GeeksforGeeks
A computer science portal for geeks

Custom Search

COURSES

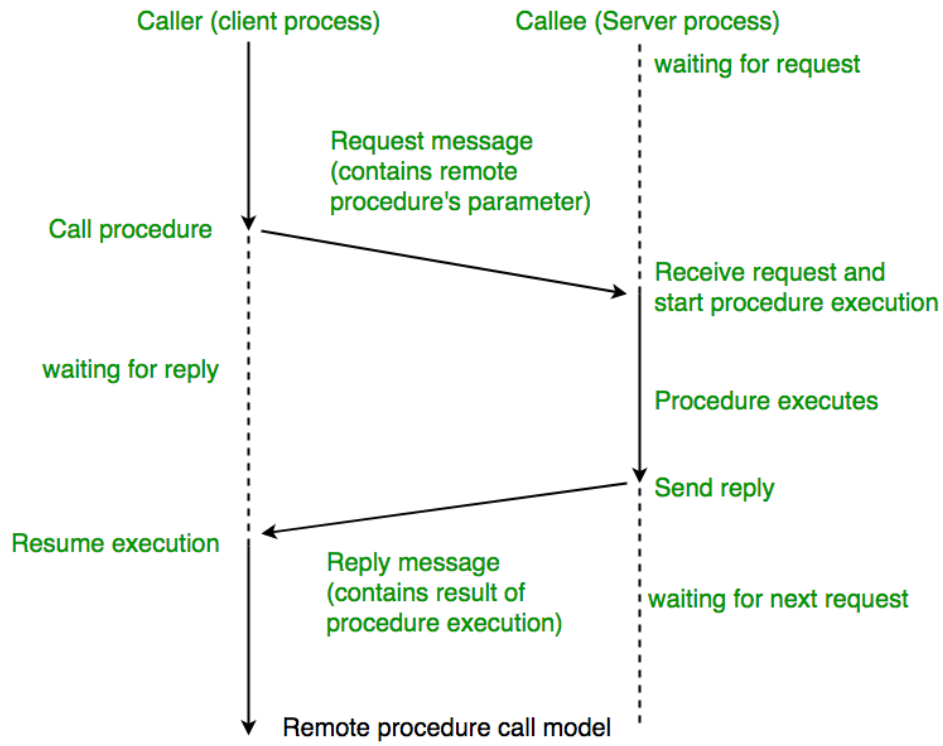
Login

HIRE WITH US

Remote Procedure Call (RPC) in Operating System

Remote Procedure Call (RPC) is a powerful technique for constructing **distributed, client-server based applications**. It is based on extending the conventional local procedure calling so that the **called procedure need not exist in the same address space as the calling procedure**. The two processes may be on the same system, or they may be on different systems with a network connecting them.

When making a Remote Procedure Call:



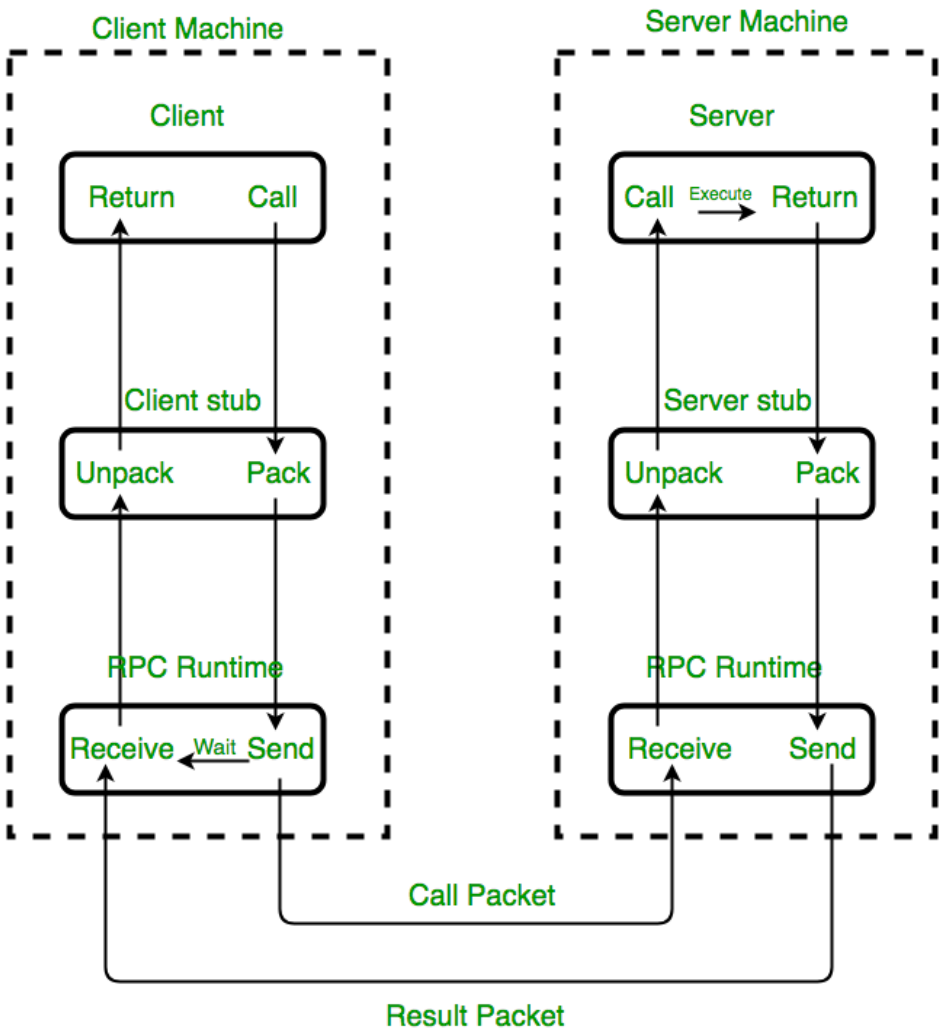
1. The calling environment is suspended, procedure parameters are transferred across the network to the environment where the procedure is to execute, and the

procedure is executed there.

