BAB II
TINJAUAN PUSTAKA

1.1 Sistem Operasi Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencakup sistem operasi, *middleware* dan aplikasi. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi. Android dipuji sebagai platform mobile pertama yang lengkap, terbuka, dan bebas.

- Lengkap (*Complete Platform*): para desainer dapat melakukan pendekatan yang komprehensif ketika mereka sedang mengembangkan platform Android. Android merupakan sistem operasi yang aman dan banyak menyediakan *tools* dalam membangun software dan memungkinkan untuk peluang pengembangan aplikasi.
- Terbuka (*Open Source Platform*): Platform Android disediakan melalui lisensi open source. Pengembang dapat dengan bebas untuk mengembangkan aplikasi. Android sendiri menggunakan Linux kernel 2.6.
- Bebas (*Free Platform*): Android adalah platform/aplikasi yang bebas untuk *develope.* Tidak ada lisensi atau biaya royalti untuk dikembangkan pada platform Android. Tidak ada biaya keanggotaan yang diperlukan. Tidak diperlukan biaya pengujian. Tidak ada kontrak yang diperlukan. Aplikasi untuk Android dapat didistribusikan dan diperdagangkan dalam bentuk apapun.

Telepon pertama yang menggunakan sistem operasi Android adalah HTC Dream, yang dirilis pada 22 Oktober 2008. Pada penghujung tahun 2010 diperkirakan hamper semua vendor seluler didunia menggunakan Android sebagai sistem operasi[1]. Adapun versi-versi Android yang pernah dirilis adalah sebagai berikut:

1. Android versi 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 versi 1.5 (*Cupcake*)

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

3. Android versi 1.6 (*Donut*)

   Donut (versi 1.6) dirilis pada September dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai indicator dan kontrol applet VPN. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus; kamera, camcorder, dan galeri yang d2ntegrasikan, CDMA/EVDO, 802.1x, VPN, Gestures, dan Text-to-speech engine, kemampuan dial kontak, teknologi text to change speech (tidak tersedia pada semua ponsel, pengadaan resolusi VWGA.

4. Android versi 2.0/2.1 (*Eclair*)

   Pada 3 Desember 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1 (Éclair), perubahan yang dilakukan adalah pengoptimalan hardware, peningkatan Google Maps 3.2.1, perubahan UI dengan browser baru dan dukungan HTML5, daftar kontak yang baru, dukungan *flash* untuk kamera 3,2MP, digital zoom, dan Bluetooth 2.1.

5. Android versi 2.2 (*Froyo: Frozen Yoghurt*)

   Pada bulan Mei 2010 Android versi 2.2 Rev 1 diluncurkan. Fitur yang tersedia di Android versi ini sudah kompleks, diantaranya adalah:

   a) Kerangka aplikasi memungkinkan penggunaan dan penghapusan komponen yang tersedia.
   b) *Dalvik Virtual Machine* dioptimalkan untuk perangkat mobile.

c) Grafik: grafik di 2D dan grafis 3D berdasarkan libraries OpenGL.
d) SQLite: untuk menyimpan data.
e) Mendukung media: audio, video, dan berbagai format gambar.
f) Kamera, *Global Positioning System* (GPS), kompas, dan *accelerometer*.

6. Android versi 2.3 (*Gingerbread*)

   Android versi 2.3 diluncurkan pada desember 2010, hal-hal yang direvisi dari versi sebelumnya adalah kemampuan seperti berikut:

   a) SIP-based VoIP
   b) *Near Field Communications* (NFC)
   c) *Gyroscope* dan sensor
   d) *Multiple cameras support*
   e) *Mixable audio effect*
   f) *Download manager*

7. Android versi 3.1 (*Honeycomb*)

   Dirilis Mei 2011, Android ini diciptakan khusus untuk perangkat tablet dan tidak bisa digunakan di ponsel. Di desain khusus sehingga kecanggihan tablet dapat berjalan dengan kinerja maksimal. Memiliki fitur yang disesuaikan dengan tablet.

