Perancangan Control Panel untuk Cluster OpenMosix

Rachman
Singgih Jatmiko
A. Benny Mutiara Q. N.

Abstrak

Openmosix Sebagai cluster yang berjalan pada level kernel dapat megalihkan pekerjaan apa saja dan kapan saja tanpa harus aplikasi tersebut dirancang untuk cluster. Dengan menggunakan konsep load-balancing maka openmosix akan mengalihkan kerja server bila server tersebut telah penuh dengan beban. Pengawasan terhadap cluster secara langsung dengan berbasiskan web akan menjadi lebih mudah.

Kata kunci : WEB, Clustering, OpenMosix, PHP, Control Panel

1. Pendahuluan

Alasan utama untuk memilih penelitian ini adalah keinginan untuk membuat suatu tools yang sangat bermanfaat untuk administrator sistem. Tools ini akan mempermudah administrator dalam melakukan pengawasan langsung terhadap cluster openmosix. Pengawasan langsung ini tentunya dapat dimanfaatkan untuk memantau kinerja cluster openmosix dan memanfaatkannya untuk berbagai kebutuhan yang berhubungan dengan kinerja sistem tersebut. Tools ini dibuat dengan memanafaatkan API openmosix dan kemudian membuat control panel, untuk memudahkan administrator sistem tools ini juga dapat digunakan secara remote atau jarak jauh dengan menggunakan web.

Metode yang digunakan dalam penelitian ini adalah dengan melakukan studi literatur pada awalnya dan kemudian melakukan percobaan termasuk pembuatan jaringan cluster openMosix, pembuatan panel kontrol web dan melakukan pengujian dan analisa terhadap panel kontrol yang dibuat.

Pembahasan diorganisasikan dengan membahas latar belakang pengetahuan, perancangan kontrol panel, pengujian dan terakhir adalah penutup.

2. Latar Belakang Pengetahuan

Suatu program paralel memerlukan koordinasi ketika sebuah tugas bergantung pada tugas lainnya. Ada dua macam bentuk koordinasi pada komputer paralel: asynchronous dan synchronous. Bentuk synchronous merupakan koordinasi pada hardware yang memaksa semua tugas agar dilaksanakan pada waktu yang bersamaan dengan mengesampingakan adanya ketergantungan tugas yang satu dengan lainnya. Sementara bentuk asynchronous mengandalkan mekanisme pengunci untuk mengkoordinasikan prosesor tanpa harus berjalan bersamaan.

OpenMosix merupakan tools untuk kernel yang termasuk jenis keluarga unix, seperti linux. Terdiri dari algoritma-algoritma penggunaan sumber daya bersama yang dapat disesuaikan. Open mosix memperbolehkan multiple Uniprocessors (UP) dan Simetric Multiprocessors (SMP) menjalankan kernel yang sama untuk bekerja dalam pekerjaan yang mirip. Algoritma-algoritma penggunaan sumber daya bersama dirancang untuk merespon langsung bermacam-macam pemakaian sumber daya pada setiap node. Hal ini dicapai dengan memindahkan proses dari satu node ke node yang lain, secara preemsi dan transparan, untuk load-balancing dan untuk mencegah terjadi tumbukan yang berhubungan dengan penukaran memori. Tujuannya untuk membuktikan performa cluster-wide dan untuk menciptakan multiuser yang sesuai, lingkungan time-sharing untuk mengerjakan aplikasi sequensial dan paralel. Standar runtime dari open mosix adalah computing cluster, dimana sumber daya cluster-wide tersedia untuk setiap node .

Teknologi OpenMosix terdiri dari 2 bagian yaitu Mekanisme Preemtive Process Migration (PPM) dan algoritma untuk penggunaan bersama sumber daya yang dapat disesuaikan. Keduanya

diimplementasikan pada level kernel menggunakan modul yang bisa dipanggil, sehingga pengantarmukaan kernel tetap tidak dirubah, meskipun benar-benar transparan terhadap level aplikasi.

PPM dapat memindahkan proses apa saja, kapan saja, ke node yang tersedia. Biasanya, Perpindahan berdasarkan pada informasi yang disediakan oleh salah satu dari algoritma penggunaan bersama sumber daya, akan tetapi pengguna dapat mengesampingkan beberapa sistem pengambilan keputusan secara otomatis dan memindahkan proses mereka secara manual.

