BAB II

LANDASAN TEORI

2.1 Penjadwalan

Penjadwalan dapat didefinisikan sebagai pengaturan pengalokasian sumber daya dalam jangka waktu tertentu untuk melakukan serangkaian tugas yang telah dirancangkan [Bak74].

Penjadwalan adalah aturan atau proses pengorganisasian, pemilihan dan penentuan waktu, penggunaan tempat atau sumber – sumber untuk mengerjakan semua aktifitas yang diperlukan yang memenuhi kendala aktiitas dan sumber daya [Mor93].

Penjadwalan proses belajar mengajar merupakan pengaturan perencanaan belajar mengajar yang meliputi mata pelajaran, guru, waktu dan tempat pada sekolah.

Penjadwalan laboratrium adalah pengaturan aktifitas kegiatan yang berhubungan dengan laboratrium tersebut.

2.2 Konsep Dasar Sistem

Untuk memahami konsep dasar sistem, berikut disajikan pengertian sistem menurut beberapa sumber diantaranya sebagai berikut :

Menurut [Ams01] :

Himpunan suatu benda nyata atau abstrak (*a set of thing*) yang terdiri dari bagian – bagian / komponen – komponen yang saling berkaitan, berhubungan, berketergantungan dan saling mendukung yang secara

keseluruhan bersatu dalam satu kesatuan (*unity*) untuk mencapai tujuan secara efesien dan efektif.

Menurut [Jog97] :

Sistem adalah kumpulan dari elemen – elemen yang berinteraksi untuk mencapai suatu tujuan.

Menurut [Sus00] :

Sistem adalah kumpulan atau group dari bagian atau komponen apapun baik fisik dan non fisik yang saling berhubungan satu sama lain dan bekerja sama secara harmois untuk mencapai satu tujuan tertentu.

2.2.1 Definisi Sistem

Dari pengertian sistem menurut beberapa sumber diatas maka dapat disimpulkan bahwa sistem merupakan kumpulan dari bagian – bagian atau komponen – komponen subsistem atau bagian yang saling berinteraksi bekerjasama untuk membentuk satu kesatuan dalam menjalankan fungsi tertentu yang mempengaruhi proses dari setiap sistem susbsistem atau bagian sistem secara keseluruhan untuk mencapai tujuan tertentu.

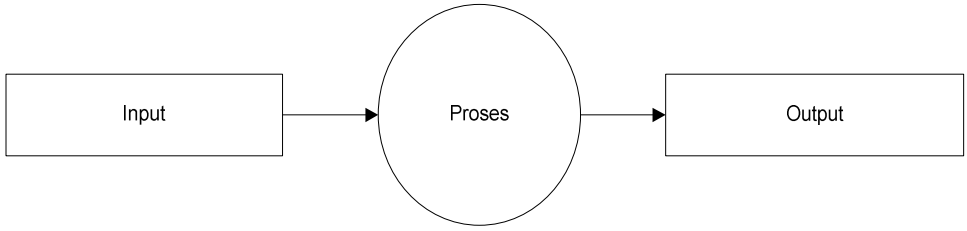
Dalam mendefinisikan pengertian dari sistem ada dua kelompok pendekatan yaitu menekankan pada prosedurnya dan menekankan pada elemennya. Pendekatam sistem yang lebih menekankan pada prosedurnya mendefinisikan sistem adalah suatu jaringan kerja dari prosedur – prosedur yang saling berhubungan berkumpul bersama – sama untuk melakukan kegiatan atau untuk menyelesaikan suatu masalah [Jog01].

Sistem itu adalah suatu kumpulan atau variabel yang terorganisasi, saling interaksi, saling bergantung satu sama lain dan terpadu.

[Ric00] *management by sistem* mendefinisikan pendekatan sistem merupakan jaringan kerja dari prosedur adalah suatu operasi tulis menulis, biasanya melibatkan beberapa bagian didalam satu atau lebih departemen yang diterapkan untuk menjamin penanganan yang seragam ditransaksi bisnis yang berlaku.

Bentuk umum dari suatu sistem terdiri atas masukan (*input*), proses dan keluaran (*output*), dalam bentuk umum sistem ini bisa melakukan satu atau lebih masukan yang akan diproses dan menghasilkan keluaran sesuai dengan rencana yang telah direncanakan sebelumnya.

Bentuk umum sistem.



Gambar 2.1 *Bentuk umum sistem*

Sumber : [Jog97]

2.2.2 Definisi Informasi

Informasi sangat dibutuhkan agar dapat mengetahui keakuratan data yang dihasilkan. Informasi ibarat data yang mengalir didalam tubuh suatu organisasi, informasi ini sangat penting dalm pengambilan keputusan didalam suatu organisasi.

Dibawah ini terdapat pengertian informasi menurut beberapa sumber yang didapat diantaranya sebagai berikut :

Menurut [Ams01] :

Informasi adalah data yang sudah diolah, dibentuk, atau dimanipulasi sesuai dengan keperluan tertentu.

Menurut [Jog97] :

Informasi adalah hasil pengolahan data, akan tetapi tidak semua hasil dari pengolahan data tersebut bisa menjadi informasi.

Menurut [Sus00] :

Informasi merupakan hasil pengolahan data yang memberikan arti manfaat.

Informasi adalah data yang diolah menjadi suatu bentuk yang lebih berguna dan lebih berarti bagi yang menerimanya. Data adalah kenyataan yang menggambarkan suatu kejadiaan pada saat tertentu. Kualitas dari suatu informasi tergantung dari keakuratan informasi, ketepatan waktu, dan kerelevanan.

Informasi merupakaan sekumpulan data mulanya sebagai kunci atas koordinasi dari semua keputusan yang diambil. Informasi itu diambil dari semua klasifikasi yang dibentuk dalam kelompok tindakan, yang berupa alat komunikasi untuk memperoleh data – data yang diinginkan. Informasi pada tiap – tiap kelompok mempunyai fungsi yang berbeda pada masing – masing bagian. Fungsi dalam informasi adalah sebagai alat untuk mengadakan hubungan antar entitas yang satu dengan entitas yang lainnya, baik dalam maupun luar dari kelompok [Wil81].

Dari pengertian beberapa sumber diatas maka informasi merupakan kumpulan data – data yang diolah sedemikian rupa sehingga dapat memberikan arti dan manfaat sesuai dengan keperluan tertentu yang bisa menjadi suatu informasi.

Suatu informasi yang bermanfaat, harus memiliki kualitas sebagai berikut :

a. Relevan

   Menambah pengetahuan atau nilai bagi para pembuat keputusan, dengan cara mengurangi ketidak pastian, menaikan kemampuan untuk memprediksi atau menegaskan ekspektasi semula.

b. Dapat dipercaya

   Bebas dari kesalahan atau bisa secara akurat menggambarkan kegiatan atau aktivitas organisasi.

c. Lengkap

   Tidak menghilangkan data penting yang dibutuhkan oleh para pemakai.

d. Tepat waktu

   Disajikan ada saat yang tepat untuk mempengaruhi proses pembuatan keputusan.

e. Dapat dipahami

   Disajikan untuk format mudah dimengerti.

f. Dapat diuji kebenarannya

   Memungkinkan dua orang yang kompeten untuk menghasilkan informasi yang sama secara independent.

