

Pemanfaatan Windows Socket Control Untuk Aplikasi Intranet Messenger

Bambang Kelana Simpony

Manajemen Informatika

AMIK BSI Tasikmalaya

e-mail: bambang.bky@bsi.ac.id

Abstrak

Pengertian *socket* adalah *interface* (antarmuka) pada jaringan yang menjadi titik komunikasi antarmesin pada *Internet Protocol*, dan tentunya tanpa komunikasi ini, tidak akan ada pertukaran data dan informasi jaringan. Pemrograman *socket* adalah cara untuk menggunakan komponen/API (*Application Programming Interface*) *socket* untuk membuat sebuah aplikasi jaringan yang bekerja dengan sistem *Client-Server* karena dengan *socket* memungkinkan kita untuk melakukan suatu komunikasi data. Untuk sistem operasi Windows biasa disebut WinSock, singkatan dari *Windows Socket*. Untuk interkoneksi jaringan dapat digunakan dua protokol yaitu UDP dan TCP. UDP atau *User Datagram Protocol* mengirimkan data tanpa menghiraukan apakah data tersebut sampai ketujuan atau tidak, sedangkan protokol TCP atau *Transmission Control Protocol* mengharuskan sebelum mengirimkan data harus terjalin suatu koneksi dahulu antara *client* dan *server* sehingga data dapat sampai dengan baik. Untuk Tugas Akhir ini digunakan protokol TCP yang pemrogramannya diatur pada pemrograman *socket* ini. Dalam pemrograman *socket* ada langkah-langkah pemrograman yang harus dilakukan sesuai dengan protokol yang digunakan. Pada penggunaan protokol TCP ada perintah *Listen* dan *Connect* sehingga menjamin reliabilitas antara *server* dan *client* sedangkan pada protokol UDP hal itu tidak digunakan. Dari sekilas perbedaan ini, maka dapat ditarik kesimpulan bahwa aplikasi *socket* yang menggunakan TCP memerlukan pertukaran data dua arah yang *valid*. Sedangkan, aplikasi *socket* yang menggunakan UDP lebih memprioritaskan pada pengumpulan data. Karena itu, aplikasi *socket* dengan TCP sering diterapkan untuk aplikasi *chat*, transfer file, ataupun transaksi-transaksi penting.

Kata Kunci: klien, server, winsock

Abstract

The interpretation from *socket* is an interface from a network that become a point to inter machine communication in *Internet Protocol*, and of course without this communication, there are no data exchange and network information. *Socket programming* is method how to use component API (*Application Programming Interface*) *socket* for create a network application that work with *Client-Server* system because with this *socket* enable us to carry out data communication. For operating system Windows usually said WinSock, abbreviation from *Windows Socket*. For network interconnection can use two protocol that is UDP and TCP. UDP or *User Datagram Protocol* sending data without slight over whether the data is arrive in destination or not, where as TCP protocol or *Transmission Control* protocol before sending data require interweave a connection between *client* and *server*, with the result that data will be arrive well. In this Tugas Akhir used TCP protocol which programming is arrange in this *socket programming*. In *socket programming* there is some programming step which have to do appropriate with protocol that used in. In TCP protocol there is *Listen* command and *Connect* command with the result that assure of reliability between *server* and *client* where as not used in UDP protocol. From this difference, so the conclusion is *socket application* with TCP had a need of valid two ways data exchange. Where as *socket application* with UDP more priority with data aggregation. As it is, *socket application* with TCP often used in for *chat application*, file transfer, or important transaction.

Key Words: client, server, winsock

1. Pendahuluan

Teknologi jaringan komputer terjadi peningkatan di tahun 80-an dan pada tahun 1988 jaringan komputer mulai digunakan di perusahaan dan universitas, sekarang memasuki era *world wide internet*, jaringan komputer telah menjadi hal yang biasa dilakukan oleh jutaan manusia di muka bumi ini.

