BAB 2

LANDASAN TEORI

2.1 Cerita

Menurut kamus besar bahasa indonesia, cerita adalah tuturan yang membentangkan bagaimana terjadinya suatu hal [2]. Sebuah cerita dibangun oleh dua unsur. Unsur yang pertama adalah unsur yang membangun cerita dari dalam, unsur ini disebut unsur intrinsik. Unsur kedua adalah unsur yang membangun dari luar yang disebut unsur ekstrinsik [3]. Unsur intrinsik meliputi:

1. Tema adalah gagasan yang menjalin struktur cerita
2. Tokoh adalah pelaku di dalam cerita.
3. Penokohan adalah cara pengarang mengembangkan watak-watak tokoh di dalam cerita.
4. Alur adalah jalan cerita, alur terbagi tiga, yakni alur maju, alur sorot balik (alur mundur), dan alur gabungan (maju mundur). Alur mempunyai beberapa tahapan, yakni perkenalan, penggambaran peristiwa, konflik, puncak konflik atau klimaks, dan penyelesaian
5. Latar, merupakan tempat, waktu, dan suasana yang digunakan dalam cerita
6. Sudut pandang, merupakan kemampuan penulis dalam menempatkan posisi tokoh. Sudut pandang yang digunakan penulis dengan menggunakan sudut pandang orang pertama dan orang ketiga. Sudut pandang orang pertama dengan menggunakan aku atau saya. Dalam hal

ini, pengarang secara langsung menjadi tokoh di dalam cerita itu. sedangkan sudut pandang orang ketiga dengan menggunakan kata ganti orang, seperti dia.ia atau nama orang.

7. Amanat, merupakan pesan yang disampaikan penulis di dalam cerita.

Sedangkan unsur ekstrinsik meliputi nilai-nilai, sepertini nilai pendidikan, moral, sosial, budaya, agama, adat istiadat, dan kejiwaan pengarang.

2.2 Nabi

Nabi adalah seorang yang diberi wahyu oleh Allah dengan suatu syari'at namun tidak diperintah untuk menyampaikannya, akan tetapi mengamalkannya sendiri tanpa ada keharusan untuk menyampaikannya. Al-Qur'an menyebut beberapa orang sebagai nabi. Nabi pertama adalah Adam, sedangkan nabi sekaligus rasul terakhir ialah nabi Muhammad. Percaya kepada para nabi dan para rasul merupakan salah satu rukun iman dalam islam.

Dalam islam terdapat banyak nabi, tetapi yang harus diketahui hanya 25 nabi dan 4 di antaranya adalah penerima Kitab Suci:

1. Nabi Daud AS (Zabur)
2. Nabi Musa AS (Taurat)
3. Nabi Isa AS (Injil)
4. Nabi Muhammad SAW (Al-Quran).

2.3 Sistem

Suatu sistem pada dasarnya merupakan suatu susunan yang teratur dari kegiatan yang berhubungan satu sama lain dan prosedur-prosedur yang berkaitan yang melaksanakan dan mempermudah pelaksanaan kegiatan utama dari suatu organisasi

2.3.1 Pengertian Sistem

Sistem adalah jaringan kerja dari prosedur-prosedur yang saling berhubungan, nerkumpul, bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran tertentu [4]. Definisi sistem dapat dilihat dari dua kelompok pendekatan yaitu yang menekankan pada prosedur dan yang menekankan pada komponen atau elemennya.

1. Berdasarkan Penekanan Prosedur

Sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama untuk melakukan sesuatu kegiatan atau untuk menyelesaikan suatu sasaran yang tertentu.

2. Berdasarkan Penekanan Komponen

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

2.3.2 Elemen Sistem

Adapun dalam suatu sistem, dikenal berbagai elemen-elemen yang menyusun suatu sistem, yaitu sebagai berikut :

a. Tujuan

Dalam suatu sistem pasti memiliki suatu tujuan, mungkin hanya satu atau mungkin juga banyak. Dalam suatu tujuan inilah yang menjadi pemotivasi yang mengarahkan akan suatu sistem. Tanpa adanya suatu tujuan, maka suatu sistem menjadi tak terarah dan tak terkendali. Hal ini dapat menyebabkan tujuan yang tidak akan tercapai dengan baik.

b. Masukan

Masukan input sistem adalah segala sesuatu yang masuk ke dalam suatu sistem yang kemudian menjadi bahan yang nantinya menjadi bahan untuk diproses. Masukan sistem ini dapat berupa hal-hal yang berwujud (kenyataan) ataupun yang tidak tampak (khayalan). Adapun beberapa contoh, seperti masukan sistem yang berjalan yang berupa bahan mentah yang berwujud. Sedangkan masukan sistem yang tidak tampak atau tidak berwujud adalah sebuah informasi, dikarnakan sebuah informasi tidak berwujud atau tidak tampak dengan kasat mata.

c. Proses

Proses adalah suatu bagian yang melakukan perubahan atau transformasi dari masukan menjadi keluaran yang berguna dan lebih bernilai dalam mencapai suatu tujuan dapat berupa informasi dan produk, akan suatu proses bisa juga berupa hal-

hal yang tidak berguna atau yang tidak diinginkan, misalnya seperti pembuangan limbah yang nantinya juga dapat menimbulkan suatu masalah dalam pencapaian suatu tujuan.

d. Keluaran

Keluaran atau output disini adalah lanjutan tahapan dari tahapan proses. Dalam output yang dihasilkan dari tahapan-tahapan sebelumnya ialah dapat berupa informasi yang diinginkan, laporan dan lain-lain.

e. Batas atau *boundary*

Batasan atau *boundary* adalah pemisah antara suatu sistem dengan daerah di luar sistem atau yang bisa disebut juga lingkungan. Dalam batas suatu sistem akan menentukan konfigurasi, ruang lingkup dan kemampuan suatu sistem. Contohnya adalah suatu perusahaan yang memiliki peraturan akan tetapi perusahaan atau organisasi tersebut memilik keterbatasan kemampuan karyawan atau pegawainya. Tentu saja batas suatu sistem dapat dikurangi atau diperbaharui sehingga sistem tersebut akan mengubah perilaku sistem itu sendiri.

f. Mekanisme Pengendalian dan Umpan Balik

Mekanisme pengendalian disini dapat diwujudkan dengan menggunakan umpan balik, yang mencuplik keluaran. Umpan balik ini digunakan dengan tujuan untuk mengendalikan suatu masukan maupun proses. Tujuannya utamanya adalah untuk mengatur agar sistem berjalan sesuai dengan tujuan.

g. Lingkungan

Lingkungan merupakan segala sesuatu yang berada diluar sistem. Lingkungan ini bisa berpengaruh terhadap suatu operasi sistem dalam arti bisa merugikan atau menguntungkan suatu sistem itu sendiri. Lingkungan yang merugikan tentu saja harus dihindari dan dikendalikan supaya tidak mengganggu akan kelangsungan operasi sistem, sedangkan yang menguntungkan tetap harus terus dijaga, dikarena akan memacu terhadap kelangsungan hidup sistem tersebut.

2.3.3 Karakteristik Sistem

Suatu sistem mempunyai karakteristik atau sifat-sifat yang tertentu, yaitu mempunyai komponen-komponen (*component*), batas sistem (*boundary*), lingkungan luar sistem (*environments*), penghubung (*interface*), masukan (*input*), pengolahan (*process*) keluaran (*output*) dan sasaran (*objektif*) atau tujuan (*goals*) .

a) Komponen Sistem

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerjasama membentuk suatu kesatuan. Komponen-komponen sistem atau elemen-elemen sistem dapat berupa suatu sub sistem atau bagian-bagian dari sistem. Setiap sub-sub sistem mempunyai sifat-sifat dari sistem yang menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan.

b) Batas Sistem

Batas sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem dengan sistem lainnya atau dengan lingkungan lainnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan. Batas suatu sistem menunjukan ruang lingkup (*scope*) dari sistem tersebut.

c) Lingkungan Luar Sistem

Lingkungan luar dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan. Lingkungan luar yang menguntungkan merupakan energi dari sistem dengan demikian harus dijaga dan dipelihara, sedangkan lingkungan luar yang merugiakan harus ditahan dan dikendalikan kalau tidak maka akan menggangu kelangsungan hidup dari sistem.

d) Penghubung Sistem

Penghubung merupakan media penghubung antara sub sistem dengan subsistem lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem lainnya. Keluaran (*output*) dari subsistem akan menjadi masukan (*input*) pada sistem lainnya dengan penghubung satu subsistem dapat berintegrasi dengan subsistem lainnya membentuk satu kesatuan.

e) Masukan Sistem

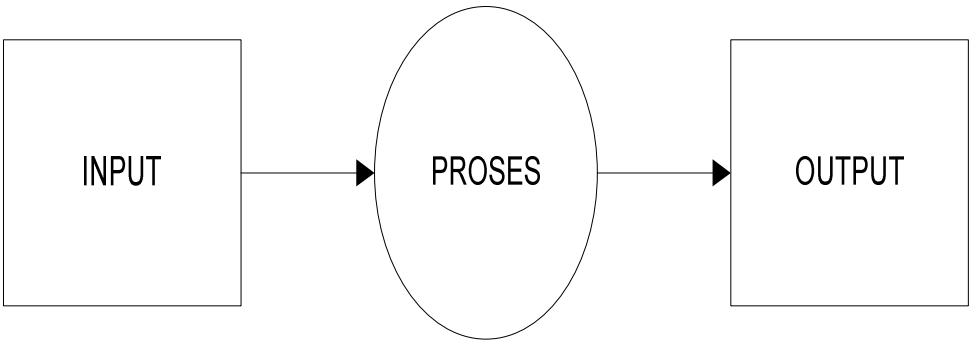
Masukan sistem adalah energi yang dimasukan ke dalam sistem. Masukan dapat berupa masukan perawatan (*maintenance input*) dan masukan (signal input). maintenance input adalah energi yang dimasukkan supaya sistem tersebut dapat beroperasi. Signal input adalah energi yang diproses untuk didapatkan keluarannya.

f) Keluaran Sistem

Keluaran adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa keluaran dapat merupakan masukan untuk subsistem yang lain atau kepada supra sistem.

g) Pengolahan Sistem

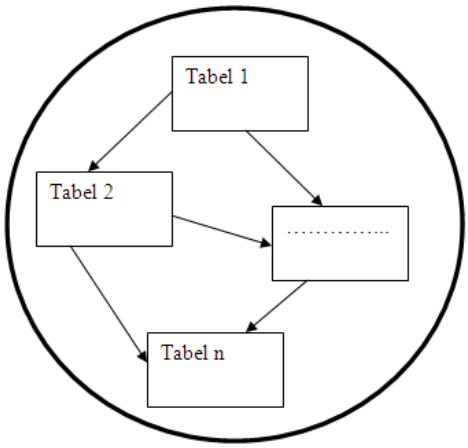
Suatu sistem dapat mempunyai suatu bagian pengolahan atau sistem itu sendiri sebagai pengolahnya. Pengolah yang akan merubah masukan menjadi keluaran.Bentuk umun dari suatu sistem terdiri atas masukan (*input*), proses dan keluaran (*output*) seperti terlihat pada gambar 2.1 dalam bentuk umum sistem ini bisa melakukan satu atau lebih masukan yang akan diproses dan menghasilkan keluaran sesuai dengan rencana yang telah ditentukan sebelumnya .



