BAB II

LANDASAN TEORI

Sistem Informasi (SI) adalah susunan orang-orang, data, proses, komunikasi, dan teknologi informasi yang berinteraksi untuk mendukung dan memperbaiki operasi sehari-hari dalam bisnis, termasuk mendukung pemecahan masalah (*problem-solving*) dan pengambilan keputusan (*decision-making*) yg dibutuhkan oleh manajemen dan pengguna.

Sistem informasi adalah sekumpulan *hardware*, *software*, *brainware*, prosedur dan atau aturan yang diorganisasikan secara integral untuk mengolah data menjadi informasi yang bermanfaat guna memecahkan masalah dan pengambilan keputusan.

Selain penulis, telah ada beberapa skripsi yang telah membahas dan mengaplikasikan Sistem Informasi untuk Program studi Teknik Informatika universitas Atmajaya. Skripsi tersebut antara lain adalah Pembangunan Sistem Informasi Pengelolaan Sumber Daya Manusia (Indah, 2005), Pembangunan Sistem Informasi Enterprise Dalam Penerapan Konsep Manajemen Rantai Pasok Untuk Perusahaan Furniture (Christina, 2005).

Kelebihan yang diambil penulis pada skripsi ini adalah penulis mencoba menerapkan Sistem Informasi Terintegrasi pada Yayasan Putri Kerahiman. Sistem Informasi ini mencakup 7 departemen yang terdiri dari Departemen Panti Asuhan Hawai, Panti Asuhan Polomo, TK Nina, Pengembangan Keluarga, Poliklinik Robertus, BKSP, dan Wisma Senja Timur yang kemudian diintegrasikan ke dalam satu sistem utama untuk membantu proses

pengolahan data. Sistem Informasi Terintegrasi ini memudahkan pengurus yayasan untuk mengambil data dan membuat laporan bulanan dan tahunan, mengurangi waktu dan biaya pembuatan laporan.

II.1. Sistem Informasi *(Information System)*

Sistem informasi adalah sekumpulan komponen pembentuk sistem yang mempunyai keterkaitan antara satu komponen dengan komponen lainnya yang bertujuan menghasilkan suatu informasi dalam suatu bidang tertentu. Dalam sistem informasi diperlukannya klasifikasi alur informasi, hal ini disebabkan keanekaragaman kebutuhan akan suatu informasi oleh pengguna informasi. Kriteria dari sistem informasi antara lain, fleksibel, efektif dan efisien.

Sistem informasi dibangun oleh enam blok pembangun yang lebih dikenal dengan 6 *building blocks*. Blok pembangun sistem informasi yaitu :

a) Blok *Input*

Masukan memiliki data masukan ke dalam sistem informasi. *Input* disini termasuk metode-metode dan media untuk menangkap data yang akan dimasukkan yang dapat berupa dokumen-dokumen dasar.

b) Blok *Output*

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

c) Blok Model

Blok model terdiri dari kombinasi prosedur, logika dan model matematik yang akan memanipulasi data

input dan data tersimpan di basis data dengan cara yang sudah tertentu untuk menghasilkan keluaran yang diinginkan.

d) Blok Basis Data

Basis data merupakan kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya. Data perlu disimpan di dalam basis data untuk keperluan penyediaan informasi lebih lanjut.

e) Blok Teknologi

Teknologi merupakan kotak alat dalam sistem informasi. Teknologi digunakan untuk menerima *input*, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran dan membantu pengendalian dari sistem secara keseluruhan.

f) Blok Kendali

Pengendalian perlu dirancang dan diterapkan untuk menyakinkan bahwa hal-hal yang dapat merusak sistem dapat dicegah ataupun bila terlanjur terjadi kesalahan-kesalahan dapat langsung cepat diatasi.

II.1.1. Pengembangan Sistem Informasi

Dalam membuat dan mengembangkan suatu sistem informasi harus melalui beberapa langkah. Proses pengembangan melalui beberapa tahapan mulai dari perencanaan sistem tersebut sampai dengan penerapan sistem, pengoperasian sistem dan pemeliharaan sistem.

