BAB II

LANDASAN TEORI

2.1 Landasan Teori

2.1.1 Defenisi Sistem Informasi

Sistem informasi didefinisikan oleh Robert A. Leitch dan K. Roscoe Davis sebagai berikut:[1]

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

2.1.2 Komponen Sistem Informasi

John Burch dan Gary Grudnitski mengemukakan bahwa sistem informasi terdiri dari komponen-komponen yang disebutnya dengan istilah blok bangunan (*building block*), yaitu blok masukan (*input block*), blok model (*model block*), blok keluaran (*output block*), blok teknologi (*technologi block*), blok basis data (*database block*), dan blok kendali (*komputers block*). Sebagai suatu sistem, keenam blok tersebut masing-masing saling berinteraksi satu dengan yang lainnya membentuk satu kesatuan untuk mencapai sasarannya.[1]

2.1 Blok Masukan

Input mewakili data yang masuk ke dalam sistem informasi. Input disini termasuk metode-metode dan media untuk menangkap data yang akan dimasukkan, yang dapat berupa dokumen-dokumen dasar.

2.2 Blok Model

Blok ini terdiri dari kombinasi prosedur, logika dan model matematik yang akan memanipulasi data input dan data yang tersimpan di basis data dengan cara yang sudah tertentu untuk menghasilkan keluaran yang diinginkan.

2.3 Blok Keluaran

Produk dari sistem informasi adalah keluaran yang merupakan informasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta semua pemakai sistem.

2.4 Blok Teknologi

Teknologi merupakan "kotak alat" (*tool box*) dalam sistem informasi. Teknologi digunakan untuk menerima output, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran dan membantu pengendalian dari sistem secara keseluruhan. Teknologi terdiri dari 3 bagian utama, yaitu teknisi (*humanware* atau *brainware*), perangkat lunak (*software*), dan perangkat keras (*hardware*). Teknisi dapat berupa orang-orang yang mengetahui teknologi dan membuatnya dapat beroperasi. Misalnya teknisi adalah operator komputer,

pemrogram, operator pengolah kata, spesialis telekomunikasi, analis sistem, penyimpanan data dan lain sebagainya.

2.5 Blok Basis Data

Basis data (*database*) merupakan kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya. Data perlu disimpan di dalam basis data untuk keperluan penyediaan informasi lebih lanjut. Data di dalam basis data perlu diorganisasikan sedemikian rupa, supaya informasi yang dihasilkan berkualitas. Organisasi basis data yang baik juga berguna untuk efisiensi kapasitas penyimpanannya. Basis data diakses atau dimanipulasi dengan menggunakan perangkat lunak paket yang disebut dengan DBMS (*Database Management Systems*).

2.6 Blok Kendali

Banyak hal yang dapat merusak sistem informasi, seperti misalnya bencana alam, api, temperatur, air, debu, kecurangan-kecurangan, kegagalan-kegagalan sistem itu sendiri, kesalahan-kesalahan, ketidak efisienan, sabotase dan lain sebagainya. Beberapa pengendalian perlu dirancang dan diterapkan untuk meyakinkan bahwa hal-hal yang dapat merusak sistem dapat dicegah ataupun bila terlanjur terjadi kesalahan-kesalahan dapat langsung cepat diatasi.

2.1.3 Arsitektur Informasi

Arsitektur Informasi adalah suatu pemetaan atau rencana kebutuhan-kebutuhan informasi di dalam organisasi. Arsitektur ini berguna sebagai penuntun bagi operasi sekarang atau menjadi *blue print* untuk arahan dimasa mendatang.

Tujuan dari arsitektur ini adalah agar bagian teknologi informasi memenuhi kebutuhan-kebutuhan bisnis strategi organisasi oleh karena itu arsitektur informasi memadukan kebutuhan informasi, komponen sistem informasi, dan teknologi pendukung.

