Tanggal :
Lokasi :

Susun!

ALAT UKUR KONSEPTUAL

1. Sistem Workload

Workload sistem komputer memiliki properti statistik yang tidak berubah untuk periode yang lama. Maka memungkinkan untuk :

- Mengkarakteristikkan workload berdasarkan distribusi permintaan yang dihasilkan pada sistem sumber secara individual.
- Mendefinisikan unit kerja dan mengirimkan workload ke setiap unit.

Prinsip Karakteristik model workload

- Representatif
- Fleksibel
- Sederhana dalam konstruksi
- Padat
- Harga kegunaan
- Sistem independen
- Kemampuan reproduksi
- Kompatibilitas

Karakteristik Workload

Parameter Workload	**Deskripsi**
Waktu Job CPU	Total waktu CPU yang dibutuhkan oleh suatu pekerjaan
Request Job I/O	Total jumlah operasi I/O yang dibutuhkan oleh suatu pekerjaan
Waktu layanan CPU	Waktu CPU yang dibutuhkan untuk memproses suatu tugas CPU
Waktu layanan I/O	Waktu I/O yang dibutuhkan untuk memproses suatu tugas IO
Waktu Interarrival	Waktu antara dua permintaan yang berturut-turut untuk layanan sistem
Prioritas	Prioritas yang ditetapkan untuk pelaksanaan tugas oleh user
Waktu Blocked	Waktu ketika suatu perkerjaan tidak sanggup dilayani oleh layanan CPU
Request Memori	Jumlah permintaan memori yang dibutuhkan oleh suatu tugas
Set Ukuran	Jumlah halaman dari suatu perkerjaan yang

Susun!

Pekerjaan	harus ada pada memori utama
Referensi Lokal	Waktu untuk semua memori referensi yang dihasilkan oleh suatu pekerjaan dalam suatu halaman atau dalam set beberapa halaman
Waktu respon User	Waktu yang dibutuhkan user dalam terminal interaktif untuk membuat request baru (think dan tipe waktu)
Intensitas User	Waktu Proses per request / waktu respon user
Jumlah user simultan	Jumlah user interaktif yang login bersamaan
Jumlah job dalam sistem	Jumlah job atau tugas yang sedang dilayani atau menunggu dalam antrian pada suatu sistem sumber
Mix Instruksi	Frekuensi Relatif dari tipe instruksi yang berbeda pada sistem yang harus dieksekusi

Model Workload

Kinerja adalah reaksi dari suatu sistem untuk suatu workload yang spesifik. Oleh karena itu, untuk mengevaluasi kinerja harus memilih workload yang tepat. Karakteristik workload harus cukup presentatif untuk menghitung semua faktor yang signifikan.

Model workload server sebagai model pengendali sistem komputer yang sesungguhnya selama percobaan pengukuran kinerja atau sebagai input ke model dari sistem yang dievaluasi.

Kegunaan model workload :

- Menyediakan workload yang representatif untuk perbandingan evaluasi kinerja dari sistem yang berbeda.
- Menyediakan reproduksi yang terkendali pada lingkungan untuk percobaan optimisasi kinerja.
- Mengurangi kuantitas data yang harus dianalisa.
- Menghasilkan workload sistem pada form sesuai yang dibutuhkan oleh sistem model.

Workload dalam dunia nyata tidak dapat direproduksi secara normal. Tetapi jika properti statistik dari sistem workload tidak diubah berdasarkan itu, maka workload secara *secara statistik dapat direproduksi*.

Sistem workload tetap tidak berubah untuk waktu yang sangat lama, tapi biasanya, karakteristiknya berubah sedikit-sedikit bersamaan dengan perubahan pada komunitas user. Jika suatu aplikasi baru ditambahkan, maka

Susun!

aplikasi yang lama akan terhenti. Dari situlah dapat kita nyatakan bahwa tren pada komunitas user dapat mengadaptasi perubahan pada suatu sistem juga. Dalam jangka panjang, workload pada sistem nyata tidak dapat di reproduksi. Maka dari itu, percobaan tidak untuk mengendalikan sistem input dan ini sangat sulit untuk mengetahui efek dari karakteristik workload yang berbeda dari suatu sistem. Oleh karena itulah, lebih disukai dan kadangkala lebih dibutuhkan untuk membangun pengendali workload yang khusus, yang tetap mungkin secara fisik dalam mengukur sistem tersebut pada keadaan workload dunia nyata.

Model dari model workload

- **Instruction mix**. Penggunaan relatif dari instruksi yang berbeda tipe dalam aplikasi khusus. Kinerja (rating eksekusi suatu instruksi) dari set instruksi prosesor harus dapat dievaluasi dengan baik sesuai dengan mix intruksi yang diminta.
- **Benchmarks**. Benchmarks didefinisikan sebagai *"titik referensi dari pengukuran yang dapat dilakukan* " (Sipp 1972). Benchmark dapat berupa instruksi, program spesial atau urutan pemanggilan kepada komponen perangkat lunak yang dipilih. Secara umum, istilah benchmark digunakan sebagai *titik pertengahan suatu job atau kumpulan job yang merepresentasikan workload yang khusus dari sistem yang dievalusasi*. Benchmark sangat penting untuk menyediakan pengendali workload pada sistem nyata, keduanya digunakan untuk melakukan evaluasi perbandingan dari sistem yang berbeda dan optimalisasi kinerja. Benchmark yang baik dapat menguji semua fungsi dari sistem dengan cara menggunakan fungsi-fungsi tersebut seakan-akan dalam lingkungan pekerjaan yang aktual. Konstruksi benchmark dari job yang sesungguhnya harus pantas untuk sistem yang bergantung satu sama lain. Biasanya, setiap benchmark tidak harus berguna sebagai pengendali workload dari sistem yang berbeda. Untuk mengevaluasi kinerja sistem untuk workload tertentu, kadangkala ditulis program khusus untuk melakukan benchmark. Pada setiap kasus, benchmark, juga mengadakan pengukuran dalam bentuk yang dibutuhkan sebagai aplikasi program baru dan antar mukanya dengan sistem tersebut. Kekurangan benchmark adalah tidak selalu memungkinkan untuk membangun pengendali workload dari suatu job yang yang dibuat dalam mewakili job yang sesungguhnya. Lainnya adalah juga karena alasan keamanan atau karena eksekusi suatu job akan mengubah suatu sistem secara permanen. Dan lagi tidak mudah untuk mencari job yang sesungguhnya yang sesuai dengan karakteristik dari kelas model yang khusus.
- **Sintetik Benchmark**. Model ini mensimulasikan pemakaian sistem sumber sebagai sesuatu yang sudah ditentukan oleh karakteristik model workload, tetapi ini bukan cara yang cukup baik. Sintetik benchmark terpisah dari konfigurasi sistem dan sistem operasi sistem

Tanggal :
Lokasi :

Susun!

yang diukurnya, berdasarkan syarat-syarat layanan dan dapat digunakan sebagai evaluasi perbandingan dari sistem yang berbeda. Syarat layanan ini dapat dihasilkan oleh pengukuran sistem pada workload yang sesungguhnya, atau dari hasil dari estimasi untuk workload yang dibebankan atau didesain untuk melakukan tes sistem dalam workload yang direpresentasikan secara terbatas atau dalam kondisi yang tidak biasa. Struktur syarat-syarat tersebut dapat digunakan langsung sebagai input dari pensimulasi sistem.

