DISTRIBUIRANI SISTEMI

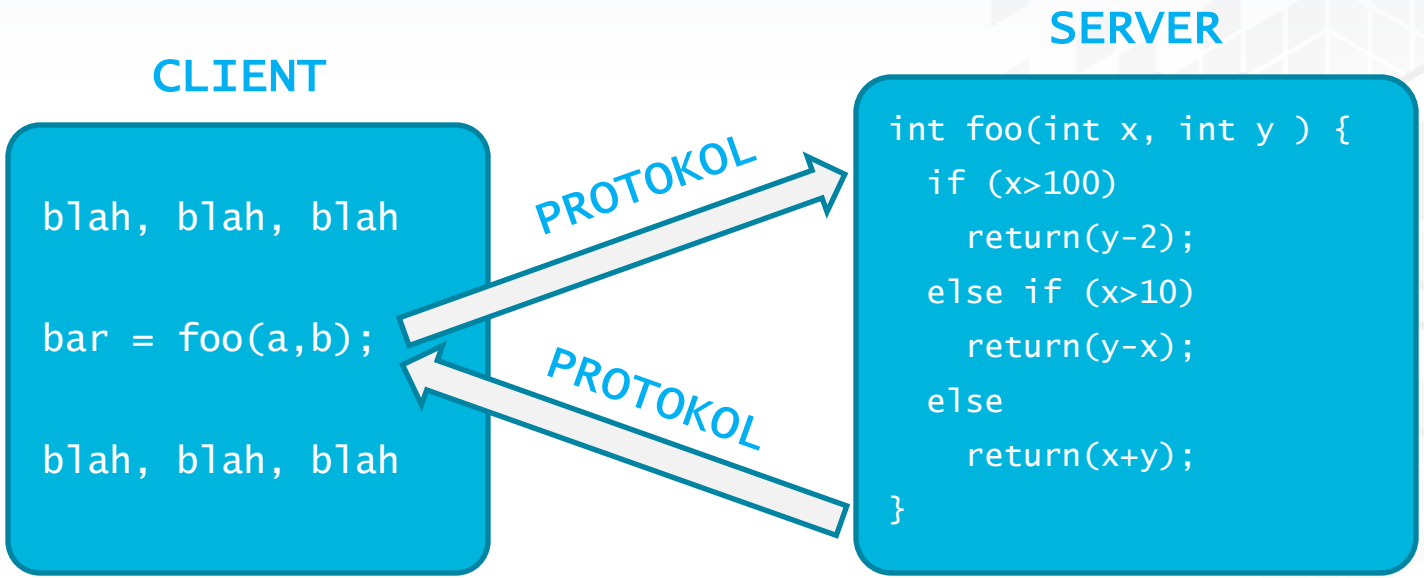
RPC

RPC

REMOTE PROCEDURE CALL

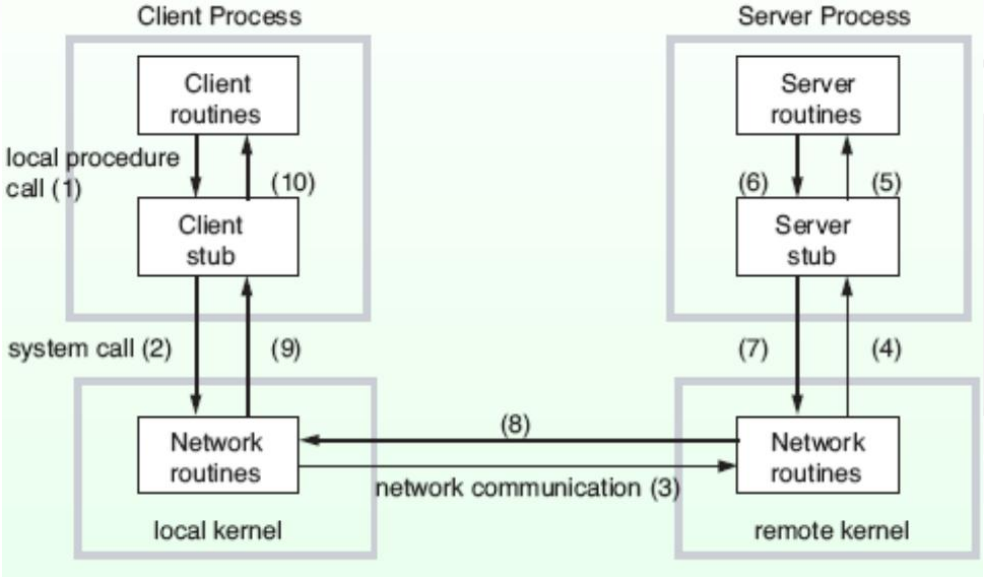
Poziv udaljene procedure (Remote Procedure Call-RPC)

RPC predstavlja mehanizam za poziv procedura na udaljenim mašinama. Klijent je program u okviru koga se poziva procedura a server onaj koji nudi procedure



RPC komunikacija

Da bi poziv udaljene procedure bio moguć i izgledao kao poziv lokalne procedure koriste se stub. Stub funkcija ima isti interfejs kao udaljena funkcija. Stub-ovi sa klijent (server) strane su zaduženi za pakovanje(marshalovanje) parametara(rezultata) u mrežnu poruku i slanje preko mreže kao i za raspakovanje (unmarshaling) rezultata (parametara). Tom prilikom obavljaju konverzije iz lokalnog u standardni (i obrnuto) način predstavljanja podataka.



