BAB II

TINJAUAN PUSTAKA

II.1. Perancangan

Model perancangan sesungguhnya adalah model objek yang mendeskripsikan realisasi fisik *use case* dengan cara berfokus pada bagaimana spesifikasi-spesifikasi kebutuhan fungsional dan *non*-fungsional, bersama dengan batasan-batasan lain yang berhubungan dengan lingkungan implementasi, memiliki imbas langsung pada pertimbangan-pertimbangan pada aktivitas-aktivitas yang dilakukan pada tahap implementasi. Tambahannya, model perancangan sesungguhnya secara langsung bertindak sebagai abstraksi implementasi sistem/perangkat lunak dan dengan sendirinya model perancangan suatu saat nanti akan menjadi asupan bagi aktivitas-aktivitas selanjutnya yang kelak akan terdefinisi pada tahap implementasi (Adi Nugroho ; 2010 : 212).

II.2. Sistem

Secara leksikal, sistem berarti susunan yang teratur dari pandangan, teori, asas dan sebagainya. Dengan kata lain, sistem adalah suatu kesatuan usaha yang terdiri dari bagian-bagian yang berkaitan satu sama lain yang berusaha mencapai suatu tujuan dalam suatu lingkungan kompleks. Pengertian tersebut mencerminkan adanya beberapa bagian dan hubungan antara bagian, ini menunjukkan kompleksitas dari sistem yang meliputi kerja sama antara bagian yang interpenden satu sama lain. Selain itu dapat dilihat bahwa sistem berusaha mencapai tujuan. Pencapaian tujuan ini menyebabkan timbulnya dinamika, perubahan-perubahan yang terus menerus perlu dikembangkan dan

dikendalikan. Definisi tersebut menunjukkan bahwa sistem sebagai gugus dan elemen-elemen yang saling berinteraksi secara teratur dalam rangka mencapai tujuan atau subtujuan (Marimin ; 2008 : 1).

II.3. Sistem Informasi

Sistem informasi bukan merupakan hal yang baru, yang baru adalah komputerisasinya. Sebelum ada komputer, teknik penyaluran informasi yang memungkinkan manajer merencanakan serta mengendalikan operasi yang telah ada. Komputer menambahkan satu atau dua dimensi, seperti kecepatan, ketelitian dan penyediaan data dengan volume yang lebih besar yang memberikan bahan pertimbangan yang lebih banyak untuk mengambil keputusan.

Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi untuk dapat menyediakan laporan-laporan yang diperlukan oleh pihak luar tertentu (Tata Sutabri ; 2012 : 38).

II.4. Akuntansi

Ada banyak pihak yang memberikan pandangan secara berbeda mengenai akuntansi, yaitu akuntansi sebagai bahasa bisnis, catatan historis, realita ekonomi, komoditi dan pertanggungjawaban.

a. Akuntansi sebagai bahasa bisnis

Akuntansi sering dianggap sebagai bahasa bisnis yang fungsinya adalah untuk mengkomunikasikan informasi mengenai perusahaan kepada pihak-pihak yang berkepentingan (*stakeholders*).

b. Akuntansi sebagai catatan historis

Kalau kita berbicara mengenai akuntansi, maka sesungguhnya yang menjadi pusat perhatian dari pelaporan adalah data transaksi keuangan (bisnis) yang telah lewat. Akuntansi dianggap sebagai wahana untuk memberikan gambaran tentang sejarah organisasi dan transaksi yang telah dilakukannya dengan lingkungannya dalam masa yang telah lewat.

c. Akuntansi sebagai realita ekonomi saat ini

Akuntansi dianggap dapat memberikan gambaran mengenai keadaan atau realita ekonomi perusahaan pada saat ini. Konsekuensinya adalah bahwa aktiva dan kewajiban perusahaan harus dicatat dan dilaporkan dengan menggunakan nilai pasar wajar saat ini, bukan biaya historis.

d. Akuntansi sebagai komoditi

Komoditi adalah barang yang dibutuhkan konsumen karena daya gunanya. *Output* akuntansi yang berupa laporan keuangan, yang berisi informasi mengenai posisi keuangan dan hasil kinerja perusahaan adalah merupakan hasil dari sebuah akuntansi.

e. Akuntansi sebagai pertanggungjawaban

Laporan keuangan, sebagai produk akhir dari serangkaian akuntansi, merupakan salah satu bentuk pertanggungjawaban manajemen kepada pihak

prinsipil (investor, pemilik dana) untuk melaporkan hasil atau kinerja yang telah dilakukan sepanjang periode (Hery ; 2012 : 34).

