BAB II KAJIAN TEORI

2.1 Sistem Informasi

Sistem adalah bagian-bagian yang saling berkaitan yang beroperasi bersama untuk mencapai beberapa sasaran atau maksud (Davis, 1995:68). Informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam mengambil keputusan saat ini atau yang akan datang. (Davis, 1995:28).

Sistem Informasi Pemasaran menurut Margianti(1994:8) adalah sebuah sistem berdasarkan komputer yang bekerja bersama dengan sistem formasi fungsional yang lain untuk mendukung manajemen perusahaan dalam memecahkan yang berhubungan dengan pemasaran produk perusahaan.

2.2 *E-Commerce*

Menurut Carter (2002:2) *E-commerce* adalah semua bentuk transaksi yang berhubungan dengan aktivitas komersial, baik itu organisasi maupun individual yang berdasarkan pengelolahan dan transmisi data yang terdigitalisasi, termasuk teks, suara, dan gambar visual. Pada umumnya *E-commerce* mengacu pada aplikasi perdagangan yang menggunakan media internet untuk melakukan transaksi *online*, seperti untuk belanja produk dan jasa. Contohnya terjadinya ketika konsumen memesan tiket , produk berwujud maupun tidak berwujud melalui internet.

Ada empat definisi *e-commerce* menurut Kotler (2009:529) :

1. B2B (*Business-to-business*)

   Hal ini berarti kedua pihak perusahaan melakukan transaksi bisnis dalam menjalankan usahanya.

2. B2C (*Business-to-Consumer*)

   Definisi ini berarti transaksi *e-commerce* merupakan transaksi di mana para pembeli merupakan konsumen individu.

3. C2C (*Consumer-to-Consumer*)

   Disini konsumen menjual secara langsung satu sama lain melalui iklan elektronik atau situ pelanggan.

4. C2B (*Consumer-to-Business*)

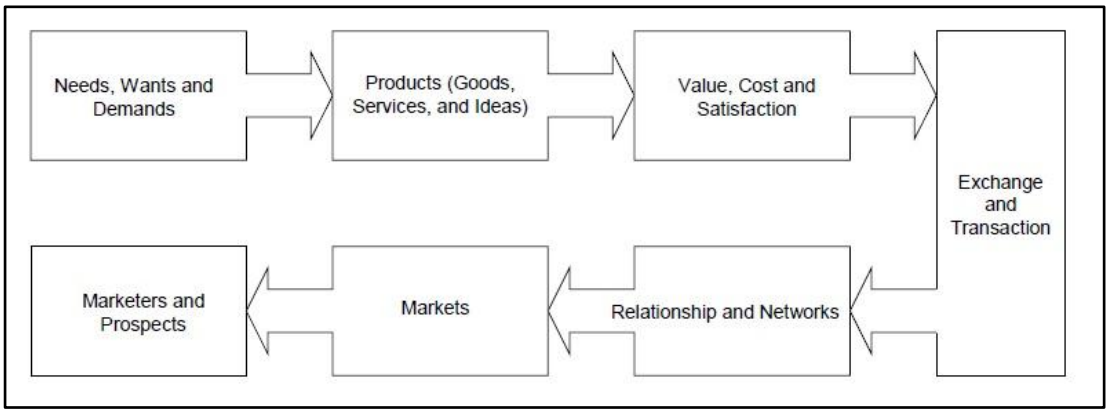
   Dalam kategori ini individu menjual barang-barang atau jasa ke perusahaan.

Definisi dari *E-Commerce* menurut Kalakota dan Robinson (2001:47) dapat ditinjau dalam perspektif berikut:

1. Dari perspektif komunikasi, *E-Commerce* adalah pengiriman barang, layanan, informasi, atau pembayaran melalui jaringan komputer atau melalui peralatan elektronik lainnya.
2. Dari perspektif proses bisnis, *E-Commerce* adalah aplikasi dari teknologi yang menuju otomatisasi dari transaksi bisnis dan aliran kerja.
3. Dari perspektif layanan, *E-Commerce* merupakan suatu alat yang memenuhi keinginan perusahaan, konsumen, dan manajemen untuk memangkas biaya layanan ketika meningkatkan kualitas barang dan meningkatkan kecepatan pengiriman.
4. Dari perspektif *online*, *E-Commerce* menyediakan kemampuan untuk membeli dan menjual barang ataupun informasi melalui *Internet* dan sarana *online* lainnya.

2.3 Pemasaran

Menurut Kotler (1997:9), pemasaran adalah suatu proses sosial dan manejerial yang dilakukan oleh individu maupun kelompok untuk mendapatkan atau memenuhi kebutuhan dan keinginannya, dengan menciptakan, menawarkan, atau saling bertukar nilai produk yang dihasilkan dengan produk dari pihak lain.



Gambar 1 Konsep Inti Pemasaran

Ada beberapa faktor yang berpengaruh dalam pemasaran suatu produk oleh produsen. Menurut Kotler (1997:92) ada 4 faktor yang berpengaruh dalam pemasaran suatu produk yang dikenal dengan nama *four* Ps. Faktor-faktor tersebut adalah :

1. *Products*, berupa barang, jasa maupun gagasan yang dihasilkan oleh produsen.
2. *Price*, yaitu harga dari produk yang dihasilkan oleh produsen. *Price* seringkali sangat berpengaruh pada produk kebutuhan sehari-hari yang dapat dengan mudah dicari penggantinya.
3. *Promotion*, merupakan suatu bentuk komunikasi dan promosi antara produsen kepada pelanggan tentang suatu produk.
4. *Place*, kegiatan yang dilakukan oleh produsen agar produknya dapat dengan mudah disediakan oleh produsen serta dapat diperoleh target pasar yang dituju.

