BAB II

TINJAUAN PUSTAKA

II.1. Sistem Pendukung Keputusan

Sistem Pendukung Keputusan merupakan sistem berbasis komputer yang membantu para pengambil keputusan mengatasi masalah melalui interaksi Pengambilan Keputusan *(Decision Making)* adalah melakukan penilaian dan menjatuhkan pilihan. Keputusan ini diambil setelah melalui beberapa perhitungan dan perhitungan alternatif, proses memilih suatu alternatif cara bertindak dengan metode yang efisien sesuai situasi. Proses itu untuk menemukan dan menyelesaikan masalah organisasi. Pernyataan ini menegaskan bahwa mengambil keputusan memerlukan satu seri tindakan, membutukan beberapa langkah. Dapat saja langkah-langkah itu terdapat dalam pikiran seseorang yang sekaligus mengajaknya berpikir sistematis. Suatu aturan kunci dalam pengambilan keputusan ialah "sekali kerangka yang tepat sudah diselesaikan, keputusan harus dibuat". Dan sekali keputusan dibuat sesuatu mulai terjadi. Dengan kata lain, keputusan mempercepat diambilnya tindakan, memdorong lahirnya gerakan dan perubahan (Muhammad Yudin Ritonga ; 2014 : 19)

SPK atau *Decission Support Sistem* (DSS) biasanya dibangun untuk mendukung solusi atas suatu masalah atau untuk suatu peluang. Aplikasi sistem pendukung keputusan (SPK) menggunakan CBIS *(Computer Based Information System)* yang fleksibel, interaktif, dan dapat di adaptasi, yang dikembangkan untuk mendukung solusi atas masalah manajemen spesifik yang tidak terstruktur.

Sistem pendukung keputusan sebagai sistem berbasis komputer yang terdiri dari tiga komponen yang saling berinteraksi, sistem bahasa (mekanisme untuk memberikan komunikasi antara dan komponen sistem pendukung keputusan lain), sistem pengetahuan (respositori pengetahuan domain masalah yang ada pada sistem pendukung keputusan atau sebagai data atau sebagai prosedur) dan sistem pemrosesan atau lebih kapabilitas manipulasi masalah umum yang diperlukan untuk pengambilan keputusan. (Dicky Nofriansyah ; 2014 : 1)

II.1.1 Karakteristik Sistem Pendukung Keputusan

Karakteristik dan kapabilitas sistem pendukung keputusan antara lain :

1. Dukungan untuk pengambilan keputusan, terutama pada situasi semi terstruktur dan tak terstruktur, dengan menyertakan penilaian manusia dan informasi terkomputerisasi. Masalah-masalah tersebut tidak dapat dipecahkan oleh sistem komputer lain atau alat kuantitatif standar.
2. Dukungan di semua fase proses pengambilan keputusan : intelegensi, desain, pilihan dan implementasi.
3. Adaptivitas sepanjang waktu. Pengambil keputusan seharusnya reaktif, dapat menghadapi perubahan kondisi secara tepat, dan dapat mengadaptasi DSS untuk memenuhi perubahan tersebut. DSS bersifat fleksibel dan karena itu pengguna dapat menambahkan, menghapus, menggabungkan, mengubah, atau menyusun kembali elemen-elemen dasar, DSS juga fleksibel dalam hal dapat dimodifikasi untuk memecahkan masalah lain yang sejenis.
4. Adanya *interface* manusia atau mesin dimana manusia *(user)* tetap memegang kontrol proses pengambilan keputusan.

5. Peningkatan terhadap keefektifan pengambilan keputusan (akurasi, *timeliness*, kualitas).
6. Kontrol penuh oleh pengambil keeputusan terhadap semua langkah proses pengambilan keputusan dalam memecahkan suatu masalah. DSS secara khusus menekankan untuk mendukung pengambil keputusan, bukannya menggantikan (Sri Hartati ; 2011 : 37).