2.2.3 Definisi Sistem Informasi

Menurut [Jog97] :

Sistem informasi adalah suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategis dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan – laporan yang diperlukan.

Sistem informasi merupakan kumpulan informasi – informasi yang saling berhubungan dan berinteraksi satu sama lain untuk keperluan tertentu.

Kegiatan sistem informasi adalah sebagai berikut :

a. Input

   Menggambarkan suatu kegiatan untuk menyediakan data untuk proses.

b. Proses

   Menggambarkan bagaimana suatu data diproses unuk menghasilkan suatu informasi yang bernilai tambah.

c. Penyimpanan

   Suatu kegiatan untuk memelihara dan menyimpan data.

d. Output

   Suatu kegiatan untuk menghasilkan laporan dari suatu proses informasi.

e. Kontrol

   Suatu aktifitas untuk menjamin bahwa sistem informasi tersebut berjalan sesuai dengan harapan.

Komponen informasi terdiri dari :

1. *Hardwere* (Perangkat keras), terdiri dari komputer, printer, jaringan.
2. *Softwere* (Perangkat lunak)
3. Data merupakan komponen dasar informasi

   User (Manusia).

2.2.4 *Unified Modelling Language*

Mengenai pemahaman alat bantu dan perancangan berorientasi objek, maka penulis membangun sistem informasi yang berorientasi objek yang merupakan suatu perancangan yang berbeda dengan berorientasi data. Namun secara konteks perancangan ini digunakan untuk membangun sistem informasi sesuai kebutuhan dari pemakai (*user*) sistem informasi.

Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan *syntax* / semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML syntax mendefinisikan bagaimana bentuk - bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: Grady Booch OOD (*Object - Oriented Design*), Jim Rumbaugh OMT (*Object Modeling Technique*), dan Ivar Jacobson OOSE (*Object - Oriented Software Engineering*).

UML sendiri menyediakan alat – alat bantu berupa diagram visual yang mnggambarkan berbagai aspek yang ada didalam sistem. Adapun alat – alat bantu diagram yang sediakan didalam UML yaitu :

1. *use case diagram*
2. *class diagram*
3. *statechart diagram*
4. *activity diagram*
5. *sequence diagram*
6. *collaboration diagram*
7. *component diagram*
8. *deployment diagram*

Berikut merupakan penjelasan diagram – diagram UML :

1. *Use Case Diagram*

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya *login* ke sistem, meng*create* sebuah daftar belanja, dan sebagainya. Seorang/ sebuah aktor adalah sebuah *entitas* manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.

Use case diagram dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem.

Sebuah *use case* dapat meng*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di*include* akan dipanggil setiap kali *use case* yang meng*include* dieksekusi secara normal. Sebuah *use case* dapat di*include* oleh lebih dari satu use case lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*.

Sebuah *use case* juga dapat meng*extend use case* lain dengan *behaviour*nya sendiri. Sementara hubungan generalisasi antar use case menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

2. *Class Diagram*

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Class menggambarkan keadaan (*atribut / properti*) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (*metoda /* fungsi). *Class* diagram menggambarkan struktur dan *deskripsi class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain - lain.

Class memiliki tiga area pokok :

1. Nama (*stereotype*)
2. *Atribut*
3. *Metoda*

Atribut dan metoda dapat memiliki salah satu sifat berikut :

a. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan.

b. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak - anak yang mewarisinya.

c. *Public*, dapat dipanggil oleh siapa saja.

Class dapat merupakan implementasi dari sebuah *interface*, yaitu *class abstrak* yang hanya memiliki *metoda. Interface* tidak dapat langsung diinstansiasikan, tetapi harus *diimplementasikan* dahulu menjadi sebuah class. Dengan demikian *interface* mendukung *resolusi* metoda pada saat *run time*. Sesuai dengan perkembangan *class model*, *class* dapat dikelompokkan menjadi *package.* Kita juga dapat membuat diagram yang terdiri atas *package*.

Hubungan Antar *Class*

a. *Asosiasi*, yaitu hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa class lain, atau class yang harus mengetahui *eksistensi class* lain. Panah *navigability* menunjukkan arah *quer*y antar *class*.

b. Agregasi, yaitu hubungan yang menyatakan bagian (“terdiri atas..”).

c. Pewarisan, yaitu hubungan *hirarkis* antar *class*. *Class* dapat diturunkan dari class lain dan mewarisi semua *atribut* dan metoda *class* asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah *generalisasi*.

d. Hubungan dinamis, yaitu rangkaian pesan (*message*) yang di*passing* dari satu *class* kepada *class* lain. Hubungan dinamis dapat digambarkan dengan menggunakan *sequence diagram*.

3. *Statechart Diagram*

Statechart diagram menggambarkan transisi dan perubahan keadaan (dari satu state ke *state* lainnya) suatu objek pada sistem sebagai akibat dari stimuli yang diterima. Pada umumnya *statechart* diagram menggambarkan *class* tertentu (satu *class* dapat memiliki lebih dari satu *statechart* diagram).

Dalam UML, *state* digambarkan berbentuk segi empat dengan sudut membulat dan memiliki nama sesuai kondisinya saat itu. Transisi antar state umumnya memiliki kondisi *guard* yang merupakan syarat terjadinya transisi yang bersangkutan, dituliskan dalam kurung siku. *Action* yang dilakukan sebagai akibat dari *event* tertentu dituliskan dengan diawali garis miring. Titik awal dan akhir digambarkan berbentuk lingkaran berwarna penuh dan berwarna setengah.

4. Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing - masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity* diagram juga dapat menggambarkan proses *paralel* yang mungkin terjadi pada beberapa *eksekusi*.

Activity diagram merupakan state diagram khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity* diagram tidak menggambarkan *behaviour* internal sebuah sistem (dan interaksi antar *subsistem*) secara eksak, tetapi lebih menggambarkan proses - proses dan jalur - jalur aktivitas dari *level* atas secara umum.

Sebuah aktivitas dapat di*realisasi*kan oleh satu use case atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case*

menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas.

Sama seperti *state*, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan *behaviour* pada kondisi tertentu. Untuk mengilustrasikan proses - proses paralel (*fork* dan *join*) digunakan titik *sinkronisasi* yang dapat berupa titik, garis *horizontal* atau *vertikal*. *Activity* diagram dapat dibagi menjadi beberapa object *swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.

5. *Sequence Diagram*

Sequence diagram menggambarkan *interaksi* antar objek di dalam dan di sekitar sistem (termasuk pengguna, display, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. Sequence diagram terdiri atar dimensi *vertikal* (waktu) dan dimensi *horizontal* (objek - objek yang terkait).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah - langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara *internal* dan *output* apa yang dihasilkan. Masing - masing objek, termasuk aktor, memiliki *lifeline vertikal*.

Message digambarkan sebagai garis berpanah dari satu objek ke objek lainnya.

Pada *fase* desain berikutnya, *message* akan dipetakan menjadi operasi / *metoda* dari *class*. *Activation* bar menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah *message*. Untuk *objek - objek* yang

memiliki sifat khusus, standar UML mendefinisikan icon khusus untuk *objek boundary*, *controller* dan *persistent entity*.

