Rule Processing dan Service pada Network

Created by Henny Pratiwi_09011281520129

Rule Processing dan Service pada Network

Created by Henny Pratiwi_09011281520129

1. Remote Procedure Call (RPC)
   Remote Procedure Call (RPC) adalah sebuah metode yang memungkinkan kita untuk mengakses sebuah prosedur yang berada di komputer lain. Untuk dapat melakukan ini sebuah server harus menyediakan layanan remote procedure. Pendekatan yang dilakuan adalah sebuah server membuka socket, lalu menunggu client yang meminta prosedur yang disediakan oleh server. Bila client tidak tahu harus menghubungi port yang mana, client bisa me-request kepada sebuah matchmaker pada sebuah RPC port yang tetap. Matchmaker akan memberikan port apa yang digunakan oleh prosedur yang diminta client.

   RPC masih menggunakan cara primitif dalam pemrograman, yaitu menggunakan paradigma procedural programming. Hal itu membuat kita sulit ketika menyediakan banyak remote procedure. RPC menggunakan socket untuk berkomunikasi dengan proses lainnya. Pada sistem seperti SUN, RPC secara default sudah ter-install kedalam sistemnya, biasanya RPC ini digunakan untuk administrasi sistem. Sehingga seorang administrator jaringan dapat mengakses sistemnya dan mengelola sistemnya dari mana saja, selama sistemnya terhubung ke jaringan.

   Kelebihan RPC:
   - Relatif mudah digunakan :

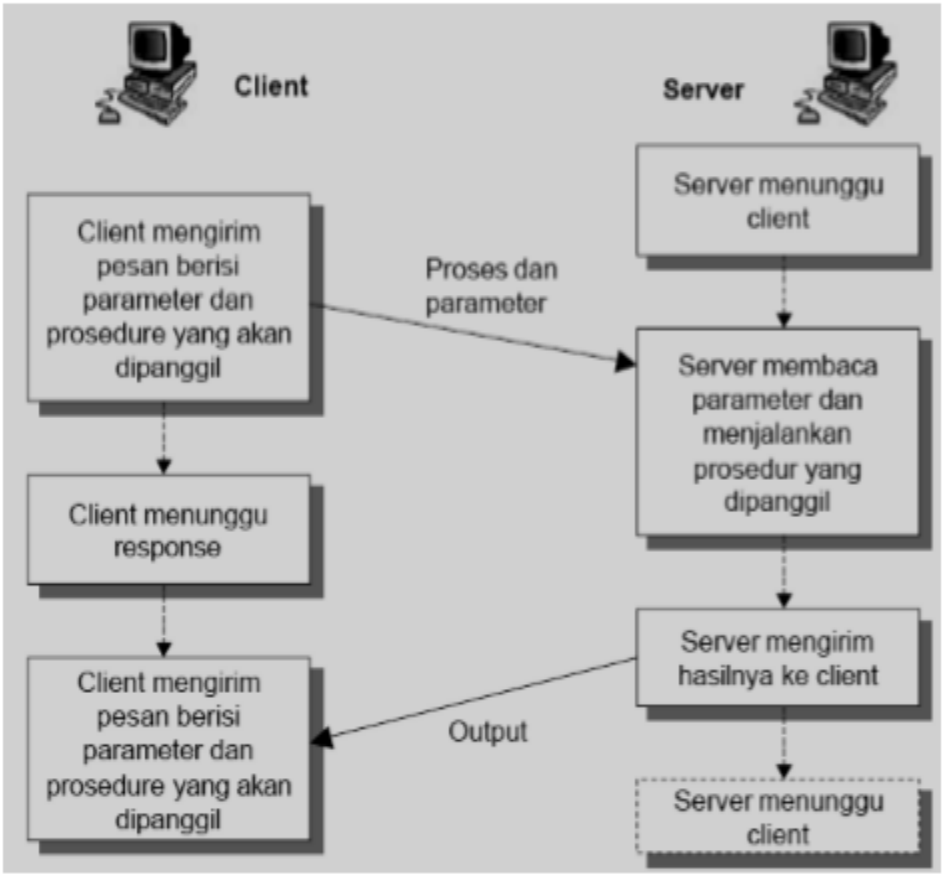
   Pemanggilan remote procedure tidak jauh berbeda dibandingkan pemanggilan local procedure. Sehingga pemrogram dapat berkonsentrasi pada software logic, tidak perlu memikirkan low level details seperti socket, marshalling & unmarshalling.
   - Robust (Sempurna):

   Sejak tahun 1980-an RPC telah banyak digunakan dlm pengembangan missioncritical application yg memerlukan scalability, fault tolerance, & reliability.