Gambar 2.1 Bentuk Umum Sistem

2.4 Basis Data

Sistem basis data adalah sistem terkomputerisasi yang tujuan utamannya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Pada intinya basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat. Basis data relasional yang diimplementasikan dengan tabel-tabel yang saling memiliki relasi seperti gambar 2.2.



Gambar 2.2 Ilustrasi Basis Data

Sistem informasi tidak dapat dipisahkan dengan kebutuhan akan basis data apapun bentuknya, entah berupa *file* teks ataupun *Database Management System* (DBMS). Kebutuhan basis data dalam sistem informasi meliputi :

1. memasukkan, menyimpan , dan mengambil data
2. membuat laporan berdasarkan data yang telah disimpan

2.4.1 Bahasa Basis Data

DBMS merupakan perantara bagi pemakai dengan basis data dalam disk. Cara berkomunikasi/berinteraksi antara pemakai dengan basis data tersebut diatur dalam suatu bahasa khusus yang ditetapkan oleh perusahaan pembuat DBMS. Bahasa tersebut dapat disebut sebagai bahasa basis data yang terdiri atas sejumlah perintah yang diformulasikan oleh *user* dan diproses oleh DBMS untuk melakukan suatu aksi atau pekerjaan tertentu.

Ada 3 bahasa yang digunakan dalam basis data yaitu :

1. DDL (*Data Definition Language*)

   Merupakan bahasa definisi data yang digunakan untuk membuat dan mengelola objek database seperti database, tabel dan view.
2. DML (*Data Manipulation Language*)

   Merupakan bahasa manipulasi data yang digunakan untuk memanipulasi data pada objek database seperti tabel.
3. DCL (*Data Control Language*)

   Merupakan bahasa yang digunakan untuk mengendalikan pengaksesan data.

2.4.2 DBMS

DBMS (*Database Management System*) atau sistem manajemen basis data adalah suatu sistem aplikasi yang digunakan untuk menyimpan, mengelola, dan menampilkan data. Suatu aplikasi disebut DBMS jika memenuhi persyaratan minimal sebagai berikut:

1. Menyediakan fasilitas untuk mengelola akses data.
2. Mampu menangani akses data.
3. Mampu menangani *backup* data.

Karena pentingnya data bagi suatu organisasi atau perusahaan, maka hampir sebagian besar perusahaan memanfaatkan DBMS dalam mengelola data yang mereka memiliki. Pengelola DBMS sendiri biasanya ditangani oleh tenaga ahli yang spesialis menangani DBMS yang disebut sebagai DBA (*Database Administrator*). Berikut ini adalah 4 macam DBMS versi komersial yang paling banyak digunakan didunia saat ini, yaitu :

1. Oracle
2. Microsoft SQL Server
3. IBM DB2
4. Microsoft Accessang

Sedangkan DBMS versi *open source* yang cukup berkembang dan palingbanyak digunakan saat ini adalah sebagai berikut :

1. MySQL
2. PostgreSQL
3. Firebird
4. SQLite

Hampir semua DBMS mengadopsi SQL sebagai bahasa untuk mengelola data pada DBMS

2.4.3 SQL

SQL(*Structured Query Language*) adalah bahasa yang digunakan untuk mengelola data pada RDBMS. SQL awalnya dikembangkan berdasarkan teori aljabar relasional dan kalkulus.

SQL mulai berkembang pada tahun 1970. SQL mulai digunakan sebagai standar yang resmi pada tahun 1986 oleh ANSI (*American National Standarts Institute*) dan pada tahun 1987 oleh ISO (*International Organization for Standardization*) dan disebut sebagai SQL-86. Berikut ini sejarah perkembangan SQL sampai saat ini :

Tabel 2.1 Sejarah perkembangan SQL

No	Tahun	Nama
1	1986	SQL-86
2	1989	SQL-89
3	1992	SQL-92
4	1999	SQL:1999
5	2003	SQL:2003
6	2006	SQL:2006
7	2008	SQL:2008

Meskipun SQL diadopsi dan diacu sebagai bahasa standar oleh hampir sebagian besar RDBMS yang beredar saat ini, tetapi tidak semua standar yang tercantum dalam SQL diimplementasikan oleh seluruh DBMS tersebut, sehingga ada perbedaan perilaku (hasil yang ditampilkan) oleh DBMS yang berbeda padahal query yang dimasukkan sama. Berikut ini merupakan contoh pengaksesan data pada DBMS sengan SQL yang secara umum terdiri dari 4 hal sebagai berikut:

1. Memasukkan data (*insert*)

```
INSERT INTO Tabel_Mahasiswa
            (nim, nama, tanggal_lahir)
VALUES
            ('10108534','Ricky Dwi Sugiono','21-08-1990')
```

Gambar 2.3 Query Insert

Query di atas digunakan untuk memasukkan data mahasiswa dengan nim 10108534, nama Ricky Dwi Sugiono, dan tanggal lahir 21 agustus 1990 ke tabel tabel_mahasiswa.

2. Mengubah data (*update*)

```
Update Tabel_mahasiswa
SET
      tanggal_lahir = '1990-03-21'
WHERE
      nim = '10108534'
```

Gambar 2.4 Query Update

Query pada gambar 2.4 digunakan untuk mengubah data lahir mahasiswa dengan nim = 10108534 menjadi 21 Maret 1990 dalam tabel_mahasiswa.

3. Menghapus data (*delete*)

```
DELETE FROM Tabel_Mahasiswa
WHERE
nim = '10108534'
```

Gambar 2.5 *Query Delete*

Query pada gambar 2.5 digunakan untuk menghapus data mahasiswa dengan NIM = 10108534 dari tabel_mahasiswa

4. Menampilkan data (*select*)

```
SELECT nim, nama
FROM Tabel_mahasiswa
WHERE nim='10108534'
```

Gambar 2.6 *Query Select*

Query pada gambar 2.6 digunakan untuk menampilkan data mahasiswa yang tersimpan dalam tabel_mahasiswa dengan nim = 10108534.

2.4.4 Alur Hidup Basis Data

Tidak hanya perangkat lunak yang memiliki alur hidup, dalam membuat perencanaan basis data juga memiliki alur hidup atau *Database Life Cycle (DBLC)* fase – fase DBLC antara lain:

1.Analisis kebutuhan/ *requirement analysis*

Hal – hal yang harus dilakukan pada tahap ini adalah:

a. Didefinisikan dengan mewawancarai produsen dan pemakai data

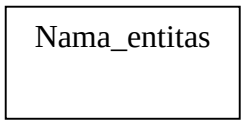
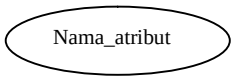
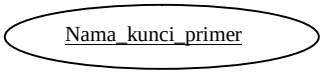
b. Membuat kontrak spesifikasi basis data

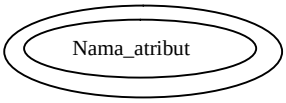
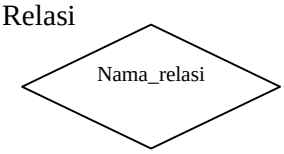
c. *Entity Relationship Diagram (ERD)*

2.4.5 ERD

Pemodelan awal basis data yang paling banyak digunakan adalah *Entity Relationship Diagram (ERD). ERD* dikembangkan berdasarkan teori himpunan dalam bidang matematika. *ERD* digunakan untuk pemodelan basis data relasional, sehingga jika penyimpanan basis data menggunakan OODBMS maka perancangan basis data tidak perlu menggunakan *ERD*. Adapun symbol – symbol yang digunakan dalam *ERD* dapat di lihat pada tabel 2.2

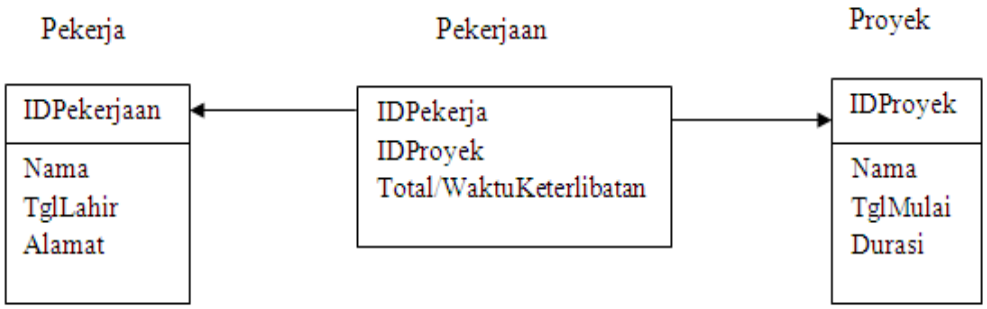
Tabel 2.2 Simbol ERD

Simbol	Deskripsi
Entitas/*entity* 	Entitas merupakan data inti yang akan disimpan; bakal tabel pada basis data
Atribut 	Field atau kolom data yang butuh disimpan dalam suatu entitas
Atribut kunci primer 	Field atau kolom data yang butuh disimpan dalam suatu entitas dan digunakan sebagai kunci akses *record* yang diinginkan, biasanya berupa id.

Atribut multinilai/multivalue 	*Field* atau kolom data yang butuh disimpan dalam suatu entitas yang dapat memiliki nilai lebih dari satu
Relasi 	Relasi yang menghubungkan antarentitas, biasanya diawali dengan kata kerja.
Asosiasi/*association* _____	Penghubung antara relasi dan entitas dimana di kedua ujungnya memiliki *multiplicity* kemungkinan jumlah pemakaian

2.4.6 Diagram Skema Relasi

Skema basisdata yang menyatakan kebergantungan kunci utama dan *foreign key* dapat digambarkan secara grafis dengan diagram skema relasi [5]. Berikut adalah diagram relasi untuk relasi. Pekerja, relasi pekerjaan dan relasi proyek.



Gambar 2.7 Diagram Skema Relasi

Masing masing relasi digambarkan sebagai kotak (persegi) dengan atribut – atribut didaftarkan didalamnya dan nama relasi muncul diatasnya. Jika tedapat atribut – atribut kunci utama direlasi, satu garis horizontal membelah persegi dimana atribut – atribut kunci utama didaftarkan diatas garis kebergantungan

foreign key muncul sebagai anak panah dari atribut – atribut *foreign key* relasi yang mengacu ke kunci utama dari relasi yang diacu.

Kita tidak boleh menyamakan skema relasi dengan diaram ER. Perbedaan utama ialah diagram ER tidak menunjukkan secara eksplisit atribut – atribut *foreign key* sementara diagram skema relasi telah menunjukkan atribut – atribut *foreign key* secara eksplisit.

Kebanyakan sistem basisdata menyediakan kakas perancangan dengan antarmuka pemakai secara grafis untuk menciptakan diagram skema relasi. Sekali lagi diagram yang langsung menyatakan *foreign key* dan kebergantungan terhadap kunci utama suatu relasi tidak disebut diagram ER melainkan skema relasi. Diagram skema relasi terdapat di Microsoft Access, PowerDesigner level fisik, serta dimasukkan dalam paket *MS SQL Server*.

