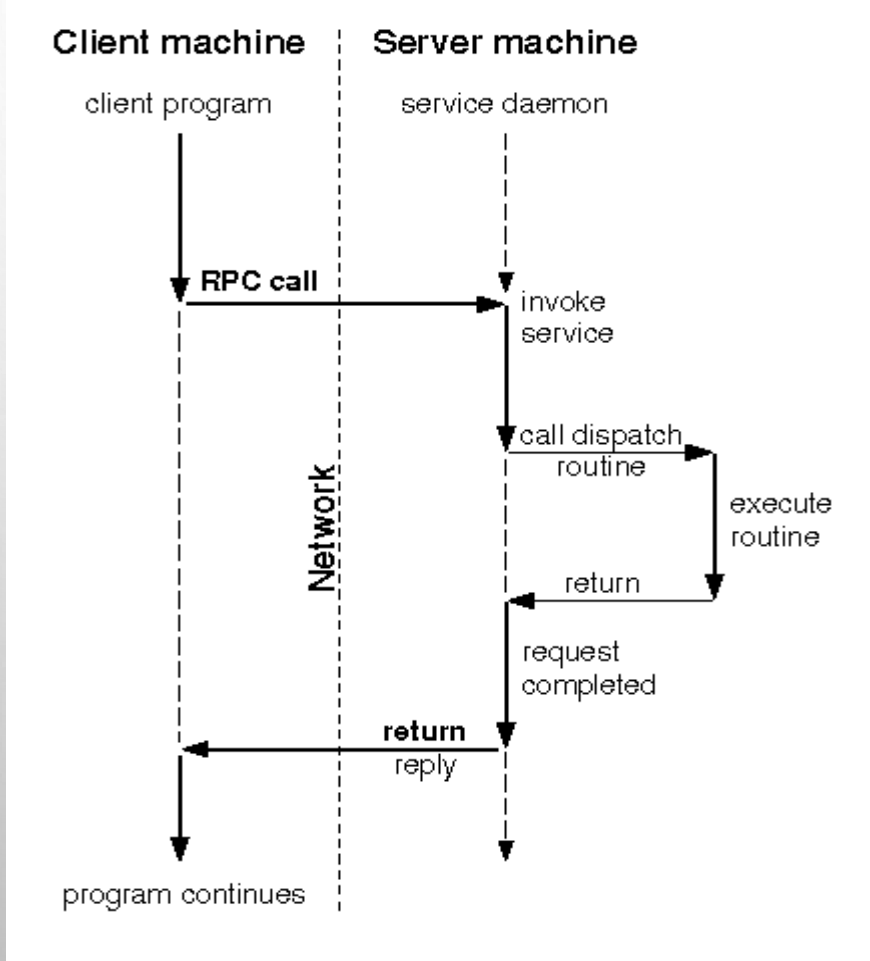
BASED ON RPC SURVEY BY YONG LI

REMOTE PROCEDURE CALL (RPC)

- A POWERFUL TECHNIQUE FOR CONSTRUCTING DISTRIBUTED, CLIENT/SERVER BASED APPLICATIONS.

- RPC IS ANALOGOUS TO A FUNCTION CALL.

RPC MECHANISM



RPC DEVELOPMENT

- SPECIFY THE PROTOCOL FOR CLIENT SERVER COMMUNICATION
- DEVELOP THE CLIENT PROGRAM
- DEVELOP THE SERVER PROGRAM

THE RPCGEN PROTOCOL COMPILER

- RPCGEN PROVIDES PROGRAMMERS A SIMPLE AND DIRECT WAY TO WRITE DISTRIBUTED APPLICATIONS
- THE OUTPUT OF RPCGEN IS:
  - A HEADER FILE OF DEFINITIONS COMMON TO THE SERVER AND THE CLIENT
  - A SET OF XDR ROUTINES THAT TRANSLATE EACH DATA TYPE DEFINED IN THE HEADER FILE
  - A STUB PROGRAM FOR THE SERVER
  - A STUB PROGRAM FOR THE CLIENT
  - A TIME-OUT FOR SERVERS (OPTIONALLY)
  - C-STYLE ARGUMENTS PASSING ANSI C-COMPLIANT CODE (OPTIONALLY)
  - (OPTIONALLY) DISPATCH TABLES THAT THE SERVER CAN USE TO CHECK AUTHORIZATIONS AND THEN INVOKE SERVICE ROUTINES.