Algoritma penggunaan sumber daya bersama yang utama dari openMosix yaitu load-balancing dan pengantar memori. Kekuatan algoritma load-balancing terus menerus mencoba untuk mengurangi perbedaaan load di antara pasangan node-node, dengan memindahkan proses-proses dari loaded yang tinggi ke loaded yang lebih sedikit. Skema ini menyebarkan semua node-node yang mengeksekusi algotritma-algoritma yang sama, dan pengurangan dari perbedaan load ditampilkan secara independen oleh sepasang node. Jumlah prosesor pada setiap node dan kecepatannya merupakan faktor yang penting dalam algorita load-balancing. Algoritma ini menanggapi perubahan-perubahan load dari node-node atau karakteristik runtime dari proses-proses tersebut. Hal ini berlaku selama tidak ada kekurangan yang berarti dari sumber daya yang lain seperti free memori atau slot proses yang kosong.

OpenMosix mendukung perpindahan proses (PPM) preemsi dan benar-benar transparan. Setelah perpindahan, sebuah proses berlanjut untuk berintraksi dengan lingkungannya tanpa memperhatikan lokasinya. Untuk mengimplementasikan perpindahan proses (PPM), proses perpindahan dibagi dalam dua konteks: user context, dimana bisa dipindahkan, dan system context, yaitu tidak tergantung dengan UHN dan tidak boleh dipindahkan.

User context, biasa disebut dengan remote, berisi kode program, stack, data, peta memori, dan register dari proses. Remote meringkas proses ketika bekerja di level user. System context, biasa disebut deputy, terdiri dari penjabaran dari sumber daya dimana proses itu terhubung, dan sebuah kernel-stack untuk pengeksekusian dari kode system pada keseluruhan proses. Deputy meringkas proses ketika bekerja dalam kernel. Deputy menahan bagian dependent dari system context pada proses; karenanya deputy harus tetap berada dalam UHN dari proses. Sementara proses dapat sering berpindah-pindah diantara node-node yang berbeda, deputy tidak pernah berpindah. Pengantarmukaan antara user-context dan system context sudah didefinisikan dengan jelas. Maka dari itu sangat mungkin untuk memberhentikan setiap interaksi antara context-context ini, dan melanjutkan interaksi ini lewat jaringan. Hal ini diimplementasikan pada layer link, dengan jalur komunikasi spesial untuk interaksi.

Waktu untuk perpindahan memiliki komponen tetap, untuk membangun suatu kerangka proses yang baru pada situs remote baru, dan komponen linear, terbagi rata ke sejumlah memory pages untuk ditransfer.

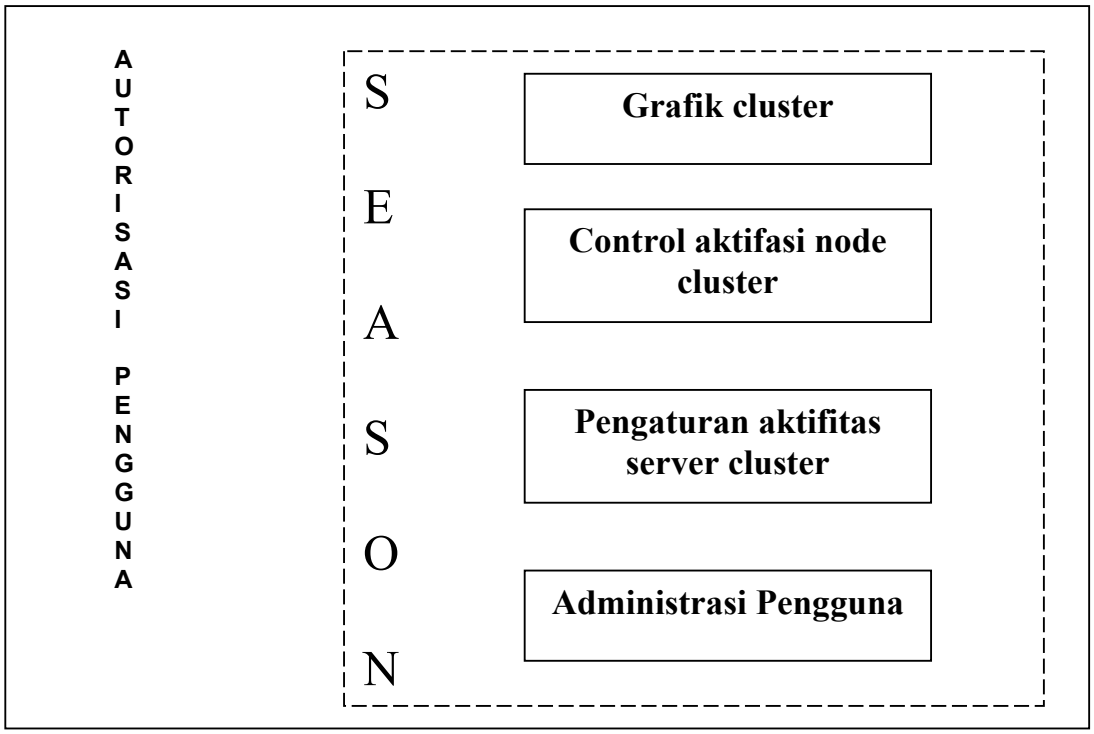
Untuk pengeksekusian sebuah proses dalam OpenMosix, transparansi lokasi dapat dicapai dengan melanjutkan pemanggilan system site-dependent ke deputy pada UHN. Pemanggilan system (system calls) adalah suatu bentuk synchronous dari interaksi antara konteks dua proses. Semua system calls yang dieksekusi oleh proses tersebut diberhentikan oleh link layer situs remote. Jika system call independent terhadap situs maka system tersebut dieksekusi secara local oleh remote. Jika tidak, system call dilanjutkan ke deputy, yang mengeksekusi system call yang mewakili proses dalam UHN. Deputy mengirimkan kembali hasil ke situs remote, dan melanjutkan untuk mengeksekusi user's code.