2.5 Arsitektur Aplikasi

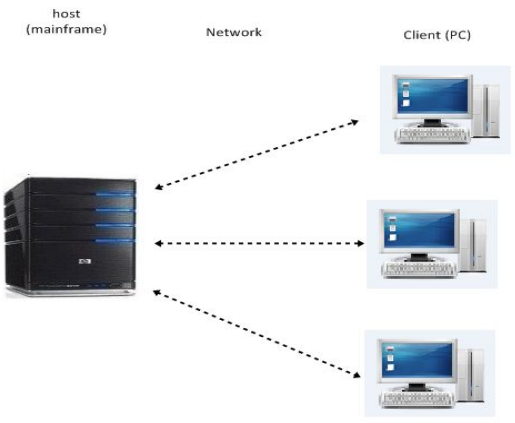
Istilah arsitektur mengacu pada desain sebuah aplikasi, atau dimana komponen yang membentuk suatu sistem ditempatkan dan bagaimana mereka berkomunikasi. Terdapat beberapa macam arsitektur aplikasi, yaitu :

1. *Standalone (one-tier)*

Pada arsitektur ini semua pemrosesan dilakukan pada *mainframe*. Kode aplikasi, data dan semua komponen sistem ditempatkan dan dijalankan pada *host*. Walaupun komputer *client* dipakai untuk mengakses *mainframe*, tidak ada pemrosesan yang terjadi pada mesin ini, dan karena mereka "*dump-client*" atau "*dump-terminal*". Tipe model ini, dimana semua pemrosesan terjadi secara

terpusat, dikenal sebagai berbasis-host. Sekilas dapat dilihat kesalahan pada model ini. Ada dua masalah pada komputasi berbasis *host*. Pertama, semua pemrosesan terjadi pada sebuah mesin tunggal, sehingga semakin banyak *user* yang mengakses *host*, semakin kewalahan jadinya. Jika sebuah perusahaan memiliki beberapa kantor pusat, *user* yang dapat mengakses *mainframe* adalah yang berlokasi pada tempat itu, membiarkan kantor lain tanpa akses ke aplikasi yang ada.

Pada saat itu jaringan sudah ada namun masih dalam tahap bayi, dan umumnya digunakan untuk menghubungkan terminal dump dan mainframe. Internet baru saja dikembangkan oleh pemerintah US dan pada saat itu dikenal sebagai ARPANET. Namun keterbatasan yang dikenakan pada user mainframe dan jaringan telah mulai dihapus.



Gambar 2.8 Arsitektur *stand alone*

2. *Client/Server (two-tier*)

Dalam model *client/server,* pemrosesan pada sebuah aplikasi terjadi pada *client* dan *server*. *Client/server* adalah tipikal sebuah aplikasi *two-tier* dengan banyak client dan sebuah server yang dihubungkan melalui sebuah jaringan, seperti terlihat dalam gambar 1.2. Aplikasi ditempatkan pada komputer *client* dan mesin *database* dijalankan pada *server* jarak-jauh. Aplikasi *client* mengeluarkan permintaan ke *database* yang mengirimkan kembali data ke *client*-nya.

Dalam *client/server, client-client* yang cerdas bertanggung jawab untuk bagian dari aplikasi yang berinteraksi dengan *user*, termasuk logika bisnis dan komunikasi dengan *server database*. Tipe-tipe tugas yang terjadi pada *client* adalah :

a. Antarmuka pengguna
b. Interaksi *database*
c. Pengambilan dan modifikasi data
d. Sejumlah aturan bisnis
e. Penanganan kesalahan

Server database berisi mesin *database*, termasuk tabel, prosedur tersimpan, dan *trigger* (yang juga berisi aturan bisnis). Dalam sistem *client/server*, sebagian besar logika bisnis biasanya diterapkan dalam *database. Server database* manangani :

a. Manajemen data
b. Keamanan
c. *Query, trigger*, prosedur tersimpan

d. Penangan kesalahan

Arsitektur client/server merupakan sebuah langkah maju karena mengurangi beban pemrosesan dari komputer sentral ke komputer *client*. Ini berarti semakin banyak user bertambah pada aplikasi *client/server*, kinerja *server file* tidak akan menurun dengan cepat. Dengan *client/server* user dair berbagai lokasi dapat mengakses data yang sama dengan sedikit beban pada sebuah mesin tunggal. Namun masih terdapat kelemahan pada model ini. Selain menjalankan tugas-tugas tertentu, kinerja dan skalabilitas merupakan tujuan nyata dari sebagian besar aplikasi. Model client/server memiliki sejumlah keterbatasan :

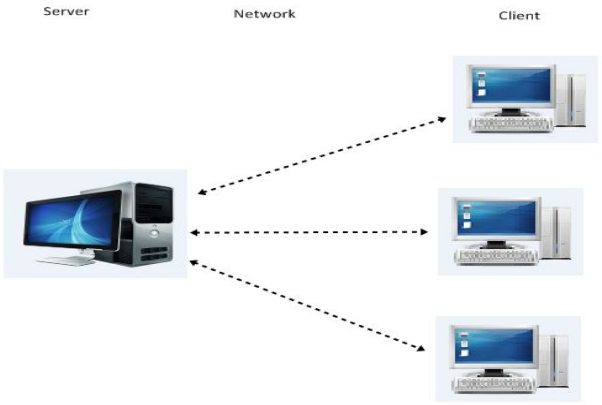
a. Kurangnya skalabilitas
b. Koneksi *database* dijaga
c. Tidak ada keterbaharuan *kode*
d. Tidak ada tingkat menengah untuk menangani keamanan dan transaksi

Aplikasi-aplikasi berbasis *client/server* memiliki kekurangan pada skalabilitas. Skalabilitas adalah seberapa besar aplikasi bisa menangani suatu kebutuhan yang meningkat – misalnya, 50 *user* tambahan yang mengakses aplikasi tersebut. Walaupun model *client/server* lebih terukur daripada model berbasis *host*, masih banyak pemrosesan yang terjadi pada *server*. Dalam model *client/server* semakin banyak client yang menggunakan suatu aplikasi, semakin banyak beban pada *server*.

Koneksi *database* harus dijaga untuk masing-masing *client*. Koneksi menghabiskan sumber daya *server* yang berharga dan masing-masing client tambahan diterjemahkan ke dalam satu atau beberapa koneksi. Logika kode tidak bisa didaur ulang karena kode aplikasi ada dalam sebuah pelaksanaan *executable* monolitik pada *client*. Ini juga menjadikan modifikasi pada kode sumber sulit. Penyusunan ulang perubahan itu ke semua komputer *client* juga membuat sakit kepala.

Keamanan dan transaksi juga harus dikodekan sebagai pengganti penanganan oleh COM+/MTS. Bukan berarti model *client/server* bukanlah merupakan model yang layak bagi aplikasi-aplikasi. Banyak aplikasi yang lebih kecil dengan jumlah user terbatas bekerja sempurna dengan model ini. Kemudahan pengembangan aplikasi *client/server* turut menjadikannya sebuah solusi menarik bagi perusahaan.

Pengembangan umumnya jauh lebih cepat dengan tipe sistem ini. Siklus pengembangan yang lebih cepat ini tidak hanya menjadikan aplikasi meningkat dan berjalan dengan cepat namun juga lebih hemat biaya.



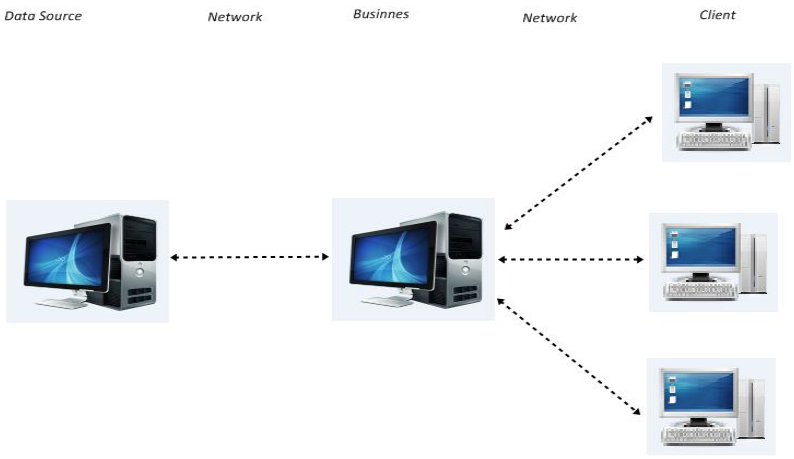
Gambar 2.9 Arsitektur *Two Tier*

3. *Three-Tier / Multi-Tier*

Model *three-tier* atau *multi-tier* dikembangkan untuk menjawab keterbatasan pada arsitektur *client/server*. Dalam model ini, pemrosesan disebarkan di dalam tiga lapisan (atau lebih jika diterapkan arsitektur *multi-tier*). Lapisan ketiga dalam arsitektur ini masing-masing menjumlahkan fungsionalitas khusus. Yaitu :

a. Layanan presentasi (tingkat *client*)
b. Layanan bisnis (tingkat menengah)
c. Layanan data (tingkat sumber data)

Layanan presentasi atau logika antarmuka pengguna ditempatkan pada mesin client. Logika bisnis dikeluarkan dari kode *client* dan ditempatkan dalam tingkat menengah. Lapisan layanan data berisi *database server*. Setiap tingkatan dalam model *three-tier* berada pada komputer tersendiri, seperti pada gambar 2.10



Gambar 2.10 Arsitektur *Three Tier*

Konsep model *three-tier* adalah model yang membagi fungsionalitas ke dalam lapisan-lapisan, aplikasi-aplikasi mendapatkan skalabilitas, keterbaharuan, dan keamanan.

2.6 Text To Speech

Suatu sistem *Text To Speech*(TTS) dapat didefinisikan sebagai sistem yang dapat mengubah suatu teks menjadi ucapan secara otomatis dengan cara fonetisasi (penyusunan fonem-fonem untuk membentuk ucapan) [6]. Sebuah sistem TTS dapat mengucapkan kata apapun, dan kosakatanya tidak terbatas. suatu sistem Text to Speech pada prinsipnya terdiri dari dua sub sistem, yaitu :

1) bagian konverter teks ke fonem (*Text to Phoneme*), serta

2) bagian konverter fonem ke ucapan (*Phoneme to Speech*).

Bagian Konverter Teks ke Fonem berfungsi untuk mengubah kalimat masukan dalam suatu bahasa tertentu yang berbentuk teks menjadi rangkaian kode-kode bunyi yang biasanya direpresentasikan dengan kode fonem, durasi serta *pitch*-nya. Bagian ini bersifat sangat language dependant. Untuk suatu bahasa baru, bagian ini harus dikembangkan secara lengkap khusus untuk bahasa tersebut.

Bagian konverter Fonem ke ucapan akan menerima masukan berupa kode-kode fonem serta *pitch* dan durasi yang dihasilkan oleh bagian sebelumnya. Berdasarkan kode-kode tersebut, bagian konverter fonem ke ucapan akan menghasilkan bunyi atau sinyal ucapan yang sesuai dengan kalimat yang ingin

diucapkan. Ada beberapa alternatif teknik yang dapat digunakan untuk implementasi bagian ini.

Dua teknik yang banyak digunakan adalah *formant synthesizer*, serta *diphone concatenation. Formant synthesizer* bekerja berdasarkan suatu model matematis yang akan melakukan komputasi untuk menghasilkan sinyal ucapan yang diinginkan. *Synthesizer* jenis ini telah lama digunakan pada berbagai aplikasi. Walaupun dapat menghasilkan ucapan dengan tingkat kemudahan interpretasi yang baik, *synthesizer* ini tidak dapat menghasilkan ucapan dengan tingkat kealamian yang tinggi.

Synthesizer yang menggunakan teknik *diphone concatenation* bekerja dengan cara menggabung-gabungkan segmen-segmen bunyi yang telah direkam sebelumnya. Setiap segmen berupa diphone (gabungan dua buah fonem). *Synthesizer* jenis ini dapat menghasilkan bunyi ucapan dengan tingkat kealamian (naturalness) yang tinggi. Struktur sistem seperti di atas pada prinsipnya merupakan konfigurasi tipikal yang digunakan pada berbagai sistem *Text to Speech* berbagai bahasa. Namun demikian, pada setiap sub-sistem terdapat sifat-sifat serta proses-proses yang sangat spesifik dan sangat tergantung dari bahasanya.

