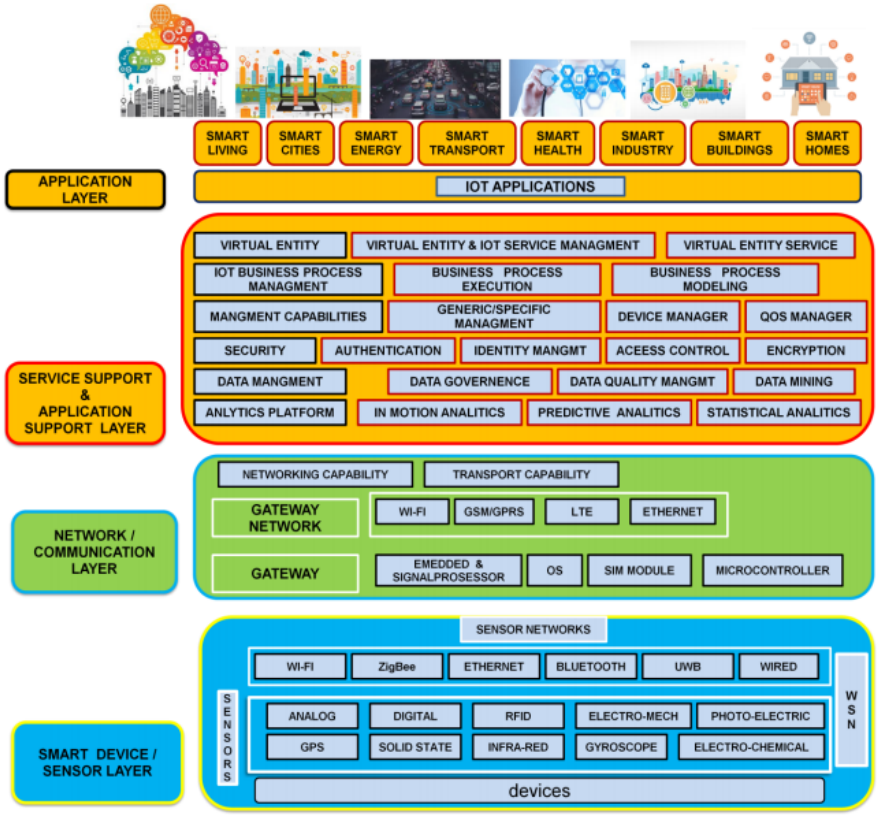
BAB II
TEORI PENUNJANG

2.1 Internet of Thing's

Internet of Thing's merupakan mekanisme jaringan nirkabel yang memungkinkan setiap benda dapat saling berkomunikasi [1]. Dengan hadirnya IoT mengendalikan dan memantau benda/objek dari jarak jauh bukan lagi hal yang mustahil. Dengan IoT sistem dapat melakukan analisa, monitoring, membuat sebuah keputusan, dan memiliki kecerdasan buatan. Saat ini teknologi IoT populer digunakan di dunia untuk membantu menyelesaikan permasalahan manusia. IoT dapat dikategorikan menjadi tiga, yaitu (1) *people to people*, (2) *people to machine*, dan (3) *machine to machine* [1]. Tujuan utama IoT adalah memungkinkan segala sesuatu dapat diakses dimana saja, kapan saja menggunakan layanan jaringan apapun [1]. Arsitektur IoT terdiri dari beberapa layer teknologi yang berbeda, namun saling berkomunikasi setiap layer untuk menunjang kinerja sistem IoT. Berikut ini layer IoT :

1. *Smart Device / Sensor Layer* : Berfungsi untuk mengambil data, mengubah besara fisik menjadi besara digital / elektrik. [1]
2. *Gateway and Networks* : Berfungsi untuk menunjang jalur komunikasi data yang dihasilkan oleh sensor-sensor, mendukung komunikasi *machine to machine* (M2M) dan aplikasi [1].
3. *Management Service Layer* : Berfungsi untuk mengolah data / informasi, analisa, kontrol keamanan, proses pemodelan dan manajemen perangkat [1].
4. *Application Layer* : pengaplikasian IoT dalam bentuk *smart environments* dalam domain : transportasi, perkotaan, retail, agrikultur, dll. [1]

