BAB 2

LANDASAN TEORI

2.1 Teori Umum

2.1.1 Basis Data

Basis data adalah sekumpulan data yang saling berhubungan secara logikal dan juga merupakan deskripsi data yang dirancang untuk memenuhi kebutuhan informasi dari suatu organisasi (Connolly dan Begg, 2010, p65).

Basis data adalah sekumpulan data persisten yang digunakan oleh sistem aplikasi dari suatu perusahaan(C.J. Date, 2000, p10). Sistem basis data pada dasarnya merupakan suatu sistem penyimpanan *record* atau data yang terkomputerisasi (C.J. Date, 2000, p5).

Melalui definisi diatas, dapat disimpulkan bahwa basis data merupakan sekumpulan data persisten yang terhubung secara logikal dan merupakan deskripsi data yang dirancang sesuai kebutuhan organisasi.

2.1.2 Database Management System (DBMS)

Database Management System (DBMS) adalah suatu sistem piranti lunak yang memungkinkan *user* untuk mendefinisikan, membuat, memelihara, dan mengendalikan akses terhadap basis data (Connolly dan Begg, 2010, p66).

Fasilitas yang disediakan DBMS antara lain adalah:

1. *Data Definition Language* (DDL)

   Memungkinkan *user* untuk membuat spesifikasi tipe data, mendefinisikan basis data, struktur data dan *constraint* untuk

disimpan dalam basis data. *Constraint* merupakan peraturan konsistensi nilai pada basis data yang tidak dapat dilanggar.

2. *Data Manipulation Language* (DML)

   Memungkinkan *user* untuk melakukan *insert*, *update*, *delete*, dan mengirim atau mengambil data dari basis data.

3. Menyediakan kontrol akses ke basis data:

   - Sistem keamanan (*security system*), mencegah user yang tidak mempunyai hak akses agar tidak dapat mengakses basis data.
   - Sistem integritas (*integrity system*), memelihara konsistensi data yang disimpan.
   - Sistem kontrol pada saat yang bersamaan (*concurrency control system*), memperbolehkan *shared* akses terhadap basis data.
   - Sistem kontrol perbaikan (*recovery control system*), mengembalikan basis data kepada keadaan sebelumnya yang konsisten saat terjadi kegagalan pada *software* atau *hardware*.
   - Katalog yang dapat diakses oleh *user (a user-accessible catalog)*, yang memuat deskripsi data dari suatu basis data.

2.1.2.1 Keuntungan dan Kerugian DBMS

Keuntungan DBMS (Connolly dan Begg, 2010, p77):

1. *Control of data redundancy*

Pendekatan basis data mencoba menghapus redudansi dengan menggabungkan file-file sehingga salinan data yang sama tidak tersimpan.

2. *Data consistency*

   Basis data membantu menghilangkan atau mengatur redundansi. Hal ini akan mengurangi terjadinya data yang tidak konsisten.

3. *More information from the same amount of data*

   Dengan integrasi data operasional, hal ini memungkinkan perusahaan untuk mendapatkan tambahan informasi dari data yang sama.

4. *Sharing of data*

   Basis data dimiliki oleh seluruh organisasi dan dapat dibagi kepada semua *user* yang memiliki hak akses. Dengan demikian, banyak *user* dapat membagi banyak data.

5. *Improved data integrity*

   Database integrity menunjuk pada keabsahan dan konsistensi dari data yang disimpan.

6. *Improved security*

   Database security adalah pelindung basis data dari *user* yang tidak memiliki hak akses. Tanpa keamanan yang tepat, integrasi hanya akan membuat data menjadi lebih mudah diserang.

7. *Enforcement of standards*

   Integrasi memperbolehkan DBA untuk menentukan dan menyelenggarakan standar yang diperlukan.

8. *Economy of scale*

   Dengan menggabungkan data operasional suatu perusahaan ke dalam satu basis data, dan membuat sebuah kumpulan aplikasi yang bekerja pada satu sumber data, akan dapat menghemat pengeluaran suatu perusahaan.

9. *Balance ofconflicting requiremens*

   Setiap *user* atau departemen memilki kebutuhan akan data yang berbeda. Karena basis data berada dibawah kontrol DBA, maka DBA dapat membuat keputusan tentang perancangan dan operasional dari suatu basis data.

10. *Improved data accessibility and responsiveness*

    Sebagai hasil dari integrasi, data yang melewati batas departemen dapat diakses oleh *end-user*. Hal ini akan menghasilkan sebuah sistem dengan fungsionalitas yang tinggi.

11. *Increase Productivity*

    DBMS menyediakan sebuah lingkungan *fourth-generation environment* yang terdiri dari *tools* yang menyederhanakan pengembangan apikasi basis data. Hal inilah yang dapat meningkatkan produktivitas *programmer* dan juga mengurangi waktu pengembangan.

12. *Improved maintenance through data independence*

    DBMS memisahkan deskripsi mengenai data dengan aplikasi. Hal ini akan membuat aplikasi bebas untuk berubah dalam data *description*. Hal inilah yang disebut dengan data *independence*.

13. *Increase concurrency*

    DBMS akan mengatur akses basis data secara bersamaan dan memastikan agar tidak terjadi kehilangan informasi atau integritas.

14. *Improvedbackupandrecovery service*

    DMBS menyediakan fasilitas yang dapat mengurangi proses yang akan mengalami kegagalan.

Kerugian DBMS (Connolly dan Begg, 2010, p80):

1. *Complexity*

   Keinginan untuk membuat DBMS yang baik akan membuat DBMS menjadi perangkat lunak (*software*) yang sangat kompleks. Perancang dan pengembang basis data, *Admin*istrator data dan basis data, serta *end-users* harus mengerti fungsionalitas DBMS agar dapat memperoleh keuntungan dari DBMS. Kesalahan dalam mengerti sistem akan menghasilkan rancangan yang buruk dan akhirnya akan berdampak buruk bagi organisasi.

2. *Size*

   Kompleksitas fungsionalitas membuat DBMS menjadi perangkat lunak (*software*) yang memiliki ukuran sangat besar dan memerlukan tempat yang besar di memori untuk dapat bekerja secara efisien.

3. *Cost of DBMS*

   Biaya DBMS bervariasi secara signifikan sesuai dengan lingkungan dan fungsionalitas yang disediakan. Selain itu terdapat biaya untuk perawatan per tahun.

4. *Additional Hardware Costs*

   Tempat penyimpanan yang dibutuhkan untuk DBMS dan basis data mengakibatkan biaya tambahan untuk membeli tempat penyimpanan tambahan. Lebih jauh lagi, untuk menghasilkan kinerja yang dibutuhkan, mungkin dibutuhkan pembelian mesin yang lebih besar, bahkan mungkin sebuah mesin tersendiri untuk menjalankan DBMS.

5. *Cost of Conversion*

   Dalam beberapa situasi, biaya DBMS dan tambahan perangkat keras (*hardware*) dapat tidak sebanding dengan biaya untuk mengkonversi aplikasi yang sudah ada pada DBMS yang baru dan perangkat kerasnya. Biaya ini juga termasuk biaya pelatihan karyawan untuk menggunakan sistem yang baru, dan kemungkinan biaya untuk

mempekerjakan karyawan spesialis yang membantu dalam mengkonversi dan menjalankan sistem.

6. *Performance*

   Biasanya, DBMS ditulis menjadi lebih umum untuk mendukung banyak aplikasi. Akibatnya beberapa aplikasi tidak dapat berjalan dengan kecepatan yang seharusnya.

7. *Higher Impact of a failure*

   Sumber data yang tersentralisasi meningkatkan kerentanan terhadap sistem. Sejak semua *user* dan aplikasi bergantung pada ketersediaan dalam DBMS, kegagalan dari beberapa komponen dapat mengakibatkan operasi terhenti.

2.1.2.2 Fungsi DBMS

Berikut merupakanfungsi dari DBMS (Connolly dan Begg, 2010, p99):

1. *Data storage, retrieval, and update*

   DBMS harus mempunyai kemampuan untuk menyimpan (*store*), memperoleh kembali (*retrieve*), dan memperbarui (*update*) data dalam basis data.

2. *A user-accessible catalog*

   DBMS harus menyediakan sebuah katalog di mana deskripsi data dapat disimpan dan diakses oleh *user*.

3. *Transaction support*

   DBMS harus menyediakan suatu mekanisme yang akan menjamin bahwa semua *update* yang berhubungan dengan suatu transaksi dibuat atau tidak ada satu pun *update* yang akan dibuat.

4. *Concurrency control services*

   DBMS harus menyediakan mekanisme yang menjamin bahwa basis data di *update* secara benar, ketika banyak *user* sedang melakukan *update* pada basis data secara bersamaan.

5. *Recovery services*

   DBMS harus menyediakan mekanisme yang dapat memperbaiki basis data ketika masalah timbul.

6. *Authorization services*

   DBMS harus menyediakan mekanisme yang menjamin bahwa hanya *user* yang mempunyai ijin yang dapat melakukan akses terhadap basis data.

7. *Support for data communication*

   DBMS harus mampu beradaptasi dengan *software* komunikasi.

8. *Integrity services*

   DBMS harus dapat menjamin bahwa data yang tersedia maupun perubahan terhadap data, tetap mengikuti peraturan yang ada.

9. *Services to promote data independence*

   DBMS harus mempunyai fasilitas yang mendukung independensi suatu program dari struktur basis data aktual.