6. *Collaboration Diagram*

Collaboration diagram juga menggambarkan interaksi antar *objek* seperti *sequence* diagram, tetapi lebih menekankan pada peran masing-masing objek dan bukan pada waktu penyampaian *message*. Setiap *message* memiliki *sequence number*, di mana *message* dari level tertinggi memiliki nomor 1. *Messages* dari *level* yang sama memiliki *prefiks* yang sama.

7. *Component Diagram*

Component diagram menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan (*dependency*) di antaranya.

Komponen piranti lunak adalah modul berisi *code*, baik berisi *source code* maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada *compile time*, *link time*, maupun *run time*. Umumnya komponen terbentuk dari beberapa *class* dan / atau *package*, tapi dapat juga dari komponen - komponen yang lebih kecil. Komponen dapat juga berupa *interface*, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lain.

8. *Deployment Diagram*

Deployment / *physical* diagram menggambarkan *detail* bagaimana komponen di*deploy* dalam *infrastruktur* sistem, di mana komponen akan terletak (pada mesin, *server* atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi *server*, dan hal-hal lain yang bersifat *fisikal*.

Sebuah *node* adalah *server*, *workstation*, atau piranti keras lain yang digunakan untuk men*deploy* komponen dalam lingkungan sebenarnya. Hubungan

antar *node* (misalnya *TCP / IP*) dan *requirement* dapat juga didefinisikan dalam diagram ini.

2.2.5 Elemen – Elemen Sistem

Ada beberapa macam elemen – elemen didalam sistem yaitu sebagai berikut :

a. Tujuan

Merupakan tujuan akhir atau sasaran dari pengolahan sistem tersebut yang dapat berupa :

1. Masalah
2. Kebutuhan
3. Tujuan Usaha
4. Prosedur pencapaian tujuan

b. Batasan

Merupakan daerah yang membatasi antara satu sistem dengan sistem yang lain atau dengan lingkungan luarnya sehingga memungkinkan suatu sistem dipandang sebagai satu kesatuan dalam sebuah ruang lingkup. Batasan – batasan yang ada dalam mencapai tujuan dari sistem dapat berupa :

1. Peraturan
2. Biaya – biaya
3. Personel
4. Peralatan

c. Penghubung Sistem

Penghubung merupakan suatu media yang menghubungkan antara subsistem yang satu dengan subsistem yng lain yang memungkinkan sumber daya mengalir dari subsistem satu ke subsistem yang lain.

d. Kontrol

Merupakan pengawasan dari pelaksanaan pencapaian tujuan sistem yang berupa :

1. Asal masukan
2. Frekuensi pemasukan data
3. Jenis pemasukan data

e. Input

Merupakan bagian dari sistem yang bertugas untuk menerima data masukan, dimana data dapat berupa :

1. Asal masukan
2. Frekuensi pemasukan data
3. Jenis pemasukan data

f. Proses

Merupakan bagian yang memproses masukan data menjadi keluaran berupa informasi yang sesuai dengan keinginan penerima dan dapat berupa :

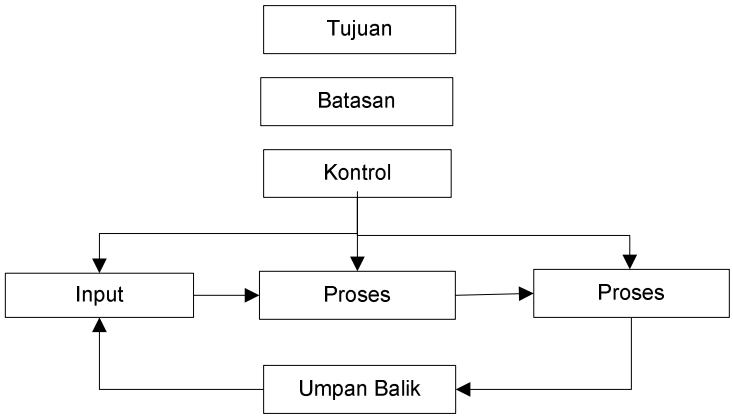
1. Laporan
2. Grafik, dll

g. Umpan balik

Umpan balik merupakan suatu reaksi yang dilakukan apabila mendapat suatu masukan dapat berupa :

1. Perbaikan
2. Pemeliharaan. dll

Keterkaitan elemen – elemen sistem yang terdapat pada sistem dapat dilihat pada gambar berikut :



Gambar 2.2 *Keterkaitan elemen – elemen sistem*

Sumber : [Jog 01]

2.2.6 Tingkah Laku Sistem

Sebagai sesuatu yang dinamik, sebuah sistem memiliki tingkah laku yang berbeda dari satu sistem dengan sistem yang lain. Berdasarkan pola tingkah laku, terdapat 5 jenis sistem yaitu :

a. Deterministic system

Adalah sistem dimana segala tingkah lakunya dapat diramalkan.

b. Probabilistic system

Adalah sistem dimana segala tingkah lakunya tidak dapat diramalkan.

c. *Closed system*

Adalah sistem yang tidak mempunyai relasi atau hubungan dengan lingkungan.

d. *Open system*

Adalah sistem yang mempunyai relasi atau hubungan dengan lingkungan.

2.2.7 Perancangan Sistem Informasi

Perancangan sistem adalah merancang atau membuat sistem baru yang diterapkan untuk mengatasi masalah yang lama.

Menurut [RR01] :

Dalam bukunya *Data processing sistem and concept* mengatakan tahap setelah analisis dari siklus pengembangan sistem adalah : pendefinisian dari kebutuhan – kebutuhan fungsional dan persiapan untuk rancang bangun implementasi, penggambaran bagaimana sistem dibentuk.

Menurut [Jog00] :

Menyatakan peranangan sistem dapat didefinisikan sebagai penggambaran, perencanaan dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah kedalam satu kesatuan yang utuh dan berfungsi.

Menurut [Geo00] :