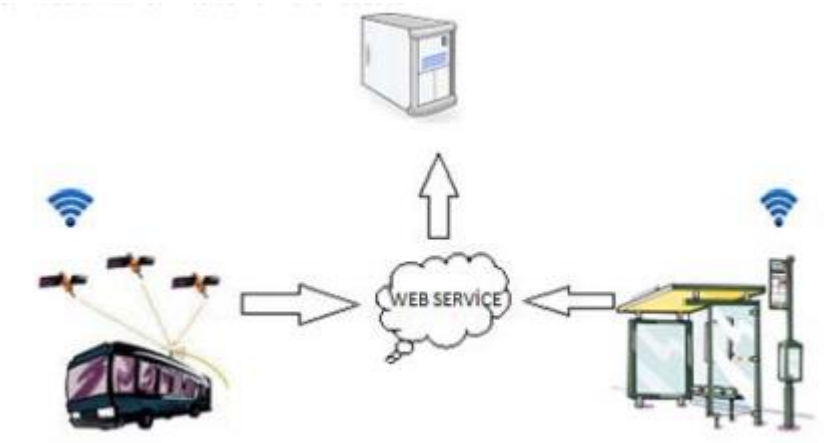


Gambar II-1 IoT Architecture

2.2 Vehicle Tracking System

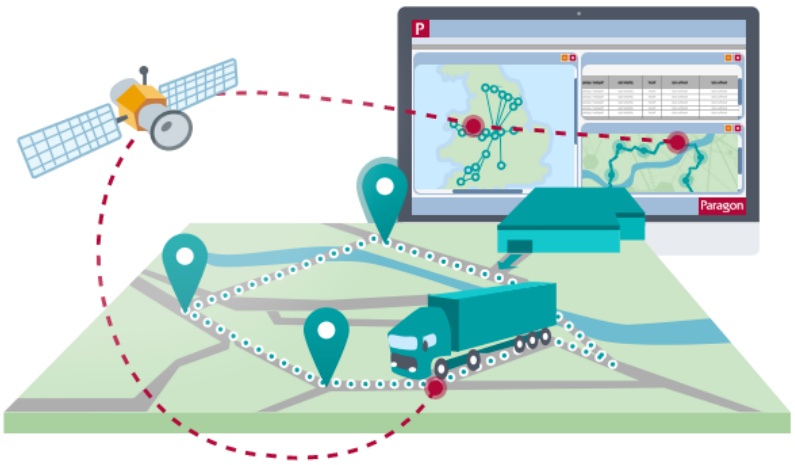
Kemajuan zaman dan teknologi membuat industri otomotif berlomba untuk menciptakan kendaraan yang canggih, mulai dari memasang sensor parkir, kamera, dan sistem keamanan. Namun saat ini, yang populer dan umum digunakan adalah *vehicle tracking system*. Pada awal kemunculannya, *vehicle tracking system* diinisialisasi dari *fleet management system* yang digunakan pada kendaraan penngirim barang di dunia industri [2]. Pada kendaraan logistik yang diterapkan di dunia industri, akan dipasangkan suatu alat yang dapat mendeteksi posisi, kecepatan, serta dapat membuka / menutup pintu otomatis. Beberapa sistem kendaraan modern menerapkan format *Automatic Vehicle Location* (AVL), konsep ini diterapkan untuk menentukan lokasi geografis kendaraan, dan mentransmisikan informasi ke server [2]. Selama beroperasi petugas / dinas terkait dapat memonitoring posisi dan kecepatan bus melalui *dashboard* website, dengan demikian maka operasional bus Damri dapat dipantau setiap saat. Bagi masyarakat sebagai penumpang, dapat mengetahui posisi, dan penjadwalan bus

secara otomatis dan *real time*, sehingga meminimalisir waktu tunggu masyarakat di shelter. Secara umum, berikut struktur sistem bus pintar :



Gambar II-2 Struktur Bus Pintar, Dikutip dari [2]

Shelter dan bus dapat saling bertukar informasi, posisi bus dapat ditampilkan pada peta digital yang tersedia di *dashboard* website. Bus yang telah dipasangkan perangkat *tracker* untuk mendeteksi posisi dan kecepatan pergerakkan bus, data akan tersimpan pada server. Data dan informasi terbuka dan dapat diakses oleh masyarakat melalui *website*.



Gambar II-3 Vehicle Tracking System

2.3 Geofencing

Geofencing merupakan fitur yang umumnya hadir pada aplikasi *mobile* yang memanfaatkan kinerja *global positioning system* (GPS) atau *radio frequency*

identification (RFID) untuk memberikan batasan geografis virtual [3]. Secara umum, sistem ini melakukan pendeteksian lokasi terhadap objek bergerak seperti manusia, kendaraan, kontainer menggunakan GPS, memberikan notifikasi apabila terdapat objek yang memasuki atau keluar dari wilayah yang telah ditentukan sistem [8].