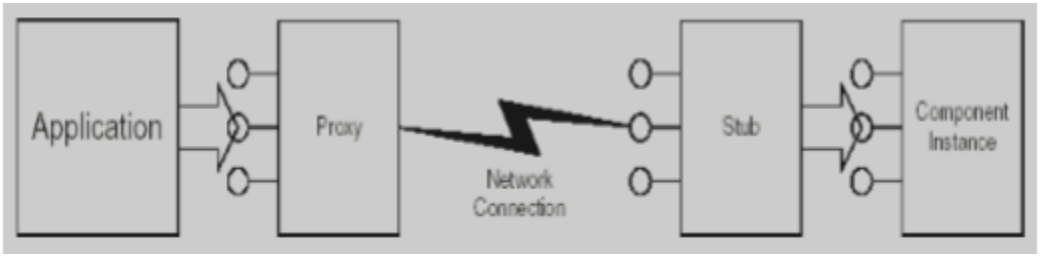
   Kekurangan RPC :
   - Tidak fleksibel terhadap perubahan:

   Static relationship between client & server at run-time.
   - Berdasarkan prosedural/structured programming yang sudah ketinggalan jaman dibandingkan OOP.

Prinsip RPC dalam program Client-Server :

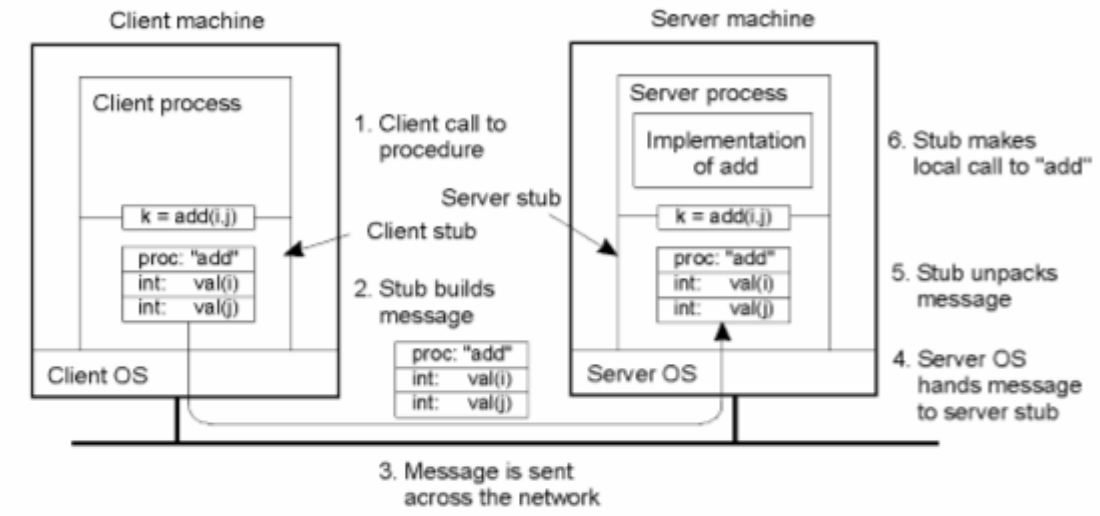


Skema RPC ini dilakukan juga pada proses-proses yang running di komputer berlainan :



- Sebelum mekanisme RPC digunakan, data harus di-packaging ke dalam format transimisi. Langkah ini dinamakan marshalling.
- Proxy bertanggung jawab untuk marshalling data, kemudian mengirimkan data dan meminta instans dari komponen (remote).
- Stub menerima request, unmarshall data, dan memanggil method yang diminta. Kemudian proses mengembalikan nilai yang diinginkan.