2.2 Rekayasa Perangkat Lunak

2.2.1. Defenisi Rekayasa Perangkat Lunak

Rekayasa perangkat lunak adalah pengembangan dari penggunaan prinsip pengembangan suara untuk memperoleh perangkat lunak secara otomatis yang reliabel dalam bekerja secara efisien pada mesin nyata.[3]

2.2.2. Model Proses Rekayasa Perangkat Lunak

Model proses untuk rekayasa perangkat lunak dipilih berdasarkan sifat aplikasi dan proyeknya, metode dan alat-alat bantu yang akan dipakai, dan kontrol serta penyampaian yang dibutuhkan. Perkembangan perangkat lunak bisa dianggap sebagai lingkaran pemecahan masalah dimana terdapat empat keadaan berbeda, yaitu status quo, definisi masalah, perkembangan teknis memecahkan masalah di

keseluruhan aplikasi dari banyak aplikasi, dan integrasi pemecahan menyampaikan hasil kepada siapa yang membutuhkan pertama kali. Bermacam-macam model proses yang berbeda pada perangkat lunak sebagai berikut:

1. Model Sekuensial Linier atau Waterfall

Sekuensial linier mengusulkan sebuah pendekatan kepada perkembangan perangkat lunak yang sistematik dan sekuensial yang mulai pada tingkat dan kemajuan sistem pada seluruh analisis, desain, kode, pengujian, dan pemeliharaan.

2. Model Prototipe

Prototyping paradigm dimulai dengan pengumpulan kebutuhan. Pengembang dan pelanggan bertemu dan mendefinisikan obyektif keseluruhan dari perangkat lunak, mengidentifikasi segala kebutuhan yang diketahui, dan area garis besar dimana definisi lebih jauh merupakan keharusan kemudian dilakukan “perancangan kilat”. Perancangan kilat berfokus pada penyajian dari aspek-aspek perangkat lunak tersebut,yang akan nampak bagi pelanggan / pemakai (contohnya pendekatan input dan format output).

3. Model RAD

Rapid Application Development (RAD) adalah sebuah model proses perkembangan perangkat lunak sekuensial linier yang menekankan siklus perkembangan yang sangat pendek. Model RAD ini merupakan sebuah adaptasi

“kecepatan tinggi” dari model sekuensial linier di mana perkembangan cepat dicapai dengan menggunakan pendekatan konstruksi berbasis komponen.

4. Model Proses Perangkat Lunak Evolusioner

Model evolusioner adalah model iteratif. Model itu ditandai dengan tingkah laku yang memungkinkan perekayasa perangkat lunak mengembangkan versi perangkat lunak yang lebih lengkap sedikit demi sedikit.

a. Model Pertambahan

Model inkeremental menggabungkan elemen-elemen model sekuensial linier (diaplikasikan secara berulang) dengan filosofi prototipe iteratif. Model pertambahan memakai urutan-urutan linier di dalam model yang membingungkan, seiring dengan laju waktu kalender. Model pertambahan berfokus pada penyampaian produk operasional dalam setiap pertambahannya.

b. Model Spiral

Model spiral (spiral model) adalah model proses perangkat lunak yang evolusioner yang merangkai sifat iteratif dari prototipe dengan cara komputer dan aspek sistematis dari model sekuensial linier.

c. Model Rakitan Komponen

Model ratikan komponen menggabungkan beberapa karakteristik model spiral. Model ini bersifat evolusioner, sehingga membutuhkan pendekatan iteratif

untuk mencapai perangkat lunak. Tetapi model rakitan komponen merangkai aplikasi dari komponen perangkat lunak sebelum dipaketkan (kadang-kadang disebut "kelas").

d. Model Perkembangan Konkuren

Model proses yang konkuren dapat disajikan secara skematis sebagai sederetan aktivitas teknik mayor, tugas-tugas, dan keadaannya yang lain. Contohnya aktivitas rekayasa yang dibatasi untuk model spiral dipenuhi dengan melakukan tugas-tugas sebagai berikut: prototyping dan atau pemodelan analisis, spesifikasi kebutuhan, dan rancangan.

5. Model Formal

Model metode formal mencakup sekumpulan aktivitas yang membawa kepada spesifikasi matematis perangkat lunak komputer. Metode formal memungkinkan perekayasa perangkat lunak untuk mengkhususkan, mengembangkan, dan memverifikasi sistem berbasis komputer dengan menggunakan notasi matematis yang tepat.

