BAB II

2.1 Jaringan Komputer

Jaringan komputer adalah suatu sistem yang terbentuk dari interkoneksi fasilitas-fasilitas yang dirancang untuk membawa *traffik* dari beragam sumber telekomunikasi. Jaringan terdiri dari *link* dan *node*. Istilah *link* digunakan untuk merepresentasikan kabel, peralatan terminasi, dsb. Istilah *node* digunakan untuk merepresentasikan sentral, junction atau keduanya. Sedangkan *traffik* adalah informasi yang terdapat didalam jaringan yang mengalir melalui *node* dan *link.*

Jaringan komputer dibedakan berdasarkan luas jangkauannya, antara lain sebagai berikut :

1. Local Area Network (LAN)
2. Metropolitan Area Network (MAN)
3. Wide Area Network (WAN)

Penulis hanya membahas tentang LAN karena luang lingkupnya lokal.

2.1.1 Local Area Network (LAN)

Local Area Network adalah suatu solusi terhadadp keterbatasan Kemampuan stand alone dan cara pemrosesan terpusat. LAN merupakan jaringan *peer*, yang berarti setiap peralatan pada jaringan dapat berkomunikasi satu sama lainnya. Sebagai pengganti terminal dumb, LAN menggunakn terminal smart yaitu komputer mikro dengan unit CPU tersendiri. LAN berfungsi sebagai sebuah penghubung (bridge) tidak hanya orang-orang dengan informasi, tetapi juga antar

pemakai. LAN menyediakan alat efektif untuk berkomunikasi melalui elektronik mail dan melalui beberapa software komunikasi lain. Penggunaan sekelompok software aplikasi mengurangi kebutuhan untuk mengadakan rapat tatap muka secara langsung dan metode penyebaran informasi yang memerlukan waktu lama. Pada saat yang bersamaan jaringan memungkinkan peningkatan interaksi antar pegawai dari workstationnnya masing-masing. Jaringan juga dapat menaikan efektivitas komunikasi. Keuntungan utama dari local area network ialah software dan data jauh lebih mudah dipelihara dan diproteksi daripada dalam keadaan stand alone. Data penting dapat di back up setiap hari atau lebih bila perlu setiap jam. Bila software memerlukan perbaikan, proses dapat dilakukan pada salah satu sumber (resource), tidak pada setiap komputer personal yang menjalankan program tersebut.

Untuk jaringan sendiri ada beberapa komponen dasar yang sangat diperlukan untuk sebuah sistem jaringan yaitu :

1) Server

   Server adalah inti dari sebuah local area network, komputer ini biasanya berupa komputer mikro yang biasanya mempunyai kecepatan tinggi, dan bertugas menjalankan sistem operasi dan mengelola aliran data dalam jaringan.

2) Workstation

   Workstation adalah komputer yang dihubungkan dalam jaringan, yang dapat menggunakan fasilitas yang disediakan oleh server seperti CD-ROOM, floppy disk dan lain sebagainya.

3) Tranceiver

   Tranceiver adalah suatu alat media transmisi biasanya digunakan dalam hubungan komputer dengan kabel.

4) Network Interface Card

   Network Interface Card (NIC) adalah suatu perangkat keras atau network controller yang fungsinya sebagai mesin protocol dalam penghubungan setiap komputer , dalam NIC ini mempunyai port untuk tempat menghubungkan kabel jaringan.

5) Komponen Software untuk Sistem Operasi

6) Setiap komputer workstation bekerja dengan sistem operasinya sendiri misalnya DOS, DOS dengan Windows, Windows NT dan masih banyak lagi sistem operasi yang dapat digunakan untuk sistem jaringan.

2.1.2 TCP/IP (Transmission Control Protocol)

TCP/IP adalah salah satu protocol yang memungkinkan kumpulan komputer untuk berkomunikasi dan bertukar data didalam suatu network (jaringan). TCP/IP merupakan protocol yang telah diterapkan pada hampir semua perangkat keras dan sistem operasi. TCP/IP adalah arsitektur standard yang banyak dipergunakan dan OSI telah menjadi model standard untuk mengklasifikasikan fungsi-fungsi komunikasi.

2.1.3 Topologi Jaringan

Topologi adalah suatu struktur atau cara bagaimana simpul dan pusat di dalam jaringan tersebut saling dihubungkan. Hubungan tersebut sangat tergantung dari jenis aplikasinya, tiap struktur mempunyai keuntungan dan kerugiannya masing-masiing.

Tipe-tipe utama dari topologi fisik yang digunakan dalam jaringan ada tiga yaitu sebagai berikut :