2. When the procedure finishes and produces its results, its results are transferred back to the calling environment, where execution resumes as if returning from a regular procedure call.

NOTE: RPC is especially well suited for client-server **(e.g. query-response)** interaction in which the flow of control **alternates between the caller and callee**. Conceptually, the client and server do not both execute at the same time. Instead, the thread of execution jumps from the caller to the callee and then back again.

Working of RPC



Implementation of RPC mechanism

The following steps take place during a RPC:

1. A client invokes a **client stub procedure**, passing parameters in the usual way. The client stub resides within the client's own address space.

2. The client stub **marshalls(pack)** the parameters into a message. Marshalling includes converting the representation of the parameters into a standard format, and

copying each parameter into the message.

3. The client stub passes the message to the transport layer, which sends it to the remote server machine.

4. On the server, the transport layer passes the message to a server stub, which **demarshalls(unpack)** the parameters and calls the desired server routine using the regular procedure call mechanism.

5. When the server procedure completes, it returns to the server stub **(e.g., via a normal procedure call return)**, which marshalls the return values into a message. The server stub then hands the message to the transport layer.

6. The transport layer sends the result message back to the client transport layer, which hands the message back to the client stub.

7. The client stub demarshalls the return parameters and execution returns to the caller.

RPC ISSUES

- **Issues that must be addressed:**

1. RPC Runtime: RPC run-time system is a library of routines and a set of services that handle the network communications that underlie the RPC mechanism. In the course of an RPC call, client-side and server-side run-time systems' code handle **binding, establish communications over an appropriate protocol, pass call data between the client and server, and handle communications errors.**

2. Stub: The function of the stub is to **provide transparency to the programmer-written application code**.

On the client side, the stub handles the interface between the client's local procedure call and the run-time system, marshaling and unmarshaling data, invoking the RPC run-time protocol, and if requested, carrying out some of the binding steps.

On the server side, the stub provides a similar interface between the run-time system and the local manager procedures that are executed by the server.

3. Binding: How does the client know who to call, and where the service resides?

The most flexible solution is to use dynamic binding and find the server at run time when the RPC is first made. The first time the client stub is invoked, it contacts a name server to determine the transport address at which the server resides.

Binding consists of two parts:

- Naming:

Remote procedures are named through interfaces. **An interface uniquely identifies a particular service, describing the types and numbers of its arguments**. It is similar in purpose to a type definition in programming languauges.

- Locating:

Finding the transport address at which the server actually resides. Once we have the transport address of the service, we can send messages directly to the server.

A Server having a service to offer exports an interface for it. Exporting an interface registers it with the system so that clients can use it.

A Client must import an (exported) interface before communication can begin.

ADVANTAGES

1. RPC provides **ABSTRACTION** i.e message-passing nature of network communication is hidden from the user.

2. RPC often omits many of the protocol layers to improve performance. Even a small performance improvement is important because a program may invoke RPCs often.

3. RPC enables the usage of the applications in the distributed environment, not only in the local environment.

4. With RPC code re-writing / re-developing effort is minimized.

5. Process-oriented and thread oriented models supported by RPC.

References:

- https://web.cs.wpi.edu/~cs4514/b98/week8-rpc/week8-rpc.html
- https://users.cs.cf.ac.uk/Dave.Marshall/C/node33.html

This article is contributed by **Yash Singla**. If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Recommended Posts:

Operating Systems | Set 1

Operating Systems | Set 2

Operating Systems | Set 3

Operating Systems | Set 4

Operating System | Critical Section

Operating Systems | Set 5

Operating Systems | Set 6

Operating Systems | Set 10

Operating Systems | Set 7

Operating Systems | Set 8

Operating Systems | Set 9

Operating Systems | Set 11

Operating Systems | Set 12

Operating Systems | Set 13

Operating Systems | Set 14

Improved By : Akanksha_Rai

Article Tags : Operating Systems

Practice Tags : Operating Systems

8

2

To-do Done

Based on **8** vote(s)

Feedback/ Suggest Improvement | Add Notes | Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

A computer science portal for geeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org

COMPANY
About Us
Careers
Privacy Policy
Contact Us

LEARN
Algorithms
Data Structures
Languages
CS Subjects
Video Tutorials

PRACTICE
Courses
Company-wise
Topic-wise
How to begin?

CONTRIBUTE
Write an Article
Write Interview Experience
Internships
Videos

@geeksforgeeks, Some rights reserved

×
Sign In Sign Up
Username or email
Password
Remember me Forgot Password
Sign In
E-mail
Password
Institution/Organization
Sign Up
or
Google
Facebook

Why Create an Account?

By creating this account, you agree to our Privacy Policy & Cookie Policy.

Please enter your email address or userHandle.

Username/Email

Back to Login

Reset Password