Langkah-langkah dalam RPC



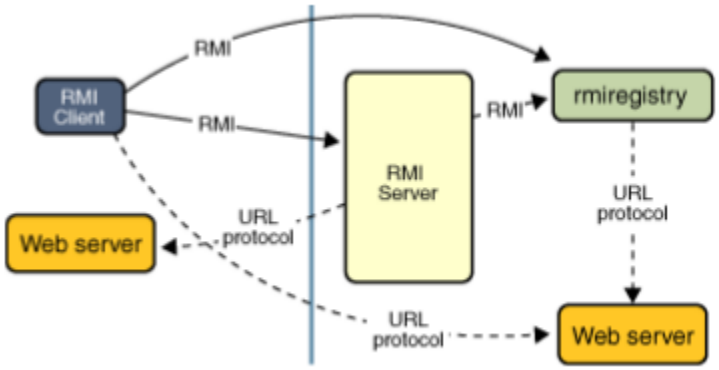
1. Prosedur client memanggil client stub
2. Client stub membuat pesan dan memanggil OS client
3. OS client mengirim pesan ke OS server
4. OS server memberikan pesan ke server stub
5. Server stub meng-*unpack* parameter-parameter untuk memanggil server
6. Server mengerjakan operasi, dan mengembalikan hasilnya ke server stub
7. Server stub mem-*pack* hasil tsb dan memanggil OS server
8. OS server mengirim pesan (hasil) ke OS client
9. OS client memberikan pesan tersebut ke client stub
10. Client stub meng-*unpack* hasil dan mengembalikan hasil tersebut ke client

2. RMI (Remote Method Invocation)

Remote Method Invocation (RMI) adalah sebuah teknik pemanggilan method remote yang lebih secara umum lebih baik daripada RPC. RMI menggunakan paradigma pemrograman berorientasi obyek (Object Oriented Programming). RMI memungkinkan kita untuk mengirim obyek sebagai parameter dari remote method. Dengan dibolehkannya program Java memanggil method pada remote obyek, RMI membuat pengguna dapat mengembangkan aplikasi Java yang terdistribusi pada jaringan.

Ilustrasi berikut menggambarkan aplikasi RMI terdistribusi yang menggunakan registry untuk mendapatkan referensi ke objek remote. Server memanggil registry untuk mengasosiasikan (mengikat) suatu nama dengan objek remote. Client mencari objek remote dengan namanya pada registry server dan meng-invoke method dari objek. Ilustrasi ini juga menunjukkan sistem RMI menggunakan Web server untuk memanggil class

bytecodes, dari server ke client dan dari client ke server, untuk objek-objek yang diperlukan.



Membangun suatu aplikasi terdistribusi menggunakan RMI meliputi 6 langkah. Keenam langkah tersebut adalah:

1. Mendefinisikan remote interface
2. Implementasi remote interface dan server
3. Pengembangan client (atau applet) yang menggunakan remote interface
4. Mengkompilasi source files dan mem-buat stub and skeletons
5. Memulai (start) RMI registry 6. Menjalankan server dan client.

Adapun contoh yang lainnya sebagai berikut :

1. CORBA (Common Object Request Broker Architecture).
2. SOAP (Simple Object Access Protocol).

Referensi:

1. Rijal Fadilah, Handout Komunikasi Data dan Jaringan Komputer, http://rijalfadilah.multiply.com/

2. Trindiana dkk., Tugas Kuliah Pengantar Sistem Terdistribusi, 2008.

3. Mudji, Model Jaringan 7 OSI Layer, http://mudji.net/press/?p=61

4. Lintang, Pengenalan Hardware dan Topologi Jaringan Komputer, http://staffsite.gunadarma.ac.id/lintang/index.php?stateid=download&id=2268&part=files

5. http://bebas.vlsm.org/v06/Kuliah/SistemOperasi/BUKU/SistemOperasi-4.X1/ch17s06.html

6. www.bebas.vlsm.org/v06/Kuliah/SistemOperasi/BUKU/bahan/bahanbab3.pdf

7. Isak Rickyanto, Tutorial Pengenalan Java RMI, http://www.benpinter.net/article.php?story=20030818005713433

8. Ayu Anggriani dkk., Tugas Kuliah Pengantar Sistem Terdistribusi, 2008.