BAB II

LANDASAN TEORI

2.1 APLIKASI

Aplikasi merupakan suatu subteks perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan satu tugas yang diinginkan pengguna (wikipedia.org).

Terdapat beberapa teori yang mendefinisikan aplikasi seperti yang di kemukanan oleh beberapa ahli, di antaranya adalah:

Menurut Pranama (2012) aplikasi adalah satu unit perangkat lunak yang dibuat untuk melayani kebutuhan akan beberapa aktivitas seperti system perniagaan, game, pelayanan masyarakat, periklanan, atau semua proses yang hampir dilakukan manisia.

Menurut Vermandkk (2009) aplikasi adalah perangkat intruksi khusus dalam komputer yang di rancang agar kita menyelesaikan tugas-tugas tertentu.

Menurut Yuhefizar (2012) aplikasi merupakan program yang dikembangkan untuk memenuhi kebutuhan pengguna dalam menjalankan pekerjaan tertentu.

Jadi aplikasi merupakan sebuah program yang dibuat dalam sebuah perangkat lunak dengan computer untuk memudahkan pekerjaan atau tugas-tugas seperti penerapan, penggunaan dan penambahan data yang dibutuhkan.

2.2 PENELITIAN TERDAHULU

Penelitian terdahulu yang mengcangkup algoritma genetika adalah sebagai berikut:

2.2.1 PENELITIAN PERTAMA

Ajib Susanto, 2012. Dalam Penelitian berjudul "Rancang Bangun Aplikasi Penjadwalan Praktikum Di Laboratorium Komputer Universitas Dian Nuswantoro Dengan Pendekatan Algoritma Genetika". Melakukan perhitungan algoritma genetika sebagai berikut:

a. Optimasi menggunakan Algoritma Genetika Evaluasi *Fitness*

Faktor-faktor yang mempengaruhi evaluasi *fitness* terhadap hasil solusi adalah sebagai berikut :

1. Kesesuaian kategori matakuliah dengan spesifikasi laboratorium.
2. Adanya tabrakan / tidak terhadap jadwal matakuliah yang lain.

Rumus fitness yang digunakan adalah sebagai berikut :

$$Fitness = \frac{1}{1 + B1xF1 + B2xF2}$$

dengan :

F1 = Kesesuaian spesifikasi lab dengan matakuliah

F2 = Terjadinya tabrakan dengan jadwal yang lain

B1 = Bobot kesesuaian lab

B2 = Bobot jika terjadi tabrakan

Dari rumus nilai fitness di atas dapat terlihat bahwa yang mempengaruhi besar nilai *fitness* adalah harga FN karena harga BN akan tetap selama

proses. Jika harga FN semakin besar maka nilai *Fitness* akan semakin kecil. Karena diinginkan solusi yang memiliki nilai *Fitness* yang besar, maka program ini diharapkan tidak terlalu banyak memunculkan faktorfaktor pengaruh ini dalam solusi yang ditawarkan.

b. Kondisi Selesai

Terdapat tiga kondisi selesai yang dapat menghentikan proses algoritma pemrograman ini, yaitu:

1. Jika setelah beberapa generasi berturut-turut nilai *fitness* terbaik dari populasi tidak mengalami perubahan kembali.
2. Jika jumlah generasi atau iterasi maksimum telah tercapai.
3. Jika nilai fitness terbaik minimal telah tercapai.
4. Jika salah satu kondisi di atas telah diperoleh maka iterasi akan dihentikan dan jika salah satu kondisi selesai ini belum tercapai maka program akan mengulang kembali proses ini / iterasi dari langkah keempat yaitu evaluasi *fitness* terhadap populasi baru tadi.

c. Evaluasi Program

Berikut ini adalah contoh penjadwalan praktikum berdasarkan sampel yang ditentukan.