- **Trace**. Ini adalah rekaman dari even yang dipilih yang dipertahankan dalam urutan yang tetap, dimana even tersebut dapat didefinisikan sebagai perubahan keadaan sistem. Trace digunakan sebagai pengendali workload untuk model simulasi, khususnya untuk analisis dimana urutan pada pola pengukuran adalah sangat penting. Trace dipersiapkan dari sistem workload nyata atau dari hasil pencampuran beberapa metode benchmark yang representatif. Urutan hasil trace dapat diperoleh dari memonitor keadaan perangkat keras maupun perangkat lunak Trace dapat digunakan untuk mengukur kinerja secara teliti dan hati-hati sampai sebagian besar tujuan yang ada di dalamnya tidak dapat direpresentasikan lagi.
- **Model Workload Probabilistik**. Digunakan dalam studi analisis dan simulasi. Sumber yang diminta diasumsikan dalam variabel random dan workloadnya didefinisikan dalam distribusi tersebut. Secara berkesinambungan dapat diperkirakan oleh distribusi eksponensial,distribusi Erland, distribusi hiper eksponensial dan sebagainya. Kebanyakan karakteristik workload, sulit disetarakan dengan distribusi matematis yang sederhana, akan tetapi justru sebagian besar hasil dapat ditunjukkan dengan cara memprediksikan sistem nyata ke dalam persamaan matematis tersebut.
- **Driver Sistem Interaktif.** Workload dalam sistem yang interaktif memiliki dipengaruhi oleh karakteristik user, waktu berfikir, waktu mengetik, pembuatan interupsi oleh user dan sebagainya. Pengendalian workload untuk sistem yang interaktif bukanlah dilakukan melalui sebuah model worload, tetapi oleh suatu generator yang terdiri dari antarmuka user dan kebutuhan proses user. Model antarmuka user akan mengambil alih semua operasi sistem untuk diarahkan pada terminal milik user tersebut dan disimulasikan dengan aksi yang tepat. Komponen-komponennya dapat diimplementasikan sebagai program atau dalam suatu perangkat keras eksternal. Driver internalnya akan mengurangi jumlah beban sistem yang digunakan untuk memproses job-job user. Model dari kebutuhan proses user terdiri dari proses benchmark yang dieksekusi dari terminal hasil simulasi. Benchmark lainnya dilakukan secara sintetik atau dibangun dari perintah-perintah pada sistem real dan proses input data. Ini yang kemudian disebut dengan script. Hal ini memungkinkan untuk menggunakan model probabilistik dari model kebutuhan user yang

Susun!

karakteristiknya dapat digambarkan secara otomatis dari distribusi probabilitas.

2. Simulasi

Teknik ini mengkombinasikan teknik model dan teknik pengukuran. Proses simulasi membutuhkan

- model sistem,
- model workload, dan
- simulator, yaitu mekanisme yang mensimulasi tingkah laku sistem seperti yang dispesifikasikan oleh model fungsional dan model workload, serta mengumpulkan data yang dibutuhkan untuk analisis kinerja.

Model kinerja berasal dari kenyataan yang didapat dari pengukuran terhadap sistem real. Kesederhanaan harus dapat diusahakan. Model simulasi dapat memasukkan faktor-faktor yang sulit digabungkan dalam model analitik. Begitu juga bagi workload tidak dapat dideskripsikan oleh secara seimbang oleh distribusi probabilitas dari pendekatan penggunaan yang berkesinambungan. Untuk analisis kinerja, simulator harus hanya mensimulasi even-even yang mungkin dapat mengubah keadaan sistem tersebut. Simulator tipe ini disebut **discrete event simulator**, yaitu suatu sistem yang memelihara waktu simulasi yang berlanjut terus setelah setiap perubahan sistem oleh sejumlah variabel yang berhubungan langsung dengan sistem real yang selalu cepat berlalu sebelum perubahan selanjutnya tiba kembali. Hal inilah yang membuat simulasi menjadi lebih berdiri sendiri dari sistem komputer ketika dijalankan.

Detil simulasi harus dapat menjadi bahan pertimbangan, detil yang berlebihan kadangkala justru membuat hasilnya menjadi tidak realibel. Maka dari itulah simulator harus cukup fleksibel dengan melihat kedaan yang berlangsung.

Fungsi penting untuk mengumpulkan statistik suatu sistem (sering disebut dengan *monitor*, dibutuhkan untuk memilih representasi yang tepat untuk sistem workload)

- **Simulasi Stochastic**. Dalam simulasi stochastic (Monte Carlo), workload digambarkan oleh *distribusi probabilitas*. Permintaan untuk resources dari model simulasi dibuat sebagai sampel acak (random) dari distribusi tertentu. Oleh karena itu diperlukan "*penghasil nilai random generator yang baik*". Workload dispesifikasikan oleh distribusi yang memiliki 5 karakteristik :
    - Waktu interval suatu Job

Susun!

- Kebutuhan memori utama untuk suatu Job
- Waktu CPU yang dibutuhkan oleh suatu Job
- Nilai I/O request yang dibuat oleh job yang diberikan
- Panjang record I/O yang dimanipulasi oleh job yang diberikan

Simulator membuat job dengan cara membentuk waktu interarrival (Waktu antara dua permintaan yang berturut-turut untuk layanan sistem) dan karakteristik permintaan (demand characteristics) dari distribusi yang tepat. Pergerakan job melewati sistem tersebut direpresentasikan oleh urutan dari even yang ditandai pada bagian awal dan pada bagian akhir dari aktivitas job yang berbeda. Ketika suatu even ditandai pada awal kedatangan job, waktu kedatangan dari even selanjutnya tergantung pada proses kalkulasi karakteristik job tersebut. Perekaman kronologi simulasi tersebut dapat digunakan untuk sebagai masukan bagi even-even pengukuran selanjutnya.

Statistik yang dikumpulkan oleh simulator digunakan untuk merepresentasikan *steady state* dari sistem yang disimulasikan. Akurasi hasil didapat berdasarkan banyaknya observasi yang berhasil dikumpulkan untuk setiap pengukuran. Melengkapi observasi pada variabel yang diukur yang berulang-ulang tersebut adalah dengan menggunakan distribusi yang independen dan identik, dan dengan itu maka interval yang jelas untuk mengestimasi secara statistik variabel ini dapat ditentukan dengan menggunakan metode statistik klasik. Biar bagaimana pun, beberapa tambahan serial waktu analisis dapat saja kita gunakan.