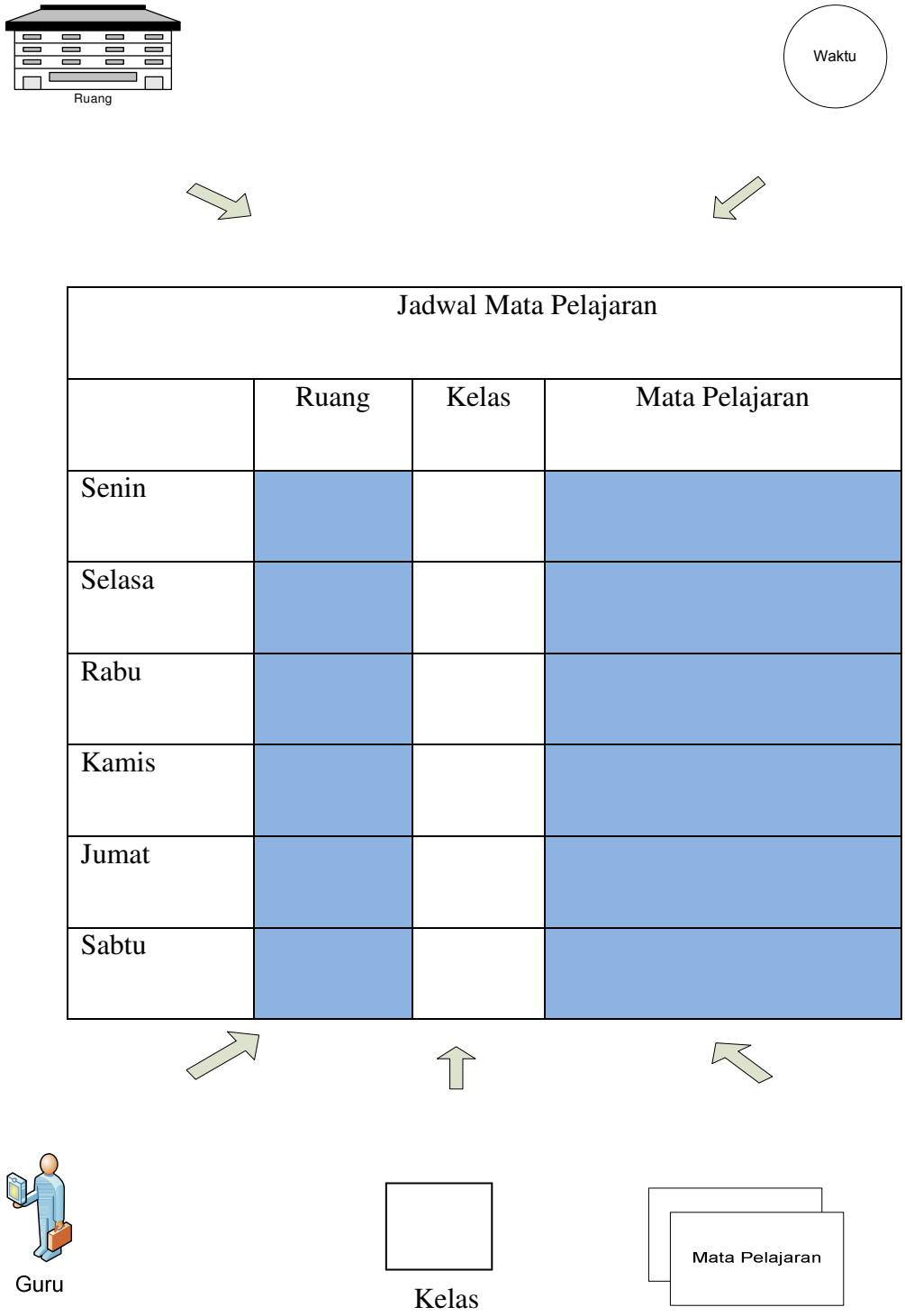
Dalam bukunya *Principles of management sistem* menetukan bagaimana suatu sistem akan menyelesaikan apa yang mesti diselesaikan,

tahap ini menyangkut mengkonfigurasikan dari komputer perngkat lunak dan perangkat keras dari suatu sistem, sehingga seekah instalasi dari sistem akan benar – benar memuaskan.

Perancangan sisem dapat diartikan sebagai tahap setelah analisis dari siklus pengembangan sistem, pendefinisian dari kebutuhan – kebutuhan fungsionalis, persiapan untuk rancangan bangun implementasi, menggambarkan bagaimana suatu sistem dibentuk (penggambaran, perencanaan, pembatasan sketsa) termasuk mengkonfigurasi komponen – komponen perangkat lunak dan perangkat keras dari suatu sistem.

2.3 Definisi Kasus Yang Dianalisis

Aplikasi penjadwalan mata pelajaran adalah bagaimana menjadwalkan sejumlah komponen yang terdiri dari guru, ruang kelas, mata pelajaran dan waktu dengan sejumlah batasan tertentu. Pembuatan jadwal mata pelajaran harus memperhatikan aturan dan batasan penjadwalan yang telah ditentukan sebelumnya. Disini dapat ditentukan adanya prioritas aturan atau batasan penjadwalan yang ingin dipakai dalam proses pembuatan jadwal mata pelajaran tersebut.



Gambar 2.3 *Batasan / Constrait*

Sumber : http://pepeel.wordpress.com/definisi-sistem/

Sesuai dengan penggambaran deskripsi umum sistem pada gambar diatas, aplikasi penjadwalan mata pelajarn ini dipengaruhi oleh beberapa

komponen yang terdiri atas guru, ruang kelas, waktu dan mata pelajaran dengan sejumlah batasan – batasan tertentu. Penerapan penjadwalan dalam aplikasi ini diharapkan mampu mengakomodasikan seluruh batasan yang telah ditentukan dengan skala prioritas agar kebutuhan guru dan siswa dapat disesuaikan dengan ruang kelas dan waktu yang tersedia. Dari batasan yang diterapkan pada penempatan jadwal mata pelajaran ini terdapat beberapa batasan yang dapat di bedakan menjadi batasan yang merupakan *hard contraint* dan *soft contraint.*

Contraint batasan sistem penjadwalan terdiri dari :

1. *Hard contraint :* batasan yang mutlak harus dipenuhi dalam proses penempatan jadwal.
2. *Soft contraint :* batasan dengan prioritas lebih rendah yang masih memungkinkan untuk tidak diterapkan atau menjadi syarat mutlak dalam proses penempatan jadwal.

Contoh *Hard contraint*

1. Tidak terdapat jam mengajar mata pelajaran dengan waktu dan hari yang sama.
2. Mata pelajaran dengan bobot kurang dari 2 jam dijadwalkan dengan 1 kali pertemuan dalam seminggu.
3. Hari aktif untuk belajar mengajar adalah hari senin hingga sabtu.

Contoh *Soft contraint*

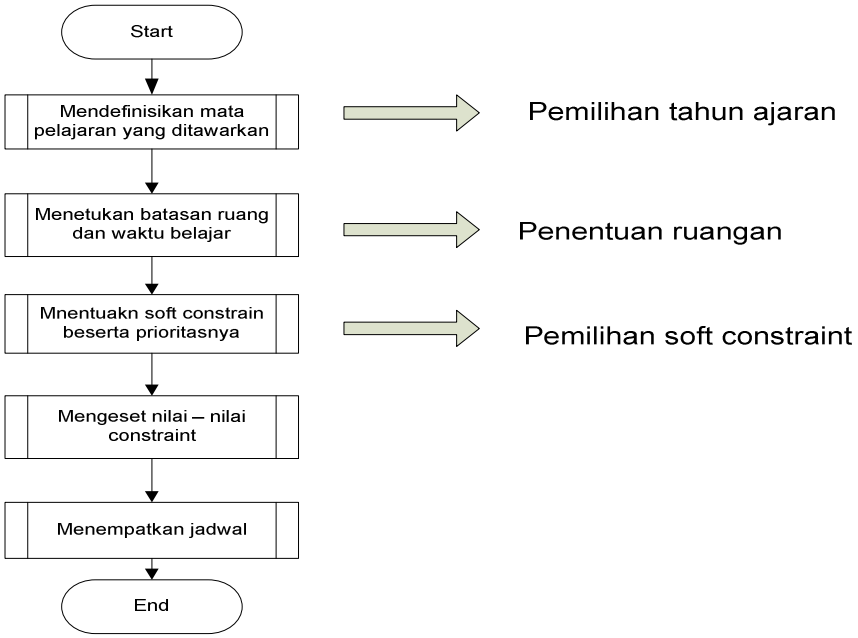
1. Guru dapat memesan waktu mengajar tertentu yang diinginkan.
2. Penepatan jadwal untuk waktu yang telah dipesan guru disesuaikan dengan prioritas guru.