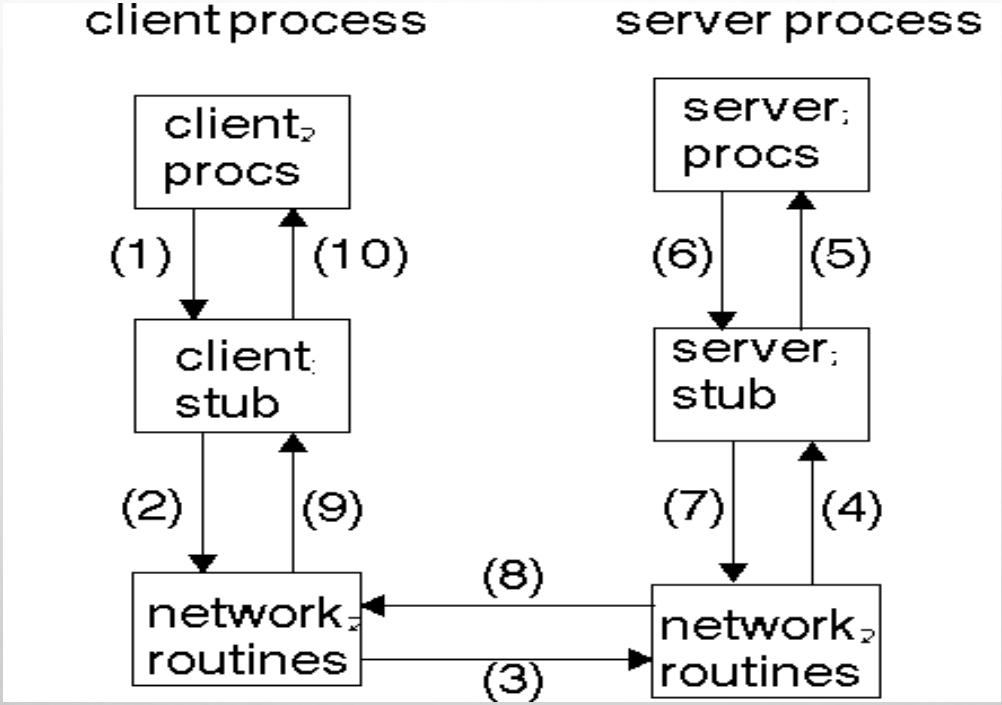
RPC LANGUAGE SPECIFICATION

program-def: **"program" identifier "{"** **version-def** **version-def *** **"}" "=" constant ";"**	**procedure-def:** **type-specifier identifier "(" type-specifier ")"** **"=" constant ";"**
version-def: "version" identifier "{" procedure-def procedure-def * "}" "=" constant ";"	

PROGRAM NUMBER

- 0 - 1FFFFFFF DEFINED BY SUN
- 20000000 - 3FFFFFFF DEFINED BY USER
- 40000000 - 5FFFFFFF TRANSIENT
- 60000000 - 7FFFFFFF RESERVED
- 80000000 - 9FFFFFFF RESERVED
- A0000000 - BFFFFFFF RESERVED
- C0000000 - DFFFFFFF RESERVED
- E0000000 - FFFFFFFF RESERVED

STUBS


client process
server process
client procs
server procs
(1)
(10)
(6)
(5)
client stub
server stub
(2)
(9)
(7)
(4)
(8)
network routines
network routines
(3)

DATA REPRESENTATION

- **XDR**(E**X**TERNAL **D**ATA **R**EPRESENTATION), A PROTOCOL FOR THE MACHINE-INDEPENDENT DESCRIPTION AND ENCODING OF DATA
- DATA ABSTRACTION NEEDED FOR MACHINE INDEPENDENT COMMUNICATION
- THE CLIENT AND SERVER NEED NOT BE MACHINES OF THE SAME TYPE
- TRANSFERRING DATA BETWEEN DIFFERENT COMPUTER ARCHITECTURES

VALID DATA TYPES SUPPORTED BY XDR

- INT
- UNSIGNED INT
- LONG
- ENUM
- BOOL
- FLOAT
- DOUBLE
- TYPEDEF
- STRUCTURE
- FIXED ARRAY
- STRING (NULL TERMINATED CHAR *)

EXAMPLE

```
/* PI.X: REMOTE PI CALCULATION PROTOCOL */
 PROGRAM PIPROG {
    VERSION CALCU_PIVERS {
        DOUBLE CALCU_PI() = 1;
    } = 1;
} = 0X39876543;
```

CONVERT LOCAL PROCEDURE INTO REMOTE PROCEDURE

- COMPILE A .X FILE USING
  - **RPCGEN –A –C PI.X**
  - WHERE: OPTION –A TELLS RPCGEN TO GENERATE ALL OF THE SUPPORTING FILES
  - OPTION –C INDICATES ANSI C IS USED
- OUTPUT
  - PI_CLNT.C -- THE CLIENT STUB
  - PI_SVC.C -- THE SERVER STUB
  - PI.H -- THE HEADER FILE THAT CONTAINS ALL OF THE XDR TYPES GENERATED FROM THE SPECIFICATION
  - MAKEFILE.PI -- MAKEFILE FOR COMPILING ALL OF THE CLIENT AND SERVER CODE
  - PI_CLIENT.C -- CLIENT SKELETON, NEED TO BE MODIFIED
  - PI_SERVER.C – SERVER SKELETON, NEED TO BE MODIFIED