Konversi dari teks ke fonem sangat dipengaruhi oleh aturan-aturan yang berlaku dalam suatu bahasa. Pada prinsipnya proses ini melakukan konversi dari simbol-simbol tekstual menjadi simbol-simbol fonetik yang merepresentasikan unit bunyi terkecil dalam suatu bahasa. Setiap bahasa memiliki aturan cara

pembacaan dan cara pengucapan teks yang sangat spesifik. Hal ini menyebabkan implementasi unit konverter teks ke fonem menjadi sangat spesifik terhadap suatu bahasa. Untuk mendapatkan ucapan yang lebih alami, ucapan yang dihasilkan harus memiliki prosodi.

Secara kuantisasi, prosodi adalah perubahan nilai pitch selama pengucapan kalimat dilakukan atau pitch sebagai fungsi waktu. Pada prakteknya, informasi pembentuk prosodi berupa data-data pitch serta durasi pengucapannya untuk setiap fonem yang dibangkitkan. Nilai-nilai yang dihasilkan diperoleh dari suatu model prosodi. Prosodi bersifat sangat spesifik untuk setiap bahasa, sehingga model yang diperlukan untuk membangkitkan data-data prosodi menjadi sangat spesifik juga untuk suatu bahasa. Beberapa model umum prosodi pernah dikembangkan, tetapi untuk digunakan pada suatu bahasa masih perlu banyak penyesuaian yang harus dilakukan.

Konverter fonem ke ucapan berfungsi untuk membangkitkan sinyal ucapan berdasarkan kode-kode fonem yang dihasilkan dari proses sebelumnya. Sub sistem ini harus memiliki pustaka setiap unit ucapan dari suatu bahasa. Pada sistem yang menggunakan teknik *diphone concatenation*, sistem harus didukung oleh suatu diphone database yang berisi rekaman segmen-segmen ucapan yang berupa diphone.

Ucapan dalam suatu bahasa dibentuk dari satu *set* bunyi yang mungkin berbeda untuk setiap bahasa, oleh karena itu setiap bahasa harus dilengkapi dengan *diphone database* yang berbeda. Tahapan-tahapan utama konversi dari

teks menjadi ucapan dapat dinyatakan dengan diagram seperti terlihat pada Gambar 2.11. Tahap normalisasi teks berfungsi untuk mengubah semua teks kalimat yang ingin diucapkan menjadi teks yang secara lengkap memperlihatkan cara pengucapannya. Lihat contoh kalimat dan hasil normalisasinya pada Gambar 2.12.

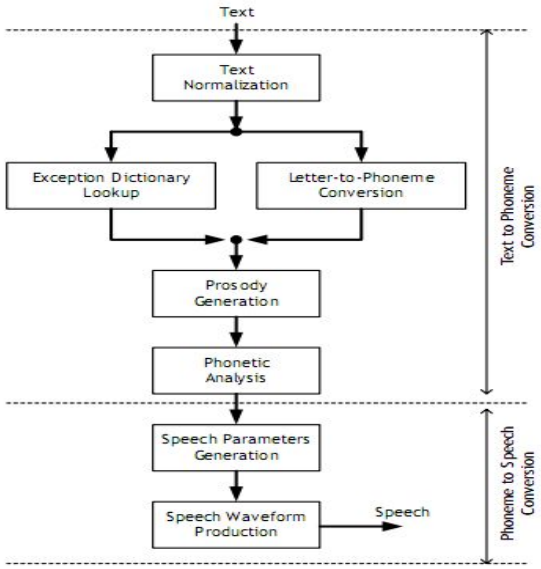
Tahap berikutnya adalah melakukan konversi dari teks yang sudah secara lengkap merepresentasikan kalimat yang ingin diucapkan menjadi kode-kode fonem. Konversi teks menjadi fonem biasanya dilakukan dengan dua cara. Sebagian proses konversi dapat dilakukan dengan aturan konversi yang sederhana dan berlaku umum untuk berbagai kondisi. Sebagian proses lainnya bersifat kondisional, tergantung dari huruf-huruf atau fonem-fonem tetangganya, bahkan terdapat bentuk-bentuk translasi yang tidak dapat ditemukan keteraturannya.

Konversi yang teratur dapat diimplementasikan dengan tabel konversi yang berisi pasangan antara urutan huruf dan urutan fonem, bahkan mungkin hanya berisi satu huruf dan satu fonem. Aturan yang lebih sulit biasanya diimplementasikan dengan tabel konversi yang akan diterapkan jika kondisi rangkaian huruf tetangga kiri dan kanannya terpenuhi. Contoh bentuk aturan konversi huruf ke fonem yang memenuhi teknik tersebut adalah sebagai berikut:

Left-context [letter-set] right-context = phoneme string

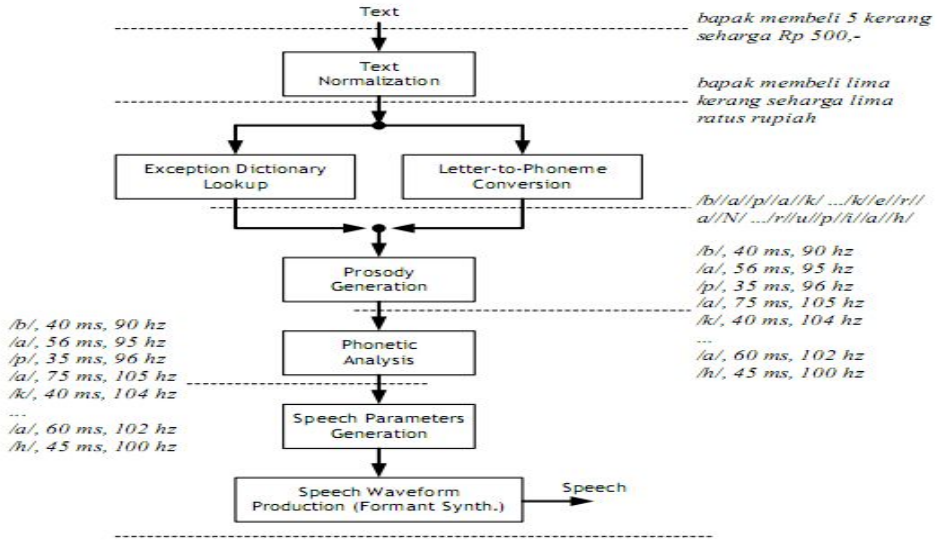
Huruf tertentu yang ditunjuk dalam posisi [*letter-set*] akan dikonversikan menjadi suatu fonem dalam “phoneme string” jika *left-context* dan *right context* terpenuhi.

Bahasa Inggris termasuk bahasa yang mempunyai keteraturan yang rendah untuk proses konversi teks ke fonem. Suatu TTS bahasa Inggris biasanya dilengkapi dengan suatu basis data yang berisi ribuan kata serta konversi padanan urutan fonemnya. Bahasa Indonesia termasuk bahasa yang jelas aturan konversinya. Sebagian besar kata dalam Bahasa Indonesia dapat dikonversikan menjadi fonem dengan aturan yang jelas dan sederhana, walaupun tetap ada kondisi-kondisi yang tidak dapat ditemukan keteraturannya



Gambar 2.11. Urutan Proses Konversi dari Teks ke Ucapan

Sebagai contoh, simbol huruf e dapat diucapkan sebagai e pepet atau e taling, artinya harus dikonversikan menjadi fonem yang berbeda untuk kondisi yang berbeda. Dalam blok diagram di atas, kondisi yang masih dapat ditangani oleh aturan diimplementasikan dengan blok *Letter to Phoneme Conversion*. Konversi yang tidak teratur ditangani oleh bagian *Exception Dictionary Lookup*.



Gambar 2.12. Besaran-besaran Dalam Setiap Tahap Proses Konversi dari Teks ke Ucapan

Hasil dari tahap tersebut adalah rangkaian *fonem* yang merepresentasikan bunyi kalimat yang ingin diucapkan. Bagian *prosody generato*r akan melengkapi setiap unit fonem yang dihasilkan dengan data durasi pengucapannya serta *pitch*-nya. Data durasi serta pitch diperoleh berdasarkan kombinasi antara tabel (*database*) serta model prosodi. Secara simbolik, hasil dari bagian ini sudah menghasilkan informasi yang cukup untuk menghasilkan ucapan yang diinginkan.Satu tahap berikutnya yang masih sering dilakukan adalah *Phonetic Analysis*. Tahap ini dapat dikatakan sebagai tahap penyempurnaan, yaitu melakukan perbaikan di tingkat bunyi. Sebagai contoh, dalam bahasa Indonesia, fonem /k/ dalam kata bapak tidak pernah diucapkan secara tegas, atau adanya sisipan fonem /y/ dalam pengucapan kata alamiah antara fonem /i/ dan /a/.

2.7 Jaringan Komputer

Jaringan komputer adalah sebuah kumpulan komputer, printer dan peralatan lainnya yang terhubung dalam satu kesatuan [7]. Informasi dan data bergerak melalui kabel-kabel atau tanpa kabel sehingga memungkinkan pengguna jaringan komputer dapat saling bertukar dokumen dan data, mencetak pada printer yang sama dan bersama-sama menggunakan *hardware/software* yang terhubung dengan jaringan. Setiap komputer, *printer* atau periferal yang terhubung dengan jaringan disebut *node*. Sebuah jaringan komputer dapat memiliki dua, puluhan, ribuan atau bahkan jutaan *node*.

2.7.1 Jenis-Jenis Jaringan Komputer

Secara umum jaringan komputer dibagi atas lima jenis, yaitu ;

1. *Local Area Network* (LAN)

Local Area Network (LAN), merupakan jaringan milik pribadi di dalam sebuah gedung atau kampus yang berukuran sampai beberapa kilometer. LAN seringkali digunakan untuk menghubungkan komputer-komputer pribadi dan *workstation* dalam kantor suatu perusahaan atau pabrik-pabrik untuk memakai bersama sumber daya (*resouce*, misalnya *printer*) dan saling bertukar informasi.

2. *Metropolitan Area Network* (MAN)

Metropolitan Area Network (MAN), pada dasarnya merupakan versi LAN yang berukuran lebih besar dan biasanya menggunakan teknologi yang sama dengan LAN. MAN dapat mencakup kantor-kantor perusahaan yang letaknya

berdekatan atau masih dalam satu kota dan dapat dimanfaatkan untuk keperluan pribadi (swasta) atau umum. MAN mampu menunjang data dan suara, bahkan dapat berhubungan dengan jaringan

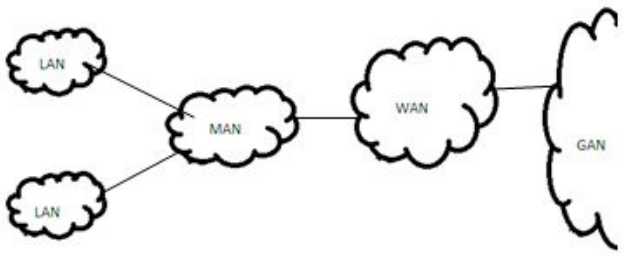
televisi kabel.

