BAB II
LANDASAN TEORI

II.1 Berkemah

Berkemah berasal dari kata dasar kemah. Dalam Kamus Besar Bahasa Indonesia (KBBI., 2016), berkemah mempunyai arti: membuat (mendirikan) kemah (untuk bermalam dsb). Macam-macam tujuan berkemah, misalnya: berkemah karena hobi, mencari pengalaman dalam pendekatan diri kepada alam, meningkatkan kesadaran dalam menjaga lingkungan, mengembangkan tanggung jawab dalam pemeliharaan keseimbangan alam, membina kerjasama dalam persatuan dan persaudaraan, dan masih banyak tujuan lainnya. Berkemah sendiri populer sebagai aktivitas rekreasi pada awal abad ke-20 dan umumnya disertai dengan kegiatan lainnya, seperti mendaki gunung, outbond, menyalakan api unggun, berenang bahkan memancing. Beberapa orang menganggap berkemah sebagai kegiatan rekreasi yang dilakukan untuk beristirahat dari ramai dan penatnya aktivitas perkotaan untuk menikmati keindahan alam bebas karena dilakukan dengan menginap di lokasi perkemahan dengan menggunakan tenda atau bahkan di alam terbuka tanpa atap sama sekali.

II.1.1 Tujuan Berkemah

Tujuan berkemah adalah untuk memberikan pengalaman adanya saling ketergantungan antara unsur-unsur alam dan kebutuhan untuk melestarikannya, menjaga lingkungan dan mengembangkan sikap bertanggung jawab akan masa depan yang menghormati keseimbangan alam. Mengembangkan kemampuan diri mengatasi tantangan yang dihadapi, menyadari tidak ada sesuatu yang berlebih di dalam dirinya, menemukan kembali cara hidup yang menyenangkan dalam kesederhanaan. Membina kerjasama dan persatuan dan persaudaraan.

II.1.2 Jenis-Jenis Perkemahan

Jenis-Jenis perkemahan ditinjau dari beberapa faktor diantaranya adalah

a. Ditinjau dari Lamanya Waktu :
    1. Perkemahan Satu Hari (Persari).
    2. Perkemahan Sabtu Malam Minggu (Persami).
    3. Perkemahan lebih dari tiga hari.
b. Ditinjau dari Tempat Pelaksanaannya :
    1. Perkemahan Menetap
    2. Perkemahan Safari (Berpindah-pindah)
c. Ditinjau dari Tujuannya :
    1. Kemah Bakti. Seperti; Perkemahan Wirakarya (PW).
    2. Kemah Pelantikan. Seperti; Perkemahan Pelantikan Tamu Ambalan, Pelantikan Bantara dan lain-lain
    3. Kemah Lomba. Seperti; Olah Terampil, WRC.
    4. Kemah Rekreasi.
    5. Kemah Jambore.
    6. Kemah Riset/Penelitian
d. Ditinjau berdasarkan jumlah pesertanya :
    1. Perkemahan satu regu/sangga
    2. Perkemahan satu Pasukan/Ambalan/Racana
    3. Perkemahan tingkat : Ranting/Cabang/Daerah/Nasional/Regional/Dunia.

II.2 Definisi Perangkat Lunak

Menurut Roger S. Pressman perangkat lunak atau sering disebut software adalah sebuah instruksi yang apabila dijalankan menghasilkan fungsi dan hasil yang diinginkan. Perangkat lunak juga berarti struktur data yang dapat memanipulasi informasi. Deskripsi informasi dari ke dua poin ini menjelaskan operasi dan penggunaan dari perangkat lunak (Pressman, 2010).

Perangkat lunak lebih mengacu kepada *logical* dari pada *physical system element*. Oleh karena itu, perangkat lunak mempunyai karakteristik yang membedakan dengan perangkat keras yaitu :

1. Perangkat lunak dikembangkan atau direkayasa bukan diciptakan.
2. Perangkat lunak tidak akan habis atau hilang.
3. Meskipun industri bergerak ke arah komponen berbasis konstruksi tetapi sebagian besar perangkat lunak tetap dibangun atau dikembangkan.

II.3 *Object Oriented Analysis Desain*

Analisis dan desain berorientasi objek adalah cara baru dalam memikirkan satu masalah dengan menggunakan model yang dibuat menurut konsep sekitar dunia nyata. Tujuan dari analisis berorientasi objek adalah untuk mengembangkan model yang menggambarkan perangkat lunak komputer karena bekerja untuk memenuhi seperangkat persyaratan yang ditentukan *user* (Divayana, 2010). *Tools* yang dapat digunakan pada pendekatan analisis pengembangan sistem secara objek dapat menggunakan UML.

II.3.1 Pengertian OOP

OOP (Object Oriented Programming) adalah sebuah konsep pemograman yang berbasis objek, OOP merupakan terobosan baru dalam pemrograman setelah pemrograman prosedural, pemrograman modular dan pemrograman abstraksi data (Fowler, 2003). OOP memberikan kemudahan bagi programmer dalam pembuatan sebuah program, memberikan fleksibilitas yang lebih, kemudahan penggunaan program, dan dipakai luas dalam teknik piranti lunak skala besar. Selain itu dengan pemrograman OOP kode yang dibuat akan lebih mudah dikembangkan dan dirawat dalam arti seorang programer bisa menambah sebuah objek baru tanpa harus menggangu dan merubah modul yang lain. Berikut ini keuntungan dari penggunaan OOP dalam pemrograman (Fowler, 2003) :

1. *Reusability*

(Kemampuan untuk digunakan kembali). Kode yang dibuat dapat digunakan kembali di aplikasi atau programlainnya.

2. *Extensibility*

(Kemampuan untuk dapat dikembangkan). Kita dapat membuat method yang baru atau mengubah yangsudah ada sesuai dengan yang kita inginkan tanpa harus membuat kode dari awal.

3. *Maintainability*

(Kemampuan untuk mudah dikelola), kode yang sudah dibuat lebih mudah untuk di maintain/manage. Apabila aplikasi yang dibuat adalah aplikasi dengan skala besar dan apabila terjadi error atau kesalahan maka dengan OOP hal tersebut mudah ditangani karena pemograman yang menggunakan OOP sudah modularitas (memecahkan program yang besar menjadi bagian-bagian kecil) sehingga perbaikan cukup pada modul-modul tertentu saja tanpa harus mengubah modul yang lainnya.

Dalam penelitian ini OOP (*Object Oriented Programming)* digunakan untuk pembangunan program android.