PI_SERVER.C

```
/*
 * pi_server.c: implementation of the remote procedure "calcu_pi"
 *
 * The formula is:
 * (pi / 4) = 1 - 1/3 + 1/5 - 1/7 ...
 */
#include <rpc/rpc.h> /* always needed */
#include "pi.h"                    /* generated by rpcgen compiler */
/** Remote verson of "calcu_pi" */
```

```
double * calcu_pi_1_svc(void *argp, struct svc_req *rqstp){
   static double  pi;

   double sum = 0;
   int i;
   int sign;

   for (i=1; i<10000000; i++ ){
                sign = (i+1) % 2;
                if ( sign == 0 )
                   sign = 1;
                else
                   sign = -1;
                sum += 1 / (2*(double)i -1) * (double)sign;
   }
   pi = 4 * sum;
   return (&pi);
}
```

REMOTE DROCEDURE

- TAKES POINTERS TO THEIR ARGUMENTS INSTEAD OF ARGUMENTS THEMSELVES
- IT RETURNS A POINTER TO A DOUBLE INSTEAD OF A DOUBLE ITSELF.
- THE NAME IN THE PROGRAM DEFINITION (HERE `CALCU_PI') IS CONVERTED TO ALL LOWER-CASE LETTERS, AN UNDERBAR (`_') IS APPENDED TO IT, AND FINALLY THE VERSION NUMBER (HERE `1') IS APPENDED.
- WHEN **RPCGEN** IS USED, IT IS ESSENTIAL TO HAVE RESULT (IN THIS EXAMPLE) DECLARED AS STATIC.

PI_CLIENT.C

```
/** pi_client.c, remote version of client program
 */

#include <stdio.h>
#include <rpc/rpc.h>   /* always needed */
#include "pi.h"        /* generated by rpcgen*/

main(int argc, char *argv[])
{
   CLIENT *clnt;
   double  *result_1;
   char *host;
   char *  calcu_pi_1_arg;

   /* must have two arguments, including server host name */
   if (argc < 2) {
                        printf("usage:  %s server_host\n", argv[0]);
                        exit(1);
   }

   host = argv[1]; /* server host name */

   /** Create client "handle" used for calling PIPROG
    *  on the server designated on the command line.
         */
   clnt = clnt_create(host, PIPROG, CALCU_PIVERS, "tcp");

   if (clnt == (CLIENT *) NULL) {
/* Couldn't establish connection with server
                        *  Print error message and die.
     */
                        clnt_pcreateerror(host);
                        exit(1);
   }

   /* call remote procedure on the server side */
   result_1 = calcu_pi_1((void *)&calcu_pi_1_arg, clnt);

   if (result_1 == (double *) NULL) {
/** An error occurred while calling the server.
*  Print error message and die.
*/
                        clnt_perror(clnt, "call failed");
exit(1);
   }

   /** Okay, we successfully called the remote procedure.
   * print the pi value
   */
   printf("PI is %f\n" , *result_1);
   clnt_destroy(clnt);
   exit(0);
}
```

CLIENT SIDE

- CLIENT *HANDLE* IS CREATED USING THE RPC LIBRARY ROUTINE CLNT_CREATE(). THIS CLIENT HANDLE WILL BE PASSED TO THE STUB ROUTINES THAT CALL THE REMOTE PROCEDURE.
- THE LAST PARAMETER TO CLNT_CREATE() IS ``TCP".
- THE REMOTE PROCEDURE CALCU_PI_1() IS CALLED EXACTLY THE SAME WAY AS IT IS A LOCAL PROCEDURE EXCEPT FOR THE INSERTED CLIENT HANDLE AS THE SECOND ARGUMENT.
- THE RPC MECHANISM ITSELF CAN FAIL
  - THE REMOTE PROCEDURE RETURNS WITH A NULL
- THERE CAN BE AN ERROR IN THE EXECUTION OF THE REMOTE PROCEDURE.
  - THE DETAILS OF ERROR REPORTING ARE APPLICATION DEPENDENT. HERE, THE ERROR IS BEING REPORTED VIA *RESULT.

COMPILE/RUN

- COMPILING
  - $ RPCGEN –C –A PI.X
  - $ GCC PI_CLIENT.C PI_CLNT.C -O PI_CLIENT -LNSL
  - $ GCC PI_SERVER.C PI_SVC.C -O PI_SERVER –LNSL
- RUNNING
  - SHELL ON THE REMOTE SYSTEM:
    - REMOTE$ PI_SERVER
  - THEREAFTER, A USER ON *LOCAL* CAN CALCULATE PI AS FOLLOWS:
    - LOCAL$ PI_CLIENT REMOTE