II.1.2 Keuntungan Sistem Pendukung Keputusan

Dengan berbagai karakter khusus yang dimiliki Sistem Pendukung Keputusan, maka SPK (Sistem Pendukung Keputusan) dapat memberikan berbagai manfaat dan keuntungan. Manfaat yang dapat diambil dari SPK adalah :

1. SPK memperluas kemampuan pengambilan keputusan dalam memproses data atau informasi bagi pemakainya.
2. SPK membantu pengambil untuk memecahkan masalah terutama berbagai masalah yang sangat kompleks dan tidak terstruktur.
3. SPK dapat menghasilkan solusi denga lebih cepat serta hasilnya dapat diandalkan.
4. Walaupun suatu SPK, mungkin saja tidak mampu memecahkan masalah yang dihadapi oleh pengambil keputusan, namun ia dapat menjadi stimulan bagi pengambil keputusan dalam memahami persoalannya, karena mampu menyajikan berbagai alternatif pemecahan.

II.2. *MultiFactor Evaluation Process* (MFEP)

Menurut jurnal ilmiah SAINTIKOM vol 13 no 2 mei 2014 dalam metode MFEP ini pengambilan keputusan dilakukan dengan memberikan pertimbangan

subyektif dan intuitif terhadap Faktor yang dianggap penting. Pertimbangan-pertimbangan tersebut berupa pemberian bobot (*weighting system*)1 atas multifactor yang terlibat dan dianggap penting tersebut. Langkah dalam metode MFEP ini yang pertama adalah menentukan faktor-faktor yang dianggap penting, yang selanjutnya membandingkan faktor-faktor tersebut sehingga diperoleh urutan faktor berdasarkan kepentingannya dari yang terpenting, kedua terpenting dan seterusnya. Proses pemilihan alternative terbaik menggunakan "*weightingsystem*", dimana metode tersebut merupakan metode *kuantitatif*, disebut sebagai metode "*MultiFactor Evaluation Process* "(MFEP). Dalam MFEP pertama-tama seluruh kriteria yang menjadi faktor penting dalam melakukan pertimbangan diberikan pembobotan (*weighting*) yang sesuai. Langkah yang sama juga dilakukan terhadap alternatif-alternatif yang akan dipilih, yang kemudian dapat dievaluasi berkaitan dengan faktor – faktor pertimbangan tersebut.

II.2.2 Konsep Dasar Penggunaan Metode MFEP

Dibawah ini merupakan langkah-langkah proses perhitungan menggunakan metode MFEP, yaitu :

1. Menentukan faktor dan bobot faktor dimana total pembobotan harus sama dengan 1 (∑ pembobotan = 1), yaitu *factor weight*.
2. Mengisikan nilai untuk setiap faktor yang mempengaruhi dalam pengambilan keputusan dari data-data yang akan diproses, nilai yang dimasukkan dalam proses pengambilan keputusan merupakan nilai objektif, yaitu sudah pasti *factor evaluation* yang nilainya antara 0-1.

3. Proses perhitungan *weigth evaluation* yang merupakan proses perhitungan bobot antara *factor weigth* dan *factor evaluation* dengan serta penjumlahan seluruh hasil *weigth evaluations* untuk memperoleh total hasil evaluasi.

Penggunaan model MFEP dapat direalisasikan dengan contoh berikut :

WE = FW x E

$\sum$WE = $\sum$(FW x E)

Keterangan :