Elemen-elemen yang umum digunakan dalam sistem geofencing yaitu, GPS, A-GPS, Cell-ID (COO/CID), dan WiFi. Selain itu, juga dapat digunakan untuk memberikan notifikasi / pemberitahuan berdasarkan radius geografis yang disebut *Geo-notification*. *Geo-notfication* terbagi menjadi tiga jenis, yaitu :

- *Static Geo-notification* : fungsional berdasarkan posisi geografis dari *mobile user* terhadap area yang telah ditetapkan (*fixed area*)
- *Dynamic Geo-notification* : fungsional berdasarkan posisi geografis dari *mobile user* terhadap perpindahan dan perubahan data.
- *Peer to peer Geo-notification* : fungsional berdasarkan posisi geografis teman / partner terdekat.

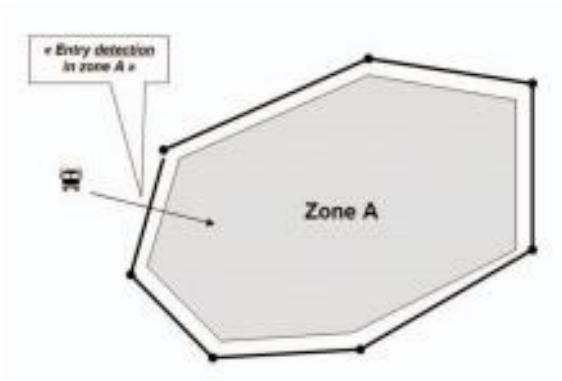
Dalam implementasinya, geofencing ini diterapkan menggunakan layanan peta digital Google, sudah disediakan *tools* yang digunakan untuk merepresentasikan sebuah area / fence dalam bentuk *circular*, *polygonal*. Layanan berbasis lokasi biasa disebut LBS (*location based service*) [13]. Metode LBS merupakan salah satu bagian dari implementasi mobile GIS yang lebih cenderung memberikan fungsi terapan sehari-hari seperti menampilkan direktori kota, navigasi kendaraan, pencarian alamat serta jejaring sosial dibanding fungsionalitas pada teknologi GIS populer untuk *Field Based* GIS [14].

Geofencing memberikan banyak manfaat dalam penerapannya pada sistem transportasi modern [8]. Berbagaimacam variasi sistem geofencing diciptakan untuk mendukung kinerja manusia. Dalam penelitian ini, penulis menerapkan teknik geofencing sebagai berikut :

2.3.1 Geofenced Area

Sistem geofencing ini memiliki tugas untuk melakukan pengawasan terhadap objek-objek yang bergerak memasukki area geofencing yang telah

ditentukan, luas area dapat ditentukan mulai dari minimal jarak 10 km atau lebih [8]. Untuk memudahkan dalam mengilustrasikan wilayah, maka dapat menggunakan *tools* polygon ditampilkan pada peta digital.

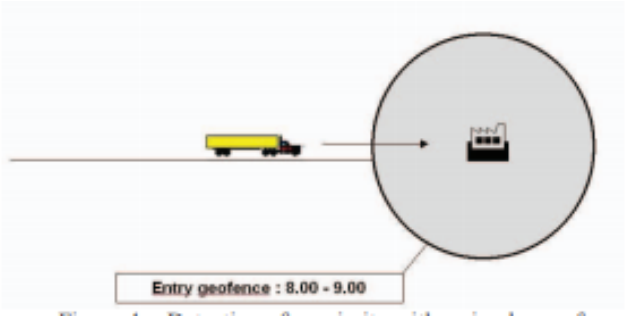


Gambar II-4 Geofenced Area

Sumber gambar : Referensi [8]

2.3.2 Proximity With a Point of Interest (POI)

Sistem geofencing ini mengukur jarak objek yang bergerak dengan lokasi titik POI (*point of interest*) [8]. Teknik geofencing ini merupakan yang paling sederhana, hanya membutuhkan dua buah lokasi GPS. Titik POI akan diletakkan di tengah area geofencing [8].