Bentuk lain dari interaksi antara konteks dua proses adalah pengiriman melalui sinyal dan event proses wakeup, seperti ketika data dari suatu jaringan dating. Event-event ini membutuhkan bahwa deputy menempatkan secara asynchronous dan berinteraksi dengan remote.

Kebutuhan lokasi ini dipertemukan oleh jalur komunikasi di antara mereka. Dengan skenario yang mirip, kernel yang berada pada UHN menginformasikan kepada deputy dari event tersebut. Deputy memeriksa apakah ada suatu tindakan yang perlu diambil, dan jika memang maka menginformasikan kepada remote. Remote memonitor jalur komunikasi untuk laporan dari event asynchronous, seperti sinyal, tepat sebelum memulai eksekusi user-level. Pendekatan cara ini dapat dikatakan kuat, dan tidak terpengaruh bahkan oleh modifikasi atau perubahan yang besar dari kernel

tersebut. Hal ini tergantung pada hampir fitur-fitur no machine-dependent dari kernel, dan hal ini tidak menghalangi jalur ke arsitektur-arsitektur yang berbeda.

3. Perancangan Kontrol Panel



Gambar 1. Bagan Control Panel

Kontrol panel ini terdiri dari:

1. Autorisasi Pengguna, yang dapat mengakses control panel ini adalah pengguna yang telah terdaftar pada database bengguna.
2. Season, merupakan suatu fungsi yang ada di PHP yang berguna untuk mengenal semua pengguna yang masuk.Ini penting karena control panel yang dibuat bersifat khusus dan tidak semua pengguna bisa mengaksesnya. Season ini terdiri dari:
    a. Grafik Cluster, merupakan modul yang dibust memanfaatkan API OpenMosix agar dapat menampilkan jenis-jenis kegiatan seperti Load setiap processor ,penggunaan Memori, CPUS pada cluster. API dari OpenMosix berada pada /tmp/openmosixcolector. Contoh apabila file ini diakses maka data yang terbaca dari file tersebut adalah :
    13.11.2003-20.52.50 0 200 5376 22 4
    Arti dari data yang tebaca diatas sebagai berikut :
    13.11.2003-20.52.50 : waktu proses dari openmosix, terdiri dari : tanggal dan jam
    0 : Load dari processor
    200 : Balancing dari prosessor
    5376 : Memori yang tersedia
    22 : Memoori yang terpakai dalam mengerjakan tugas
    4 : total node yang tersambung ke server.

    b. Control Aktifasi Node Cluster, karena mekanisme kerja Openmosix menggunakan metode Load-Balancing maka pada level kerja tertentu semua node tidak perlu diaktifkan. Hal ini bisa dilakukan dengan memutuskan hubungan dari server ke node

cluster melalui alamat IP pada /etc/hosts yang ada di server, maka beban kerja yang dibuang dari server tidak akan sampai ke node yang terputus hubungannya dari server.