Masalah dalam menghasilkan estimasi yang akurat dengan simulasi adalah jika sistem yang disimulasikan tersebut hanya memiliki satu poin regenerasi saja. Poin regenerasi adalah suatu status keadaan sistem dimana perilaku sistem di masa yang akan datang tidak tergantung pada perilaku yang telah lalu dan pada sistem yang dikembalikan. Setiap siklus dari periode antara pengembalian hasil yang berturut-turut untuk pembentukan hasil estimasi dapat dilihat pada satu kali observasi, yaitu observasi dengan menggunakan distribusi yang independen dan identik.

- **Simulasi Trace driven**. Dalam metode ini, simulasi workload dari sistem komputer direpresentasikan oleh *urutan permintaan layanan resources yang sudah ditentukan*. Urutan dapat dihasilkan secara artifisial atau diambil dari trace even yang diperoleh dengan cara mengukur sistem yang sedang berjalan. Simulasi ini dapat digunakan untuk mempelajari kinerja sistem yang memiliki konfigurasi yang berbeda, kemudian juga untuk menganalisis perbedaan dari strategi manajemen resources yang dibuat dan tentu untuk mengevaluasi sistem yang lebih baru.

Kelebihan metode ini adalah kemampuan mendeskripsikan workload secara mendetail. Baik secara korelasi maupun metode menginterferensi

Tanggal :
Lokasi :

Susun!

sumber yang diukur, sehingga mendapatkan hasil yang mendekati akurat. Masalah yang mungkin terjadi misalnya pada karakteristik permintaan individual yang bercampur dengan efek multitasking. Validasi simulator dihasilkan dengan membandingkan penggunaan trace pada output simulator tersebut. Validasi tersebut dapat berperan hanya dengan cara mengetes beberapa metode trace yang berbeda pada simulator dan menangkap semua aspek workload sistem tersebut. Dan kemudian simulator harus menghasilkan sesuatu yang valid bukan hanya untuk versi sistem yang diukur saja, tapi juga semua variasi sistem yang telah diinvestegasi.

3. Utilitas untuk Pengukuran

Pengukuran sistem komputer dapat dilihat dari 2 pendekatan :

- **Pendekatan Stimulus**. Sistem dianggap sebagai balack box(kotak hitam) yang mengandung nilai yang terbatas dari fungsi yang sudah diketahui. Pengukuran terdiri dari observasi respon sistem terhadap suatu benchmark atau dengan simulator khusus. Respon sistem diterima sebagai pekerjaan yang selesai.Pendekatan ini biasanya berupa evaluasi dari perbandingan, cepat tapi tidak cukup teliti.
- **Pendekatan Analitik**. Pengukuran melibatkan perilaku internal sistem tersebut. Biasanya pengukuran ini dilakukan untuk :
    - Memastikan suatu operasi itu benar.
    - Mengisolasi sumber dari masalah saat itu maupun yang potensial.
    - Mengembangkan pemahaman terhadap sistem dan lingkungannya.

Untuk pengukuran analitik, poin observasi khusus harus disediakan dalam sistem tersebut.Evaluasi kinerja memerlukan kombinasi dari 2 pendekatan di atas.

Masalah Pengukuran adalah untuk menetapkan :

- Apakah informasi tersebut merupakan bagian yang berhubungan dengan tujuan dari pengukuran khusus,
- Di mana informasi dapat ditemukan,
- Bagaimana informasi dapat di ekstrak dan direkam.

Perilaku sistem diobservasi dengan melakukan perubahan pada status sistem. Biar bagaimana pun analisis kinerja sistem itu berhubungan dengan model sistem. Model yang dibuat harus dapat diukur, mendeskripsikan detail sistem dan semua pertanyaan yang mungkin tentang sistem tersebut. Model dikembangkan sebagai basis dari wawasan yang dibangun oleh pengukuran aktual.

Tanggal :
Lokasi :

Susun!

Perubahan dalam sistem yang statusnya ditandai dilakukan pada bagian awal atau pada bagian akhir dari masa aktif (atau tidak aktifnya) komponen sistem (Perangkat keras, perangkat lunak atau proses). Setelah komponen dapat aktif lagi secara simultan, perubahan status sistem dilakukan pada level aktivitas sistem tersebut.

Aktivitas tersebut digunakan pada koneksi dengan komponen tunggal dan diletakkan pada "level aktivitas" dari seluruh sistem. Perubahan dalam sistem tersebut disebut *even*.

Status sistem tersebut dapat dideskripsikan oleh vektor yang disusun oleh elemen biner yang merepresentasikan keadaan status (0 atau 1) dari keadaan individual. Dalam aktifitas *ak* dapat direpresentasikan oleh fungsi logika yang memiliki nilai 1 pada subset *Xk* dari segala kemungkinan keadaan himpunan *X*, $X_k \subset X$,

Ketika aktifitas *ak* dimulai, sistem akan berubah dari keadaan $x_{new} \in X_k$ ke keadaan baru $x_{new} \in X_k$

Transisi tersebut kemudian disebut dengan even inisiasi *ek*.

Ketika aktifitas *ak* berhenti, sistem dari $x_{old} \in X_k$ ke keadaan $x_{new} \notin X_k$

Transisi tersebut kemudian disebut even berhenti $\overline{e}_k$.

Pengukuran dapat terbagi dalam 4 kategori termasuk tipe informasi yang direkam ketika aktifitas pengukuran dilakukan. Berikan *t0* dan *t* pada waktu awal dan akhir percobaan pengukuran.

1. Trace. Aktivitas *ak* dideskripsikan oleh urutan pasangan (t_k^i, T_k^i) , ketika t_k^i dalam waktu *i* kejadian pada aktifitas tersebut dan T_k^i berhubungan dengan durasi dari aktifitas tersebut. Informasi ini didapatkan dengan cara trace even, dimana urutan record dari semua peristiwa mulai dari even inisiasi sampai even berhenti, *ek* dan *ek*. Selama interval pengukuran [*t0*, *t*].

2. Aktifitas Relatif. Aktifitas relatif *rk* adalah rasio dari total waktu aktifitas *ak* dan total waktu yang telah dilalui :

$$r_k = \frac{1}{t - t_0} \int_{t_0}^{t} a_k(\tau)\, d\tau$$

Tanggal :
Lokasi :

Susun!

$t \geq t_0$, where $a_k = 1$ if $x(\tau) \in X_k; a_k = 0$ otherwise

Aktifitas relatif memanfaatkan frekuensi yang digunakan untuk melakukan pengukuran kinerja (CPU utilization, channel utilization).

3. Frekuensi Even. Frekuensi yang membawa keadaan-keadaan yang khusus diukur dengan menghitung even yang merepresentasikan transisi dalam keadaan-keadaan tersebut. Frekuensi dari kejadian aktifitas *ak , ck* diukur dengan menghitung even inisialisasi *ek :*

$$c_k = \frac{1}{t - t_o} \sum_{t_n} e_k(\tau),$$

$t \geq t_n \geq t_0$, where $e_k = 1$ for $\tau = t_n$; $e_k 0$ otherwise

dan *t*n adalah waktu peristiwa dari even *ek*.