10. *Utility services*

    DBMS harus menyediakan satu set fasilitas pelayanan, seperti fasilitas untuk ekspor impor, pengecekan kegunaan basis data dan operasinya, dan sebagainya.

2.1.3 Standart Query Language (SQL)

SQL adalah salah satu contoh dari bahasa berbasistransformasi ataubahasa yang dirancang menggunakan relasigunamentransformasikandata masukan (*input*) menjadi datakeluaran (*output*) yang dibutuhkan(Connolly dan Begg, 2010, p184).

Operasi dasar yang terdapat dalam SQL terdiri dari dua yaitu :

1. *Data Definition Language* (DDL)

   DDL adalah suatu bahasa yang memungkinkan *Admin*istrator basis data atau *user* untuk mendefinisikan, menerangkan, dan memberi nama entitas, atribut, dan hubungan yang dibutuhkan untuk aplikasi (Connolly dan Begg, 2010, p92).

   Beberapa pernyataan dasar dari DDL (Connolly dan Begg, 2010, p237):

   - *Create Table*

     Berguna untuk membuat tabel dengan mengidentifikasi tipe data untuk tiap kolom.

- *Alter Table*

  Berguna untuk menambah atau membuang kolom dan *constraint*.

- *Drop Table*

  Berguna untuk membuang atau menghapus tabel beserta semua data yang terkait di dalamnya.

- *Create Index*

  Berguna untuk membuat index pada suatu tabel.

- *Drop Index*

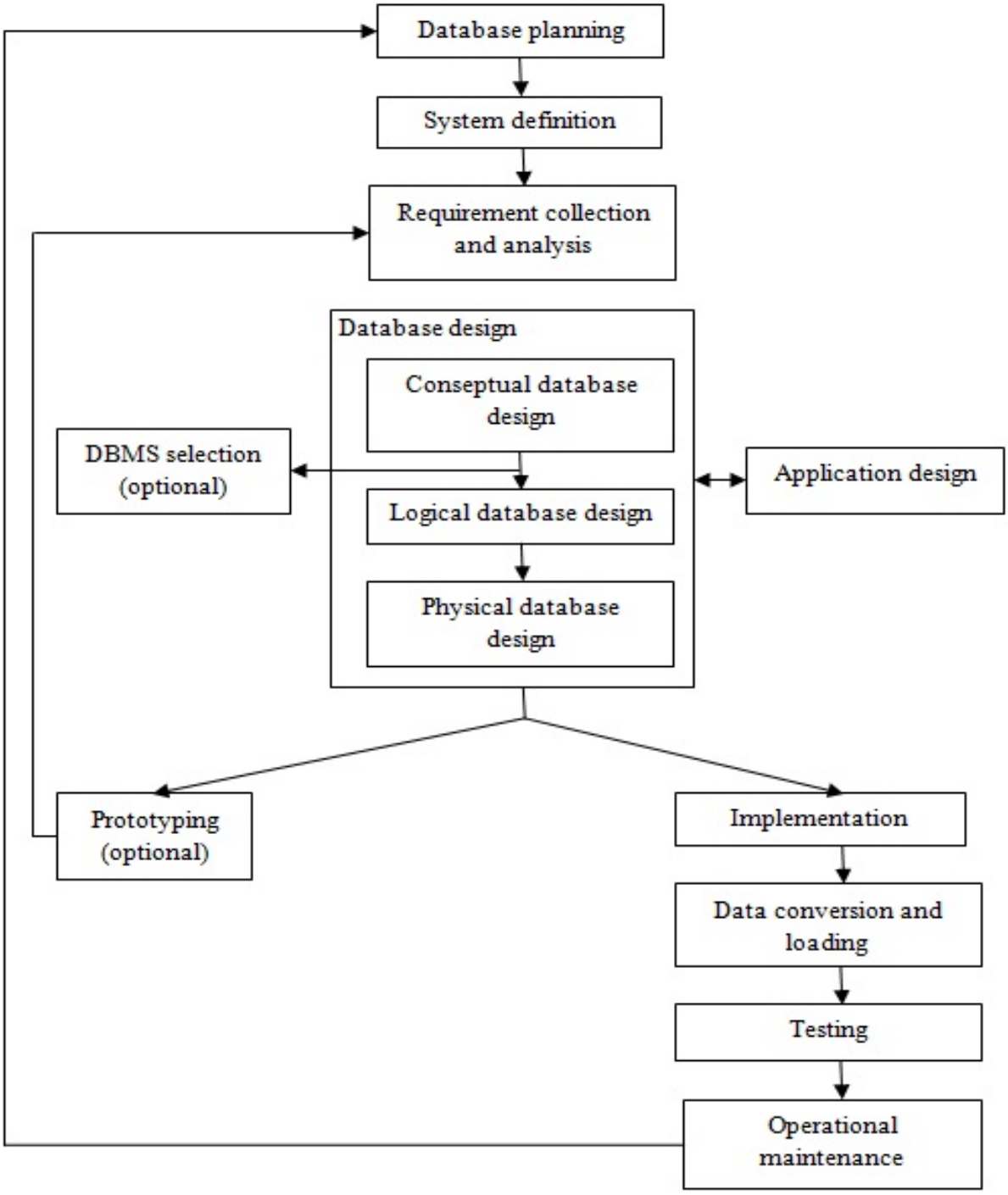
  Berguna untuk membuang atau menghapus indeks yang telah dibuat sebelumnya.

2. *Data Manipulation Language* (DML)

   DML adalah suatu bahasa yang menyediakan kumpulan operasi yang diinginkan untuk mendukung operasi manipulasi data terutama pada data yang diperoleh dalam basis data (Connolly dan Begg, 2010, p92).

   Data Manipulation di SQL terdiri dari operasiSELECT, INSERT, UPDATE, dan DELETE. SELECTberguna untuk mengerjakan batasan relasional, proyek,dan menggabungkan operasi pada data. INSERT untukmemasukkan ke dalam tabel-tabel data baru. UPDATEberguna untuk memperbaharui isi dalam tabel dengan databaru. DELETE berguna untukmenghapus salah satu isitabel dengan kondisi yang diinginkan.

2.1.4 Siklus Basis Data



Gambar 2.1 Diagram Siklus Basis Data (Connolly dan Begg, 2010, p314)

Ketika sistem basis data menjadi komponen mendasar suatu sistem informasi organisasi yang lebih besar dan luas, maka siklus aplikasi basis data dihubungkan dengan siklus hidup sistem informasi.

Penjelasan siklus aplikasi basis data (Connolly dan Begg, 2010, p313-p335):

1. Perencanaan Basis Data (*Database Planning*)

   Perencanaan basis data merupakan aktivitas manajemen yang memungkinkan tahapan dari siklus aplikasi basis data direalisasikan seefektif mungkin. Perencanaan basis data harus terintegrasi dengan keseluruhan strategi sistem informasi dari organisasi.

   Terdapat tiga hal pokok yang berkaitan dengan strategi sistem informasi, yaitu :

   - Identifikasi rencana dan sasaran dari perusahaan termasuk mengenai informasi yang dibutuhkan.
   - Evaluasi sistem informasi yang ada untuk menetapkan kelebihan dan kekurangan yang dimiliki.
   - Penaksiran kesempatan teknologi informasi yang mungkin memberikan keuntungan kompetitif.

2. Pendefinisian Sistem (*Sistem Definiton*)

   Pendefinisian sistem menjelaskan batasan-batasan dan cakupan dari aplikasi basis data dan sudut pandang user (*user view*) yang utama. *User view* mendefinisikan apa yang diwajibkan dari suatu

aplikasi basis data melalui beberapa perspektif. Suatu aplikasi basis data dapat memiliki lebih dari satu *user view*.

Identifikasi *user view* membantu memastikan bahwa tidak ada pengguna utama dari suatu basis data yang terlupakan ketika pembuatan aplikasi baru dibutuhkan. *User view* juga membantu dalam pengembangan aplikasi basis data yang kompleks memungkinkan permintaan dipecah ke dalam bagian-bagian yang lebih mudah diatur.

3. Analisis dan Pengumpulan Kebutuhan (*Requirements Collection and Analysis*)

Analisis dan pengumpulan kebutuhan merupakan suatu proses pengumpulan data dan analisa informasi mengenai bagian organisasi yang didukung oleh aplikasi basis data, dan menggunakan informasi tersebut untuk identifikasi kebutuhan pengguna pada sistem baru. Informasi yang dikumpulkan untuk setiap *user view* utama meliputi :

- Deskripsi data yang digunakan atau dihasilkan
- Detail mengenai bagaimana data digunakan atau dihasilkan
- Beberapa kebutuhan tambahan untuk aplikasi basis data yang baru.

Informasi-informasi tersebut kemudian dianalisa untuk diidentifikasi kebutuhan agar disertakan dalam aplikasi basis data yang baru. Selain mengumpulkan informasi, aktivitas penting lain yang dikerjakan pada bagian ini adalah menentukan bagaimana

mengatur aplikasi basis data dengan banyak *user view*. Hal tersebut dapat dikerjakan melalui tiga pendekatan, yaitu :

- Pendekatan Terpusat (*Centralized Approach*)

  Pendekatan terpusat menggabungkan kebutuhan dari setiap *user view* menjadi sekumpulan kebutuhan. Berdasarkan kumpulan kebutuhan tersebut dapat dibuat global data model yang mempresentasikan seluruh *user view*.