Bila saat sistem dioperasikan timbul suatu permasalahan yang tidak memungkinkan untuk diatasi hanya dengan pemeliharaan sistem maka dilakukan kembali

perencanaan sistem. Inilah yang disebut dengan siklus hidup sistem (*system life cycle*). Adapun tahapan siklus hidup sistem yaitu :

1. Perencanaan Sistem

Pada tahap ini dilakukan perencanaan sistem informasi yang akan dibuat.

2. Analisis Sistem

Analisis sistem adalah penguraian dari suatu sistem informasi yang utuh ke dalam bagian-bagian komponennya dengan maksud untuk mengidentifikasikan dan mengevalusi permasalahan, kesempatan, hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikannya.

Pada tahap ini dilakukan pengidentifikasian masalah (*identify*), memahami kerja dari sistem yang ada (*understand*), menganalisis sistem (*analyze*), dan membuat laporan hasil analisis (*report*).

3. Perancangan Sistem

Desain sistem menentukan bagaimana suatu sistem akan menyelesaikan apa yang mesti diselesaikan, tahapan ini menyangkut mengkonfigurasi dari komponen-komponen perangkat keras dan perangkat lunak dari sistem sehingga instalasi dari sistem akan benar-benar memuaskan rancang bangun yang telah diterapkan pada akhir tahap analisis sistem.

Komponen sistem informasi yang dirancang adalah *input*, *output*, model, basis data, teknologi, kontrol.

3.1 Desain *Input*

Di sini akan didesain bentuk dari dokumen dasar yang digunakan untuk menangkap data, kode *input* yang digunakan, dan bentuk tampilan dari alat *input*. *Input* yang didesain dapat ditentukan dari DFD sistem yang telah dibuat. *Input* dari DFD ditunjukkan oleh arus data dari suatu kesatuan luar ke suatu proses dan bentuk tampilan input pada alat *input* ditunjukkan oleh suatu proses memasukkan data.

3.2 Desain *Output*

Output adalah produk dari sistem informasi yang dapat dilihat. Pada tahap perancangan ini, *output* yang dimaksud adalah *output* yang berupa tampilan di media kertas atau di layar monitor. Yang harus diperhatikan dalam perancangan *output* adalah tipe *output* dan format *output*.

Tipe *output* terdiri dari *output intern* yang dimaksudkan untuk mendukung kegiatan manajemen perusahaan. *Output* tipe ini akan disimpan sebagai arsip perusahaan, misal berupa laporan terinci.

Tipe *output* yang lain adalah *output ekstern* yaitu *output* yang didistribusikan kepada pihak luar yang membutuhkannya, misal faktur dan cek. Format atau bentuk laporan dapat berupa keterangan, tabel, grafik.

3.3 Desain Model

Alat untuk merancang model yaitu DFD (*Data Flow Diagram*), ERD (*Entity Relationship Diagram*), DD (*Data Dictionary*).

DFD (*Data Flow Diagram*)

DFD atau Diagram Arus Data merupakan diagram untuk menggambarkan arus dari sistem. DFD sering dipakai untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertimbangkan lingkungan fisik di mana data tersebut mengalir atau lingkungan fisik di mana data tersebut disimpan.

ERD (*Entity Relationship Diagram*)

ERD atau Diagram Hubungan Entitas menggambarkan hubungan antar entitas/data dan tidak menggambarkan proses sistem. ERD terdiri dari entitas, relasi antar entitas, dan atribut entitas.

3.4 Desain Basis Data

Basis data / database adalah kumpulan data yang saling berhubungan satu dengan yang lainnya, tersimpan di penyimpan luar komputer dan digunakan perangkat lunak tertentu untuk memanipulasinya.

Dalam desain ini dilakukan pengidentifikasian file-file yang diperlukan oleh sistem informasi. File-file tersebut dapat dilihat pada desain model yang digambarkan dalam bentuk DFD.

3.5 Desain Teknologi

Teknologi dipakai untuk menerima *input*, menjalankan model, menyimpan dan mengakses data, mengahasilkan dan mengirimkan keluaran, dan membantu pengendalian sistem secara keseluruhan.

Tiga bagian utama teknologi yaitu perangkat keras (*hardware*), perangkat lunak (*software*), dan teknisi (*brainware*). Dalam desain ini dilakukan pengidentifikasian jenis teknologi yang dibutuhkan dan jumlah yang diperlukan oleh sistem informasi.