Sun RPC

Postoji veliki broj RPC implemenatcija. Jedna od najčešće korišćenih je Sun RPC. Programski jezici ne podržavaju RPC. Zato se koristi Sun RPC pre-kompajler koji generiše neophodne stub-ove sa klijent i server strane. Sun RPC kompajler se zove rpcgen.

Sun RPC obezbeđuje jezik za definiciju interfejsa IDL. IDL omogućava deklaraciju procedura slično deklaraciji prototipa funkcija u C-u. IDL fajl može sadržati definicije tipova, deklaracije konstanti, deklaracije procedura I druge informacije potrebne za korektno pakovanje i raspakovanje parametara.

IDL (interface definition language)

```
type definitions
  program identifier {
    version version_id {
      procedure list
    } = value2;
    …
  } = value1;

 Npr.
 program PROG {
   version VER {
     void PROC_A(int) = 1;
   }=1;
 } = 0x3a3afeeb;
```

identifier-ime programa

value1-broj programa

version_id-ime verzije

value2-broj verzije

Broj programa je 32-bitna vrednost sa sledećim ograničenjima:

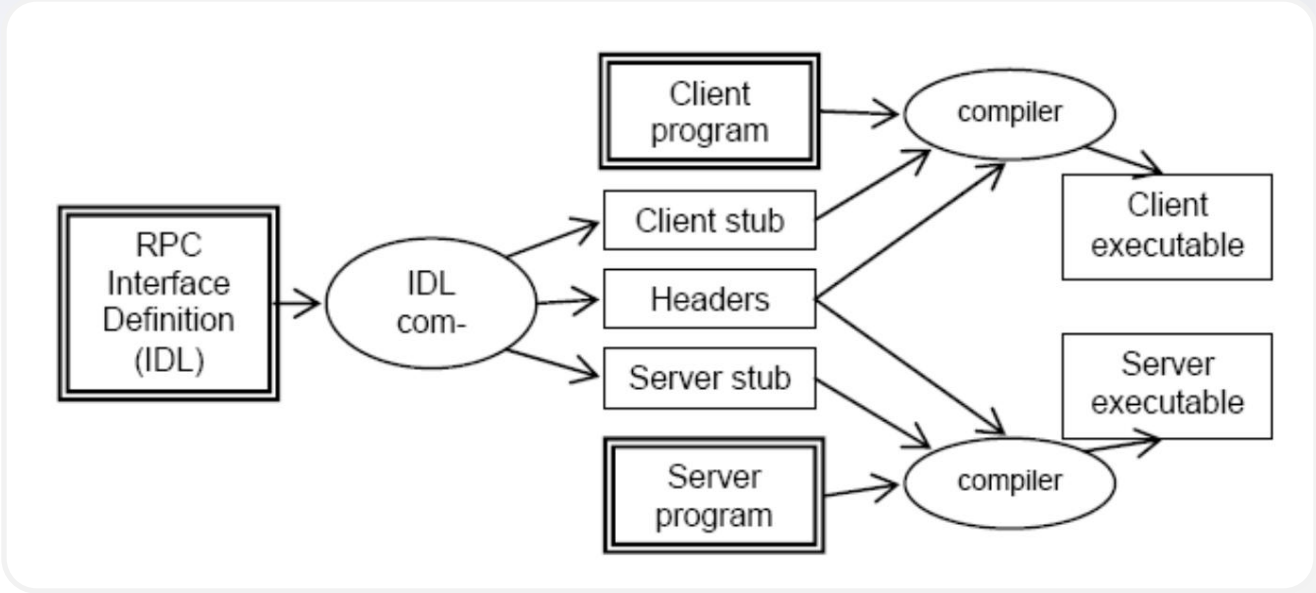
0x00000000-0x1fffffff: defined by sun

0x20000000-0x3fffffff defined by the user

0x40000000-0x5fffffff transient processes

0x60000000-0x7fffffff reserved

RPC prekompajler



Sun RPC kompajler (rpcgen) kao ulaz on uzima listu procedura (interfejsa) definisanih IDL-om (neka su npr. fajlu example.x) a izlaz iz njega su sledeći fajlovi:

rpcgen –C example.x

RPC prekompajler

1. Header fajl (example.h) koji sadrži definiciju tipova, konstanti i prototipova funkcija. Npr. u njemu su definisane vrednosti konstanti broj programa i broj verzije preuzeti iz IDLa. Sa #include se uključuje i u klijent stub i u server stub

2. Klijent stub program (example_clnt.c) koji sadrži procedure koje će pozivati klijent program. One imaju ime imeprocedure_broj verzije. Ove procedure su zadužene za pakovanje parametara u poruke u standardni format, slanje poruka, prijem poruka, raspakovanje rezultata u interni format mašine i

prosleđivanje rezultata klijentu

RPC prekompajler

3. Server stub program, example_svc.c koji sadrži procedure koje se pozivaju kad stigne poruka do servera. On sadrži server stub procedure koje vrše raspakovanje parametara pristigle poruke iz standardnog u format lokalne mašine, pozivaju odgovarajuću proceduru sa tim parametrima i pakuju rezultat procedure u standardni format.