II.5. Sistem Informasi Akuntansi

Menurut Kusrini (2007:10) dalam bukunya *Tuntunan Praktis Membangun Sistem Informasi Akuntansi Dengan Visual Basic dan Microsoft Sql Server*, Sistem informasi akuntansi merupakan sebuah sistem informasi yang mengubah data transaksi bisnis menjadi informasi keuangan yang berguna bagi pemakainya.

Tujuan dari sistem informasi akuntansi adalah :

1. Mendukung operasi sehari-hari.
2. Mendukung pengambilan keputusan manajemen.
3. Memenuhi kewajiban yang berhubungan dengan pertanggungjawaban.

Komponen-komponen yang terdapat dalam sistem informasi akuntansi adalah sebagai berikut :

1. Orang-orang mengoperasikan sistem tersebut.
2. Prosedur-prosedur, baik manual maupun yang terotomatisasi, yang dilibatkan dalam pengumpulan, pemrosesan dan penyimpanan data aktivitas-aktivitas organisasi.
3. Data tentang proses-proses bisnis.
4. *Software* yang dipakai untuk memproses data organisasi.
5. Infrastruktur teknologi informasi.

Di dalam organisasi, sistem informasi akuntansi berfungsi untuk :

1. Mengumpulkan dan menyimpan aktivitas yang dilakukan di suatu organisasi, sumber daya yang dipengaruhi oleh aktivitas-aktivitas tersebut dan para pelaku aktivitas tersebut.
2. Mengubah data menjadi informasi yang berguna bagi manajemen.
3. Menyediakan pengendalian yang memadai.

II.6. Metode Cash Basis

Ketika bagian akuntansi suatu perusahaan akan menyiapkan (menyusun) laporan keuangan, mereka menyadari bahwa periode pembukuan perusahaan yang akan dilaporkannya dapat dibagi ke dalam beberapa periode. Dengan menggunakan konsep periode akuntansi ini, atau yang dikenal dengan sebutan *accounting period concept*, akuntan harus berhati-hati dan setepat mungkin dalam menentukan besarnya jumlah pendapatan dan beban yang harus dilaporkan dalam laporan keuangan.

Apabila dasar pencatatan akuntansi yang digunakan adalah *cash basis*, maka pendapatan dan beban akan dilaporkan dalam laporan laba rugi (*income statement*) dalam periode di mana uang kas diterima (untuk pendapatan) atau uang kas dibayarkan (untuk beban). Jadi, dapat disimpulkan di sini bahwa transaksi pendapatan dan beban yang akan dilaporkan dalam laporan laba rugi adalah transaksi-transaksi yang melibatkan arus uang kas masuk (untuk pendapatan) ataupun arus uang kas keluar (untuk beban). Besarnya laba bersih (*net income*) atau rugi bersih (*net loss*) yang dihasilkan dari selisih antara pendapatan dengan

beban, akan mencerminkan jumlah bersih uang kas yang dikeluarkan (untuk *net loss*) (Hery ; 2012 : 40).

II.7. Java

Menurut Hariyanto (2011:1) Bahasa pemrograman Java merupakan karja Sun Microsystem Inc. Rilis resmi level beta dilakukan pada November 1995. Dua bulan berikutnya, Netscape menjadi perusahaan pertama yang memperoleh lisensi bahasa Java dari Sun.

Pada tahun 1996, Sun mengeluarkan JSDK (*Java Software Development Kit*) yang kemudian secara berturut-turut:

1. Versi 1.02 yang mendukung konektifitas basis data dan objek-objek tersebar.
2. Versi 1.1 pada tahun 1997 ditambahkan model kejadian (*event model*) yang handal, *internationalization* dan model komponen JavaBeans.
3. Versi 1.2 pada tahun 1998 mempunyai banyak peningkatan diantaranya User Interface Toolkit Swing yang memungkinkan pemrogram membuat aplikasi berbasis GUI yang sepenuhnya portable. Sejak ini, disebut Java 2.
4. Versi 1.3 dirilis tahun 200 dengan banyak peningkatan.
5. Versi 1.4 ditambahkan fasilitas asersi untuk dukungan design-by-contract.
6. Versi 1.5 ditambahkan fitur-fitur baru di level bahasa diantaranya generics (parameterized type), enumeration, dan metadata. versi ini disebut JDK 5.