3.6 Desain Kontrol

Pengendalian (kontrol) perlu diterapkan dalam sistem informasi karena sangat berguna untuk mencegah atau menjaga hal-hal yang tidak diinginkan. Pengendalian yang baik merupakan cara bagi suatu sistem informasi untuk melindungi dirinya dari hal-hal yang merugikan. Pengendalian tersebut terdiri dari pengendalian umum (misal keamanan data), dan pengendalian aplikasi (pengendalian *input*, *output*, pengolahan).

4. Implementasi Sistem

Merupakan tahap lanjutan dari tahap-tahap diatas. Pada tahap ini sudah memasuki tahap pengkodean.

5. Pemeliharaan Sistem

Pada tahap ini sistem informasi yang dibuat sudah bisa digunakan dan di sini dilakukan pemeliharaan dari sistem yang sudah ada agar selalu berjalan sesuai fungsinya.

Dalam tahapan perancangan sistem, terdapat komponen desain basis data (*database*).

II.2. Basis Data *(Database)*

Database adalah sekelompok *file* yang disimpan bersama untuk dapat digunakan oleh beberapa aplikasi. Hal ini berarti bahwa data (*file*) tersebut hanya perlu sekali saja disimpan secara fisik di dalam sistem komputer. Data di dalam *database* adalah terpisah dari program-program aplikasinya.

Basis data (*database*) merupakan kumpulan data yang berhubungan dengan suatu obyek, topik atau tujuan khusus tertentu. Beberapa istilah yang sering digunakan dalam basis data, yaitu :

1. *Database* merupakan kelompok fakta atau keterangan yang dapat diatur berhubungan dengan pengolahan referensi.
2. *Field* merupakan tempat peletakan suatu informasi.
3. *Record* merupakan kumpulan fakta dan keterangan yang berhubungan dengan informasi yang ada di dalam *field.*
4. *Primary key* adalah bagian yang berguna sebagai sarana indentifikasi yang unik dari suatu baris pada suatu relasi.

Fungsi basis data adalah untuk mempermudah kita dalam menyimpan informasi atau data-data yang ada dan menyajikannya kembali dengan bentuk yang baik, teratur dan pasti. Agar basis data yang telah ada dapat disajikan kembali maka kita harus memperhatikan cara pengolahan dan penyimpanan basis data. Cara pengolahan dan penyimpanan basis data yang kita gunakan harus sesuai dengan bahasa pemrograman yang dipakai.

II.2.1. DBMS (*Database Management System*)

DBMS adalah suatu kumpulan dari data-data yang saling berelasi dan diakses oleh suatu set program yang mempunyai kemampuan untuk menambah data, menghapus data, mengedit data.

II.2.1.1. Tujuan DBMS (*Database Management System*)

Tujuan dalam merancang dan menyusun sistem *management database* adalah sebagai berikut :

a. Menyediakan tempat penyimpanan massal untuk data yang relevan.

b. Membuat agar pemakainya mudah mendapatkan data.

c. Memungkinkan respon yang segera atas permintaan data dari pemakai.

d. Melakukan modifikasi terakhir dengan segera pada *database*.

e. Menghapus data yang berlebihan.

f. Memungkinkan penggunaan secara serentak dalam beberapa pemakai.

g. Memungkinkan perkembangan lebih lanjut dalam sistem *database*.

h. Melindungi data dari kerusakan fisik dan pemakaian yang tidak diotorisasi.

II.2.1.2. Keuntungan DBMS (*Database Management System*)

Keuntungan *Database Management System* (DBMS) adalah sebagai berikut:

a. Lebih banyak data yang dapat dipertimbangkan dalam menghasilkan informasi bagi manajemen untuk mengambil keputusan.

b. Informasi dapat disajikan secara lebih cepat untuk manajemen.

Bagi kebanyakan professional teknologi informasi, SQL Server adalah salah satu nama database yang paling populer di dunia saat ini.

II.3. SQL Server

SQL Server adalah sebuah sistem berarsitektur terbuka yang memungkinkan pengembang program memperluas dan menambahkan fungsi-fungsi ke dalam database tersebut. SQL Server mendukung pengoperasian database seperti *backup*, replikasi, *profiler*, dan sebagainya.