Dengan berkembangnya teknologi tersebut, fungsi komputer tunggal yang tadinya melayani seluruh tugas komputasi suatu organisasi kini telah diganti dengan sekumpulan komputer yang terpisah akan

tetapi saling terkoneksi dalam melaksanakan tugas. Dua buah komputer dikatakan terkoneksi bila keduanya dapat saling bertukar informasi. Microsoft Windows, sebagai salah satu sistem operasi yang banyak digunakan saat ini tentunya menyediakan aplikasi untuk bertukar informasi dalam jaringan lokal. Diantara aplikasi standar Windows NT, XP, 2000, 2003 yang terkadang terlupakan adalah fitur pengiriman pesan melalui *Command Prompt* yang dinamakan *Net Send*. *Net Send* merupakan salah satu perintah *Net* yang paling banyak digunakan untuk mengirimkan pesan ke komputer tujuan (Subakti, 2003).

Net Send yang berbasis teks, memerlukan beberapa cara agar pesan bisa sampai ke tujuan yang diinginkan. Fitur yang masih sangat terbatas membuat *net send* jarang dipakai oleh kebanyakan pengguna komputer.

Berbagai aplikasi pun bermunculan untuk menggantikan fungsi *net send* ini. Untuk mempermudah dalam bertukar informasi tersebut, Microsoft Visual Basic 6.0 menyediakan satu objek agar kita dapat membuat program aplikasi yang bisa saling bertukar informasi. Objek tersebut dinamakan *WinSock (Windows Socket)*. Dalam aplikasi ini, penulis membuat dua program utama yaitu *server* dan *client*.

2. Metode Penelitian

Metode penelitian yang dilakukan yaitu menganalisa permasalahan yang telah didefinisikan dan dikelompokkan dalam tahap awal serta mengumpulkan dan menganalisa kebutuhan pendukung berupa *software* dan *hardware* untuk memecahkan masalah tersebut secara keseluruhan dengan rinci. Selanjutnya tahapan desain, dalam tahap ini hasil analisis dibuat dalam struktur program, dan rancangan antar muka keseluruhan secara rinci. Selanjutnya adalah mengaplikasikan pemodelan sistem (*design*) ke dalam bahasa pemrograman dengan *source code/syntax* yang sesuai dan bahasa yang digunakan dalam penelitian ini yaitu menggunakan bahasa pemrograman Visual Basic.

Selanjutnya tahap dokumentasi yaitu melakukan dokumentasi/pengumpulan dokumen untuk semua bahan, data-data, modul-modul yang telah selesai dikerjakan dari tahap awal sampai dengan tahap akhir pengerjaan yang berhubungan dengan program yang dikerjakan.

3. Hasil Dan Pembahasan

3.1. Tinjauan Kasus

3.1.1. Permasalahan Pokok

Proses pengiriman pesan dengan menggunakan *net send* merupakan proses yang sederhana. Pengguna mengetikkan perintah *net send* di *Command Prompt* untuk ditujukan ke satu user yang akan menerima pesan ini dalam bentuk *Message Box* (Kotak Pesan). Setelah sebelumnya mengaktifkan dulu servis untuk fitur ini melalui jendela *Service* yang berasal dari *Control Panel*.

Dari sedikit gambaran mengenai *net send* tersebut, tentunya masih banyak kekurangan yang didapat dari aplikasi tersebut.

Diantaranya, tampilan aplikasi yang masih menggunakan perintah, utilitas dari aplikasi yang masih kurang, serta tidak adanya

fasilitas untuk penyimpanan pesan yang digunakan sebagai *log file* ataupun *history*.

3.1.2. Alternatif Pemecahan Masalah

Dari uraian yang penulis kemukakan, maka sebagai alternatif pemecahan masalah, penulis membuat aplikasi pengiriman pesan berbasis visual. Tentunya dengan beberapa fungsi tambahan yang dibuat. Penulis mengembangkan aplikasi ini menggunakan komponen *Winsock (Windows Socket)* pada Visual Basic 6.0 yang telah terintegrasi didalamnya.

Tentunya pada implementasi aplikasi ini harus memiliki lebih dari satu *socket* yang mana *socket* yang satu digunakan untuk *listening* dan yang lain untuk menghubungi program yang lain.

