BAB II
TINJAUAN PUSTAKA

2.1. Teori Judul

2.1.1. Pengertian Arsitektur Model View Controller (MVC)

Aplikasi yang dikembangkan sebagian besar berinteraksi dengan pengguna (*user*). Oleh karena itu memerlukan cara agar aplikasi yang bersangkutan dapat berkomunikasi dengan para penggunanya. Untuk lebih mengoptimalkan proses perancangan aplikasi yang berkomunikasi dengan pengguna (*user*) maka dilakukan pemisahan komponen-komponen antarmuka berbasis GUI dengan logika-logika bisnisnya (*business logic*). Beberapa pendekatan telah dilakukan untuk memisahkan aspe presentasi dengan logika bisnis milik aplikasi, salah satunya adalah MVC (*Model View Controller*).

Menurut David Naista (2016:10) menyatakan, Model View Contoller atau disingkat MVC adalah sebuah metode untuk membuat sebuah aplikasi dengan memisahkan antara data (Model) dari tampilan (View) dan cara bagaimana memprosesnya (Controller).

Jadi, MVC (*Model View Contoller*) merupakan salah satu pendekatan pemisahan komponen presentasi dan logika bisnis yang populer digunakan pada *design pattern* atau arsitektur aplikasi.

2.1.2. Pengertian Aplikasi

Teknologi pada saat ini dapat membuat suatu pekerjaan menjadi begitu praktis dan cepat untuk dilakukan. Salah satunya adalah pemanfaatan komputer dan aplikasi yang digunakan. Pengguna teknologi baik individu maupun kelompok pada saat ini sangat membutuhkan suatu aplikasi yang dapat mempercepat mereka dalam menyelesaikan pekerjaannya.

Hampir setiap perusahaan kini memiliki sebuah aplikasi yang dipesan melalui *software developer* untuk mengerjakan suatu tugas tertentu. Lebih kompleks, penggunaan aplikasi sendiri dimiliki oleh setiap bagian atau tingkat

manajemen yang ada di suatu perusahaan dengan kegunaan dan fungsi yang berbeda-beda sesuai dengan tugas pokok dan fungsi dari tingkat manajemen tersebut.

Aplikasi adalah alat terapan yang difungsikan secara khusus dan terpadu sesuai kemampuan yang dimilikinya (Sutabri, 2012:147). Pendapat lainnya mengemukakan bahwa aplikasi adalah kumpulan perintah program yang dibuat untuk melakukan pekerjaan-pekerjaan tertentu (khusus) Hendrayudi (2009:143). Sehingga dengan aplikasi manusia dapat mengerjakan tugas-tugas mereka dengan cepat sesuai dengan kemampuan yang dimiliki aplikasi tersebut.

2.1.3. Pengertian Web

Web ada karena perkembangan teknologi informasi dan komunikasi. Melalui perkembangan teknologi informasi, terciptalah suatu jaringan antar komputer yang saling berkaitan. Jaringan yang dikenal dengan istilah internet secara terus-menerus menjadi pesan–pesan elektronik seperti *e-mail*.

Web merupakan halaman situs sistem informasi yang dapat diakses secara cepat dan sebagai media yang digunakan untuk melakukan penyebaran informasi, pengolahan data, dan sebagainya yang cukup banyak digunakan pada saat ini. Web didefiniskan sebagai sistem interkoneksi komputer internet (disebut server) yang mendukung dokumen-dokumen berformat multimedia (Williams dan Sawyer, 2007:17).

Williams dan Sawyer juga menjelaskan bahwa web adalah teknologi berbasis multimedia yang memungkinkan anda untuk mengakses lebih dari sekadar teks. Sehingga, anda bisa mendownload gambar seni, audio, video, animasi, dan game interaktif.

2.1.4. Pengertian Pengiriman

Kamus Besar Bahasa Indonesia (2014:703) menjelaskan, “Pengiriman adalah proses, cara, perbuatan, mengirimkan bantuan, bahan pangan.”