c. Pengaturan Aktifitas Server Cluster, aktifitas server berbasis linux dapat dilihat dengan perintah TOP pada consol, namun dengan fungsi yang telah tersedia di PHP memungkinkan untuk memantau aktifitas server dari web. Selain untuk memantau aktifitas dari server, kita juga bisa secara langsung menghentikan aktifitas yang tidak diperlukan dari web.
d. Administrasi Pengguna, perancangan administrasi pengguna ini untuk mendukung fungsi season yang dibuat, adapun data-data pengguna tersebut disimpan pada database. Disini digunakan MySQL sebagai database karena ringan dalam pengolahan data dan alasan lisensi. Table database dapat dilihat pada tabel berikut

Tabel 1. Tabel Database Pengguna

Nama Field	Tipe Data	Fungsi
ID_Pengguna	VarChar < 30>	ID pengguna sifatnya unik
Nama_Pengguna	VarChar < 30 >	Nama pengguna
Passwod_Pengguna	VarChar <30>	Password Pengguna

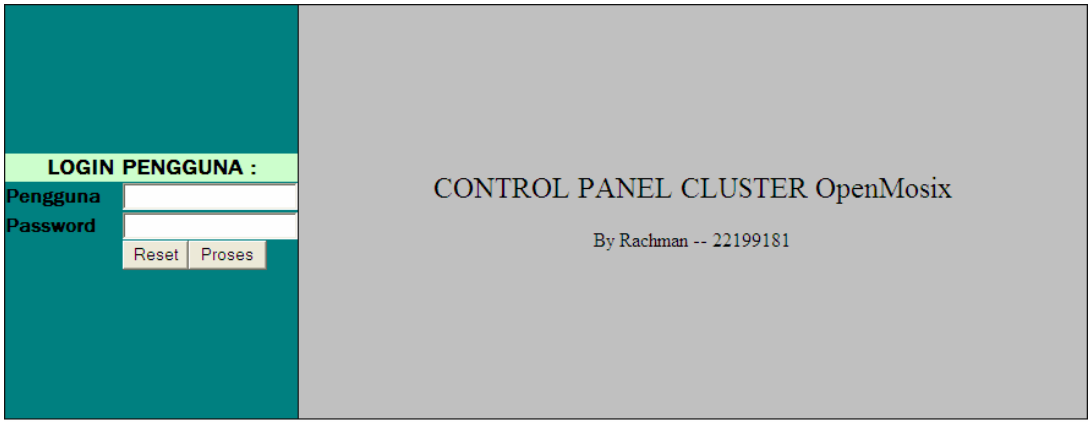
4. Pengujian Kontrol Panel Cluster

Pengujian kontrol panel cluster dilakukan dengan menggunakan spesifikasi hardware sebagai berikut

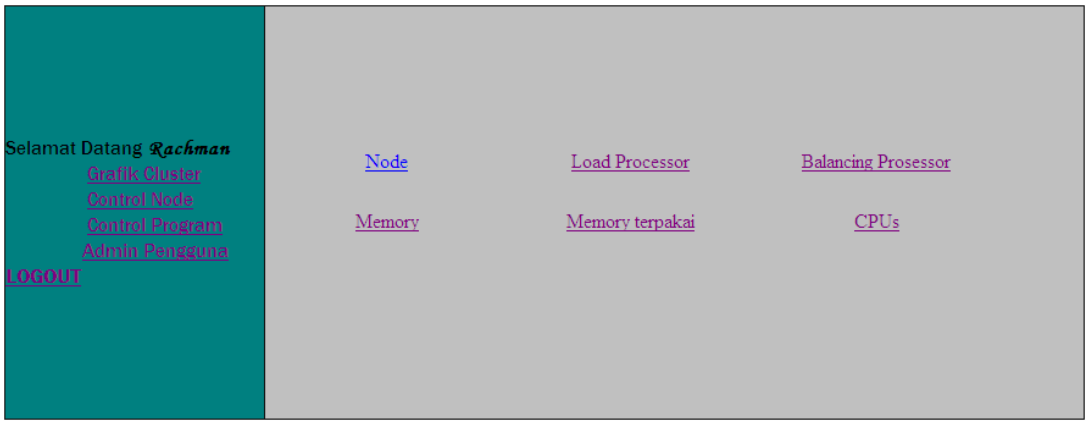
Server :
- Processor AMD Duron 1300 MHz On Board Motherboard ECS K7SOM+
- Memori DDR 256 MB
- Hard Disk 40 GB 5400 RPM
- LAN Card 10/100 MBPS Edimex
- Switch 100 MBPS
- VGA On board 32 MB share memory
- Sistem Operasi Linux RedHat 9.0
- Apache Server
- PHP 4