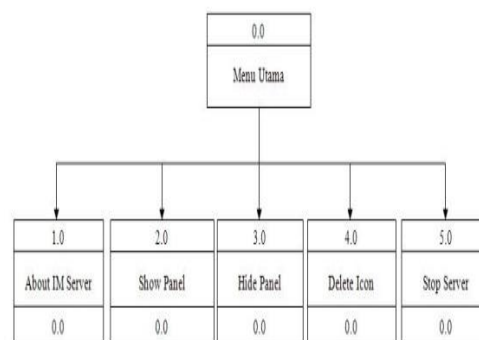
Komunikasi *socket* ini membutuhkan sebuah port untuk berkomunikasi, layaknya sebuah gerbang untuk lalu lintas agar si A dapat pergi ketempat si B. Port ditentukan dengan angka.

Dari sekian banyak jumlah port yang disediakan oleh Windows yaitu sebanyak 65536 port. Penulis menentukan port 3030 untuk digunakan dalam komunikasi ini. Karena port ini tidak digunakan secara internal oleh sistem Windows. Dan port ini merupakan *Registered Port* yaitu port yang tidak digunakan oleh Windows, *range*-nya dari port 1024 hingga port 49151.

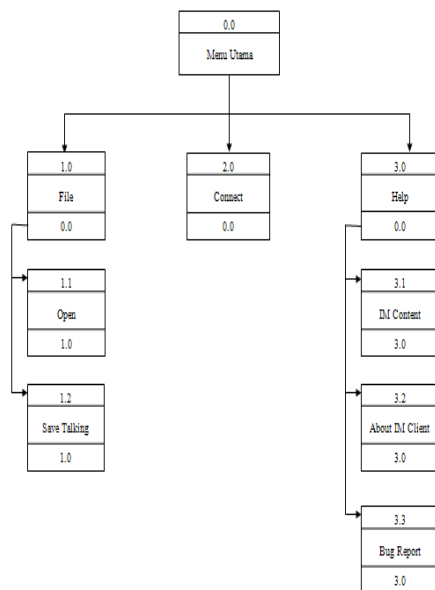
Agar dapat berkomunikasi lebih dari dua komputer, maka objek *winsock* pada *server* dibuat menjadi array. Pada saat akan melakukan komunikasi, objek *winsock* ini harus di-*load* terlebih dahulu. Dan jika koneksi telah berakhir, hendaklah meng-*unload* objek yang tidak terpakai tersebut. Dengan adanya aplikasi ini diharapkan komunikasi data akan lebih baik dan ditunjang dengan utilitas-utilitas yang ditambahkan.