- Pendekatan Integrasi *View* (*View Integration Approach*)

  Pendekatan Integrasi *View* merupakan pendekatan dimana kebutuhan untuk setiap *user view* digunakan untuk membangun model data terpisah untuk merepresentasikan *user view* tersebut. Hasil dari model data tersebut nantinya digabungkan dalam tahapan perancangan basis data.

- Kombinasi dari pendekatan terpusat dan pendekatan integrasi *view*.

4. Perancangan Basis Data (*Database Design*)

   Perancangan basis data merupakan suatu proses pembuatan sebuah desain basis data yang akan mendukung tujuan dan operasi dari perusahaan.

   Tujuan utama perancangan basis data adalah sebagai berikut :

   - Merepresentasikan data dan *relationship* antar data yang dibutuhkan oleh seluruh area aplikasi utama dan grup pengguna.

- Menyediakan model data yang mendukung segala transaksi yang diperlukan pada data.
- Menspesifikasikan rancangan minimal yang secara tepat disusun untuk memenuhi kebutuhan performa yang ditetapkan pada sistem.

Beberapa pendekatan yang digunakan dalam perancangan basis data :

- *Top-Down*

  Pendekatan *Top-Down* diawali dengan pembentukan model data yang berisi beberapa entitas *high level* dan relationship, yang kemudian menggunakan pendekatan *Top-Down* secara berturut-turut untuk mengidentifikasi entitas *lower level*, *relationship*, dan atribut lainnya.

- *Bottom-Up*

  Pendekatan *Bottom-Up* dimulai dari atribut dasar dengan analisis dari penggabungan antar atribut, yang dikelompokkan ke dalam suatu relasi yang merepresentasikan tipe dari entitas dan *relationship* antar entitas.

- *Inside-Out*

  Pendekatan *inside-out* berhubungan dengan pendekatan *bottom-up* tetapi sedikit berbeda dengan identifikasi awal entitas utama dan kemudian menyebar ke entitas, *relationship*, dan atribut terkait lainnya yang lebih dulu diidentifikasi.

- *Mixed*

  Pendekatan *mixed* menggunakan pendekatan *bottom-up* dan *top-down* utuk bagian yang berbeda sebelum pada akhirnya digabungkan menjadi satu.

Kegunan utama dari pemodelan data adalah untuk membantu dalam memahami arti (semantik) dari data sekaligus untuk memfasilitasi komunikasi mengenai informasi yang dibutuhkan. Kriteria untuk menghasilkan model data yang optimal :

- Validitas Struktural (*Structural Validity*) harus konsisten dengan definisi perusahaan dan informasi organisasi.
- Kesederhanaan (*Simplicity*), mudah dimengerti baik oleh profesional sistem informasi maupun pengguna non-teknik.
- Ketetapan (*Expressiblity*), kemampuan untuk membedakan antar data yang berlainan, *relationship* antar data dan batasan-batasan.
- Tidak Rangkap (*Nonredundancy*), pengeluaran informasi yang tidak berhubungan, dengan kata lain, representasi setiap bagian informasi hanya satu kali.
- Dapat di gunakan bersama (*Shareability*), tidak ditentukan untuk aplikasi atau teknologi tertentu dan dapat digunakan oleh banyak pengguna.

- Perluasan penggunaan (*Extensibilty*), kemampuan untuk menyusun dan mendukung kebutuhan baru dengan akibat sampingan yang minimal terhadap pengguna yang sudah ada.
- Integritas (*Integrity*), konsitensi dengan cara yang digunakan *enterprise* dan pengaturan informasi.
- Representasi Diagram (*Diagrammatic Representation*), kemampuan untuk merepresentasikan model menggunakan notasi diagram yang mudah dimengerti.

Terdapat tiga fase dalam perancangan basis data, yaitu :

1. Perancangan basis data konseptual

   Suatu proses pembentukan model dari informasi yang digunakan dalam perusahaan, tidak tergantung dari keseluruhan aspek fisik, Model data dibangun dengan menggunakan informasi dalam spesifikasi kebutuhan pengguna. Model data konseptual merupakan sumber informasi untuk fase perancangan logikal.

2. Perancangan basis data logical

   Suatu proses pembentukan model dari informasi yang digunakan dalam perusahaan berdasarkan model data tertentu, tetapi tidak tergantung terhadap DBMS tertentu dan aspek fisik lainnya. Model data konseptual yang telah dibuat

sebelumnya, diperbaiki dan dipetakan ke dalam model data logikal.

3. Perancagan basis data fisikal

Suatu proses yang menghasilkan deskripsi implementasi basis data pada penyimpanan sekunder. Menggambarkan struktur penyimpanan dan metode akses yang digunakan untuk mencapai akses yang efisien terhadap data.

5. Penyeleksian DBMS (*DBMS Selection*)

Pemilihan DBMS yang tepat untuk mendukung aplikasi basis data. Dapat dilakukan kapanpun sebelum menuju desain logical asalkan terdapat cukup informasi mengenai kebutuhan sistem.

6. Perancangan Aplikasi (*Aplication Design*)

Perancangan antarmuka pengguna dan progam aplikasi yang menggunakan dan memproses basis data. Perancangan basis data dan aplikasi merupakan aktifikas parallel yang meliputi dua aktivitas penting, yaitu :

- Perancangan Transaksi (*Transaction Design*)

Transaksi adalah salah satu aksi atau serangkaian aksi yang dilakukan oleh pengguna tunggal atau progam aplikasi yang mengakses atau merubah isi dari basis data. Kegunaan dari perancangan transaksi adalah untuk menetapkan keterangan karakteristik *high level* dari suatu transaksi yang dibutuhkan pada basis data, diantaranya sebagai berikut :

- Data yang akan digunakan oleh transaksi

- Karakteristik fungsional dari suatu transaksi
- Output transaksi
- Keuntungannya bagi pengguna
- Tingkat kegunaan yang diharapkan

- Perancangan Antarmuka Pengguna (*User Interface Design*)

  Sebelum mengimplementasikan suatu *form* atau *report*, sangat penting untuk merancang layout. Beberapa aturan pokok untuk merancang *form* atau *report* adalah sebagai berikut :

  - Judul yang sesuai arti
  - Instruksi yang komprehensif
  - Pengelompokan logikal dan *field* yang berurutan
  - Tampilan layout *form* atau *report* menarik
  - Nama *field* mudah dikenali
  - Terminologi dan penggunaan singkatan yang konsisten
  - Penggunaan warna yang konsisten
  - Tempat dan batasan tempat untuk data yang akan dimasukkan
  - Pergerakan kursor yang mudah
  - Koreksi kesalahan untuk satu karakter atau seluruh *field*
  - Pesan kesalahan dimunculkan untuk nilai yang tidak dapat diterima
  - *Field* yang bersifat opsional diberi tanda yang jelas

- Penjelasan untuk setiap *field* dapat dibaca dengan jelas
- Pemberian suatu tanda jika proses telah selesai

7. Pembuatan prototype (*Propotyping*)

Prototyping adalah kegiatan membuat model kerja suatu aplikasi database. Tujuan utama dari pembuatan *prototyping* adalah sebagai berikut :

- Mengidentifikasi fitur dari sistem berjalan dengan baik atau tidak
- Memberikan perbaikan atau penambahan fitur baru
- Mengklarifikasi kebutuhan pengguna
- Mengevaluasi feasibilitas dari rancangan sistem khusus.

Dalam membuat *prototype*, ada dua macam strategi *prototyping* yang digunakan, yaitu :

- *Requirements Prototyping*

  Requirements Prototyping menggunakan *prototype* untuk menentukan kebutuhan dari aplikasi basis data yang diinginkan dan ketika kebutuhan itu terpenuhi maka *prototype* akan dibuang.

- *EvolutionaryPrototyping*

  Evolutionary Prototyping digunakan untuk tujuan yang sama dengan *requirement prototyping*, tetapi *prototype* tidak

dibuang melainkan pengembangan lanjutan menjadi aplikasi basis data yang digunakan.

8. Implementasi (*Implementation*)

   Implementasi merupakan realisasi fisik dari basis data dan rancangan aplikasi. Implementasi basis data dicapai dengan menggunakan :

   - DDL untuk membuat skema basis data dan *file* basis data kosong
   - DDL untuk membuat *user view* yang diinginkan
   - 3GL dan 4GL untuk membuat progam aplikasi. Termasuk basis data transaksi disertakan dengan menggunakan DML atau ditambahkan pada bahasa pemrograman.

9. Loading dan Konversi Data (*Loading and Data Conversion*)

   Pemindahan data yang ada ke dalam basis data baru dan mengkonversikan aplikasi yang ada agar dapat digunakan pada basis data yang baru. Tahapan ini dibutuhkan ketika sistem basis data baru menggantikan sistem yang lama. DBMS biasanya memiliki utilitas yang memanggil ulang *file* yang sudah ada ke dalam basis data baru.

10. Pengujian (*Testing*)

    Pengujian merupakan proses eksekusi progam aplikasi dengan tujuan untuk menemukan kesalahan. Dengan menggunakan strategi tes yang direncanakan dan data yang sesungguhnya. Pengujian hanya akan terlihat jika terjadi kesalahan *software*. Mendemonstrasikan

basis data dan progam aplikasi terlihat berjalan seperti yang diharapkan.

11. Perawatan operasional (*Operational Maintenance*)

    Suatu proses pengawasan dan pemeliharaan sistem setelah instalasi, meliputi

    - Pengawasan performa sistem, jika performa menurun maka memerlukan perbaikan atau pengaturan ulang basis data.
    - Pemeliharaan dan pembaharuan aplikasi basis data (jika dibutuhkan)
    - Penggabungan kebutuhan baru ke dalam aplikasi basis data.