Client :
- Processor AMD Duron 1300 MHz On Board Motherboard ECS K7SOM+
- Memori DDR 128 MB
- Hard Disk 40 GB 5400 RPM
- LAN Card 10/100 MBPS Edimex
- VGA Onboard 32 MB Share memory
- Sistem Operasi Linux RedHat 9.0

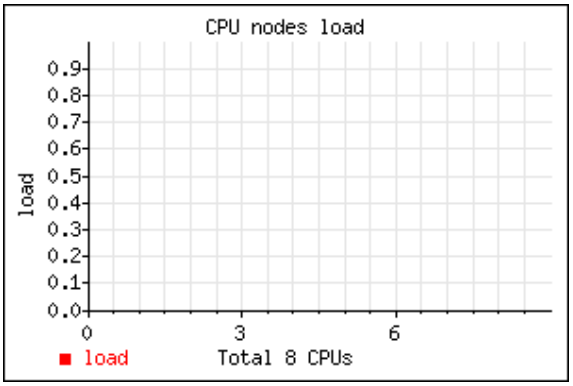
Pengujian cluster menggunakan web server akan menampilkan data-data kerja dari server open mosix dan clientnya dan bekerja dengan baik seperti terlihat pada gambar gambar dibawah.



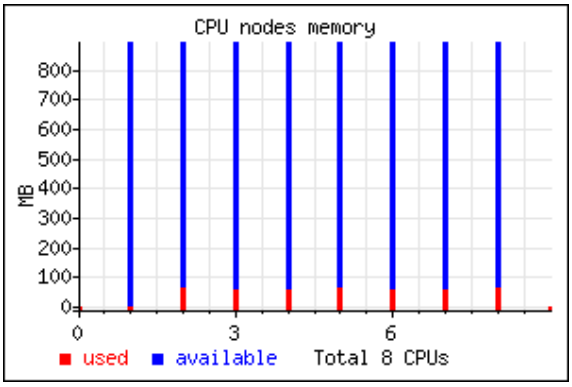
Gambar 2. Tampilan awal Web Control Panel



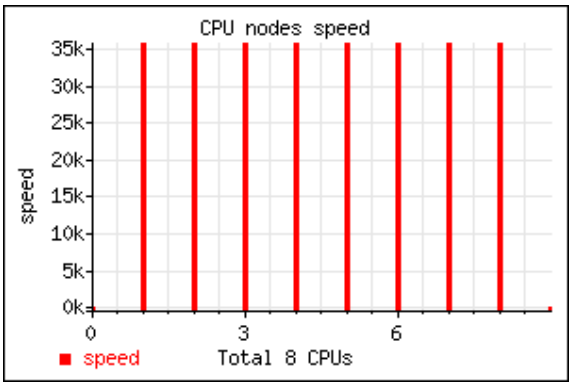
Gambar 3. Tampilan Menu Web Control Panel



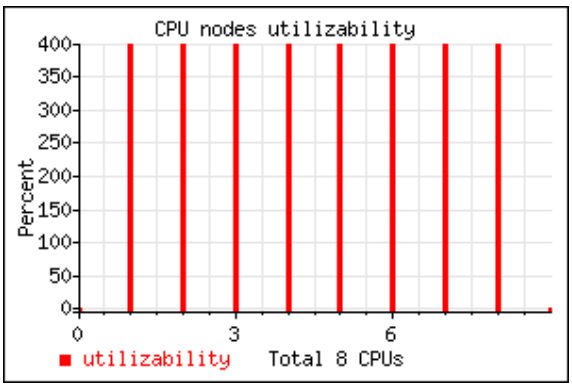
Gambar 4 Tampilan Monitoring Web Openmosix