2.4 CRM *(Customer Relationship Management)*

2.4.1 Pengertian CRM *(Customer Relationship Management)*

CRM adalah infrastruktur yang memungkinkan penggambaran dan peningkatan nilai pelanggan, dan cara yang benar yang digunakan untuk memotivasi pelanggan berharga untuk tetap setia. (Dyche, 2002:4). *Customer Relationship Management* (CRM) digunakan untuk mendefinisikan proses menciptakan dan mempertahankan hubungan dengan konsumen bisnis atau pelanggan. CRM adalah proses mengidentifikasi, menarik, membedakan dan mempertahankan *customer*. (Strauss, 2001:285)

2.4.2 Manfaat CRM *(Customer Relationship Management)*

Keuntungan bagi *customer*, banyaknya pilihan produk yang ditawarkan oleh perusahaan-perusahaan membuat konsumen sulit untuk membuat pilihan. Prinsip dasar dari CRM adalah pengurangan pilihan. Hal ini berdasarkan ide bahwa *customer*, ingin berlangganan pada toko, *mall* dan *service provider* yang sama karena efisiensi yang didapatkan.

CRM bersifat *cost effective*, karena lebih murah untuk mempertahankan seorang *customer* daripada untuk mendapatkan seorang *customer*, dan juga karena lebih mudah serta murah menjual lebih banyak produk kepada seorang *customer* daripada menjual jumlah yang sama kepada dua *customer*. Keuntungan lainnya dari CRM adalah pengaruh positif dari komunikasi mulut ke mulut oleh konsumen-konsumen yang puas. (Strauss, 2001:298).

2.4.3 Tujuan CRM (*Customer Relationship Management*)

Menurut Kalakota dan Robinson (2001:173) tujuan dari kerangka bisnis CRM adalah sebagai berikut:

1. Menambah hubungan yang telah ada untuk menambah pendapatan

   Perusahaan memandang pelanggan secara luas untuk memaksimalkan hubungan diantara mereka sehingga dapat meningkatkan profibilitas perusahaan dengan mengidentifikasi, menarik, dan mempertahankan pelanggan yang potensial.
2. Menggunakan informasi yang terintegrasi untuk pelayanan yang terbaik.

   Dengan menggunakan informasi pelanggan untuk memberikan pelayanan yang lebih baik bagi kebutuhannya, maka pelanggan tidak perlu berulang kali meminta informasi yang mereka butuhkan kepada perusahaan sehingga menghemat waktu dan mengurangi frustasi mereka.
3. Memperkenalkan saluran proses dan prosedur yang konsisten dan dapat ditiru.

   Dengan perkembangan saluran komunikasi bagi pelanggan, maka semakin banyak karyawan yang terlibat dalam transaksi penjualan, sehingga perusahaan harus memperbaiki konsistensi proses dan prosedurat.

2.5 *Direct Marketing*

Menurut Kotler (1997:718), *direct maketing* saat ini dipandang sebagai suatu metoda pemasaran yang cukup efektif, hal ini disebabkan oleh beberapa hal, antara lain makin berharganya waktu bagi manusia, kemajuan teknologi yang menyebabkan ruang dan waktu bukan lagi merupakan suatu hambatan yang signifikan, dan efisiensi yang terus dilakukan oleh perusahaan-perusahaan belakangan ini.

Menurut Kotler (1997:720), keuntungan penggunaan *direct marketing*, antara lain :

1. Belanja dari rumah merupakan sesuatu yang menyenangkan.
2. Menghemat waktu.
3. Pemilihan produk dan produsen yang lebih banyak.

Sedangkan untuk pihak produsen :

1. Produsen dapat mengumpulkan *database* tentang pelanggannya sehingga mereka dapat mengirimkan informasi sesuai dengan kebutuhan dan keinginan dari pelanggan tersebut.
2. Produsen dapat menjaga hubungan dengan masing-masing dari pelanggannya.
3. Bisa merupakan langkah yang *cost effective* dimana penawaran hanya diberikan kepada pelanggan-pelanggan yang sesuai dan cocok dengan minat, kebutuhan dan keinginan dari pelanggan tersebut.
4. Pemilihan media komunikasi yang sesuai.
5. Hasil dari penawaran mereka dapat diukur.

2.6 *Cross-Selling*

Cross-selling adalah upaya untuk meningkatkan jumlah produk atau jasa yang pelanggan gunakan dalam suatu perusahaan. *Cross-selling* produk dan jasa kepada pelanggan yang ada membutuhkan biaya yang lebih rendah dari biaya untuk memperoleh pelanggan baru, karena perusahaan sudah memiliki hubungan dengan pelanggan. Sebuah implementasi yang tepat dari *cross-selling* hanya dapat dicapai jika ada suatu infrastruktur informasi yang memungkinkan manajer untuk menawarkan produk dan layanan kepada pelanggan yang cocok dengan kebutuhan mereka, tetapi belum dijual kepada mereka (Kamakura et al, 2003:145).

Selain itu, *cross-selling* juga efektif untuk retensi pelanggan dengan meningkatkan biaya peralihan dan meningkatkan loyalitas pelanggan, sehingga secara langsung berkontribusi terhadap keuntungan yang didapat dari pelanggan dan *lifetime value*. Semakin banyak layanan perusahaan yang digunakan oleh pelanggan, semakin tinggi pula biaya peralihan ke perusahaan lain, yang pada akhirnya akan mengarah pada kesetiaan dan kepemilikan pelanggan atas perusahaan. (Kamakura et al, 2003:148).

2.7 Algoritma Apriori

Algoritma apriori termasuk jenis aturan asosiasi pada *data mining.* Aturan yang menyatakan asosiasi antara beberapa atribut disebut *affinity analysis* atau *market basket analysis.* (Kusrini dan Luthfi, 2009:149). Analisis asosiasi atau *association rule mining* adalah teknik *data mining* untuk menemukan aturan asosiatif antara suatu kombinasi *item*. Aturan asosiasi biasanya dinyatakan dalam bentuk :

{roti, mentega} → {susu} {*support* = 40%, *confidence* = 50%}

Aturan tersebut berarti "50% dari transaksi di *database* yang memuat *item* roti dan mentega juga memuat *item* susu. Sedangkan 40% dari seluruh transaksi yang ada di *database* memuat ketiga *item* itu."

Dapat juga dikatakan: "Seorang konsumen yang membeli roti dan mentega punya kemungkinan 50% untuk membeli susu. Aturan ini cukup signifikan karena mewakili 40% dari catatan transaksi selama ini."