2.1.5 Entity Relationship Modeling (ER Modelling)

ER Modelling adalah sebuah pendekatan *top-down* untukmerancang basis data yang dimulai dengan mengidentifikasi datayang penting yang disebut entitas dan *relationship* antar data harusdipresentasikan dalam model (Connolly dan Begg, 2010, p371).

2.1.5.1 Konsep Dasar ER Modelling

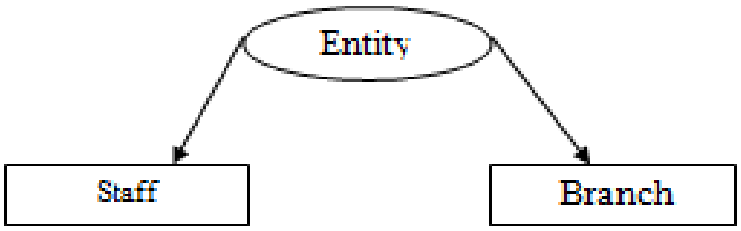
Beberapa konsep dasar dalam model ER yaitu :

1. Entitas (*Entity*)

   Entity adalah sekumpulan objek yang memiliki *property* yangsama, yang diidentifikasikan di dalam organisasi karenakeberadaannya yang bebas (*independent existence*)(Connolly dan Begg,2010, p372).

Sedangkan *entity occurence* adalah sebuah objekdari satu *entity* yang dapat diidentifikasikan secara unik(Connolly dan Begg, 2010, p373).

Setiap *entity*dilambangkan dengan sebuah persegi panjangyang diberi nama dari *entity* tersebut. Nama *entity* biasanyaadalah kata benda tunggal.



Gambar 2.2 Representasi Diagram dari Tipe *entity* (Connolly dan Begg, 2010, p374)

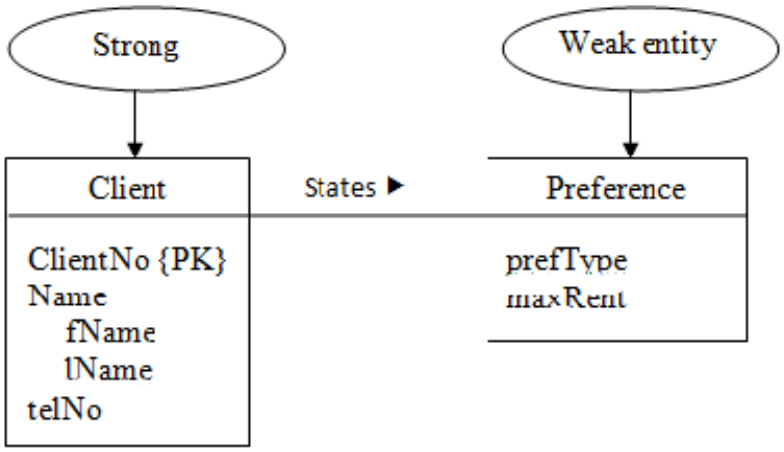
Entity dapat diklasifikasikan menjadi :

- *Strong Entity*

  Tipe *entity* yang keberadaannya tidakbergantung pada tipe *entity* lainnya (Connolly dan Begg, 2010,p383).

- *Weak Entity*

  Tipe *entity* yang keberadaannyabergantung pada tipe *entity* lainnya (Connolly dan Begg, 2010,p383).



Gambar 2.3 Representasi Diagram *StrongEntity* dan*WeakEntity* (Connolly dan Begg, 2010, p384).

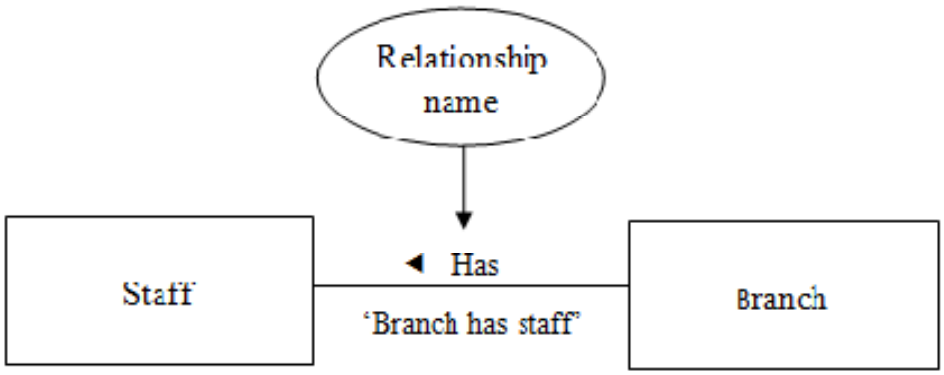
2. *Relationship*

Relationship adalah hubungan antar *entity* yang memiliki arti. Sedangkan *relationship occurrence*adalah sebuah hubungan yang dapat diidentifikasikan secara unik,yang meliputi sebuah kejadian (*occurrence*) dari setiap entity didalam *relationship*(Connolly dan Begg, 2010, p374).

Relationship digambarkan dengan sebuah garis yangmenghubungkan *entity* yang saling berhubungan. Garistersebut diberi nama sesuai dengan nama hubungannya dan diberitanda panah satu arah di samping nama hubungannya.

Biasanya sebuah *relationship* dinamakan denganmenggunakan kata kerja, seperti frase singkat yang meliputisebuah kata kerja seperti DisewaOleh. Sedangkan tanda panahditempatkan di samping nama *relationship* yang

mengindikasiarah bagi pembaca untuk mengartikan nama dari suatu*relationship* ditulis dengan huruf besar.



Gambar 2.4 Representasi Diagram dari *Relationship*(Connolly dan Begg, 2010, p376)

3. Atribut

   Atribut adalah *property* sebuah *entity* atau *relationship*(Connollydan Begg, 2010, p379).

   - *Domain Attribute*

     Domain attribute merupakan kumpulan dari nilai – nilaiyang diperbolehkan untuk satu atau lebih atribut. Misalnyauntuk atribut NIM harus diisi nilai antara satu sampai dua puluh.

   - *Simple and Composite Attributes*

     Simple attribute adalah atribut yang terdiri darikomponen tunggal dengan keberadaan yang bebas(Connollydan Begg, 2010, p379). Misalnya adalah jabatan, gajipada *entity* staff.

Composite attribute adalah atribut yang terdiri daribeberapa komponen, dan keberadaan komponentersebut adalah bebas (Connollydan Begg, 2010, p380). Misalnya adalah atribut pada alamat dengan nilai (*163 Main St, Glasgow, G11 9QX*), dapat dibagi menjadi nama jalan, kota, dan kodepos.

- *SingleValue and MultiValue Attributes*

  Single value attribute adalah atribut yang hanyamemiliki sebuah nilai untuk setiap tipe *entity*(Connollydan Begg, 2010, p380). Sebagian besar atributmerupakan *singlevalue*, contohnya pada *entity*cabang yang memiliki NoCabang (C001), noCabang iniadalah *single-value attribute.*

  Multivalue attribute merupakan atribut yangmemiliki banyak nilai untuk setiap *entity* (Connollydan Begg, 2010, p380). Misalnya pada NoCabang (C001) memilikiatribut NoTelp 021-432-5334 dan 021-526-3256. Padakasus ini NoTelp merupakan *multi-value attribute.*

- *Derived Attributes*

  Derived attributes merupakan sebuah atribut yangmerepresentasikan sebuah nilai yang berasal dari nilaisebuah atribut yang berhubungan atau set atribut, dantidak harus berada dalam tipe *entity* yang sama

(Connollydan Begg, 2010, p380). Misalnya nilai dari LamaPenyewaan dapat dihitung dari MulaiPenyewaan hingga SelesaiPenyewaan.

4. Key

- *Candidate key*

  Himpunan atribut yang minimal yangsecara unik mengidentifikasikan setiap *occurence* dari sebuah*entity* (Connollydan Begg, 2010, p381). Misalnya atribut BranchNo merupakan *Candidate key*untukentitas Branch. *Candidate key* harus mempunyai nilai yang unik untuk setiap *occurrence*dari suatu entitas.