Pengiriman merupakan proses, cara, perbuatan mengirimkan suatu barang. (http://kbbi.web.id/data)

Jadi, pengiriman adalah suatu cara, perbuatan mengirimkan barang melalui suatu proses atau cara tertentu.

2.1.5. Pengertian Barang

Barang yang sering kita gunakan untuk memenuhi kebutuhan-kebutuhan kita diantaranya memiliki ciri-ciri yaitu Berwujud, Memiliki nilai dan manfaat yang dapat dirasakan saat digunakan, Bila digunakan, nilai, manfaat dan bendanya sendiri dapat berkurang atau bahkan habis

Kamus Besar Bahasa Indonesia (2014:139), menyatakan "Barang adalah sesuatu yang berwujud atau berjasad".

Suherman Rosyidi (2014:45), "Barang adalah benda-benda yang dapat dipakai untuk memenuhi kebutuhan manusia."

Jadi, Barang adalah suatu benda yang berwujud digunakan untuk memenuhi kebutuhan manusia.

2.1.6. Pengertian Inkubator Bisnis

Inkubator bisnis merupakan salah satu strategi yang dilakukan dan dikelola oleh pemerintah atau pihak terkait guna memberikan nilai tambah terhadap perekonomian melalui percepatan pengembangan usaha baru. Inkubator memiliki peran sebagai pencipta lapangan pekerjaan baru, menumbuhkan pelaku usaha baru, dan menjadi wadah untuk mengimplementasikan berbagai inovasi produk yang dihasilkan.

Inkubator bisnis adalah organisasi yang menyediakan fasilitas bisnis bersama dengan biaya rendah secara temporer bagi usaha-usaha baru. Inkubator bisnis dijalankan oleh organisasi nirlaba, perguruan tinggi, dan perusahaan bertujuan membantu usaha kecil dengan biaya rendah, layanan pendukung, saran manajemen, pertukaran ide dan pemberian akses yang mudah kepada pengusaha (Soegoto, 2010:96).

2.1.7. Pengertian Politeknik

Selain akademi, institut, sekolah tinggi, dan universitas terdapat suatu lembaga yang menyelenggarakan pendidikan vokasi dalam sejumlah bidang pengetahuan khusus yang disebut Politeknik. Istilah politeknik berasal dari bahasa Yunani *polú* atau *polý* yang berarti "banyak" dan *tekhnikós* yang berarti "seni".

Meski setara sebagai lembaga pendidikan tinggi, akademi atau universitas memiliki perbedaan yang paling menonjol dengan politeknik yaitu pada politeknik memiliki porsi praktek yang lebih banyak. Perbedaan antara akademi dengan politeknik selain jumlah bidang ilmu yang dipelajari adalah pada politeknik porsi praktek lebih banyak (Djojodibroto, 2004:18).

Politeknik lebih bertujuan untuk menyiapkan mahasiswanya agar memiliki kemampuan profesional dalam mengoperasikan, mengembangkan, dan menerapkan ilmu pengetahuan dan teknologi informasi. Pengertian politeknik sendiri adalah salah satu bentuk perguruan tinggi yang menyelenggarakan program pendidikan profesional dalam sejumlah bidang pengetahuan khusus (Djojodibroto, 2004:18).

2.1.8. Pengertian Politeknik Negeri Sriwijaya

Politeknik Negeri Sriwijaya adalah perguruan tinggi negeri yang terdapat di kota Palembang, Sumatera Selatan, Indonesia. Politeknik Negeri Sriwijaya, dahulunya bernama Politeknik Universitas Sriwijaya secara resmi dibuka pada tanggal 20 September 1982.

2.1.9. Pengertian Arsitektur Model View Controller (MVC) pada Aplikasi Web Pengiriman Barang di Inkubator Bisnis Politeknik Negeri Sriwijaya

Arsitektur Model View Controller (MVC) pada Aplikasi Web Pengiriman Barang di Inkubator Bisnis Politeknik Negeri Sriwijaya adalah Merancang sebuah metode untuk membuat sebuah aplikasi dengan memisahkan antara data (Model)

dari tampilan (View) dan cara bagaimana memprosesnya (Controller) pada aplikasi pengiriman barang di Inkubator Bisnis Politeknik Negeri Sriwijaya.