3. Kelas paralel ditempatkan pada waktu bersamaan, kecuali apabila mempunyai guru pengajar yang sama.

Tahap – tahap penjadwalan

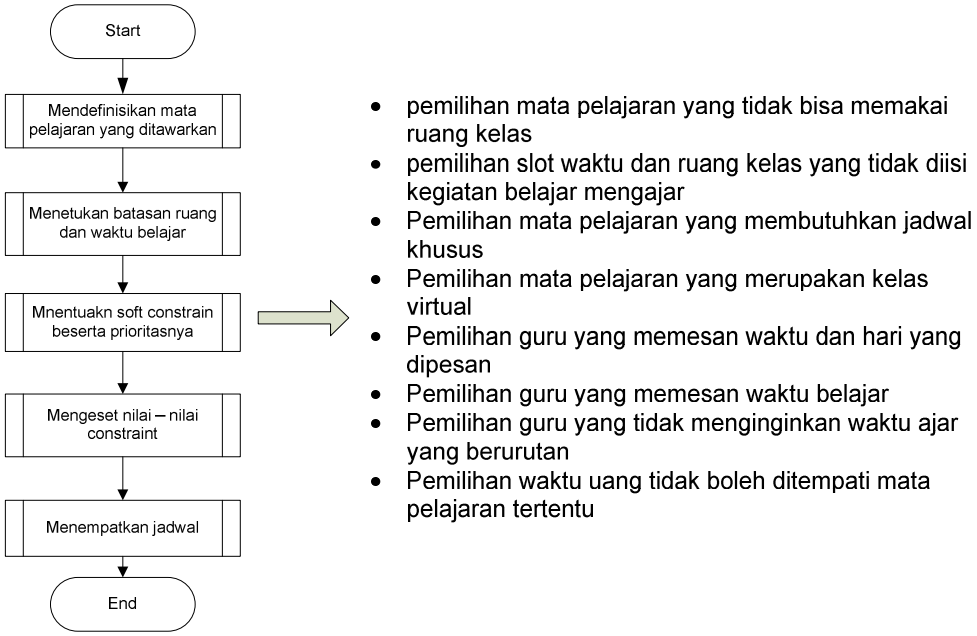
1. Pemilihan tahun ajaran dan semester.
2. Penentuan dan ruangan.
3. Pemilihan *soft contraint* dan prioritasnya.
4. Pemilihan mata pelajaran yang tidak membutuhkan ruang kelas.
5. Pemilihan slot waktu dan ruang kosong yang tidak diisi kegiatan belajar mengajar.
6. Pemilihan mata pelajaran yang mempunyai jadwal khusus.
7. Pemilihan mata pelajaran yang merupakan kelas *virtual*.
8. Pemilihan guru yang memesan waktu dan hari yang dipesan.
9. Penentuan prioritas guru yang memesan waktu belajar.
10. Pemilihan guru yang tidak menginginkan waktu ajar yang berurutan.
11. Pemilihan waktu yang tidak boleh ditempati mata pelajaran kelas tertentu.
12. Penempatan jadwal mata pelajaran.
13. Penempatan jadwal ujian.

Alur – alur proses penjadwalan



Gambar 2.4 *alur proses penjadwalan*

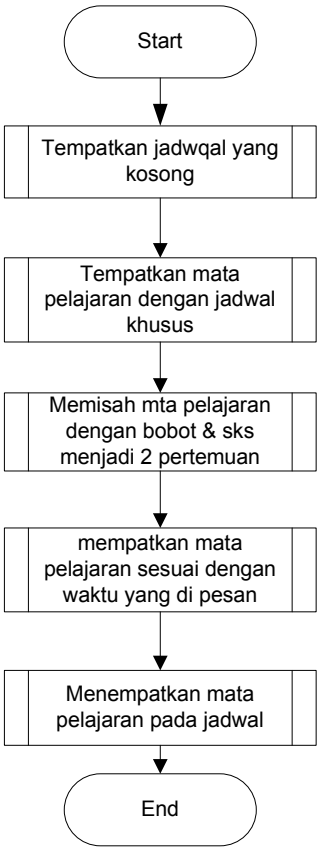
Sumber : http://pepeel.wordpress.com/definisi-sistem/



Gambar 2.5 *alur proses penjadwalan*

Sumber : http://pepeel.wordpress.com/definisi-sistem/

Menetapkan jadwal



Gambar 2.6 *alur menetapkan jadwal*

Sumber : http://pepeel.wordpress.com/definisi-sistem/

Fungsi - fungsi pokok

Secara garis besar, proses yang dilakukan dalam penjadwalan mata pelajaran ini di bagi menjadi :

1. Membuka *file project* data yang disimpan dalam file project akan di load dan dimasukan sebagai nilai batasan (*setting*) daiam proses penjadwalan.
2. Mendefinisikan mata pelajaran yang ditawarkan untuk mendefinisikan mata pelajaran yang ditawarkan, pertama - tama pengguna harus memilih tahun ajaran dan semester yang diinginkan dan menyimpannya sebagai

setting project. Kemudian dilakukan *query* dari *database* berdasarkan tahun ajaran, semester, dan kode jurusan.

3. Menentukan batasan ruang dan waktu pelajaran proses diawali dengan memilih jumlah ruang kelas yang akan digunakan dalam proses penjadwalan serta yang terdapat dalam satu hari. Hari aktif belajar mengajar ditetapkan sebanyak 6 hari dalam satu minggu, yaitu hari senin hingga sabtu. Nama dari masing – masing ruang kelas dapat diubah oleh *admin*. Jika tidak, nama dari masing-masing ruangan tersebut akan diset dengan nilai *default*.
4. Menentukan *soft constraint* beserta prioritas - prioritasnya menentukan urutan bagaimana batasan tersebut akan lebih diutamakan untuk dipenuhi. Semakin tinggi prioritas, maka batasan tersebut akan lebih di utamakan untuk di penuhi dibandingkan dengan batasan yang memiliki prioritas lebih rendah.

   Soft constarint yang dapat dipilih adalah:

   a. Guru memesan waktu tertentu untuk mengajar.
   b. Kelas pararel berada di shift yang sama, ruangan yang beda.
   c. Guru tidak mengajar di waktu yang berurutan.
   d. Mengeset nilai – nilai batasan.

   Beberapa batasan yang perlu diset nilainya seperti:

   a. Mata pelajaran yang tidak menempati ruangan kelas.
   b. Jadwal tertentu yang dikosongi.
   c. Mata pelajaran yang mempunyai jadwal khusus (misalnya lab komputer).

d. Guru yang memesan waktu tertentu untuk mengajar.
e. Urutan prioritas guru yang memesan waktu mengajar.
f. Guru yang tidak menginginkan waktu mengajar yang berurutan.
g. Hari yang tidak ditempati mata pelajaran semeter tertentu.