Microsoft SQL Server adalah sebuah sistem manajemen basis data relasional (RDBMS) produk Microsoft. Bahasa kueri utamanya adalah Transact-SQL yang merupakan implementasi dari SQL standar ANSI/ISO yang digunakan oleh Microsoft dan Sybase. Umumnya SQL Server digunakan di dunia bisnis yang memiliki basis data berskala kecil sampai dengan menengah, tetapi kemudian berkembang dengan digunakannya SQL Server pada basis data besar.

II.3.1. SQL Replikasi *(SQL Replication)*

SQL Replikasi adalah proses *copy* data antar database pada server yang sama atau server yang berbeda yang terhubung dengan LAN, WANs, atau internet

SQL Replikasi adalah proses berbagi data antar database dalam lokasi yang berbeda. Dengan menggunakan replikasi, user dapat membuat *copy* database dan membagi *copy* tersebut dengan user yang lain sehingga mereka

dapat melakukan perubahan pada *copy* database lokal tersebut yang kemudian perubahan tersebut dapat disinkronisasikan dengan database asal.

II.3.2. Keuntungan SQL Replikasi

Keuntungan SQL Replikasi adalah sebagai berikut:

a. *User* yang berada di lokasi gegrafi yang berbeda dapat bekerja pada *copy* lokal data yang berasal dari *copy database* pusat.
b. Replikasi *database* dapat mendukung rencana pemulihan kesalahan dengan cara menduplikasi data dari server *database* lokal ke server *database* lain. Jika server utama gagal, aplikasi dapat terus melanjutkan operasi dengan menggunakan *copy* data dari replikasi.
c. Dapat melakukan *backup database* secara otomatis dengan menyimpan data replikasi pada komputer lain. Berbeda dengan metode *backup* tradisional yang mencegah user untuk mengakses *database* pada saat *backup*, replikasi mengijinkan *user* untuk melanjutkan *user* melakukan perubahan pada *database* tersebut.
d. *User* dapat mereplikasi *database* pada server jaringan tambahan dan menugaskan kembali *user* untuk menyeimbangkan beban antar server. *User* dapat mengakses *database* replikasi secara konstan sehingga mengurangi total lalu lintas jaringan.
e. *Database* replikasi memasukkan transaksi *database* tertentu ke dalam kumpulan internal tabel replikasi manajemen sehingga kemudian dapat disinkronisasikan ke *database* asal.

II.3.3. Platform SQL Server untuk Replikasi

Microsoft SQL Server menggunakan model industri penerbitan untuk merepresentasikan komponen dan proses dalam arsitektur replikasi. Industri penerbitan buku mempunyai komponen: *Distributor* dan *Agent* yang membawa buku/hasil penerbitan ke *Subscribers*. *Subsribers* mengambil hasil *copy* dari penerbitan tersebut dan membaca artikel dalam *copy* tersebut. Berikut adalah gambar alur industri penerbitan yang juga digunakan pada model kerja SQL Server Replikasi.

Entitas model SQL Server Replikasi adalah sebagai berikut:

a. Publisher

Publisher adalah server yang menyediakan data untuk dibagikan ke server atau *database* lain. Untuk memungkinkan data direplikasi, *publisher* harus mengidentifikasi data apa yang telah berubah pada *subscriber* selama proses sinkronisasi berlangsung. Bergantung pada tipe replikasi, perubahan data diidentifikasi pada instans data yang berbeda.

b. Distributor

Distributor adalah server yang mengatur distribusi data pada sistem replikasi. Fungsi distributor bervariasi tergantung pada tipe replikasi. Dua tipe *distributor* yaitu *remote distributor* dan *local distributor*. *Remote distributor* adalah *distributor* yang terpisah dari *publisher* dan dikonfigurasikan sebagai sebuah *distributor*. Sedangkan *local distributor* adalah

sebuah server yang dikonfigurasikan sebagai sebuah *publisher* sekaligus sebagai *distributor*.

c. Agent

Agent adalah agen/proses yang bertanggung jawab dalam meng-*copy* dan mendistribusikan data antara *publisher* dan *subscriber*. Ada beberapa tipe agen yang mendukung berbagai tipe replikasi, yaitu *Snapshot Agent, Log Reader Agent, Distribution Agent*, dan *Merge Agent*. Berikut ini penjelasan untuk masing-masing agen:

- *Snapshot Agent*

  Snapshot agent adalah agen replikasi yang membuat *file snapshot*, menyimpan *snapshot* pada *distributor*, dan mencatat informasi tentang status sinkronisasi dalam *database* distribusi. *Snapshot agent* digunakan pada semua tipe replikasi (*Snapshot, Transactional*, dan *Merge Replications*).