3. *Wide Area Network* (WAN)

Wide Area Network (WAN), jangkauannya mencakup daerah geografis yang luas. Seperti hubungan dari satu kota ke kota lain didalam suatu negara. Cakupan WAN bisa meliputi 100 km sampai 1000 km, dan kecepatan antara kota bisa bervariasi antara 1,5 Mbps sampai 2,4 Gbps.

1. *Global Area Network* (GAN)

GAN merupakan suatu jaringan yang menghubungkan negara – negara diseluruh dunia. Kecepatan GAN bervariasi mulai dari 1,5 Mbps sampai dengan 100 Gbps dan cakupannya mencakupi ribuan kilometer. Contoh yang sangat baik dari GAN ini adalah Internet LAN, MAN, WAN, dan GAN dapat berintraksi satu sama lain. Gambar 2.8 memperliatkan interaksi antara jaringan – jaringan tersebut.



Gambar 2.13 Interaksi antara LAN, MAN, WAN, dan GAN

2.8 Internet

Sebenarnya terdapat banyak jaringan didunia ini, seringkali menggunakan perangkat keras dan perangkat lunak yang berbeda-beda . Orang yang terhubung ke jaringan sering berharap untuk bisa berkomunikasi dengan orang lain yang terhubung ke jaringan lainnya. Keinginan seperti ini memerlukan hubungan antar jaringan yang seringkali tidak kampatibel dan berbeda. Biasanya untuk melakukan hal ini diperlukan sebuah mesin yang disebut gateway guna melakukan hubungan dan melaksanakan terjemahan yang diperlukan, baik perangkat keras maupun perangkat lunaknya. Kumpulan jaringan yang terinterkoneksi inilah yang disebut dengan internet.

2.9 UML

Unified Modeling Language (UML) adalah bahasa spesifikasi standar untuk mendokumentasikan, menspesifikasikan, dan membangun sistem perangkat lunak. UML adalah himpunan struktur dan teknik untuk pemodelan desain program berorientasi objek (OOP) serta aplikasinya. UML adalah metodologi untuk mengembangkan sistem OOP dan sekelompok perangkat *tool* untuk mendukung pengembangan sistem tersebut. UML mulai diperkenalkan *Object Management Group*, sebuah organisasi yang telah mengembangkan model, teknologi, dan standar OOP sejak tahun 1980-an.

Sekarang UML sudah mulai banyak digunakan oleh para praktisi OOP. UML merupakan dasar bagi perangkat (*tool*) desain berorientasi objek dari IBM. UML adalah suatu bahasa yang digunakan untuk menentukan, memvisualisasikan,

membangun, dan mendokumentasikan suatu sistem informasi. UML dikembangkan sebagai suatu alat untuk analisis dan desain berorientasi objek oleh Grady Booch, Jim Rumbaugh, dan Ivar Jacobson. Namun demikian UML dapat digunakan untuk memahami dan mendokumentasikan setiap sistem informasi. Penggunaan UML dalam industri terus meningkat. Ini merupakan standar terbuka yang menjadikannya sebagai bahasa pemodelan yang umum dalam industri peranti lunak dan pengembangan sistem.

2.9.1 Sejarah UML

Sampai era tahun 1990 puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. Diantaranya adalah: metodologi booch, metodologi coad, metodologi OOSE, metodologi OMT, metodologi shlaer-mellor, metodologi wirfs-brock, dan sebagainya. Masa itu terkenal dengan masa perang metodologi (method war) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasi sendiri-sendiri, yang mengakibatkan timbul masalah baru apabila kita bekerjasama dengan kelompok/perusahaan lain yang menggunakan metodologi yang berlainan.

Dimulai pada bulan Oktober 1994 Booch, Rumbaugh dan Jacobson, yang merupakan tiga tokoh yang boleh dikata metodologinya banyak digunakan mempelopori usaha untuk penyatuan metodologi pendesainan berorientasi objek. Pada tahun 1995 di-*release draft* pertama dari UML (versi 0.8). Sejak tahun 1996 pengembangan tersebut dikoordinasikan oleh *Object Management Group*.

2.9.2 Diagram UML

UML menyediakan 10 macam diagram untuk memodelkan aplikasi berorientasi objek, yaitu:

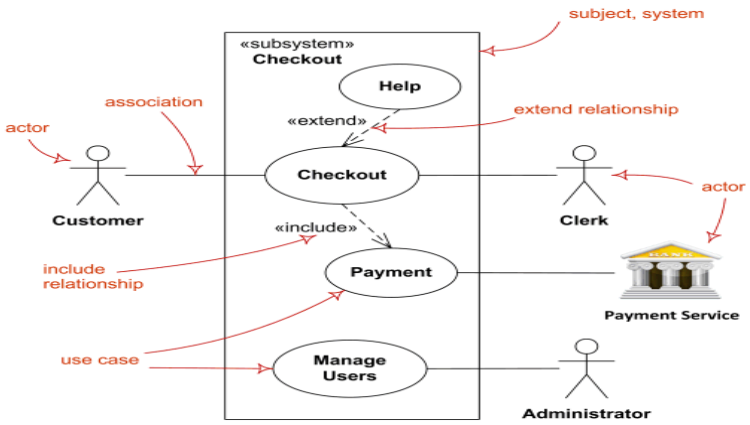
1. *Use Case Diagram* untuk memodelkan proses bisnis.
2. *Conceptual Diagram* untuk memodelkan konsep-konsep yang ada di dalam aplikasi.
3. *Sequence Diagram* untuk memodelkan pengiriman pesan (*message*) antar objek.
4. *Collaboration Diagram* untuk memodelkan interaksi antar objek.
5. *State Diagram* untuk memodelkan perilaku *objects* di dalam sistem.
6. *Activity Diagram* untuk memodelkan perilaku *Use Cases* dan objek di dalam sistem.
7. *Class Diagram* untuk memodelkan struktur kelas.
8. *Object Diagram* untuk memodelkan struktur objek.
9. *Component Diagram* untuk memodelkan komponen object.
10. *Deployment Diagram* untuk memodelkan distribusi aplikasi.

Berikut akan dijelaskan empat macam diagram yang paling sering digunakan dalam pembangunan aplikasi berorientasi objek, yaitu *use case diagram, sequence diagram, collaboration diagram, dan class diagram.*

1. *Use Case Diagram*

Use case diagram digunakan untuk memodelkan bisnis proses berdasarkan perspektif pengguna sistem. *Use case* diagram terdiri atas diagram untuk *use case*

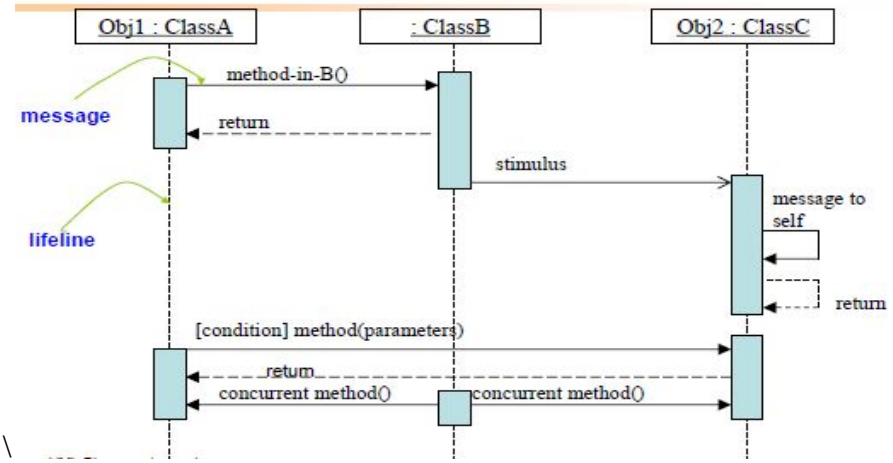
dan aktor. Aktor merepresentasikan orang yang akan mengoperasikan atau orang yang berinteraksi dengan sistem aplikasi. *Use case* merepresentasikan operasi-operasi yang dilakukan oleh aktor. *Use case* digambarkan berbentuk elips dengan nama operasi dituliskan didalamnnya. Aktor yang melakukan operasi dihubungkan dengan garis lurus ke *usecase*. Contoh dari *use case* bisa dilihat pada gambar 2.14



Gambar 2.14 *Use Case Diagram*

2. *Sequence Diagram*

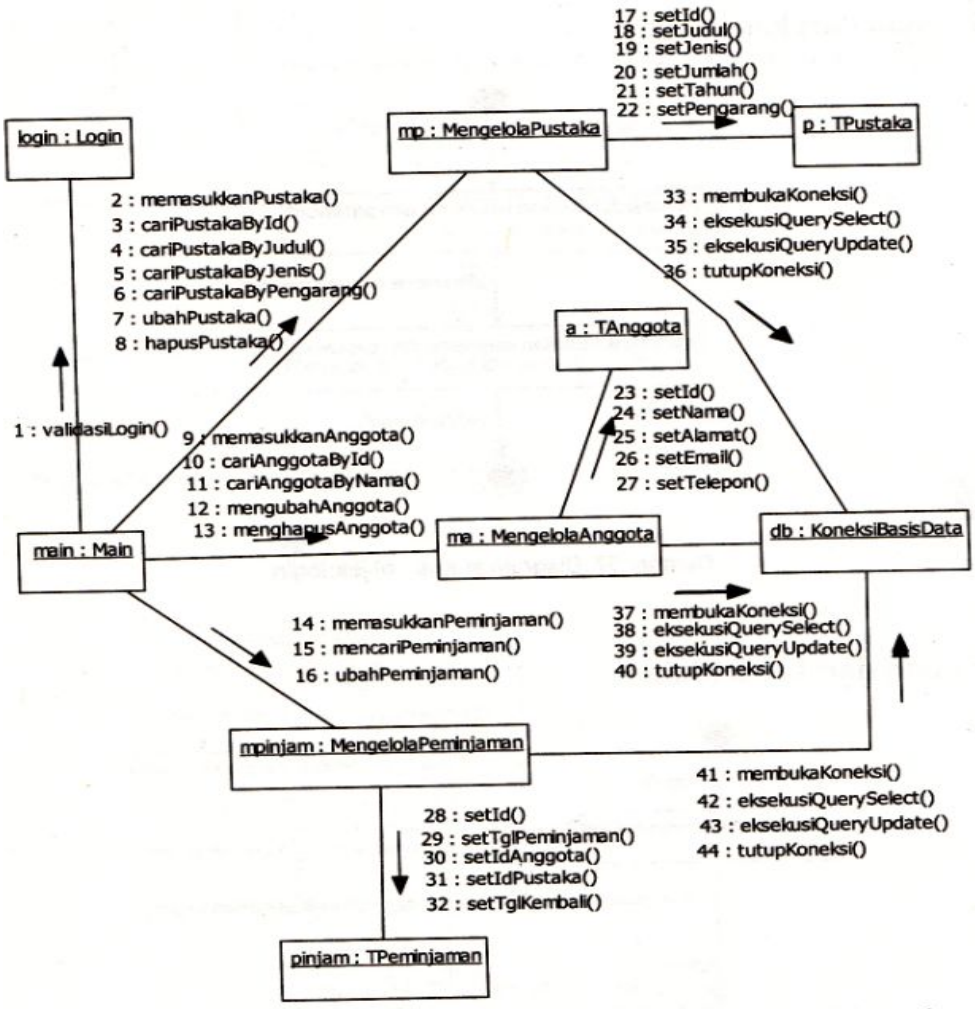
Sequence diagram menjelaskan secara detil urutan proses yang dilakukan dalam sistem untuk mencapai tujuan dari *use case*: interaksi yang terjadi antar *class*, operasi apa saja yang terlibat, urutan antar operasi, dan informasi yang diperlukan oleh masing-masing operasi. Contoh dari *sequence diagram* bisa dilihat pada gambar 2.15