II.3.2 Konsep OOP

Dalam pemrograman OOP, ada empat hal yang diperhatikan karena hal tersebut merupakan konsep vital dari OOP tersebut.

Berikut ini konsep utama tersebut (Fowler, 2003):

1. *Encapsulation*/pembungkusan, merupakan penyembunyian informasi melalui *private* dan *protected*
2. *Inheritance* /pewarisan, adalah kemampuan untuk membuat class turunan yang mewarisi *property* dan method dari kelas induk.3.
3. *Abtraction*/abtraksi, adalah sebuah konsep yang merupakan representasi abstrak dari konsep dalam pemograman,misalnya objek mobil merupakan representasi abstrak dari mobil dalam dunia nyata.
4. *Polymorphism* /banyak bentuk, seorang programer bisa memakai method dengan nama yang sama pada class-class yang berbeda.

II.3.3 *Unified Modeling Language* (UML)

Unified Modelling Language (UML) adalah sebuah bahasa yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menggunakan *class* dan *operation object* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa bahasa berorientasi objek (Hermawan, 2010). Dalam membangun block UML ada 3 hal

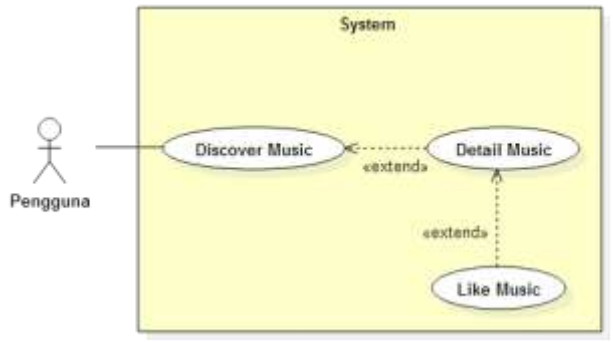
yang harus diperhatikan, yaitu *object* (memodelkan konsep), *relationship* (mengkoneksikan *object*), dan *diagram* (*grouping* yang saling mengkoneksikan antara *object* dan *relationship*. Diagram yang umum dipakai dalam analisis dan desain adalah:

1. *Use Case Diagram*

Use Case Diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *Use Case* merepresentasikan sebuah interaksi antara aktor dengan sistem. Sebuah *Use Case* dapat meng-*include* fungsionalitas *Use Case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *Use Case* yang di-*include* akan dipanggil setiap kali *Use Case* yang meng-*include* dieksekusi secara normal. Sebuah *Use Case* dapat di- include oleh lebih dari satu *Use Case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang serupa. Sebuah *Use Case* juga dapat meng-*extend Use Case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *Use Case* menunjukkan bahwa *Use Case* yang satu merupakan spesialisasi dari yang lain.

Dasar menentukan sebuah *Use Case* adalah *Use Case* merupakan sesuatu yang menyediakan beberapa hasil terukur kepada pengguna atau sistem eksternal. *Use Case* harus memiliki sangat jelas kriteria lulus / gagal. Pengembang, *tester*, penulis teknis, dan pengguna harus secara eksplisit tahu apakah sistem memenuhi kasus penggunaan atau tidak. Setiap bagian dari *Use Case* yang memenuhi tes sederhana ini mungkin menjadi kandidat yang baik untuk *Use Case* (Miles, 2006). Berikut adalah contoh dari *Use Case diagram* dapat dilihat pada Gambar II.1 Contoh *Use Case Diagram* dan simbol yang terdapat pada *Use Case diagram* dapat dilihat pada

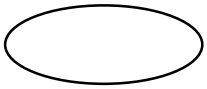
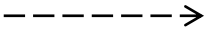
Tabel II.*1 Simbol Use Case* ***Diagram***.



Gambar II.1 Contoh *Use Case Diagram*

Simbol yang terdapat pada *Use Case diagram* dapat dilihat pada

Tabel II.1 Simbol *Use Case Diagram*

Simbol	Nama Simbol	Keterangan
	Actor	Menggambarkan pihak yang berhubungan dengan system baik itu merupakan pengguna atau sistem lainnya yang berada dari sistem yang sedang dibahas.
	*Use Case*	Simbol yang menggambarkan suatu kegiatan (aktifitas) yang terjadi pada system
	Asosiasi	Menghubungkan keterkaitan antara aktor dengan *Use Case*
	*Include*	Relasi *Use Case* dimana proses bersangkutan akan dilanjutkan ke proses yang dituju
	Extend	Relasi *Use Case* tambahan ke sebuah *Use Case* yang ditambahkan dapat berdiri sendiri walau tanpa *Use Case* tambahan itu
	Generalisasi	Sebuah relasi di mana fungsi yang satu adalah fungsi yang umum dari yang lainnya.

2. *Use Case Scenario*

Sebuah diagram yang menunjukkan *Use Case* dan aktor mungkin menjadi titik awal yang bagus, tetapi tidak memberikan *detail* yang cukup untuk desainer sistem untuk benar-benar memahami persis bagaimana sistem dapat terpenuhi. Cara terbaik untuk mengungkapkan informasi penting ini adalah dalam bentuk penggunaan *Use Case scenario* berbasis teks per *Use Case*nya. Berikut adalah dasar format penulisan *Use Case scenario* (Miles, 2006). Dasar pembangunan *Use Case scenario* dapat dilihat pada Tabel II.2 Dasar Pembangunan *Use Case Scenario*

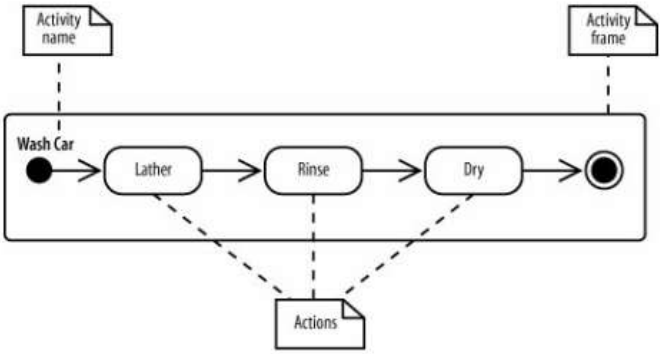
Tabel II.2 Dasar Pembangunan *Use Case Scenario*