4. Distribusi Interval Aktivitas. Berikan $f_k^n(T)$ yang akan didistribusikan sebagai waktu durasi T dari aktifitas *ak* pada waktu *n* sampai berhenti dari suatu aktifitas. Maka

$$f_k^n(T) = \frac{1}{n} \sum_{i=1}^{n} g(T, T_k^i),$$

where $g(T,d) = 1$ for $T = d$; $g(T,d) = 0$ otherwise

4. Measurabilitas

Kompleksnya suatu sistem membuat pengukuran menjadi sulit. Hardware dan software tidak didesain untuk dapat dimonitor. Beberapa tipe dari pengukuran tidak bisa dilakukan tanpa melakukan perubahan pada sistem. Software dan hardware dari sistem individual harus dapat ditentukan apa yang akan diukur dan bagaimana mengukurnya. Masalah privacy user termasuk yang harus dipertimbangkan pula.

Measurabilitas dari sistem komputer dapat didefinisikan dalam :

- Himpunan Total informasi yang dapat diakses oleh monitor kinerja, dan
- Harga dari pengukuran

Tanggal :
Lokasi :

Susun!

Pengukuran dibutuhkan oleh :

- User, untuk konfigurasi optimal, perencanaan pertumbuhan isstem dan sistem dalam pengembangan.
- Sistem itu sendiri, mengendalikan kinerja sistem dalam lingkungan yang selalu berubah secara dinamis.

Peralatan untuk memonitoring sistem dapat saja disalah gunakan. Masalahnya adalah berapa banyak bagian yang tidak dapat diukur karena alasan pembatasan privasi ternyata justru adalah bagian yang sangat dibutuhkan untuk perbaikan sistem dan untuk perencanaan sistem baru.

Karakteristik utama Alat bantu pengukuran :

- Inteferensi. Jika tool memanfaatkan resources sistem, operasi akan diinterferensi oleh sistem itu sendiri dan mempengaruhi nilai kuantitas yang diukur. Ini tentu membuat kinerja menjadi terdegradasi dan hasil pengukuran menjadi tidak akurat.
- Akurasi. Terdapat beberapa kesalahan yang dapat menyebabkan kuantitas nilai pengukuran menjadi berbeda dengan nilai yang sesungguhnya. Tool harusnya presisi dan beresolusi tinggi dalam pengukuran. Perbedaan sumber atau terjadinya error adalah suatu malfungsi.
- Resolusi. Frekuensi maksimum dari even yang dapat dideteksi atau terekam dengan tepat.
- Scope (lingkup). Sistem yang fleksibel.
- Kapabilitas yang dapat menurun
- Komputabilitas
- Harga yang memadai
- Mudah dalam instalasi
- Mudah dalam penggunaan

Klasifikasi lainnya :

- Kendali secara internal
- Kendali secara eksternal

Tipe tool :

- Alat bantu perangkat keras (Fixed Hardware tools)
- Program bantu untuk pengkabelan (Wired program tools)
- Program bantu yang tersimpan (Stored program tools)
- Alat bantu Perangkat lunak dan firmware

Tanggal :
Lokasi :

ALAT UKUR SECARA FISIK

Susun!

1. Defenisi

Instrumentasi adalah kumpulan utilitas (alat Bantu) yang digunakan untuk mendeteksi even-even yang terjadi dan mengkuantisasi hasil-hasil yang diobservasi.

2. Karakteristik Alat Ukur

1. Perkembangan alat-alat untuk mengukur sistem komputer dilihat dari jenis beban kerja yang diberikan dikarakteristikkan berdasarkan implementasi penggunaannya, yaitu :

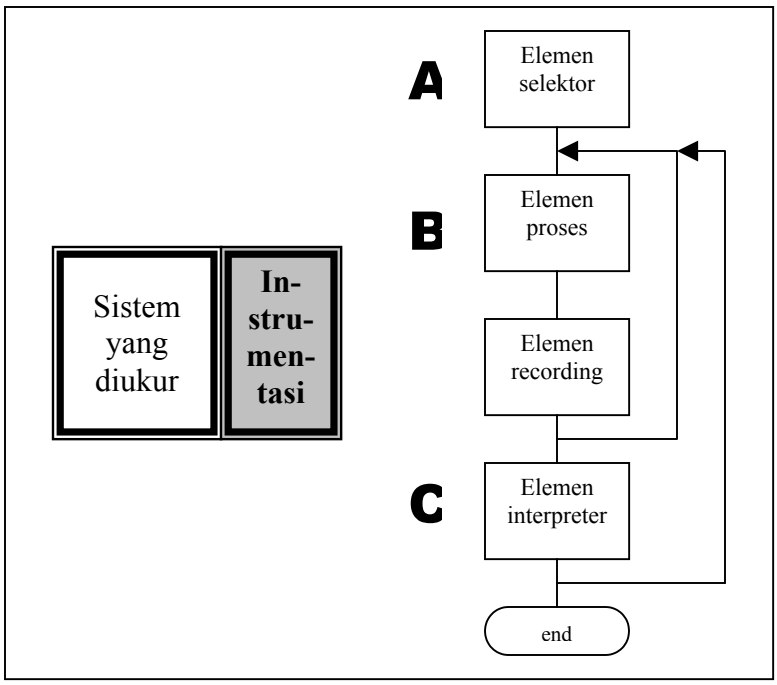
- Monitor software
  Disusun dari program-program atau kumpulan instruksi yang dapat mendeteksi keadaan suatu sistem dan even yang terjadi dalam suatu sistem, sering pula disebut *software probes*.
- Monitor hardware
  Disusun dari komponen-komponen elektronika yang dihubungkan dengan titik-titik tertentu pada suatu sistem komputer, untuk dapat mendeteksi sinyal (berupa tingkatan voltase atau pulsa) yang merupakan karakteristik fenomena yang sedang diamati.

2. Dilihat dari penggunaan energi oleh suatu alat ukur, instumen pengukuran komputer sudah pasti akan membagi dan mengurangi energi sistem yang sedang diukur, untuk itu instrumen pengukuran haruslah dapat menggunakan energi serendah mungkin sehingga energi overhead pada alat ukur tersebut tidak mengubah hasil pengukuran.

3. Sebuah instrumen pengukuran harus dapat mengungkapkan tingkat akurasinya dengan juga memperhitungkan error (kesalahan) yang dapat mempengaruhi nilai perolehan bagi kuantitas yang diukur.

Tanggal :
Lokasi :

Susun!



Skema konseptual alat Bantu pengukuran

Keterangan :

- Hubungan antara alat monitor dengan sistem yang diukur menggunakan interface instrumentasi. Interface yang meliputi :
  - Kode-kode yang disisipkan pada sistem operasi
  - Sistem timer (pewaktuan)
  - Pin-pin pada papan elektronik
- Filtering element (elemen selector) (A) : untuk memilih observasi secara spesifik dari aktivitas yang diukur.
- Processing element (B) : memproses tes-tes pada state/keadaan komponen sistem yang akan diukur, lalu hasilnya akan dicatat dalam media penyimpanan
- Interpreting element (C) : menganalisis hasil proses dan memberikan kesimpulan.
- Kadangkala fase interpreter dilakukan pada saat yang bersamaan dengan deteksi even dan pengumpulan data, inilah yang disebut pengukuran waktu nyata (*Real Time Measurement*), yang menyediakan kemungkinan kendali kinerja sistem secara dinamis.