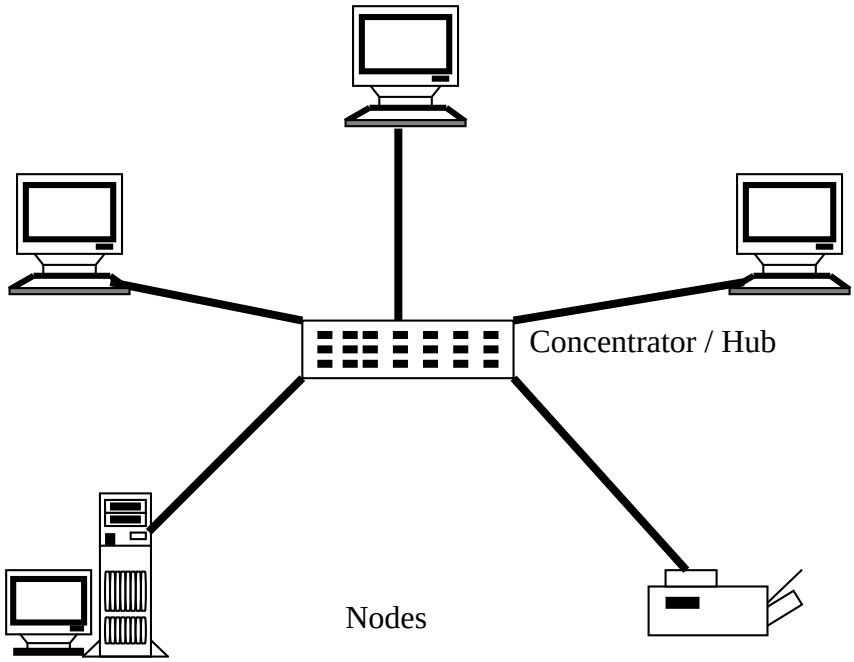
1) Topologi Ring
2) Topologi Linear Bus
3) Topologi Star

Dalam hal ini penulis hanya akan membahas tentang topologi star karena topologi yang digunakan di SMP kartika XIII – 1 adalah topologi star.

2.1.3.1 Topologi Star

Topologi model ini didesain dimana setiap node (File Server, workstasion, dan perangkat lainnya) terkoneksi ke jaringan melewati sebuah hub atau concentrator. Hub ini bisa aktif dan bisa pasif. Aktif Hub menggenerasi sinyal elektrik dan mengirimnya ke semua komputer yang terhubung atau disebut hub multiport repeater. Sedangkan pasif hub sama seperti panel pemasangan kabel atau blok punch-down, aksi belaka sebagai hubungan point dan tidak mengnuatkan atau generasi sinyal. Data yang terkirim ke jaringan akan melewati hub/concentrator sebelum melanjutkan ke tempat tujuannya. Hub/concentrator akan mengatur dan mengontrol keseluruhan fungsi jaringan. Dia juga bertindak sebagai repeater/penguat aliran data. Konfigurasi pada jaringan model ini menggunakan kabel twisted pair, dan dapat digunakan bersama kabel koaksial

atau kabel fiber optic. Hybrid star network adalah network yang satu dihubungkan dengan network star lainnya.



Kelebihannya yaitu :

1) Mudah dimodifikasi dan menambahan komputer baru untuk star network.
2) Tidak mengakibatkan gangguan pada jaringan ketika akan memasang atau memindahkan perangkat jaringan lainnya.
3) Mudah untuk mendeteksi kesalahan dan memindahkan perangkat lainnya.
4) Keamanan tinggi
5) Kerusakan satu komputer tidak membawa down pada kebutuhan network
6) Bisa menggunakan beberapa tipe kabel dalam beberapa network dengan hub yang bisa mengakomodasi tipe-tipe kabel.

Kelemahannya :

1) Membutuhkan lebih banyak kabel daripada topologi jaringan garis lurus.
2) Beberapa network star membutuhkan peralatan pada pusat point untuk broadcast kembali atau switch trafik network.
3) Membutuhkan hub atau concentrator, dan bilamana hub atau concentrator itu rusak maka node-node yang terkoneksi akan sulit terdeteksi.

Biaya relatif lebih mahal daripada topologi linear bus karena adanya tambahan biaya untuk pengadaan hub atau concentrator. Selain itu biaya untuk kabel network star lebih tinggi.

2.2 Rekayasa Perangkal Lunak

Rekayasa Perangkat Lunak (RPL, atau dalam bahasa inggris *Software Engineering* atau SE) adalah satu bidang profesi yang mendalami cara-cara pengembangan perangkat lunak termasuk pembuatan, pemeliharaan, management organisasi pengembangan software dan sebagainya.

2.2.1 Unified Modeling Language (UML)

Unified Modeling Language merupakan paradigma baru dalam suatu Pemodelan aplikasi atau sistem analis yang akan dikembangkan. UML juga merupakan jembatan antara sistem analis dengan pemrogram, dimana pemrogram cukup melihat model UML yang akan dijadikan dasar pemrograman sehingga lebih memudahkan untuk melakukan pengkodean atau penulisan program karena sudah terdapat prosedur tersebut.

Unified Modeling Language (UML) adalah bahasa pemodelan untuk menspesifikasikan, visualisasi, perancangan dan dokumentasi terhadap bagian sistem yang sedang berjalan (*current sistem*) atau sistem yang lama (**J.W. Schmidt, TU Hamburg-Hamburg, 2000**). Namun UML bukan merupakan :

1) Bahasa pemrograman visual, tetapi bahasa pemodelan dari sistem yang akan diambil secara visual atau secara nyata ditampilkan dan diperjelas.
2) Alat untuk menspesifikasikan, namun UML merupakan bahasa pemodelan dari sistem secara spesifik atau lebih detail terhadap penggambaran sistem tersebut. Dan hasil akhir ini pemodelan dengan UML akan lebih spesifik dan lebih jelas dalam proses pemrograman lebih lanjut.
3) Suatu proses, namun UML menggambarkan atau menampilkan suatu proses dari sistem yang sedang berjalan (*current sistem*).

Unified Modeling Language (UML) merupakan suatu model baru yang diterapkan untuk menstandarisasi dalam tahap Software Engineering atau Rekayasa Perangkat Lunak. Pada awal tahun 1994 Grady Booch dan Rumbaugh bekerja sama untuk membuat suatu model yang modern yaitu UML.