- *Primary key*

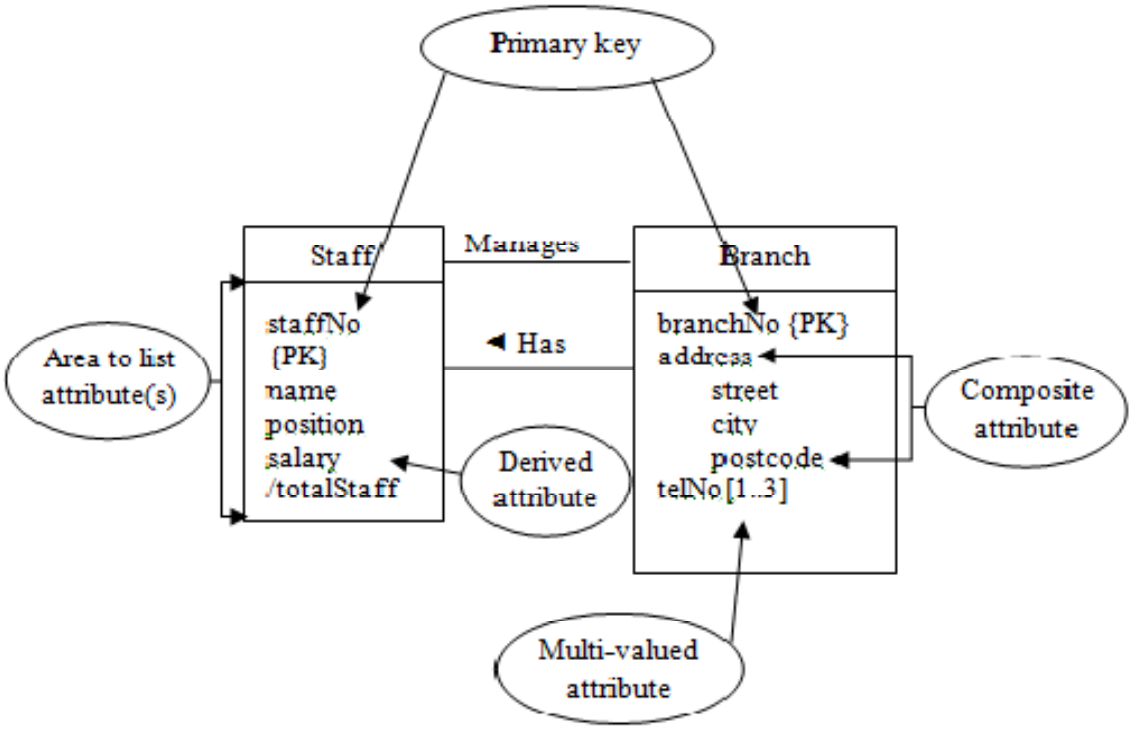
  Candidate key yang terpilih untukmengidentifikasikan secara unik setiap *occurence* dari sebuah*entity* (Connollydan Begg, 2010, p381). Pada setiap *entity* biasanyaterdapat lebih dari satu *candidate key* yang kemudian akan dipilihsalah satu untuk dijadikan *primary key*. Pemilihan *primary key*berdasarkan panjang atribut, jumlah minimal atribut, dankeunikannya.

- *Composite key*

  Sebuah *candidate key* yang terdiri atasdua atau lebih atribut(Connollydan Begg, 2010, p382).

- *Alternate key*

  Setiap *candidate key* yang tidak terpilihmenjadi *primary key*, atau biasa disebut dengan *secondary key*.



Gambar2.5 Representasi Diagram *Entity* Pegawai dan CabangBeserta Atributdan *Primary Key* (Connolly dan Begg, 2010, p382)

5. Batasan Struktural (*Structural Constraints*)

   Batasan yang menggambarkan pembatasan pada*relationship* seperti yang ada pada "*real world*" harus diterapkanpada *entity* yang ikut serta pada sebuah *relationship* dinamakan*multiplicity* (Connollydan Begg, 2010, p385).

   Multiplicity merupakan sejumlah kemunculan (*occurence*)yang mungkin dari sebuah tipe *entity* yang berhubungan dengankemunculan tunggal dari sebuah tipe

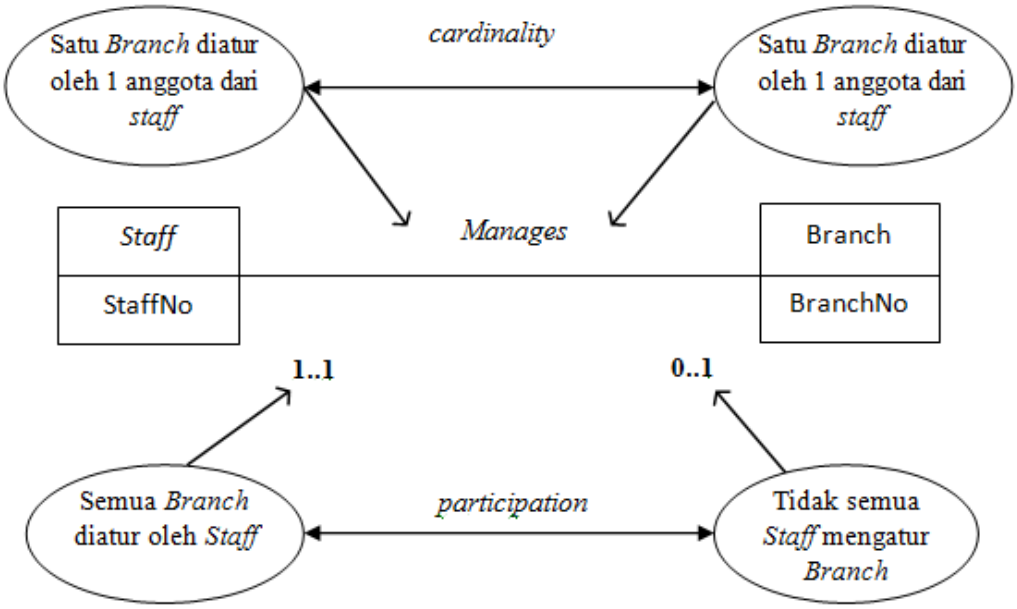
entity yang berhubunganmelalui relasi tertentu (Connollydan Begg, 2010, 385).

Derajat yang umum pada suatu *relationship* adalah *binaryrelationship*, yang terdiri atas :

- *One-to-one* (1:1) *Relationship*
- *One-to-many* (1:*) *Relationship*
- *Many-to-many* (*:*) *Relationship*

6. *Cardinality and Participation Constraints*

Multiplicity terdiri dari dua batasan yaitu *cardinality* dan*participation*. *Cardinality* menggambarkan jumlah maksimumrelasi yang mungkin terjadi dari sebuah *entity* yang berpartisipasidalam tipe relasi. *One-to-one* (1:1), *one-to-many* (1:*), dan *many-to-many* (*:*) merupakan cardinality dari relasi *binary*,*participation* menentukan apakah semua atau hanya sebagian dari*entity* yang berpartisipasi dalam relasi(Connollydan Begg, 2010, p390-p391).



Gambar 2.6 *Cardinality* dan *Participation* antara Branch dan Staff (Connolly dan Begg, 2010, p391)

2.1.6 Normalisasi

Normalisasi adalah suatu teknik untuk menghasilkan kumpulan relasi dengan properti yang diperlukan dan berguna untuk menyediakan kebutuhan data dari perusahaan (Connolly dan Begg, 2010, p416).

Langkah-langkah normalisasi dapat dijelaskan sebagai berikut (Connollydan Begg, 2010, p430-p436) :

1. *Unnormalized Form* (UNF)

   UNF merupakan suatu tabel yang berisikan satu atau lebih grup data yang berulang-ulang. UNF dilakukan dengan memindahkandata dari sumber informasi ke dalam format tabel dengan baris dankolom.

2. *First Normal Form* (1NF)

1NF merupakan sebuah relasi dimana setiap irisan antar baris dankolom berisikan satu dan hanya satu nilai saja. Cara mengubah daribentuk UNF ke 1NF adalah sebagai berikut:

- Tunjuk satu atau sekumpulan atribut sebagai kunci untuktabel *unnormalized*
- Identifikasi grup yang berulang dalam tabel *unnormalized*yang berulang untuk kunciatribut.
- Hapus grup yang berulang dengan cara memasukkan datayang semestinya ke dalam kolom yang kosong pada barisyang berisikan data yang berulang atau dengan caramenggantikan data yang ada dengan salinan dari kunciatribut yang sesungguhnya.

3. *Second Normal Form* (2NF)

2NF berdasarkan pada konsep ketergantungan fungsional penuh(*full functional dependency*) yang mengindikasikan bahwa, jika A dan B merupakan atribut dari sebuah relasi, B dikatakantergantung penuh terhadap A jika B tergantung secara fungsionalkepada A tetapi tidak pada proper dari *subset* dari A. 2NFmerupakan sebuah relasi dalam 1NF dan setiap atribut *non primarykey* bersifat *fully functionally dependent*pada *primary key*.

Caramerubah 1NF menjadi 2NF adalah sebagai berikut :

- Mengidentifikasi *primary key* untuk relasi 1NF
- Mengidentifikasi *functional dependency* dalam relasi

- Jika terdapat *partial dependency* terhadap *primary key*, makahapus dengan menempatkannya dalam relasi baru bersamadengan salinan determinannya.

4. *Third Normal Form* (3NF)

   3NF adalah suatu relasi yang ada dalam 1NF dan 2NF dan dimanatidak terdapat atribut *non primary key* yang bersifat *transitivelydependent* pada *candidate key*. Atribut yang tidak memberikan kontribusi terhadap penjelasan karakteristik primary key, akan dipindahkan ke sebuah tabel yang terpisah.

2.1.7 Waterfall Model

Menurut Pressman (2010, p39) salah satu model pengembangan sistem adalah dengan model *waterfall*. *Waterfall model* adalah model yang paling populer dan sering dianggap sebagai pendekatan klasik dalam daur hidup pengembangan sistem. Adapun tahapannya sebagai berkut :

1. *Communication*

   Pada tahap ini akan dilakukan inisiasi proyek, seperti menganalisis masalah yang ada dan tujuan yang akan dicapai. Selain itu dilakukan juga *requirements gathering*, dimana akan dikumpulkan *requirement* dari perusahaan melalui analisis wawancara.

2. *Planning*

   Tahap ini merupakan tahap dimana akan dilakukan estimasi mengenai kebutuhan-kebutuhan yang diperlukan untuk membuat

sebuah sistem. Selain itu, penjadwalan dalam proses pengerjaan juga ditentukan pada tahap ini.