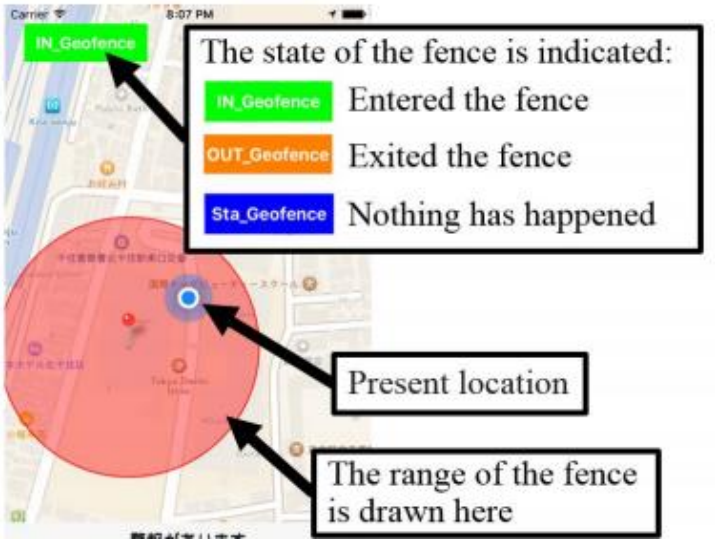


Gambar II-5 Proximity With a Point of Interest

Sumber gambar : Referensi [8]

Sistem Geofencing memiliki keterbatasan, khususnya dalam jumlah konsumsi daya batre perangkat. Karena sistem ini membutuhkan data lokasi, maka sistem akan seringkali meminta *request data* lokasi dari sensor / modul

GPS. Sehingga dengan meningkatnya *device performance*, berdampak menurunnya daya tahan batre perangkat. Selain itu, sistem geofencing juga membutuhkan persetujuan dan partisipasi *user* yang dapat memicu sistem *running* [9]. Gambar dibawah ini menunjukkan interaksi *user* terhadap sistem geofencing



Gambar II-6 User – Geofencing

Sumber gambar : Referensi [10]

Berdasarkan pada Gambara II-6, didapatkan tiga buah status kondisi geofencing, yitu IN, objek memasuki wilayah geofencing, OUT objek keluar dari wilayah geofencing, dan STA objek tidak berinteraksi dengan wilayah geofencing. Lingkaran merah menunjukkan radius wilayah geofencing.

Pada penelitian ini, geofencing diimplementasikan pada *website* sebagai subsistem utama, untuk membantu *user* menemukan bus yang memiliki radius jarak terdekat dengan shelter. Untuk menentukkan bus terdekat dengan shelter, maka dibutuhkan metode penghitungan dengan melibatkan dua titik lokasi (latitude-longitude). Metode penghitungan yang umum digunakan dalam sistem geofencing yaitu *Haversine Formula* .

2.3.3 Haversine Formula

Metode Haversine merupakan persamaan yang digunakan untuk menghitung jarak antara dua titik lokasi menggunakan *latitude-longitude* sebagai variabel *input*, serta merupakan metode yang umum digunakan dalam sistem navigasi [11]. Konsep perhitungan haversine menggunakan pendekatan bentuk bumi [12]. Data latitude merepresentasikan simbol *phi* yang merupakan sudut antara garis lurus dengan lokasi tertentu dan bidang ekuator. Longitude merepresentasikan *lambda*, merupkan sudut terhadap titik barat atau timur dari *prime meridian* dan Greenwich *meridian* [12].

Terdapat beberapa tahapan untuk menentukkan jarak radius menggunakan haversine, diantaranya :

1. Tentukkan lokasi kordinat dalam bentuk data desimal
2. Tentukkan data *latitude-longitude* (lintang dan bujur)
3. Tentukkan lokasi kordinat sebagai radian
4. Tentukkan garis *latitude* awal dan *longitude* awal
5. Lakukan perhitungan dengan menggunakan rumus haversine

Apabila diubah dalam bentuk persamaan, dengan konstanta radius bumi (R = 6367,45 km) dan sudut l = 0.0174532925 radians maka dapat dituliskan sebagai berikut :

- $\Delta lat = lat2 - lat1$
  $\Delta long = long2 - long1$
  $a = Sin2\left(\frac{\Delta lat}{2}\right) + \cos(lat1)\cos(lat2)\, sin2(\frac{\Delta long}{2})$
  $c = 2atan2(\sqrt{a}\,, \sqrt{l - a}\,)$
  $d = R\ x\ c$

Persamaan II-1 Haversine Formula

2.4 One Time Password

One time password (OTP) merupakan kode unik yang terdiri dari minimal 4 digit yang terdiri dari angka dan karakter, berupa kode acak yang di *generate*