7. Versi 1.5 ditambahkan lightweight database system yaitu Derby. Derby merupakan hasil pengembangan dari proyek basis data Apache. Derby mulanya merupakan CloudScape dari IBM.

Java telah berkembang dari semula ditunjukan untuk pemrograman *applet* di *web browser* menjadi bahasa pemrograman pengembangan aneka ragam aplikasi, mulai dari yang berjalan di *handheld devices* seperti *handphone*, PDA sampai aplikasi tersebar skala *enterprise* di beragam komputer *server*. Java merupakan bahasa berorientasi objek untuk pengembangan aplikasi mandiri, aplikasi berbasis internet, aplikasi untuk perangkat cerdas yang dapat berkomunikasi lewat jaringan. Melalui teknologi Java dimungkinkan perangkat audio stereo di rumah terhubung dengan jaringan komputer. Java tidak lagi hanya bahasa untuk membuat *applet* yang memperindah halaman *web* tapi Java telah menjadi bahasa untuk pengembangan aplikasi skala *enterprise* berbasis jaringan besar.

II.8. NetBeans

NetBeans merupakan salah satu proyek *open source* yang disponsori oleh *Sun Microsystem*. Proyek ini bediri pada tahun 2000 dan telah menghasilkan 2 produk, yaitu NetBeanss IDE dan NetBeans Platform. NetBeans IDE merupakan produk yang digunakan untuk melakukan pemrograman baik menulis kode, meng-*compile*, mencari kesalahan dan mendistribusikan program. Sedangkan NetBeans Platform adalah sebuah modul yang merupakan kerangka awal / pondasi dalam bangun aplikasi desktop yang besar.

NetBeans juga menyediakan paket yang lengkap dalam pemrograman dari pemrograman standar (aplikasi *desktop*), pemrograman enterprise, dan pemrograman perangkat mobile. Saat ini NetBeans telah mencapai versi 6.8 (Wahana Komputer ; 2010 : 15).

II.9. Database

Database adalah sekumpulan tabel-tabel yang saling berelasi, relasi tersebut bisa ditunjukkan dengan kunci dari tiap tabel yang ada. Satu *database* menunjukkan satu kumpulan data yang dipakai dalam satu lingkup perusahaan atau instansi.

Database mempunyai kegunaan dalam mengatasi penyusunan dan penyimpanan data, maka seringkali masalah yang dihadapi adalah:

a. Redundansi dan Inkonsistensi data
b. Kesulitan dalam pengaksesan data
c. Isolasi data untuk standarisasi
d. Multi *user*
e. Keamanan data
f. Integritas data
g. Kebebasan data (Asrianda ; 2008 : 1)

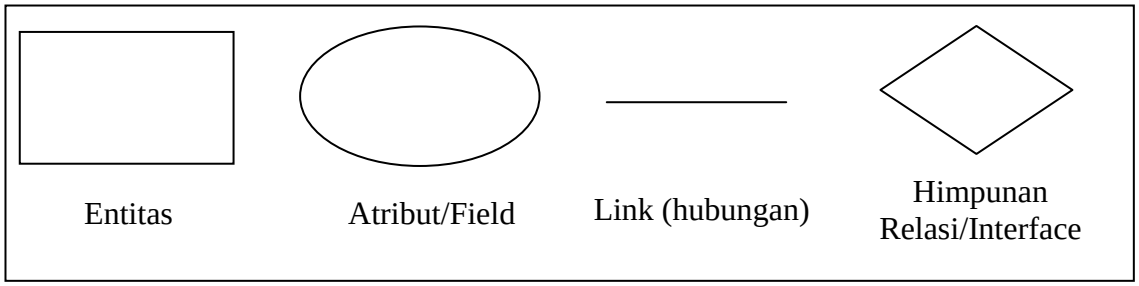
II.10. MySQL

MySQL adalah suatu sistem manajemen basis data relasional (RDBMS-*Relational Database Management System*) yang mampu bekerja dengan cepat, kokoh, dan mudah digunakan. Contoh RDBMS lain adalah *Oracle, Sybase*. Basis data memungkinkan anda untuk menyimpan, menelusuri, menurutkan dan mengambil data secara efesien. *Server* MySQL yang akan membantu melakukan fungsionalitas tersebut. Bahasa yang digunakan oleh MySQL tentu saja adalah SQL-*standar* bahasa basis data relasional di seluruh dunia saat ini.