8. Android 4.0 (*Ice Cream Sandwich*)

   Android pertama yang menggunakan fitur pembuka kunci dengan pengenalan wajah. Android ini juga bisa mampu menghasilkan interface yang lebih bagus dan halus dibandingkan versi sebelumnya dan juga terdapat penambahan font pada versi ini.

9. Android 4.1 (*Jelly Bean*)

   Menggunakan fitur baru dan beberapa fitur di perbarui diantaranya adalah pencarian dengan suara yang lebih cepat, informasi cuaca, lalu lintas, dll. Juga mempunya keyboard virtual yang lebih sempurna. Dan Android jelly bean ini terus berkembang hingga versi 4.3.

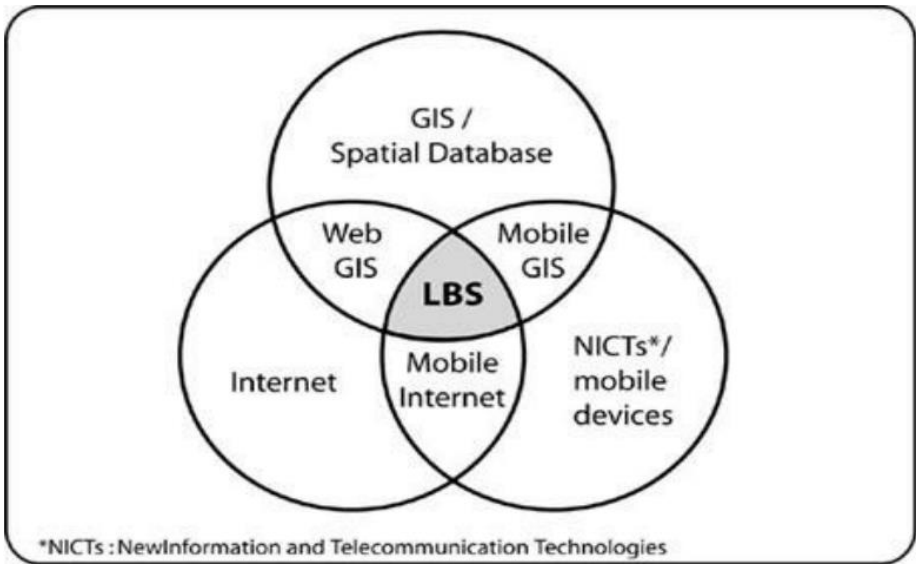
10. Android 4.4 (*Kitkat*)

    Sistem operasi ini mempunyai fitur yang sangat baru diantaranya adalah SMS yang terintegrasi langsung kedalam aplikasi Google *Hangouts.* Dalam versi ini juga terdapat fasilitas *Cloud Printing* serta dapat mendengarkan perintah suara dari google now tanpa menguras baterai dan bisa mengakses aplikasi kamera walaupun layar dalam keadaan terkunci.

1.2 Location Based Service

Layanan Berbasis lokasi adalah layanan informasi yang dapat diakses melalui *mobile network*, yang dilengkapi kemampuan untuk memanfaatkan lokasi daru *mobile device* tersebut. LBS memberikan kemungkinan komunikasi dan interksi dua arah[2].

Oleh Karena itu pengguna memberitahu penyedia layanan untuk mendapatkan informasi yang dia butuhkan, dengan referensi posisi pengguna tersebut. Layanan berbasis lokasi dapat digambarkan sebagai suatu layanan yang berada pada pertemuan tiga teknologi yaitu : *Geographic System, Internet System*, dan *Mobile Devices*, hal ini dapat dilihat pada gambar LBS adalah pertemuan dari tiga teknologi[2].



Gambar 2.1 Teknologi Location Based Service

LBS (*Location Based Service)* merupakan suatu layanan yang bereaksi aktif terhadap perubahan entitas posisi sehingga mampu mendeteksi letak objek dan memberikan layanan sesuai dengan letak objek yang telah diketahui tersebut.