2.2. Teori Khusus

2.2.1. Pemrograman Berorientasi Objek

2.2.1.1.Pengertian Pemrograman Berorientasi Objek

Menurut Sukamto dan Shalahuddin (2013:100), pemrograman berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya.

Williams dan Sawyer (2007:538), pemrograman berorientasi objek (OOP, diucapkan ("oop") merupakan data dan instruksi untuk pemrosesan dimana data dikombinasikan menjadi "objek" yang cukup memadai yang bisa digunakan pada program lain.

Sehingga berdasarkan kedua definisi di atas maka dapat disimpulkan bahwa Pemrograman Berorientasi Objek merupakan suatu strategi dalam membangun perangkat lunak dimana data dikombinasikan menjadi objek yang berisi data dan operasi yang bisa digunakan.

2.2.1.2.Ciri Pemrograman Berorientasi Objek

Menurut Siallagan (2009:149), Ciri-ciri atau karakteristik pemograman berorientasi objek, antara lain:

a. Abstraksi (*Abstraction*)

   Abstraksi adalah pengabstrakan atau melakukan seleksi terhadap aspek-aspek tertentu suatu masalah. Abstraksi digunakan untuk penyembunyian kerumitan dari suatu proses. Sebagai contoh, dalam membuat suatu sistem, ada tombol-tombol yang dapat digunakan. Operator atau pengguna tidak perlu berpikir tentang pembuatan tombol tersebut, tetapi yang penting mereka dapat menggunakannya.

b. Pembungkusan (*Encapsulation*)

Pembungkusan sering pula disebut pengkapsulan. Artinya, proses membuat paket (memaketkan) data objek bersama dengan metode-metodenya. Berdasarkan kode program, proses memisahkan aspek-aspek objek dilakukan dengan pembungkusan. Proses pembungkusan itu sendiri merupakan cara atau mekanisme untuk melakukan abstraksi.

c. Pewarisan (*Inheritance*)

Pewarisan adalah memberikan atau mewariskan sesuatu kepada keturunan berikutnya. Misalnya, seorang anak pasti akan mewarisi beberapa sifat atau perilaku yang dimiliki oleh ibu/bapaknya. Dalam konteks ini, suatu kelas dalam program dapat diturunkan menjadi kelas-kelas baru lainnya yang akan mewarisi beberapa sifat atau perilaku dari kelas induknya.

d. Polimorfisme (*Polymorphism*)

Polimorfisme adalah suatu kejadian ketika objek dapat mengungkap banyak hal melalui satu cara yang sama.

2.2.2. Metodelogi RUP (*Rational Unified Process*)

Menurut Sukamto dan Shalahudin (2014:30), "*Rational Unified Process (RUP)* merupakan suatu metode rekayasa perangkat lunak yang dikembangkan dengan mengumpulkan berbagai *best practises* yang terdapat dalam industri pengembangan perangkat lunak. Ciri khas metode ini adalah menggunakan *use-case driven* dan pendekatan iteratif untuk siklus pengembangan perangkat lunak. Gambar dibawah ini menunjukkan secara keseluruhan arsitektur yang dimiliki *RUP*."

RUP menggunakan konsep *object oriented*, dengan aktifitas yang berfokus pada pengembangan model dengan menggunakan *Unified Model Language* (*UML*).