3. *Modeling*

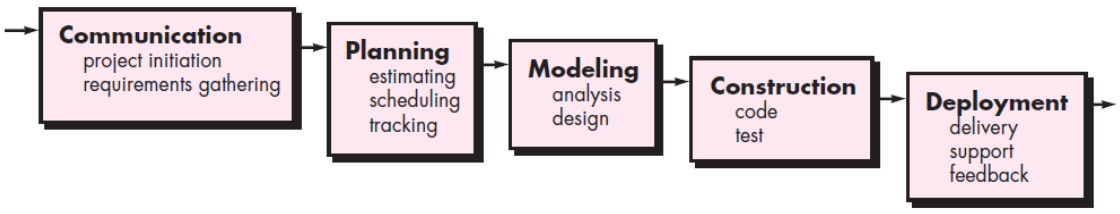
   Kemudian mulai masuk pada tahap perancangan dimana perancang menerjemahkan kebutuhan sistem ke dalam representasi untuk menilai kualitas sebelum tahap selanjutnya dikerjakan. Tahap ini lebih difokuskan pada perancangan basis data, perancangan struktur menu dan perancangan layar.

4. *Construction*

   Tahap ini merupakan tahap dimana perancangan diterjemahkan ke dalam bahasa yang dimengerti oleh mesin. Setelah itu dilakukan pengetesan / pengujian terhadap sistem yang telah dibuat.

5. *Deployment*

   Setelah proses pengkodean dan pengujian selesai, tahap selanjutnya adalah implementasi. Pada tahap ini juga dilakukan pemeliharaan, perbaikan, dan pengembangan agar sistem tersebut tetap dapat berjalan sebagaimana fungsinya.



Gambar 2.7 Waterfall Model (Pressman, 2010, p39)

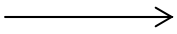
2.1.8 Data Flow Diagram (DFD)

Definisi *Data Flow Diagram* (DFD)

DFD adalah suatu teknik analisa struktur dimana *system analyst*bisa meletakkan semua representasi grafis dari proses data yangmelalui organisasi (Kendall, 2005, p192).

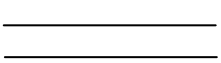
DFD mampumenggambarkan sistem informasi secara logikal maupun fisikal. DFDmemiliki empatjenis simbol, yaitu : aliran data (*data flow*),penyimpanan data (*data store*), proses (*proccess*), sumber (*source*). Keempatjenis komponen DFD tersebut adalah sebagai berikut:

1. Aliran data (*Data Flow*)

   Simbol : 

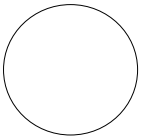
   Aliran data adalah data yang bergerak dan berpindah sebagaisuatu unit dari satu tempat ke tempat lain di dalam sistem.Aliran data bisa terdiri dari banyak bagian data individualyang digunakan secara bersamaan dan bergerak bersama kesuatu tujuan.

2. Penyimpanan data (*Data Store*)

   Simbol : 

   Peyimpanan data adalah data saat data tidak digunakan.Penyimpanan data bisa merepresentasikan satu atau banyaklokasi fisikal untuk data termasuk suatu *folder file*, satu ataulebih *file* berbasis komputer, atau suatu *notebook*.Konfigurasi fisikal tidak terlalu penting untuk mengertipergerakan dan penanganan data di dalam sistem.

3. Proses (*Process*)

   Simbol : 

   Proses adalah suatu pekerjaan atau aksi yang di kerjakan didata sehingga mereka ditransformasikan, disimpan, dandidistribusikan. Saat permodelan pemrosesan data dari sistem,tidak perlu diperhatikan proses di lakukan secara manual atauoleh komputer.

4. Sumber (*Source*)

   Simbol : 

   Sumber adalah tempat dan atau tujuan dari data. Sumberterkadang berarti *external entities* karena berada di luarsistem. Sekali diproses, data atau informasi meninggalkansistem dan pergi ke tempat lain.

Context Diagram

Context Diagram adalah Diagram aliran data dari lingkup darisistem organisasi perusahaan yang menunjukkan batasan, *externalentities* yang berikteraksi dengan sistem, dan informasi-informasi yangmengalir atara *entity* dan sistem(Valacich, George, dan Hoffer, 2001,p149).

Context Diagram hanya mempunyai satu proses yangmenggambarkan secara umum seluruh sistem. Proses tersebut di berilabel 0 (nol). Sumber menggambarkan lingkup lingkungan sistem.Karena peyimpanan data dari sistem secara konseptual ada di dalamsatu proses, maka tidak ada

penyimpanan data dalam *Context Diagram*(Valacich, George, dan Hoffer, 2001, p149-150).

Data Flow Diagram Level 0

Diagram Level 0 Adalah diagram yang merepresentasikanproses-proses sistem, aliran data, dan penyimpanan data dengan detaillevel tinggi (Valacich, George, dan Hoffer, 2001, p149-150).Proses-proses yang di gambarkan ke dalam Diagram level 0adalah sebagai berikut (Marakas, 2006, p122) :

- Proses yang menangani *inflow* dan atau *outflow* daripenyimpanan data
- Proses yang secara langsung bertanggung jawab atasproduksi dan atau distribusi data ke satu atau lebih *entity*sumber
- Proses yang secara langsung menangkap data dari satu ataulebih sumber
- Proses yang menyajikan *high level descriptor* dari operasitrasnformasi *multistep data*.

Kelebihan dari penggunan DFD adalah sebagai berikut :

- Kebebasan dari keterikatan implementasi secara teknis dari sistemyang terlalu awal
- Pengertian lebih mendalam dari ketersaling hubungan antar sistemdan subsistem

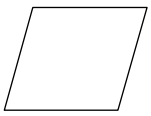
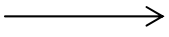
- Mengkomunikasikan pengetahuan sistem yang saat ini kepada usermelalui diagram aliran data
- Analisa dari sistem yang di tawarkan untuk menentukan apakahproses dan data yang dibutuhkan sudah ditentukan semua ataubelum.

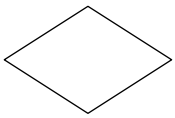
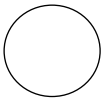
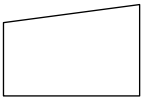
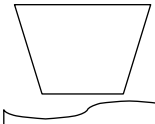
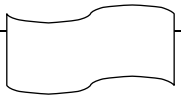
2.1.9 Flow chart

Flowchart adalah penggambaran secara grafik dari langkah-langkahdan urutan-urutan prosedur dari suatu progam. *Flowchart* menolong analis dan *programmer* untuk memecahkan lain masalah ke dalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian.

Simbol-simbol yang terdapat dalam *flow chart*:

Tabel 2.1 Flow Chart

Nama	Simbol	Arti
Input / Output		Merepresentasikan*input*data yang diproses dan*ouput* data yang telah diproses
Proses (*proccess*)		Merepresentasikanoperasi
Anak panah (*arrow*)		Merepresentasikan alur kerja

Keputusan (*decision*)		Keputusan dalamprogam
Sub Program (*predefined proccess*)		Rincian operasi berada ditempat lain
Persiapan (*preparation*)		Pemberian harga awal
Titik terminal (*terminal point*)		Awal atau akhir *flowchart*
Dokumen (*document*)		*Input* dan *output* dalam format yang dicetak
Tampilan (*display*)		*Input* atau *output* yang ditampilkan di monitor
Penghubung (*connector*)		Penghubungbagian-bagian*flowchart* di halaman lain
Manual input		*Input* yang dimasukkan secara manual dari*keyboard*
Operasi manual (*manualoperation*)		Operasi secara manual
Punched Tape		*Input* atau *output* yang menggunakan pita kertas

		berlubang

2.1.10 State Transition Diagram (*STD*)

STD merupakan suatu model diagram yang dibuat berdasarkan ketergantungan waktu dari sistem (*time-dependent*). Jika suatu sistem berhadapan dengan banyak data masukkan dari banyak *terminal* yang berbeda secara cepat, maka akan terjadi masalah ketergantungan waktu yang biasa dimiliki oleh aplikasi *real-time*.

Notasi *STD* terdiri dari :

- Keadaan (*state*) yang menggambarkan keadaan objek.Keadaan disimbolkan dengan kotak.
- Tanda panah (*arrow*) yang menggambarkan kejadian yangmengakibatkan perubahan *state* di dalam sistem. Keadaan disimbolkan dengan tanda panah.

Perubahan keadaan di dalam *STD* direpresentasikan denganmenghubungkan pasangan keadaan (*state*) yang sesuai dengan tanda panah.

2.1.11 Fact Finding

Pengembangan basis data biasanya menggunakan beberapa teknik *fact findin*g. Terdapat lima teknik *fact finding* yang sering digunakan yaitu :

1. *Examining Documentation*

   Memeriksadokumentasi dapatberguna ketikakita mencobauntuk mendapatkan beberapawawasan tentang bagaimanakebutuhanuntuk databasemuncul. Kitajuga dapat menemukanbahwa dokumentasidapat

membantuuntuk memberikan informasipada bagian dariperusahaanyang terkait denganmasalah.

2. *Interviewing*

   Wawancaraadalah teknik yang paling umumdigunakandan biasanyapaling bermanfaat. Wawancaradilakukan untuk mengumpulkaninformasi daritatap muka dengan seseorang.