Tanggal :
Lokasi :

3. Monitoring Kinerja

Susun!

Apa itu Monitoring ?

Monitor adalah suatu alat yang digunakan untuk mengamati aktivitas pada suatu sistem, mengumpulkan statistik kinerja, menganalisa data dan menyampaikan hasilnya.

Suatu monitor akan mengamati aktivitas di dalam suatu sistem :

- Mengumpulkan statistik kinerja
- mencatat Kejadian suatu even

Siapa yang menggunakan monitor?

- Para programmer yang berusaha untuk mengoptimalkan program.
- Para manajer sistem yang menemukan bottlenecks, mengukur pemanfaataan sumber daya.
- Para manajer sistem yang melakukan tuning sistem, melakukan penyesuaian parameter sistem.
- Analis Sistem yang berusaha untuk mengkarakteristikkan beban kerja, melakukan perencanaan kapasitas, menciptakan test WL.
- Analis sistem yang berusaha menemukan parameter untuk suatu model, validasi model atau mengembangkan masukan untuk suatu model.

Monitoring adalah langkah dan kunci pertama dalam pengukuran kinerja.

Terminologi Monitor

- even - Suatu perubahan dalam status sistem
- trace - Suatu catatan log dari even
- Overhead - Berapa banyak kegiatan monitoring membebani.
- Domain - Satuan aktivitas yang dapat yang diamati oleh suatu monitor.
- Input rate - Frekuensi terjadinya even yang maksimum yang dapat dengan tepat diamati.
- Resolusi - Kemampuan untuk memilah-milah even
- input width - Jumlah bit informasi
- Perturbence - Monitoring dapat menyebabkan dasar suatu sistem basis bertindak berbeda; berapa banyak perubahan sistem yang terjadi dalam suatu monitoring?
- Pemuluran waktu (time dilation) atau Faktor perluasan (dilation factor) - serupa dengan overhead, menyatakan berapa kali lebih lambat program berjalan dalam kaitan dengannya dengan aktivitas monitoring.

Tanggal :
Lokasi :

Susun!

Klasifikasi Monitor

Implementasi Monitor :

- Perangkat keras, firmware atau perangkat lunak

Metode sampling :

- Event-Driven monitor - yang aktif hanya ketika suatu peristiwa terjadi.
- Monitor sampling - yang aktif pada suatu jadwal berkala

Metode Display :

- Monitor on-line - menampilkan status sistem secara terus-menerus.
- Batch (off-line) monitor - mengumpulkan data yang kemudian akan diproses oleh suatu alat analisa.

4. Perangkat Lunak Monitoring

Even = transisi dari suatu keadaan yang dapat mengindikasikan permulaan atau akhir sebuah periode aktivitas (atau ketidakaktifan) dari setiap komponen hardware maupun software pada suatu sistem komputer.

Konsep Perangkat Lunak monitoring

Sistem akan tetap dalam keadaan tertentu untuk beberapa waktu, agar dapat mendeteksi suatu transisi atau perubahan kejadian even pada suatu sistem tersebut dapat dilakukan dengan memberikan suatu bentuk kondisi. Transisi dan perubahahn kondisi ini akan terlihat pada lokasi memori. Gagasan monitoring software ini adalah menciptakan sebuah program yang dapat menangkap isi dari lokasi memori tersebut.

Oleh karena itulah. penggunaan software monitor ini memerlukan pemasukan kode tambahan dalam sistem yang terukur, yaitu dengan 3 cara :

1. Tambahan sebuah program.
   Dengan menambah program lain di luar program terukur akan dapat mempertahankan integritas program terukur, selain itu pula lebih mudah menggunakannya pada saat dibutuhkan dan sekaligus memindahkannya manakala tidak dibutuhkan lagi.. Metode ini cukup memadai untuk mendeteksi aktivitas keseluruhan sistem atau mengukur aktivitas program tunggal.
2. Modifikasi software terukur.
   Menggunakan software penyelidik, yaitu sekelompok instruksi yang dimasukkan pada nilai kritis pada program yang diamati yang harus dapat mendeteksi kedatangan aliran kontrol pada titik kritis dimana instruksi tersebut diletakkan. Metode ini dapat mengetahui jumlah waktu running logika program yang dieksekusi, dan pula isi dari area

Tanggal :
Lokasi :

memori (table, struktur data, operasi tunggal) ketika eksekusi sedang berjalan.

3. Modifikasi sistem operasi atau program sistem yang ada.
   Metode ini digunakan ketika variabel data sistem yang diukur tidak secara langsung tersedia (tidak berada dalam tabel sistem). Metode ini dapat digunakan untuk menghitung waktu respon yang dibutuhkan untuk memproses suatu transaksi dalam lingkungan interaktif. Instruksi dapat diletakkan ke dalam program yang mengatur respon antrian agar terdeteksi waktu yang dibutuhkan bagi suatu hasil transaksi mencapai output antrian.

Susun!

Karakteristik instrumen Perangkat lunak monitor

1. Harus dapat menambahkan data kuantitatif dan deskriptif dari sistem. Termasuk pula informasi mengenai identitas program yang aktif, file-file yang diakses dan sebagainya.
2. Harus mampu memodifikasi sistem operasi sekecil mungkin.
3. Harus menggunakan deteksi even(kejadian) dan teknik pengumpulan data yang tidak mengubah karakteristik beban kerja dan kinerja sistem terukur.
4. Harus memiliki kebutuhan penggunaan memori sekecil mungkin.
5. Syarat Proses pengambilan data :
   - Dapat mendeskripsikan data yang mampu mengeliminasi kebutuhan yang berkolerasi pada waktu kemudian, contoh : mengetahui nama file pada disk yang secara teratur diakses pada waktu tertentu, atau menentukan nama program yang paling sering dipakai.
   - Pengetahuan yang didapatkan dapat dieksploitasi dalam lingkungan memori virtual dalam rangka mengevaluasi ketepatan restrukturisasi program agar dapat diperbaiki kinerjanya.
   - Dilakukan dengan cara membaca muatan tabel memori tertentu. Jumlah data yang dikumpulkan tergantung pada struktur sistem operasi, dengan tidak hanya mengikuti jumlah pada fitur intrumen tersebut.
   - Interferensi (overhead) yang disebabkan oleh deteksi even dan pengumpulan data maupun pengambilan sampling harus dapat terukur dan ditoleransi.

     Cara mengontrol interferensi adalah dengan menggunakan sampler, yaitu dengan cara :
     1. Seleksi metode pengukuran yang ditampilkan.
     2. Pemilihan variabel interval sampling yang tepat.

     Langkah-langkah pengontrolan interferensi :

Tanggal :
Lokasi :

Susun!