Takođe RPC kompajler omogućava kreiranje kostura aplikacije klijentskog i serverskog programa nakon čega se dobijaju fajlovi:

RPC prekompajler

4. Klijent program, example_client.c koji uspostavlja vezu sa serverom dobijanjem porta na kome se izvršava server program, na osnovu imena servera, broja programa i broja verzije. Nakon toga, obavlja pozive udaljenih procedura.

5. Server program, example_server.c koji sadrži implementaciju odgovarajućih procedura. Procedure u serveru imaju ime imeprocedure_brojverzije_svc. Procedure u serveru uzimaju kao parametar pointer na podatak koji se prenosi, vraćaju pointer na rezultat. Ako se prenosi više parametara ili rezulata onda se to postiže preko struktura

RPC jezik(IDL)

RPC jezik je sličan programskom jeziku C.

RPC definicije tipova se kompajliraju i nalaze u odgovarajućem izlaznom header fajlu(example.h)

Postoji više vrsta definicija u RPC fajlu. Definicije nisu isto što i deklaracije jer se definicijom ne alocira nikakav memorijski prostor, već su to sam definicije jednog ili više elemenata. Promenljive će tek naknadno biti deklariasne.

Definicija programa

```
program TIMEPROG {
 version TIMEVERS {
  unsigned int TIMEGET(void) = 1;
     void TIMESET(unsigned) = 2;
 }=1;
} = 0x30000044;
```

RPC jezik(IDL)

Nakon provođenja korišćenjem rpcgen kompajlera u heder fajlu se nalazi:

#define TIMEPROG 0x30000044
#define TIMEVERS 1
#define TIMEGET 1
#define TIMESET 2

Ostale definicije

Konstante:

const DOZEN = 12; --> #define DOZEN 12

Tipovi:

typedef ST NT; --> typedef ST1 NT1;

RPC jezik(IDL)

Strukture:

```
struct intpair { int a, b };   -->   struct int pair { int a, b };
                                     typedef struct intpair intpair;
```

Tip nabrajanja:

```
enum colortype {
    RED = 0,
    GREEN = 1,
    BLUE = 2
};
```

-->

```
enum colortype {
    RED = 0,
    GREEN = 1,
    BLUE = 2
};
typedef enum colortype
colortype
```

RPC jezik(IDL)

Deklaracije

Deklaracije ne mogu stajati samostalno već moraju biti deo strukture ili u okviru typedef. RPc podržava sledeće deklaracije:

Prosta deklaracija:

```
colortype color; --> colortype color;
```

Nizovi fiksne veličine:

```
colortype palette[8]; --> colortype palette[8];
```

(niz fiksne veličine od 8 celih brojeva)

RPC jezik(IDL)

Nizovi promenljive veličine:

int heights<50>; (maksimalne veličine 50 long vrednosti)

ili long x_vals<>; (max 232-1 vrednosti)

PR.

typedef heights<50>; --> struct { u_int heights_len; int *heights_val; } heights;

gde je heights_len broj elemenata niza,

heights_val pokaz ivač na prvi element niza

Stringovi:

string x_vals<50>; (maksimalne veličine 50 karaktera)

string x_vals<>; (bilo koji broj karaktera)

RPC jezik(IDL)

Pointeri:

Deklaracija pointera je ista kao što je u C-u.

Koriste se za slanje lanč. listi ili stabala.

listitem *next; --> listitem *next;

ZADATAK 1

Napisati klijent server aplikaciju koja omogućava da klijent poziva sledeće udaljene procedure na serveru

- fact(int n) vraća n!.
- power(double x, double y) vraća x^^y.
- strconc(struct strings) nadovezuje src na dest

ZADATAK 2

Napisati klijent-server aplikaciju koja omogućava da klijent poziva udaljenu proceduru na serveru za sabiranje elemenata niza. Elementi niza se zadaju kao argumenti poziva klijentske aplikacije.

ZADATAK 3

Napisati klijent server aplikaciju koja omogućava da klijent poziva udaljenu proceduru na serveru za nalaženje zbira elemenata lančane liste. Elementi lančane liste se zadaju kao argumenti poziva klijentske aplikacije.

ZADATAK 4

Napisati klijent-server aplikaciju koja omogućava da klijent poziva udaljenu proceduru koja određuje zbir dva niza (C=A+B) i prikazuje elemente rezultujućeg niza na ekranu. Svaki element rezultujućeg niza se dobija kao zbir odgovarajućih elemenata ulaznih nizova. Nizovi mogu biti različitih dužina. U tom slučaju, na kraj rezultujućeg niza prepisuju se elementi dužeg ulaznog niza. Prilikom startovanja klij.aplikacije kao argumenti se unose broj elemenata prvog niza, elementi prvog niza i elementi drugog niza.