2.3 Konsep Basis Data

2.3.1 Defenisi Konsep Basis Data

Basis data dimaksudkan untuk mengatasi problem pada sistem yang memakai pendekatan berbasis berkas.[4]

Sistem basis data adalah suatu sistem menyusun dan mengelola record-record menggunakan komputer untuk menyimpan atau merekam serta memelihara data opersional lengkap sebuah organisasi/perusahaan sehingga mampu menyediakan informasi yang optimal yang diperlukan pemakai untuk proses mengambil keputusan.[4]

Untuk mengelola basis data diperlukan perangkat lunak yang disebut DBMS. DBMS adalah perangkat lunak sistem yang memungkinkan para pemakai membuat, memelihara, mengontrol, dan mengakses basis data dengan cara yang praktis dan efisien.[4]

2.3.2 Tahapan Analisis Basis Data

Analisis basis data merupakan langkah untuk menentukan basis data yang diharapkan dapat mewakili seluruh kebutuhan pengguna. analisis basis data terdiri atas analisis basis data secara konseptual, analisis basis data secara logis, dan analisis basis data secara fisis.[4]

2.3.3 Model ERD *(Entity-Relationship Diagram)*

ERD hanya berfokus pada data, dengan menunjukkan “jaringan data” yang ada untuk suatu sistem yang diberikan. ERD sangat berguna bagi aplikasi di mana data dan hubungan yang mengatur data sangatlah kompleks. ERD pada mulanya diusulkan oleh Peter Chen untuk desain sistem database relasional dan telah dikembangkan oleh yang lainnya. Serangkaian komponen utama diidentifikasikan

untuk ERD : objek data, atribut, hubungan dan berbagai tipe indikator. Tujuan utama dari ERD adalah untuk mewakili objek data dan hubungan mereka.[4]

Kardinalitas model data harus dapat merepresentsikan jumlah peristiwa dari objek di dalam hubungan yang diberikan. Tillmann mendefinisikan kardinalitas dari object-relationship pair dengan cara sebagai berikut: kardinalitas merupakan spesifikasi dari sejumlah peristiwa dari satu [objek] yang dapat dihubungkan ke sejumlah peristiwa dari [objek] yang lain. [4]

Secara skematik, basis data dapat dideskripsikan secara grafis dengan ERD yang memiliki komponen utama, yaitu :[4]

1. Entity

Entity adalah sesuatu yang dapat diberikan dalam dunia nyata dengan keberadaan yang bebas baik secara fisik maupun secara abstrak (konsep), mempunyai karakteristik tertentu, dimana informasi yang berkaitan dengannya dikumpulkan. Tipe entitas adalah sekumpulan entitas yang menggunakan sifat dan karakteristik yang sama, sedangkan instance entitas adalah empat persegi panjang serta pemberian nama biasanya dengan menggunakan kata benda.

2. Relasi

“Relasi adalah hubungan yang terjadi antara instance dari satu atau lebih tipe entitas. Relationship tidak mempunyai keberadaan fisik kecuali yang mewarisi dari

hubungan entitas tersebut. Relationship set adalah kumpulan relatioanship yang sejenis. Simbol yang menggunakan kata kerja.

3. Kardinalitas Pemetaan (*mapping constraint*)

Kardinalitas pemetaan atau rasio kardinalitas menunjukkan jumlah entity yang dapat dihubungkan ke satu entity lain dengan suatu relationship sets. Kardinalitas pemetaan meliputi :

a. Hubungan satu ke satu (*one to one*)

Yaitu satu entity dalam A dihubungkan dengan maksimum satu entity dalam B.

b. Hubungan satu ke banyak (*one to many*)

Yaitu satu entity dalam A dihubungkan dengan sejumlah entity dalam B. Satu entity dalam B dihubungkan dengan maksimum satu entity dalam A.

c. Hubungan banyak ke satu (*many to one*)

Yaitu satu entity dalam A dihubungkan dengan maksimum satu entity dalam B. Satu entity dalam B dapat dihubungkan dengan sejumlah entity dalam A.

d. Hubungan banyak ke banyak (*many to many*)

Satu entity dalam A dihubungkan dengan sejumlah entity dalam B.

2.4 Metode Analisis Sistem

2.4.1 Diagram Konteks

Diagram konteks menggambarkan hubungan antara sistem dengan entitas luarnya. Diagram konteks berfungsi sebagai transformasi dari satu proses yang melakukan transformasi data input menjadi data output. Entitas yang dimaksud adalah entitas yang mempunyai hubungan langsung dengan sistem.[3]

Suatu diagram konteks selalu mengandung satu dan hanya satu proses saja. Diagram konteks ini menggambarkan hubungan input atau output antara sistem dengan dunia luarnya (kesatuan luar).[3]

2.4.2 *Data Flow Diagram* (DFD)