- *Log Reader Agent*

  Log reader agent adalah agen replikasi yang memindahkan transaksi yang digunakan untuk replikasi dari log transaksi pada *Publisher* ke *database* distribusi. *Log reader agent* tidak digunakan pada *Snapshot replication*.

- *Distribution Agent*

  Distribution agent adalah agen replikasi yang memindahkan *job snapshot* dari *database* distribusi ke *subscribers* serta memindahkan semua transaksi yang menunggu untuk didistribusikan ke *subscribers*. *Distribution agent* digunakan pada *Snapshot replication* dan *Transactional replication*.

- *Merge Agent*

  Merge agent adalah agen replikasi yang menginisialisasi *job snapshot* dari tabel *database publication* ke *subscribers* serta menggabungkan perubahan data yang terjadi sejak pembuatan *snapshot* awal. *Merge agent* hanya digunakan pada *Merge replication*.

d. Subscriber

Subscriber adalah server yang menerima data yang dipublikasikan dari server atau *database* lain. Modifikasi data pada level *subscriber* dapat disebarkan kembali ke *publisher*. Pada kasus tertentu, *subscriber* dapat mempublikasikan kembali data kepada *subscriber* lain.

e. Article

Sebuah *article* dapat berupa *objek database* seperti tabel, *view, index view, stored procedure, user defined function*.

f. Publication

Publication adalah koleksi dari banyak *article*.

g. Subscription

Subscription adalah kumpulan data yang akan diterima oleh server atau *database*.

Perubahan pada *subcription* di *publisher* dapat direplikasikan kepada *subscriber* melalui *PUSH subscription* atau *PULL subscription*. Pada *PUSH subscription, publisher* secara periodik akan mengirim transaksi atau mensinkronkan semua perubahan ke *subsccriber*. Pada *subscription* PULL, *subscriber* secara periodik akan terhubung ke *database* distribusi dan meminta informasi/transaksi terbaru.

h. *Database* Distribusi

Database distribusi adalah sebuah sistem *database* yang berada pada *distributor* dan tidak memiliki tabel *user*. *Database* ini digunakan untuk menyimpan *job snapshot* dan semua transaksi yang menunggu untuk didistrubsikan ke *subscribers*.

II.3.4. Tipe Replikasi

Ada tiga tipe replikasi yaitu replikasi *snapshot*, replikasi *transactional*, dan replikasi *merge*.

II.3.4.1. Replikasi *Snapshot*

Replikasi *snapshot* juga dikenal sebagai replikasi statis. Replikasi *snapshot* meng*copy* dan mendistribusikan data serta *objek database* secara sama persis dengan aslinya.

Karakteristik replikasi *snapshot* :

- Perubahan data pada *subscriber* tidak di*update* kepada *subscriber* secara kontinyu.
- *Subscriber* di*update* dengan modifikasi data yang lengkap dan bukan dengan transaksi individu.
- Menyebarkan perubahan pada *subscriber* membutuhkan waktu yang lama karena merupakan satu kali proses atau proses yang terjadwal.

Kondisi ideal untuk menggunakan replikasi *snapshot* adalah sebagai berikut:

- Data atau *objek database* bersifat statis atau tidak berubah secara rutin.
- Replikasi mengacu pada tabel yang tidak berubah secara rutin.
- Total data yang akan direplikasi adalah kecil.

- *User* sering bekerja pada mode *offline* dan tidak selalu tertarik dengan data terbaru.

II.3.4.2. Replikasi Transactional

Replikasi *transactional* juga dikenal sebagai replikasi dinamis. Pada replikasi *transactional*, modifikasi *publication* pada *publisher* disebarkan kepada *subscriber* secara bertahap. Pada replikasi *transactional*, semua perubahan pada *article* disimpan dalam *database* distribusi kemudian dikirim ke *subscribers* secara berurut.