WE = *Weighted Evaluation*

FW = *Factor Weight*

E = *Evaluation*

$\sum$WE = *Total Weighted Evaluation*

Maka perhitungan perkalian anatar lain bobot *weight* dengan nilai bobot *evaluation* sesuai dengan evaluasi pihak kampus pada setiap mahasiswa. Steve Marcel, seorang lulusan sarjana bidang bisnis mencari beberapa lowongan pekerjaan. Setelah mendiskusikan gambaran pekerjaan yang akan dikerjakannya dengan penasehat didiknya dan departement direktur pusat penemuan pegawai, steve mendapatkan bahwa dari ketiga faktor yang terpenting baginya yaitu gaji, peluang karis yang lebih baik, dan lokasi tempat kerja. Steve sudah memutuskan bahwa peluang jenjang karir merupakan faktor yang terpenting baginya. Faktor tersebut diberinya nilai skala 0.6. Steve menempatkan gaji diurutan berikutnya dengan nilai skala 0.3. Terakhir, steve memberikan nilai skala jika di jumlahkan harus sama denga satu seperti tabel II.1. (Jurnal Pelita Informatika Budi Darma volume VI nomor 3 : 2004)

Tabel II.1. Nilai Bobot Untuk Faktor

Faktor	Importance (Weight)
Salary	0.6
Career Advancement	0.3
Location	0.1

(Sumber : Jurnal Sistem Informasi Volume V nomor 2 : 2014)

Pada saat itu steve merasa yakin bahwa ia di terima di perusahaan AA, untuk perusahaan EDS,Ltd dan perusahaan PW,Inc. Untuk setiap perusahaan, steve menghitung rata-rata variasi faktor dari nilai skala 0 sampai 1. Untuk perusahaan AA, setiap memberikan faktor gaji dengan nilai skala 0.4. Peluang jenjang karir dengan nilai skala 0.9 dan lokasi tempat kerja dengan nilai skala 0.6. Untuk perusahaan EDS,Ltd, steve membrikan faktor gaji dengan nilai skala 0.8, peluang jenjang karir dengan nilai skala 0.7 dan lokasi tempat kerja dengan nilai skala 0.8. Untuk perusahaan PW.Inc, steve memberikan nilai faktor gaji dengan nilai skala 0.9, peluang jenjang karir dengan nilai skala 0.6 dan lokasi tempat kerja dengan nilai skala 0.9. Hasilnya dapat dilihat pada tabel II.2.

Tabel II.2. Tabel Nilai Faktor dari Setiap Data Uji

Faktor	AA.CO	EDS.LTD	PW.INC
Salary	0.7	0.8	0.9
Career advancement	0.9	0.7	0.5
Location	0.6	0.8	0.9

(Sumber : Jurnal Sistem Informasi Volume V nomor 2 : 2014)

Dari informasi yang diperoleh, steve dapat menghitung total bobot evaluasi dari setiap kriteria pekerjaan. Setiap pekerjaan menghasilkan nilai evaluasi dari tiga faktor dan bobot faktor dikalikan dengan nilai evaluasi dan dijumlahkan untuk memperoleh total hasil evaluasi.

Tabel II.3. Tabel Nilai Evaluasi Perusahaan AA

Factor Namae	Factor Weight		Factor Evaluation		Weight ed evaluation
Salary	0.3	X	0.7	=	0.21
Career advancement	0.6	X	0.9	=	0.54
Location	0.1	X	0.6	=	0.06
Total	1				0.81

(Sumber : Jurnal Sistem Informasi Volume V nomor 2 : 2014)

Tabel II.4. Tabel Nilai Evaluasi Perusahaan EDS.Ltd

Factor Namae	Factor Weight		Factor Evaluation		Weight ed evaluation
Salary	0.3	X	0.8	=	0.24
Career advancement	0.6	X	0.7	=	0.42
Location	0.1	X	0.8	=	0.08
Total	1				0.74

(Sumber : Jurnal Sistem Informasi Volume V nomor 2 : 2014)

Tabel II.5. Tabel Nilai Evaluasi Perusahaan PW.Inc

Factor Namae	Factor Weight		Factor Evaluation		Weight ed evaluation
Salary	0.3	X	0.9	=	0.27
Career advancement	0.6	X	0.6	=	0.36
Location	0.1	X	0.9	=	0.09
Total	1				0.72

(Sumber : Jurnal Sistem Informasi Volume V nomor 2 : 2014)