***Use Case* Name**	Berisi nama dari *Use Case* yang akan digunakan	
Goal In Context	Menjelaskan apa yang aktor coba untuk dapatkan dari *Use Case*	
Description	Menjelaskan gambaran dari *Use Case*	
Related *Use Case*	Daftar *Use Case* yang berhubungan dengan *Use Case* tersebut	
Successful End Condition	Kondisi *Use Case* jika berhasil	
Failed End Condition	Kondisi *Use Case* jika gagal	
Actors	Daftar aktor yang dapat mengakses *Use Case*	
Trigger	Aktifitas yang dilakukan untuk mengawali *Use Case*	
Main Flow	Step	Action
	1	Deskripsi urutan aksi dari aktifitas *Use Case*
	2	
	3	
Extension	Step	Branching Action
	2.1	Deskripsi urutan aksi lain selain urutan aksi utama
	2.2	

3. *Activity Diagram*

Activity diagram menggambarkan berbagai alur aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing aliran berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Sebuah aktivitas dapat direalisasikan oleh satu *Use Case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *Use Case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas. Standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan *behaviour* pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis *horizontal* atau *vertikal*. *Activity diagram* dapat dibagi menjadi beberapa object *swimlane* untuk menggambarkan objek mana yang

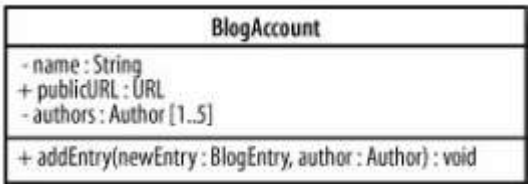
bertanggung jawab untuk aktivitas tertentu (Hermawan, 2010). Contoh dari *activity diagram* dapat dilihat pada Gambar II.2 Contoh *Activity Diagram*



Gambar II.2 Contoh *Activity Diagram*

4. *Class Diagram*

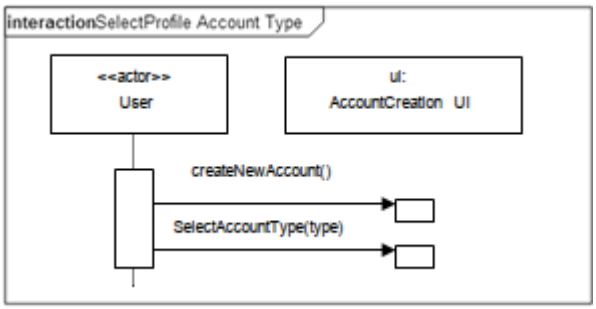
Class diagram merupakan diagram yang selalu ada di permodelan sistem berorientasi objek. Class diagram menunjukkan hubungan antar class dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai suatu tujuan. Kelas pada kelas diagram terdiri dari 3 bagian utama yaitu nama kelas, isi properti dari kelas beserta metode yang ada pada kelas tersebut. Kelas juga memiliki jenis-jenis hubungan seperti asosiatif, dependensi, agregasi, komposisi, spesifikasi dan generalisasi. Hubungan ini digunakan untuk menggambarkan bagaimana hubungan dan interaksi yang terjadi antar kelas. Masing-masing komponen penyusun kelas memiliki hak akses seperti public, private, dan protected.



Gambar II.3 *Class Diagram*

5. *Sequence Diagram*

Sequence diagram menjelaskan secara detil urutan proses yang dilakukan dalam sistem untuk mencapai tujuan dari *Use Case*: interaksi yang terjadi antar class, operasi apa saja yang terlibat, urutan antar operasi, dan informasi yang diperlukan oleh masing-masing operasi.



Gambar II.4 *Sequence Diagram*

Dalam penelitian ini *Unified Modeling Language* (UML) digunakan untuk menggambarkan alur kerja sistem, batasan sistem, dan fungsi-fungsi sistem.

II.4 *Object Relational Mapping*

Object Relational Mapping (ORM) adalah salah satu teknik untuk memetakan basis data relasional ke model objek. Pemetaan dapat dikategorikan menjadi 4, yaitu (Fowler, 2003):

1. *Basic mapping*
2. *Relationship mapping*
3. *Composition mapping*
4. *Inheritance mapping*

Dalam penelitian ini ORM digunakan untuk mempermudah antara koneksi database dengan bahasa pemrograman java.

II.5 *Prototype*

Prototype adalah sebuah model yang mula mula (model asli) yang menjadi sebuah contoh baku, untuk digunakan sebagai contoh untuk dipelajari ataupun dikembangkan. (KBBI, 2016)

Prototype adalah sebuah sistem atau alat yang akan dibuat dalam penelitian ini.

II.6 *Prototyping*

Prototyping adalah sebuah metode pembangunan sebuah sistem maupun perangkat atau sebuah siklus hidup menggunakan *protoyping (life cycle using prototyping)* adalah salah satu metode siklus hidup sistem yang didasarkan pada konsep model bekerja (*working model*). Tujuannya adalah mengembangkan model

menjadi sistem final. Artinya sistem akan dikembangkan lebih cepat daripada metode tradisional dan biayanya menjadi lebih rendah. Ada banyak cara untuk memprotoyping, begitu pula dengan penggunaannya. Ciri khas dari metodologi adalah pengembang sistem *(system developer)*, klien, dan pengguna dapat melihat dan melakukan eksperimen dengan bagian dari sistem komputer dari sejak awal proses pengembangan. (KBBI., 2016) *Prototyping* digunakan sebagai metode pembangunan sistem di penelitian ini.

II.7 Bahasa Pemrograman yang digunakan

Adapun beberapa bahasa pemrograman yang digunakan pada pembangunan perangkat lunak ini adalah sebagai berikut :

II.7.1 JAVA

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan bagian dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam p-code (bytecode) dan dapat dijalankan pada berbagai Mesin Virtual Java (JVM). Java merupakan Bahasa pemrograman yang bersifat umum/non-spesifik (general purpose), dan secara khusus didisain untuk memanfaatkan dependensi implementasi seminimal mungkin (Team).

Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform sistem operasi yang berbeda, java dikenal pula dengan slogannya, "Tulis sekali, jalankan di mana pun". Saat ini java Merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web.

Dalam pembangunan perangkat lunak ini bahasa pemrograman java digunakan untuk membangun APK android.

II.7.2 PHP

PHP adalah salah satu bahasa Server-side yang didesain khusus untuk aplikasi web. PHP dapat disisipkan diantara bahasa HTML dan karena bahasa *Server side*, maka bahasa PHP akan dieksekusi di server, sehingga dikirimkan ke browser adalah "hasil jadi" dalam bentuk HTML, dan kode PHP anda tidak akan terlihat (Team).