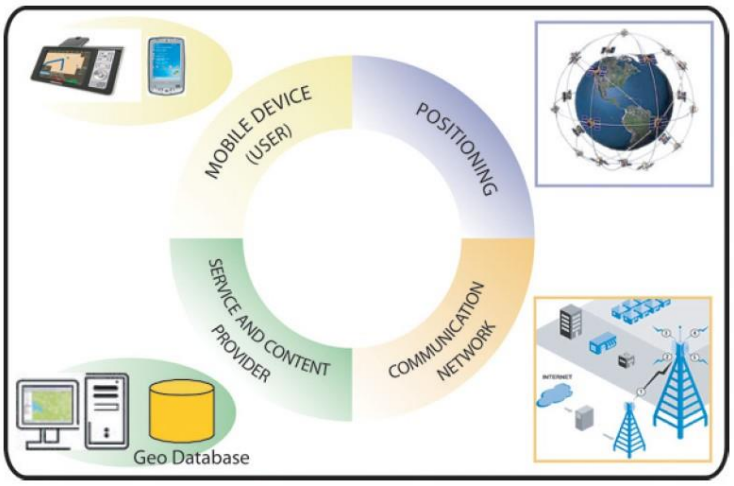
Pada teknologi LBS berbasis jaringan selular, oenentuan posisi sebuah peralatan komunikasi bergerak ditentukan berdasarkan posisi relatif peralatan tersebut terhadap lokasi BTS (*Base Transceiver Station).* Dalam menentukan posisi dari sebuah handphone yang sedang aktif, secara umum terdapat tiga tingkat metode yang digunakan saat ini yaitu :

- Metode *Basic Positioning* yang Berbasis pada *Cell Identification* (Cell ID)
- Metode *Enchanced Positioning*
- Metode *Advanced Positioning*

Komponen *Location Based Service (*LBS)

Dalam menggunakan layanan berbasis lokasi elemen yang diperlukan antara lain:

1. *Mobile Device* yaitu sebuah alat yang digunakan untuk meminta informasi yang dibutuhkan. Biasanya perangkat yang memungkinkan yaitu PDA, *Mobile Phones*, Laptop, dan perangkat lainnya yang mempunyai fasilitas navigasi.
2. *Communication Network* adalah jaringan selular yang mengirimkan data pengguna dan permintaan layanan.
3. *Positioning Component* untuk pengolahan layanan biasanya posisi pengguna harus ditentukan. Posisi pengguna dapat diperoleh menggunakan jaringan komunikasi atau dengan menggunakan *Global Positioning System (GPS).*
4. *Service and Application Provider* adalah penyedia layanan pengguna selular yang bertanggung jawab untuk memproses layanan.
5. *Data and Content Provider* yaitu penyedia layanan informasi data yang dapat diminta oleh pengguna[2].

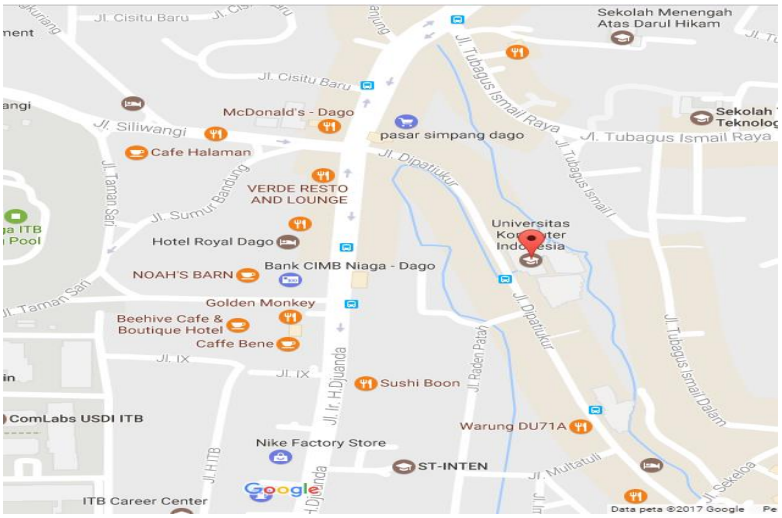


Gambar 2.2 Komponen LBS

1.3 Google Maps

Google *Maps* merupakan sebuah layanan gratis dari Google mengenai berbagai macam teknologi pemetaan. Google *Maps* menampilkan peta dunia secara digital sehingga dapat dengan mudah 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 [3].