3.2. Spesifikasi Rancangan Program

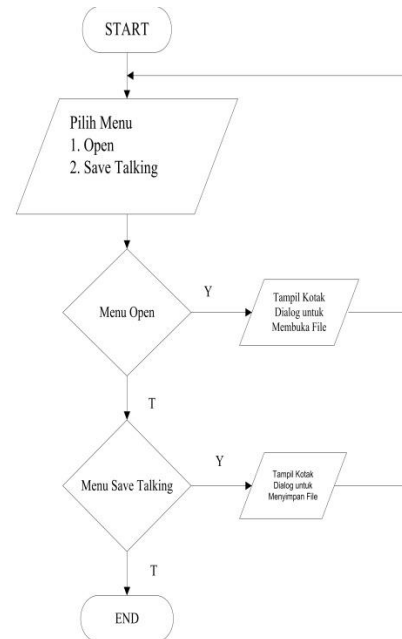
3.2.1. HIPO



Gambar 1. HIPO Menu Utama IM Server

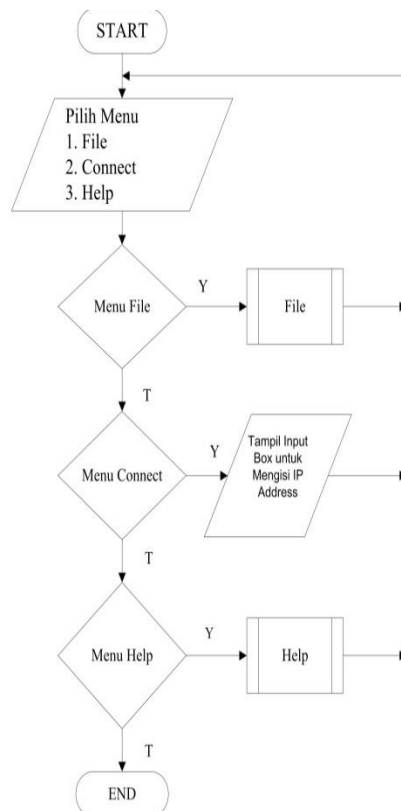


Gambar 2. HIPO Menu Utama IM Client

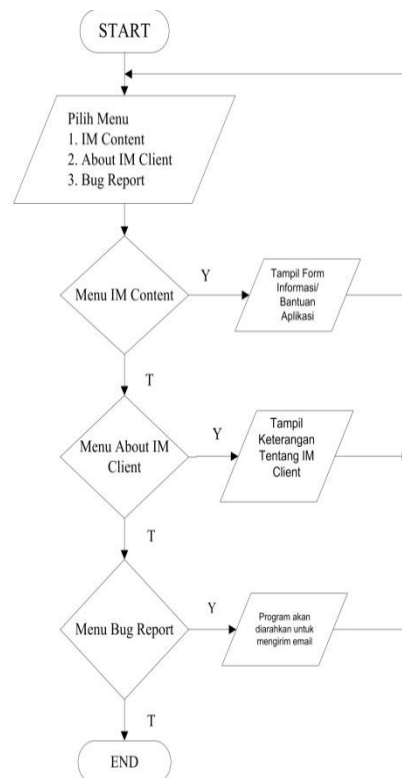


Gambar 4. Flowchart Menu 2 di IM Client

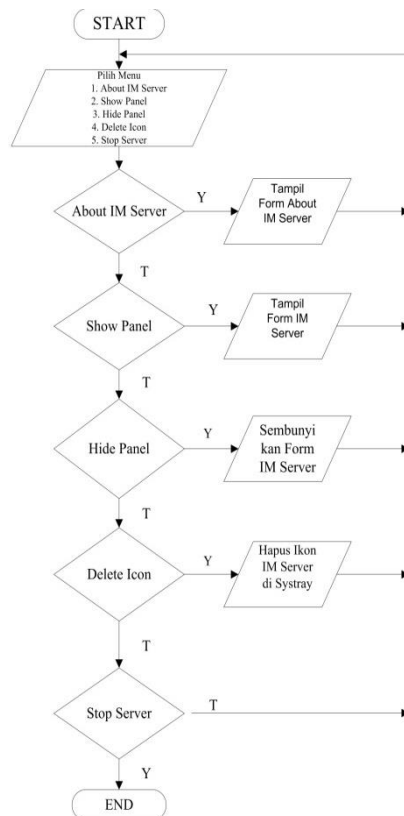