Gambar 2.15 *Sequence Diagram*

3. *Collaboration Diagram*

Collaboration diagram dipakai untuk memodelkan interaksi antar objek di dalam sistem. Berbeda dengan *sequence diagram* yang lebih menonjolkan kronologis dari operasi-operasi yang dilakukan, *collaboration diagram* lebih focus pada pemahaman atas keseluruhan operasi yang dilakukan oleh objek. Contoh dari *collaboration diagram* bisa dilihat pada gambar 2.16

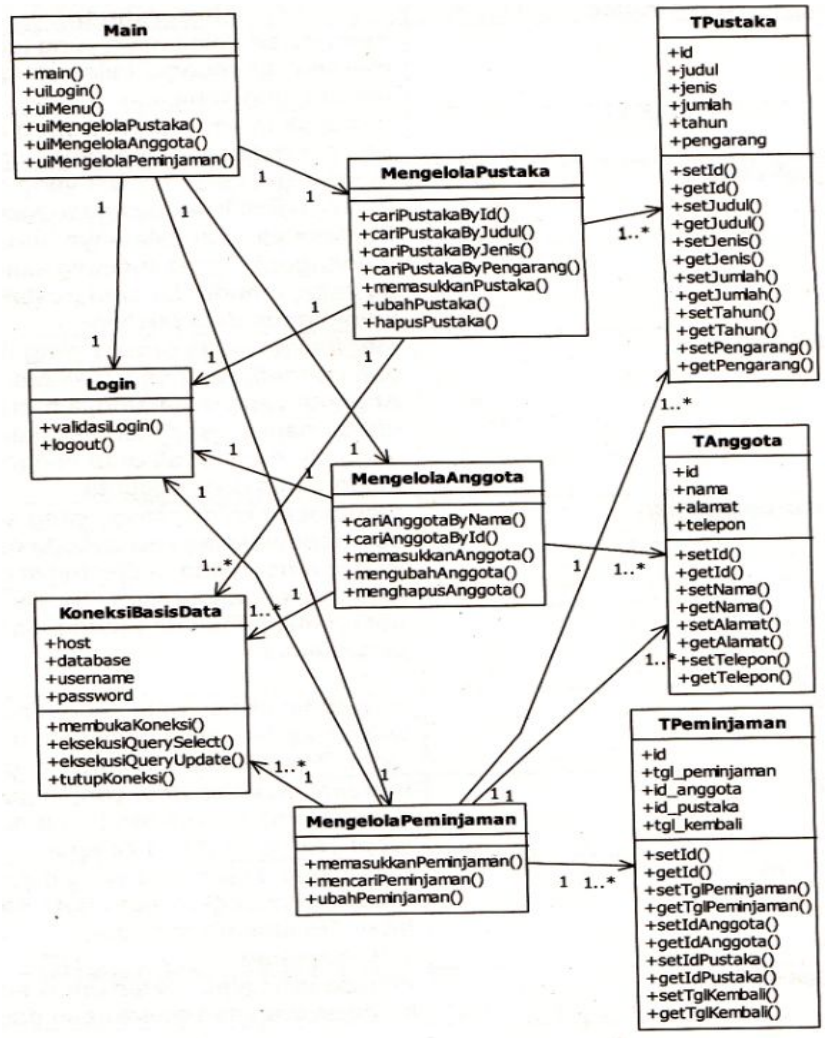


Gambar 2.16 ***Collaboration diagram***

4. *Class Diagram*

Class diagram merupakan diagram yang selalu ada di permodelan sistem berorientasi objek. *Class* diagram menunjukkan hubungan antar class dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk

mencapai suatu tujuan. Contoh dari *class diagram* bisa dilihat pada gambar 2.17



Gambar 2.17 *Class Diagram*

2.10 Android

Android merupakan subset perangkat lunak untuk perangkat *mobile* yang meliputi sistem operasi, *middleware* dan aplikasi inti yang di-*release* oleh Google [8]. Sedangkan Android SDK (*Software Development Kit*) menyediakan Tools dan API yang diperlukan untuk mengembangkan aplikasi pada platform Android dengan menggunakan bahasa pemrograman Java. Dikembangkan bersama antara

Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, NVIDIA yang tergabung dalam OHA (*Open Handset Alliance*) dengan tujuan membuat sebuah standar terbuka untuk perangkat bergerak (*mobile device*).

2.10.1 Sejarah Android

Pada Juli 2000, Google bekerjasama dengan Android Inc., perusahaan yang berada di Palo Alto, California Amerika Serikat. Para pendiri Android Inc. bekerja pada Google, di antaranya Andy Rubin, Rich Miner, Nick Sears, dan Chris White. Saat itu banyak yang menganggap fungsi Android Inc. hanyalah sebagai perangkat lunak pada telepon seluler. Sejak saat itu muncul rumor bahwa Google hendak memasuki pasar telepon seluler. Di perusahaan Google, tim yang dipimpin Rubin bertugas mengembangkan program perangkat seluler yangdidukung oleh kernel Linux. Hal ini menunjukkan indikasi bahwa Google sedang bersiap menghadapi persaingan dalam pasar telepon seluler. versi android terbaru yaitu versi 3.0. Android juga sudah bergabung dengan beberapa *smart Mobile*

Sekitar September 2007 sebuah studi melaporkan bahwa Google mengajukan hak paten aplikasi telepon seluler (akhirnya Google mengenalkan Nexus One, salah satu jenis telepon pintar GSM yang menggunakan Android pada sistem operasinya. Telepon seluler ini diproduksi oleh *HTC Corporation* dan tersedia di pasaran pada 5 Januari 2010).

Pada 9 Desember 2008, diumumkan anggota baru yang bergabung dalam program kerja Android ARM Holdings, *Atheros Communications*, diproduksi oleh Asustek Computer Inc, Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, dan

Vodafone Group Plc. Seiring pembentukan *Open Handset Alliance*, OHA mengumumkan produk perdana mereka, Android, perangkat bergerak (Mobile) yang merupakan modifikasi kernel Linux 2.6. Sejak Android dirilis telah dilakukan berbagai pembaruan berupa perbaikan bug dan penambahan fitur baru. Telepon pertama yang memakai sistem operasi Android adalah HTC Dream, yang dirilis pada 22 Oktober 2008. Pada penghujung tahun 2009 diperkirakan di dunia ini paling sedikit terdapat 18 jenis telepon seluler yang menggunakan Android.

1. Android versi 1.1

Pada 9 Maret 2009, Google merilis Android versi 1.1. Android versi ini dilengkapi dengan pembaruan estetis pada aplikasi, jam alarm, *voice search* (pencarian suara), pengiriman pesan dengan Gmail, dan pemberitahuan email.

2. Android versi 1.5 (Cupcake)

Pada pertengahan Mei 2009, Google kembali merilis telepon seluler dengan menggunakan Android dan SDK (*Software Development* Kit) dengan versi 1.5 (Cupcake). Terdapat beberapa pembaruan termasuk juga penambahan beberapa fitur dalam seluler versi ini yakni kemampuan merekam dan menonton video dengan modus kamera, mengunggah video ke Youtube dan gambar ke Picasa langsung dari telepon, dukungan Bluetooth A2DP, kemampuan terhubung secara otomatis ke headset Bluetooth, animasi layar, dan keyboard pada layar yang dapat disesuaikan dengan sistem.

3. Android versi 1.6 (Donut)

Donut (versi 1.6) dirilis pada September dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai indicator dan kontrol applet VPN. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus pada kamera, camcorder dan galeri yang dintegrasikan pada CDMA / EVDO, 802.1x, VPN, Gestures, dan Textto- speech engine. Kemampuan dial kontak teknologi text to change speech (tidak tersedia pada semua ponsel.

4. Android versi 2.0/2.1 (Eclair)

Pada 3 Desember 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1 (Eclair), perubahan yang dilakukan adalah pengoptimalan hardware, peningkatan Google Maps 3.1.2, perubahan UI dengan *browser* baru dan dukungan HTML5, daftar kontak yang baru, dukungan flash untuk kamera 3,2 MP, digital Zoom, dan Bluetooth 2.1. Untuk bergerak cepat dalam persaingan perangkat generasi berikut, Google melakukan investasi dengan mengadakan kompetisi aplikasi Mobile terbaik (killer apps - aplikasi unggulan). Kompetisi ini berhadiah $25,000 bagi setiap pengembang aplikasi terpilih. Kompetisi diadakan selama dua tahap yang tiap tahapnya dipilih 50 aplikasi terbaik. Dengan semakin berkembangnya dan semakin bertambahnya jumlah handset Android, semakin banyak pihak ketiga yang berminat untuk menyalurkan aplikasi mereka kepada sistem operasi Android. Aplikasi terkenal yang diubah ke dalam sistem operasi Android adalah Shazam, Backgrounds, dan WeatherBug. Sistem operasi Android

dalam situs Internet juga dianggap penting untuk menciptakan aplikasi Android asli, contohnya oleh MySpace dan Facebook.

5. Android versi 2.2 (Froyo: Frozen Yoghurt)

Pada 20 Mei 2010, Android versi 2.2 (Froyo) diluncurkan. Perubahan-perubahan umumnya terhadap versi-versi sebelumnya antara lain dukungan Adobe Flash 10.1, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, intergrasi V8 JavaScript engine yang dipakai Google Chrome yang mempercepat kemampuan rendering pada browser, pemasangan aplikasi dalam *SD Card*, kemampuan WiFi Hotspot portabel, dan kemampuan *auto update* dalam aplikasi Android Market.

6. Android versi 2.3 (Gingerbread)

Pada 6 Desember 2010, Android versi 2.3 (Gingerbread) diluncurkan. Perubahan-perubahan umum yang didapat dari Android versi ini antara lain peningkatan kemampuan permainan (*gaming*), peningkatan fungsi *copy paste*, layar antar muka *(User Interface)* didesain ulang, dukungan format video VP8 dan WebM, efek audio baru *(reverb, equalization, headphone virtualization*, dan *bass boost)*, dukungan kemampuan *Near Field Communication (NFC)*, dan dukungan jumlah kamera yang lebih dari satu.

7. Android versi 3.0 (Honeycomb)

Android Honeycomb dirancang khusus untuk tablet. Android versi ini mendukung ukuran layar yang lebih besar. *User Interface* pada Honeycomb juga

berbeda karena sudah didesain untuk tablet. Honeycomb juga mendukung multi prosesor dan juga akselerasi perangkat keras (*hardware*) untuk grafis. Tablet pertama yang dibuat dengan menjalankan Honeycomb adalah Motorola Xoom.

2.10.2. *Features*

Features yang tersedia pada platform Android saat ini antara lain:

1. Framework aplikasi yang mendukung penggantian komponen.
2. Integrated browser berdasarkan open source engine WebKit.
3. Mesin virtual Dalvik berjalan diatas Linux kernel dan dioptimalkan untuk perangkat mobile.
4. Grafis yang dioptimalkan dan didukung oleh *library* grafis 2D yang terkostumisasi, grafis 3D berdasarkan spesifikasi openGL ES 1.0 (Opsional akselerasi *hardware*).
5. SQLite untuk penyimpanan data.
6. Media Support yang mendukung audio, video, dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
7. GSM Telephony.
8. Bluetooth, EDGE, 3D, dan WiFi.
9. Dukungan Perangkat Tambahan : Android dapat memanfaatkan kamera, layar sentuh, *accelerometers*, *magnetometers*, GPS, akselerasi 2D (dengan perangkat orientasi, *scalling*, konversi format piksel) dan akselerasi grafis 3D.