Google *Maps* dibuat dengan menggunakan kombinasi dari gambar peta, *database*, serta objek-objek interaktif yang dibuat menggunakan bahasa pemograman *Hyper Text Markup Language* (HTML), Java dan beberapa bahasa pemograman lainnya. Gambar-gambar peta yang muncul pada layar merupakan hasil komunikasi dari pengguna dengan *database* pada *web server* Google untuk menampilkan gambar yang diminta [3].



Gambar 2.3 *Google Maps*

1.3.1 Google Maps Application Programing Interface (Google Maps Api)

Google *Maps* API merupakan sebuah layanan yang memungkinkan Google *Maps* agar dapat diintegrasikan kedalam aplikasi yang memang membutuhkan fitur dari Google *Maps* ini. Dengan menggunakan Google *Maps* API ini memudahkan para pengembang dalam membuat aplikasi yang membutuhkan teknologi peta didalamnya. Dengan menggunakan Google *Maps* API ini sudah tersedia semua yang dibutuhkan terkait teknologi peta *digital* [3].

1.4 Global Positioning System (GPS)

Global Positioning System (GPS) merupakan sistem navigasi atau penentu posisi berbasis satelit. Sistem ini mengirimkan sinyal gelombang mikro ke bumi dengan menggunakan 24 satelit. Sinyal yang dikirimkan oleh satelit akan diterima oleh GPS *receiver* [4]. Contohnya sebagai berikut :

1. Waktu

   GPS receiver menerima informasi waktu dari jam atom yang mempunyai keakurasian sangat tinggi.

2. Lokasi

   GPS memberikan informasi lokasi dalam tiga dimensi, yaitu *latitude*, *longitude* dan *elevasi*.

3. Kecepatan

   Ketika berpindah tempat, GPS dapat menunjukkan informasi kecepatan berpindah.

4. Arah Perjalanan

   GPS dapat menunjukkan ara tujuan.

5. Simpan Lokasi

   Tempat-tempat yang sudah pernah atau ingin dikunjungi bisa disimpan oleh GPS receiver.

6. Komulasi Data

   GPS receiver dapat menyimpan informasi track, seperti total perjalanan yang sudah pernah dilakukan, kecepatan rata-rata, kecepatan paling tinggi, kecepatan paling rendah, waktu/jam sampai tujuan dan sebagainya.

GPS terdiri dari tiga bagian yaitu sistem kontrol, satelit dan pengguna. Sistem kontrol merupakan 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 didalamnya. GPS biasanya digunakan untuk menunjukan suatu lokasi yang berada dipermukaan bumi dengan tingkat akurasi yang cukup baik yaitu kurang dari 10 meter selama tidak ada benda padat yang dapat menghambat sinyal untuk mendapatkan lokasi pengguna [4].

1.4.2 Akurasi Global Positioning System (GPS)

Posisi yang ditunjukkan oleh suatu GPS mempunyai faktor kesalahan atau juga disebut tingkatan akurasi. Sebagai contoh suatu alat GPS menunjukkan titik koordinat dengan tingkat akurasi 5 meter, itu berarti posisi pengguna bisa berada dalam range radius 5 meter dari titik yang ditunjukkan tersebut.

Ada beberapa hal yang mempengaruhi tingkat akurasi tersebut, antara lain[4].

1. Kesalahan *Ephemeris*

   Terjadi jika satelit tidak dapat mentransmisikan posisi diorbit dengan tepat.

2. Keadaan *Lonosphere*

Lonosphere berada pada jarak sekitar 43.50 mil di atas permukaan bumi. Satelit yang melewati *ionosphere* akan menjadi lambat dikarenakan adanya *plasma* (gas dengan tingkat kepadatan rendah). Walaupun GPS *receiver* berusaha untuk mengkoreksi/memperbaiki faktor keterlambatan yang terjadi tetap saja aktivitas tertentu dari *plasma* bisa menyebabkan kesalahan perhitungan.