Karakteristik replikasi *transactional* :

- *Publisher* dan *subscriber* selalu melakukan sinkronisasi.
- Batasan transaksi dilindungi, contohnya jika ada modifikasi pada 5 baris data, maka kelima modifikasi tersebut disebarkan kepada *subscriber* atau tidak sama sekali.
- *Publisher* dan *subscriber* sebaiknya selalu terhubung.

Kondisi ideal untuk menggunakan replikasi *transactional* adalah sebagai berikut:

- Replikasi *database* dengan informasi *rollup*, *database* dengan penjualan lokal, penjualan pusat atau *database* persediaan yang di*update* dan direplikasikan kepada berbagai tempat.
- *Subscriber* selalu membutuhkan data terbaru untuk melakukan proses.

II.3.4.3. Replikasi Merge

Replikasi *merge* memiliki keuntungan dari replikasi *snapshot* dan *transactional* yaitu perubahan data pada *publisher* dan *subscriber* dapat disinkronisasi. Mulanya *snapshot* melakukan *request* pada *subscriber* kemudian SQL server melacak perubahan data pada *publisher* dan *subscriber*. Data tersebut disinkronkan secara terjadwal atau pada saat ada permintaan.

Karakteristik replikasi *merge*:

- *Update* data dilakukan sendiri pada lebih dari satu server.
- Data digabungkan secara terjadwal atau sesuai permintaan.
- Mengijinkan *user* untuk bekerja *online/offline* dan mensinkronkan *publisher* dan *subscriber* secara terjadwal atau saat ada permintaan.

Kondisi ideal untuk menggunakan replikasi *merge* adalah sebagai berikut:

- *Multi subscriber* harus melakukan *update* data baik pada saat yang sama atau pada waktu yang berbeda serta menyebarkan perubahan tersebut kepada *publisher*.

II.3.5. Implementasi Replikasi

Ada beberapa cara untuk melakukan replikasi dan mengawasi replikasi berdasarkan tipe replikasi. Namun secara umum, replikasi meliputi langkah-langkah berikut:

- Mengubah *account MSSQLServer* dan *SQLServerAgent* menjadi *account* dengan hak akses *administrator*.

- Mendaftar server yang akan direplikasi sebagai *remote server*.
- Konfigurasi replikasi.
  Konfigurasi meliputi langkah-langkah sebagai berikut:
  1. Melakukan konfigurasi *publisher* dan *distributor*. *Distributor* bisa berada pada server yang sama atau server yang berbeda.
  2. Melakukan konfigurasi *SQL Server Agent*.
  3. Menentukan lokasi *folder snapshot*.
  4. Membuat *publication* berdasarkan data, subset data, dan *objek database*.
  5. Menentukan tipe replikasi yang akan digunakan dan tipe *database subscriber*.
  6. Membuat *push/pull subscription* baik pada *publisher* atau *subscriber*.
  7. Menentukan *subscriber* dan *database* pada *subscriber*.
  8. Melakukan konfigurasi kapan sinkronisasi akan terjadi dan pilihan yang akan digunakan pada *subscription*
- Membuat dan mengaplikasikan *snapshot*.
  SQL server 2000 membuat sebuah *snapshot* dan skema data lalu menyimpannya dalam lokasi *folder snapshot*. Setelah *subscription* dibuat, *snapshot* diaplikasikan dan dijalankan pada jadwal yang sudah dikonfigurasi. Membuat sebuah *publication* atau sebuah *snapshot* dapat dijalankan secara manual. Agen *snapshot* bertanggung jawab dalam membuat *file snapshot* dan menyimpannya dalam lokasi *folder snapshot*.

