BAB II

LANDASAN TEORI

Pada bab ini akan diuraikan tentang teori-teori yang melandasi penulisan Laporan Tugas Akhir ini. Teori-teori tersebut dijabarkan sebagai berikut:

2.1 Definisi *Tracking System*

Tracking system adalah suatu sistem yang dapat melakukan pelacakan atau pencarian dan pelacakan terhadap suatu hal,dengan memberikan informasi mengenai hal tersebut. Misalnya terjadi kehilangan ponseldampak yang di akibatkan ialah kehilangan data ataupun *contact* penting yang terdapat pada ponsel tersebut.Dan dikhawatirkan digunakan oleh yang tidak berkepentingan. Maka dengan adanya *tracking system* agar memudahkan pelacakan barang, yang dikirimkan dapat hilang dalam perjalanan ataupun terlambat sampai tujuan. Jika pada saat terjadi masalah, pihak terkait dapat langsung mengetahui masalah mengenai barang mereka, tentu mereka dapat segera mengambil suatu keputusan sehingga masalah yang terjadi tidak terlalu merugikan mereka.

2.2 Definisi GPS (*Global Positioning System)*

GPS adalah singkatan dari *Global Positioning System* yang merupakan sistem untuk menentukan posisi dan navigasi global dengan mengunakan satelit. Sistem yang pertama kali dikembangkan oleh departemen pertahanan Amerika ini digunakan untuk kepentingan militer maupun sipil (survey dan pemetaan).

Sistem GPS, yang nama aslinya adalan NAVSTAR GPS (NAVIGATION Satelite Timing and Ranging Global Positioning System) mempunyai tiga segmen yaitu: satelit, pengonrol, dan penerima/pengguna. Satelit GPS yang mengorbit bumi, dengan orbit dan kedudukan yang tetap (koordinatnya pasti), seluruhnya berjumlah 24 buah dimana 21 buah aktif bekerja dan 3 buah sisanya adalah cadangan.

1. **Satelit**, bertugas untuk menerima dan menyimpan data yang ditranmisikan oleh stasiun-stasiun pengontrol. Menyimpan dan menjaga informasi waktu berketelitian tinggi (ditentukan dengan jam atomik di satelit), dan memancarkan sinyal dan informasi secara kontinyu ke pesawat penerima (*receiver*) dari pengguna
2. **Pengontrol**, bertugas untuk mengendalikan dan mengtrol satelir dari bumi baik untuk mengecek kesehatan satelit, penentuan dan prediksi orbit waktu, singkronisasi waktu antar satelit, dan mengirim data ke satelit.
3. **Penerima** (*receiver*), bertugas menerima data dari satelit dan memprosesnya untuk menentukan posisi (posisi tiga dimensi yaitu koordinat di bumi plus ketinggian), arah, jarak dan waktu yang diperlukan oleh pengguna. Ada dua macam penerima (*receiver*) yaitu tipe NAVIGASI dan tipe GEODETIC. Yang termasuk tipe *receiver* NAVIGASI antara lain : Trimble Ensign, Trimble Pathfinder, Garmin, Sony dan lain-lainnya. Sedangkan tipe GEODETIC antara lain : Topcon, Leica, Astech, Trimble seri 4000 dan lain-lain.

Dalam bidang survei dan pemetaan wilayah terumbu karang, GPS dapat digunakan untuk menentukan posisi titik-titik lokasi penyelaman maupun transek. Posisi yang diperoleh adalah posisi yang benar terhadap sistem koordinat bumi. Dengan mengetahui posisinya yang pasti, lokasi-lokasi penyelaman maupun transek dapat di-plot-kan kedalam peta kerja.

2.2.1 Penentuan posisi dengan GPS

Pada dasarnya penentuan posisi dengan GPS adalah pengukuran jarak secara bersama-sama ke beberapa satelit (yang koordinatnya telah diketahui) sekaligus. Untuk menentukan koordinat suatu titik di bumi, receiver setidaknya membutuhkan 4 satelit yang dapat ditangkap sinyalnya dengan baik. Secara *default* posisi atau

koordinat yang diperoleh bereferensi ke global datum yaitu World Geodetic System 1984 atau disingkat WGS'84.

Secara garis besar penentuan posisi dengan GPS ini dibagi menjadi dua metode yaitu **metode absolut** dan **metode relatif.**

- **Metode absolut** atau juga dikenal sebagai ***point positioning,*** menentukan posisi hanya berdasarkan pada 1 pesawat penerima (*reciever*) saja. Ketelitian posisi dalam beberapa meter (tidak berketelitian tinggi) dan umumnya hanya diperuntukan bagi keperluan NAVIGASI.
- **Metode relatif** atau sering disebut ***differential positioning,*** menentukan posisi dengan menggunakan lebih dari sebuah *receiver.* Satu GPS dipasang pada lokasi tertentu dimuka bumi dan secara terus menerus menerima sinyal dari satelit dalam jangka waktu tertentu dijadikan sebagai referensi bagi yang lainnya. Metode ini menghasilkan posisi ketelitian tinggi (umunya kurang dari 1 meter) dan diaplikasikan untuk keperluan survei GEODESI ataupun pemetaan yang memerlukan ketelitian tinggi.

Untuk keperluan survei di wilayah terumbu karang metode absolute yang menggunakan satu *receiver* tipe NAVIGASI rasanya sudah cukup mewadahi. Akan tetapi bila ingin mempelajari tentang pergeseran terumbu karang dari waktu ke waktu misalnya, diperlukan metode relative dengan menggunakan *receiver* tipe GEODETIC.

2.2.2 Sumber kesalahan GPS

Beberapa kesalahan dalam penentuan posisi dengan metode absolut ini antara lain disebabkan oleh : efek **multipath,** efek **selective availability (SA),** maupun kesalahan karena ketidak sinkronan antara peta kerja dan *setting* yang dilakukan saat menggunakan GPS.

1. **Multipath** adalah fenomena dimana sinyal dari satelit tiba di antenna *receiver* melalui dua atau lebih lintasan yang berbeda. Hal ini biasa terjadi jikalau kita melakukan pengukuran lokasi-lokasi yang dekat dengan benda reflektif, seperti disamping gedung tinggi, dibawah kawat tranmisi tegangan tinggi atau lainnya. Untuk mengatasinya : menghindari pengamatan dekat benda reflektif, pakai satelit yang benar-benar baik saja, lakukan pengukuran berulang-ulang dan dirata-rata hasilnya.