Analisis asosiasi didefinisikan suatu proses untuk menemukan semua aturan asosiasi yang memenuhi syarat minimum untuk *support* (*minimum support*) dan syarat minimum untuk *confidence* (*minimum confidence*). (Kusrini dan Luthfi, 2009:150)

Metodologi dasar analisis asosiasi terbagi menjadi dua tahap (Kusrini dan Luthfi, 2009:150) :

1. Analisis pola frekuensi tinggi

   Tahap ini mencari kombinasi *item* yang memenuhi syarat minimum dari nilai *support* dalam *database*.

   Rumus untuk memperoleh nilai *support* :

   Support (A) = $\sum transaksi\ berisi\ A / \sum transaski$

   Sementara itu, nilai *support* dari 2 *item* diperoleh dari rumus :

   Support (A, B) = $\sum transaksi\ berisi\ A\ dan\ B / \sum transaski$

   Sebagai contoh, ada *database* dari transaksi seperti ditunjukkan di tabel I (Kusrini dan Luthfi, 2009:151).

Tabel I Daftar Transaksi

Transaksi	*Item* yang dibeli
1	Susu, Teh, Gula
2	Teh, Gula, Roti
3	Teh, Gula
4	Susu, Roti
5	Susu, Gula, Roti
6	Teh, Gula
7	Gula, Kopi, Susu
8	Gula, Kopi, Susu
9	Susu, Roti, Kopi
10	Gula, Teh, Kopi

Dalam *database* transaksional, biasa direpresentasikan seperti Tabel II (Kusrini dan Luthfi, 2009:151).

Tabel II Representasi Data Transaksi dalam *Database* Transaksional

ID	*Item* yang dibeli
1	Susu
1	Teh
1	Gula
2	Teh
2	Gula
2	Roti
3	Teh
3	Gula
4	Susu

ID	*Item* yang dibeli
4	Roti
5	Susu
5	Gula
5	Roti
6	Teh
6	Gula
7	Gula
7	Kopi
7	Susu

ID	*Item* yang dibeli
8	Gula
8	Kopi
8	Susu
9	Susu
9	Roti
9	Kopi
10	Gula
10	Teh
10	Kopi

Dan dalam bentuk tabular, maka akan seperti tabel III (Kusrini dan Luthfi, 2009:152).

Tabel III Format Tabular Data Transaksi

Transaksi	Teh	Gula	Kopi	Susu	Roti
1	1	1	0	1	0
2	1	1	0	0	1
3	1	1	0	0	0
4	0	0	0	1	1
5	0	1	0	1	1
6	1	1	0	0	0
7	0	1	1	1	0
8	0	1	1	1	0
9	0	0	1	1	1
10	1	1	1	0	0

Aturan yang kuat adalah aturan-aturan yang melebihi kriteria *min. support* dan *min. confidence*. Misalnya seorang analis menginginkan aturan yang memiliki *support* lebih dari 20% dan *confidence* lebih dari 60%.

Sebuah *itemset* adalah himpunan *item-item* yang ada dalam I, dan *k-itemset* adalah *itemset* yang berisi *k item.* Misalnya {Teh, Gula} adalah sebuah *2-itemset* dan {Teh, Gula, Roti} merupakan *3-itemset.*

Tabel IV menunjukkan calon *2-itemset* dari data transaksi pada Tabel I. (Kusrini dan Luthfi, 2009:154). *Minimum Support* = 20% (1/5).

Tabel IV Calon *2-itemset*

Kombinasi	Jumlah	***Support***
Teh, Gula	5 dari 10 transaksi	5/10 = 50%
~~Teh, Kopi~~	1 dari 10 transaksi	1/10 = 10%
~~Teh, Susu~~	1 dari 10 transaksi	1/10 = 10%
~~Teh, Roti~~	1 dari 10 transaksi	1/10 = 10%
Gula, Kopi	3 dari 10 transaksi	3/10 = 30%
Gula, Susu	4 dari 10 transaksi	4/10 = 40%
Gula, Roti	2 dari 10 transaksi	2/10 = 20%
Kopi, Susu	3 dari 10 transaksi	3/10 = 30%
~~Kopi, Roti~~	1 dari 10 transaksi	1/10 = 10%
Susu, Roti	3 dari 10 transaksi	3/10 = 30%

Dari tabel IV, kombinasi yang memenuhi syarat : F2 = {Teh, Gula}, {Gula, Kopi}, {Gula, Susu}, {Gula, Roti}, {Kopi, Susu},

{Susu, Roti}. Kombinasi ini dapat kita gabungkan menjadi calon *3-itemset* seperti Tabel V di bawah ini. (Kusrini dan Luthfi, 2009:154). *Minimum Support* = 20% (1/5).

Tabel V Calon *3-itemset*

Kombinasi	Jumlah	*Support*
~~Teh, Gula, Kopi~~	1 dari 10 transaksi	1/10 = 10%
~~Teh, Gula, Susu~~	1 dari 10 transaksi	1/10 = 10%
Gula, Susu, Kopi	2 dari 10 transaksi	2/10 = 20%
~~Gula, Susu, Roti~~	0 dari 10 transaksi	0/10 = 0%
~~Gula, Kopi, Roti~~	0 dari 10 transaksi	0/10 = 0%
~~Kopi, Susu, Roti~~	1 dari 10 transaksi	1/10 = 10%

Dengan demikian kombinasi yang memenuhi syarat : F3 = {Gula, Susu, Kopi}.

2. Pembentukan aturan asosiasi

Setelah semua pola frekuensi tinggi didapatkan, kemudian dicari aturan asosiasi yang memenuhi syarat minimum untuk *confidence* dengan menghitung *confidence.*

Nilai *confidence* dari aturan A → B diperoleh dari rumus berikut:

$$Confidence = p(B|A) = \frac{\sum transaksi \quad berisi \quad A \; dan \; B}{\sum transaski \quad \text{berisi} \quad A}$$

Dari F3 yang telah ditemukan, dapat dilihat besarnya nilai *support* dan *confidence* seperti tampak pada Tabel VI. (Kusrini dan Luthfi, 2009:155). *Minimum Confidence* = 60% (3/5).