Data Flow Diagram (DFD–DAD/Diagram Alir Data) memperlihatkan hubungan fungsional dari nilai yang dihitung oleh sistem, termasuk nilai masukan, nilai keluaran, serta tempat penyimpanan internal. DFD adalah gambaran grafis yang memperlihatkan aliran data dari sumbernya dalam objek kemudian melewati proses yang mentransformasinya ke tujuan yang lain, yang ada pada objek lain. DFD sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertimbangan lingkungan fisik dimana data tersebut mengalir.[3]

DFD merupakan alat yang digunakan pada metodologi pengembangan sistem yang terstruktur (structured analysis and design). DFD merupakan alat yang cukup

populer sekarang ini, karena dapat menggambarkan arus data di dalam sistem dengan terstruktur jelas.[3]

Untuk memudahkan pembacaan DFD, maka penggambaran DFD dibantu dengan beberapa simbol diantaranya :

1. Kesatuan Luar (*External Entity*)

Setiap sistem pasti mempunyai batas sistem yang memisahkan suatu sistem dengan lingkungan luarnya. Namun sistem akan menerima input dan menghasilkan output kepada lingkungan luarnya.

2. Arus data (*Data Flow*)

Suatu data flow atau alur data dapat dipresentasikan dengan anak panah yang menunjukan arah menuju kedalam dan keluar dari suatu proses. Aliran daya ini digunakan untuk menerangkan perpindahan data atau paket data atau informasi dari suatu bagian sistem ke bagian lainnya aliran data dapat disimbolkan dengan panah.

3. Proses (*Process*)

Proses adalah kegiatan atau kerja yang dilakukan oleh orang, mesin, atau komponen dari hasil suatu atau data yang masuk ke dalam proses untuk dihasilkan atau data yang akan keluar dari proses.

4. Penyimpanan data (*Data Store*)

Penyimpanan data (*data store*) merupakan simpanan dari atau yang dapat berupa sebagai berikut ini :

a. Suatu file / database di dalam sistem computer

b. Suatu arsip atau catatan manual

c. Suatu agenda atau buku

2.4.3 Kamus Data

Kamus data adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi. Dengan menggunakan kamus data, analisis sistem dapat mendefinisikan data yang mengalir di sistem dengan lengkap. Kamus data dibuat pada tahap analisis sistem dan digunakan baik pada tahap analisis maupun pada tahap perancangan sistem. [3]

Pada tahap analisis dan perancangan, kamus data dapat digunakan sebagai alat komunikasi antara analsisis sistem dengan pemakai sistem tentang data yang mengalir di sistem, yaitu tentang data yang masuk ke sistem dana tentang informasi yang dibutuhkan oleh pemakai sistem.[3]

Pada tahap perancangan, kamus data digunakan untuk merancang input, merancang laporan-laporan dan database. Kamus data dibuat berdasarkan arus data yang ada di DFD. Arus data di DFD sifatnya global, hanya ditunjukkan nama arus

datanya saja. Keterangan lebih lanjut tentang struktur data dari arus data di DFD secara lebih rinci dapat dilihat di kamus data. Kamus data harus dapat mencerminkan keterangan yang jelas tentang data yang dicatatnya, maka kamus data harus memuat nama arus data, alias, bentuk data, arus data, penjelasan, periode, volume, dan struktur data.[3]

2.5 Borland Delphi 7.0

Delphi merupakan salah satu bahasa pemrograman tingkat tinggi berbasis Windows. Delphi digolongkan ke dalam bahasa pemrograman visual yang menitik beratkan pada pemrograman berorientasi objek (*object oriented programming*). Membuat program menggunakan Delphi tidaklah terlalu sulit, karena bahasa pemrograman ini dikembangkan menggunakan bahasa pascal. Bagi yang sudah terbiasa dengan bahasa pemrograman pascal, akan merasa lebih mudah dalam mempelajari Delphi. Karena bentuk penulisan *listing* programnya tidak memiliki perbedaan yang terlalu jauh. Dalam perkembangannya, Delphi memiliki beberapa objek-objek yang canggih, dalam bentuk kontrol program yang dikelompokkan ke dalam *toolbox* yang biasa disebut dengan *component palette*.[5]