2. **Selective Availability (SA)** adalah teknik pemfilteran yang diaplikasikan untuk memproteksi ketelitian tinggi GPS bagi khalayak umum dengan cara mengacak sinyal-sinyal dari satelit terutama yang berhubungan dengan informasi waktu. Koreksinya hanya dapat dilakukan oleh pihak yang berwenang mengelola GPS ataupun pihak militer Amerika saja. Pihak-pihak lain yang mempunyai ijin untuk menggunakan data berketelitian tinggi biasanya juga diberi tahu cara koreksinya. **SA** ini merupakan sumber kesalahan paling besar bagi penentuan posisi dengan metode absolut. Namun dengan menerapkan metode relatif (*differential positioning*) kesalahan tersebut dapat dikurangi. Selain itu belum lama ini pihak militer Amerika telah merevisi kebijakan dalam menerapkan **SA** ini sehingga saat ini dengan metode absolut-pun ketelitianya sudah sangat baik disbanding sebelumnya (sudah tidak dalam puluhan meter lagi kesalahanya).

3. Ketidak akuratan posisi karena *setting receiver* yang tidak pas ini hanya dapat diatasi dengan menge-set parameter GPS saat dipakai sesuai dengan parameter peta kerja yang dipergunakan. Hal tersebut biasanya terkait dengan sistem proyeksi dan koordinat, serta datum yang digunakan dalam peta kerja.

2.2.3 Kelebihan dan Kekurangan GPS

- Kelebihan gps :

    1. GPS untuk NAVIGASI aplikasi GPS di bidang militer pada umumnya dapat dibagi menjadi beberapa bagian misalnya,pemetaan (penentuan posisi titik-titik target terutama pada masalah topografi angkatan darat, pencitraan, foto udara, dan beberapa analisis spasial yang ditujukan untuk mendukung perencanaan operasi), navigasi, tracking (monitoring atau pemantauan), atau bahkan sebagai tools penuntun posisi-posisi sasaran peluru kendali, rover, uav, dan auv cthny bisa di lihat di toko gps. NAVIGASI sering kali dilakukan oleh personel militer yang sedang menempuh perjalanan dari suatu tempat ke tempat-tempat lain yang menjadi targetnya. Oleh karena itu, dengan mengkombinasikan peta, kompas, dan GPS (*receiver*), maka proses navigasi menjadi lebih mudah dan menyenangkan bagi siapapun.

Demikian pula bagi personel militer yang bergerak dengan menggunakan platform (kendaraan), bila menggunakan peta (terutama digital) dan GPS (*receiver*), navigasinya menjadi jauh lebih mudah, menyenangkan, dan cepat.

- Kekurangan GPS :

    1. Penggunaan GPS untuk mengetahui posisi yang mengandalkan setidaknya tiga satelit ini tidak selamanya akurat.
    2. Terkadang, dibutuhkan satu satelit untuk memperbaiki sinyal yang diterima. Ketidakakuratan posisi yang ditunjukkan.
    3. Gps ini dipengaruhi oleh posisi satelit yang berubah dan adanya proses sinyal yang ditunda. Kecepatan sinyal GPS ini juga seringkali berubah

karena dipengaruhi oleh kondisi atmosfer. Selain itu, sinyal GPS juga mudah berinteferensi dengan gelombang elektromagnetik lainnya

2.3 Definisi *SmartPhone*

Ponsel pintar (*SmartPhone*) adalah ponsel yang mempunyai kemampuan tingkat tinggi kadang-kadang dengan fungsi yang menyerupai komputer. Belum ada standar pabrik yang menentukan definisi telepon pintar. Bagi beberapa orang, telepon pintar merupakan telepon yang bekerja enggunakan seluruh piranti lunak sistem operasi yang menyediakan hubungan standar dan mendasar bagi pengembang aplikasi. Bagi yang lainnya, telepon pintar hanyalah merupakan sebuah telepon yang enyajikan fitur canggih seperti surel (surat elektronik), internet dan kemampuan membaca buku elektronik (*e-book*) atau terdapat papan ketik (baik *built-in* maupun eksternal) dan konektor VGA. Dengan kata lain, ponsel pintar merupakan komputer mini yang mempunyai kapabilitas sebuah telepon.

Pertumbuhan permintaan akan alat canggih yang mudah dibawa kemana-mana membuat kemajuan besar dalam prosesor, memori, *layar* dan sistem operasi yang di luar dari jalur telepon genggam sejak beberapa tahun ini.

Belum ada kesepakatan dalam industri ini mengenai apa yang membuat telepon menjadi "pintar", dan pengertian dari telepon pintar itu pun berubah mengikuti waktu. Menurut David Wood, Wakil Presiden Eksekutif PT Symbian OS, "Telepon pintar dapat dibedakan dengan telepon genggam biasa dengan dua cara fundamental: bagaimana mereka dibuat dan apa yang mereka bisa lakukan." Pengertian lainnya memberikan penekanan berbedaan dari dua faktor ini.

"Dengan menggunakan ponsel pintar hanya merupakan sebuah hanya evolusi dari jenjang-jenjang evolusi, jadi kemungkinan alat ini pada titik tertentu akan menjadi lebih kecil dan kita tidak akan menyebutnya ponsel lagi, tetapi ia akan terintegrasi kesepakatannya di sini adalah untuk membuat alat ini menjadi se-tidak

terlihat mungkin, antara anda, dan apa yang anda ingin lakukan" kata Sacha Wunsch-Vincent pada OECD (Organisasi untuk Kerjasama dan Pengembangan Ekonomi).

Kebanyakan alat yang dikategorikan sebagai ponsel pintar menggunakan sistem operasi yang berbeda. Dalam hal fitur, kebanyakan telepon pintar mendukung sepenuhnya fasilitas surel dengan fungsi pengatur personal yang lengkap. Fungsi lainnya dapat menyertakan miniatur papan ketik QWERTY, layar sentuhatau D-pad, kamera, pengaturan daftar nama, penghitung kecepatan, navigasi piranti lunak dan keras, kemampuan membaca dokumen bisnis, pemutar musik, penjelajah foto dan melihat klip video, penjelajah internet, atau hanya sekedar akses aman untuk membuka surel perusahaan, seperti yang ditawarkan oleh BlackBerry. Fitur yang paling sering ditemukan dalam ponsel pintar adalah kemampuannya menyimpan daftar nama sebanyak mungkin, tidak seperti ponsel biasa yang mempunyai batasan maksimum penyimpanan daftar nama.

2.4 Definisi Aplikasi pada Ponsel

Aplikasi adalah suatu subkelas perangkat lunakponsel yang memanfaatkan kemampuan operating sistem langsung untuk melakukan suatu tugas yang diinginkan pengguna. Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan ponsel, tapi tidak secara langsung menerapkan kemampuan tersebut untuk mengerjakan suatu tugas yang menguntungkan pengguna. Contoh utama perangkat lunak aplikasi adalah chating, browsing, dan pemutar media.