1. Melakukan pengumpulan data sebanyak mungkin.
2. Menambahkan jumlah beban tertentu ke dalam sistem atau memperlambat eksekusi program.
3. Menganalisis hasil awal dan melakukan uji spesifik secara lebih detail dari hasil awal tersebut sampai ditemui sampel yang representatif yaitu sampel dengan error dibawah 1 % dari 99 % kasus.

Contoh Perangkat Lunak Monitoring

Perangkat lunak monitor vs perangkat keras monitor :

- input rate rendah
- resolusi rendah
- overhead tinggi

Yang harus diperhatikan dalam perangkat lunak monitor :

- Mekanisme pengaktifan (traps-menangkap, trace-melacak, timer-pengaturan waktu dan perangkat lunak).
- buffer - ukuran dan jumlah buffer, dan bagaimana cara
- overflow ditangani. Kadang-Kadang yang dilaksanakan di dalam data terkompresi atau analisis secara paralel - sekarang atau kemudian?
- Prioritas monitoring proses
- Monitoring even abnormal

Perangkat lunak Monitor UNIX yang umum

- iostat - atribut Subsistem I/O

```
[foobar-93] iostat 2
     tty     fd0      rz0      rz1      rz2      cpu
 tin tout bps tps  bps tps  bps tps  bps tps  us ni sy id
   0   10   0   0    6   1    8   1    7   1   1  0  1 98
   0   29   0   0    0   0    0   0    0   0   0  0  1 99
   0   29   0   0    0   0    0   0    0   0   0  0  0100
   0   29   0   0    0   0    0   0    0   0   0  0  1 99
   0   29   0   0    0   0   16   1   32   2   0  0  1 99
   0   29   0   0    0   0    0   0    0   0   0  0  0100
   0   29   0   0    0   0    0   0    0   0   0  0  0 99
   0   29   0   0    0   0    0   0    0   0   0  0  0100
```

Tanggal :
Lokasi :

Susun!

- netstat - atribut sistem jaringan

```
[foobar-94] netstat 2
input   (tu1)    output          input   (Total)    output
 packets  errs  packets  errs colls   packets  errs  packets  errs colls
3292529    0 3192032    2    0  3690563    0 3598733 8669    0
   5    0     5    0    0      5    0     5    0    0
   5    0     2    0    0      7    0     4    0    0
 692    0  1215    0    0    694    0  1217    0    0
2424    0  4793    0    0   2424    0  4793    0    0
   4    0     4    0    0      8    0     8    0    0
   1    0     1    0    0      1    0     1    0    0
2432    0  4741    0    0   2436    0  4745    0    0
 721    0  1292    0    0    723    0  1294    0    0
  99    0    15    0    0     99    0    15    0    0
 180    0   111    0    0    180    0   111    0    0
```

- vmstat - virtual memori sistem

```
[foobar-96] vmstat 2
Virtual Memory Statistics: (pagesize = 8192)
 procs    memory       pages                          intr     cpu
 r  w  u  act  free wire fault cow zero react pin pout  in  sy  cs  us  sy  id
 3 88 20  25K  873 5040   5M  1M   1M  12K   1M  278  18 114 273   1   1  98
 3 88 20  25K  858 5040   87   4   78    0    8    0   8  39 254   0   1  98
 3 88 20  25K  858 5040   54   0   54    0    0    0   2  25 242   0   1  99
 3 89 20  25K  824 5054  200  29  131    0   29    0  90  5K 426   1   5  93
 3 88 20  25K  849 5050  162  24   91    0   36    0  3K  3K  3K   2  41  57
 3 88 21  25K  705 5184  120  18   70    0   15    0 204  2K 832   3  17  81
 3 88 20  25K  660 5251   68   0   55    0    0    0 101 952 539   1   7  91
```

Tanggal :
Lokasi :

Susun!

- monitor - tool monitoring sistem secara umum (I/O, dll)

```
foobar.cs.co  Wed Jan 22 17:46:31 1997    0.05  0.03  0.02      7 users

Mem:  act inact wired  free  Forks: fork vfork    Char:  in    out
  186864 18256 42728  4088       0.00  0.00         0.0   75.4

Paging: re  pin  pout flts  cow   zf  hit%        Disk: kbps  tps queue
      0   0    0    0    0    0   0        cam0    0   0    0
                                          cam1    0   0    0
Swap: Reserved  Free  Cache:  Namei  Buffer          cam2    0   0    0
       4%   96%          92%    99%           cam3    0   0    0
                                          cam6    0   0    0
CPU: user nice  sys idle wait  swtch   intr  scall      cam7    0   0    0
#0    2   0   12  87   0    255    16  3181       fd0     0   0    0
#1    0   0   0  100   0                     rz0     0   0    0
                                          rz1     0   0    0
Net:    ipkts  ierrs  opkts  oerrs collis            rz2     0   0    0
tu0      0.0   0.0    0.0    0.0   0.0             rz3     0   0    0
tu1      7.5   0.0    9.0    0.0   0.0             rz8     0   0    0
sl0      0.0   0.0    0.0    0.0   0.0             rz12    0   0    0
lo0      0.0   0.0    0.0    0.0   0.0
```

- paket - filter untuk memilih paket untuk direkam.
- top - alat monitoring biasanya digunakan
- trace - menjejaki sistem call yang dibuat oleh suatu program.

```
[foobar-97] trace echo foo
echo
Tracing process /proc/315
obreak (0x14000ea60) = 0
ioctl (1, 0x40067408<Out,TIOCGETP,6>, 11ffff3f8) = 0
write (1, 0x3ffc0093990, 4) = foo
4
close (0) = 0
close (1) = 0
close (2) = 0
sigprocmask (1, 0xffffffffffffff137, 0x0) = 0
exit (0) = (0)
EXIT [status=0]
```

Tanggal :
Lokasi :

Susun!

- tcpdump - merekam even pada jaringan, penggunaan

```
rtsg.1023 > csam.login: S 768512:768512(0) win 4096 <mss 1024>
csam.login > rtsg.1023: S 947648:947648(0) ack 768513
                        win 4096 <mss 1024>
rtsg.1023 > csam.login: . ack 1 win 4096
rtsg.1023 > csam.login: P 1:2(1) ack 1 win 4096
csam.login > rtsg.1023: . ack 2 win 4096
rtsg.1023 > csam.login: P 2:21(19) ack 1 win 4096
csam.login > rtsg.1023: P 1:2(1) ack 21 win 4077
csam.login > rtsg.1023: P 2:3(1) ack 21 win 4077 urg 1
csam.login > rtsg.1023: P 3:4(1) ack 21 win 4077 urg 1
```