Melalui gambar dibawah dapat dilihat bahwa *RUP* memiliki, yaitu:

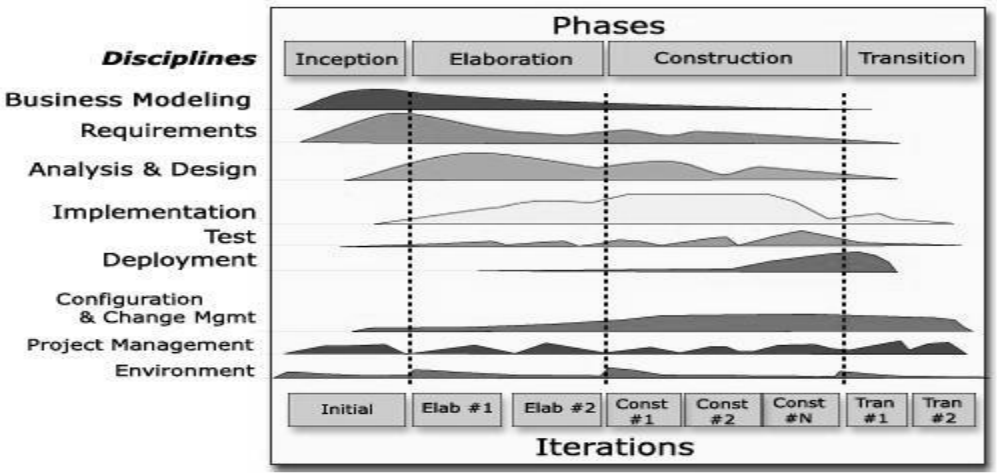
a. Dimensi Pertama

Digambarkan secara horizontal. Dimensi ini mewakili aspek-aspek dinamis dari pengembangan perangkat lunak. Aspek ini dijabarkan dalam tahapan pengembangan atau fase. Setiap fase akan memiliki suatu *major milestone* yang menandakan akhir dari awal dari *phase* selanjutnya. Setiap *phase* dapat berdiri dari satu beberapa iterasi. Dimensi ini terdiri atas *Inception, Elaborationn, Construction,* dan *Transition.*

b. Dimensi Kedua

Digambarkan secara vertikal. Dimensi ini mewakili aspek-aspek statis dari proses pengembangan perangkat lunak yang dikelompokkan ke dalam beberapa disiplin. Proses pengembangan perangkat lunak yang dijelaskan kedalam beberapa disiplin terdiri dari empat elemen penting, yakni *who is doing, what, how,* dan *when.* Dimensi ini terdiri atas:

Bussines Modeling, Requirement, Analysis and Design, Implementation, Test, Deployment, Configuration, dan Change Management, Project Management,Environment.



Gambar 2.1 Arsitektur *Rational Unified Process*

2.2.3.Penerapan Tahap Metodelogi Pengembangan Perangkat Lunak dengan RUP

Tedapat 4 (empat) tahap dalam pengembangan perangkat lunak menggunakan metodelogi RUP, yaitu:

1. *Inception*

   Pada tahap ini pengembangan mendefinisikan batasan kegiatan, melakukan analisis kebutuhan *user*, dan melakukan perancangan awal perangkat lunak (perancangan arsitektural dan *use case*). Pada akhir fase ini prototipe perangkat lunak versi *Alpha* harus sudah dirilis.

2. *Elaboration*

   Pada tahap ini dilakukan perancangan perangkat lunak mulai dari menspesifikasikan fitur perangkat lunak hingga perilisan prototipe versi *Betha* dari perangkat lunak.

3. *Construction*

   Pengimplementasian rancangan perangkat lunak yang telah dibuat dilakukan pada tahap ini. Pada akhir tahap ini, perangkat lunak versi akhir yang sudah disetujui administrator dirilis beserta dokumentasi perangkat lunak.

4. *Transition*

   Instanlasi, deployment dan sosialisasi perangkat lunak dilakukan pada tahap ini.

2.2.4. Aliran Kerja Utama RUP

Adapun aliran kerja utama pada Metodologi RUP sebagai berikut:

a. Pemodelan Bisnis (*Bussines Modeling*)

   Mendeskripsikan struktur dan proses-proses bisnis organisasi.

b. Kebutuhan (*Requirement*)

   Mendefinisikan kebutuhan perangkat lunak dengan menggunakan metode *use case.*

c. Analisis dan Perancangan (*Analysis and Design*)

Mendeskripsikan berbagai arsitektur perangkat lunak dari berbagai sudut pandang.

d. Implementasi (*Implementation)*

Menuliskan kode-kode program, menguji, dan mengintegrasikan unit-unit programnya.

e. Pengujian (*Test*)

Mendeskripsikan kasus uji, prosedur, dan alat ukur pengujian.

f. *Deployment*

Menangani konfigurasi sistem yang akan diserahkan.

2.2.3.1. Aliran Kerja Pendukung RUP

Adapun aliran kerja pendukung *RUP* adalah sebagai berikut:

a. Manajemen konfigurasi dan perubahan (*configuration and change management*)

mengendalikan perubahan dan memelihara artifak-artifak proyek.

b. Manajemen proyek (*Project Management*)

Mendeskripsikan berbagai stategi pekerjaan dengan proses yang berulang.

c. Lingkungan (*Environment*)

Menangani infrastuktur yang dibutuhkan untuk mengembangkan sistem.

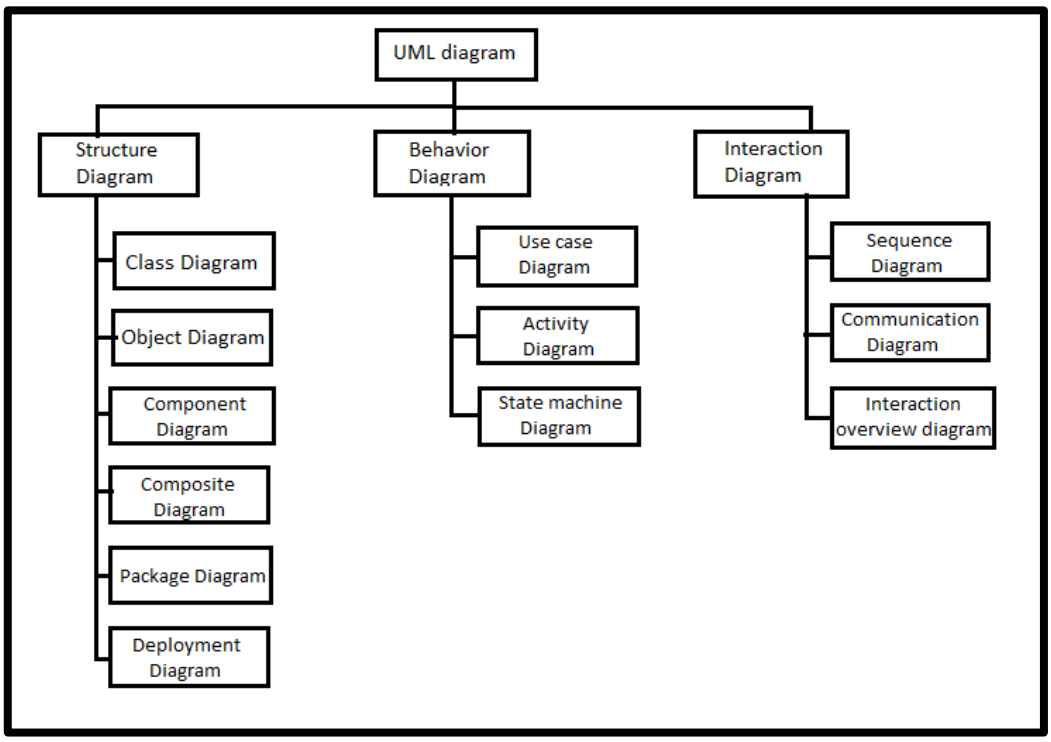
2.2.4. UML (*Unified Modeling Language*)

2.2.4.1. Pengertian UML (*Unified Modeling Language*)

Menurut Pratama (2014:48), "UML (Unified Modeling Language) adalah standarisasi internasional untuk notasi dalam bentuk grafik, yang menjelaskan tentang analisis dan desain perangkat lunak yang dikembangkan dengan pemrograman berorientasi objek."

2.2.4.2. Macam-macam Diagram UML (*Unified Modeling Language*)

Sukamto dan Shalahuddin (2013:137) menjelaskan bahwa UML terdiri dari 13 (tiga belas) macam diagram yang dikelompokkan dalam 3 (tiga) kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar di bawah ini:



Gambar 2.2 Macam-macam diagram UML

Berikut ini penjelasan singkat dari pembagian kategotri 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.

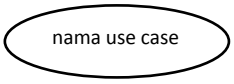
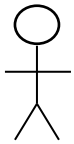
2.2.4.3. Diagram *Use Case* (*Use Case Diagram*)

Menurut Sukamto dan Shalahuddin (2013:155), "Use Case digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu."

Al Fatta (2007:91), "Use Case adalah metode berbasis teks untuk menggambarkan dan mendokumentasikan proses yang kompleks."

Sukamto dan Shalahuddin juga menjelaskan bahwa terdapat simbol-simbol yang digunakan di dalam Diagram *Use Case*, yaitu:

Tabel 2.1 Simbol-simbol dalam Diagram *Use Case*

No.	Simbol	Nama	Keterangan
1.		***Use Case*** *Use case*	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar antar unit atau aktor.
2.	 nama aktor	**Aktor** *Actor*	Merupakan seseorang atau sesuatu yang berinteraksi dengan sistem.
3.		**Asosiasi** *Asosiation*	Komunikasi antara aktor dan *use case* berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
4.		**Generalisasi** *Generalization*	Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.

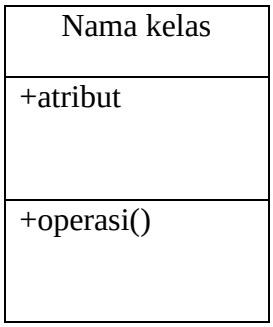
5.	<<include>> ---------->	***Include*** *Include*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya.
6.	<<extend>> ---------->	**Ekstensi** *Extend*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu.

(Sumber: Sukamto dan Shalahuddin, 2013:156-158)

2.2.4.4. Diagram Kelas (*Class Diagram*)

Menurut Sukamto dan Shalahuddin (2013:141), "Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem."

Kelas memiliki yang disebut atribut dan metode atau operasi. Berikut ini adalah struktur dari diagram kelas:



Gambar 2.3 Struktur Diagram Kelas (*Class Diagram*)

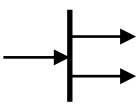
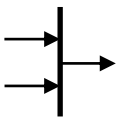
a. Atribut, merupakan variabel-variabel yang dimiliki oleh suatu kelas.
b. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

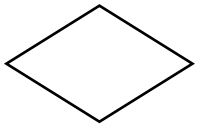
2.2.4.5. Diagram Aktivitas (*Activity Diagram*)

Menurut Sukamto dan Shalahuddin (2013:161), "Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atasu menu yang ada pada perangkat lunak."

Berikut adalah simbol-simbol yang digunakan di dalam Diagram Aktivitas atau *Activity Diagram*:

Tabel 2.2 Simbol-simbol dalam Diagram Aktivitas (*Activity Diagram*)

No.	Simbol	Nama	Keterangan
1.		**Titik Awal** *Start Point*	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2.		**Titik Akhir** *End Point*	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
3.		**Aktivitas** *Activities*	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
4.		**Percabangan** *Fork*	Asosiasi percabangan dimana jika ada pilihan aktivitas yang lebih dari satu.
5.		**Penggabungan** *Join*	Asosiasi penggabunggan dimana lebih dari satu aktivitas digabungakan menjadi satu.

6.		**Keputusan** *Decision*	Pilihan untuk mengambil keputusan.