Tabel VI Calon Aturan Asosiasi dari F3

Aturan	*Confidence*	
~~Jika membeli gula dan susu, maka akan membeli kopi.~~	2/4	50%
Jika membeli gula dan kopi, maka akan membeli susu.	2/3	67%
Jika membeli kopi dan susu, maka akan membeli gula.	2/3	67%

Dari F2 yang telah ditemukan, calon aturan asosiasinya dapat dilihat di Tabel VII. (Kusrini dan Luthfi, 2009:155). *Minimum Confidence* = 60% (3/5).

Tabel VII Calon Aturan Asosiasi dari F2

Aturan	*Confidence*	
Jika membeli teh, maka akan membeli gula.	5/5	100%
Jika membeli gula, maka akan membeli teh.	5/8	62,5%
~~Jika membeli gula, maka akan membeli kopi.~~	3/8	37,5%

Aturan	*Confidence*	
Jika membeli kopi, maka akan membeli gula.	3/4	75%
~~Jika membeli gula, maka akan membeli susu.~~	4/8	50%
Jika membeli susu, maka akan membeli gula.	4/6	67%
~~Jika membeli gula, maka akan membeli roti.~~	2/8	25%
~~Jika membeli roti, maka akan membeli gula.~~	2/4	50%
Jika membeli kopi, maka akan membeli susu.	3/4	75%
~~Jika membeli susu, maka akan membeli kopi.~~	3/6	50%
~~Jika membeli susu, maka akan membeli roti.~~	3/6	50%
Jika membeli roti, maka akan membeli susu.	3/4	75%

Aturan asosiasi final dapat dilihat pada Tabel VIII. (Kusrini dan Luthfi, 2009:156).

Tabel VIII Aturan Asosiasi Final

Aturan	*Support*	*Confidence*
Jika membeli teh, maka akan membeli gula.	50%	100%
Jika membeli gula, maka akan membeli teh.	50%	62,5%
Jika membeli susu, maka akan membeli gula.	40%	67%
Jika membeli kopi, maka akan membeli gula.	30%	75%
Jika membeli kopi, maka akan membeli susu.	30%	75%
Jika membeli roti, maka akan membeli susu.	30%	75%
Jika membeli gula dan kopi, maka akan membeli susu.	20%	67%
Jika membeli kopi dan susu, maka akan membeli gula.	20%	67%

2.8 ASP.NET

Menurut Dan Hurwitz dan Jesse Liberty (2002:3) ASP.NET adalah nama telah diberikan *Microsoft* kepada kombinasi dari dua pengembangan teknologi *web*: *web form* dan *web service*. *Active Server Pages* .NET (sering disingkat sebagai ASP.NET) adalah sebuah teknologi layanan *web* dinamis, aplikasi *web*, dan XML *web service* yang dikembangkan oleh Microsoft sebagai pengganti *Active Server Pages* (ASP) yang telah lama. Teknologi ini berbasis .NET *Framework* dan dibangun di atas *Common Language Runtime* (CLR), sehingga para *programmer* dapat menulis kode ASP.NET dengan menggunakan semua bahasa pemrograman .NET.

2.9 Basis Data

Basis Data adalah kumpulan data (*elementer*) yang secara logik berkaitan dalam mempresentasikan fenomena / fakta secara terstruktur dalam *domain* tertentu untuk mendukung aplikasi pada sistem tertentu. Basis Data adalah kumpulan data yang saling berhubungan yang merefleksikan fakta-fakta yang terdapat di organisasi (Hariyanto, 2004:4)

2.9.1 Sistem Manajemen Basis Data

Sistem manajemen basis data atau DBMS (*Database Management System*) adalah perangkat lunak untuk mendefinisikan, menciptakan, mengelola dan mengendalikan pengaksesan basis data. Fungsi sistem manajemen basis data saat ini yang paling penting adalah menyediakan basis untuk sistem informasi manajemen. (Hariyanto, 2004:4)

2.9.2 *Microsoft SQL Server™*

Microsoft SQL Server adalah RDBMS berkinerja tinggi yang dirancang untuk mendukung pengolahan transaksi bervolume tinggi (seperti pemasukan pesanan *online*, inventori, akutansi, dan *manufacturing*), juga sebagai *platform* untuk *data warehouse* dan aplikasi pendukung keputusan (seperti aplikasi analisis penjualan). *SQL Server* menyediakan banyak *client*, kakas dan antarmuka jaringan untuk sistem operasi.

SQL Server merupakan bagian paket terpadu *Microsoft Back Office* yang menyediakan kakas terpadu yang lengkap untuk mendukung aplikasi *enterprise* yang meliputi kakas pengembangan, kakas manajemen sistem, komponen sistem tersebar dan sebagainya.

Bahasa pendefinisian dan menipulasi basis data di *MS SQL Server* merupakan perluasan SQL standar. Bahasa SQL perluasan ini disebut *Transact-SQL*. *Transact-SQL* digunakan untuk mendefinisikan

basis data serta memanipulasi objek-objek basis data secara deklaratif seperti standar SQL serta dilengkapi fasilitas pemrograman prosedural sebagai perluasan. *Transact-SQL* memiliki peningkatan kemampuan DDL (*Data Definition Language*) dibanding standar SQL yang dapat ditabelkan sebagai berikut (Hariyanto, 2004:13) :

Tabel IX Peningkatan Kemampuan DDL yang dimiliki *Transact-SQL*

No	Fitur	Pernyataan yang ditingkatkan
1	Pendefinisian tabel	*ALTER TABLE* *CREATE TABLE* *Constraints* *IDENTITY Property* *Data Integrity*
2	Pendefinisian *view*	*CREATE VIEW*
3	*Dropping* objek	Pernyataan *all drop* *DROP DEFAULT* *DROP PROCEDURE* *DROP TABLE*
4	Pendefinisian terenkripsi : *procedure*, *trigger*, dan *view*	*CREATE PROCEDURE* *CREATE TRIGGER* *CREATE VIEW* *syscomments*
5	*Parameter stored procedure*	*CREATE PROCEDURE*
6	Objek-objek lokal dan global : prosedur dan tabel	*CREATE PROCEDURE* *CREATE TABLE*

2.9.3 *Entity Relationship*

Model ER (*Entity Relationship Model*) adalah model data konseptual tingkat tinggi untuk perancangan basis data. Model data konseptual adalah himpunan konsep yang mendeskripsikan struktur basis data, transaksi pengambilan dan pembaruan basis data. Model ER adalah persepsi terhadap dunia nyata sebagai terdiri dari objek-objek dasar yang disebut entitas, *relationship*, dan atribut. (Hariyanto, 2004:165).