Dari setiap perusahaan, seperti yang dapat dilihat pada tabel II.3, perusahaan AA memperoleh total bobot evaluasi 0.81. Analisis yang sama dilakukan juga untuk perusahaan EDS,Ltd dan perusahaan PW.Inc pada tabel II.4 dan tabel II.5. Sesuai dengan yang dapat dilihat dari hasil analisis, perusahaan AA memperoleh bobot faktor yang tinggi, setelahnya adalah perusahaan EDS.Ltd yang memperoleh total bobot evaluasi 0.74. Dengan menggunakan *Multifactor Evaluation Process*, steve mengambil keputusan untuk bekerja di perusahaan AA karena perusahaan tersebut memiliki nilai bobot faktor tertinggi dari yang lainnya**.**

II.3. Sekilas Sejarah *Java*

Bahasa pemrograman *java* dimulai dari sebuah tim pengembangan software dari Sun Microsystem yang dipimpin oleh james Gosling dan Patrick Naughton. Pada tahun 1991, Sun Microsystem mengembangkan sebuah bahasa pemrograman yang berukuran kecil untuk diimplementasikan pada alat elektronik rumah tangga seperti switchbox TV kabel. Berhubungan alat tersebut tidak memiliki banyak memori, maka bahasa yang digunakan harus sangat kecil dan menghasilkan kode yang kecil pula. Permasalahan lainnya adalah alat-alat tersebut memiliki CPU yang berbeda-beda karena dibuat oleh manufactur yang

berbeda. Jadi sangatlah diharuskan bahasa pemrograman tersebut tidak terikat pada sebuah arsitektur mesin tertentu saja. (Sri Sulistiani ; 2011 : 1)

Oleh karena itu adanya keharusan sebuah pemrograman yang kecil, menghasilkan kode yang kecil pula dan harus platform independen (tidak terikat pada platform) membuat tim pada proyek tersebut terinspirasi oleh ide pemrograman yang sama yang telah ditemukan oleh Niklaus Wirth, Penemu pascal. Jadi penemu pascal memiliki pemikiran tentang sebuah software bahasa pemrograman portable dan tidak tergantung pada sebuah platform atau mesin. Bahasa pemrogaman komersial yang disebut UCSD pascal tersebut menghasilkan kode intermediate yang diperuntukkan bagi sebuah mesin virtual. Jadi, kode asli dari bahasa pemrograman tersebut tidak tergantung pada mesin ataupun platform sistem operasi karena USCD pascal menghasilkan intermediate code yang selanjutnya akan dikompilasi atau diterjemahkan oleh mesin virtual ke kode mesin dimana kode tersebut dijalankan. (Sri Sulistiani ; 2011 : 2)

Booming bahasa java dimulai pada tahun 1995 ketika Netscape memutuskan untuk menggunakan java pada *web browser*nya, yaitu Netscape Navigator apada tahun 1996. Hal ini kemudian diikuti oleh raksasa – raksasa software seperti IBM, Symantec, Inprise dan masih banyak yang lain termasuk Microsoft dengan internet explorernya. Sun sendiri merilis java pertama kalinya pada tahun 1996, kemudian diikuti dengan versi 1.02 beberapa bulan kemudian. Pada awalnya java masi mampu memenuhi kebutuhan para pengembang untuk membangun *software* secara profesional. Baru pada tahun 1998 muncul versi java 1.2 yang dirilis pada bulan Desember dan beberapa hari kemudian namanya diganti dengan java 2. (Sri Sulistiani ; 2011 : 3)

II.3.1. *Java Language Spesification*, API, JDK dan IDE

Java language spesification adalah teknis dari bahasa pemrograman java yang didalamnya terdapat aturan penulisan sintaks dan semantik java. Referensi lengkap tentang java language *spesification* dapat dilihat pada website resmi java, yaitu http://java.sun.com/docs/books/jl.