MySQL dikembangkan, dipasarkan dan disokong oleh sebuah perusahaan Swedia bernama MySQL AB. RDBMS ini berada di bawah bendera GNU GPL sehingga termasuk produk *Open Source* dan sekaligus memiliki lisensi komersial. Apabila menggunakan MySQL sebagai basis data dalam suatu situs Web. Anda tidak perlu membayar, akan tetapi jika ingin membuat produk RDBMS baru dengan basis MySQL dan kemudian mengenalnya, anda wajib bertemu mudah dengan lisensi komersial (Antonius Nugraha Widhi Pratama ; 2010 : 10).

II.11. *Entity Relationship Diagram* (ERD)

Entity Relationship Diagram atau ERD merupakan salah satu alat (*tool*) berbentuk grafis yang populer untuk *desain database. Tool* ini relatif lebih mudah dibandingkan dengan Normalisasi. Kebanyakan sistem analis memakai alat ini, tetapi yang jadi masalah, kalau kita cermati secara seksama, *tool* ini mencapai 2NF (Ir. Yuniar Supardi ; 2010 : 448).



Gambar. II.1 Bentuk Simbol ERD
(Sumber : Ir. Yuniar Supardi ; 2010 : 448)

II.12. Kamus Data

Perancangan *database* digunakan dengan data *dictionary* (kamus data) yang merupakan daftar semua elemen/field. kamus data diperoleh pada saat analisis dengan diagram arus data. Pada perancangan ini dibuat dengan sekaligus contoh yaitu pada permasalahan komputerisasi karyawan yang mengerjakan proyek-proyek pada suatu perusahaan (Harianto Kristanto ; 2006 : 38).

Kamus data (data *dictionary*) mencakup definisi-definisi dari data yang disimpan di dalam basis data dan dikendalikan oleh sistem manajemen basis data. Figur 6.5 menunjukkan hanya satu tabel dalam basis data jadwal. Struktur basis data yang dimuat dalam kamus data adalah kumpulan dari seluruh definisi *field*, definisi tabel, relasi tabel, dan hal-hal lainnya. Nama *field* data, jenis data (seperti

teks atau angka atau tanggal), nilai-nilai yang valid untuk data, dan karakteristik-karakteristik lainnya akan disimpan dalam kamus data. Perubahan-perubahan pada struktur data hanya dilakukan satu kali di dalam kamus data, program-program palikasi yang mempergunakan data tidak akan ikut terpengaruh (Raymond McLeod ; 2008 : 171).

II.13. Teknik Normalisasi

Normalisasi adalah teknik perancangan yang banyak digunakan sebagai pemandu dalam merancang basis data relasional. Pada dasarnya, normalisasi adalah proses dua langkah yang meletakkan data dalam bentuk tabulasi dengan menghilangkan kelompok berulang lalu menghilangkan data yang terduplikasi dari tabel rasional.

Teori normalisasi didasarkan pada konsep bentuk normal. Sebuah tabel relasional dikatakan berada pada bentuk normal tertentu jika tabel memenuhi himpunan batasan tertentu. Ada lima bentuk normal yang telah ditemukan.

II.13.1. Bentuk-bentuk Normalisasi

a. Bentuk normal tahap pertama (1" Normal Form)

Contoh yang kita gunakan di sini adalah sebuah perusahaan yang mendapatkan barang dari sejumlah pemasok. Masing-masing pemasok berada pada satu kota. Sebuah kota dapat mempunyai lebih dari satu pemasok dan masing-masing kota mempunyai kode status tersendiri.

b. Bentuk normal tahap kedua (2^{nd} normal form)

Definisi bentuk normal kedua menyatakan bahwa tabel dengan kunci utama gabungan hanya dapat berada pada 1NF, tetapi tidak pada 2NF. Sebuah tabel relasional berada pada bentuk normal kedua jika dia berada pada bentuk normal kedua jika dia berada pada 1NF dan setiap kolom bukan kunci yang sepenuhnya tergantung pada seluruh kolom yang membentuk kunci utama.