Apabila dibandingkan dengan bahasa pemrograman lain yang berbasis visual, Delphi memiliki kelebihan yang terletak pada ketersediaan berbagai macam kontrol program yang lebih banyak dan lebih canggih. Walaupun sedikit mirip dengan

kontrol program yang dimiliki Visual Basic, Delphi memiliki keunggulan dalam penulisan *listing* program yang lenih canggih dan serba otomatis.[5]

Menggunakan bahasa pemrograman Delphi kita dapat menghasilkan program-program yang canggih, dimulai dari pemrograman multimedia, grafis sampai dengan pemrograman *database* yang menggunakan jaringan. Bahkan dengan Delphi kita juga bisa membuat program yang dapat mengakses data dari internet.[5]

2.6 SQL Server

SQL Server adalah bahasa komputer standar ANSI (*American National Standard Institute*) untuk mengakses dan memanipulasi sistem basisdata. Pernyataan SQL digunakan untuk mengambil dan meng-update data dalam basisdata.

SQL adalah kependekan *Structured Query Language* yang merupakan bahasa basisdata standar industri. SQL menyediakan Data Definition Language (DDL) dan Data Manipulation Language (DML). Walaupun dalam beberapa bagian saling berkaitan, perintah DDL memungkinkan dapat membuat dan mendefinisikan tabel (*CREATE TABLE*), dan indeks (*CREATE INDEKS*). Sebaliknya, perintah DML memungkinkan dapat memanipulasi data dan membangun query untuk mengambil data (*SELECT*) bebrapa tabel, menyisipkan (*INSERT*) data baru, memperbaiki (*UPDATE*) data, dan menghapus (*DELETE*) data.

SQL (*Structured Query Language*) pertama kali dikembangkan pada akhir tahun 1970-an di *Laboratorium IBM San Jose*, California. SQL umumnya dibaca

"*sequel*" dan pada mulanya, dikembangkan untuk produk DB2 yang dimiliki oleh IBM. SQL adalah bahasa nonprosedural kontras dengan bahasa generasi ketiga (*Generation Language* / 3GL).

2.7 Konsep Dasar *Client/Server*

Sistem *Client* / *server* mempunyai dua komponen utama, yaitu komputer *client* dan komputer *server*. *Server* merupakan komputer induk yang melakukan pemrosesan terbanyak untuk memenuhi permintan-permintaan dari komputer *client* dan bertindak sebagai *sever database* yang menyimpan data. *Client* yaitu komputer / w*orkstation* yang melakukan pengiriman permintaan-permintaan data pada *server* kemudian menampilkan data tersebut pada *interface* aplikasi yang dimilikinya.

Sistem *client* / *server* merupakan suatu sistem komputer yang melibatkan proses-proses *client* yang meminta suatu pelayanan data kepada komputer *server* yang menyediakan layanan data tersebut, sehingga baik *client* maupun *server* sama-sama melakukan suatu pekerjaan. Dengan adanya kombinasi *client* dan *server* ini maka kumpulan dari modul-modul program tidak dieksekusi dalam memori yang sama namun terbagi dalam komputer *client* dan *server*. Dengan arsitektur seperti ini, modul yang memanggil menjadi *client* yang meminta suatu layanan data dan modul yang dipanggil mencari *server* yang menyediakan layanan tersebut.

Dari jenis-jenis jaringan yang ada seperti (LAN, WAN, MAN, GAN) yang akan dijelaskan dalam oleh penulis adalah jenis LAN (*Local Area Network*), karena LAN merupakan jaringan yang akan penulis pakai.

2.7.1 Pengertian LAN *(Local Area Network)*

Local Area Network merupakan salah satu arsitektur jaringan yang paling sederhana dan dapat dikembangkan menjadi arsitektur jaringan yang lebih luas cakupannya. Luas cakupan LAN itu sendiri tidak melebihi dari satu area yang terdiri dari beberapa terminal yang saling dihubungkan sehingga menambahkan fungsi dari terminal itu sendiri layanan-layanan yang dapat diberikan LAN adalah penggunaan file bersama (*file SHAring)* atau penggunaan printer bersama *(printer SHAring*).