PHP termasuk dalam *Open Source Product*. Jadi anda dapat merubah source code dan mendistribusikan secara bebas. Adapun bebeapa kelebihan dari PHP yaitu:

1. PHP mudah dibuat dan kecepatan akses tinggi.
2. PHP dapat berjalan dalam web server yang berada dan dalam system operasi yang berbeda pula. PHP dapat berjalan disistem operasi UNIX, Windows98, Windows NT dan Macintosh.
3. PHP diterbitkan secara gratis.
4. PHP juga dapat berjalan pada web server Microsoft Personal Web Server, Apache, IIS, Xitami dan sebagainya.
5. PHP adalah termasuk bahasa yang embeddeb (bisa ditempel atau diletakkan dalam tag HTML).
6. PHP termasuk server-side programming.

Dalam pembangunan perangkat lunak ini PHP bergfungsi sebagai bahasa pemrogaman yang digunakan dalam membangun sebuah web services.

II.7.3 MySQL

Dalam bahasa SQL data akan diatur dalam baris dan kolom pada table, karena pada database di dalamnya pasti akan anda temukan beberapa table. Ada banyak server database yang bisa digunakan, namun salah satu server database yang popular dan banyak digunakan orang adalah MySQL. Selain karena bisa didapat dengan mudah dan gratis, MySQL juga dapat berjalan pada sistem operasi apa pun (Corp) (Bachtiar, 2012).

Berikut beberapa kelebihan yang dimiliki MySQL :

1. Program database MySQL memang merupakan pasangan setia yang sering diintegrasikan dengan Apache.
2. MySQL memiliki kemampuan server multithread sehingga dapat menangani banyak permintaan layanan secara bersamaan.
3. Dapat berjalan pada banyak sistem operasi (Linux, Windows, SunOS, MacOS).
4. Memiliki metode enkripsi yang baik.
5. Mampu menangani data yang besar dan banyak.
6. Menggunakan autentifikasi pengguna (*user*) dan kata sandi (password) sehingga menjadikannya lebih aman.

Dalam perangkat lunak yang akan dibangun ini MySQL digunakan sebagai tempat penyimpanan data, *user*, admin, alat-alat kemah, dan lokasi perkemahan.

II.8 Android

Android adalah sistem operasi berbasis Linux yang dirancang untuk perangkat seluler layar sentuh seperti telepon pintar dan komputer tablet. Android awalnya dikembangkan oleh Android, Inc., dengan dukungan finansial dari Google, yang kemudian membelinya pada tahun 2005. Sistem operasi ini dirilis secara resmi pada tahun 2007, bersamaan dengan didirikannya Open Handset Alliance, konsorsium dari perusahaan-perusahaan perangkat keras, perangkat lunak, dan telekomunikasi yang bertujuan untuk memajukan standar terbuka perangkat seluler (Google Inc., 2014).

Dalam pembangunan perangkat lunak ini android berfungsi sebagai *tools* antara pengguna dan perangkat (*device*).

II.8.1 Versi Android

Adapun versi-versi android yang sudah dirilis adalah sebagai berikut :

1. Android 1.1

Pada 9 Maret 2009, Google merilis Android versi 1.1. Android versi ini dilengkapi dengan pembaruan estetis pada aplikasi, jam alarm, voice search (pencarian suara), pengiriman pesan dengan Gmail, dan pemberitahuan email.

2. Android 1.5 (Cupcake)

Pada pertengahan Mei 2009, Google kembali merilis telepon seluler dengan menggunakan Android dan SDK (*Software Development Kit*) dengan versi 1.5 (*Cupcake*). Terdapat beberapa pembaruan termasuk juga penambahan beberapa fitur dalam seluler versi ini yakni kemampuan merekam dan menonton video dengan modus kamera, mengunggah video ke Youtube dan gambar ke Picasa langsung dari telepon, *Bluetooth* A2DP *support*, kemampuan terhubung secara otomatis ke *headset Bluetooth*, animasi layar, dan *keyboard* pada layar yang dapat disesuaikan dengan sistem.

3. Android 1.6 (Donut)

Donut (versi 1.6) dirilis pada September dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai indikator dan kontrol applet VPN. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus, kamera, camcorder dan galeri yang dintegrasikan, *Gestures*, dan *Text-to-speech engine*, kemampuan dial kontak, teknologi *text to change speech* dan juga pada pengetikan kata yang lebih mudah.

4. Android 2.0/2.1(Eclair)

Pada 3 Desember 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1 (Eclair). Perubahan yang dilakukan adalah pengoptimalan *hardware*, peningkatan Google Maps 3.1.2, perubahan UI dengan browser baru dan dukungan HTML5, daftar kontak yang baru, dukungan flash untuk kamera 3,2 MP, digital *Zoom*, dan *Bluetooth* 2.1. Dengan semakin berkembangnya dan semakin bertambahnya jumlah handset Android, semakin banyak pihak ketiga yang

berminat untuk menyalurkan aplikasi mereka kepada sistem operasi Android. Aplikasi terkenal yang diubah ke dalam sistem operasi Android adalah Shazam, *Backgrounds*, dan *WeatherBug*. Sistem operasi Android dalam situs Internet juga dianggap penting untuk menciptakan aplikasi Android asli, contohnya oleh MySpace dan Facebook.

5. Android 2.2 (froyo : *Frozen Yoghurt*)

Pada 20 Mei 2010, Android versi 2.2 (Froyo) diluncurkan. Perubahan-perubahan umumnya terhadap versi-versi sebelumnya antara lain dukungan Adobe Flash 10.1, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, intergrasi V8 JavaScript engine yang dipakai Google Chrome yang mempercepat kemampuan rendering pada browser, pemasangan aplikasi dalam SD Card,

kemampuan WiFi *Hotspot portable*, dan kemampuan *auto update* dalam aplikasi Android Market.

6. Android 2.3 (*Gingerbread*)

Pada 6 Desember 2010, Android versi 2.3 (*Gingerbread*) diluncurkan. Perubahan-perubahan umum yang didapat dari Android versi ini antara lain peningkatan kemampuan permainan (*gaming*), peningkatan fungsi *copy paste*, layar antar muka (*User Interface*) didesain ulang, dukungan format video VP8 dan WebM, efek audio baru (*reverb, equalization, headphone virtualization*, dan *bass boost*), dukungan kemampuan *Near Field Communication* (NFC), dan dukungan jumlah kamera yang lebih dari satu.

7. Android 3.0 (*Honeycomb*)

Android *Honeycomb* dirancang khusus untuk tablet. Android versi ini mendukung ukuran layar yang lebih besar. *User Interface* pada *Honeycomb* juga berbeda karena sudah didesain untuk tablet. *Honeycomb* juga mendukung multi prosesor dan juga akselerasi perangkat keras (*hardware*) untuk grafis. Tablet pertama yang dibuat dengan menjalankan *Honeycomb* adalah Motorola Xoom.