Tabel 2.1 Sampel Data Penjadwalan

id_jadwal	id_pakai	id_matkul	id_kelompok	id_dosen
3	0	1	1	6
4	0	1	4	13
5	0	6	1	1
7	0	9	4	10
9	0	6	4	12
…dan seterusnya				

Tabel di atas masih menggunakan kode dari masing-masing matakuliah, kelompok, dan dosen. id_pakai masih belum diisi karena jadwal belum di proses, sehingga belum ada pemakaian laboratorium. Keterangan dari tabel di atas dapat dilihat di tabel berikut ini:

Tabel 2.2 Keterangan Sampel Tabel Penjadwalan

Nama Matakuliah	**Dosen**	**Kelompok**
Dasar Pemrograman	Ibnu Utomo WM, S.Kom	1101
Dasar Pemrograman	Suharnawi, M.Kom	1102
Jaringan Komputer	Abdussalam, S.Kom	1101
Bahasa Assembly	Hanny Haryanto	1102
Jaringan Komputer	Agustinus T, S.Kom	1102
...dan seterusnya		

Dengan menggunakan data sampel diatas, proses penjadwalan akan dilakukan sesuai dengan inputan user pada proses algoritma genetika.

Langkah selanjutnya adalah menjalankan algoritma genetika

1. Menentukan nilai *fitness* minimal
2. Membuat generasi baru
3. Melakukan evaluasi terhadap populasi
4. Menyusun populasi
5. Jika nilai fitnes terbaik kurang dari *fitness* minimal maka lakukan mutasi

6. *Looping*

Pada percobaan pertama yang telah dilakukan, generasi dengan nilai *fitness* terbaik akan dimasukan ke tabel kromosom.

Tabel 2.3 Tabel Kromosom

Id_Kromosom	Id_Jadwal	Id_Lab	Hari	Jam	Fitness
1	3	7	1	4	1
2	4	6	1	7	1
3	5	5	4	5	1
4	7	1	3	8	1
5	9	9	4	5	1
dan seterusnya.....					

Setelah ditemukan kromosom dengan *fitness* terbaik, maka dilakukan pengisian pada tabel pemakaian_lab:

Tabel 2.4 Pengisisan Tabel Pemakaian Lab

id_pakai	id_lab	id_jadwal	hari	jam
1	7	3	1	4
2	6	4	1	7
3	5	5	4	5
4	1	7	3	8
5	9	9	4	5
...dan seterusnya				

Berikut adalah update tabel jadwal:

Tabel 2.5 Update Tabel Pemakaian Lab

id_jadwal	id_pakai	id_matkul	id_kelompok	id_dosen
3	1	1	1	6
4	2	1	4	13
5	3	6	1	1
7	4	9	4	10
9	5	6	4	12
...dan seterusnya				

Setelah dilakukan proses optimasi dengan menggunakan algoritma genetika, jadwal matakuliah praktikum berdasarkan sampel di atas menjadi sebagai berikut:

Tabel 2.6 Jadwal Matakuliah

Nama Matakuliah	Dosen	Kelompok	Lab	Hari/Jam
Dasar Pemrograman	Ibnu Utomo WM, S.Kom	1101	D.2.H	Senin 12.30-14.10
Dasar Pemrograman	Suharnawi, M.Kom	1102	D.2.G	Senin 18.30-20.10
Jaringan Komputer	Abdussalam, S.Kom	1101	D.2.E	Kamis 14.10-15.50
Bahasa Assembly	Hanny Haryanto, M.T	1102	D.2.A	Rabu 20.10-21.50
Jaringan Komputer	Agustinus T, S.Kom	1102	D.2.J	Selasa 14.10-15.00
...dan seterusnya				

2.2.1 PENELITIAN KEDUA

Didit Damur Rochman, 2013. Dalam Penelitian berjudul "PENJADWALAN 20 JOB 8 MESIN DENGAN METODE GENETIC ALGORITHM (GA)". Melakukan perhitungan algoritma genetika sebagai berikut:

Pada penelitian ini, CV Boeing Teknik Mandiri mengerjakan 20 *job* pesanan konsumen dengan daftar *job* sebagai berikut:

Tabel 2.7 Daftar *Job*

JOB			
No	**Nama**	***Qty***	***Due date***
100	BUSHING	10	6
200	PILOT PIN SC 037	5	6
300	PILOT PIN SC 038	5	6
400	PILOT PIN SC 039	5	6
500	PILOT PIN SC 040	5	6
600	PILOT PIN SC 041	5	6
700	PILOT PIN SC 042	5	6

JOB			
No	**Nama**	***Qty***	***Due date***
800	PILOT PIN SC 043	5	6
900	PILOT PIN SC 083	5	7
1000	PILOT PIN (ROUGH) SC-121	8	7
1100	PILOT PIN (ROUGH) SC 122	8	7
1200	PIN B06-06	10	7
1300	PIN B06-05	6	7
1400	PIN B04-05	5	6
1500	BUS-JBN.8	4	5
1600	PIN GURIS	4	5
1700	CARRIER YR9	24	7
1800	HING PIN	24	5
1900	HANDLE 02	4	7
2000	HANDLE 03	4	6

Tabel di atas merupakan daftar *job* pesanan yang diterima oleh CV Boeing Teknik Mandiri. Dari tabel di atas diketahui bahwa batas penyelesaian waktu pesanan paling cepat yang diminta konsumen adalah 5 hari, dan untuk yang paling lama adalah 7 hari. Urutan proses operasi dibuat untuk mengetahui urutan suatu *job* dalam melewati mesin pada lantai produksi. Urutan proses operasi dapat dilihat pada tabel berikut:

Tabel 2.8 Proses Operasi Job

Job	**Operasi**						
	1	**2**	**3**	**4**	**5**	**6**	**7**
100	2	1	3	6	7		
200	1	3	5	4	6	7	
300	3	4	1	6	7		
400	1	4	3	6	7		
500	1	3	4	6	7		
600	3	5	1	4	6	7	
700	1	3	4	5	6	7	
800	3	4	1	6	7		
900	1	3	4	6	7		
1000	3	4	1	6	7		
1100	3	5	4	6	1	7	

Job	Operasi						
	1	**2**	**3**	**4**	**5**	**6**	**7**
1200	3	5	4	1	6	7	
1300	3	5	4	1	6	7	
1400	1	3	5	4	6	7	
1500	2	3	1	6	7		
1600	1	4	5	3	6	8	7
1700	3	5	1	7			
1800	4	3	5	1	6	7	
1900	3	1	4	6	7	8	
2000	1	3	4	6	7	8	

Tabel diatas menunjukan urutan mesin yang dilalui sebuah *job* pada proses produksi. Sebagai contoh *job* 100 operasi pertama melalui mesin 2, operasi kedua melalui mesin 1, operasi ketiga melalui mesin 3, operasi keempat melalui mesin 6, danoperasi kelima melalui mesin 7.

Tabel berikut menunjukan waktu siklus masing-masing mesin yang ada di lantai produksi CV Boeing Teknik Mandiri. Waktu siklus yang diolah merupakan waktu siklus tiap mesin setiap operasi dikalikan dengan banyaknya jumlah pesanan.

Tabel 2.9 Siklus Tiap Mesin

Job	Waktu Operasi (Detik)							
	M1	**M2**	**M3**	**M4**	**M5**	**M6**	**M7**	**M8**
100	3777	1264	4096			7465	9032	
200	1237		2406	2526	2756	4240	4532	
300	1871		1771	1282		4500	4532	
400	1437		2121	1876		4660	4532	
500	2372		1981	1866		3535	4532	
600	1396		1367	886	2861	3265	4532	
700	2257		2651	1341	1751	4365	4532	
800	727		1861	1311		3825	4532	
900	1797		2591	1651		4725	4532	
1000	1235		1736	3487		5663	7232	
1100	1083		4224	2455	2122	7863	7232	
1200	3227		2876	2541	1536	9335	9032	

Job	Waktu Operasi (Detik)							
	M1	**M2**	**M3**	**M4**	**M5**	**M6**	**M7**	**M8**
1300	2049		2218	1627	1028	5019	5432	
1400	1375		1926	1471	1651	1367	4532	
1500	1155	736	1128			1439	3632	
1600	995		1288	1147	1150	1319	3632	607
1700	13059		9640		43226		21632	
1800	11139		5776	4351	2906	21759	21632	
1900	887		1288	1163		1407	3632	551
2000	1419		1060	1491		1067	3632	599

Tabel diatas menunjukan waktu yang dibutuhkan sebuah job dalam melewati sebuah mesin sesuai dengan jumlah pesanan. Sebagai contoh job 100 dengan jumlah pesanan sebanyak 10 melewati mesin 1 selama 3777 detik, melewati mesin 3 selama 1264 detik, melewati mesin 3 selama 4096 detik, melewati mesin 6 selama 7465 detik, dan melewati mesin 7 selama 9032 detik.

Beberapa parameter yang digunakan dalam metode Genetic Algorithm pada penelitian ini adalah populasi awal sebanyak 200 individu, probabilitas crossover sebesar 0,95 dan probabilitas mutasi sebesar 0,05. Hasil kriteria yang diperoleh untuk penjadwalan mesin CV Boeing Teknik Mandiri dengan menggunakan metode *Genetic Algorithm* yaitu:

Tabel 2.10 Hasil Metode *Genetic Algorithm*

Prioritas	**Detik**	**Menit**	**Jam**	**Hari**
Cmax :	143461	2391,017	39,85	4,98
Fmax :	143461	2391,017	39,85	4,98
Lmax :	-9063	-151,05	-2,51	0,31
Tmax :	0	0	0	0,00

2.2.3 PENELITI KETIGA

Kustanto. 2011. Melakukan penelitian yang berjudul **Optimasi Rute Distribusi Tabung Gas Elpiji Menggunakan Algoritma Genetika (Studi Kasus: PT. Restu Ajimanunggal Surakarta)**. Melakukan perhitungan algoritma genetika sebagai berikut:

Pada penelitian ini masalah optimasi yang dipilih adalah dalam bidang tansportasi distribusi tabung gas elpiji, dimana akan dicari optimasi dalam pencarian rute terpendek, waktu tercepat dan hambatan dalam perjalanan distribusi tabung gas elpiji dari gudang Restu Ajimanunggal menuju pelanggan dan kembali ke gudang lagi dengan algoritma Genetika. Tujuan dari penelitian ini mengimplementasikan algoritma Genetika dalam menentukan optimasi rute distribusi tabung gas elpiji di PT. Restu Ajimanunggal Surakarta. Metode penelitian yang digunakan adalah materi penelitian, alat penelitian dan cara penelitian meliputi: observasi dan *interview*, membaca *literature*, instalasi program aplikasi, analisa bisnis distribusi tabung gas elpiji, perancangan dan implementasi, pengujian, analisa hasil pengujian dan menarik kesimpulan. Penelitian ini dilakukan dengan cara merancang model graf sistem distribusi tabung gas elpiji sesuai dengan data yang diperoleh, kemudian dari graf tersebut diberi bobot masing-masing berupa jarak dan kecepatan standar penggunaan jalan antar simpul dengan menggunakan program ArcView GIS 3.3. Selanjutnya dihitung dan disimulasikan oleh komputer untuk mendapatkan rute optimal sistem distribusi tabung gas elpiji dengan menggunakan algoritma Genetika. Penelitian ini menghasilkan informasi berupa waktu komputasi algoritma Genetika, nama-nama jalan sebagai rute distribusi tabung gas elpiji yang disertai hambatan perjalanan yang

ada, jarak total tempuh perjalanan dan waktu tempuh total perjalanan distribusi tabung gas elpiji dari gudang menuju pelanggan dan kembali lagi ke gudang PT. Restu Ajimanunggal Surakarta yang disertai animasi rute optimal. Berdasarkan hasil pengujian terlihat bahwa algoritma Genetika dapat menghasilkan rute mendekati optimal dalam kasus sistem distribusi tabung gas elpiji dibandingkan dengan hasil pencarian algoritma Greedy, algoritma Dijkstra dan rute optimal rutinitas seorang driver PT. Restu Ajimanunggal Surakarta sendiri.

Penelitian terdahulu yang mengcangkup algoritma genetika adalah sebagai berikut:

Tabel 2.11 Penelitian Terdahulu

No	Judul	Penulis	Isi	Kontribusi
1.	Rancang Bangun Aplikasi Penjadwalan Praktikum Di Laboratorium Komputer Universitas Dian Nuswantoro Dengan Pendekatan Algoritma Genetika	Ajib Susanto, 2012	Hasil dari proses optimasi penjadwalan praktikum ini mempermudah koordinator penjadwalan praktikum dalam membuat jadwal matakuliah praktikum yang akan dilaksanakan pada semester yang sedang berlangsung.	Memberikan referensi pengambilan data, perancangan dan cara perhitungan algorima genetika pada penjadwalan.
2.	Optimalisasi Penjadwalan Produksi Dengan Metode Algoritma Genetika Di PT. Progress Diecast	Lily Amelia, 2011	Hasil dari penelitian berupa suatu rancangan program aplikasi penjadwalan dengan metode Algoritma Genetika menggunakan software MATLAB	Memberikan gambaran perhitungan algoritma genetika menggunakan data dari tabel.
3	PENJADWALAN 20 JOB 8 MESIN DENGAN METODE GENETIC ALGORITHM	Didit Damur Rochman ,2013	Penjadwalan mesin untuk CV Boeing Teknik Mandiri dengan menggunakan metode *Genetic Algorithm* menghasilkan nilai	Memberikan referensi pebandingan hasil perhitungan metode

No	Judul	Penulis	Isi	Kontribusi
	(GA)		kriteria *Cmax* sebesar 2391,017 menit, nilai kriteria *Fmax* sebesar 2391,017 menit, nilai kriteria *Lmax* sebesar -151,05 menit, dan nilai kriteria *Tmax* sebesar 0 menit.	algoritma genetika dengan metode *Heuristic Dispatching Rules.*
4	Optimasi Rute Distribusi Tabung Gas Elpiji Menggunakan Algoritma Genetika (Studi Kasus: PT. RestuAjimanunggal Surakarta).	Kustanto. 2011.	Penelitian menghasilkan informasi berupa waktu komputasi algoritma Genetika, nama jalan sebagai rute distribusi tabung gas elpiji yang disertai hambatan perjalanan yang ada, jarak total tempuh perjalanan dan waktu tempuh total perjalanan distribusi tabung gas elpiji dari gudang menuju pelanggan yang disertai animasi rute optimal.	Memberikan referensi perhitungan waktu komputasi algoritma genetika dengan panduan jarak tempuh dan waktu total.

2.3 METODE ALGORITMA GENETIKA (AG)

Genetic Algorithm (GA) atau Algoritma Genetika merupakan cabang dari algoritma evolusi yang digunakan untuk memecahkan masalah optimasi. Algoritma ini didasarkan pada proses genetik yang terjadi pada makhluk hidup, dimana perkembangan generasi pada suatu populasi yang alami secara lama kelamaan akan mengikuti seleksi alam yaitu dimana yang kuat yang akan bertahan. Dengan mengikuti teori evolusi tersebut maka algoritma genetik ini dapat digunakan untuk memecahkan masalah yang terjadi pada sehari-hari.

Teori ini pertama kali ditemukan oleh John Holland, dalam algoritma ini bekerja dalam sebuah populasi yang terdiri dari individu-individu yang masing-masing individu merepresentasikan solusi yang ada. Dalam kaitan ini individu dilambangkan sebagai nilai fitness yang akan digunakan untuk menetukan solusi terbaik yang ada. Dalam prosesnya masing-masing individu tersebut akan melakukan reproduksi dengan cara perkawinan silang dengan individu lainnya untuk menghasilkan keturunan baru yang lebih baik. Dengan cara tersebut maka individu baru akan terus bermunculan, sedangkan untuk individu yang lemah akan mati sendiri. Semakin banyak proses perkawinan silang dilakukan, maka akan semakin banyak kemungkinan solusi yang akan diperoleh (Entin, 2010).

Algoritma Genetika pertama kali ditemukan pada tahun 1960. Algoritma Genetika merupakan salah satu algoritma pemodelan evolusi (*evolutio-nary modeling*) yang dikembangkan oleh John Holland pada dekade 1960 – 1970-an dengan tujuan memodelkan perkembangan kemampuan adaptasi sebuah sistem, salah satunya sistem penjadwalan produksi. Prosedur penjadwalan dengan metode Algoritma Genetika adalah sebagai berikut :

1. Inisialisasi adalah tahapan membentuk *fitness*, constraint dan jumlah populasi awal. Pemben-tukan populasi awal dilakukan dengan cara membentuk kromosom-kromosom, dimana setiap kromosom berisi gen-gen yang diacak.
2. Perhitungan nilai fitness tiap kromosom se-banyak populasi awal.

   Seleksi adalah tahapan memilih 10 kromosom terbaik (kromosom elit) dari perhitungan nilai fitness tiap kromosom. Tujuan pemilihan 10 kromosom terbaik adalah sebagai alternatif pemilihan individu terbaik pada iterasi ke -T.

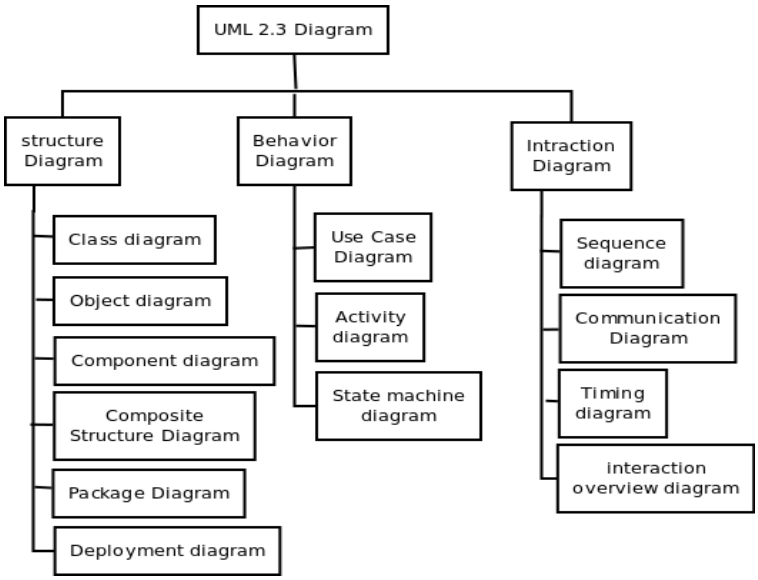
3. Crossover adalah pertukaran gen dalam kedua kromosom orang tua (*parent*) yang kemudian menghasilkan kromosom anak (*child*). Kromo-som yang mengalami crossover adalah kromo-som yang mempunyai peluang crossover (pc) lebih kecil dari pada bilangan acak yang akan dibangkitkan. Proses crossover akan menggu-nakan metode order crossover.
4. Mutasi adalah proses membentuk keturunan (*offspring*) dengan cara mengubah gen dari parent. Proses ini akan merekontruksi bentuk dari gen-gen tiap kromosom. Individu yang akan mengalami proses mutasi adalah individu pada gen tertentu yang mempunyai peluang mutasi (pm) lebih kecil dari bilangan acak yang akan dibangkitkan.
5. Keputusan kapan proses iterasi berhenti ter-gantung pada penentuan N iterasi. Bilamana iterasi ke-T sama dengan N iterasi maka proses perhitungan akan berhenti.
6. Pemilihan individu terbaik adalah kromosom dengan *fitness* terbaik setelah proses iterasi berhenti. (Lily, 2011)