Perancangan sistem dengan metode *Unified Modeling Language* (UML), dalam pengembangannya dapat digunakan untuk berbagai gambaran perancangan sistem antara lain :

1) Menampilkan batasan sistem, memperlihatkan fungsi-fungsi dari proses
2) menggunakan Aktor dan Use-Case.
3) Mengilustrasikan interaksi proses dengan menggunakan diagram interaksi.

4) Mempresentasikan dan menjabarkan struktur statik dari sebuah sistem dengan menggunakan diagram kelas.
5) Sikap dan perilaku dari model objek dengan menggunakan diagram transisi.
6) Mengembangkan fungsi dengan menggunakan *stereotypes.*
7) Mengimplentasikan fisik dengan menggunakan *component* dan *deployment diagram.*

Didalam proses development atau pengembangan suatu sistem dengan pemodelan *Unified Modeling Language* dilakukan dalam beberapa tahap :

1) Inception

Tahap permulaan ini digunakan untuk dasar pemikiran dari keseimbangan bisnis suatu project yang akan dikembangkan dan untuk penentuan suatu project dan bidang dari project tersebut. Tahap ini juga akan melakukan penentuan ***Goals*** dan estimasi keuntungan dan kerugian dari hasil peranacangan software yang akan dibangun.

2) Elaboration

Proses perluasan ini adalah tahap yang akan dilakukan untuk mengumpulkan secara lebih rinci dari kebutuhan yang akan dikembangkan, yang harus dilakukan pada tahap ini adalah tahap analisis atau tahap high-level dan melakukan desain untuk keseimbangan dari basis atau dasar arsitektur, dan pembentukan rencana pembangunan atau development suatu sistem yang akan dirancang, dan akan melakukan pemahaman masalah yang dihadapi antara lain menetapkan apa yang akan dibuat, bagaimana cara pembuatan serta teknologi yang akan digunakan berkaitan dengan sistem yang akan dirancang.

Dalam tahap elaboration ini dilakukan secara matang dari awal hingga akhir pengembangan, serta memperhatikan beberapa kemungkinan resiko yang terjadi dalam proyek tersebut.

3) Construction

Tahap pengembangan ini sudah dalam kondisi yang sebenarnya bukan untuk percobaan dan peningkatan dari proses. Setiap interaksi dalam tahap ini bertujuan untuk membangun produksi prototipe dan perangkat lunak yang berkualitas, pada tahap ini dilakukan pengetesan dan mengintegrasikan dari bagian-bagian kebutuhan sumber daya kedalam suatu proyek yang akan dirancang.

4) Transition

Transisi adalah peralihan dari sistem yang dulu kedalam sistem yang dirancang yang mengandung kegiatan antara lain ***Beta Testing***, hasil perbaikan dari testing awal sampai ketahap training user terhadap software yang telah dibuat.

Pada tahap elaboration atau perluasan dari sistem maka dilakukan proses Analisa dari berbagai keinginan dan tujuan yang akan dihasilkan dari proses pengembangan software tersebut, dalam pengembangannya harus lebih memahami dari inti permasalahan, tujuan pengembangan sistem dan berbagai kebutuhan yang akan digunakan dalam pengembangan sistem tersebut.

Dalam proses pengembangannya *Unified Modeling Language* akan melakukan beberapa proses lain : ***Inception, Elaboration, Construction, Transition***. Dari tahap-tahap tersebut secara garis besar adalah melakukan proses

pembuatan Use-Case dari sistem yang diambil, kemudian dari Use-Case tersebut akan dilakukan proses iterasi yang didalamnya terdapat beberapa tahap antara lain : ***Elaboration, Construction, Transition.*** Dari masing-masing tahap tersebut terdapat beberapa proses yang dilakukan antara lain : ***Analisys, design, coding, debugging, demonstration, integration.***

2.2.1.1 Diagram Use Case

Diagram ini berfungsi memperlihatkan susunan dari perilaku sebuah sistem, dapat juga diartikan sebagai penggunaan kasus. Sebuah use-case adalah sebuah rangkaian dari transaksi yang berhubungan dan saling berinteraksi antara aktor dengan sistem.

Use case mewakili kumpulan skenario typical yang membantu untuk menyusun, menghubungkan dan mengerti tentang kebutuhan-kebutuhan terpenting dari suatu sistem atau kasus tersebut.

1) Suatu use case merupakan hubungan dari kumpulan interaksi typical antara pengguna dan sistem komputer.
2) Suatu Use Case menangkap beberapa aksi pada tingkatan aplikasi yang dilakukan.
3) Suatu use case mencapai hasil atau tujuan tersendiri pada pengguna.
4) Use case dipakai sebagai dasar untuk pengujian apakah sistem telah dapat melakukan apa yang dibutuhkan pengguna.

Didalam use case terdapat dua komponen yaitu Aktor dan Use Casenya sendiri, disertai dengan tambahan penjelasan tentang use case yang berupa skenario typical dari sebuah use case.