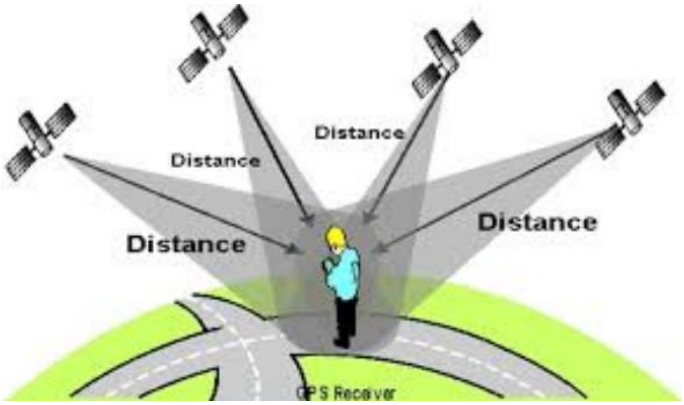
oleh sistem. Kode ini memiliki *expired time* (waktu berlaku). Seperti namannya, kode ini hanya dapat digunakan satu kali. Mekanisme otentikasi sperti ini umum digunakan untuk kegiatan transaksi dan validasi tiket. OTP diberikan kepada pengguna yang baru pertama kali menggunakan sebuah sistem, misalnya seperti masuk ke website *e-commerce*, sistem akan meminta kode OTP yang dikirimkan melalui SMS atau email.

Gambar II-7 OTP Authentication

Dalam penelitian ini, kode OTP akan digunakan untuk validasi tiket penumpang bus Damri, yang akan dikirimkan melalui email pengguna.

2.5 Global Positioning System (GPS)

Global positioning system (GPS) merupakan sebuah sistem satelit navigasi yang juga dapat menentukan posisi yang dikelola oleh Amerika Serikat. Sistem ini didesain untuk memberikan posisi dan kecepatan tiga dimensi, serta informasi mengenai waktu, secara kontinyu [5]. Saat ini hampir seluruh produk teknologi dan *gadget* memiliki kemampuan untuk mendeteksi posisi objek atau manusia. Prinsip penentuan posisi GPS, yaitu menggunakan metode reseksi jarak, dimana pengukuran dilakukan secara simultan ke beberapa satelit yang telah diketahui koordinatnya. Pada pengukuran GPS, setiap epoknya memiliki empat parameter. Yaitu : tiga parameter koordinat X, Y, Z, atau L, B, H, dan satu parameter kesalahan waktu yang diakibatkan ketidakselarasan jam osilator di satelit dengan jam di *receiver* GPS [5].



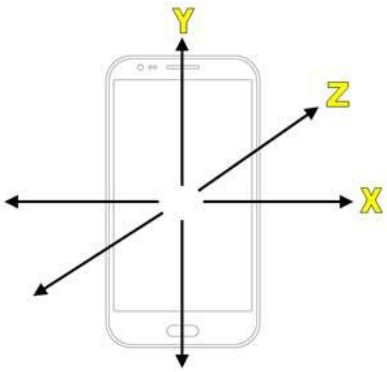
Gambar II-8 GPS Sattelite

Penggunaan GPS sudah dirancang dalam bentuk komponen *module* elektronik, beberapa pabrikan semikonduktor merancang GPS untuk kebutuhan pengembangan produk alat pintar berbasis posisi menggunakan GPS.

Gambar II-9 GPS Module

2.6 Accelerometer & Gyroscope

Accelerometer dan Gyroscope merupakan sensor-sensor yang dipasangkan umumnya pada *smartphone* untuk mendeteksi orientasi gerak (accelero) dan rotasi arah gerak benda (gyroscope). Dengan sensor accelerometer, benda / objek dapat mendeteksi pergerakan ke segala arah, mengukur percepatan bahwa benda mengalami perubahan yang relatif sesuai dengan tiga sumbu X, Y, Z atau kanan, kiri, atas, bawah, dan datar. *Smartphone* menggunakan fitur ini untuk mengetahui apakah *smartphone* dalam orientasi berdiri (*potrait*) atau memanjang (*landscape*).



Gambar II-10 Accelerometer

Gyroscope sensor yang digunakan untuk memberikan orientasi lebih presisi hingga putaran 360 derajat.

Gambar II-11 Gyroscope

Selain diimplementasikan pada *smartphone*, sensor accelero-gyro (gabungan antara accelerometer dan gyroscope) umum dipasangkan pada perancangan alat pintar, yang membedakan, sensor ini sudah dalam bentuk *module* elektronik.