3. Keadaan *Troposphere*

   Troposphere adalah bagian terendah dari *atmosfer* sampai dengan ketinggian sekitar 11 mil dari permukaan tanah. Variasi pada temperatur, tekanan dan kelembaban bisa menyebabkan perbedaan kecepatan penerima gelombang radio.

4. Kesalahan Waktu

   Kesalahan waktu dari GPS *receiver* yang tidak presisi dapat menimbukan ketidakakurasian.

5. Kesalahan *Multipath*

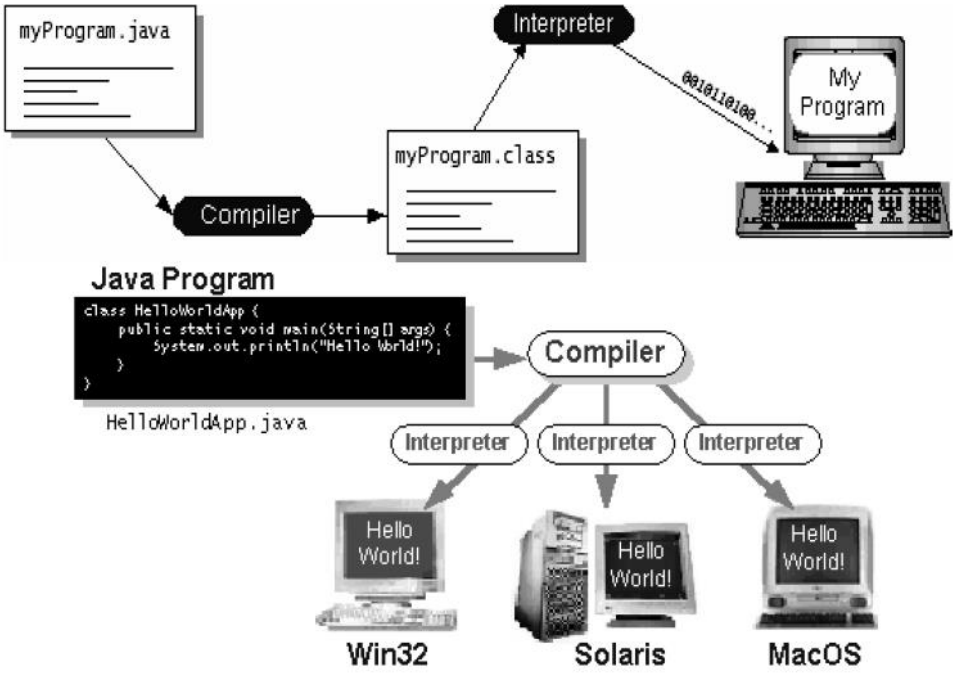
   Terjadi karena sinyal satelit membentur permukaan keras (seperti bangunan atau tebing) sebelum mencapai GPS *receiver*. Hal tersebut bisa menyebabkan terjadinya *delay* sehingga perhitungan jarak menjadi tidak akurat.

6. Buruknya Sinyal Satelit

   Keadaan sekitar atau keadaan lingkungan dapat juga menyebabkan GPS sulit untuk menerima data satelit.

1.5 Pemrograman Java

Java adalah program yang sangat populer di lingkungan pendidikan saat ini, hampir di seluruh kampus memiliki satuan ajar pemrograman ini. Hal yang paling mendasar dan harus dikuasai oleh Programer Java adalah memasang library (kelas dalam bentuk biner). Karena java yang dirancang secara orientasi objek, tentunya program yang terdiri atas kesatuan antar objek yang membentuk aplikasi. Sehingga sebuah objek tertentu bisa digunakan dalam beberapa aplikasi yang berbeda. Objek ini biasanya bersifat umum penggunaannya, misalkan objek kalender, objek *driver* basis data, objek *report* dan lain-lain[6].



Gambar 2.4 Cara Kerja Pemrograman Java

1.6 Firebase Realtime Database

Firebase Realtime Database adalah database yang di-host di cloud. Data disimpan sebagai JSON dan disinkronkan secara realtime ke setiap klien yang terhubung. Ketika membuat aplikasi lintas-platform dengan SDK Android, iOS, dan JavaScript, semua klien akan berbagi sebuah instance Realtime Database dan menerima update data terbaru secara otomatis.