1) Aktor

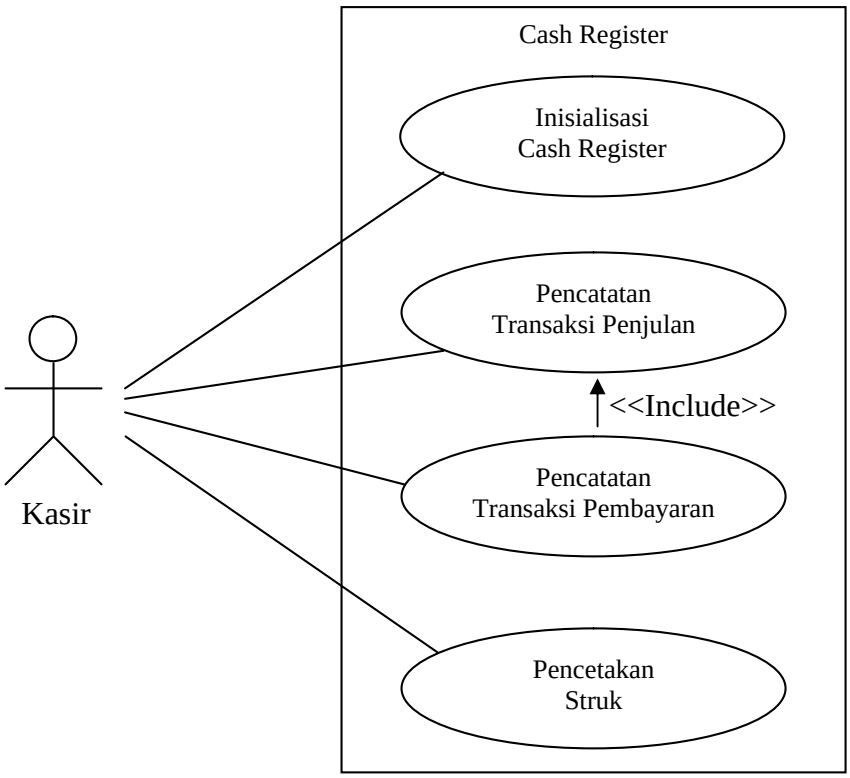
Suatu aktor adalah seseorang atau sebuah sistem lain yang berinteraksi dengan sistem tersebut. Aktor bisa merupakan suatu peran yang diramaikan oleh seseorang atau sistem lain terhadap sistem yang dikembangkan.

2) Use-case

Use case berasal dari kata kasus yang mewakili deskripsi tingkat tinggi dari suatu aksi yng diinginkan dimana suatu aktor terlibat, dan juga dapat berisi informasi yang akan dicapai untuk membentuk beberapa diagram. Use case adalah tentang kebutuhan fungsional dilihat dari luar sistem.

Atribut <<uses>> dan <<extend>>

1) <<uses>> merupakan relasi uses (menggunakan), dibutuhkan saat ada sekelompok perilaku yang mirip dari beberapa use case, atau melakukan hal yang sama pada dua atau lebih use case. Uses digunakan jika ingin memisahkan pengulangan pada beberapa use case.
2) <<extend>> pengembangan dipakai untuk menghindari pemodelan setiap kasus ekstra dengan sebuah use case, dan memperlihatkan perilaku dari target use case yang ditambahkan kesumber use case. Extend digunakan jika ingin menjelaskan variasi dari perilaku normal.



Gambar 2.1 Contoh Use case diagram

2.2.1.2 Diagram Kelas (Class Diagram)

Diagram kelas melukiskan tipe dari suatu objek yang terdapat didalam sistem yang saling berhubungan, hubungan tersebut akan tetap ada didalamnya. Diagram kelas memperlihatkan hubungan antar kelas dan penjelasan detail tiap-tiap model desain dari suatu sistem.

Diagram kelas juga akan menampilkan suatu atribut dan operasi dari suatu kelas yang dipakai sebagai jalan suatu objek untuk saling berhubungan.

Model UML pada diagram kelas (Class Diagram).

1) Kelas-kelas beserta struktur dan perilakunya.
2) Asosiasi, agregasi, depedensi, dan inheritansi.

3) Multiplicity (keserba-ragaman) dan indikator navigasi.
4) Nama aturan (rule name).

Kelas (Class) :

1) Sebuah kelas adalah sekumpulan objek dengan struktur, perilaku, hubungan, dan semantik biasa.
2) Kelas dapat ditemukan pada saat mempelajari diagram sekuen dan diagram kolaborasi.
3) Kelas digambarkan dengan sebuah kotak dengan 3 bagian.
4) Kelas harus diberi nama dengan menggunakan kumpulan dari domain. Penamaan pada kelas harus dibuat standar, misalnya denga nama kata benda tunggal yang diawali dengan huruf besar.

Atribut :

Atribut dapat diartikan sebagai sebuah struktur dari sebuah kelas yang direpresentasikan oleh atribut. Dan ini dapat ditemukan dengan mempelajari dari definisi kelas, kebutuhan masalah, dan oleh pengetahuan domain. Atribut dapat diartikan pula sebagai objek yang dapat diceritakan berdasarkan nama atribut tersebut, juga dapat disebut sebagai variabel atau field.

Operation :

Operasi adalah perilaku dari sebuah kelas yang direpresentasikan oleh operasi, dan dapat ditemukan pada diagram interaksi.