3.2.2. Flowchart Program



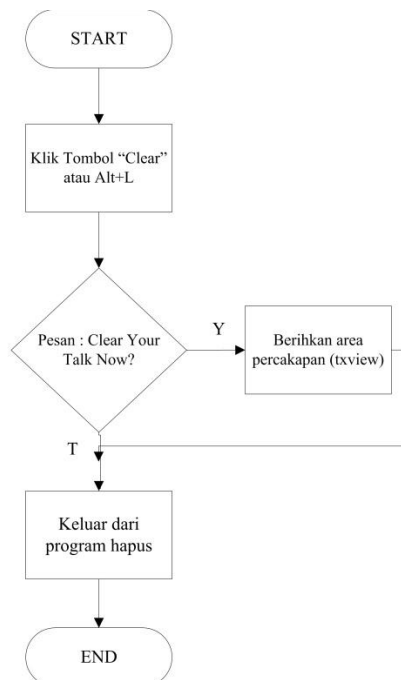
Gambar 3. Flowchart Menu 1 di IM Client



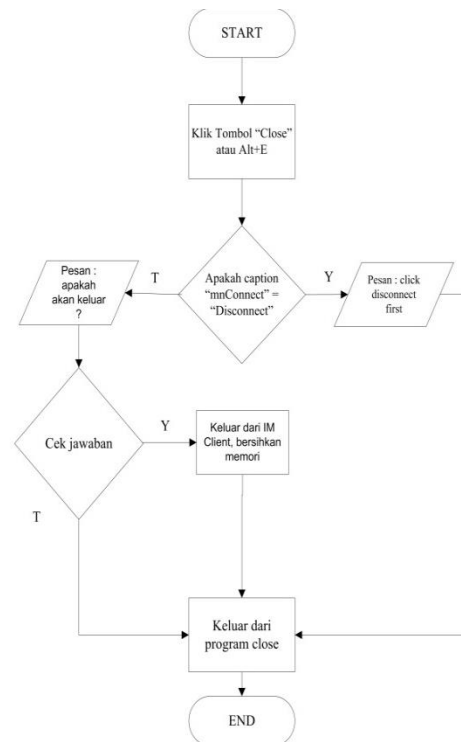
Gambar 5. Flowchart Menu 3 di IM Client



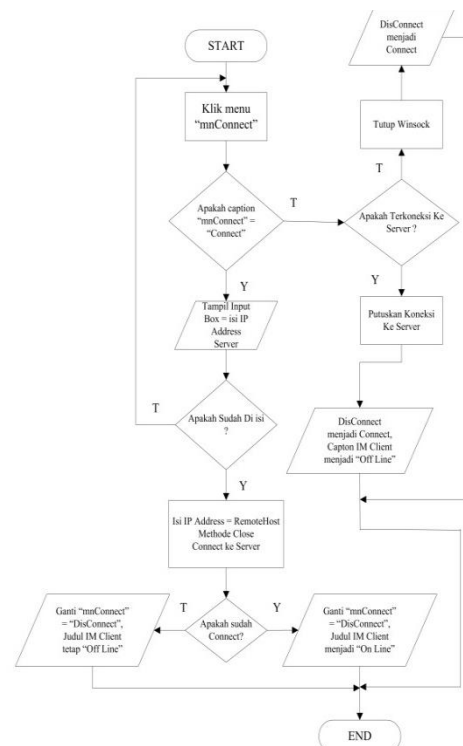
Gambar 6.
Flowchart Menu Utama IM Server



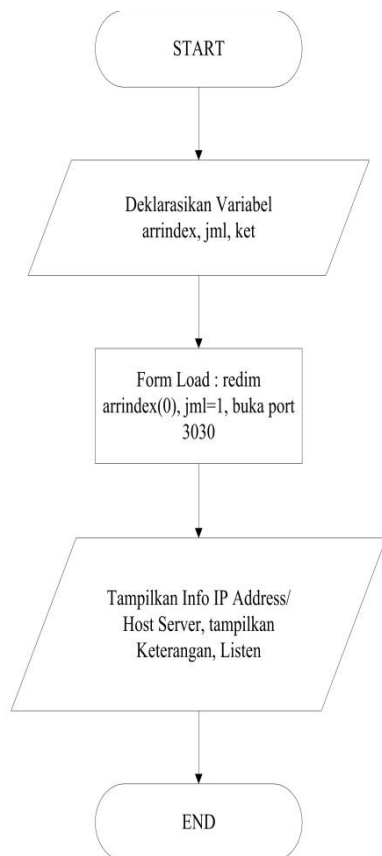
Gambar 7. Flowchart perintah Clear.