2.4 UML

Unified Modeling Language merupakan salah satu alat bantu yang dapat digunakan dalam bahasa pemograman yang berorientasi objek, saat ini UML akan mulai menjadi standar masa depan bagi industri pengembangan sistem atau perangkat lunak yang berorientasi objek sebab pada dasarnya UML digunakan oleh banyak perusahaan raksasa seperti IBM, *Microsoft*, dan sebagainya. (Sumber: http://informatika.web.id/pengertian-uml.htm).

1. Diagram UML

Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar dibawah ini.



Gambar 2.1 Diagram UML
(Sumber: Rosa, 2012:121)

Berikut ini penjelasan singkat dari pembagian kategori tersebut.

a. *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
b. *Behavior diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
c. *Interaction Diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

2. *Class Diagram*

Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas ang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribute dan metoda atau operasi.

a. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.

b. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas

Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut:

a. Kelas main

Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.

b. Kelas yang menangani tampilan sistem

Kelas yang mendefinisikan dan mengatur tampilan ke pemakai.

c. Kelas yang diambil dari pendefinisian *use case*

Kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian *use case*

d. Kelas yang diambil dari pendefinisian data

Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data

Tabel 2.12 Simbol-simbol yang ada pada diagram kelas

Sumber: Rosa, 2012:123-124

Simbol	Deskripsi
kelas nama_kelas +atribut +operasi()	Kelas pada struktur sistem
Antarmuka/*interface* nama_interface	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi/*association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*
Asosiasi berarah/*direct association*	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi	Relasi antar kelas denganmakna generalisasi-spesialiasi (umum khusus)
Kebergantungan/*depedency*	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi/aggregation	Relasi antar kelas dengan makna semua bagian (whole-part)

3. *Object Diagram*

Diagram objek menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem. Pada diagram objek kelas harus dipakai objeknya, karena jika tidak, pendefinisian kelas itu tidak dapat dipertanggung jawabkan. Hubungan *link* pada diagram objek merupakan hubungan memakai dan dipakai di mana dua buah objek akan dihubungkan oleh *link* jika ada objek yang dipakai oleh

objek lainnya.

Tabel 2.13 Simbol-simbol yang ada pada diagram objek
Sumber: Rosa, 2012:124

Simbol	Deskripsi
Objek nama_objek:nama_kelas atribut = nilai	Objek dari kelas yang berjalan saat sistem dijalankan
Link	Relasi antar objek

4. *Use Case Diagram*

Diagram *Use Case* 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. Secara kasar, use case digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami.

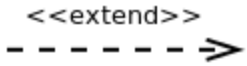
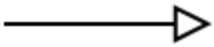
Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan use case.

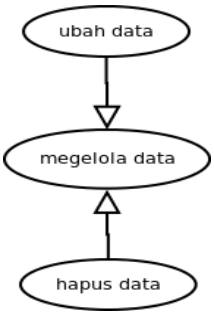
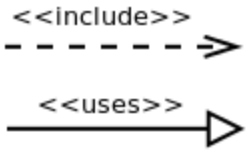
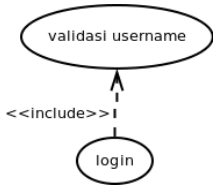
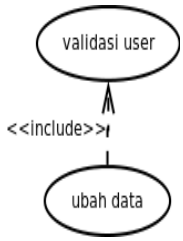
a. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.

b. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antarunit atau aktor.