Komponen pokok model ER adalah :

1. Entitas (*entity*)

   Entitas memodelkan objek-objek yang berada di perusahaan / lingkungan. Entitas dapat dapat berupa objek kongkret di dunia nyata seperti Mahasiswa, Pekerja, Mobil, dan sebagainya, namun entitas dapat juga berupa objek abstrak seperti Rekening. (Hariyanto, 2004:166).

2. *Relationship*

   Relationship memodelkan koneksi/hubungan di antara entitas-entitas.Tipe-tipe *relationship* yaitu :

   a. *Relationship* keberadaan (misalnya karyawan mempunyai anak).
   b. *Relationship* fungsional (misalnya professor mengajar mahasiswa).
   c. *Relationship* kejadian (misalnya pembeli membeli pesanan). (Hariyanto, 2004:176)

3. Atribut-atribut (properti-properti)

   Memodelkan properti-properti dari entitas dan *relationship*. Atribut adalah properti atau ciri atau karakteristik dari tipe entitas yang dipentingkan di satu sistem / organisasi. Atribut juga akan diasosiasikan dengan *relationship*.

   Kita mengklasifikasikan tipe-tipe entitas berdasarkan semantiks kumpulan atribut bersama yang digunakan di semantiks perusahaan. Contohnya, penduduk memiliki atribut serupa seperti nama, tanggal lahir, jenis kelamin, alamat, dan sebagainya. (Hariyanto, 2004:169)

4. Konstrain-konstrain integritas

   Konstrain-konstrain ketentuan validitas. Terdiri dari :

   a. Konstrain Kardinalitas

Konstrain kardinalitas menyatakan rasio jumlah entitas terhadap entitas lain yang diasosiasikan pada *relationship*, yaitu :

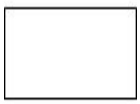
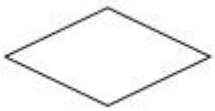
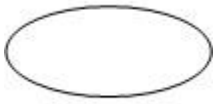
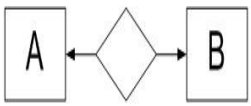
i. Satu-ke-satu (*one-to-one,* 1 – 1).
ii. Satu-ke-banyak (*one-to-many*, 1 – N).
iii. Banyak-ke-banyak (*many-to-many*, M – N).

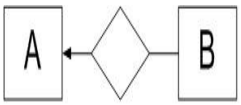
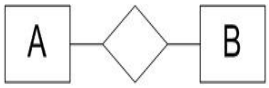
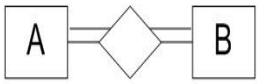
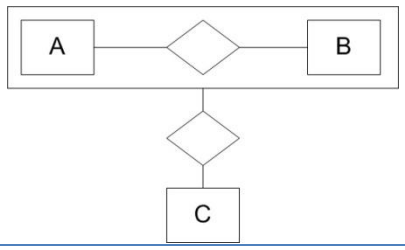
b. Konstrain partisipasi

Konstrain partisipasi menyatakan apakah keberadaan entitas bergantung terhadap entitas lain melalui *relationship*. Terdapat dua jenis partisipasi, yaitu partisipasi total dan partisipasi parsial.

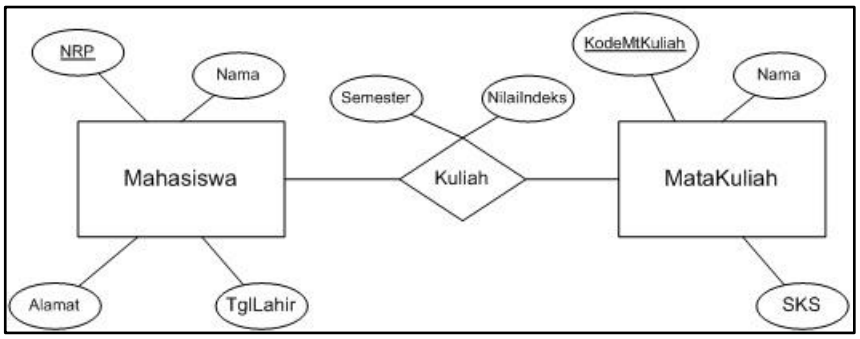
Partisipasi total adalah jika keberadaan entitas memerlukan keberadaan entitas yang diasosiasikan suatu *relationship* tertentu. Selain itu, partisipasi adalah partisipasi parsial.(Hariyanto, 2004:177)

Tabel X Model *Entity Relationship Diagram*

No	Komponen	Gambar	Keterangan
1	Entitas		Entitas menggambarkan individu suatu objek pada dunia nyata.
2	Relasi		Hubungan antara sejumlah entitas.
3	Atribut / properti		Atribut mendeskripsikan properti dari entitas.
4	Relasi satu-ke-satu		Setiap entitas pada himpunan entitas A berhubungan paling banyak dengan satu entitas B.

No	Komponen	Gambar	Keterangan
5	Relasi satu-ke-banyak		Setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas B.
6	Relasi banyak-ke-banyak		Setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas B.
7	Partisipasi total		Setiap anggota dari himpunan entitas berpartisipasi sedikitnya satu relasi.
8	Agregasi		Mengakomodasikan kondisi dimana relasi yang secara kronologis mensyaratkan telah adanya relasi lain

Berikut ini adalah contoh model *ER-Diagram* (Hariyanto, 2004:193)