Definisi *mobile computing* yaitu:

1. *Mobile computing* merupakan paradigma baru dari teknologi yang mampu melakukan komunikasi walaupun pengguna melakukan perpindahan.
2. Merupakan kemajuan teknologi komputer, sering disebut sebagai *mobile computer* (portable komputer) yang dapat berkompunikasi dengan jaringan tanpa kabel.

Adapun platform aplikasi mobile seperti BREW (*Binary Runtime Environment for Wireless*), Symbian, Windows Mobile, IOS, Blackberry OS, Palm, Meego, Bada, Android, dll.

2.5 Definisi Sistem Operasipada Ponsel

Seperti halnya sistem operasi pada komputer, sistem operasi ponsel adalah software utama yang melakukan menejemen dan kontrol terhadap hardware secara langsung serta memanajemen dan mengontrol software-soft lain sehingga software-software lain tersebut dapat bekerja. Sehingga suatu sistem operasi ponsel (*mobile operating system*) akan bertanggung jawab dalam mengoperasikan berbagai fungsi dan fitur yang tersedia dalam perangkat ponsel tersebut seperti, *schedulling task*, *keyboard*, WAP, *email*, *text message*, sinkronisasi dengan aplikasi dan perangkat lain, memutar musik, kamera, dan mengontrol fitur-fitur lainnya. Banyak perusahaan ponsel yang memasukan sistem operasi dalam produknya baik pada PDA, Smartphone maupun handphone. Perkembangan aplikasi atau game selular (*mobile content*)sangat cepat, perusahaan pembuat *mobile Operating System* (OS) telah berlomba untuk memasarkan produk-produk mereka dengan menciptakan fungsi-fungsi dan teknologi yang kian hari kian memanjakan pengguna *smartphone* (selular yang ber-OS) dari segi entertainment dan fungsionalitas penggunaan selular untuk memudahkan tugas sehari-hari. Selain berfungsi untuk mengkontrol sumber daya hardware dan software ponsel seperti keypad, layar, phonebook, baterai, dan koneksi ke jaringan, sistem operasi juga mengontrol agar semua aplikasi bisa berjalan stabil dan konsisten. Sistem operasi harus dirancang fleksibel sehingga para software developer lebih mudah menciptakan aplikasi-aplikasi baru yang canggih. Keunggulan lain dari ponsel yang ber-OS adalah memiliki kebebasan lebih untuk men-download berbagai aplikasi tambahan yang tidak disediakan oleh vendor ponsel.Sistem operasi yang dapat ditemukan di telepon pintar adalah Symbian OS, iPhone OS, RIM BlackBerry, Windows Mobile, Linux, Palm WebOS dan Android. Android dan

WebOS dibuat oleh Linux, dan iPhone OS dibuat oleh BSD dan sistem operasi NeXTSTEP berhubungan dengan Unix.

2.6 Definisi Sistem Operasi Android

Pertanyaan pertama yang akan kita coba jawab dalam bab ini, untuk menghindari kesalahpahaman yang sering terjadi tentang android ini. Kita mulai dengan mengartikan android dengan singkat.

Android bukanlah sebuah *mobile phone* (Google-phone) seperti Apple dengan Iphone-nya, dan juga Android bukanlah bahasa pemrogaman, walaupun mememiliki SDK dengan API dan *runtime environment* tersendiri (virtual mesin sendiri) untuk aplikasi-nya (semuanya ditulis dalam bahasa pemgrogaman Java)

Android Inc yang merupakan pendatang baru yang membuat piranti lunak untuk ponsel / *smartphone*. Kemudian untuk mengembangkan Android, dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan piranti keras, piranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia.

Apakah itu android, Android antara lain sebuah *operating system* yang dibuat oleh perusahaan besar google tapi tak hanya itu saja, pada saat yang sama dalam singkat kata Android merupakam *software stack* untuk mobile phone yang termasuk pada sistem operasi middleware dan aplikasi kunci

2.6.1 Kenapa Android ?

Android adalah *software* yang cocok untuk perangkat ponselyang mencakup sistem operasi *middle-ware* dan yang berfungsi sebagai kuncinya aplikasi.Android memiliki kerangka aplikasi dalvik *virtual machine* , mendukung multimedia, integrasi *browser* dan dapatmengoptimalkan grafis. Dan Android juga mendukung perangkat-perangkat lain seperti GPS, *Bluetooth*, Sensor , *Acceleromoter*, kamera, jaringan wifi, 3G hingga HSPA. Maka android merupakan sebuah sarana yang menjembatani antar

komponen. Gambar 2.1 dibawah ini menunjukan komponen utama dari sistem operasi Android. Projek ini langsung memanfaatkan aplikasi dan lapisan kerangka aplikasi dan juga memanfaatkan *library* SQLite

Beberapa fitur yang terdapat dalam Android yaitu:

1. Application Framework, memungkinkan penggunaan dan penggantian komponen.
2. Dalvik Virtual Machine, mengoptimalkan perangkat mobile.
3. Integrated Browser, berbasis open source WebKit engine.
4. Optimized Graphics, didukung oleh library grafis 2D dan 3D berbasis spesifikasi OpenGL ES 1.0.
5. SQLite, untuk menyimpanan data terstruktur.
6. Media Support, untuk audio dan video dengan format (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
7. GSM Telephony (tergantung handset)
8. Bluetooth, EDGE, 3G,HSPA and WiFi (tergantung handset).
9. Camera, GPS, Compass, Accelerometer (tergantung handset).
10. Rich Development Environment, tersedianya dukungan yang penuh untuk para pengembang aplikasi termasuk emulator, tools untuk debugging, memori dan kinerja profil, dan plugin untuk IDE Eclips

Saat ini Sistem Operasi yang mendominasi telpon selular Apple dengan 'IOS, Google dengan Android, Nokia dengan Symbian, Blackberry dengan OS dan terakhir Mircrosoft dengan Windows nya. Kriteria yang dipertimbang untuk dipilih adalah pangsa pasar, *support tablet*, kemudahan dalam pengembangan aplikasi, mendukung *desktop platform* dan lisensi aplikasi. Tabel 2.1 membandingkan sistem

operasi menurut kriteria-kriteria tersebut dengan bahasa pemrograman dan yang didukung oleh platform pengembang resmi.