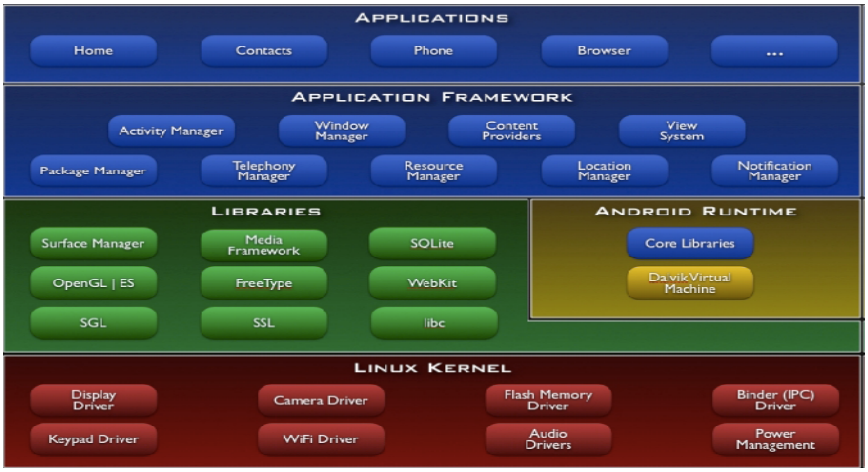
10. Multi-touch : kemampuan layaknya handset modern yang dapat menggunakan dua jari atau lebih untuk berinteraksi dengan perangkat.
11. Lingkungan development yang lengkap dan kaya termasuk perangkat emulator, tools untuk debugging, profil dan kinerja memori, dan plugin untuk

    Eclipse IDE.
12. Market : seperti kebanyakan handphone yang memiliki tempat penjualan aplikasi, market pada Android merupakan katalog aplikasi yang dapat didownload

    dan di-install pada handphone melalui internet.

2.10.3. Arsitektur Android

Gambar 2.14 menunjukkan komponen utama dari sistem operasi Android.



Gambar 2.18 Arsitektur Platform Android

1. Linux Kernel

Linux Kernal adalah layer dimana inti dari operating sistem dari android itu berada. Berisi file – file system yang mengatur sistem *processing, memory, resource, drivers*, dan sistem –sistem operasi android lainnya. Linux kernel yang digunakan android adalah linux kernel release 2.6.

2. *Libraries*

Libraries ini adalah layer dimana fitur – fitur android berada, biasanya para pembuat aplikasi mengakses *libraries* untuk menjalankan aplikasinya. Berjalan diatas kernel, Layer ini meliputi berbagai *library* C/C++ inti seperti Libc dan SSL, serta :

a. *Libraries* media untuk pemutaran media audio dan video
b. *Libraries* untuk manajemen tampilan
c. *Libraries Graphics* mencakup SGL dan OpenGL untuk grafis 2D dan 3D
d. *Libraries* SQLite untuk dukungan *database*
e. *Libraries* SSL dan Webkit terintegrasi dengan web browser dan security
f. *Libraries* LiveWebcore mencakup modern web browser dengan engine embedded web view
g. *Libraries* 3D yang mencakup implementasi OpenGL ES 1.0 API's.

3. Android *Runtime*

Android terdiri dari satu *set core libraries* yang menyediakan sebagian besar fungsi yang sama dengan yang terdapat pada *core libraries* bahasa pemrograman

Java. Setiap aplikasi menjalankan prosesnya sendiri dalam Android, dengan masing-masing instan dari mesin virtual Dalvik (Dalvik VM). Dalvik dirancang agar perangkat dapat menjalankan multiple VMs secara efisien. Mesin Virtual Dalvik mengeksekusi file dalam Dalvik executable (.dex), sebuah format yang dioptimalkan untuk memori yang kecil. Dalvik VM berbasis, berjalan, dan dikompilasi oleh compiler bahasa Java yang telah ditransformasikan ke dalam .dex format oleh tool “dx” yang telah disertakan. Dalvik VM bergantung pada kernel Linux untuk berfungsi seperti threading dan manajemen memori tingkat rendahnya.

4. *Framework* Aplikasi

Pengembang memiliki akses penuh menuju API *framework* yang sama dengan yang digunakan oleh aplikasi inti. Arsitektur aplikasi dirancang agar komponen dapat digunakan kembali *(reuse)* dengan mudah. Setiap aplikasi dapat memanfaatkan kemampuan ini (sesuai dengan batasan keamanan yang didefinisikan oleh *framework*). Mekanisme yang sama memungkinkan komponen untuk diganti oleh pengguna. Semua aplikasi merupakan rangkaian set layanan dan sistem, termasuk:

a. *Views* yang kaya dan *extensible* yang dapat digunakan untuk membangun aplikasi, termasuk *list, grids,* kotak teks, tombol, dan bahkan sebuah *embedded web*.
b. *Content Provider* yang memungkinkan aplikasi untuk mengakses data (seperti dari daftar kontak telepon) atau membagi data tersebut.

c. *Resource Manager*, yang menyediakan akses ke kode nonsumber lokal seperti string, gambar, dan tata letak file.
d. *Notifications Manager* yang memungkinkan semua aplikasi menampilkan *alert* yang bisa dikostumisasi di dalam status bar.
e. *Activity Manager* yang mengelola siklus hidup aplikasi dan menyediakan navigasi umum *backstack.*

5. *Applications*

Android telah menyertakan aplikasi inti seperti email *client*, SMS, kalender, peta, *browser*, kontak, dan lainnya. Semua aplikasi tersebut ditulis dengan menggunakan bahasa pemrograman Java. Pada layer inilah *developer* atau kita menempatkan aplikasi yang dibuat. Yang istimewa adalah pada Android semua aplikasi baik aplikasi inti (*native*) maupun aplikasi pihak ketiga berjalan pada layer aplikasi dengan menggunakan *library* API yang sama. Ini berarti semua aplikasi yang dibuat untuk Android akan memiliki akses yang setara dalam mengakses seluruh kemampuan handset, tanpa membedakan apakah itu merupakan aplikasi inti atau aplikasi pihak ketiga. Dalam kata lain dengan platform Android ini, *Programmer* atau *Developer* secara penuh akan bisa mengkostumisasi perangkat Androidnya.

2.11. Java

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Dikembangkan oleh Sun Microsystems dan

diterbitkan tahun 1995. Java tidak boleh disalahpahami sebagai JavaScript. JavaScript adalah bahasa scripting yang digunakan oleh web browser.

2.11.1. Sejarah Perkembangan Java

Bahasa pemrograman Java pertama lahir dari *The Green Project*, yang berjalan selama 18 bulan, dari awal tahun 1991 hingga musim panas 1992. Proyek tersebut belum menggunakan versi yang dinamakan Oak. Proyek ini dimotori oleh Patrick Naughton, Mike Sheridan, James Gosling dan Bill Joy, beserta Sembilan pemrogram lainnya dari Sun Microsystems. Salah satu hasil proyek ini adalah maskot Duke yang dibuat oleh Joe Palrang. Pertemuan proyek berlangsung di sebuah gedung perkantoran Sand Hill Road di Menlo Park. Sekitar musim panas 1992 proyek ini ditutup dengan menghasilkan sebuah program Java Oak pertama, yang ditujukan sebagai pengendali sebuah peralatan dengan teknologi layar sentuh (touch screen), seperti pada PDA sekarang ini. Teknologi baru ini dinamai "*7" (Star Seven).

Setelah era Star Seven selesai, sebuah anak perusahaan Tv kabel tertarik ditambah beberapa orang dari proyek *The Green Project*. Mereka memusatkan kegiatannya pada sebuah ruangan kantor di 100 Hamilton Avenue, Palo Alto. Perusahaan baru ini bertambah maju, jumlah karyawan meningkat dalam waktu singkat dari 13 menjadi 70 orang. Pada rentang waktu ini juga ditetapkan pemakaian Internet sebagai medium yang menjembatani kerja dan ide di antara mereka.

Pada awal tahun 1990-an, Internet masih merupakan rintisan, yang dipakai hanya di kalangan akademisi dan militer. Mereka menjadikan perambah *(browser*) Mosaic sebagai landasan awal untuk membuat perambah Java pertama yang dinamai *Web Runner*, terinsipirasi dari film 1980-an, Blade Runner. Pada perkembangan rilis pertama, Web Runner berganti nama menjadi Hot Java.

Pada sekitar bulan Maret 1995, untuk pertama kali kode sumber Java versi 1.0a2 dibuka. Kesuksesan mereka diikuti dengan untuk pemberitaan pertama kali pada surat kabar San Jose Mercury News pada tanggal 23 Mei 1995. Sayang terjadi perpecahan di antara mereka suatu hari pada pukul 04.00 di sebuah ruangan hotel Sheraton Palace. Tiga dari pimpinan utama proyek, Eric Schmidt dan George Paolini dari Sun Microsystems bersama Marc Andreessen, membentuk Netscape. Nama Oak, diambil dari pohon oak yang tumbuh di depan jendela ruangankerja "bapak java", James Gosling. Nama Oak ini tidak dipakai untuk versi release Java karena sebuah perangkat lunak sudah terdaftar dengan merek dagang tersebut, sehingga diambil nama penggantinya menjadi "Java".

Nama ini diambil dari kopi murni yang digiling langsung dari biji (kopi tubruk) kesukaan Gosling. Konon kopi ini berasal dari Pulau Jawa. Jadi nama bahasa pemrograman Java tidak lain berasal dari kata Jawa (bahasa Inggris untuk Jawa adalah *Java*).

2.11.2. Versi Awal

Versi awal Java ditahun 1996 sudah merupakan versi *release* sehingga dinamakan Java Versi 1.0. Java versi ini menyertakan banyak paket standar awal yang terus dikembangkan pada versi selanjutnya:

1. Java.lang: Peruntukan kelas elemen-elemen dasar.
2. Java.io: Peruntukan kelas input dan output, termasuk penggunaan berkas.
3. Java.util: Peruntukan kelas pelengkap seperti kelas struktur data dan kelas-kelas penanggalan.
4. Java.net: Peruntukan kelas TCP/IP, yang memungkinkan berkomunikasi dengan komputer lain menggunakan jaringan TCP/IP.
5. Java.awt: Kelas dasar untuk aplikasi antarmuka dengan pengguna (GUI)
6. Java.applet: Kelas dasar aplikasi antar muka untuk diterapkan pada penjelajah web.

2.11. 3. Kelebihan Java

a) *Multiplatform*. Kelebihan utama dari Java ialah dapat dijalankan di beberapa platform / sistem operasi komputer, sesuai dengan prinsip tulis sekali, jalankan di mana saja.

b) OOP (*Object Oriented Programming* - Pemrogram Berorientasi Objek) yang artinya semua aspek yang terdapat di Java adalah Objek. Hal ini sangat memudahkan pemrogram untuk mendesain, membuat, mengembangkan dan mengalokasi kesalahan sebuah program dengan basis Java secara cepat, tepat, mudah dan terorganisir.

c) Perpustakaan Kelas yang Lengkap, Java terkenal dengan kelengkapan library/perpustakaan (kumpulan program program yang disertakan dalam pemrograman java) yang sangat memudahkan dalam penggunaan oleh para pemrogram untuk membangun aplikasinya.

d) Bergaya C++, memiliki sintaks seperti bahasa pemrograman C++ sehingga menarik banyak pemrogram C++ untuk pindah ke Java.

e) Pengumpulan sampah otomatis, memiliki fasilitas pengaturan penggunaan memori sehingga para pemrogram tidak perlu melakukan pengaturan memori secara langsung (seperti halnya dalam bahasa C++ yang dipakai secara luas).

2.11.4. Kekurangan Java

1. Tulis sekali, perbaiki di mana saja - Masih ada beberapa hal yang tidak kompatibel antara platform satu dengan platform lain. Untuk J2SE, misalnya SWT-AWT bridge yang sampai sekarang tidak berfungsi pada Mac OS X.
2. Mudah didekompilasi. Dekompilasi adalah proses membalikkan dari kode jadi menjadi kode sumber. Ini dimungkinkan karena kode jadi Java merupakan *bytecode* yang menyimpan banyak atribut bahasa tingkat tinggi, seperti nama-nama kelas, metode, dan tipe data. Hal yang sama juga terjadi pada Microsoft .NET Platform. Dengan demikian, algoritma yang digunakan program akan lebih sulit disembunyikan dan mudah dibajak/di-reverseengineer.

3. Penggunaan memori yang banyak. Penggunaan memori untuk program berbasis Java jauh lebih besar daripada bahasa tingkat tinggi generasi sebelumnya seperti C/C++ dan Pascal.

2.11. 5. Tahap Kompilasi Java

1. Tulis / Ubah. Pemrogram menulis program dan menyimpannya di media dalam bentuk berkas '.java'.
2. Kompilasi. Pengkompilasi membentuk bytecodes dari program menjadi bentuk berkas '.class'.
3. Muat. Pemuat kelas memuat bytecodes ke memori.
4. Verifikasi. Peng-verifikasi memastikan bytecodes tidak mengganggu sistem keamanan Java.
5. Jalankan. Penerjemah menerjemahkan bytecodes ke bahasa mesin tidak bisa di pakai.

2.12. *Aplication Programming Interface* (API)

Antarmuka pemrograman aplikasi (*Application Programming Interface*) adalah sekumpulan perintah, fungsi, dan protokol yang dapat digunakan oleh pengembang saat membangung perangkat lunak untuk sistem operasi tertentu. API memungkinkan pengembang untuk menggunakan fungsi standar untuk berinteraksi dengan sistem operasi.

2.13. XAMPP

XAMPP adalah aplikasi *web server* instan yang dibutuhkan untuk membangun aplikasi berbasis *web*. Fungsi XAMPP adalah sebagai *serve*r yang berdiri sendiri (localhost), yang terdiri atas program Apache, http server, MySQL, *database*, dan penterjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan Perl. Nama XAMPP merupakan singkatan dari X (X=*Cross Platform*), Apache, MySQL, PHP dan Perl. Program ini tersedia dalam lisensi GNU (General Public Universitas Sumatera UtaraLicense) dan gratis. Dengan menginstal XAMPP, kita tidak perlu menginstal aplikasi server satu persatu karena di dalam XAMPP sudah terdapat :

a) Apache 2.2.14 (Ipv6 Enabled) + open SSL 0.9.8l

b) MySQL 5.1.41 + PBXT engine

c) PHP 5.3.1

d) PHPMyAdmin 3.2.4

e) Perl 5.10.1

f) Filezilla FTP Server 0.9.33.

g) Mercury Mail Transport System 4.72.

informasi terkait. Dokumentasi ini juga tersedia dalam bundel yand dapat diunduh dan diinstal pada komputer.

2.14. SQLite

SQLite merupakan sebuah perpustakaan proses yang menerapkan kemandirian, tanpa server, tidak adanya konfigurasi mesin, dan basis data SQL transaksional. Kode untuk SQLite berada dalam domain publik dengan demikian bebas untuk digunakan untuk tujuan apapun, komersial atau swasta. SQLite adalah mesin basis data SQL yang tertanam. Tidak seperti kebanyakan basis data SQL lainnya, SQLite tidak memiliki proses server yang terpisah. SQLite membaca dan menulis secara langsung ke file disk biasa. Basis data SQL lengkap dengan berbagai tabel, indeks, triggers, dan views terkandung dalam sebuah file disk tunggal.

2.15. Dreameaweaver

Macromedia Dreamweaver adalah sebuah HTML editor profesional untuk mendesain secara visual dan mengelola situs web maupun halaman web. Bilamana kita menyukai untuk berurusan dengan kode-kode HTML secara manual atau lebih menyukai bekerja dengan lingkungan secara visual dalam melakukan editing, Dreamweaver mambuatnya menjadi lebih mudah dengan menyediakan tool-tool yang sangat berguna dalam peningkatan kemampuan dan pengalaman dalam mendesain web

2.16. Eclipse

Eclipse adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua *platform* (*platform-independent*). Berikut ini adalah sifat dari Eclipse:

1. *Multi-platform*: Target sistem operasi Eclipse adalah Microsoft Windows, Linux, Solaris, AIX, HP-UX dan Mac OS X.
2. *Mulit-language*: Eclipse dikembangkan dengan bahasa pemrograman Java, akan tetapi Eclipse mendukung pengembangan aplikasi berbasis bahasa pemrograman lainnya, seperti C/C++, Cobol, Python, Perl, PHP, dan lain sebagainya.
3. *Multi-role*: Selain sebagai IDE untuk pengembangan aplikasi, Eclipse pun bisa digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, test perangkat lunak, pengembangan web, dan lain sebagainya. Eclipse pada saat ini merupakan salah satu IDE favorit dikarenakan gratis dan open *source*, yang berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Selain itu, kelebihan dari Eclipse yang membuatnya popular adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan komponen yang dinamakan plug-in.

2.16.1 Sejarah Eclipse

Eclipse awalnya dikembangkan oleh IBM untuk menggantikan perangkat lunak IBM Visual Age for Java 4.0. Produk ini diluncurkan oleh IBM pada tanggal 5 November 2001, yang menginvestasikan sebanyak US$ 40 juta untuk pengembangannya. Semenjak itu konsursium Eclipse Foundation mengambil alih untuk pengembangan Eclipse lebih lanjut dan pengaturan organisasinya.

2.16.2 Arsitektur Eclipse

Sejak versi 3.0, Eclipse pada dasarnya merupakan sebuah kernel, yang mengangkat plug-in. Apa yang dapat digunakan di dalam Eclipse sebenarnya adalah fungsi dari plug-in yang sudah diinstal. Ini merupakan basis dari Eclipse yang dinamakan *Rich Client Platform (RCP)*. Berikut ini adalah komponen yang membentuk RCP:

1. Core platform

2. OSGi

3. SWT (Standard Widget Toolkit)

4. JFace

5. Eclipse Workbench

Secara standar Eclipse selalu dilengkapi dengan JDT (*Java Development Tools*), plug-in yang membuat Eclipse kompatibel untuk mengembangkan program Java, dan PDE (*Plug-in Development Environment*) untuk mengembangkan plug-in baru. Eclipse beserta plug-in-nya diimplementasikan dalam bahasa pemrograman Java. Konsep Eclipse adalah IDE yang terbuka (*open*), mudah diperluas (*extensible*) untuk apa saja, dan tidak untuk sesuatu yang spesifik. Jadi, Eclipse tidak saja untuk mengembangkan program Java, akan tetapi dapat digunakan untuk berbagai macam keperluan, cukup dengan menginstal plug-in yang dibutuhkan. Apabila ingin mengembangkan program C/C++ terdapat plug-in CDT (C/C++ Development Tools).

Selain itu, pengembangan secara visual bukan hal yang tidak mungkin oleh Eclipse, plug-in UML 2 tersedia untuk membuat diagram UML. Dengan menggunakan PDE setiap orang bisa membuat plug-in sesuai dengan keinginannya.

2.17. *Software Development Kit* (SDK)

Software Development Kit (*SDK*) adalah suatu kit atau *library* dari bahasa pemrograman untuk pengembangan atau pembangunan suatu perangkat lunak dan biasanya *SDK* terdiri dari kumpulan tools yang dibutuhkan. Misalnya bahasa pemrograman java, mempunyai *SDK* yang berisi suatu *library* yang dapat digunakan untuk membuat suatu aplikasi berbasis java.

2.18. *Java Development Kit (JDK)*

Java Development Kit (JDK) adalah sekumpulan perangkat lunak yang dapat digunakan untuk mengembangkan perangkat lunak yang berbasis Java, Sedangkan JRE adalah sebuah implementasi dari *Java Virtual Machine* yang benar-benar digunakan untuk menjalankan program java.Biasanya, setiap JDK berisi satu atau lebih JRE dan berbagai alat pengembangan lain seperti sumber kompiler java, *bundling, debuggers, development libraries* dan lain sebagainya. Perbedaan JDK dengan SDK (*Software Development Kit*) yaitu JDK adalah sebuah SDK tetapi sebuah SDK tidak harus menjadi sebuah JDK.

2.19. ADT (*Android Development Tools)*

Android Development Tools (ADT) adalah plugin untuk Eclipse *Intergrated Development Environment* (IDE) yang dirancang untuk memberikan lingkungan yang terpadu di mana untuk membangun aplikasi Android. ADT memperluas kemampuan Eclipse untuk membiarkan para *developer* lebih cepat dalam membuat proyek baru Android, membuat aplikasi UI, menambahkan komponen berdasarkan Android *Framework* API, debug aplikasi dalam pengunaan Android SDK, dan membuat file APK untuk mendistribusikan aplikasi. Mengembangkan aplikasi di Eclipse dengan ADT sangat dianjurkan dan merupakan cara tercepat untuk memulai membuat aplikasi android, karena banyak kemudahan-kemudahan sebagai *tools* yang terintegrasi seperti, *custom XML editor*, dan debug panel ouput. Selain itu ADT memberikan dorongan luar biasa dalam mengembangkan aplikasi Android.

2.20. *TTS Library*

TTS Library merupakan suatu paket tambahan yang berisikan kelas – kelas yang mendukung *text to speech* dalam mengembangkan aplikasi pada *platform* android [9]. Dengan *TTS Library* memungkinkan pengembang aplikasi android untuk mengimplementasikan teknologi *text to speech* dengan banyak bahasa yang berbeda termasuk bahasa indonesia. Dengan menggunakan TTS Library maka kelas yang digunakan untuk mengembangkan aplikasi android akan bertambah, salah satu kelas yang ada pada *TTS Library* adalah kelas *TextToSpeechBeta* adapun beberapa *method* yang terkandung dalam kelas dapat dilihat pada tabel 2.3

Tabel 2.3. *Method* pada kelas *TextToSpeechBeta*

No	*Method*	Fungsi
1	*isInstalled()*	Memeriksa apakah smartphone yang digunakan mendukung TTS
2	*isSpeaking()*	Memeriksa apakah proses konversi teks ke ucapan sedang aktif atau tidak
3	*setLanguage(Locale Language)*	Mengatur bahasa yang digunakan untuk mengucapkan teks yang ingin dikonversi ke ucapan
4	*setPitch(float pitch)*	Mengatur nada ucapan hasil konversi dari teks
5	*setSpeechRate(float speechRate)*	Mengatur tingkat kecepatan pengucapan yang dihasilkan dari konversi teks
6	*speak(text, queueMode, params)*	Mengkonversi teks ke ucapan sesuai dengan antrian *string* dan parameter *speech* yang telah ditentukan sebelumnya
7	*stop()*	Menghentikan proses konversi teks ke ucapan yang sedang berjalan atau aktif