Tabel 2.14 Simbol-simbol diagram *use case*

Sumber: Rosa, 2012:131-133

Simbol	Deskripsi
Use Case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*
 Aktor/actor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
Asosiasi/association	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor
Ekstensi/extend 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case t*ambahan itu; mirip dengan prinsip inheritance pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan, misal: arah panah mengarah pada *use case* yang ditambahkan.
Generalisasi/generalization 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum lainnya, misalnya:

Simbol	Deskripsi
	 arah panah mengarah pada *use case* yang menjadi generalisasinya (umum)
Menggunakan/include /uses 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankannya *use case* ini. Ada dua sudut pandang yang cukup besar mengenai include di *use case*: a. *include* berarti use case yang ditambahkan akan selalu dipanggil saat use case tambahan dijalankan. Misal  pada kasus berikut: b. *include* berarti use case yang tambahan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan, misal pada kasus berikut:  kedua interpretasi diatas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.

5. *Activity Diagram*

Activity diagram menggambarkan workflow (aliran kerja) atau aktifitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan disini adalah bahwa diagram aktifitas sistem bukan apa yang dilakukan aktor, jadi aktifitas yang dapat dilakukan oleh sistem.

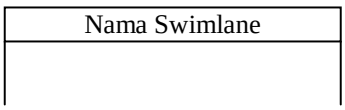
Diagram aktifitas juga banyak digunakan untuk mendefinisikan hal-hal berikut:

a. Rancangan proses bisnis di mana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
b. Urutan atau pengelompokkan tampilan dari sistem / *user interface* di mana setiap aktifitas dianggap memiliki sebuah rancangan antarmuka tampilan.
c. Rancangan pengujian dimana setiap aktifitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.

Tabel 2.15 Simbol-simbol diagram aktifitas

Sumber: Rosa, 2012:134-135

Simbol	Deskripsi
Staus Awal 	Status awal aktivitas sistem, sebuah diagram aktifitas memiliki sebuah status awal
Aktifitas 	Aktifitas yang dilakukan sistem, aktifitas biasanya diawali dengan kata kerja.
Percabangan/decision 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan/join 	Asosiasi penggabungan dimana lebih dari satu aktifitas digabungkan menjadi satu.

Simbol	Deskripsi
Status akhir 	Staus akhir yang dilakukan sistem, sebuah diagram aktifitas memiliki sebuah status akhir.
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktifitas yang terjadi

2.5 *JAVA*

Java adalah nama sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer yang berdiri sendiri *(standolone)* ataupun pada lingkungan jaringan, java berdiri di atas mesin penerjemah (*interpreter*) yang diberi nama *Java Virtual Machine* (JVM). JVM inilah yang membuka kode bit *(bytecode)* dalam file, class dari suatu progam sebagai representasi langsung progam yang berisi bahasa mesin, oleh karna itu bahasa *java* disebut bahasa pemrogaman yang portable karna dapat dijalankan pada berbagai sistem operasi, asalkan padsa sistem operasi tersebut terdapat JVM, bahasa pemrogaman java digunakan untuk membuat aplikasi-aplikasi yang dapat diletakkan diberbagai macam perangkat elektronik, sehingga java harus bersifat tidak bergantung pada platform *(platform independent)*, iyulah yang menyebabkan dalam dunia pemrogaman java di kenal adanya istilah '*write once, run everywhere*', yang berarti kode progam hanya ditulis sekali, namun dapat dijalankan di bawah kumpulan pustaka *(platform)* manapun tanpa harus melakukan perubahan kode progam. (shalahudin, 2014).

2.6 MYSQL

MySQL (dibaca : mi-se-kyu-el) merupakan *software* yang tergolong sebagai DBMS (*Database Management System*) yang bersifat *open source*. *Open source* menyatakan bahwa sistem ini dilengkapi dengan *source code* (*code* yang dipakai untuk membuat MySQL). Selain tentu saja bentuk *executable*-nya atau kode yang dapat dijalankan secara langsung dalam sistem operasi. MySQL awalnya dibuat oleh perusahaan konsultan yang bernama TeX yang berlokasi di Swedia. Saat ini pengembangan MySQL berada dibawah naungan MySQL AB (Sianipar, R.H, 2015:2).