Api adalah *Aplication Programming Interface* yaitu sebuah layer yang berisi *class-class* yang sudah didefinisikan dan antarmuka pemrograman yang akan membantu para pengembangan aplikasi dalam perancangan sebuah aplikasi. Pada saat ini dikenal ada tiga macam API dari java, yaitu:

1. J2SE, yaitu Java 2 Standard Edition adalah sebuah API yang dapat digunakan untuk mengembangkan aplikasi-aplikasi yang bersifat *client – side standalone* atau *applet*.
2. J2EE, yaitu Java 2 Enterprise Edition adalah API yang digunakan untuk melakukan pengembangan aplikasi-aplikasi yang bersifat server-side seperti Java Servlet, dan Java Server Pages.
3. J2ME, yaitu Java 2 Micro Edition adalah API yang merupakan subset dari J2SE tetapi memiliki kegunaan untuk mengembangkan aplikasi pada *handheld device* seperti Smart Phone atau PDA tentu saja yang didalamnya telah ditanamkaninterpreter java.

IDE (*Integreted Development Environment*), yaitu sebuah lingkungan pengembangan aplikasi lengkap dan dapt membantu proses pengembangan sebuah aplikasi menjadi lebih cepat. Berikut ini situs *web download* java dan netbeans.

II.4. MySQL

MySQL adalah sebuah perangkat lunak system manajemen basis data SQL *(bahasa inggris : database management system)* atau DBMS yang *multithread*, multiuse, denga sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi komersial untuk kasus-kasus dimana penggunaanya tidak cocok dengan penggunaan GPL.

Tidak sama dengan proyek-proyek sepert Apache, dimana perangkat lunak di kembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh penulisnya masing-masing, MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia MySQL AB, dimana memegang hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan MySQL AB adalah : David Axmark, Alan Larsson, dan Michael "Monty" Widenius. (Alan Nur Aditya ; 2011 : 61)

II.4.1 Normalisasi

Normalisasi merupakan sebuah teknik dalam logical desain sebuah basis data yang mengelompokkan atribut dari suatu relasi sehingga membentuk struktur relasi yang baik (tanpa redudansi). Normalisasi adalah proses pembentukan struktur basis data sehingga sebagian besar ambiguity bisa dihilangkan.

Tujuan normalisasi yaitu untuk menghilangkan kerangkapan data, untuk mengurangi kompleksitas, serta untuk mempermudah pemodifikasian data, proses normalisasi data diuraikan dalam bentuk tabel, selanjutnya dianalisis berdasarkan persyaratan tertentu ke beberapa tingkat, apabila tabel tersebut perlu dipecah

menjadi beberapa tabel yang lebih sederhana sampai memenuhi bentuk yang optimal.

Adapun Tahapan Normalisasi yaitu :

1. Bentuk Normal Tahap Pertama (1NF)

   Bentuk normal 1NF terpenuhi jika sebuah tabel tidak memiliki atribut bernilai banyak (multivalued attribute), atribut composite atau kombinasinya dalam domain data yang sama. Setiap atribut dalam tabel tersebut harus bernilai atomic (tidak dapat dibagi-bagi lagi).

2. Bentuk Normal Tahap Kedua (2NF)

   Bentuk Normal 2NF terpenuhi dalam sebuah tabel jika telah memenuhi bentuk 1NF, dan semua atribut selain primary key, secara utuh memiliki Functional dependency pada primary key. Sebuah tabel tidak memenuhi 2NF, jika ada atribut yang ketergantungannya (Functional Dependency) hanya bersifat parsial saja (hanya tergantung pada sebagian dari primary key). Jika terdapat atribut yang tidak memiliki ketergantungan terhadap primary key, maka atribut tersebut harus dipindah atau dihilangkan.