c. Bentuk normal tahap ketiga (3rd normal form)

Bentuk normal ketiga mengharuskan semua kolom pada tabel relasional tergantung hanya pada kunci utama. Secara definisi, sebuah tabel berada pada bentuk normal ketiga (3NF) jika tabel sudah berada pada 2NF dan setiap kolom yang bukan kunci tidak tergantung secara transitif pada kunci utamanya.

d. Boyce Code Normal Form (BCNF)

Setelah 3NF, semua masalah normalisasi hanya melibatkan tabel yang mempunyai tiga kolom atau lebih dan semua kolom adalah kunci. Banyak praktisi berpendapat bahwa menempatkan entitas pada 3NF sudah cukup karena sangat jarang entitas yang berada pada 3NF bukan merupakan 4NF dan 5NF.

e. Bentuk Normal Tahap Keempat dan Kelima

Sebuah tabel relasional berada pada bentuk normal keempat (4NF) jika dia dalam BCNF dan semua ketergantungan multivalue merupakan ketergantungan fungsional. Bentuk normal keempat (4NF) didasarkan pada konsep ketergantungan multivalue (MVD).

Sebuah tabel berada pada bentuk normal kelima (5NF) jika ia tidak dapat mempunyai dekomposisi lossless menjadi sejumlah tabel lebih kecil. Empat bentuk normal pertama berdasarkan pada konsep ketergantungan fungsional, sedangkan bentuk normal kelima berdasarkan pada konsep ketergantungan gabungan (*join dependence*) (Janner Simarmata ; 2010 : 76).

II.14. UML (*Unified Modeling Language*)

Menurut Windu Gata (2013 : 4) Hasil pemodelan pada OOAD terdokumentasikan dalam bentuk *Unified Modeling Language* (UML). UML adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak.

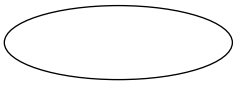
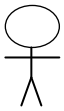
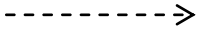
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

1. *Use case* Diagram

Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram, yaitu :

Tabel II.1. Simbol *Use Case*

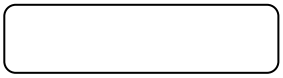
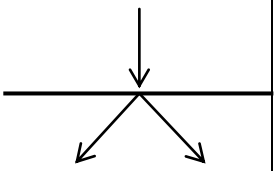
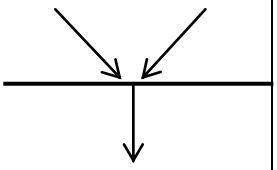
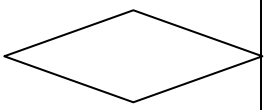
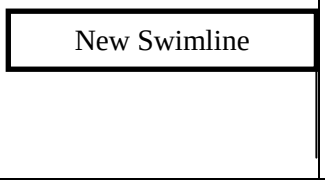
Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki control terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengindikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	*Include*, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	*Extend*, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber : Windu Gata & Grace Gata ; 2013 : 4-6)

2. Diagram Aktivitas (*Activity Diagram*)

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram*, yaitu :

Tabel II.2. Simbol *Activity Diagram*

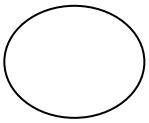
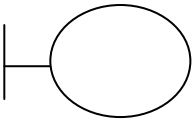
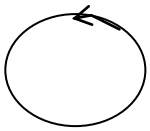
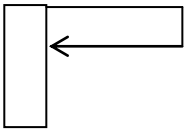
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true*, *false*.
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

(Sumber : Windu Gata & Grace Gata ; 2013 : 6-7)

3. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram*, yaitu :

Tabel II.3. Simbol *Sequence Diagram*

Gambar	Keterangan
	*Entity Class,* merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class,* berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control class,* suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message,* simbol mengirim pesan antar *class*.
	*Recursive,* menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation, activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Windu Gata & Grace Gata ; 2013 : 7-8)

4. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem.

Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti.

Tabel II.4. *Multiplicity Class Diagram*

Multiplicity	**Penjelasan**
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Windu Gata & Grace Gata ; 2013 : 9)