Firebase Realtime Database memungkinkan untuk membuat aplikasi kolaboratif dan kaya fitur dengan menyediakan akses yang aman ke database, langsung dari kode sisi klien. Data disimpan di drive lokal. Bahkan saat offline sekalipun, peristiwa realtime terus berlangsung, sehingga pengguna akhir akan merasakan pengalaman yang responsif. Ketika perangkat kembali terkoneksi, Realtime Database akan menyinkronkan perubahan data lokal dengan update jarak jauh yang terjadi selama klien offline, sehingga setiap perbedaan akan otomatis digabungkan.

Realtime Database menyediakan bahasa aturan berbasis ekspresi yang fleksibel, atau disebut juga Aturan Keamanan Firebase Realtime Database, untuk menentukan metode strukturisasi data dan kapan data dapat dibaca atau ditulis. Ketika diintegrasikan dengan Firebase Authentication, developer dapat

menentukan siapa yang memiliki akses ke data tertentu dan bagaimana mereka dapat mengaksesnya.

Realtime Database adalah database NoSQL, sehingga memiliki pengoptimalan dan fungsionalitas yang berbeda dengan database terkait. API Realtime Database dirancang agar hanya mengizinkan operasi yang dapat dijalankan dengan cepat. Hal ini memungkinkan untuk membangun pengalaman realtime yang luar biasa dan dapat melayani jutaan pengguna tanpa mengorbankan kemampuan respons. Firebase terbagi menjadi beberapa layanan yang disediakan antara lain *Develop*, *Grow*, dan *Earn*. Masing-masing layanan ini terbagi menjadi beberapa macam dan memiliki fungsi-fungsi tersendiri[7].

1.7 Pendekatan Berorientasi Objek

Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktivitas analisis berorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek.

Pada saat ini metode berorientasi objek banyak dipilih karena metodologi lama banyak menimbulkan masalah seperti adanya kesulitan pada saat mentransformasi hasil dari suatu tahap pengembangan ke tahap berikutnya, misalnya pada metode pendekatan terstruktur, jenis aplikasi yang dikembangkan saat ini berbeda dengan masa lalu. Aplikasi yang dikembangkan pada saat ini sangat beragam (aplikasi bisnis, *real-time, untility*, dan sebagainya) dengan *platform* yang berbeda-beda, sehingga menimbulkan tuntutan kebutuhan metodologi pengembangan yang dapat mengakomodasi ke semua jenis aplikasi tersebut.

Keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut:

1. Meningkatkan produktivitas

   Karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (*reusable*).

2. Kecepatan pengembangan

   Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan pada saat pengkodean.

3. Kemudahan pemeliharaan

   Karena dengan model objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering berubah-ubah.

4. Adanya konsistensi

   Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.

5. Meningkatkan kualitas perangkat lunak

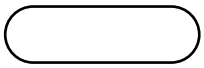
   Karena pendekatan pengembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat pengembangannya, perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

Saat ini sudah banyak bahasa pemrograman berorientasi objek. Banyak yang berfikir bahwa pemrograman berorientasi objek identik dengan bahasa Java. Memang bahasa java merupakan bahasa yang paling konsisten dalam mengimplementasikan paradigma pemrograman berorientasi objek. Namun sebenarnya bahasa pemrograman yang mendukung pemrograman berorientasi objek tidak hanya bahasa Java[8].

1.7.2 Unified Modelling Language (UML)

Berikut akan dijelaskan mengenai diagram yang akan digunakan dalam pembangunan sistem penyewaan sepeda berbasis Android ini, yaitu *use case diagram*, *Activity diagram*, *sequence diagram*, dan *class diagram[8]*.