5. Menyimpan hasil jadwal, setelah mata pelajaran ditempatkan pada jadwal, maka hasil akan ditampilkan pada aplikasi. Untuk mempermudah pendokumentasian.
6. Menyimpan *file project* seluruh nilai - nilai batasan yang telah diset dapat disimpan kedalam sebuah *file project* sehingga apabila pengguna akan menjalankan apliakasi kembali dilain waktu dan ingin menggunakan nilai batasan yang pernah digunakan, pengguna dapat membuka *file project* tersebut.
7. Menempatkan jadwal ujian selain jadwal mata pelajaran untuk tahun ajaran dan semester tertentu, batasan yang ditambahkan pada penempatan jadwal ujian untuk periode tersebut. Batasan yang ditambahkan pada penempatan jadwal ujian ini adalah adanya mata pelajaran yang tidak mengadakan ujian sehingga tidak diikut sertakan dalam proses penempatan mata pelajaran dalam mata pelajaran. Selain itu itu, kelas pararel akan ditempatkan di waktu yang sama. Batasan *default* pada penempatan jadwal ujian ini sama dengan batasan pada proses penempatan jadwal mata pelajaran. Tetapi karena pada jadwal ujian ini penempatan kelas pararel pada waktu yang sama lebih diutamakan, maka batasan yang telah ditetapkan akan disesuaikan sehingga penempatan kelas pararel pada

waktu yang sama menempati prioritas paling tinggi. Secara garis besar, proses penempatan jadwal ujian ini juga hampir sama dengan penempatan jadwal mata pelajaran.

2.4 Arsitektur Aplikasi

Arsitektur aplikasi terdiri dari pengertian jaringan komputer, jenis – jenis jaringan komputer, topologi jaringan komputer dan manfaat komputer.

2.4.1 Pengertian Jaringan Komputer

Menurut [FJ05] :

Jaringan komputer adalah sekumpulan komputer autonomous berjumlah banyak yang terpisah – pisah tetapi saling berhubungan dalam melaksanakan tugasnya.

Tujuan dibangunnya jaringan komputer adalah membawa informasi secara tepat dan tanpa adanya kesalahan dari pengirim (*transisi*) menuju ke sisi penerima (*receiver*) melalui komunikasi. TCP / IP (*Transmission control protocol / internet protocol*) merupakan *protocol standard* internet yang digunakan untuk melakukan koneksi ke internet protocol. TCP / IP memiliki beberapa subyek *protocol* yang berbeda yang beroperasi pada lapisan yang berbeda dan mempunyai tugas masing – masing.

Berkat adanya *protocol* ini setiap komputer dapat berhubungan secara fleksibel dengan host – host yang terkoneksi.

2.4.2 Jenis – Jenis Jaringan Komputer

Ada beberapa macam jaringan komputer antara lain :

1. LAN

Local area network (LAN) merupkan jaringan milik pribadi didalam sebuah gedung atau kampus yang berukuran sampai beberapa kilometer.

2. MAN

Metropolitan area network (MAN) merupakan versi LAN yang berukuran lebih besar dan biasanya memakai teknologi yang sama dengan LAN.

3. WAN

Widel area network (WAN) merupakan sebuah jaringan yang memiiki jarak sangat luas, karena radiusnya mencakup sebuah negara dan benua.

4. Jaringan Tanpa kabel

Jaringan yang tidak menggubakan kabel atau disebut *wireless.*

2.4.3 Topologi Jaringan Komputer

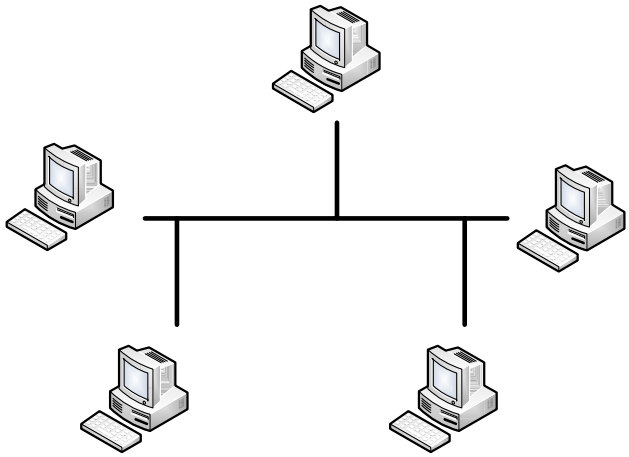
Topologi jaringan adalah gambaran secara fisik dari pola hubungan antara komponen – komponen jaringan, yang meliputi *server, workstation, hub* dan pengkabelan. Terdapat tiga macam topologi jaringan umum digunakan, yaitu bus, star, dan ring.

1. *Topologi Bus*

Pada *topologi bus* digunakan sebuah kabel tunggal atau kabel pusat dimana seluruh *workstation* dan *server* dihubungkan.

Keunggulan topologi bus adalah pengembangan jaringan atau penambahan *workstation* baru dapat dilakukan dengan mudah tanpa mengganggu *workstation* lainnya.

Kelemahan dari *topologi* ini adalah bila terdapat gangguan disepanjang kabel pusat maka keseluruhan jaringan akan mengalami ganuguan.



Gambar 2.7 *Topologi bus*

Sumber : [FJ05]

2. *Topologi Star*

Pada *topologi star*, masing – masing *workstation* dihubungkan langsung ke *server* atau *hub.*

Keunggulan dari *topologi star* adalah bahwa dengan adanya kabel tersendiri untuk setiap *workstation* ke *server*, maka *bandwidth* atau lebar jalur komunikasi dalam kabel akan semakin lebar sehingga akan meningkatkan untuk kerja jaringan secara keseluruhan. Dan juga bila

terdapat gangguan di suatu jalur kabel maka gangguan hanya akan terjadi dalam komunikasi antara *workstation* yang bersangkutan dengan *server*, jaringan secara keseluruhan tidak mengalami gangguan.

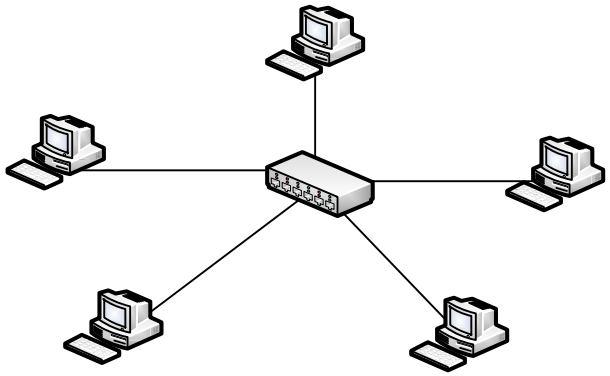
Kelemahan dari *topologi star* adalah kebutuhan kabel yang lebih besar dibandingkan dengan topologi lainnya.