8. Android 4.0 (*IceCreamSandwich*/ICS)

Android pertama yang mempunyai fitur baru membuka kunci dengan pengenalan wajah. Fitur ini belum dimiliki oleh Android versi dibawah *Ice Cream Sandwich tetapi* sudah disempurnakan pada versi diatasnya. Selain itu *Ice Cream Sandwitch* juga mempunyai penampilan *Interface* yang bersih dan smooth. *Ice Cream Sandwitch* juga mempunyai font yang bernama Roboto. Jika digunakan font ini terlihat sangat simpel namun elegan, tampilan yang diciptakan oleh font Roboto sungguh terkesan unik.

9. Android 4.1 (*JellyBean*)

Jelly Bean-Android versi 4.1 yang diluncurkan pada acara Google I/O membawa fitur fitur baru yang menawan, beberapa fitur yang diperbaharui dalam sistem operasi ini antara lain, pencarian dengan menggunakan *Voice Search* yang lebih cepat, informasi cuaca, lalu lintas, hasil pertandingan olahraga yang cepat dan tepat, selain itu versi 4.1 ini juga mempunyai fitur *keyboard virtual* yang lebih baik. Permasalahan umum yang sering ditemui pengguna Android adalah baterai, namun Baterai dalam sistem Android *Jelly Bean* versi 4.1 ini diklaim cukup hemat.

10. Android 4.4 (Kitkat)

Android 4,4 KitKat sendiri merupakan versi terbaru dari seri sebelumnya yaitu 4,3 Jelly Bean dan sudah diperkenalkan pada umum pada bulan September 2013. Android KitKat merupakan penyempurnaan dari versi sebelumnya.

1. Penyempurnaan yaitu mampu mengoptimasi memori dan memperbaiki respon layar sentuh agar lebih cepat dan akurat di bandingkan sebelumnya.
2. Fitur multi-tasking dimana dapat membuka 2 aplikasi secara bersamaan.
3. Pada aplikasi baru secara otomatis akan memprioritaskan nomor kontak yang sering dihubungi. Fitur pencarian berdasarkan lokasi terdekat, bisnis, kontak dan orang-orang pada aplikasi Google.
4. Pencocokan daftar nomor yang tidak ada dengan daftar nomor di peta Google secara otomatis.
5. Tampilan karakter Bahasa jepang lebih nyaman terlihat dalam layar *handphone*.

11. Android 5.0 (Lollipop)

Android 5.0 Lollipop merupakan salah satu versi sistem operasi android terbaru yang merupakan hasil upgrade dari OS Android 4.4 KitKat.Pada versi OS Android terbaru ini juga dibekalkan visual yang sangat signifikan dibandingkan pada saat Android versi 4.0 ICS (*ice cream sandwich*).Android versi terbaru OS L atau Lollipop tersebut juga dibekalkan kemampuan integrasi antar perangkat seperti *smartphone* , tablet, smartwatch berbasis Android. Android 5.0 Lollipop dibekalkan pertama pada *smartphone*.

II.8.2 Fitur Android

Android memiliki beberapa fitur utama yang sering digunakan dalam proses pembangunan aplikasi diantaranya adalah:

1. *Multi-proses dan App Widgets*

Sistem operasi android tidak melarang prosesor menjalankan lebih dari satu aplikasi dalam satu waktu. Sistem operasi android dapat mengatur aplikasi dan thread yang berjalan secara *multitasking*. Keuntungan yang didapat adalah ketika aplikasi berjalan dan berinteraksi dengan pengguna di layer depan sistem operasi, proses dari aplikasi lain dapat berjalan untuk melakukan pembaruan informasi. Sebagai contoh misalnya ketika pengguna memainkan game, proses lain dapat berjalan di belakang aplikasi seperti memeriksa harga saham dan memunculkan peringatan (Google Inc., 2014).

App Widgets adalah mini aplikasi yang dapat *embedded* dalam aplikasi seperti home screen. App widgets dapat menjalankan proses request seperti musik streaming atau mendeteksi suhu ruangan secara *background*.

Multi-proses dapat memberikan manfaat berupa *user experience* yang lebih banyak, namun penggunaan fitur tersebut dapat menghabiskan banyak energi baterai jika penggunaan tidak benar (Schwarz, et al., 2013).

2. *Touch Gestures dan Multi-touch*

Touchscreen adalah *user interface* intuitif yang digunakan banyak *smartphone* di dunia. Dengan fitur ini interaksi dapat dibuat lebih mudah karena cukup dengan menggunakan jari tangan. *Multi-touch* adalah kemampuan yang dapat melakukan tracking lebih dari satu tangan dalam satu waktu, Fitur ini sering digunakan untuk

interaksi memperbesar atau memutar objek. Selain itu pengembang dapat membuat interaksi baru dengan memanfaatkan fitur tersebut.

3. *Hard dan Soft Keyboard*

Salah satu fitur pada perangkat smartphone adalah tombol fisik dan non fisik, tombol fisik digunakan untuk navigasi pendukung dalam pengoperasian android.. Pengembang aplikasi tidak perlu secara manual untuk mengintegrasikan tombol tersebut dalam aplikasi. Tombol non fisik adalah tombol yang dibuat oleh sistem operasi seperti keyboard virtual, dan tombol navigasi aplikasi.

II.9 Web Services

W3C mendefinisikan web service sebagai sebuah software aplikasi yang dapat teridentifikasi oleh URI dan memiliki *interface* yang didefinisikan, dideskripsikan, dan dimengerti oleh XML atau JSON dan juga mendukung interaksi langsung dengan software aplikasi yang lain dengan menggunakan message berbasis XML atau JSON melalui protokol internet Web service adalah sebuah software aplikasi yang tidak terpengaruh oleh platform, menyediakan method-method yang dapat diakses oleh network. Web Service juga akan menggunakan XML untuk pertukaran data, khususnya pada dua entities bisnis yang berbeda. Beberapa karakteristik dari web service adalah :

1. *Message-based*
2. *Standards-based*
3. *Programming language independent*
4. *Platform-neutral*

Beberapa key standard di dalam web service adalah: JSON, XML, SOAP, WSDL and UDDI (Avestro, 2007). Dalam penelitian ini web services ini berfungsi sebagai tempat penyimpanan data dalam format JSON

II.10 JSON

JSON (JavaScript Object Notation) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (generate) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman JavaScript, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON

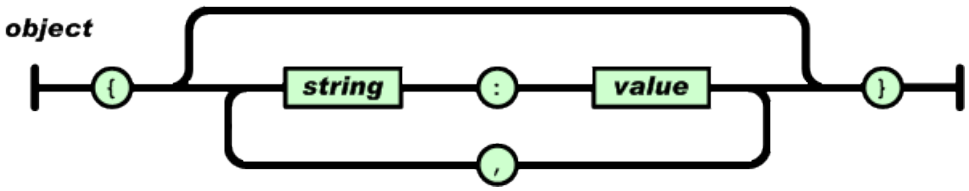
merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data. JSON terbuat dari dua struktur (JSON.Org):

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel hash (*hash table*), daftar berkunci (*key list*), atau associative array.
2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (array), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemprograman modern mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini. JSON menggunakan bentuk sebagai berikut:

1. Objek

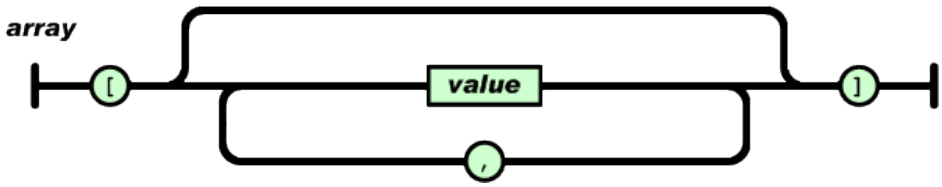
Objek adalah sepasang nama / nilai yang tidak terurutkan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma). Objek biasanya digunakan untuk menyimpan data tunggal dalam bentuk JSON.



Gambar II.5 Objek JSON

2. Larik

Larik adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma). Larik dalam JSON dapat digunakan sebagai value dari JSON object hal ini dapat berguna jika JSON menyimpan data bertingkat.



Gambar II.6 Array JSON

Bentuk data JSON objek dan larik dapat saling dikombinasikan untuk mendukung struktur data yang lebih kompleks. JSON mendukung beberapa tipe data untuk menjadi value seperti Angka, String, Bollean dan nilai NULL.

Dalam pembangungan perangkat lunak ini JSON digunakan sebagai format pertukaran data antara *server* dan perangkat android.

II.11 GPS

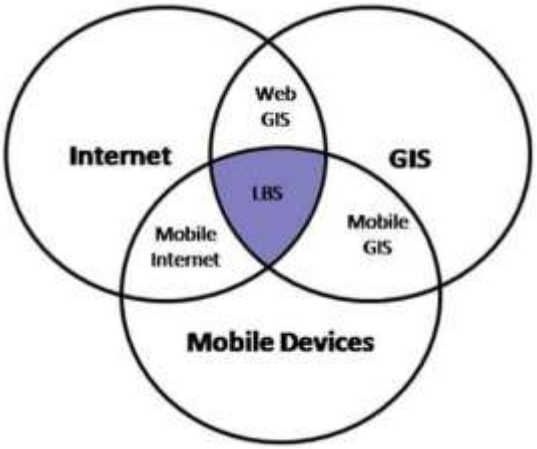
Global Positioning System (GPS) merupakan sistem navigasi yang menggunakan satelit dalam penggunaannya. GPS terdiri dari tiga bagian yaitu sistem kontrol, satelit dan pengguna. Sistem kontrol adalah bagian yang mengontrol pergerakan satelit – satelit yang ada dan saling berinteraksi satu sama lain kemudian pengguna adalah alat navigasi yang digunakan seperti perangkat *mobile* yang kini sudah memiliki fitur GPS di dalamnya (Tanoe, 2011).

GPS dalam pembangunan perangka lunak ini digunakan untuk mendapatkan lokasi pengguna.

II.12 *Location Based Service*

Location Based Service (LBS) atau layanan berbasis lokasi adalah layanan informasi yang dapat diakses melalui *mobile device* dengan menggunakan *mobile* network, yang dilengkapi dengan kemampuan untuk memanfaatkan lokasi dari

mobile device tersebut. LBS memungkinkan untuk melakukan komunikasi dan interaksi dua arah. LBS ini dapat digambarkan sebagai suatu layanan yang berada pada pertemuan tiga teknologi yaitu *Geographic Information System* (GIS), *Internet Service* dan *mobile device*, seperti pada Gambar II.7 Teknologi *Location Based Service*:



Gambar II.7 Teknologi *Location Based Service*

Dalam penelitian ini LBS digunakan untuk melakukan pencarian lokasi kemah dan tempat penyewaan alat kemah terdekat dengan pengguna.

II.12.1 Metode Penentu Lokasi

LBS merupakan layanan yang mampu bereaksi aktif pada suatu perubahan posisi sehingga dapat mendeteksi letak objek dan memberikan layanan sesuai dengan letak objek yang telah diketahui tersebut. Pada teknologi LBS berbasis jaringan seluler, penentuan posisi pada sebuah peralatan komunikasi ditentukan berdasarkan posisi relatif terhadap lokasi BTS (*Base Transceiver Station*). Dalam menentukan posisi dari sebuah perangkat *mobile* yang sedang aktif, secara umum terdapat tiga tingkat metode yang digunakan saat ini, di antaranya adalah sebagai berikut (Tanoe, 2011):

1. Metode *Basic Positioning* berbasis pada *Cell Identification* (Cell ID)

Penentuan posisi didasarkan pada daerah geografis yang tercakup oleh sebuah Cell berhubungan dengan daerah cakupan dari sinyal radio. Ketika sebuah

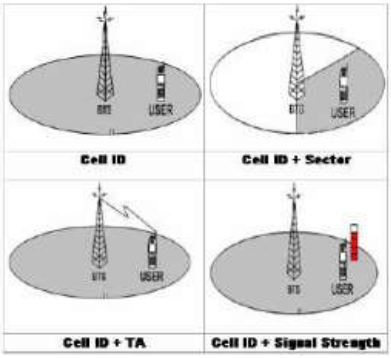
handphone terhubung secara aktif dengan sebuah base station, berarti handphone tersebut diasumsikan berada dalam cell dari base stadion tersebut. Untuk mengukur jarak dan arah handset dari base station tidak dapat diketahi dengan pasti, oleh karena itu untuk lebih meningkatkan lagi akurasi hasil pencarian pada metode ini seringkali dikombinasikan dengan metode lain seperti (Tanoe, 2011):

a. *Timing Advance* (TA)

   Dengan menggunakan TA ini metode Cell ID akan ditambahkan sebuah fungsionalitas untuk menghitung *Round Trip Time* (RTT), yaitu waktu transmisi sebuah frame (dari base station ke handphone) dan waktu sebuah *frame* (dari handphone ke station). Dengan tambahan ini jarak antara handphone dan base station dapat ditentukan dengan keakuratan 50 m.

b. *Network Measurement Report (NMR)*

   Berdasarkan pada besar kecilnya sinyal (Receiver Signal Strength) yang diterima handphone yang ada di suatu "sector cell", maka posisi itu dapat ditentukan dengan lebih akurat.