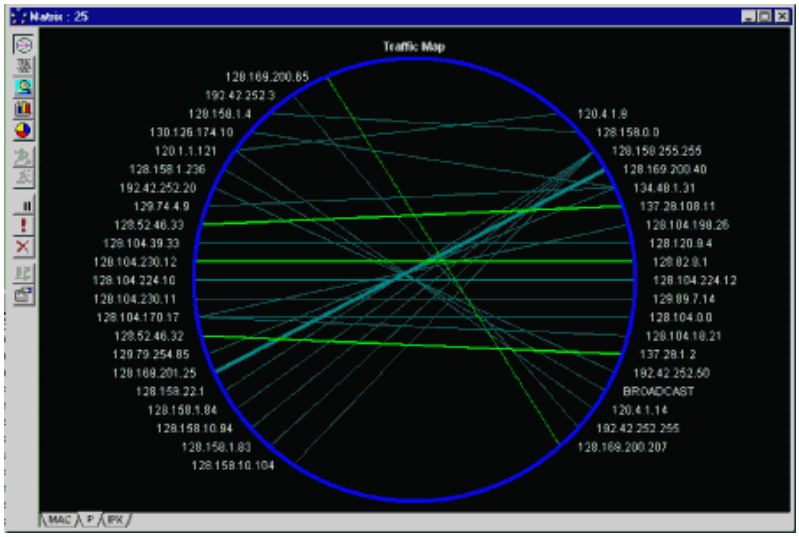
Perangkat lunak Monitor WINDOWS yang umum

Fungsi :

1. Untuk memantau network :
    - Penggunaan network
    - Adanya masalah, error
2. Mengelola network :
    - Mengganti routing

Jenis :

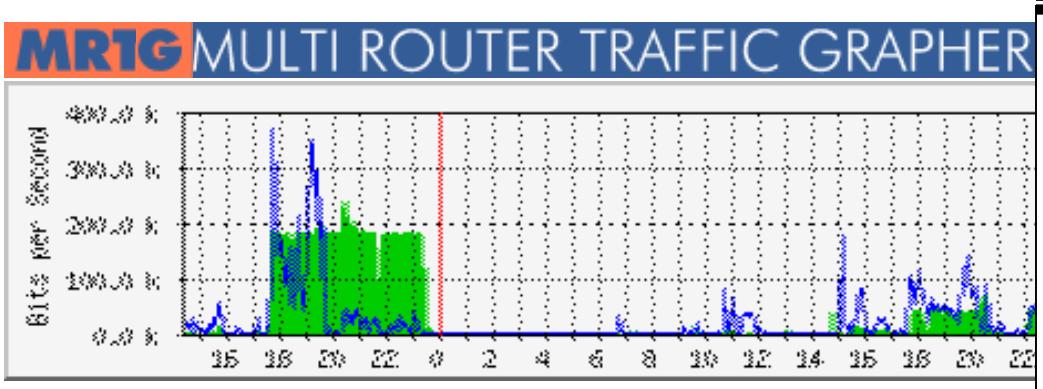
- Komersial : SnifferPro



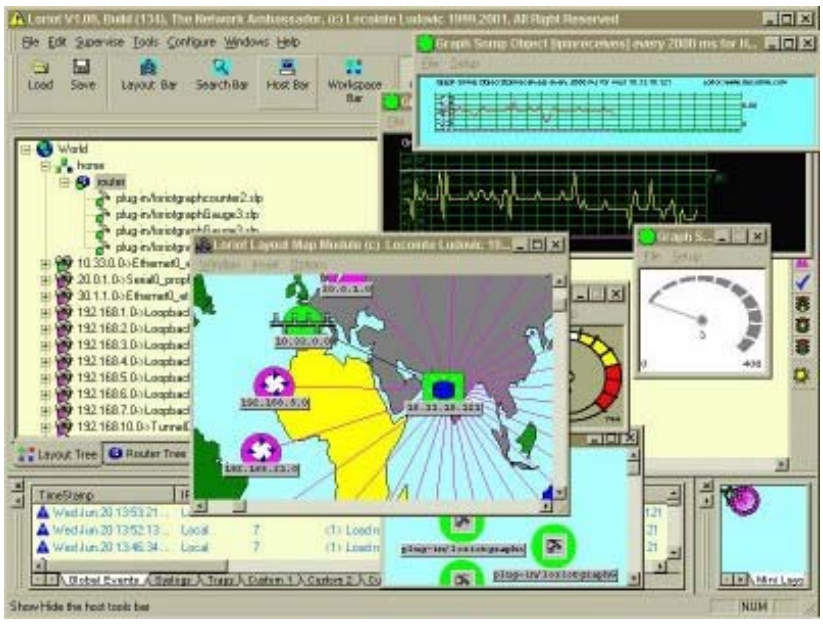
Tanggal :
Lokasi :

Susun!

- Opersource : SNMP &MRTG



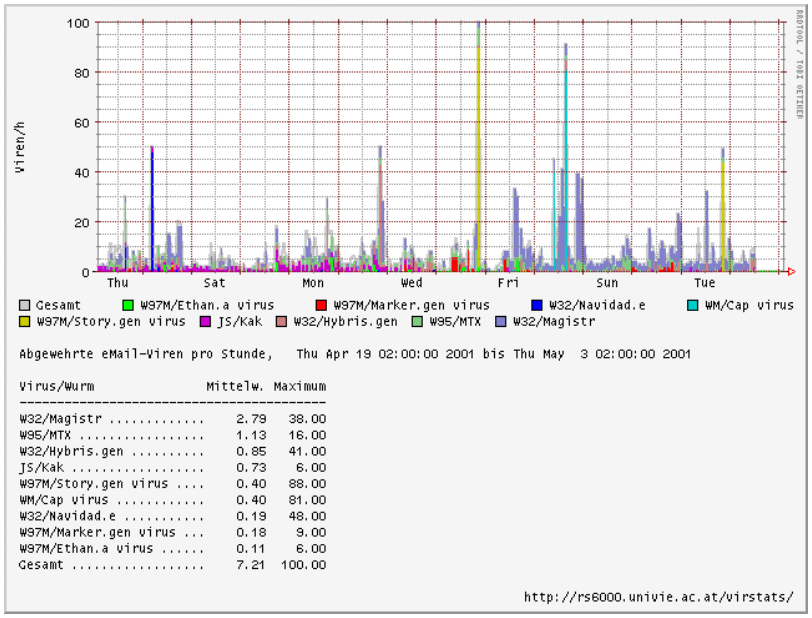
- freeware : LORIOT (http://www.llecointe.com/)



Tanggal :
Lokasi :

Susun!

- RRD TOOL



Masalah Pada Monitoring Jaringan :

- Terjadi karena salah konfigurasi
  - Penggunaan community “public” di SNMP
- Confidentiality / privacy
  - Melihat isi traffic
  - Melihat pola traffic
  - Melihat routing table
- Authentication, integrity
  - Routing diubah oleh orang yang tidak berhak via SNMP
- Availability
  - DoS attack

5. Perangkat Keras Monitoring

- Masukan (Input) tinggi, mahal, sukar untuk dioperasikan.
- Terdiri dari:
  1. Probe - untuk mengamati sinyal
  2. Counter, elemen pembanding (comparator) dan logika - untuk memproses informasi dari probe
- Beberapa kesulitan dalam monitoring perangkat keras :
  1. single-chip komputer :
     - Tidak (ada) tempat untuk meletakkan probe.
     - Kebanyakan komputer mempunyai beberapa perangkat keras yang berbeda konfigurasinya.