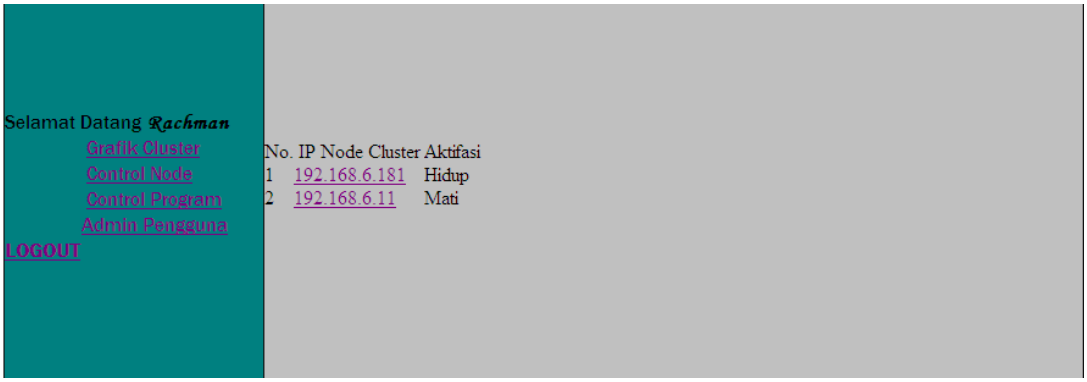
Gambar 5. Tampilan Memori Load



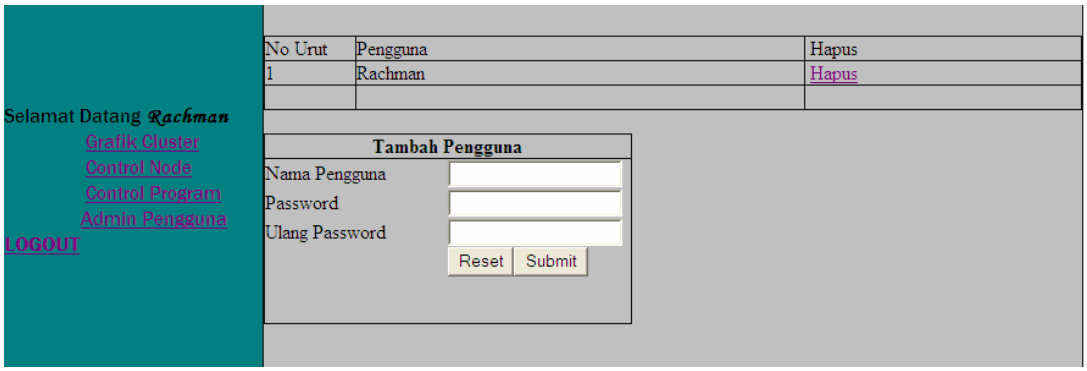
Gambar 6. Tampilan Load Processor



Gambar 7. Tampilan Utilisasi Cluster



Gambar 8 Tampilan Control Node



Gambar 9. Tampilan administrasi Pengguna

5. Penutup

Dari hasil yang diperoleh selama pembuatan contol panel cluster openMosix maka dapat mengambil kesimpulan bahwa pengontrolan cluster tidak selalu melalui consol linux, hal ini dkarenakan API OpenMosix yang memudahkan pembuatan pengantarmukaan untuk pengontrolan cluster tersebut bahkan dengan pembuatan control panel ini akan sangat memudahkan sistem administrator untuk mengetahui kerja mesin clusternya.

Dari pengalaman yang dialami selama penulisan Tugas Akhir ini maka ada beberapa hal-hal yang menjadi saran dari penulis, yaitu :

- Pembuatan pengantarmukaan pengalihan beban kerja untuk lebih memudahkan pengontrolan secara jarak jauh.
- Pembuatan pengantarmukaan untuk menjalankan program secara langsung dari web.

6. Daftar Pustaka

[1] Betha Sidik, Ir., *Pemrograman Web Dengan PHP,* INFORMATIKA Bandung 2002
[2] Firrar Utdirartatmo, *Pemrograman Paralel dengan PVM di LINUX dan WINDOWS, ANDI* Yogyakarta, 2002.
[3] Anonim, *Pemrograman Paralel di LINUX Berbasis DSM (Distributed Shared Memory),* ANDI Yogyakarta, 2003.
[4] AnonimThe userland-tools of the openMosix-system , http://openmosix.sourceforge.net/
[5] Instalasi Cluster dengan openMosix, URL : http://www.clustering.org
[6] Kompilasi dan Konfigurasi Kernel, URL : http://www.kernel.org
[7] Michael J. Quinn, *Parallel Computing : Theory and Practice Second Edition*, 1994.
[8] Mulyadi Santosa, *Clustering di Linux dengan OpenMosix*
[9] Ted G. Lewis & Hesham El-rewini, *Introduction to Parallel Computing*, Prentice-Hall Internasional Editions, 1992.