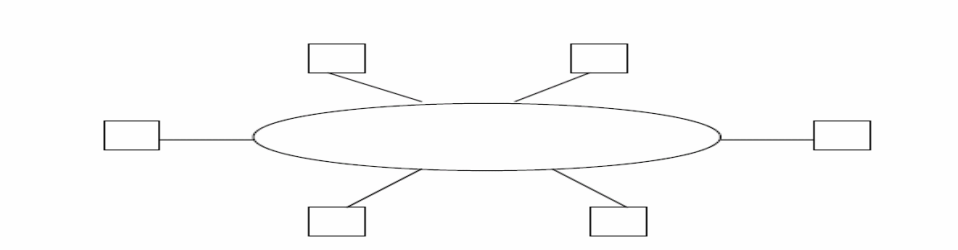
Biasanya LAN menggunakan satu *server* untuk melayani kebutuhan *client*-nya, tetapi tidak menutup kemungkinan untuk menggunakan lebih dari satu *server*, tergantung kebutuhan dari *client* itu sendiri. Biasanya yang menjadi pertimbangan adalah jenis layanan yang dibutuhkan dan performansi jaringan itu sendiri.

2.7.2 Topologi Jaringan

Apabila dilihat dari jenis hubungannya, maka topologi jaringan dapat dibagi menjadi tiga, yaitu:

1. Topologi cincin (*ring topology*)

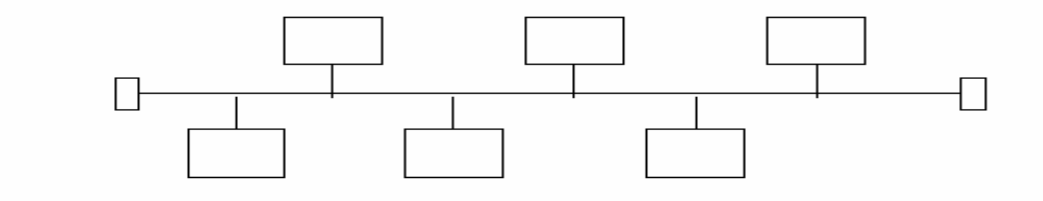
Untuk membentuk jaringan cincin, setiap sentral harus dihubungkan seri satu dengan yang lain dan hubungan ini akan membentuk loop tertutup. Dalam sistem ini setiap sentral harus dirancang agar dapat berinteraksi dengan sentral yang berdekatan maupun berjauhan. Dengan demikian kemampuan melakukan switching ke berbagai arah sentral. Keuntungan dari topologi jaringan ini antara lain : tingkat kerumitan jaringan rendah (sederhana), juga bila ada gangguan atau kerusakan pada suatu sentral maka aliran trafik dapat dilewatkan pada arah lain dalam sistem.



Gambar 2.1 ***Topologi Jaringan***

2. Topologi bus (*bus topology*)

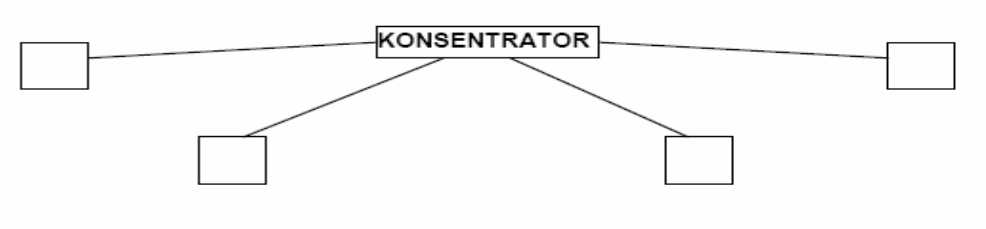
Topologi bus berupa komputer-komputer yang dihubungkan melalui satu jalur kabel. Kelemahan topologi bus adalah bila pada sutau titik jaringan mengalami kerusakan, seluruh jaringan akan mati. Implementasi topologi ini menggunakan kabel koaksial dan membutuhkan sedikitnya duah buah alat yang disebut terminator pada ujung-ujung kabel. Terminator berfungsi untuk memberikan hambatan sebesar 50 *ohm* antara kabel inti dengan serabut kawat.



Gambar 2.2 ***Topologi Bus***

3. Topologi bintang (*star topology*)

Implementasi topologi star memerlukan hardware tambahan, yaitu konsentrator berupa switch atau hub yang berfungsi untuk mengatur lalu lintas data. Topologi ini lebih tahan terhadap gangguan dibandingkan topologi bus karena kerusakan di salah satu titik tidak akan mematikan seluruh jaringan, tetapi membutuhkan biaya lebih karena membutuhkan konsentrator. Dalam topologi *star* digunakan kabel *twisted pairs*.



Gambar 2.3 ***Topologi Bintang (Star)***