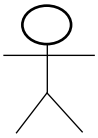
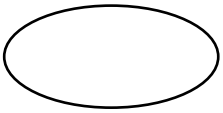
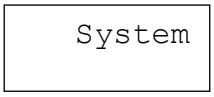
Tabel I.1 Definisi Simbol Activity Diagram

Simbol	Nama Simbol	Fungsi
	*Initial State*	Pernyataan awal dari sebuah aktivitas
	*Final State*	Pernyataan akhir dari sebuah aktivitas
	Transisi	Menggambarkan alur dari aksi-aksi yang ada pada aktivitas
	Aksi	Menggambarkan aksi yang ada pada aktivitas

1. Use Case Diagram

   Use case diagram merupakan pemodelan untuk (*behavior)* aplikasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan aplikasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah aplikasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Penjelasan tentang simbol-simbol *use case* dapat dilihat pada Tabel 2.2 berikut ini.

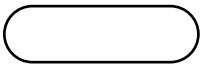
Tabel I.2 Definisi Simbol Use Case Diagram

Simbol	Nama Simbol	Fungsi
	Aktor	Merupakan manusia, proses, atau sistem lain yang berinteraksi dengan aplikasi yang akan dibuat. Walaupun simbol aktor adalah manusia, tapi aktor tidak harus selalu manusia
	*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
	Association	Merelasikan Aktor dengan *use case*
	*System Boundary*	Menggambarkan batasan sistem terhadap lingkungannya

2. Activity Diagram

Activity diagram menggambarkan aliran kerja atau aktivitas dari sebuah sistem, proses, atau menu yang ada pada perangkat lunak. Satu hal yang perlu diperhatikan adalah *Activity diagram* menggambarkan aktivitas sistem, bukan apa yang dilakukan oleh aktor. Tabel 2.3 adalah penjelasan tentang simbol-simbol pada *Activity diagram*.

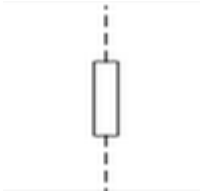
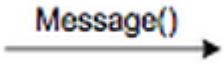
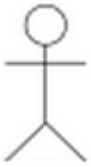
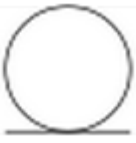
Tabel I.3 Definisi Simbol Activity Diagram

Simbol	Nama Simbol	Fungsi
	*Initial State*	Pernyataan awal dari sebuah aktivitas
	*Final State*	Pernyataan akhir dari sebuah aktivitas
	Transisi	Menggambarkan alur dari aksi-aksi yang ada pada aktivitas
	Aksi	Menggambarkan aksi yang ada pada aktivitas

3. Sequence Diagram

Sequence diagram menggambarkan *behavior* objek pada *use case* dengan mendeskripsikan waktu hidup dan pesan, yang dikirim/ diterima antar objek. Oleh karena itu, untuk menggambar *sequence diagram* harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang d2nstansiasi menjadi objek itu. Dalam membangun *sequence diagram*, harus melihat dan mengacu pada skenario use case. Penjelasan tentang simbol-simbol *sequence diagram* dapat dilihat pada Tabel 2.4.

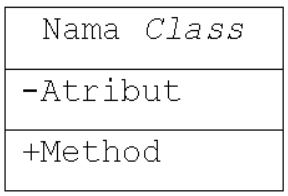
Tabel I.4 Definisi Simbol Sequence Diagram

Simbol	Nama Simbol	Fungsi
	*Lifelines*	Menggambarkan waktu mulai dan berakhirnya sebuah message
	*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi tentang aktfitas yang terjadi
	*Actor*	Digunakan untuk menggambarkan pengguna / pengguna
	*Boundary*	Digunakan untuk menggambarkan sebuah form atau antarmuka
	*Control Class*	Digunakan untuk menghubungkan boundary dengan entity
	*Entity Class*	Menyimpan data/informasi yang digunakan dalam perangkat lunak

4. Class Diagram

Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Class diagram dibuat agar pembuat program atau *programmer* membuat kelas-kelas sesuai dengan rancangan di dalam class diagram, agar antara dokumen perancangan dan perangkat lunak sesuai. Gambar 2.5 merupakan contoh dari class diagram. Berikut penjelasan mengenai atribut dan operasi pada class diagram.

a) Atribut; merupakan variabel-variabel yang dimiliki oleh suatu kelas.
b) Operasi (atau Metode); adalah fungsi-fungsi yang dimiliki oleh suatu kelas.



Gambar I.5 Contoh Class Diagram