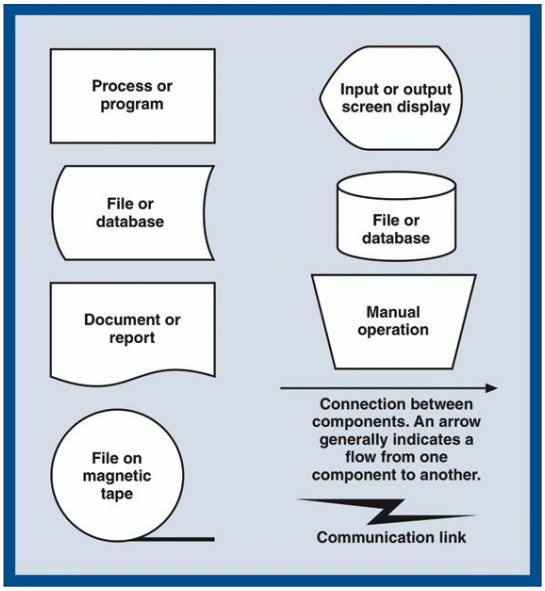
Gambar 2 Contoh *ER Diagram*

2.10 *Flowchart*

Flowchart dapat diartikan sebagai representasi dari berbagai program komputer, *file*, *database*, dan terkait pengguna proses yang membentuk sistem lengkap. *Flowchart* dibangun selama kegiatan analisis, berupa grafis yang menggambarkan organisasi subsistem

menjadi komponen-komponen otomatis dan manual. *Flowchart* dapat menunjukkan jenis sistem pemrosesan transaksi. (Satzinger et al, 2006:355)

Beberapa contoh simbol yang digunakan dalam *flow chart* adalah (Satzinger et al, 2006:368)



Gambar 3 Simbol – simbol dalam *Flowchart*

2.11 UML (*Unified Model Language*)

Unified Model Language adalah keluarga notasi grafis yang didukung oleh *meta-model* tunggal, yang membantu pendeskripsian dan desain perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek. (Fowler, 2005:1)

2.11.1 *Use Case*

Use Case adalah teknik untuk merekam persyaratan fungsional sebuah sistem. *Use Case* mendeskripsikan interaksi tipikal antara para pengguna sistem dengan sistem itu sendiri, dengan memberi sebuah narasi tentang bagaimana sistem itu ditentukan.

Dalam bahasan *use case*, para pengguna disebut sebagai aktor. Aktor merupakan sebuah peran yang dimainkan seorang pengguna dalam kaitannya dengan sistem. Aktor dapat meliputi

pelanggan, petugas layanan konsumen, manajer penjualan, dan analis produk. Seorang aktor dapat menggunakan banyak *use case*, sebaliknya sebuah *use case* juga dapat digunakan oleh beberapa aktor. Aktor tidak harus manusia, jika sebuah sistem melakukan sebuah layanan untuk sistem komputer lain, sistem lain tersebut merupakan aktor. (Fowler, 2005:142)

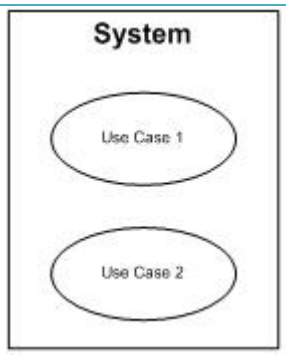
2.11.2 *Use Case Diagram*

Use Case Diagram menampilkan aktor, *use case*, dan hubungan antara mereka (Fowler, 20005:146) :

1. Aktor mana yang menggunakan *use case* yang mana.
2. *Use Case* mana yang memasukkan *use case* lain.

Berikut ini adalah tabel notasi *use case diagram* beserta penjelasannya :

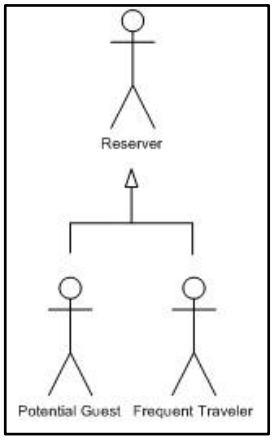
Tabel XI Notasi dalam *Use Case Diagram*

No	Komponen	Gambar	Penjelasan
1	Aktor		Aktor adalah pengguna sistem
2	*Use Case*		*Use case* adalah peran dari seorang aktor
3	Batasan Sistem		Batasan sistem digunakan untuk mengumpulkan *use case* ke dalam grup

Di dalam *use case diagram* terdapat beberapa tipe *relationship* (Fowler, 2005:147), yaitu :

1. *Generalization*

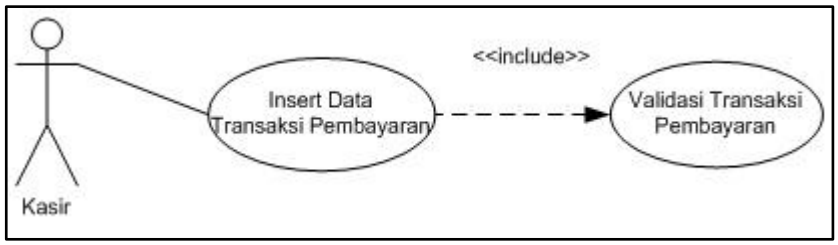
   Beberapa *use case* atau aktor yang memiliki kesamaan dapat digeneralisasi menjadi satu *use case* atau aktor.



Gambar 4 Contoh *Generalization Actor*

2. *Include*

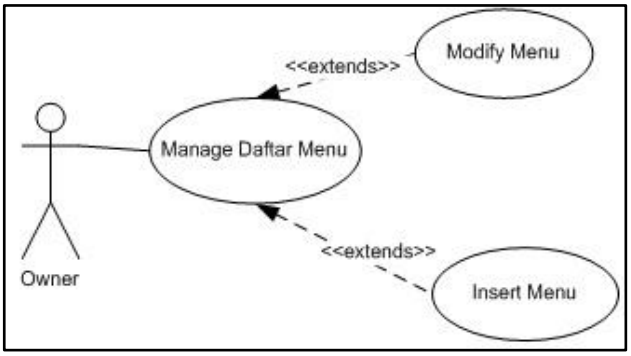
   Hubungan antar dua *Use Case* dimana yang satu memanggil yang lain.