- Modifikasi data replikasi.
  Bergantung pada tipe replikasi dan pilihan replikasi, *subscriber* akan dapat memodifikasi data setelah *snapshot* diaplikasikan dan menyebarkan perubahan kembali kepada *publisher* atau *subscriber* lainnya.
- Sinkronisasi dan menyebarkan data.
  Sinkronisasi menunjuk pada penyebaran perubahan data antara *subsciber* dan *publisher*. Bagaimana data disinkronkan bergantung pada tipe replikasi yang digunakan. Pada replikasi *snapshot*, sebuah file *snapshot* diaplikasikan kembali pada *subscriber*. Pada replikasi *transactional*, semua modifikasi data melalui *insert/update/delete* didistribusikan antara *publisher* dan *subscriber*. Sedangkan pada replikasi *merge*, modifikasi data pada berbagai server digabungkan. Masalah (konflik), jika ada, dapat dideteksi dan dipecahkan secara otomatis.

II.4. Microsoft .NET

Microsoft .NET adalah *software* yang menghubungkan informasi, orang, sistem dan *device*/perangkat yang menjangkau *client*, *server*, dan *tool* pengembang.

Microsoft .NET disusun oleh :

1) .NET Framework, digunakan untuk membangun dan menjalankan semua macam *software* termasuk aplikasi berbasis *Web*, aplikasi *smart client*, dan layanan *web* XML (*Extensible Markup Language*). Menyediakan komponen untuk berbagi data melalui *Network*

menggunakan *platform* protokol independen seperti XML, SOAP, dan HTTP.

2) *Tool-tool* pengembang seperti Microsoft Visual Studio .NET yang menyediakan IDE (*Integrated Development Environment/* lingkungan pengembangan terintegrasi) untuk memaksimalkan produktivitas pengembangan menggunakan .NET Framework.
3) Serangkaian server termasuk Microsoft Windows Server 2003, Microsoft SQL Server, dan Microsoft BizTalk Server yang terintegrasi, untuk menjalankan, mengoperasikan dan mengelola *services Web* dan aplikasi berbasis *Web*.
4) *Software client* seperti Windows XP, Windows CE dan Microsoft Office XP yang membantu pengembang untuk menyebarkan dan mengelola aplikasinya.

II.4.1 .NET Framework

.NET Framework adalah lingkungan untuk membangun, *deploying*/menyebarkan, dan menjalan *sevices Web* dan aplikasi lainnya.

.NET Framework disusun oleh dua komponen utama, yaitu *Common Language Runtime* dan *.NET Framework Class Library* (pustaka class .NET Framework). Secara sederhana .NET Framework adalah *platform* tunggal dimana semua orang dapat mengembangkan aplikasi menggunakan suatu sistem yang mirip JVM (*Java Virtual Machine*), namun tidak ada penghalang bahasa dengan .NET sehingga aplikasi dapat dikembangkan menggunakan bahasa: VB, C++, C#, J# dan 20 bahasa pemrograman lainnya yang kompatibel dengan .NET Framework.

Tujuan dari .NET Framework adalah :

- Menyediakan lingkungan pemrograman berorientasi objek.
- Menyediakan lingkungan untuk menjalan suatu kode yang meminimalkan konflik saat *software deployment*/ disebarkan dan *versioning*/tentang versi.
- Menyediakan lingkungan untuk menjalan suatu kode yang menjamin keamanan saat kode dijalankan, termasuk kode yang dibuat oleh pihak yang tidak diketahui atau pihak ketiga yang setengah dipercaya.
- Menyediakan lingkungan untuk menjalan suatu kode yang dapat mengeliminasi masalah performa dari lingkungan *scripted* dan *interpreted*.

II.4.2. Visual Basic .NET (VB.NET)

Visual Basic .NET adalah bahasa pemrograman untuk membuat aplikasi berbasis Windows, aplikasi *form Web* ASP.NET (*Active Server Pages*), layanan Web XML dan aplikasi *mobile* seperti komputer Palm dan Pocket PC.

Visual Basic .NET dibangun di atas fondasi .NET Framework. Visual Basic .NET merupakan langkah berikut dari hasil evolusi bahasa Visual Basic 6.0. Sebagai hasil dari evolusi Visual Basic 6.0, VB.NET dapat mengatasi keterbatasan yang dihadapi dalam Visual Basic 6, antara lain masalah *deployment*/penyebaran, kekurangan *interoperability* dengan *platform* lainnya, kekurangan dalam OOP (*Object Oriented Programming*), keterbatasan dalam pengembangan internet, windows API (*Application Interface Programming*) yang merugikan, dan kekurangan dalam penerapan *multithreading*.