Tanggal :
Lokasi :

Susun!

2. Monitoring built in dalam sistem :
    - Monitoring jaringan menggunakan LAN analizer, tidak semua perangkat hardware memilikinya.

Contoh Perangkat Keras Monitoring pada DEC Alpha

- Sistem mendukung dua register kinerja
- Masing-Masing register dapat mempunyai satu dari banyak data sumber :
    - Tidak bisa membaca hasil perhitungan secara langsung
    - Register memiliki asosiasi granularity, misalnya terdapat 25665536 even, ketika N peristiwa ditemukan, interrupt dipanggil.
- Dapat mengambil sampel status sistem dari interupt tersebut.
    - Program akan meng-counter interupsi program bagi suatu status sistem tertentu (baik user maupun kernel)
- Yang dapat di monitor oleh perangkat keras 21064A :

First Register	Second Register	Synthesis
Issues	Dual Issues	Dual Issue Rate
	Icache Misses	Insn Miss Rate
Loads	Data Cache Misses	Data Miss Rate
Branches	Branch Mispredicts	Branch Miss rate
pipe_dry	Integer ops	
pipe_frozen	Floating Point ops	
Cycles	Stores	
Palcode cycles	B-Cache Miss	
Bcache Victem		

6. Memonitor Program Komputer

Monitor Pelaksanaan Program

Alasan untuk melakukan monitoring program :

- Tracing - untuk menemukan alur eksekusi (execution path) dari suatu program.
- Timing - untuk menemukan waktu masuk yang tepat bagi pengukuran suatu program.
- Tuning - untuk menemukan sumberdaya maksimal yang dapat digunakan.
- Assertion Checking - untuk memverifikasi hubungan antara
- variabel yang ada dalam suatu program.
- Analisis cakupan program - untuk menentukan ketercukupan dari suatu test run program.

Tidak semua penggunaan suatu program berhubungan dengan tuning kinerja.

Tanggal :
Lokasi :

Susun!

Yang harus diperhatikan dalam mendesain suatu proses monitor eksekusi program

- Unit Pengukuran :
  Perlukah laporan atas modul? Subroutines? atau Statemen?

- Teknik Pengukuran :
  Tracing bisa dilakukan menggunakan hook, ptrace() atau instrumentasi Sampling menggunakan interrupt secara periodik.

- Mekanisme Instrumentasi :
  Kapan sebaiknya instrumentasi ditambahkan?
  Menyusun waktu pelaksanaannya ? link time? load time? run time?

- Bagaimana cara laporan tersusun :
  hirarkis vs flat

Memonitor Perangkat Lunak melalui Kode Instrumentasi

Perangkat Lunak monitor trap-based sederhana :
- UNIX Ptrace() Call, yang digunakan oleh debuggers
- Sangat lambat dalam konteks switch sistem (misalnya tango)

Perangkat lunak Instrumentasi yang dapat menghindari perangkap Kesalahan :
- Pixie - instrumentasi biner, memancarkan " jejak (hasil trace)" eksekusi program.
- epoxie, moxie, shade --- tool utama, kebanyakan
- menggunakan instrumentasi biner.
- Mint, aint, proteous, tango lite - penafsiran paralel suatu program untuk menghasilkan even. Pemilihan waktu even sangat ditentukan oleh sistem simulasi.
- ATOM, SimOS - instrumentasi biner lebih fleksibel.

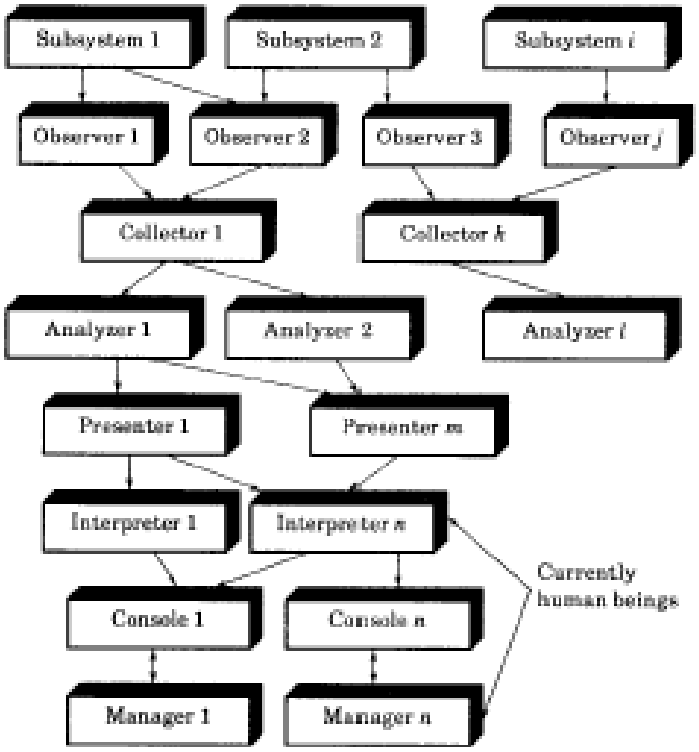
7. Monitor Sistem Terdistribusi

Fungsi Monitor terdistribusi terbagi dalam lapisan :
- Pengamatan: Mengumpulkan data mentah pada komponen individual.
- Koleksi: Mengumpulkan data dari berbagai observer.
- Analisis: rekaman Ikhtisar statistik
- Presentasi: Alat penghubung dengan pemakai (user)
- seperti menghasilkan laporan, display (tampilan), alarm.

Tanggal :
Lokasi :

- Penafsiran: entitas kecerdasan manusia
- Console: manajemen antar muka sistem kendali sebagai Pembuat keputusan untuk menetapkan atau mengubah parameter sistem.

Komponen Monitor Sistem terdistribusi



Susun!

Tanggal :
Lokasi :

Susun!

Contoh Utilitas Pengukuran :

Tipe	Contoh tool
Icmp-based	ping, Nikhef ping, fping, gnuplotping, Imeter/Lachesis
per-hop analysis	traceroute, Nikhef traceroute, traceroute server, pathchar, OTTool
throughput	treno, bing, bprobe, cprobe
bulk throughput	netperf, ttcp, nettest, netspec
web avalibility	wwping
packet collection	argus, tcpdump, libpcap, pcapture, Packetman (free) etherfind, iptrace, netsnoop, snoop (bundled software collecion) Century LAN Analyzer, EtherPeek, LANSleuth, Monet, netMinder, Observer (commercial) Cellblaster, HP Internet Advisor, Sniffer, W&G (hardware) fs2flow, Coral/OC3Mon, NeTraMet (flow collectors) tcptrace, tracelook, xplot (analysisi/plotting tools)
flow statistics	NetFlow interface, cflowd, Oc3mon, mrtg
mbone	mtrace, mview
route behaviour	NPD, NetNow, IPMA