3. *Observing the Enterprise in Operation*

   Pengamatanadalah salah satufakta yang palingefektifmenemukanteknikuntuk memahamisistem.Dengan teknik ini seseorangdapat berpartisipasi atau menontonseseorang melakukan kegiatanuntuk belajar tentangsistem.

4. *Research*

   Faktayang bergunauntuk risetaplikasidan masalah. Jurnal,bukuilmiah, dan internet(termasuk kelompok pengguna danpapan buletin) merupakan sumberinformasi yang dapat digunakan. Informasi tentangbagaimana orang laintelah memecahkanmasalah yang sama, ditambahapakahpaket perangkat lunakada untukmemecahkan ataubahkan sebagianmemecahkan masalah.

5. *Questionnaires*

   Teknikpencarian faktalainadalah untukmelakukan surveimelalui kuesioner. Kuesioneradalah dokumenyang memungkinkanfakta-faktayang dikumpulkandarisejumlahbesar orang. Ketika berhadapan denganaudiens yang besar, teknik ini dapat berguna secara efisien.

2.1.12 Interaksi Manusia dan Komputer

Menurut Sneiderman (2010, p32), ada lima faktor manusia terukur yang dapat dijadikan sebagai pusat evaluasi, yaitu :

1. Waktu belajar

   Waktu yang dibutuhkan oleh *user* untuk mempelajari cara relevan dalam mengerjakan tugas dengan lancar.

2. Kecepatan kinerja

   Waktu yang diperlukan untuk mengerjakan suatu tugas yang diberikan.

3. Tingkat kesalahan

   Berapa banyak kesalahan yang dilakukan oleh *user* dan kesalahan-kesalahan seperti apa yang bisa terjadi saat *user* mengerjakan tugas tersebut.

4. Daya ingat

   Kemampuan *user* mempertahankan pengetahuannya setelah jangka waktu tertentu.

5. Kepuasan subjektif

   Kepuasan *user* terhadap berbagai aspek dari sistem.

Menurut Shneiderman (2010, p88), terdapat delapan aturan emas dalam desain antarmuka yaitu:

1. Berusaha untuk konsisten

   Aturan ini merupakan aturan yang paling sering dilanggar, karena terdapat banyak bentuk konsistensi. Konsisten dalam hal urutan aksi,

istilah yang digunakan, menu, *layout*, penggunaan warna, tata letak, kapitalisasi, *font*, dan sebagainya.

2. Menyediakan kebutuhan universal

   Dengan memahami kebutuhan *user* yang bermacam-macam dan membuat desain fleksibel yang mendukung perubahan dalam konten.

3. Memberikan umpan balik yang informatif

   Untuk setiap aksi *user*, harus ada sistem umpan balik. Untuk aksi kecil yang sering dilakukan, respon dapat dibuat dengan sederhana. Sedangkan untuk aksi yang besar dan jarang dilakukan, respon harus dibuat lebih tegas dan jelas.

4. Desain dialog untuk menghasilkan penutupan atau keadaan akhir

   Urutan aksi hendaknya disusun ke dalam kelompok kategori awal, tengah, dan akhir. Umpan balik yang informatif dapat memberikan kepuasan pencapaian, rasa lega, sinyal untuk mempersiapkan diri memasuki kelompok kategori aksi selanjutnya.

5. Penawaran pencegahan dan penanganan kesalahan sederhana

   Usahakan dalam mendesain suatu sistem, diarahkan agar *user* tidak membuat kesalahan yang serius. Misalnya menyediakan pilihan menu, tidak mengizinkan karakter alfabet pada kotak entri numerik. Jika *user* melakukan kesalahan, sistem harus dapat mendeteksi kesalahan dan menawarkan instruksi yang sederhana, konstruktif, dan spesifik untuk perbaikan. Contoh, *user* tidak perlu mengetik ulang seluruh perintah, melainkan hanya memperbaiki bagian yang salah saja.

6. Mengizinkan pembalikan aksi yang mudah

   Pada suatu sistem aplikasi harus terdapat pembalikan aksi. Fitur ini dapat memperkecil kesalahan, selama *user* tahu bahwa aksi dapat dibatalkan. Pembalikan aksi dapat berupa tindakan tunggal, tugas data entry, atau serangkaian aksi seperti entri nama dan alamat.

7. Mendukung pusat kendali internal

   User yang sudah terbiasa dengan suatu aplikasi, biasanya ingin memiliki kendali atas antarmuka dan tanggapan dari aksinya. Aksi antarmuka yang tidak umum, urutan entri data yang membosankan, kesulitan dalam memperoleh informasi yang dibutuhkan, sertaketidakmampuan menghasilkan aksi yang diinginkan dapat menimbulkan keresahan dan ketidakpuasan pada *user*.

8. Mengurangi beban ingatan jangka pendek

   Manusia mempunyai keterbatasan dalam memproses informasi dengan waktu yang singkat. Oleh karena itu diperlukan tampilan yang sederhana, pengurangan jendela-gerak frekuensi, pemberian waktu pelatihan yang cukup untuk kode-kode, hafalan dan rangkaian aksi. Jika diperlukan, akses *online* untuk sintaks, singkatan, kode, dan informasi yang terkait harus disediakan.

2.2 Teori Khusus

2.2.1 Pemasaran

Pemasaran adalah proses sosial dan manajerial dimana pribadi atau organisasi memperoleh apa yang mereka butuhkan dan inginkan melalui penciptaan dan pertukaran nilai dengan yang lain. Dalam konteks bisnis yang lebih sempit, pemasaran mencakup menciptakan hubungan pertukaran muatan nilai dengan pelanggan yang menguntungkan. Karena itu, kita mendefinisikan pemasaran sebagai proses dimana perusahaan menciptakan nilai bagi pelanggan dan membangun hubungan yang kuat dengan pelanggan, dengan tujuan menangkap nilai dari pelanggan sebagai imbalannya (Kotler dan Armstrong, 2008, p6).

2.2.2 Pemesanan

Menurut Kamus Besar Bahasa Indonesia Edisi Ketiga (2002, p478), pemesanan adalah proses, perbuatan, cara memesan atau memesankan. Dalam bisnis atau perdagangan, pemesanan dinyatakan sebagai sesuatu yang terlibat dalam transaksi komersial untuk produk atau jasa tertentu. Dari sudut pandang pembeli disebut *order* pembelian. Dari sudut pandang penjual disebut sebagai suatu *sales order*.

2.2.3 Penjualan

Menurut Kamus Besar Bahasa Indonesia Edisi Ketiga (2002, p478), penjual adalah orang yang menjual. Sedangkan penjualan adalah proses, cara, perbuatan menjual, harga barang menurut harga bila barang itu dijual.

Penjualan adalah suatu usaha yang terpadu untuk mengembangkanrencana-rencana strategis yang diarahkan pada usaha pemuasan kebutuhandan keinginan pembeli, guna mendapatkan penjualan yang menghasilkanlaba.

2.2.4 E-commerce

Definisi E-commerce

Lahirnya web, yang difasilitasi oleh pertumbuhan dan semakin canggihnya *database*, telah mendorong terbentuknya *e-commerce* atau *electronic commerce*, yakni pembelian dan penjualan produk dan jasa melalui jaringan komputer. *E-commerce* membentuk kembali keselurahan industri dan mengubah hal paling mendasar dari sebuah perusahaan (William dan Sawyer, 2007, p436).

Terdapat tiga macam sistem *e-commerce*, yaitu (William dan Sawyer, 2007, p437-p439):

- Sistem *Business - to - Business* (B2B)

  Dalam sistem Sistem *Business - to - Business (B2B)*, sebuah bisnis menjual ke bisnis lain dengan menggunakan internet atau jaringan khusus, tujuannya untuk memotong biaya transaksi dan meningkatkan efisiensi. Selain menjual produk, ia juga bisa menjual iklan, pelatihan pegawai, riset pasar, dukungan teknis, layanan perbankan, dan layanan dukungan bisnis lainnya.

Salah satu contoh B2B yang paling terkenal adalah industri penjualan kendaraan bermotor yang dikembangkan oleh tiga besar pembuat kendaraan bermotor di AS – General Motor, Ford, dan DaimlerChryler.

- Sistem Business - to - Consumer (B2C)

  Dalam sistem Sistem *Business - to - Consumer (B2C)*, perusahaan menjual barang atau jasa kepada konsumen. Sistem *e-commerce* ini pada dasarnya menggeser karyawan di posisi penyedia dan bahkan took fisik (*"bricks-and-mortar"*). Contoh sistem B2C adalah Amazon.com dan BarnesanNoble.com, dan banyak lembaga keuangan serta pemerintah AS.

- Sistem Consumer - to - Consumer (C2C)

  Dalam sistem Sistem *Consumer - to - Consumer (C2C)*, konsumen menjual barang atau jasa secara langsung ke konsumen lain, kerap kali dengan bantuan pihak ketiga, semisal eBay, sebuah perusahaan lelang *online*. Pihak ketiga ini menjadi perantara atau mediator antara konsumen yang ingin membeli dan menjual, dan mereka mengambil sebagian kecil dari keuntungan penjual sebagai *fee* mereka.

  E-commerce C2C sangat mengurangi biaya; basis pelanggan pun lebih kecil, namun menguntungkan. Melalui C2C pebisnis kecil dapat mejual barang mereka tanpa harus mempunyai took yang memerlukan banyak biaya.

2.2.5 Web Database

Web database merupakan aplikasi penggunaan *web* sebagai*platform* yang menghubungkan pengguna dengan antarmuka satuatau lebih basis data.