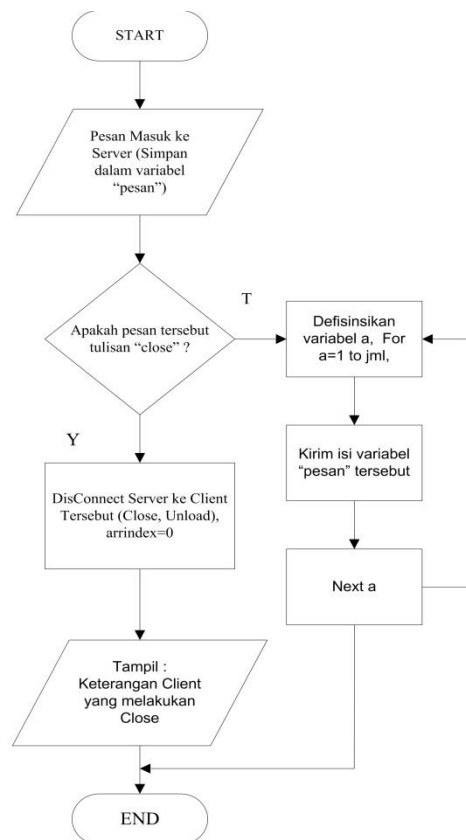
Gambar 8. Flowchart Perintah Close



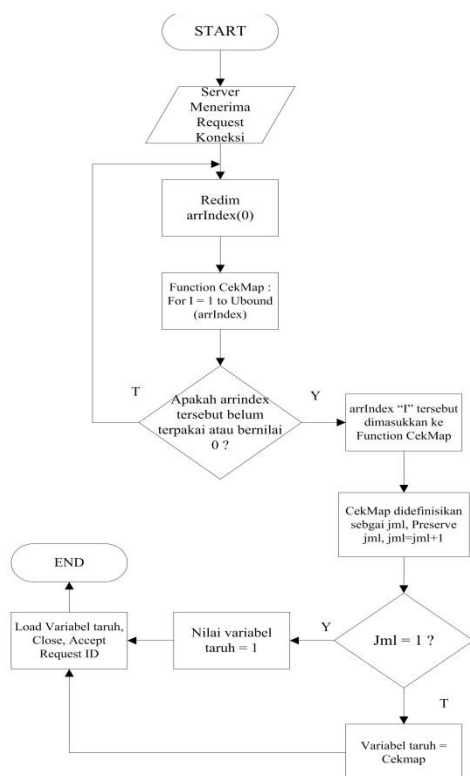
Gambar 9.
Flowchart Connect/Disconnect ke Server



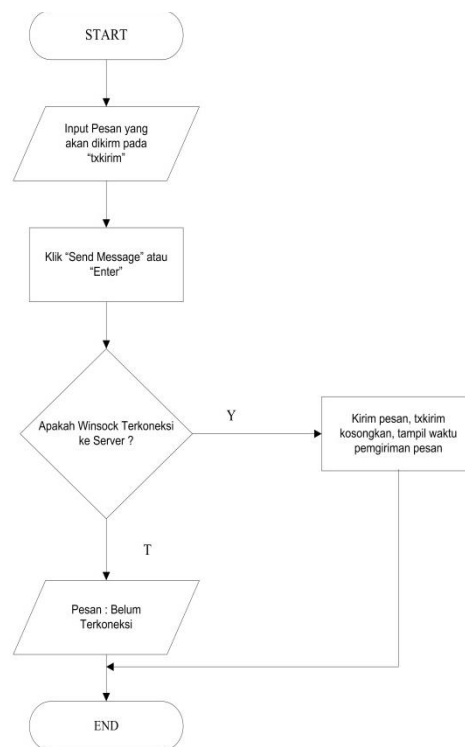
Gambar 10. Flowchart IM Server dalam kondisi *Listen*



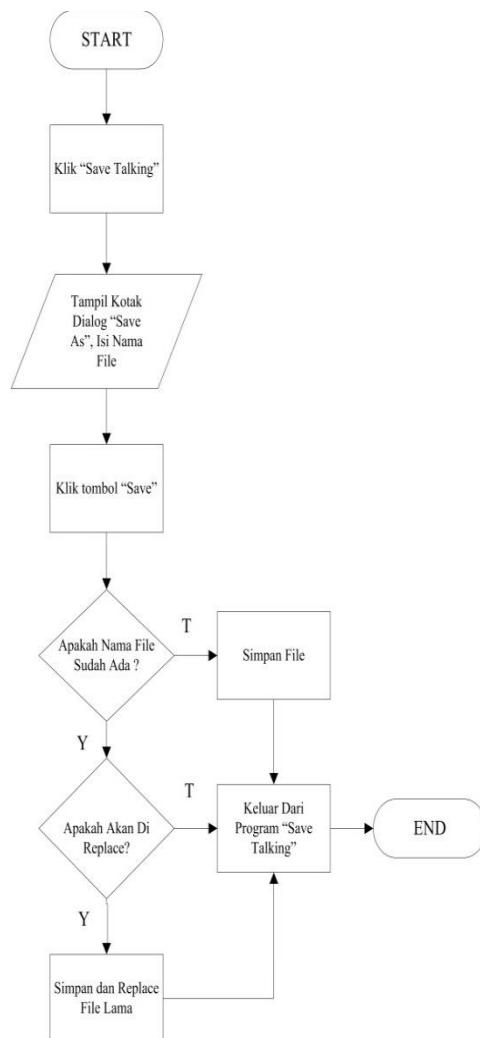
Gambar 12. Flowchart IM Server Menerima dan Mengirimkan Pesan



Gambar 11. Flowchart IM Server Menerima koneksi dari client



Gambar 13. Flowchart Pengiriman Pesan



Gambar 14. Flowchart Perintah Save Talking

3.2.3. Spesifikasi Bentuk Masukan

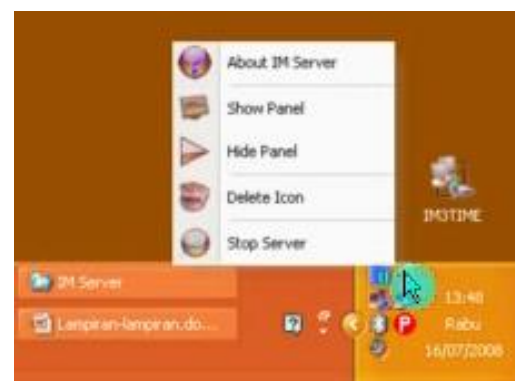
Langkah awal untuk memulai menggunakan program ini yaitu dengan menjalankan terlebih dahulu aplikasi *Intranet Messenger Server* yang selanjutnya akan ditulis IM Server dan hanya dijalankan di satu computer yang menjadi server.

IM Server ini nantinya akan mengatur komunikasi dari *client-client* yang *connect* dan *disconnect* serta mem-*broadcast* pesan ke seluruh *client*. Di jendela IM Server akan ditampilkan informasi *IP Address* dan nama komputer yang nanti akan dihubungi oleh IM Client.

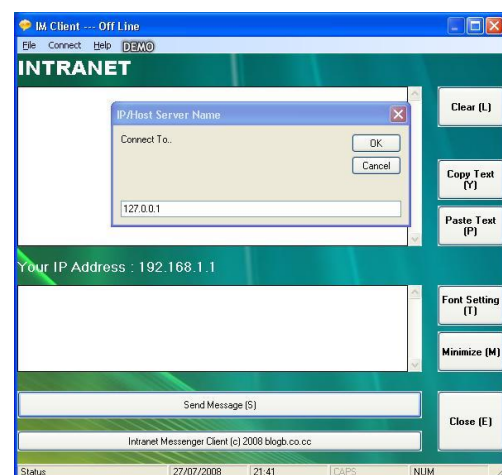
Setelah IM Server berjalan, selanjutnya aplikasi IM Client dijalankan. Sebagai bentuk masukan pertama, IM Client akan meng-*input* nomor IP Address ataupun nama komputer dari IM Server yang sedang aktif. Setelah sebelumnya membuka menu *connect* pada menu utama IM Client. Seperti yang ditampilkan pada Gambar 15,16, dan 17.



Gambar 15. Intranet Messenger Server saat pertama dijalankan



Gambar 16. IM Server bisa diakses di system tray Windows



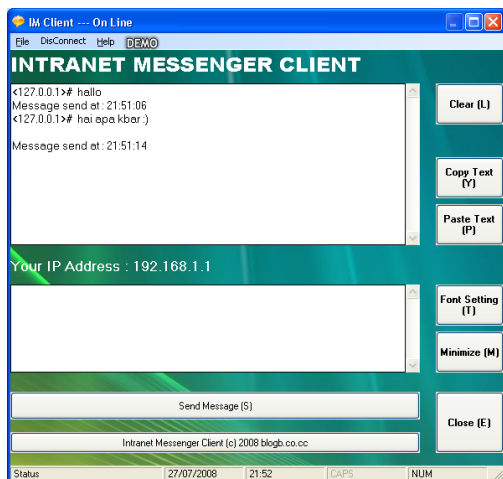
Gambar 17.
IM Client saat input IP Address