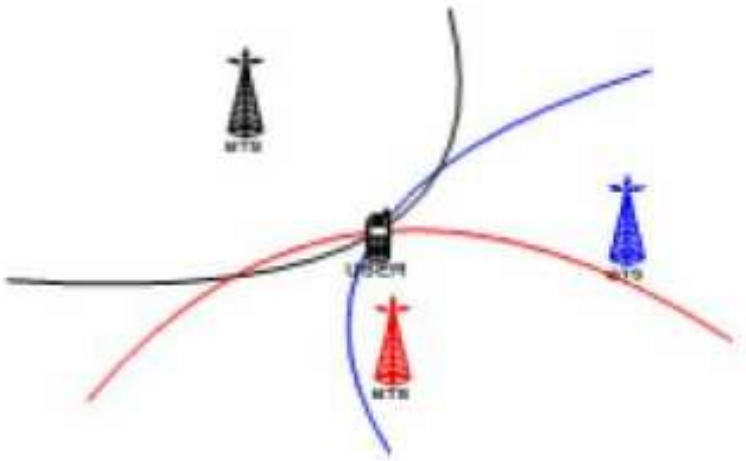


Gambar II.8 ***Network Measurement Report***

2. *Metode Enhanced Positioning*

Metode ini menggunakan pendekatan *Observe Time Difference* (OTD) dengan jaringan GSM yang sering digunakan seperti Enhanced-OTD (EOTD). E-OTD adalah metode pencarian posisi yang berdasarkan pada waktu. Untuk menentukan posisi relative, sebuah handphone harus aktif terhadap tiga base station dan perlu ditentukan terlebih dahulu jarak handphone terhadap masing- masing base station

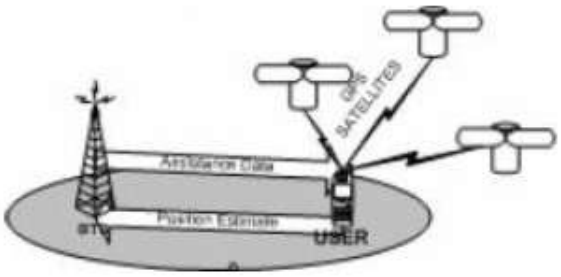
berdasarkan waktu yang ditempuh oleh sebuah sinyal dari handphone ke masing – masing *base station*. Untuk menentukan posisi dari handphone yang sedang aktif digunakan rumus matematika untuk triangulasi. Dengan metode ini akurasinya akan meningkat hingga ketilitian kurang dari 50 m (Tanoe, 2011).



Gambar II.9 *Enhaced Positioning*

3. Metode *Advance Positioning*

Metode ini menggunakan teknologi *Assisted-Global Positioning System* (AGPS). A-GPS ini merupakan metode yang berbasis pada waktu. Pada metode ini, akan dilakukan pengukuran waktu tiba dari sebuah sinyal yang dikirim dari tiga buah satelit GPS. Hal ini berarti handset harus memiliki fasilitas untuk mengakses GPS. A-GPS juga menghasilkan akurasi secara vertical dan estimasi jarak yang baik. Akurasinya pun kurang dari 10 m (Tanoe, 2011).



Gambar II.10 *Advance Positioning*

II.12.2 Komponen *Location Based Service*

Dalam menggunakan *Location Based Service* atau layanan berbasis lokasi terdapat beberapa elemen yang diperlukan, antara lain (Tanoe, 2011) :

1. *Mobile Devices*

   Sebuah alat yang digunakan untuk meminta informasi yang dibutuhkan seperti perangkat *mobile smartphone* yang memiliki fasilitas navigasi.
2. *Communication Network*

   Jaringan selular yang mengirimkan data pengguna dan permintaan layanan.
3. *Positioning Component*

   Untuk dapat mengolah layanan harus menentukan lokasi pengguna. Posisi pengguna ini dapat diperoleh menggunakan jaringan komunikasi atau dengan menggunakan *Global Positioning System* (GPS).
4. *Service and Application Provider*

   Penyedia layanan pengguna selular yang bertanggung jawab untuk memproses suatu layanan.
5. *Data and Content Provider*

   Penyedia layanan informasi data yang dapat diminta oleh pengguna

II.13 *Latitude* (Garis Lintang)

Latitude adalah garis yang melintang di antara kutub utara dan kutub selatan yang menghubungkan sisi bagian bumi dari timur dan barat. Garis lintang memiliki posisi membentangi bumi seperti halnya garis khatulistiwa tetapi dengan kondisi nilai tertentu. *Latitude* dibedakan menjadi dua wilayah yaitu lintang utara dan selatan dimana nilai koordinat di bagian utara selalu positif dan koordinat selatan selalu negatif (Tanoe, 2011).

II.14 *Longitude* (Garis Bujur)

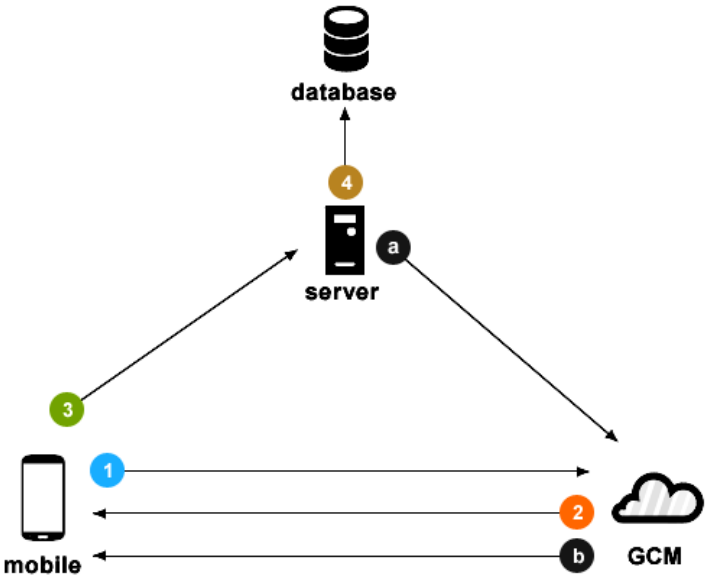
Longitude adalah garis bujur yang menghubungkan antara sisi utara dan sisi selatan permukaan bumi. Garis bujur digunakan untuk mengukur sisi barat dan timur koordinat suatu titik di belahan bumi. Longitude dibedakan menjadi dua wilayah yaitu bujur timur dan bujur barat dimana nilai koordinat yang berada di timur selalu negatif dan nilai koordinat dibagian barat selalu positif (Tanoe, 2011).