Gambar 5 Contoh *Use Case Include*

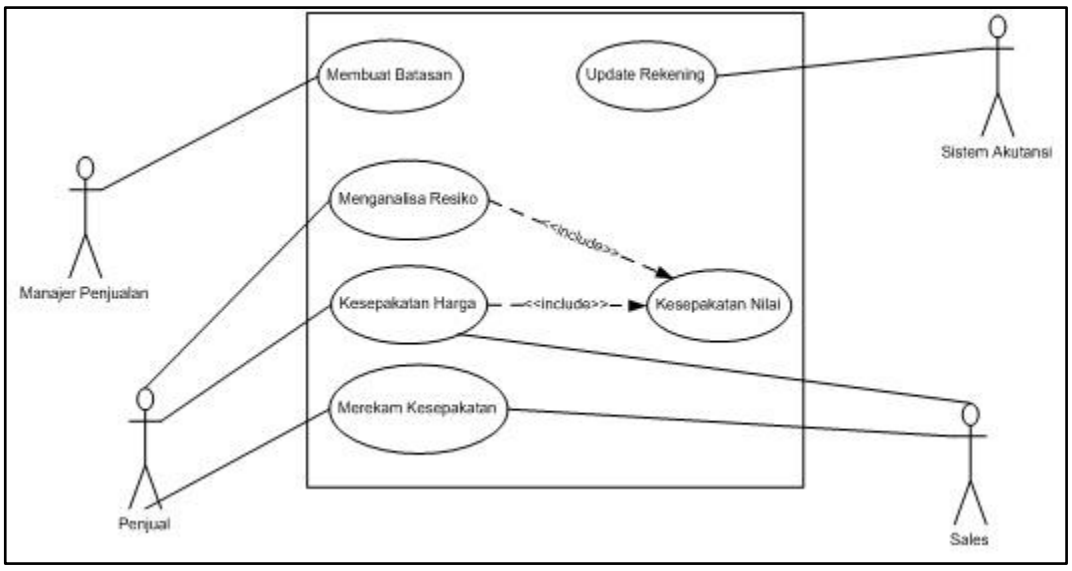
3. *Extends*

   Jika pemanggilan memerlukan adanya kondisi tertentu maka berlaku hubungan *extends*.



Gambar 6 Contoh *Use Case Extends*

Berikut ini adalah contoh dari *use case diagram* (Fowler, 2005:147)



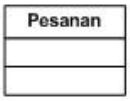
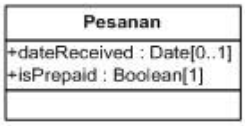
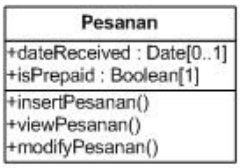
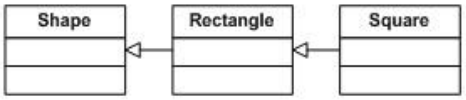
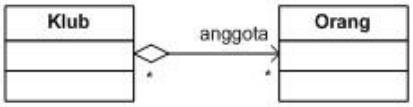
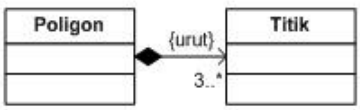
Gambar 7 Contoh *Use Case Diagram*

2.11.3 *Class Diagram*

Class Diagram mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat di antara mereka. *Class Diagram* juga menunjukkan properti dan operasi sebuah *class* dan batasan-batasan (Fowler, 2005:53)

Berikut ini adalah notasi yang digunakan dalam *class diagram* beserta penjelasannya:

Tabel XII Notasi dalam *Class Diagram*

No	Komponen	Gambar
1	*Class*	
2	*Class Attributes*	
3	*Class Operation (Method)*	
4	*Class visibility*	private (-) public (+) protected (#)
5	*Class Generalization*	
6	*Class Association*	
7	*Class Aggregation*	
8	*Class Composition*	

Komponen *class diagram* menurut Fowler(2005:54) antara lain:

1. Properti

   Properti memiliki fitur-fitur struktural dari sebuah *class*. Properti merupakan sebuah konsep tunggal, tetapi tampak seperti dua notasi yang sedikit berbeda : atribut dan asosiasi. Meskipun mereka tampak berbeda dalam sebuah *diagram*, mereka sebenarnya adalah hal yang sama. (Fowler, 2005:54)

2. Atribut

   Notasi atribut mendeskripsikan properti dengan sebaris teks di dalam kotak *class* tersebut. Bentuk atribut yang sebenarnya adalah :

   `visibility name: type multiplicity = default {property-string}`

   Contoh atribut ini adalah :

   - `name: String [1] = "Untitled" {readOnly}`

   Hanya `name` yang diperlukan.

   a. Tanda `visibility` ini menandakan atribut tersebut *public* (+) atau *private* (-).
   b. `name` atribut – bagaimana *class* tersebut mengacu pada atribut – sama dengan nama bidang dalam sebuah bahasa pemrograman.
   c. `type` atribut menunjukkan sebuah batasan tentang objek apa yang dapat diletakkan dalam atribut tersebut.
   d. `default value` nilai objek yang baru dibuat jika atribut tidak dispesifikasi selama pembuatannya.
   e. `{property-string}` memungkinkan anda untuk menunjuk properti tambahan untuk atribut tersebut. Contoh diatas digunakan `{readOnly}` untuk menunjukkan bahwa klien tidak dapat memodifikasi properti tersebut. Jika ekspresi dihilangkan, anda dapat mengasumsikan bahwa atribut tersebut dapat dimodifikasi. (Fowler, 2005:56)

3. Asosiasi

   Asosiasi merupakan garis *solid* antara dua buah *class*, ditarik dari *class* sumber ke *class target*. Nama properti bergerak sampai tujuan akhir sebuah asosiasi bernama *multiplicity*. Tujuan akhir sebuah asosiasi adalah

menghubungkan dengan *class* yang merupakan jenis properti. (Fowler, 2005:57)

4. *Multiplicity*

   Multiplicity sebuah properti merupakan indikasi tentang berapa banyak objek yang akan mengisi properti. *Multiplicity* yang akan sering ditemui adalah :

   a. 1 (Sebuah pesanan hanya bisa memiliki seorang pelanggan.)
   b. 0..1 (Seorang pelanggan perusahaan dapat memiliki sebuah *sales rep*.)
   c. * (Seorang pelanggan tidak perlu membuat sebuah pesanan dan tidak ada batas maksimal berapa jumlah pesanan yang dapat dibuat oleh seorang pelanggan – nol atau lebih pesanan.) (Martin Fowler, 2005:57)

   Dalam atribut, ada berbagai macam istilah yang mengacu pada *multiplicity* :

   a. *Optional* menunjukkan sebuah batas bawah yang bernilai 0.
   b. *Mandatory* menunjukkan sebuah batas bawah yang bernilai 1 atau mungkin lebih.
   c. *Single-valued* menunjukkan sebuah batas atas yang bernilai 1.
   d. *Multivalued* menunjukkan sebuah batas atas yang bernilai lebih dari 1 : biasanya *. (Fowler, 2005:58)

5. Generalisasi

   Generalisasi menunjukkan relasi antara kelas *parent* (kelas yang lebih abstrak) dengan kelas anak (kelas yang lebih spesifik).