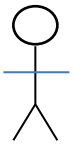
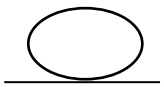
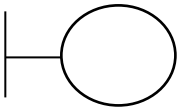
(Sumber: Sukamto dan Shalahuddin, 2013:148-149)

2.2.4.6. Diagram Sekuensial (*Squence Diagram)*

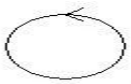
Sukamto dan Shalahuddin (2013:165), "Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *mesage* yang dikirimkan dan diterima antar objek."

Berikut adalah simbol-simbol yang ada pada diagram sekuen:

Tabel 2.3 Simbol-simbol dalam Diagram Sekuensial (*Sequence Diagram*)

No.	Simbol	Nama	Keterangan
1.		**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 gamabr orang, belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
2.		**Kelas Entitas** *Entity Class*	Menggabarkan hubungan kegiatan yang akan dilakukan.
3.		**Kelas Batas** *Boundary Class*	Menggambarkan sebuah penggambaran dari form.

Lanjutan Tabel 2.3 Simbol-simbol dalam Diagram Sekuensial (*Sequence Diagram*)

4.		**Kelas Kendali** *Control Class*	Menggambarkan penghubung *boundary* dengan tabel.
5.		***Garis Hidup*** *A Focus of Control and A Life Line*	Tempat menggambarkan tempat mulai dan berakhirnya sebuah *message*.
6.		**Spesifikasi** *Specification*	Spesifikasi komunikasi dari atar objek yang memuat informasi-infromasi tentang aktivitas.

(Sumber: Sukamto dan Shalahuddin, 2013:165-168)

2.2.3. Pemrograman Java

2.2.5.1.Pengertian Pemrograman Java

Menurut Sukamto dan Shalahuddin (2013:103), "*Java* adalah bahasa pemrograman objek murni karena semua kode programnya dibungkus dalam kelas."

2.2.5.2. Tipe Data dalam Java

Tabel 2.4 Tipe Data dalam Java

No.	Tipe data		Ukuran Memori	Batasan Nilai
1	*Integer*	*Int*	4 *byte*	-2.147.486.648 s/d 2.147.486.6
		Short	2 *byte*	-32.768 s/d 32.767
		Long	8 *byte*	-9.223.372.036.854.775.808L

				s/d 9.223.372.036.854.775.807L
		Byte	1 *byte*	-128 s/d 127

Lanjutan Tabel 2.4 *Tipe Data dalam Java*

2	*floating point*	*Float*	4 *byte*	± 3.40282347E+38F (7 digit signifikan)
		double	8 *byte*	± 1.79769313486231570E+308 (15 digit signifikan)
3	karakter dan *string*	*Char*	1 karakter	Sebuah objek string dan manipulasinya.
		String	Banyak karakter	

2.2.5.3. Operator dalam Java

a. Operator Aritmatika

Operator aritmatika adalah operator-operator yang digunakan untuk mengoprasikan perhitungan (aritmatika). Bahasa pemograman *java* menyediakan operator-operator aritmatika untuk memanipulasi variabel data.

Tabel 2.5 Operator Aritmatika

Operator	Keterangan
+	Penjumlahan
-	Pengurangan
*****	Perkalian
/	Pembagian
%	Modulus (sisa bagi)

b. Operator Relasional

Operator relasional adalah operator hubungan (relasi) yang membandingkan kedua nilai *operand* dan hasilnya berupa nilai *boolean*, yaitu benar (*true*) atau salah (*false*).

Tabel 2.6 Operator Relasional

Operator	Keterangan
==	Sama dengan (membandingkan bukan penugasan)
!=	Tidak sama dengan
>	Lebih besar
<	Lebih kecil
>=	Lebih besar sama dengan
<=	Lebih kecil sama dengan

c. Operator Logika/*Boolean*

Operator logika adalah operator yang digunakan terhadap *operand* bertipe *Boolean* yang hasilnya benar (*true*) atau salah (*false*).

Tabel 2.7 Operator Logika/*Boolean*

Operator	Keterangan
&	Logika AND
|	Logika OR
^	Logika XOR
!	Logika NOT