Keuntungan *topologi star* adalah :

1. Paling fleksibel dibanding dengan *topologi* lain
2. Pemasangan / perubahan stasiun sangat mudah dan tidak mengganggu bagian jaringan yang lain.
3. Kontrol terpusat
4. Kemudahan deteksi dan isolasi kesalahan kritis
5. Kemudahan mengola jaringan

Kerugian *topologi star* adalah :

1. Boros kabel
2. Perlu penanganan khusus
3. Control terpusat (*hub*) jadi elemen kritis.



Gambar 2.8 *Topologi Star*

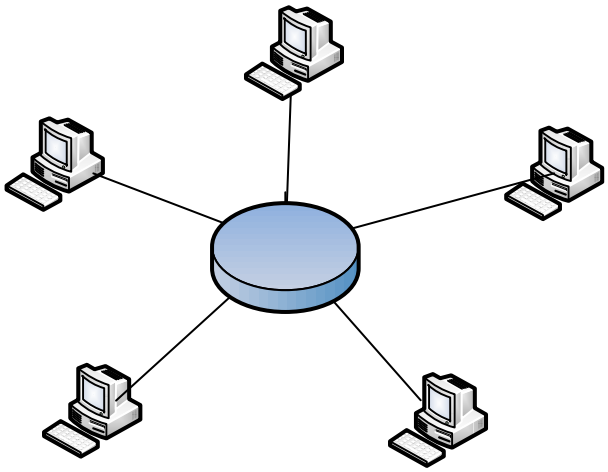
Sumber : [FJ05]

3. Topologi Ring

Di dalam *topologi ring* semua *workstation* dan *server* dihubungkan sehingga terbentuk suatu pola lingkaran atau cincin. Tiap *workstation* maupun *server* akan menerima dan melewatkan informasi dari satu komputer ke komputer lain, bila alamat – alamat yang dimaksud sesuai maka informasi diterima dan bila tidak informasi akan dilewatkan.

Kelemahan dari *topologi ring* adalah setiap node dalam jaringan akan selalu ikut serta mengelola informasi yang akan dilewatkan dalam jaringan, sehingga bila terdapat gangguan disuatu node maka seluruh jaringan akan terganggu.

Kelebiham dari *topologi ring* adalah tidak terjadinya *collision* atau tabrakan pengiriman data seperti pada *topologi bus*, karena hanya satu node dapat mengirimkan data pada suatu saat.



Gambar 2.9 *Topologi ring*

Sumber : [FJ05]

Adapun metode yang saat ini digunakan adalah *topologi star* dimana *topologi* ini mempunyai ciri : control terpusat, semua *link,* harus

melewati pusat yang menyalur data tersebut kesemua simpul / *client yang dipilihnya.* Simpul pusat dinamakan *stasiun primer* atau *server* dan lainnya dinamakan *stasiun sekunder* atau *client server.* Setelah hubungan jaringan dimulai oleh *server* maka setiap *client server* sewaktu – waktu dapat menggunakan hubungan jaringan tersebut tanpa menunggu perintah dari *server.*

2.4.4 Manfaat Jaringan Komputer

Manfaat yang didapat dari membangun jaringan komputer adalah sebagai berikut :

1. *Sharing resources*

Sharing Resources betujuan agar seluruh program, peralatan / *periperal* lainnya dapat dimanfaatkan oleh setiap orang yang ada pada jaringan komputer tanpa terpengaruh oleh lokasi maupun pengaruh oleh pemakai. Dengan kata lain, seorang pemakai yang letaknya jauh sekalipun dapat memanfaatkan data maupun informasi yang lainya tanpa mengalami kesulitan

2. Media komunikasi

Jaringan komputer memungkinkan terjadinya komunikasi antara pengguna, baik untuk *teleconference* maupun untuk mengirim pesan / informasi yang penting lain.

3. Integrasi data

Pembangunan jaringan komputer dapat mencegah ketergantungan pada komputer pusat. Setiap proses data tidak harus dilakukan satu komputer saja.

4. Pengembangan dan pemeliharaan

Dengan adanya jaringan komputer ini, maka perkembangan peralatan dapat dilakukan dengan mudah dan dapat menghemat biaya. Jaringan komputer bisa memudahkan pemakai dalam merawat *harddisk* dan peralatan lainnya. Misalnya untuk memberikan perlindungan serangan *virus* maka pemakai cukup memusatkan perhatian pada *harddisk* yang ada komputer pusat.

5. Keamanan data

Sistem jaring komputer memberikan perlindungan terhadap data jaminan keamanan data tersebut diberikan melalui pengaturan hak akses para pemakai dan *password,* serta teknik perlindungan terhadap *harddisk* sehingga data mendapatkan perlindungan yang efektif.

6. Sumber daya lebih efesien dan informasi terkini

Dengan adanya pemakaian sumber daya secara bersama – sama maka pemakai bisa mendapatkan hasil dengan maksimal dan kualitas yang tinggi. Selain itu data atau informasi yang diakses selalu terbaru, karena setiap ada perubahan yang terjadi dapat secara langsung diketahui oleh setiap pemakai.

2.5 Pengertian *Client / Server*

Pada mulanya *client* dirancang untuk menghasilkan kinerja yang lebih besar dan hanya mengubah sedikit biaya, dengan cara memindahkan sebagian tugas pemprosesan dari komputer *client* ke komputer *server*.

a. Pengertian *client*

Menurut [FJ05] :

Client adalah komputer-komputer yang menerima atau menggunakan fasilitas yang di sediakan oleh *server*.

Sementara menurut [Mad03] :

Client adalah sebuah *aplikasi* yang memungkinkan pengguna untuk mengakses servis atau layanan dari komputer *server*.

b. Pengertian server

Menurut [FJ05] :

Server adalah komputer yang menyediakan fasilitas bagi komputer-komputer lain di dalam jaringan dan *server*, dijaringan tipe *client - server* disebut dengan *dedicated server* karena murni berperan sebagai *server* yang menyediakan fasilitas kepada *workstation*.

Server menurut [Mad03] :

1. Sebuah komputer di internet atau di jaringan lainnya yang menyimpan *file* dan membuat *file* tersebut tersedia untuk di ambil jika di butuhkan.
2. Sebuah aplikasi jaringan komputer yang di gunakan untuk melayani banyak pengguna dalam satu jaringan.

Pengertian *client - server*

Menurut [Mad03] :

Client - sever adalah suatu bentuk arsitektur dimana *client* adalah perangkat yang menerima yang akan menampilkan antarmuka pemakai dan menjalankan aplikasi (komputer) dan *server* adalah perangkat yang menyediakan dan bertindak sebagai pengelola aplikasi, data dan keamanannya (*server* atau *mainframe*).

Sistem *client - sever* biasanya berjalan pada setidaknya dua sistem yang berbeda. Satu komputer bertindak sebagai client dan lainnya sebagai *server*, tapi *client* dan *server* juga bisa berada pada satu sistem komputer. Biasanya sebuah *server* melayani beberapa komputer *client* walaupun mungkin juga hanya melayani satu *client*.

Fungsi *client - server* biasanya dilakukan oleh *file server*, kecuali apabila dibutuhkan kinerja yang maksimal maka di gunakan *server* yang khusus. *Client* biasanya berupa komputer destop yang terhubung dalam jaringan. Apabila pemakaian mengambil atau menyimpan informasi bagian aplikasi *client* akan mengeluarkan permintaan yang akan di kirim ke *server*, *sever* kemudian menjalankan permintaan dan mengirimkan informasi kepada *client*.