Tabel 2.1 Perbandingan Sistem Operasi Ponsel

	Lisensi	**Bahasa pemrograman**	**Pengembangan Os platform**
Android	Open Source	Mainly Java	Cross platform
iOS	Propirietary	Objective-C	Mac only
Windows Mobile	Propirietary	Dot NET(C#, C++, VB. Net)	Windows Only
Blackberry Os	Propirietary	Java	Windows
Symbian	Propirietary	C++	Cross platform

Android disukai karena sifat open source, kemudahan pengembangan, nol hambatan masuk dan kegunaan. "cepat memperoleh tanggapan oleh pengembang, karena fitur-fiturnya sepenuhnya dikembangkan untuk mengeksploitasi model awan-komputasi yang ditawarkan oleh *web resources* dan juga meningkatkan pengalaman dengan penyimpanan data lokal pada handset itu sendiri" saat iniadalah pemimpin pasar dalam ponsel pintar aplikasi ditulis untuk handset juga dapat dengan mudah digunakan untuk perangkat tablet dengan perubahan teknis sedikit atau tidak dibutuhkan. Android dirilis di bawah dua lisensi sumber terbuka yang berbeda. Kernel Linux yang didasarkan pada, dirilis di bawah GNU Public License (GPL) seperti yang diperlukan bagi siapa saja lisensi kernel sistem operasi Linux. Platform Android, tidak termasuk kernel, dilisensikan di bawah Apache Software License (ASL).

2.6.2 *Open Handset Alliance* Android

Platform perangkat lunak android dikembangkan oleh Google dalam *Open Handset Alliance,* sebuah aliansi bisnis yang didirikan pada novermber 2007 sebesar 34 mobile dan perusahaan teknologi dan sekarang yang melibatkan hamper 50 anggota, diantaranya internasional operator mobile (seperti T-Mobile, Telecom italia, telefonicam Vodafone, NTT DoCoMo) perusahaan perangkat lunak, (yang paling penting adalah google sendiri), komersialisasi perusahaan, pembuat chip (Intel, Broadcom, Nvidia, Texas Instruments, ARM, dll) dan handset produsen (Motorola, Samsung, LG, ASUSTekm Sony Ericsson,Toshiba dan HTC, yang pertama memulai sebuah "G-phone" ke pasar Amerika dan Inggris selama terakhir mengembangkan Google klien Android mobile untuk layanan webstudi kasus 28 musim gugur). Konsorsium ini diciptakan dengan tujuan utama mengembangkan standar terbuka untuk perangkat mobile untuk perangkat mobile untuk mempercepat inovasi dalam dunia perangkat mobile dan bersaing dengan perusahaan terkemuka seperti Apple (yang baru-baru ini meluncurkan iPhone), Microsoft (dengan windows mobile), Nokia dengan Symbian. Anggota OHA menggambarkannya cara ini: "Komitmen terhadap keterbukaan,visi bersama untuk masa depan, dan rencana konkret untuk membuat visi menjadi kenyataan. Untuk mempercepat inovasi dalam konsumen mobile dan menawarkan lebih kaya, lebih murah , dan pengalam yang lebih baik".

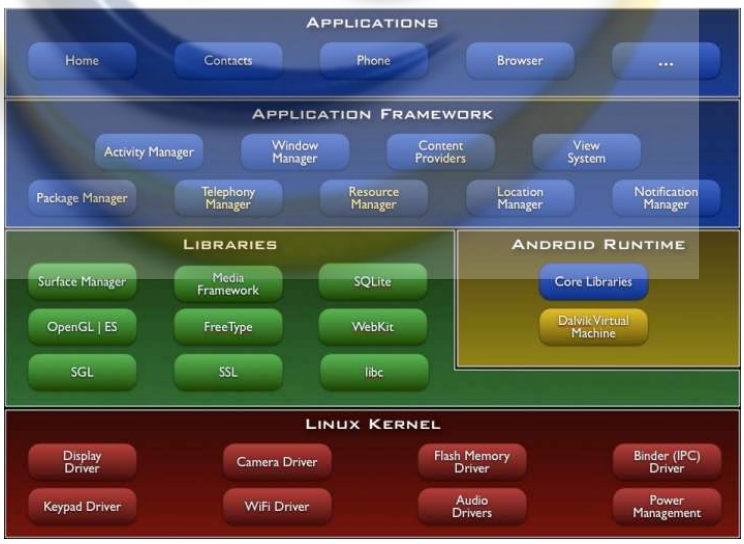
OHA berharap untuk memberikan software mobile yang lebih baik untuk konsumen dengan menyediakan platform yang dibutuhkan untuk pengembangan mobile inovasi pada tingkat yang lebih cepat dan lebih tinggi berkualitas tanpa biaya lisensi untuk pengembangan perangkat lunak atau produsen handset. Yang benar-benar membuat Android menarik adalah filosofi terbuka yang harus memastikan bahwa setiap kekurangan dalam antarmuka pengguna atau desain aplikasi asli dapat dengan mudah diperbaiki oleh masyarakat pengembang.

Google menyajikan Android sebagai “Platform yang benar-benar terbuka dan komprehensif pertama untuk perangkat mobile, dan semua perangkat lunak untuk menjalankan sebuah ponsel tapi tanpa menghambat dalam inovasi ponsel”

Membuka akses ke dalam rincian sistem yang mendasari mendorong pengembang perangkat lunak dan antar platform. Sejauh ini, itu tidak terjadi pada ponsel dan itulah salah satu alas an mengapa ada begitu sedikit aplikasi telepon selaluar yang baik dan lebih sedikit tersedia secara gratis dengan konsekuensi bahwa kebanyakan ponsel tetap hamper identik dengan diri merka sendiri selama bertahun-tahun. Sebaliknya, Android memungkinkan dan bahkan mendorong sebuah revoluasi dalam dunian ponsel. Itu memberikan pengembang berpeluang besar untuk mengubah cara konsumen dalam menggunakan ponsel mereka.

2.6.3 Arsitektur Android

Gambar berikut menunjukkan komponen utama dari sistem operasi Android. Setiap bagian dijelaskan lebih rinci pada bagian bawah.



Gambar 2.1 Arsitektur Android

1. *Applications*

   Applications adalah layer dimana kita berhubungan dengan aplikasi inti termasuk email *client*, program SMS, kalender, peta, browser, kontak, dan lain-lain. Semua aplikasi ditulis menggunakan bahasa pemrograman Java.

2. *Application Framework*

   Dengan menyediakan sebuah platform pengembangan terbuka, Android menawarkan kemampuan pengembang untuk membangun aplikasi yang sangat kaya dan inovatif. Pengembang bebas untuk mengambil keuntungan dari perangkat keras informasi, akses lokasi, menjalankan background *service*, mangatur alarm, menambahkan status notofikasi, dan sebagainya. Pengembang memiliki akses penuh menuju API *framework* seperti yang telah dilakukan oleh aplikasi inti. Arsitektur aplikasi dirancang supaya kita dengan mudah menggunakan komponen yang sudah digunakan (*re-use*).

3. *Library*

   Android mencakup satu set library C/ C++ yang digunakan oleh berbagai komponen dari sistem Android. *Library* tersebut dapat digunakan pengembang melalui *framework* Android. Beberapa library inti yang dapat digunakan yaitu:

   a. System C library – implementasi dari BSD (Berkeley Software Distribution) dari sistem standar library C (libc), diset untuk perangkat berbasis Linux.

   b. Media library – berbasis OpenCORE PacketVideo, dukungan library untuk pemutaran dan perekaman audio dan video populer seperti MPEG4, H.264, MP3, AAC, AMR, JPG, dan PNG.

   c. Surface manager – mengelola akses ke subsistem layar 2D dan 3D dari lapisan beberapa aplikasi.

   d. LibWebCore – mencakup modern web browser dengan engine embedded web view.

e. SGL – mendasari mesin grafis 2D.

f. 3D libraries – yang mencakup implementasi OpenGL ES 1.0 API's

g. FreeType – me-render huruf bitmap dan vector.

h. SQLite – mesin database relational yang powerful dan ringan, tersedia untuk semua aplikasi.

4. Android *Run Time*

Layer yang membuat aplikasi Android dapat dijalankan dimana dalam prosesnya menggunakan implemantasi Linux. Dalvik *Virtual Machine* (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi Android. Di dalam Android Run Time dibagi menjadi beberapa bagian, yaitu:

a. Core Libraries: Aplikasi Android dibangun dalam bahasa java, sementara Dalvik sebagai *virtual* mesinnya. Bukan *virtual machine* java (JVM), sehingga dibutuhkan *library* yang berfungsi untuk menterjemahkan bahasa java/c yang ditangani oleh Core Libraries.

b. Dalvik *Virtual Machine*: Virtual mesin berbasis register yang dioptimalkan untuk menjalankan fungsi-fungsi secara efisien, dimana merupakan pengembangan yang mampu membuat Linux kernel untuk melakukan threading dan manajemen tingkat rendah.

5. Linux Kernel

Linux kernel adalah layer dimana inti dari operating sistem dari Android itu berada. Berisi file-file sistem yang mengatur sistem *processing*, *memory*, *resource*, *drivers*,dansistem-sistem operasi Android lainnya. Android bergantung pada Linux kernel versi 2.6.

2.6.4 Fundamental Aplikasi Android

Aplikasi Android ditulis dalam bahasa pemrograman Java. Kode Java dikompilasi bersama dengan data file resource yang dibutuhkan oleh aplikasi, dimana prosesnya di-package oleh tools yang dinamakan “apt tools” ke dalam paket Android sehingga menghasilkan file dengan ekstensi apk. File apk tersebut yang dinamakan aplikasi yang dapat diinstal pada perangkat mobile. Ada empat jenis komponen pada aplikasi Android, yaitu:

1. Activity

   Suatu activity akan menyajikan user interface (UI) kepada pengguna, sehingga pengguna dapat melakukan interaksi. Sebuah aplikasi Android bisa jadi hanya memiliki satu activity tergantung pada tujuan aplikasi.

2. Service

   Service tidak memiliki Graphical User Interface (GUI), tetapi service berjalan secara background.

3. Content Providers

   Content provider membuat kumpulan aplikasi data secara spesifik sehingga bisa digunakan oleh aplikasi lain. Data disimpan dalam file sistem seperti database SQLite. Content provider menyediakan cara untuk mengakses data yang dibutuhkan oleh suatu activity.

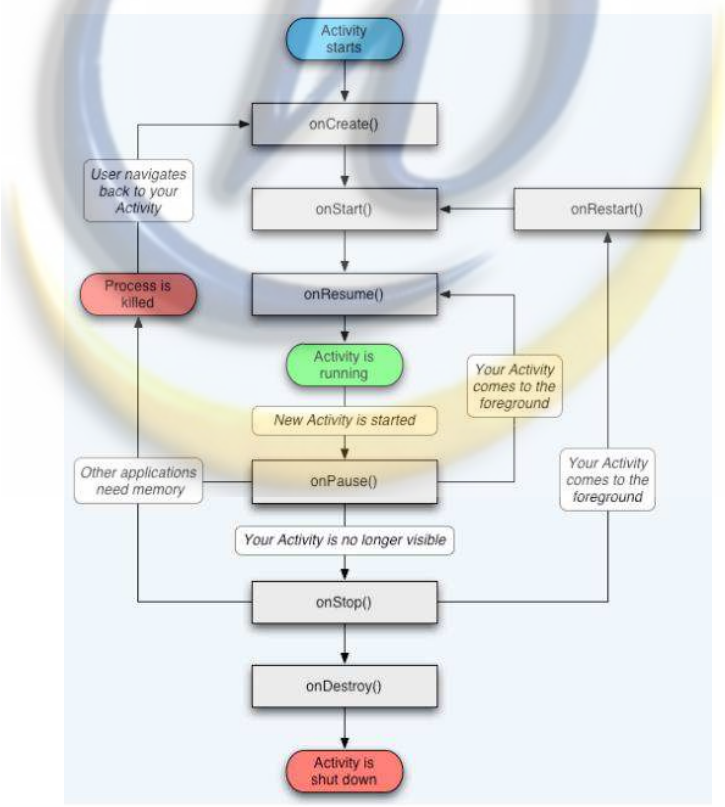
4. Broadcast Receiver

   Broadcast receiver berfungsi menerima dan bereaksi untuk menyampaikan notifikasi zona waktu berubah, baterai low, gambar telah selesai diambil oleh camera, atau pengubahan referensi bahasa yang digunakan.

2.6.5 Pengembangan Android

Apa yang terjadi di dalam aplikasi Android ini terutama dibagi menjadi dua, termasuk bagian visual, dimana pengguna berinteraksi dengan dan non-visual, yang berjalan di latar belakang. Bagian visual disebut kegiatan. Suatu kegiatan biasanya satu layar yang pengguna melihat pada perangkat pada satu waktu. Sebuah aplikasi biasanya memiliki 13 beberapa kegiatan, dan pengguna membalik bolak-balik di antara mereka. Kegiatan sebelumnya mungkin berhenti atau hancur dan yang baru mungkin baru dibuat atau hanya dilanjutkan. Gambar 2.2 di bawah ini menunjukkankegiatan *life cycle* dan *state-state* yang berbeda keluar masuk.

Gambar berikut ini merupakan *life cycle* dalam pengembangan android :



Gambar 2.2 *Activity Life Cycle*pada Android

Bagian yang tidak terlihat disebut *service.* Mereka berjalan di belakang layar dan tidak memiliki komponen antarmuka pengguna apapun,. Mereka melakukan tindakan seperti polling data dari server di internet, bermain music di belakang layar dll. mereka bertangung jawab untuk tindakan-tindakan yang harus terus berjalan sementara pengguna melakukan kegiatan lain, baik dari aplikasi saya sama ataupun dari aplikasi yg berbeda. *BroadcastReceiver* juga merupakan komponen penting dari pengembangan Android. Komponen ini adalah cara kerja sistem aplikasi memperingatkan peristiwa tertentu dalam sistem ketika saat itu terjadi. Sebagai contoh, saat menerima kedatangan SMS, ketika SMS baru tiba di *handphone* . maka selanjutnya semua aplikasi yang terdaftar untuk menerima kegiatan ini akan diberitahu yang kemudian masing-masing akan merespon yang dan menyesuaikan.

2.6.6 SQLite pada Android

Dengan semakin beragamnya pengembangan berbagai aplikasi di platform Android dewasa ini, beberapa aplikasi mulai berkembang semakin kompleks dan membutuhkan media penyimpanan informasi dalam bentuk database terstruktur. Para programmer Android sangat beruntung karena Android menyediakan sebuah database yang secara default sudah ada di dalam library Android, yaitu SQLite. Untuk keperluan operasi database pada smartphone atau tablet Android, SQLite sangat memadai karena ukurannya yang kecil, cepat dan ringan dalam hal sumber daya. Karena sifatnya sebagai embedded database, SQLite tidak memiliki server namun bentuknya adalah library yang akan dipanggil saat program dijalankan.

Seperti halnya database pada umumnya, SQLite memiliki objek-objek seperti table, view dan index. Perintah-perintah SQL-nya pun sangat mirip seperti yang biasa digunakan yaitu SELECT, INSERT, UPDATE, DELETE dan sebagainya. Informasi mengenai SQLite dapat dibaca secara lengkap pada situs resminya di:

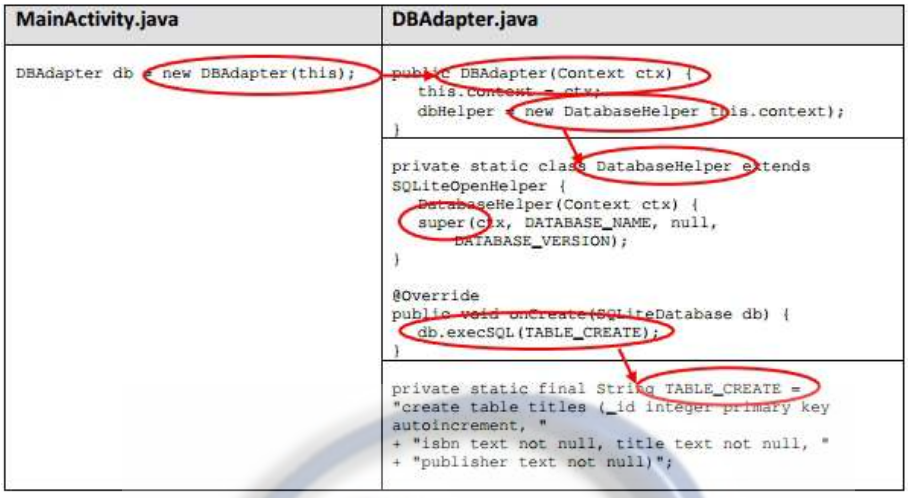
http://www.sqlite.org

SQLite adalah *Database Open Source* yang tertanam ke Android. SQLight mendukung fitur database relasional standar seperti sintaks SQL, *transactions* dan *prepared statements*. Selain itu hanya memerlukan sedikit memori pada saat runtime (sekitar 250 KByte).Menggunakan SQLite di Android tidak memerlukan *setup database* atau administrasi. Kita menentukan SQL untuk bekerja dengan database dan database secara otomatis dikelola untuk kita.

Bekerja dengan database di Android bisa lambat karena diperlukan I / O. Oleh karena itu dianjurkan untuk melakukan work ini dalam sebuah AsyncTask.SQLite mendukung tipe data TEXT (mirip dengan String di Java), INTEGER (mirip dengan yang lama di Java) dan REAL (mirip dengan ganda di Java). Semua jenis lain harus dikonversi ke pada bidang ini sebelum menyimpannya dalam database. SQLight sendiri tidak memvalidasi apakah jenis ditulis ke kolom sebenarnya dari jenis didefinisikan, Kita dapat menulis sebuah integer ke dalam kolom string.

Jika aplikasi Anda menciptakan database database ini disimpan di direktori "DATA / data / APP_NAME / database / FILENAME". "DATA" adalah jalan yang Environment.getDataDirectory () mengembalikan, "APP_NAME" adalah nama aplikasi Anda dan "FILENAME" adalah nama yang Anda berikan pada saat pembuatan database. Environment.getDataDirectory () biasanya kembali SD card sebagai lokasi.

Database SQLite adalah swasta untuk aplikasi yang menciptakan itu. Jika Anda ingin berbagi data dengan aplikasi lain Anda dapat menggunakan Content Provider.



MainActivity.java

```
DBAdapter db = new DBAdapter(this);
```

DBAdapter.java

```
public DBAdapter(Context ctx) {
    this.context = ctx;
    dbHelper = new DatabaseHelper(this.context);
}

private static class DatabaseHelper extends
SQLiteOpenHelper {
    DatabaseHelper(Context ctx) {
    super(ctx, DATABASE_NAME, null,
        DATABASE_VERSION);
}

@Override
public void onCreate(SQLiteDatabase db) {
    db.execSQL(TABLE_CREATE);
}

private static final String TABLE_CREATE =
"create table titles (_id integer primary key
autoincrement, "
+ "isbn text not null, title text not null, "
+ "publisher text not null)";
```

Gambar 2.3 Contoh SQLite pada Android

1. SQLightDatabase dan cursor

"SQLightDatabase" menyediakan metode masukkan (), update () dan menghapus () dan metode execSQL () yang memungkinkan untuk mengeksekusi SQL secara langsung. Objek "ContentValues" memungkinkan untuk mendefinisikan kunci / nilai untuk insert dan update. Kuncinya adalah kolom dan nilai tersebut adalah nilai untuk kolom ini.

Query dapat dibuat melalui rawQuery metode () yang menerima SQL atau query () yang menyediakan sebuah antarmuka untuk menentukan data yang dinamis atau SQLightQueryBuilder. SQLightBuilder mirip dengan antarmuka dari penyedia konten oleh karena itu biasanya digunakan dalam ContentProviders. Sebuah permintaan kembali selalu "cursor".

Query metode memiliki parameter String dbname, int [] columnNames, String whereClause, String [] valuesForWhereClause, String [] groupBy, String [] memiliki, String [] OrderBy. Jika semua data harus dipilih, Anda bisa lulus "nol" sebagai klausa mana. Dimana klausa yang ditentukan tanpa "mana", misalnya "_id = 19 dan ringkasan =?". Jika beberapa nilai yang diperlukan melalui? Anda lulus dalam array

valuesForWhereClause untuk query. Secara umum, jika ada sesuatu yang tidak diperlukan Anda bisa lulus “nol”, misalnya untuk kelompok dengan klausa.

Cursor mewakili hasil query. Untuk mendapatkan jumlah elemen menggunakan metode getCount (). Untuk beralih di antara baris data individu, Anda dapat menggunakan metode moveToFirst () dan moveToNext (). Melalui metode isAfterLast () Anda dapat memeriksa apakah masih ada beberapa data.

2.7 Android SDK dan Eclipse

SDK Android menyediakan *tools* dan API diperlukan untuk mulai pengembangan aplikasi pada platform Android dengan menggunakan bahasa pemrograman Java. SDK ini tersedia pada Windows, Linux dan Mac.

Eclipse adalah multi-platform yang mengembangkan environment yang berjalan pada semua sistem operasi. Salah satunya Android menyediakan *plug-in* untuk Eclipse yang memungkinkan penggunaan mudah dan mengkontrol dari fasilitas SDK Android. *Plug-in* ini disebut ADT *plug-in* (Android Developemnt Tool). Setelah menginstal ADT *plug-in* dibuat untuk menunjuk ke lokasi SDK setelah itu sebagian besar *tool* SDK kemudian dapat diakses melalui Ecplipse. Banyaknya IDE lainnya yang telah ada untuk pembangunan Android, ternyata Eclipse lebih disukai karena mendukung e*nvironment* yang resmi.

1. *Setup*

   Pada fase ini dilakukan pengaturan lingkungan pengembangan. Pada fase ini juga dilakukan pembuatan Android *Virtual Devices* (AVD) dan menghubungkan perangkat keras tempat dimana dapat menginstal Aplikasi yang akan dikembangkan.

2. *Development*

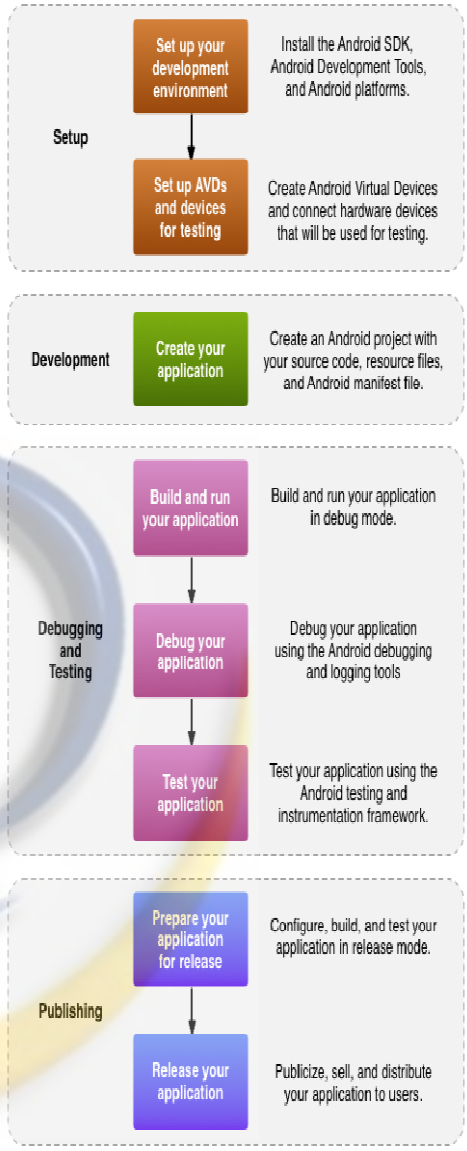
   Selama fase ini dilakukan pengaturan dan pengengembangan sebuah proyek Android, yang berisi semua kode sumber dan file sumber daya untuk aplikasi yang dikembangkan.

3. *Debugging and Testing*

   Pada fase ini, dilakukan pembangunan project ke dalam paket .apk. Pada fase ini dapat menginstal aplikasi yang dikambangkan untuk berjalan pada *emulator* atau perangkat Android.

4. *Publishing*

   Fase ini mengkonfigurasi aplikasi yang telah dibangun untuk dirilis dan didistribusikan kepada pengguna melalui Android Market



Gambar 2.4 Proses Pengembangan Aplikasi Android

2.7.1 Definisi ADB *(Android Debug Bridge)*

Android Debug Bridge (adb) adalah alat serbaguna yang memungkinkan mengelola keadaan sebuah contoh emulator atau perangkat Android. Ini merupakan bagian dari alat yang dibundel dengan Android SDK. Maka dengan itu seseorang dapat mengeluarkan perintah ke emulator atau perangkat dan memeriksa atau

memodifikasi *internal states*. Tabel 2.2 menunjukan perintah adb yang umum digunakan selama pengembangan proyek ini

Tabel 2.2 *Commonly used adb commands*

Perintah	Deskripsi
Devices	*Print a list of all attachedemulator/device instances*
Shell	*Starts a remote shell in the target emulator/device instance*
Logcat	*Print log data to screen*
Push <local><remote>	*Copies a speciefied file from your development computer to an emulator/device instance*
Pull <remote><local>	*Copes a speciefied file from emulator/device instance to your development computer*

Beberapa fungsi bari perintah Linux juga dapat diakses melalui adb. Pertama-ta adb *shell* haarus dijalankan untuk login, setelah itu perintah seperti ps, ls dan top dapat dijalankan. Sementara Eclipse *plug-in* menyediakan GUI *front-end* untuk sebagian besar fungsionalitas adb, baris perintah dengan baik cocok atau kadang-kadang diperlukan untuk mengakses perangkat atau emulator. Inovasi *Command-line* juga disukai, saat Eclipse *plug-in* kehilangan komunikasi dengan *server emulator*.

Selama pengembangan proyek ini, command-line telah menjadi sesuatu yang sering digunakan dalam melihat database, melihat log sistem dan mengelola lebih dari satu emulator pada satu waktu. Melihat database dilakukan dengan membangkitkan perintah SQLite3 dengan path lengkap ke lokasi database. Gambar 2.4 dibawah ini menunjukan permintaan dari shell dan perintah logcat.

```
lawal@lawal-ubuntu:~$ adb shell
# sqlite3 /data/data/com.tracker/databases/itracker
SQLite version 3.6.22
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> .tables
android_metadata  assets       assettypes      data
sqlite>


lawal@lawal-ubuntu:~$ adb logcat | grep System.out
I/System.out(  324): DeleteActivity.onClick() null
I/System.out(  324): MainMapActivity.onActivityResult() 44 -1
Intent { (has extras) }
I/System.out(  324): MainMapActivity.onRestart()
I/System.out(  324): MainMapActivity.removeLastPosMarker()
I/System.out(  324): MainMapActivity.onResume()
I/System.out(  324): MainMapActivity.onPause()
I/System.out(  324): MainMapActivity.onDestroy()
```

Gambar 2.5 adb *command line*

2.7.2 Definisi Android API Levels

Ketika mengembangkan aplikasi Android API Level berguna untuk memahami pendekatan umum platform untuk manajemen perubahan API, selain itu untuk memastikan kompatibilitas aplikasi yang akan dikembangkan dengan perangkat yang mungkin akan diinstal. Dibawah ini akan memberikan penjelasan tentang API Level dan bagaimana hal tersebut mempengaruhi aplikasi yang akan dikembangkan.

2.7.3 Apa itu API Level?

API level adalah nilai integer yang secara unik mengidentifikasi revisi framework API yang ditawarkan oleh versi dari platform Android. Platform Android menyediakan framework API yang dapat digunakan untuk berinteraksi dengan sistem Android yang mendasarinya. Framework API terdiri dari:

1. Satu set inti dari paket dan kelas.
2. Satu set elemen XML dan atribut untuk menyatakan file manifest.

3. Satu set elemen XML dan atribut untuk menyatakan dan mengakses sumber daya.
4. Satu set Intens.
5. Satu set permission bahwa aplikasi dapat meminta izin termasuk izin ke dalam sistem.

Setiap versi terbaru dari platform Android dapat mencakup update untuk aplikasi framework API sebelumnya. Pembaharuan framework API dirancang sehingga API baru tetap kompatibel dengan versi sebelumnya, artinya perubahan dalam API adalah aditif dan memperkenalkan fungsionalitas baru atau penggantian. Sebagai bagian dari API yang ditingkatkan, diganti bagian yang lebih usang tapi tidak dihapus, sehingga aplikasi yang ada masih bisa menggunakannya. Dalam sejumlah kecil kasus, bagian dari API dapat dimodifikasi atau dihapus, meskipun perubahan tersebut biasanya hanya diperlukan untuk memastikan ketahanan API dan kemanan sistem. Semua bagian dari revisi API lainnya diteruskan tanpa modifikasi.

Framework API platform Android memberikan identifier bilangan bulat yang disebut API Level. Setiap versi platform Android mendukung tepat satu level API, meskipun dukungan implisit untuk semua tingkat API sebelumnya. Rilis awal platform Android yang disediakan API Level 1 dan rilis berikutnya telah bertambah level API. Tabel berikut menetapkan API Level yang didukung oleh setiap versi platform Android.

Tabel 2.3 Versi Platform Android

Versi Platform	API Level	Code Versi
Android 4.0	14	Ice Cream Sandwich
Android 3.2	13	Honeycomb MR2
Android 3.1.x	12	Honeycomb MR1

Versi Platform	API Level	Code Versi
Android 3.0.x	11	Honeycomb
Android 2.3.6 Android 2.3.5 Android 2.3.4 Android 2.3.3	10	Gingerbread MR1
Android 2.3	9	Gingerbread
Android 2.2.x	8	Froyo
Android 2.1.x	7	Éclair MR1
Android 2.0.1	6	Éclair 0 1
Android 2.0	5	Eclair
Android 1.6	4	Donut
Android 1.5	3	Cupcake
Android 1.1	2	Base 1 1
Android 1.0	1	Base

2.7.4 Struktur *Project* Android

Ketika membuat sebuah project Android, secara otomatis akan terbentuk file-file sebagai berikut:

1. AndroidManifest.xml, sebuah file XML yang menjelaskan aplikasi yang sedang dibangun serta menjelaskan tentang component, activities, services, dll yang disediakan oleh aplikasi.
2. default.properties, file properti yang digunakan oleh script.

3. libs/ folder berisi beberapa third-party Java JARs yang Anda butuhkan.
4. gen/ folder berisi kode yang digenerator oleh project sendiri pada saat kita menambahkan resources untuk aplikasi.
5. src/ folder berisi java source code untuk aplikasi.
6. res/ folder berisi resources seperti icons, GUI layout, dan sejenisnya dalam sebuah package.
7. asset/ berisi file statis yang berfungsi untuk mengemas aplikasi ketika ingin disebarkan (deployment).

2.8 Memulai Pembangunan Android di Eclipse

Pembangunan aplikasi Android dimulai dengan menginstall Android SDK, menambahkan plugin ADT pada Eclips, serta mengupdate plugin ADT. Selanjutnya dijelaskan pada subbab berikut:

2.8.1 Menginstall Android SDK

Versi terakhir Android SDK dapat didownload pada:

http://code.google.com/android/download.html

Pada link tersebut tersedia versi .zip maupun .exe yang dapat diinstall dengan mudah setelah didownload.

2.8.2 Android Development Tools (ADT)

Android menyediakan sebuah plugin Eclips yang disebut ADT untuk membuat pemrograman dan debugging menjadi lebih mudah. ADT menyediakan akses mudah terhadap LogCat, Android-Manifest/Resource-Editor, File, Thread, dan Heap Control. Berikut cara menginstall Eclips Plugin (ADT):

1. Start Eclips, kemudian pilih **Help > Software Update > Find and Install…**.

2. Pada dialog yang muncul, pilih **Search for new features to install** kemudian pilih **Next**.
3. Tekan **New Remote Site**.
4. Pada saat muncul kotak pencarian, masukkan remote site (dalam hal ini Android plugin) dan masukkan url berikut:

   https://dl-ssl.google.com/android/eclipse/

   Tekan **OK**.
5. Anda sekarang harus melihat situs baru ditambahkan ke daftar pencarian (dan telah dicentang). Tekan **Finish**.
6. Pada kotak dialog hasil pencarian berikutnya, pilih centang untuk Android **Plugin** > **Developer Tools**. Hal ini akan memeriksa kedua fitur: "Android Developer Tools", dan "Android Editor". Android Editor fitur adalah opsional, tetapi dianjurkan. Kemudian tekan **Next**.
7. Baca perjanjian lisensi dan kemudian pilih **Accept terms of the license agrement**, jika sesuai tekan **Next**.
8. