

Aplikasi Pengontrol Komputer Personal Dengan *Interface Web Browser* Pada Laboratorium Teknik Informatika UKP

Rudy Adipranata¹⁾ Djoni H. Setiabudi²⁾ Andylo

1) Teknik Informatika – Fakultas Teknologi Informasi
Universitas Kristen Petra, Jl. Siwalankerto 121-131, Surabaya, email : rudya@petra.ac.id
2) Teknik Informatika – Fakultas Teknologi Informasi
Universitas Kristen Petra, Jl. Siwalankerto 121-131, Surabaya, email : djonihs@petra.ac.id

Abstrak - Jurusan Teknik Informatika UKP memiliki enam laboratorium yang digunakan sebagai fasilitas penunjang kegiatan perkuliahan. Didalam pengoperasian laboratorium, terdapat beberapa masalah yang sering dihadapi sekitar penggunaan komputer pada laboratorium yang tidak semestinya antara lain para praktikan yang membuka dan menggunakan program selain yang diperlukan pada waktu praktikum berlangsung serta membuka website yang tidak diperkenankan di laboratorium. Kegiatan ini dapat menyebabkan terganggunya konsentrasi pada kegiatan praktikum.

Untuk menanggulangi masalah tersebut maka pada penelitian ini dibuat program aplikasi yang memiliki fasilitas untuk membantu pengontrolan penggunaan komputer pada laboratorium. Program ini dibuat dengan memanfaatkan teknologi *client server*, dimana *client* adalah komputer yang digunakan oleh praktikan, sedangkan *server* adalah komputer yang digunakan khusus untuk mengontrol *client* tersebut. Sebagai *interface pengontrol client* tersebut, digunakan *web browser* yang mempunyai keuntungan dalam mobilitas, sehingga pengontrolan *client* dapat dilakukan dari komputer manapun yang terhubung dalam jaringan komputer.

Dari pengujian penggunaan program, didapat kesimpulan bahwa program aplikasi ini dapat memenuhi kebutuhan untuk melakukan pengontrolan penggunaan komputer *client*. Fasilitas yang dibuat pada program aplikasi ini ialah kemampuan untuk melakukan eksekusi program pada *client*, melakukan penguncian dan pembebasan kembali terhadap masukan keyboard dan mouse pada komputer *client*, melakukan *shut down* dan *reboot* komputer *client*, melakukan pengiriman pesan ke *client*, melihat tampilan layar monitor *client*, menampilkan daftar program yang sedang aktif dijalankan di komputer *client*, serta mematikan program yang sedang aktif dijalankan di komputer *client*.

Kata kunci : *client server, PC remote control*

1. PENDAHULUAN

Pada Jurusan Teknik Informatika UKP, terdapat enam laboratorium yang digunakan untuk menunjang kegiatan perkuliahan. Pengoperasian kegiatan pada laboratorium dikoordinasi oleh asisten laboratorium dan kepala laboratorium. Sejalan dengan

berjalannya kegiatan pada laboratorium, terdapat beberapa masalah yang sering dihadapi. Masalah tersebut berkisar pada penggunaan komputer yang tidak semestinya, antara lain para praktikan yang membuka dan menggunakan program selain yang diperlukan pada waktu praktikum berlangsung serta membuka *website* yang tidak diperkenankan di laboratorium. Kegiatan ini dapat menyebabkan terganggunya konsentrasi pada kegiatan praktikum. Para asisten sering mengalami kesulitan untuk mengatasi permasalahan ini dikarenakan sedikitnya jumlah asisten sementara jumlah komputer yang digunakan cukup banyak.

Untuk mengatasi permasalahan-permasalahan tersebut di atas, maka dirancanglah sebuah program aplikasi yang dapat mengontrol komputer dari jarak jauh dengan memanfaatkan infrastruktur jaringan komputer. Program aplikasi tersebut dirancang untuk dapat menyediakan informasi keadaan di dalam laboratorium sehingga dapat membantu para asisten dan kepala laboratorium dalam melakukan pengawasan terhadap penggunaan komputer. Informasi ini meliputi data-data komputer yang sedang digunakan serta status komputer tersebut. Disamping itu, terdapat fasilitas untuk mengirim pesan kepada praktikan, mengunci komputer praktikan, melakukan *shut down* dan *reboot* komputer.