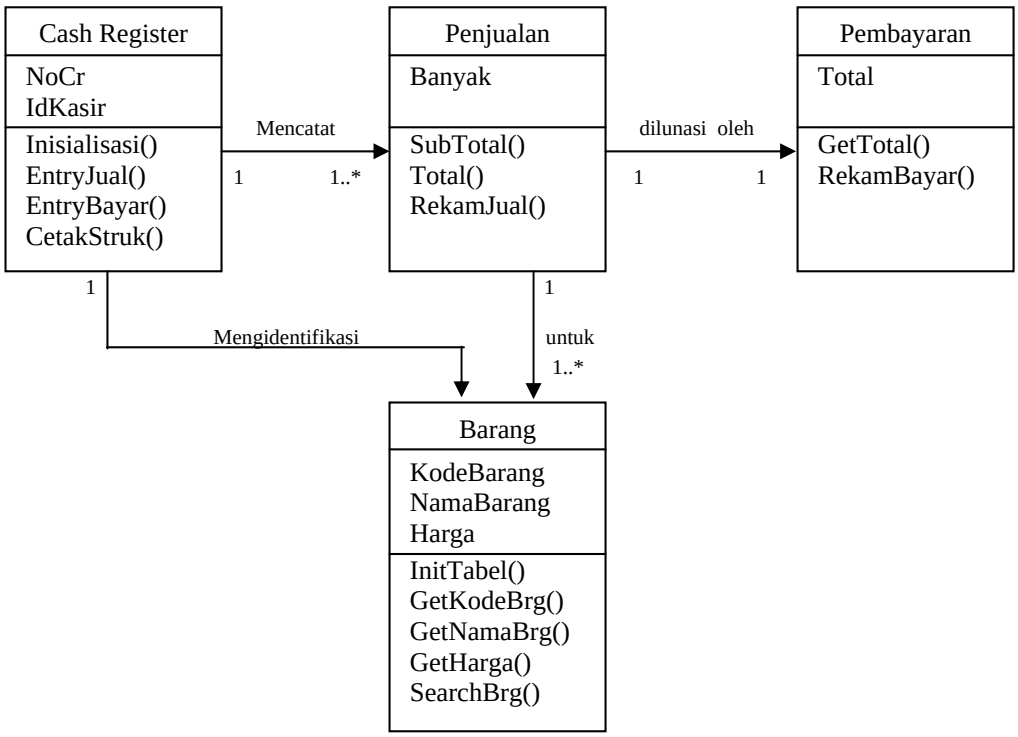
Relasi :

Relasi adalah hubungan komunikasi antar objek. Pada diagram sekuen dan atau diagram kolaborasi, dipelajari hubungan apa yang dibutuhkan antar objek. Jika objek tersebut membutuhkan komunikasi maka diperlukan komunikasi ini.

Multiplicity dan Navigasi :

Multiplicity mendefinisikan berapa banyak objek yang berhubungan dengan sebuah relasi. Untuk setiap asosiasi dan agregasi ada dua keputusan Multiplicity yang diambil yaitu satu untuk setiap relasi.

Meskipun asosiasi dan agregasi adalah hubungan bolak-balilk secara defaultnya, sering kali diperlukan juga untuk membatasi navigasi tersebut ke satu arah. Jika navigasi dibatasi, tambahkan tanda anak panah dari navigasi tersebut ke arah kelas yang lainnya.

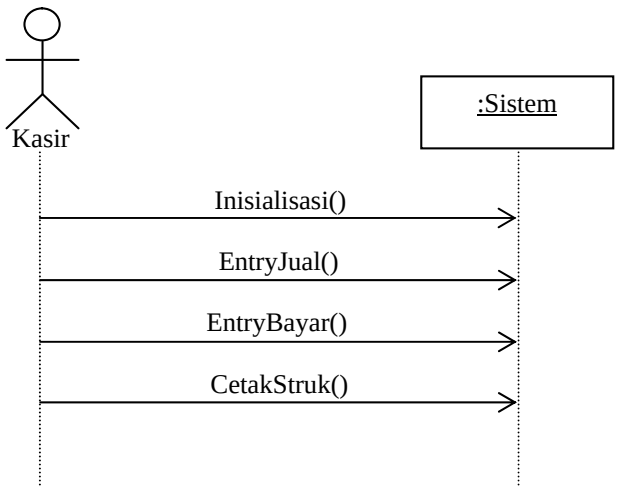


Gambar 2.2 Contoh Diagram Kelas

2.2.1.3 Diagram Interaksi (Interaction Diagram)

Diagram interaksi memperlihatkan bagaimana use case – use case Merealisasikan interaksi antar objek tersebut. Diagram interaksi merupakan penggambaran tentang bagaimana tingkah laku objek yang saling berinteraksi, diagram interaksi dapat ditangkap dari tingkah laku atau kegiatan dari salah satu use case. Ada dua tipe dari diagram interaksi, yaitu :

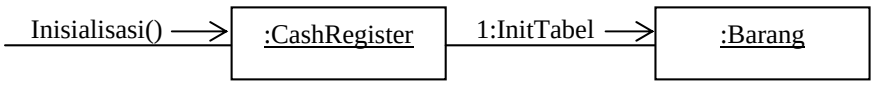
1) Diagram sekuen (Sequence Diagram) : Memperlihatkan interaksi dari objek yang diatur berdasarkan waktu dan menekankan pada perintah atau persetujuan dari interaksi tersebut.



Gambar 2.3 Contoh Diagram Sekuens

2) Diagram kolaborasi (Collaboration Diagram) : Memperlihatkan dimana objek berinteraksi dari satu ke yang lainnya dan menekankan pada interaksi dari objek tersebut.

Perbedaan antara diagram sekuen dengan diagram diagram kolaborasi yaitu : Diagaram kolaborasi lebih menekankan kepada struktur sedangkan diagram sekuen akan menitik beratkan pada interaksi tersebut.