Gambar II-12 Accelero-Gyro Module MPU 9250

Sensor dapat mendeteksi orientasi akselerasi dan rotasi hanya dengan satu komponen saja, contoh : Accelerometer Gyro Mpu9250. Pada penelitian ini, sensor accelero-gyro digunakan untuk mendeteksi pergerakkan bus Damri dan mengantisipasi terjadinya kecelakaan. Apabila pergerakkan bus anomali, maka alat akan mengirimkan notifikasi ke server.

2.7 UML

Unified Modeling Language (UML) adalah sebuah pemodelan umum untuk mengvisualisasikan struktur dari sebuah sistem perangkat lunak [6]. Pemodelan ini seringkali digunakan oleh kalangan pengmbang perangkat lunak yang tersaji dalam dokumentasi perancangan sistem. Berikut ini beberapa jenis diagram UML, yaitu :

- *Class Diagram*

  Memodelkan himpunan kelas-kelas, antarmuka, kolaborasi, serta relasi. Sering dijumpai pada pemodelan sistem berorientasi objek. [7]

Tabel II-1 Class Diagram

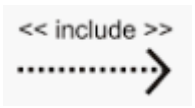
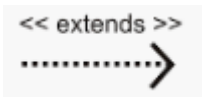
No	Simbol	Nama Komponen	Keterangan
1.	Nama *Class* + atribut + atribut + atribut + *method* + *method*	*Class*	*Class* adalah blok pembangun dari sistem, terdiri dari tiga, yaitu : nama kelas, atribut, dan metode
2.	1..n Owned by 1	*Association*	Merupakan sebuah relasi / hubungan umum antar kelas
3.		*Composition*	Simbol yang digunakan apabila sebuah kelas tidak dapat berdiri sendiri, dan membutuhan kelas

			lain, maka diperlukan relasi terhadap kelas induk.
4.		*Dependency*	Simbol yang digunakan untuk menggambarkan sebuah kelas yang menggunakan kelas lain
5.		*Aggregation*	Simbol yang digunakan untuk mengindikasikan keseluruhan bagian relasi.

- *Usecase Diagram*

  Memodelkan himpunan *usecase* dan aktor-aktor mengenai perilaku dari suatu sistem yang dibutuhkan serta diharapkan pengguna [7]

Tabel II-2 Usecase Diagram

No	**Simbol**	**Nama Komponen**	**Keterangan**
1.		*Actor*	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*
2.		*Include*	Menspesifikasikan bahwa *use case* sumber secara eksplisit
3.		*extends*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber
4.		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas

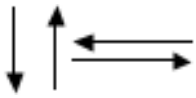
			
5.		*Usecase*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suati aktor

- *Activity Diagram*

  Diagram in merupakan jenis khusus dari diagram *state* yang memperlihatkan aliran dari suatu aktifitas ke aktifitas lainnya dalam suatu sistem. Diagram ini penting untuk memodelkan fungsi-fungsi dalam sistem [7].

Tabel II-3 Activity Diagram

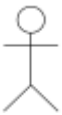
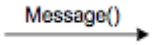
No	**Simbol**	**Nama Komponen**	**Keterangan**
1.		*Activity*	Memodelkan bagaimana masing-masing kelas antarmuka saling berinteraksi
2.		*Action*	*State* dari sistem yang mencerminkan eksekusi dari suatu aksi
3.		*Initial*	Titik awal, untuk memulai suatu aktivitas
4.		*Finish*	Titik akhir, untuk mengakhiri suatu aktivitas
5.		*Decision*	Simbol menggambarkan apabila terdapat lebih dari

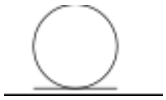
			satu aktivitas.
6.		*Life Connector*	Menghubungkan satu simbol dengan simbol lainnya

- *Sequence Diagram*

  Memodelkan interaksi yang menekankan kepada pengiriman pesan dalam suatu waktu tertentu [7].

Tabel II-4 Sequence Diagram

No	**Simbol**	**Nama Komponen**	**Keterangan**
1.		*Life line*	Objek entiti, antarmuka yang saling berinteraksi
2.		*Actor*	Digunakan untuk menggambarkan *user* / pengguna
3.		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi – informasi tentang aktifitas yang sedang terjadi
4.		*Boundary*	Digunakan untuk menggambarkan sebuah form
5.		*Control Class*	Digunakan untuk menghubungkan *boundary* dengan tabel

6.		*Entity Class*	Digunakan untuk menggambarkan hubungan kegiatan yang akan dilakukan