Keuntungan dari penggunaan *web database* adalah sebagaiberikut (Connolly dan Begg, 2010, p1036-p1038) :

- Kesederhanaan (*Simplicity*)

  Dalam bentuk asli, HTML sebagai *markup language* sangatmudah untuk di pelajari oleh *developer* maupun penggunaakhir.

- Tidak bergantung pada *platform* (*Platform independence*)

  Salah satu alasan di buatnya aplikasi basis data berbasis *web* adalah karena pada umumnya *web-client* (*browser*) tidak bergantung pada *platform* sehingga jika suatu aplikasi akan di jalankan pada sistem operasi yang berbeda tidak memerlukan modifikasi.

- Grafis antarmuka pengguna (*Graphical User Interface* / GUI)

  Menyederhanakan dan meningkatkan pengaksesan basisdata, tetapi GUI memerlukan progam tambahan yang akanmenyebabkan ketergantungan pada *platform*. *Web browser*menyediakan GUI yang mudah digunakan untuk mengaksesbanyak hal. Memiliki GUI yang baik akan mengurangibiaya pelatihan untuk pengguna akhir.

- Standarisasi (*Standardization*)

  HTML standar secara *de facto* dimana seluruh *web* browserberada, memungkinkan sebuah dokumen HTML pada satumesin dibaca oleh

pengguna pada mesin lainnya di bagianlain dunia melalui koneksi di *internet* dan *web browser*.

- Dukungan lintas *platform* (*Cross-platform Support*)

  Web browser ada secara *virtual* untuk setiap tipe dari*platform* komputer memungkinkan pengguna pada sebagianbesar jenis komputer mengakses basis data dari manapun diseluruh bagian dunia.

- Memungkinkan Akses jaringan yang transparan(*Transparent Network Access*)

  Akses jaringan terlihat transparan bagi pengguna, kecualiuntuk spesifikasi URL, ditangani sepenuhnya oleh *webbrowser* dan *web server*. Dukungan *built-in* untuk jaringanakan menyederhanakan akses basis data dan mengeliminasikebutuhan perangkat lunak jaringan dan kompleksitas dari*platform* yang berbeda untuk saling berkomunikasi.

- Penyebaran bisa diukur (*Scalable Deployment*)

  Solusi berbasis *web* mampu membuat arsitektur *three-tier*yang menyediakan dasar untuk *scalability*. Denganmenyimpan aplikasi di *server* terpisah, maka *web*mengeliminasi waktu dan biaya yang berhubungan denganaplikasi penyebaran. Hal ini akan menyederhanakanpenanganan *upgrade* dan *Admin*istrasi pengaturan daribanyak *platform* yang melalui banyak kantor.

- Inovasi (*Innovation*)

Web memampukan organisasi untuk menyediakan jasa barudan mencapai konsumen baru melalui aplikasi yang dapatdiakses secara global.

Selain memiliki kelebihan-kelebihan yang telah dijelaskan diatas, *web database* juga tidak lepas dari beberapa kelemahan sebagaiberikut (Conolly dan Begg, 2010, p1038-p1040):

- Kehandalan (*Reliability)*

  Ketika sebuah perintah diberikan melalui *internet*, tidakdapat dipastikan perintah tersebut pasti akan dikirim.

- Keamanan (*Security*)

  Autentifikasi pengguna dan transmisi data sangat kritiskarena banyak pengguna yang tidak dikenal bisa mengaksesdata.

- Biaya (*Cost*)

  Biaya perawatan bisa menjadi semakin mahal bersamaandengan meningkatnya permintaan dari pengguna.

- *Skalabilitas (Scalability)*

  Aplikasi *web* bisa menghadapi *load* data yang sangatbanyak yang datang secara di terduga. Hal ini memerlukanpembangunan performa yang kuat dari arsitektur *server*yang sangat sulit diukur.

- Fungsi terbatas dari HTML (*Limited Functionality of HTML)*

Beberapa aplikasi basis data interaktif ada yang tidakterkonversi secara mudah ke aplikasi berbasis *web* selamamasih menyediakan kemudahan yang sama bagi pengguna.

- *Statelessness*

  Statelessness dari lingkungan *web* membuat pengaturan darikoneksi basis data dan transaksi pengguna menjadi sulit danmembutuhkan aplikasi untuk merawat informasi tambahan.

- *Bandwidth*

  Kegiatan di dalam *internet* memerlukan *bandwidth* bahkanuntuk tugas yang sangat sederhana sekalipun sehinggamasalah *bandwidth* menjadi penting.

- Kinerja (*Performance*)

  Banyak bagian dari *web database* yang kompleks berpusatpada interpretasi bahasa yang akhirnya membuatnya lebihlambat daripada basis data tradisional.

- Perangkat pembangunan yang belum siap (*Imaturity ofdevelopmet tools*)

  Pembangun aplikasi basis data untuk *web* secara cepat bisamengidentifikasi ketidaksiapan perangkat pembangunanyang ada. Teknologi *web* masih belum matang danpembangunan lingkungan yang lebih baik dibutuhkanuntuk meringankan beban dari *programmer* aplikasi.

2.2.6 Perangkat lunak yang digunakan

2.2.6.1 Dreamweaver

Dreamweaver merupakan perangkat lunak (*software*) pengembangan *website*yang lengkapdan manajemen program. Dreamweaver bekerja denganteknologi *web*seperti HTML, XHTML, CSS, JavaScript, dan PHP.

Dreamweaverjuga mencakupbanyak *tools*untuk mengelola*website*setelah membangun *website* tersebut. *Tools* yang ada dapat memeriksa*link* yang rusak, menggunakan *template*untuk mempersingkatperubahanhalaman*side-wide*,dan mengorganisasikan kembalisitus di dalam sebuah*flash* dengan*tool* manajemensitus program (McFarland, 2011, p2).

2.2.6.2 MySQL

MySQL adalah *relational database management system* yang melakukan semua pekerjaan untuk menyimpan, mengambil, mengelola, dan memanipulasi data. Berikut adalah kelebihan dari MySQL (Forta, 2006, p13):

- MySQL bersifat *open source*
- MySQL mempunyai kinerja yang cepat
- MySQL dipakai oleh banyak perusahaan atau organisasi
- MySQL mudah untuk di-*install*, *get up*, dan *run*.

2.2.6.3 Apache

Apache adalah *web server*modular, yangberartibahwa serverinti (yang pada dasarnya berperan untukmelayani dokumen HTML) dan dapat diperpanjang denganmenggunakan berbagaimodul-modul tambahan (Vugt, 2008, p326).

2.2.6.4 Web Server

Web server adalah suatu proses yang berjalan di komputer *host* untuk menunggu *HTTPrequests* yang masuk .

Web menyediakan fasilitas untuk menyimpan dan mengakses informasi dan *services*. Informasi dan *services* diakses dan disimpan di *website* yang diiplementasikan di *web servers*, atau secara alternatif *web servers* mungkin menyediakan rute (*gateway*) oleh informasi yang disimpan di sistem lain yang dapat diakses (Eaglestone dan Ridley, 2001, p209).

2.2.7 Bahasa Program yang digunakan

2.2.7.1 PHP

PHP adalah bahasa *script open source HTMLembedded* yang mendukung banyak *web server*, seperti *Apache HTTP Server* dan *Microsoft's Internet Information Server.* Bertujuanagar *web developer* untuk mampu menulis banyakhalaman dinamis secara cepat (Connoly dan Begg, 2010,p1043).

2.2.7.2 Jquery

Jquery adalah *cross browser* yang memberikan seperangkat fungsi di semua *browser*. *Jquery* dapat memberikan kontrol atas apa yang ada di halaman dengan membiarkan pengguna membuat dan menghapus halaman *HTML* kapan saja. *Jquery* memberikan akses pada elemen-elemen dalam halaman *web* tanpa menunggu semua gambar untuk dimuat. *Jquery* juga memiliki banyak pilihan animasi dan efek visual (Holzner, 2009, p2).

2.2.7.3 Java Script

Javascript adalah bahasa pemrogaman yang bisa digunakan untuk menambahkan fitur-fitur interaktif pada halaman *web*. *Javascript* memiliki kemampuan untuk membuat halaman *web* lebih interaktif dan menampilkan halaman pengguna secara lebih menarik karena *Javascript* mampu menciptakan antarmuka pengguna yang aktif dan memberikan *feedback* kepada pengguna saat pengguna melakukan navigasi di halaman *web*. Selain itu, *Javascript*memastikan hanya pengguna yang memiliki hak akses yang dapat memasuki halaman-halaman *web* tertentu yang bersifat rahasia(Connoly dan Begg, 2010,p1041).

2.2.7.4 Ajax

Ajax, yang merupakan singkatan dari*Asynchronous JavaScript and XML*, adalah satu setteknik untuk menciptakansitus*web* dan

aplikasi *web* yang sangat interaktif. Tujuannyaadalah untuk membuatapa yang ada di*web*tampak secaralokal denganmemberikan banyakpengalaman kepada user, menawarkanfitur yangbiasanya hanyamuncul dalamaplikasi *desktop*.

Penekanandalam aplikasiAjax adalahuntuk memperbaruihalaman *web*, menggunakandata yang diambil dariinternet, tanpamelakukan *refresh* padahalaman*web* di*browser*. Contohnyapada *Google Suggest*, dimanadaftar *drop-down*yang muncul di *browser*tanpa melakukan*refresh* halaman(Holzner, 2009, p5).