II.15 *Google Maps*

Google Maps adalah sebuah layanan gratis dari Google tentang berbagai macam teknologi pemetaan. Google Maps menampilkan peta dunia secara digital sehingga lebih memudahkan ketika ingin mengetahui suatu lokasi tertentu. Google Maps menyediakan beberapa keunggulan seperti peta jalan, menunjukan rute untuk suatu perjalanan yang diinginkan baik itu melalui jalur mobil sampai pejalan kaki. Dalam penelitian ini google maps digunakan rute (Tanoe, 2011).

II.16 *Google Cloud Messaging*

Google Cloud Messaging adalah teknologi untuk membantu pengembang mengirim data dari server menuju device android secara langsung melalui server google, pemanfaatan teknologi ini biasanya digunakan untuk *push messaging.* Pengiriman pesan menggunakan teknologi ini mengharuskan adanya API key, key ini terkait langsung dengan akun google yang digunakan untuk mempublikasikan aplikasi melalui google play (Google Inc., 2014).



Gambar II.11 GCM Model

Berdasarkan ilustrasi dari Gambar II.12 GCM Model terdapat dua server yang terlibat dalam proses *push messaging* , pada setiap server yang terlibat terdapat proses pengiriman data yang berbeda-beda diantaranya.

1. *Device* android mengirimkan sender id dan *application* id menuju GCM Server untuk registrasi.
2. Setelah berhasil registrasi, GCM server akan mengirimkan registration id menuju *device* android.
3. Setelah *device* android mendapatkan registration id, *device* tersebut akan mengirimkan registration id menuju server aplikasi.
4. Server aplikasi akan menyimpan registrasi id yang dikirimkan untuk digunakan sebagai id saat menggunakan *push messaging*.

II.17 Metode Haversine

Rumus Haversine adalah persamaan pada navigasi, yang memberikan jarak lingkaran besar antara dua titik bumi berdasarkan garis bujur (longitude) dan garis lintang (latitude). Perhitungan dari rumus ini juga cukup akurat dimana rumus ini mengabaikan ketinggian bukit dan kedalaman lembah di permukaan bumi. Secara matematis dapat dituliskan didalam persamaan sebagai berikut pada Gambar II.12 Rumus Haversine (Hidayat, 2014):

$$d = 2r \arcsin\left(\sqrt{\sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + \cos(\phi_1)\cos(\phi_2)\sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)$$

Gambar II.12 Rumus Haversine

II.18 Pengujian *Black Box*

Metode pengujian *black box* fokus pada keperluan penelusuran kesalahan fungsional dari *software*. Ujicoba *black box* berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya :

1. Fungsi-fungsi yang salah atau hilang
2. Kesalahan *interface*
3. Kesalahan dalam struktur data atau akses database eksternal
4. Kesalahan performa
5. kesalahan inisialisasi dan terminasi

Teknik pengujian black box terdiri dari 10 jenis diantaranya *Equivalence Partitioning, Boundary Value Analysis/Limit Testing, Comparison Testing, Sample Testing, Robustness Testing, Behavior Testing, Requirement Testing, Performance Testing, Endurance Testing, Cause-Effect Relationship Testing.* Salah satunya yang akan digunakan adalah *Equivalence partitioning*.

Equivalence partioning merupakan metode ujicoba *blackbox* yang membagi domain *input* dari program menjadi beberapa kelas data dari kasus ujicoba yang dihasilkan. Kasus uji penanganan single yang ideal menemukan sejumlah kesalahan (misalnya : kesalahan pemrosesan dari seluruh data karakter) yang merupakan syarat lain dari suatu kasus yang dieksekusi sebelum kesalahan umum diamati.

II.19 Pengujian Kuesioner

Kuesioner adalah sebuah daftar pernyataan yang harus diisi oleh orang yang akan dievaluasi (responden). Metode yang digunakan dalam kuesioner pada penelitian ini adalah skala *Likert*. Dalam skala *likert*, responden diminta untuk membaca dengan seksama setiap pernyataan yang disajikan, kemudian ia diminta untuk menilai pernyataan-pernyataan tersebut.

Derajat penilaian responden terhadap suatu pernyataan terbagi dalam 5 kategori yang tersusun secara bertingkat, mulai dari Sangat Tidak Setuju (ST), Tidak Setuju (T), Tidak Memutuskan (N), Setuju (S), dan Sangat Setuju (SS). Atau dapat pula sebaliknya. Pernyataan tiap kuesioner dibuat berdasarkan aspek-aspek yang diteliti. Bobot pemberian skor yang digunakan dapat dilihat pada Tabel II.3 Panduan pemberian skor

Tabel II.3 Panduan pemberian skor

Bobot pendapat				
SS	S	N	T	TS
5	4	3	2	1

Skor yang telah dihitung pada setiap pernyataan kemudian dikalikan dengan masing-masing bobot tersebut sesuai dengan skenario kuesioner yang telah dibuat. Setelah itu total kan seluruh bobot jawaban tersebut kemudian bagi dengan total responden yang nantinya menjadi nilai rata-rata. Nilai rata-rata inilah yang diambil sebagai acuan sikap dimana jika nilai rata-rata kurang dari skala netral, maka dapat diartikan responden bersikap negatif, jika nilai rata-rata lebih dari skala netral, maka dapat diartikan responden bersikap positif, dan jika nilai rata-rata sama dengan skala netral maka responden bersikap netral terhadap tujuan yang ingin peneliti capai (Kothari, 2004). Untuk lebih jelasnya dapat dilihat pada rumus seperti berikut:

$$\overline{X} = \frac{\sum Total}{n} ,$$

Dimana,

$\overline{X} > 3$ bersikap positif

$\overline{X} < 3$ bersikap negatif

$\overline{X} = 3$ bersikap netral

Keterangan :

$\overline{X}$ = rata-rata

$\sum$ Total = jumlah seluruh nilai setelah dikalikan dengan bobot

n = total responden