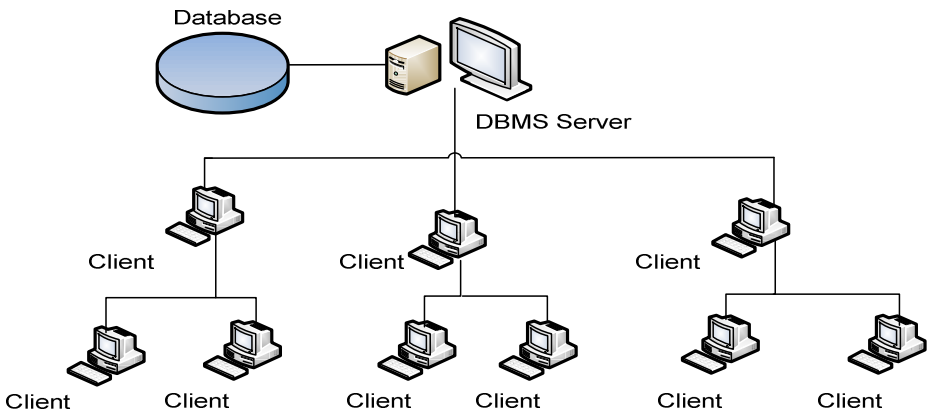
Keunggulan :

1. Kecepatan akses lebih tinggi karena penyediaan fasilitas jaringan dan pengelolaanya di lakukan secara khusus oleh satu komputer (*server*) yang tidak di bebani dengan tugas lain sebagai *workstasion*.

2. Sistem keamanan dan administrasi jaringan lebih baik, karena terdapat seorang pemakai yang bertugas sebagai adminstrator jaringan, yang mengelola administrasi dan sistem keamanan jaringan.
3. Sistem *back - up* data lebih baik, karena pada jaringan *client - server back - up* dilakukan terpusat di *server*, yang akan *memback - up* seluruh data yang di gunakan di dalam jaringan.

Kelemahan :

1. Biaya operasional relatif lebih mahal.
2. Diperlukan adanya satu komputer khusus yang berkemampuan lebih untuk di tugaskan sebagai *server*.
3. Kelangsungan jaringan sangat tergantung pada *server*. Bila *server* mengalami gangguan maka secara keseluruhan jaringan akan terganggu.



Gambar 2.10 *Sistem clien server kompleks*

Sumber : [Fat99]

2.6 Perangkat Lunak Pendukung

Perangkat lunak pendukung yang digunakan dalam pembuatan program ini adalah sebagai beikut :

1. *Visual Basic* 6.0

Visual Basic adalah salah satu bahasa pemprograman yang sudah dikenal oleh pemakai komputer. Bahasa ini dapat dikatakan sebagai bahasa pemprograman dasar atau yang paling mudah yang paling sesuai dengan namanya. Namun sebenarnya nama *basic* adalah kependekan dari kata-kata :

B (*Beginner's*), A(*All-Purpose*), S(*Symbol*), I(*Intrucsion*), C(*code*).

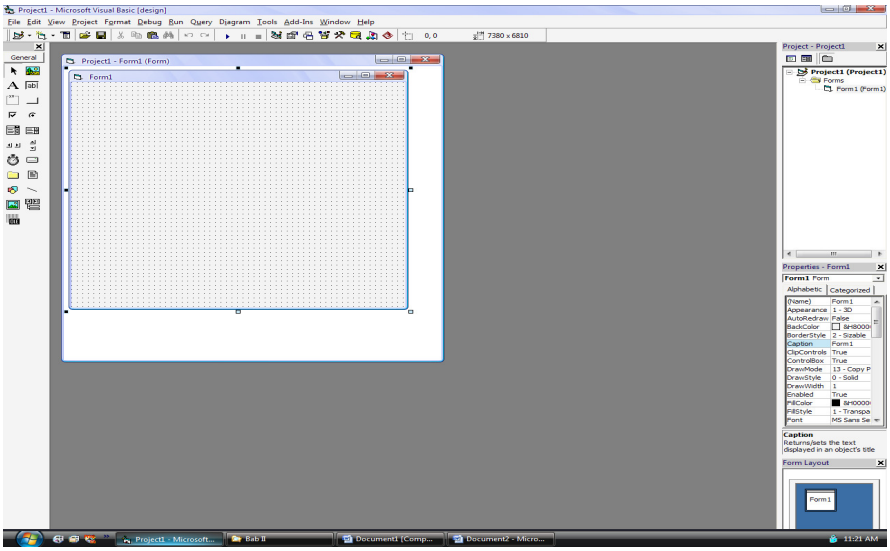
Bahasa ini partama kali muncul pada tahun 1960 dan di perkenalkan oleh *dartmouth college*.

Sejak kemunculannya pada tahun 1960, bahasa *basic* telah mengalami perkembangan yang pesat sekali. Di tahun 1970 digunakan oleh Bill Gates dan Paul Allen untuk mengontrol *mikro* komputer *alltair* dengan menggunakan pita kaset. Kemudian bahasa *basic* diikuti oleh pengembangan – pengembangan *softwere* lain dengan nama yang berbeda, namun aturan dan bahasa yang digunakan adalah sama.

Munculnya *GW – Basic, Qbasic, Quick Basic,* dan lain sebagainya semakin mempopulerkan bahasa *basic* ini untuk digunakan pada *mikro* komputer sebagai bahasa pemrograman untuk membuat aplikasi. *Visual Basic* untuk *DOS* dan untuk *windows* diperkenalkan pada tahun 1991. Versi 3.0 dari *Visual Basic* dikeluarkan pada tahun 1993 dan lebih mengalami kemajuan yang pesat dibandingkan dengan versi sebelumnya. Pada akhr tahun 1995 keluar versi baru *Visual Basic* 4.0 mendukung proses 32 bit. Pada akhir tahun1996 diluncurkan *Visual*

Basic versi 5.0 dengan kelebihan yang dapat mendukung *control activex* dan mulai menghapus atau menghilangkan dukungan proses terhadap 16 bit. Sekarang muncul versi *Visual Basic* 6.0 yang memiliki beberapa kelebihan seperti :

1. *Kompiler* yang sangat cepat.
2. *Control* data objek untuk *activex* yang baru.
3. Dapat mendukung *database* yang terintegrasi dengan variasi aplikasi yang sangat luas.
4. Perancangan data laporan yang lebih baru.
5. Adanya *package* dan *deployment wizard* yang bisa digunakan untuk membuat distribusi *disk* dari aplikasi yang kita buat.
6. Adanya tambahan dukungan terhadap internet.



Gambar 2.11 *Tampilan Aplikasi Visual Basic*

2. *Database Microsoft SQL Server 2000*

Pada saat ini banyak sekali program *database* yang ditawarkan seperti : *Acces, SQLServer, MySQL, Perl, PostgreeSQL, python, Java*, dll.

Microsoft SQL Server 2000 adalah *Relation database management sytem (RDBMS)* yang handal. Di *design* untuk mendukung proses transaksi besar seperi order *entry online, inventory,* akuntansi atau manufaktur. *Microsoft SQL Server 2000* dapat dijalan kan pada NT 4.0 *Server* atau *Microsoft 2000 server* selain itu dapat juga di install pada personal deskop di windows 2000 profesional, atau windows 98 milenium.

Kelebihan *Microsoft SQL Server 2000* dalam pembuatan *database* adalah sebagai berikut :

1. Mempunyai *transaction log* tersendiri dan mengatur transaksi dalam *database.*
2. Data dapat berkisar antara 1 MB sampai 1.048.516 TB
3. Dapat menambah ukuran data secara manual atau otomatis
4. Dapat di set sesuai dengan keinginan, misalnya sebuah *database* hanya dapat dibaca tetapi tidak bisa di edit.