Program aplikasi ini dibuat dengan memanfaatkan teknologi *client server*, dimana *client* adalah komputer yang digunakan oleh praktikan, sedangkan *server* adalah komputer yang digunakan khusus untuk mengatur *client* tersebut. Dari *server* akan mengirimkan perintah ke komputer *client* dan kemudian komputer *client* akan melaksanakan perintah tersebut ataupun memberikan informasi balasan kepada komputer *server*. Secara konsep hal ini terbalik dengan konsep *client server* umum, dimana *server* adalah komputer yang melakukan perintah dari *client*. Tetapi di sini digunakan istilah *client* untuk menunjukkan komputer yang dipakai oleh praktikan yang dikontrol oleh asisten. Sedangkan komputer yang melakukan pengontrolan disebut dengan *server*. Untuk *interface* pengontrolan *client*, selain berupa program desktop, dapat juga menggunakan *web browser* yang mempunyai keuntungan dalam mobilitas, sehingga pengontrolan *client* dapat dilakukan dari komputer manapun yang terhubung dalam jaringan komputer.

2. TCP/IP

TCP/IP adalah sebuah protokol atau aturan yang hingga sekarang diterima secara umum sebagai standar *de facto* untuk koneksi jaringan global internet. Sebuah komputer dapat terhubung ke dalam jaringan internet jika pada komputer tersebut mengimplementasikan protokol TCP/IP. TCP/IP dibagi menjadi 5 *layer* yang masing-masing *layer* memiliki fungsi yang berbeda-beda. *Layer-layer* tersebut adalah *application layer*, *transport layer*, *internet layer*, *network layer*, dan *physical layer* [1] yang masing-masing memiliki fungsi sebagai berikut :

1) *Application Layer*

Layer ini berfungsi sebagai tempat dimana program-program aplikasi akan berjalan.

2) *Transport Layer*

Layer ini berfungsi untuk menentukan jenis koneksi yang akan dipergunakan untuk menghubungkan antar *host*.

3) *Internet Layer*

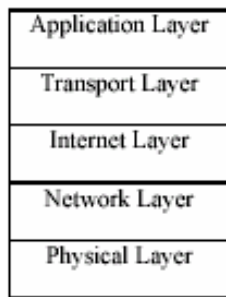
Layer ini berfungsi untuk menentukan jenis paket yang akan dipergunakan pada saat pengiriman data.

4) *Network Layer*

Layer ini berfungsi untuk menghubungkan komputer dengan jaringan yang ada.

5) *Physical Layer*

Layer ini berfungsi untuk mengatur pengiriman paket-paket data yang dilakukan pada jalur fisik.



Gambar 1: *Layer* TCP/IP

Terkait dengan pembagian *layer* tersebut, maka aplikasi yang dikembangkan di sini terletak pada *Application Layer*

3. SOCKET PROGRAMMING

Aplikasi yang dikembangkan di sini berjalan pada sistem operasi Windows terutama Windows 98 untuk sisi *client*. Karena itu dalam mengembangkan aplikasi ini digunakan Windows *socket programming*. Windows *socket* adalah salah satu *standard programming interface* TCP/IP pada semua versi Windows. Windows *socket* secara natural menunjang pemrograman dengan bahasa pemrograman C, tetapi dapat pula digunakan dengan menggunakan bahasa pemrograman yang lain. Aplikasi ini dikembangkan dengan menggunakan bahasa pemrograman Delphi. Walaupun dimungkinkan untuk menggunakan Windows *socket* dengan bahasa pemrograman Delphi, tetapi dalam kenyataannya penggunaan secara langsung cukup rumit dikarenakan harus terdapat konversi untuk mengubah tipe variable serta format *syntax* dari bahasa pemrograman C ke Delphi. Untuk

itu pada aplikasi ini digunakan komponen yang telah mengenkapsulasi semua *interface* pada windows *socket* menjadi format *syntax* serta tipe variabel dalam bahasa pemrograman Delphi. Komponen yang digunakan disini adalah dWinsock.

4. APLIKASI WEB SERVER

Karena pada aplikasi yang dibuat menggunakan *interface web browser*, maka harus dilakukan juga pembuatan aplikasi pada *web server*. Sebelum melakukan pembuatan aplikasi pada *web server*, diperlukan pemilihan jenis aplikasi yang akan dibuat. Jenis aplikasi pada *web server* dapat dikelompokkan menjadi 4 jenis yaitu

1) *Internet Server Application Program Interface* (ISAPI)

2) *Netscape Server Application Program Interface* (NSAPI)

3) *Common Gateway Interface* (CGI)

4) *Common Gateway Interface-Window* (CGI-WIN)

ISAPI/NSAPI (*Internet Server Application Program Interface / Netscape Server Application Program Interface*) pada dasarnya adalah sesuatu yang sama. ISAPI merupakan spesifikasi dari Microsoft sedangkan NSAPI merupakan spesifikasi dari Netscape. ISAPI digunakan untuk aplikasi pemrograman *interface* dari internet *server*. Sekarang ini banyak dilakukan penggabungan antara ISAPI dengan NSAPI menjadi hanya ISAPI. NSAPI makin lama makin menghilang dan membentuk menjadi standar ISAPI.

CGI (*Common Gateway Interface*) merupakan standar aplikasi pada *web server* yang paling lama. CGI ini hanya membaca informasi dari standard input dan menulis dengan menggunakan standard output. Aplikasi akan dijalankan saat terjadi permintaan dari *web browser* dan menghasilkan sesuatu yang dikirim kembali ke *web browser*. Sedangkan Win-CGI (*Common Gateway Interface-Windows*) adalah pengembangan dari CGI yang dikhususkan untuk bekerja pada sistem operasi Windows.

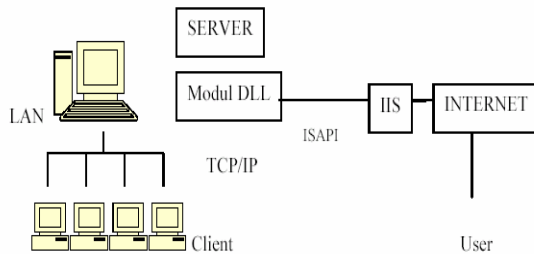
5. PERANCANGAN APLIKASI

Aplikasi ini terbagi menjadi tiga bagian yaitu *client*, *server* serta *interface* pada *web server*. Untuk aplikasi *client* dan *server* dibuat dengan menggunakan bahasa pemrograman Delphi 5.0, sedangkan aplikasi pada *web server* dibuat dengan menggunakan ISAPI juga dengan bahasa pemrograman Delphi 5.0 serta bahasa pemrograman ASP. Aplikasi ini berjalan pada sistem operasi Windows khususnya Windows 98 dan untuk aplikasi pada *web server*, berjalan pada *web server* yang bisa mendukung ASP yaitu IIS (*Internet Information Server*) ataupun PWS (*Personal Web Server*).

Client mewakili seluruh komputer yang ada. Pada posisi *client* ini, perintah-perintah dari *server* seperti *shut down*, *reboot*, *lock*, *unlock*, dan lain-lain akan dijalankan dan akan mengirimkan hasilnya kepada *server* kembali. *Server* berperan mengatur seluruh *client* dan menerima perintah dari *user*. Dimana *user* ini mewakili *user* yang melakukan akses

ke server dengan *interface web browser*. Perintah-perintah dari *user* akan diterima oleh *server* dan akan dilanjutkan ke *client* untuk dilaksanakan. Setelah selesai dilaksanakan, *client* akan mengirimkan kembali dan *server* yang menerima akan mengirimkan ke *user*.

Jika digambarkan maka hubungan antar program aplikasi pada masing-masing *client* dengan *server* dan *user* adalah sebagai berikut :



Gambar 2: Diagram Aplikasi

Sistem kerja aplikasi ini adalah *user* atau *web browser* mengirimkan permintaan melalui *interface* berupa HTTP dimana permintaan tersebut dikirimkan ke *server* oleh PWS, dan dari PWS diteruskan ke modul DLL melalui perantara ISAPI. Modul DLL ini merupakan aplikasi dari bahasa pemrograman Borland Delphi 5. Modul DLL mengirimkan permintaan dari *user* tersebut ke *server*. Dari *server* permintaan tersebut dilanjutkan menuju sisi *client*.

Client disini merupakan komputer-komputer yang berada di dalam laboratorium. Setelah proses permintaan telah dijalankan oleh sisi *client*, *client* mengirimkan jawaban telah dilaksanakannya perintah tersebut ke *server*. *Server* mengirimkan kembali hasil kerja ke *user*. Selain pengiriman perintah yang dilakukan pada *user* dengan menggunakan *web browser*, *user* dapat juga mengirimkan perintah langsung dari *server*. *Server* memiliki fasilitas agar dapat mengakses *client* secara langsung.

5.1 Perancangan Aplikasi Client

Aplikasi *client* ini berfungsi untuk menerima perintah dari *server* dan kemudian menjalankan perintah tersebut ataupun memberikan balasan atas suatu perintah. Pada saat aplikasi *client* dijalankan maka tidak terdapat *interface* apapun juga, dan aplikasi ini akan berjalan sebagai proses *background*.

Pada aplikasi *client* ini, komponen utama yang digunakan adalah *TTextClient* dari *dWinsock*. *TTextClient* ini berguna untuk menerima data bertipe string yang dikirim oleh *server* [2]. Dari data yang diterima tersebut akan didapat perintah yang diinginkan oleh *server* sehingga *client* dapat melakukan perintah tersebut. Sebagai contoh prosedur utama pada aplikasi *client* dapat dilihat pada cuplikan program di bawah ini :

```

Procedure
MainForm.TextClientLineReceived(Socket: TLineSocketBase;
const Line: string; Complete: Boolean);
begin
if Line = '[SHUTDOWN]' then

```

```

begin
CheckReg;
TextClient.Close;
ExitWindowsEx(EWX_SHUTDOWN or
EWX_FORCE or EWX_POWEROFF, 0);
End;
end;

```

Dari cuplikan program tersebut dapat dikembangkan untuk semua perintah yang telah didefinisikan dengan mengganti perbandingan pada baris 'if Line = '[SHUTDOWN]' then' dan mendefinisikan hal-hal yang harus dilakukan sesuai kebutuhan.

5.2 Perancangan Aplikasi Server

Pada aplikasi *server*, proses utama adalah mengirim perintah ke *client* sesuai dengan yang diinginkan oleh *user*. Komponen utama yang digunakan adalah *TTextServer* dari *dWinsock*. Komponen *TTextServer* ini berguna untuk mengirimkan perintah ke *client* [2]. Berikut adalah cuplikan program pada aplikasi *server* :

```

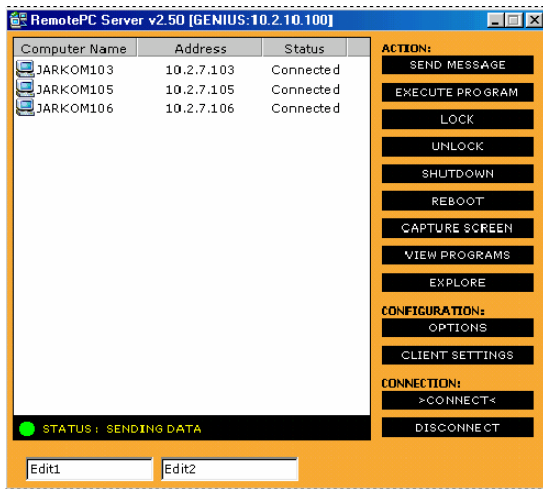
procedure TMainForm.SendMsg(Msg : string);
var
i : Integer;
Pos : Integer;
begin
with ListView1 do
for i := 0 to ListView1.Items.Count - 1 do
if Items[i].Selected or Items[i].Focused then
begin
Pos := FindSocket(Integer(Items[i].Data^));
if Pos <> -1 then
TLineSocket(TextServer1.Client[Pos]).WriteLine(Msg);
end;
end;
end;

```

Jika hendak melakukan suatu perintah, prosedur *SendMsg* di atas dipanggil dengan parameter *Msg* yang sesuai. Di samping tugas utama untuk mengirim perintah ke *client*, *server* juga bertugas untuk memonitor status koneksi *client*, jika terdapat *client* yang memutuskan koneksi ataupun *client* baru yang terhubung ke *server*. Status *client* tersebut akan ditampilkan pada sebuah *List*.

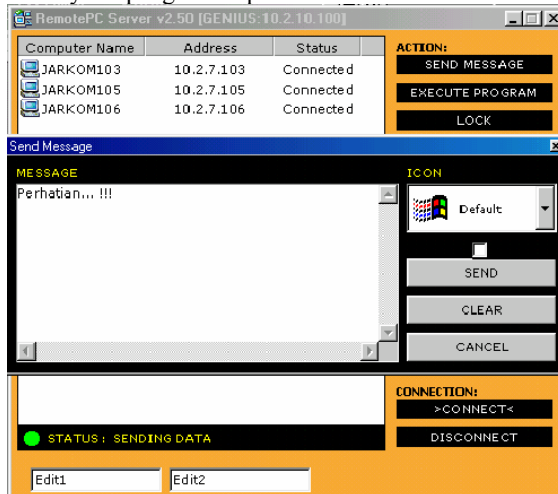
6. HASIL APLIKASI

Bagian utama dari aplikasi ini ialah aplikasi *server*, dimana aplikasi *server* ini pada saat pertama kali dijalankan akan menunggu *client* yang masuk. Jika terdapat *client* yang masuk, maka data *client* berupa nama komputer dan IP *address* beserta status akan ditampilkan. Tampilan aplikasi *server* dapat dilihat pada gambar di bawah.



Gambar 3: Tampilan Server

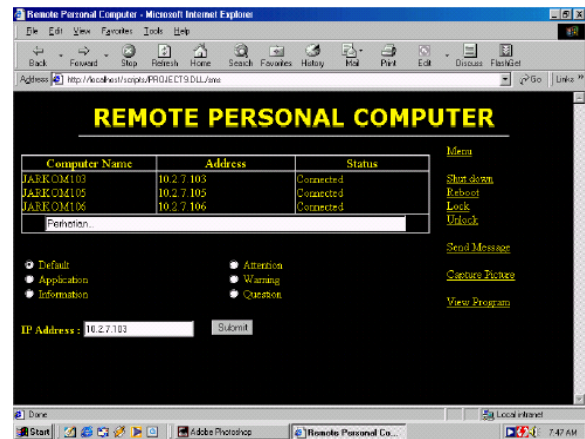
Pada bagian *server* terdapat tombol-tombol untuk melakukan perintah pada *client*. Secara lengkap perintah yang dapat dilakukan *server* adalah : mengirim pesan ke *client*, mengeksekusi program pada *client*, mengunci / membuka kunci pada komputer *client* sehingga keyboard maupun mouse pada komputer *client* tidak dapat digunakan, me-reboot dan men-shutdown komputer *client*, mengambil tampilan layar *client*, melihat program yang sedang aktif pada komputer *client* serta mematikan program tersebut. Berikut ini ditampilkan pelaksanaan salah satu perintah yang terdapat pada *client* yaitu pengiriman pesan.



Gambar 4: Pengirim Pesan ke Client

Dengan adanya aplikasi *server* ini, *user* dalam hal ini asisten laboratorium harus berada di depan komputer yang terpasang aplikasi *server* ini, sehingga hal ini mengurangi kebebasan pengontrolan. Untuk itu jika *user* sedang berada pada komputer yang tidak terpasang aplikasi *server*, *user* dapat menggunakan *web browser* untuk melakukan pengontrolan.

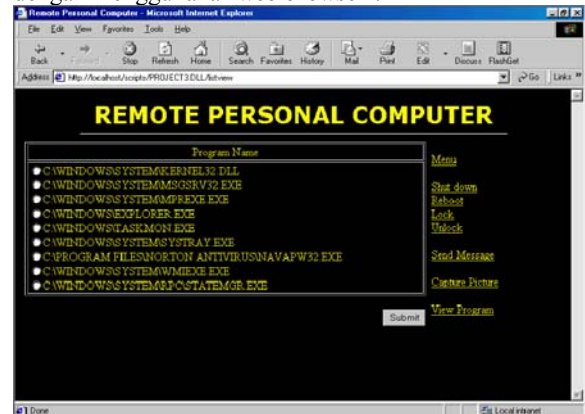
Tampilan *interface* dengan menggunakan *web browser* dapat dilihat pada gambar di bawah ini :



Gambar 5: Tampilan Aplikasi pada Web Browser

Fasilitas yang terdapat pada *web browser* ini sama dengan fasilitas pada aplikasi *server*, sehingga pengontrolan tetap dapat dilakukan oleh *user* dari komputer manapun selama komputer tersebut terhubung ke jaringan komputer.

Berikut ini adalah tampilan dari hasil melihat program yang sedang berjalan pada *client*, dilakukan dengan menggunakan *web browser* :



Gambar 6. Melihat Proses pada Client

7. KESIMPULAN

Dari pengujian yang telah dilakukan terlihat bahwa aplikasi pengontrol komputer yang telah dibuat dapat membantu mengatasi permasalahan yang terdapat pada laboratorium. Karena aplikasi ini terutama digunakan pada jaringan lokal, maka tidak terdapat *delay* yang signifikan walaupun digunakan untuk proses pengiriman data yang cukup besar seperti melihat tampilan layar monitor komputer *client*.

DAFTAR PUSTAKA

- [1] Stallings, William. *Data and Computer Communication 5 th Ed*, New Jersey : Prentice Hall, Inc, 1987
- [2] Söderberg, Ulf; Palmer, Marc and Hawes, Keith. *Help for dWinsock Components for Delphi and C++ Builder Version 2.51*, <http://www.aait.com/dwinsoc/>, 1997
- [3] Hunt, Craig. *TCP/IP Network Administration*, Sebastopol : O'Reilly & Associates, Inc, 1993
- [4] Watson, Blake. *Delphi by Example*, Indianapolis : Que Corporation, 1995