Gambar 2.4 Contoh Diagram Kolaborasi

2.2.1.4 Diagram Status (State Diagram)

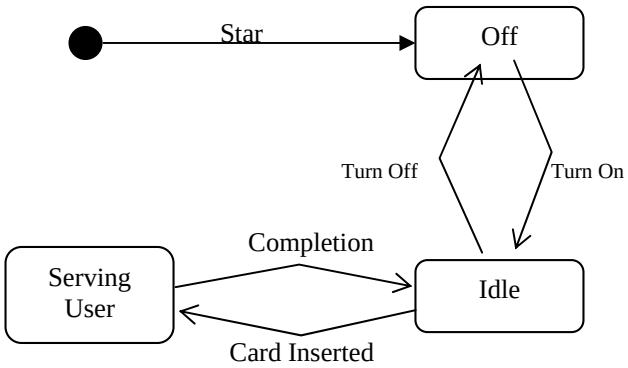
Diagram status akan memperlihatkan beberapa kejadian, antara lain :

1) Memperlihatkan kehidupan dari kelas tersebut.
2) Kejadian yang dapat menjadikan perubahan transisi dan status dari satu ke yang lainnya.
3) Aksi yang dihasilkan dari perubahan transisi tersebut.

Ada beberapa tahap yang harus dibuat dalam penggambaran diagram transisi tersebut, antara lain :

1) **State :** Akan saling berhubungan satu dengan yang lainnya dan diawali dengan initial state kemudian hubungan akan diakhiri dengan final state.
2) **Event :** Menceritakan suatu kejadian dari suatu atribut dan kegiatan yang sudah dideklarasikan pada diagram kelas.
3) **Transition :** Menceritakan kondisi perubahan dari suatu state didalam objek dan transisi biasanya sebagai triger.

4) **Action :** Merupakan proses tindakan yang akan segera dilaksanakan, didalam action diawali dengan action yang akan dilakukan beserta argumen dari action tersebut, dan hasil dari action tersebut adalah event dan action lainnya, dan transition mungkin merupakan triger dari action.

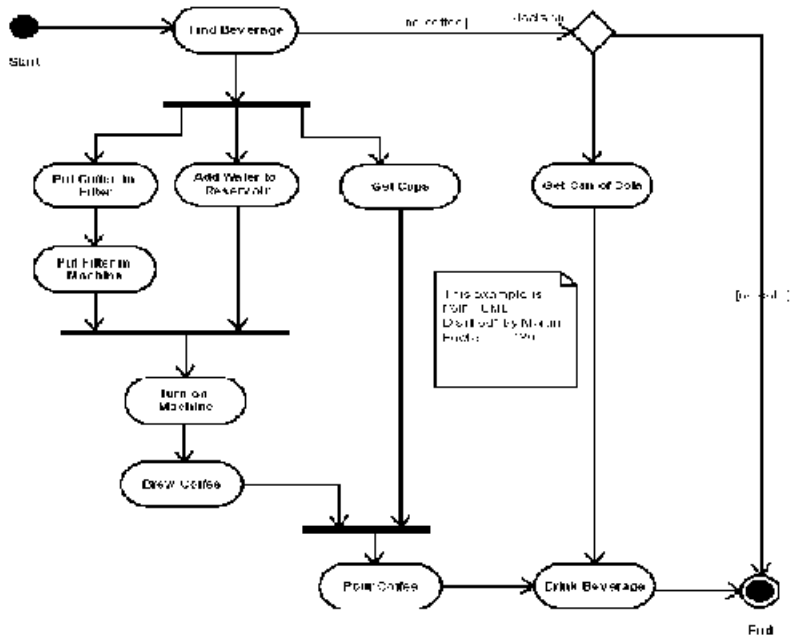


Gambar 2.5 Contoh State Diagram

2.2.1.5 Diagram Aktivitas (Activity Diagram)

Diagram aktivitas secara konseptual merupakan aktivitas atau kegiatan yang akan dilakukan pada masa mendatang, apakah dilakukan oleh manusia atau dilakukan oleh komputer, dan aktivitas tersebut digambarkan dalam bentuk box.

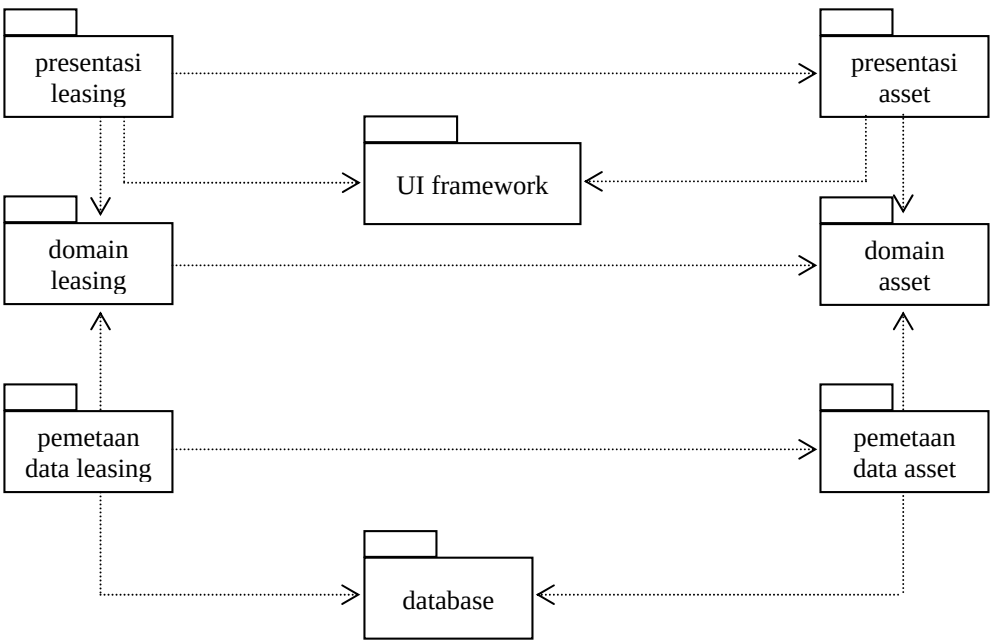
Diagram aktivitas menggambarkan perilaku yang membagi melewati beberapa use case yang menjelaskan alur kerja pada keseluruhan proses. Diagram ini tidak dapat memperlihatkan bagaimana objek berkolaborasi dan juga diagram ini tidak secara akurat bagaimana objek melewati garis batas waktu.



Gambar 2.6 Contoh Diagram Aktivitas

2.2.1.6 Diagram Paket (Package Diagram)

Diagram paket digunakan untuk mempersatukan beberapa kelas menjadi paket-paket yang saling berhubungan. Karena kelas-kelas yang terlalu banyak maka dibuatlah diagram paket ini. Dalam paket diagram dapat berisikan kelas, diagram kelas, dan paket itu sendiri.

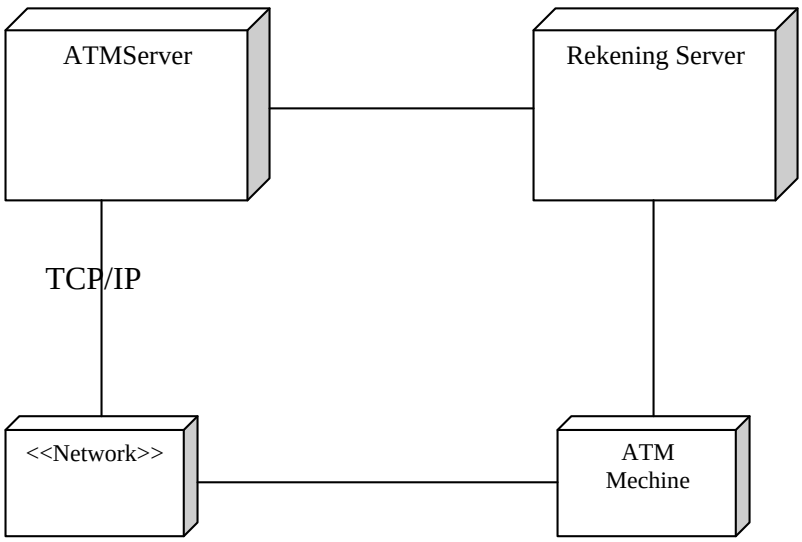


Gambar 2.7 Contoh Diagram Paket

2.2.1.7 Diagram Penempatan (Deployment Diagram)

Dalam diagram penempatan akan memperlihatkan pemetaan perangkat lunak kepada perangkat keras. Diagram ini menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, dimana komponen akan terletak (pada mesin, server atau piranti keras lainnya).

Dalam proses penggunaan suatu sistem dengan mengunakan UML tidak harus menggunakan seluruh diagram yang ada, tetapi dapat disesuaikan dengan kebutuhan pengembangan sistem.



Gambar 2.8 Deployment Diagram

2.3 Delphi 6.0

Delphi merupakan pemrograman berbasis visual yang dibuat oleh perusahaan pengembang perangkat lunak yang sangat terkenal yaitu Borland. Delphi merupakan bahasa pemrograman yang cukup canggih dan mempunyai

banyak fasilitas, antara lain fasilitas untuk mengolah angka, databasae, internet, grafik, dan animasi dan tidak hanya aplikasi yang berjalan di windows.

Kemampuan Delphi secara umum adalah menyediakan fasilitas-fasilitas dengan komponen-komponen dalam bentuk objek visual yang memungkinkan pemrogram membuat aplikasi sesuai dengan keinginannya dengan tampilan yang menarik dan kemampuan yang canggih. Komponen-komponen dalam bentuk ini merupakan komponen yang mempunyai bentuk fisik dan dapat dilihat (*visual*).

Selain dalam bentuk objek (komponen visual), Delphi juga menyediakan fasilitas dengan bahasa pemrograman. Bahasa pemrograman yang digunakan adalah bahasa pemrograman objek pascal. Bentuk dari bahasa pemrograman ini tidak berupa komponen atau objek visual melainkan kumpulan teks yang disusun dalam bentuk tertentu berdasarkan aturan atau kaidah tertentu dan mempunyai arti tertentu.

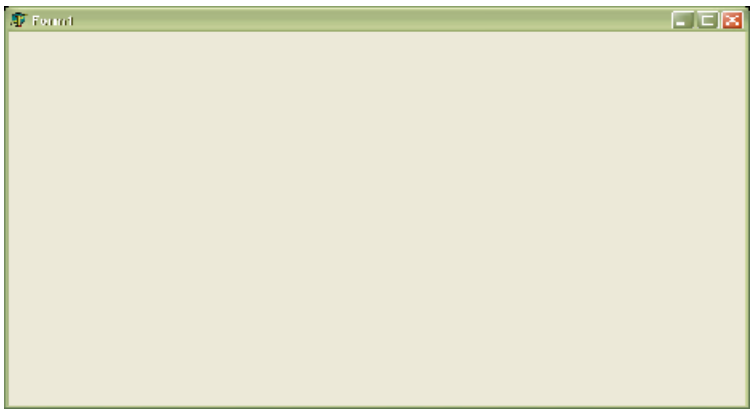
Gabungan dari objek dan bahasa pemrograman ini disebut bahasa pemrograman berorientasi objek atau OOP (*Object Oriented Programming*).

Beberapa fasilitas bahasa pemrograman Delphi antara lain :

1) Kompoonen-komponen visual (*form, button, panel, label, dll*), tersedia dengan lengkap, seahingga memudahkan pemrogram dalam menangani tampilan pada program yang dibangun.
2) Penanganan kesalahan penulisan listing program (*source code*) untuk fungsi-fungsi yang dimasukan ke dalam program dapat diketahui dengan jelas. (Apabila terjadi kesalahan penulisan listing program, Delphi menampilkan

pesan apa yang menjadi kesalahannya, dan menunjukkan baris mana pada listing program yang menjadi kesalahan penulisan).

3) Delphi menyediakan fasilitas yang dapat digunakan untuk mengakses beberapa format database, dalam hal ini misalnya komponen ADO yang dapat mengkoneksikan Delphi dengan MS Access, sehigga data yang dibuat dengan MS Access dapat diintegrasikan dengan Delphi.



Gambar 2.9 Form

2.4 Basis Data

Basis data merupakan kumpulan data yang saling berhubungan, yang menyimpan informasi. Tujuan menggunakan basis data yang paling penting adalah untuk mengolah data secara teratur. Beberapa keuntungan yang dapat diperoleh dengan digunakannya basis data pada pembangunan suatu sistem antara lain :

1) Pengawasan terpusat

Sesuai dengan suatu konsep pada basis data bahwa pengelolaan data dilakukan secara terpusat sehingga pengawasan dilakukan dipusat pengelolaan.

2) Konsistensi dan terintegrasi

   Dalam penggunaan data secara bersama-sama, akan dijamin kebutuhan pemakai terpenuhi. Selain itu data harus dengan mudah digunakan dan digabungkan dengan data lain untuk memperoleh data yang lebih lengkap, tanpa mempengaruhi data asalnya.

3) Penghematan memori.

4) Pemakaian bersama (s*hared*) sesuai dengan kebutuhan.

5) Keamanan data terjamin.

2.5 Dasar Pemrograman Database Client Server

Aplikasi database client server terdiri dari dua bagian, yaitu server dan client. Server, yang sering disebut *back-end*, akan berhubungan dengan media penyimpanan data/storage device. Tugasnya adalah mengatur agar semua data dapat tersimpan dengan benar pada media penyimpanannya. Server menyediakan fungsi-fungsi untuk melakukan proses inquiry maupun manipulasi data. Berhubung serever ini berhubungan erat dengan data, maka banyak orang menyebutnya sebagai *database server*. Bagian lain dari aplikasi database client server adalah client. Client yang sering disebut sebagai *front-end*, berinteraksi dengan para pemakai aplikasi. Fungsi dari client ini sangatlah bervariasi perlu disesuaikan dengan keperluan para pemakainya. Dalam melaksanakan fungsinya, client akan membuat hubungan ke database server dan memanggil fungsi-fungsi yang tersedia pada server.

2.5.1 Interbase

interbase adalah suatu database management system (DBMS). database pada interbase dapat menyimpan banyak tabel sekaligus dalam satu database berbeda dengan database paradok dimana satu database (.db) hanya berupa satu tabel saja. database dengan interbase dapat digunakan pada pc stand alone maupun berbasis server. jadi aplikasi database dapat digunakan oleh banayk pc dengan satu pc sebagai server tempat penyimpanan database.

2.5.2 Interbase Express

Delphi memberikan komponen-komponen untuk mengakses database interbase. Komponen-komponen tersebut dinamakan interbase express (disingkat dengan IBX). Interbase express bekerja lebih cepat dan lebih baik, serta memberi fasilitas pengontrolan khusus pada interbase.

1) IBDatabase, dipakai utuk berhubungan dengan database, yang dapat menangani satu atau beberapa transaksi. Tidak seperti BDE, IBX memakai komponen transaksi yang terpisah, sehingga kita dapat memisahkan transaksi dengan hubungan database.
2) IBTransaction, dipakai untuk menangani transaksi yang terpisah dari hubungan database, keuntungannya kita dapat memanfaatkan fungsionalitas commit dua fase dari interbase (transaksi dari beberapa hubungan) dan beberapa transaksi bersamaan yang menggunakan hubungan yang sama.
3) IBTable, untuk mengeset dataset yang dapat diubah tanpa menggunakan SQL.

4) IBQuery, untuk menjalankan perintah SQL. IBQuery dapat dipakai bersama-sama dengan IBUpdateSQL untuk menghasilkan dataset yang dapat diubah.
5) IBStoredProd, untuk menjalankan stored procedure.
6) IBDataSet, untuk menghasilkan dataset yang dihasilkan oleh perintah select dan dapat diubah. Komponen ini merupakan gabungan dari IBQuery dan IBUpdateSQL.
7) IBSQLMonitor, dipakai untuk debug sistem.
8) IBEvents, menerima event-event yang di post oleh server.

Komponen-komponen tersebut memberikan pengontrolan yang lebih baik pada database. Misalnya dengan menggunakan komponen transaksi yang terpisah, kita dapat mengelola beberapa transaksi konkuren pada satu atau beberapa database.



Gambar 2.10 Page Interbase

This document was created with Win2PDF available at http://www.daneprairie.com.
The unregistered version of Win2PDF is for evaluation or non-commercial use only.