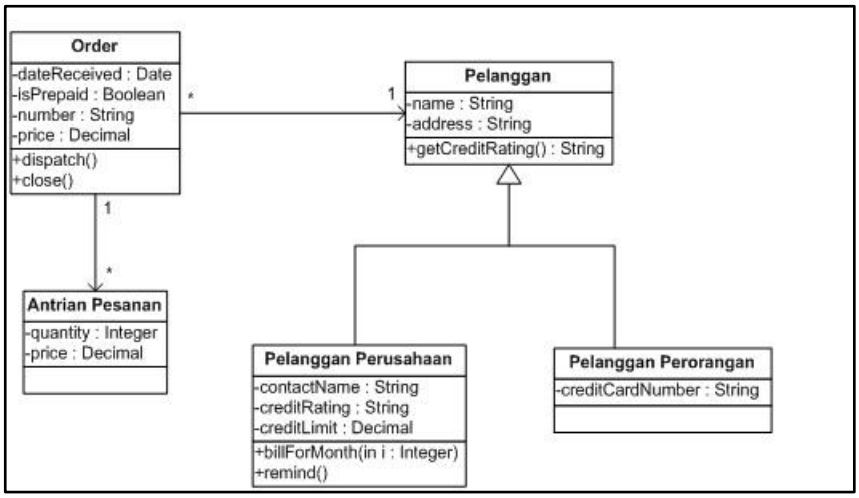
   Sebuah contoh umum tentang generalisasi melibatkan pelanggan perorangan dan perusahaan dari sebuah bisnis. Keduanya memilki perbedaan tetapi juga banyak persamaan. Persamaan-persamaan tersebut dapat dimasukkan ke dalam

class Pelanggan umum (*supertype*), dengan Pelanggan Perorangan dan Pelanggan Perusahaan sebagai *subtype*.

6. Agregasi dan Komposisi

   Agregasi merupakan bagian dari hubungan antara dua objek kelas. Dapat dikatakan seperti halnya sebuah mobil memiliki sebuah mesin dan roda-roda.

   Komposisi menunjukkan hubungan yang eksklusif antara dua objek kelas (Contohnya, sebuah fakultas tidak akan ada tanpa adanya sebuah universitas.) (Fowler, 2005:100)



Gambar 8 Sebuah *Class Diagram* Sederhana

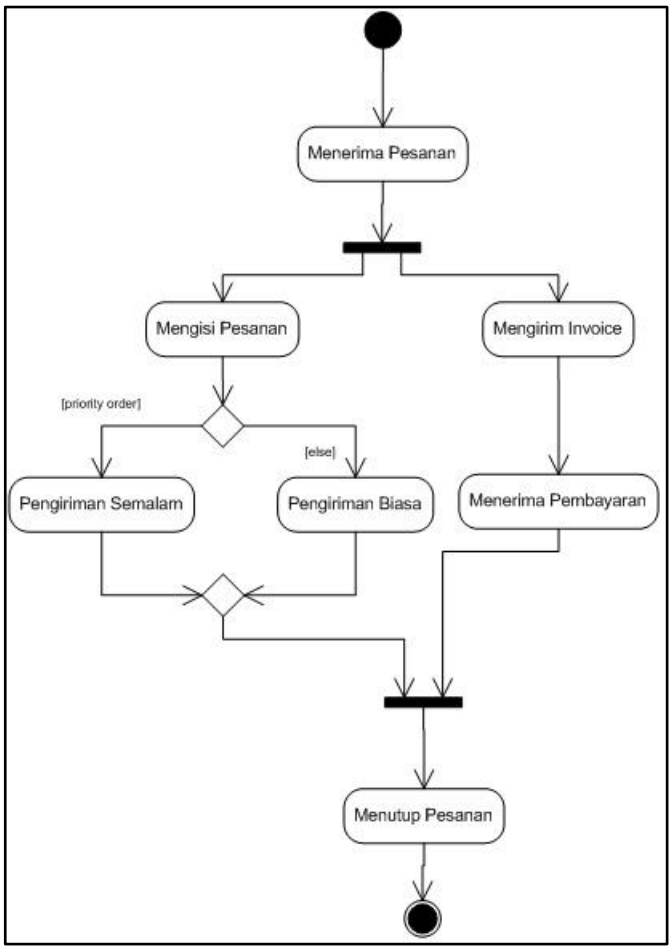
2.11.4 *Activity Diagram*

Activity Diagram adalah teknik untuk menggambarkan logika prosedural, proses bisnis, dan jalur kerja. Dalam beberapa hal, *diagram* ini memainkan peran mirip sebuah *diagram* alir, tetapi perbedaan prinsip antara diagram ini dan notasi *diagram* alir adalah *diagram* ini mendukung *behavior parallel.*

Gambar 9 ini menunjukkan sebuah contoh sederhana dari sebuah *activity diagram*. Kita mulai pada *node action* awal dan kemudian melakukan *action* Menerima Pesanan. Sekali kita melakukannya, kita mendapatkan percabangan. Sebuah percabangan memiliki satu aliran masuk dan beberapa aliran keluar.

Gambar 9 menjelaskan bahwa Mengisi Pesanan, Mengirim *Invoice*, dan *action* setelahnya yang dilakukan secara *parallel*. Pada dasarnya hal ini berarti bahwa rangkaian-rangkaian antara mereka tidak saling berkaitan.

Kita dapat mengisi pesanan, mengirim *invoice*, mengantar barang, dan kemudian menerima pembayaran, atau kita dapat mengirim *invoice*, menerima pembayaran, mengisi pesanan dan kemudian mengantar barang. (Fowler, 2005:163)



Gambar 9 Sebuah *Activity Diagram* Sederhana