3. Bentuk Normal Tahap Ketiga (3NF)

   Bentuk normal 3NF terpenuhi jika telah memenuhi bentuk 2NF, dan jika tidak ada atribut non primary key yang memiliki ketergantungan terhadap atribut non primary key yang lainnya, Untuk setiap Functional Dependency dengan notasi X-->A, maka : X harus menjadi superkey pada tabel tersebut. Atau A merupakan bagian dari primary key pada tabel tersebut.

4. Boyce-Code Normal Form (BCNF)

Bentuk BCNF terpenuhi dalam sebuah tabel, jika untuk setiap Functional Dependency terhadap setiap atribut atau gabungan atribut dalam bentuk : X --> Y maka X adalah Super Key. Tabel tersebut harus di dekomposisi berdasarkan Functional Dependency yang ada, sehingga X menjadi super key dari tabel-tabel hasil dekomposisi. Setiap tabel dalam BCNF merupakan 3NF. Akan tetapi setiap 3NF belum tentu termasuk BCNF. Perbedaannya, untuk Functional Dependency X--> A, BCNF tidak membolehkan A sebagai bagian dari primary key.

5. Bentuk Normal Tahap Keempat (4NF) atau MVD dan PJNF

Bentuk normal 4NF terpenuhi dalam sebuah tabel jika telah memenuhi bentuk BCNF, dan tabel tersebut tidak boleh memiliki lebih dari sebuah multivalued attribute. Untuk setiap multivalued attribute (MVD) juga harus merupakan Functional Dependency.

6. Bentuk Normal Tahap Kelima (5NF)

Bentuk normal 5NF terpenuhi jika memiliki sebuah loseloss decomposition menjadi tabel-tabel yang lebih kecil. Jika 4 bentuk normal sebelumnya dibentuk berdasarkan Functional Dependency, 5NF dibentuk berdasarkan konsep Join Dependence. Yakni apabila sebuah tabel telah di dekomposisi menjadi tabel-tabel lebih kecil, harus bisa digabungkan lagi untuk membentuk tabel semula.

7. Overnormalisasi

Analisa Overnormalisasi diperlukan jika : Database ini digunakan untuk sistem multi user. Tabel-tabel yang sudah normal ini digabungkan dengan fungsi lain yang ada di lapangan, misalnya; untuk fungsi retur, untuk fungsi inventori, untuk fungsi sales order maupun order pembelian, untuk fungsi keamanan database, dan lain-lain.

II.5. UML (*Unifed Modeling Language)*

Menurut Windu Gata (2013 : 4) Hasil pemodelan pada OOAD terdokumentasi dalam bentuk *UML (Unified Modeling Language)* adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasi 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 dipergunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagi berikut :

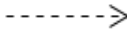
1. *Use case* Diagram

Use case diagram merupakan pemodelan untuk kelakuan (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.

Simbo-simbol yang digunakan dalam use case diagram, yaitu :

Tabel II.6. Simbol Use Case

GAMBAR	KETERANGAN
	Use case menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan 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 mengidikasikan aliran data.
	Asosiasi antara aktor dan use case yang menggunakan panah terbuka untuk mengidinkasikan bila actor berinteraksi secara pasif dengan sistem

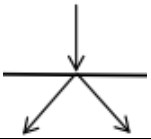
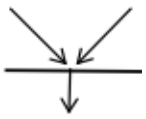
	Include, merupakan di dalam use case lain (required) atau pemanggilan use case oleh use case lain, contohnya adalah pemanggilan sebuah fungsi program.
	Extend, merupakan perluasan dari use case lain jika kondisi atau syarat terpenuhi.

(Sumber : Windu Gata ; 2013 : 4)

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.6. 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 pararel 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 ; 2013 : 6)

3. *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.

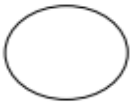
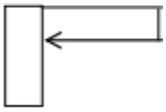
Tabel II.7. Tabel *Multiplicity Class* Diagram

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

4. 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.8. Tabel *Sequence* Diagram

GAMBAR	KETERANGAN
	EntityClass, 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 ; 2013 : 7)