3.2.4. Spesifikasi Bentuk Keluaran

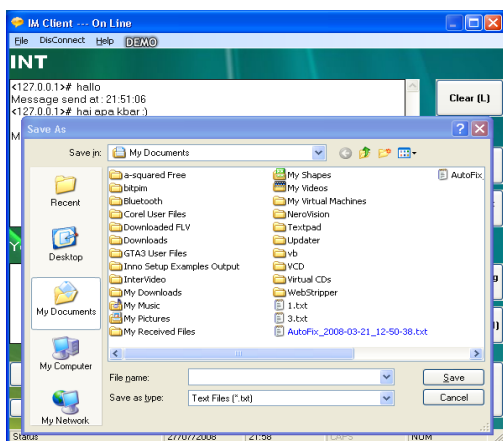
Sebagai bentuk keluarannya, setelah *client* bisa menghubungi *server* pengiriman pesan pun bisa dilakukan. Seperti yang ditampilkan pada Gambar 18.

Tampilan percakapan selain ditampilkan di area pesan, juga bisa disimpan ke dalam file teks biasa. Yang nantinya

digunakan untuk *history*. Percakapan disimpan dengan cara menekan tombol Ctrl+S ataupun membuka menu File lalu *Save Talking*. Seperti yang ditampilkan pada Gambar 19.



Gambar 18.
Contoh pengiriman pesan di IM Client



Gambar 19. Menyimpan *history* pesan

4. Kesimpulan

Aplikasi *socket* yang menggunakan TCP memerlukan pertukaran data dua arah yang *valid*. Ini ditandai dengan adanya perintah *listen* dan *connect* sehingga menjamin reliabilitas antara *server* dan *client* dalam berkomunikasi.

Walaupun aplikasi *socket* ini dicontohkan dalam *environment* Windows, bukan berarti aplikasi *socket* yang menggunakan *winsock* tidak dapat berkomunikasi dengan aplikasi *socket* berbasis Unix/Linux, karena komunikasi tetap dapat terjadi selama aplikasi tersebut menggunakan protokol jaringan yang sama.

Berbicara mengenai komunikasi jaringan, ternyata erat kaitannya dengan keamanan jaringan. Port yang terbuka dengan keluar masuknya data, sebenarnya juga

merupakan celah keamanan yang dapat dimanfaatkan pihak-pihak tertentu yang berniat buruk.

Karena itu, pada saat mengembangkan sebuah aplikasi *socket*, maka perlu dipertimbangkan pula *issue* keamanan jaringan dan sistem komputer yang digunakan. Jika data yang dikirimkan merupakan data yang sensitif dan rahasia, maka dapat dilakukan enkripsi data ataupun tindak pengamanan lainnya.

Dalam komunikasi data didalam jaringan, maka perlu dipertimbangkan pula besarnya data yang lalu-lalang pada jaringan, baik dengan menggunakan TCP maupun UDP. Keduanya harus dipersiapkan untuk mampu menangani data yang besar jika memang pengguna aplikasi *socket* tersebut sangat luas.

Referensi

- Anharku. (2017, March 17). *Flowchart*. Retrieved from Flowchart: <http://ilmukomputer.org/wp-content/uploads/2009/06/anharku-flowchart.pdf>
- Kurniawan, Y. (2006). Belajar Pemrograman dari Dasar. In *Belajar Pemrograman dari Dasar*. Jakarta: PC+.
- Masya, F., & Fiade, A. (2011). *Socket Programming*. Yogyakarta: Graha Ilmu.
- Nugroho, A. S. (2011). *Implementasi Winsock Berbasis TCP/IP Untuk Membangun Aplikasi Administrator*. Yogyakarta: STMIK AMIKOM.
- Nurjadi, J. (2008). *Pemrograman Socket Jaringan*. Jakarta: PC Media.
- Prasetyo, D. D. (2006). *101 Tip Trik Visual Basic 6.0*. Jakarta: PT.Elex Media Komputindo.
- Sopandi, D. (2008). *Instalasi dan Konfigurasi Jaringan Komputer*. Bandung: Informatika.
- Subakti, I. (2003). *Jaringan Komputer dan Internet*. Surabaya: Lembaga Penelitian ITS.
- Viva, V. (2008). *Trik Pemrograman Jaringan dengan Visual Basic 6*. Yogyakarta: Gava Media.
- Wehmann, A. (2017, March 17). *Visual Basic Array Tutorial*. Retrieved from Visual Basic Array Tutorial: <http://www.patorjk.com/programming/tutorials/vbarrays.htm>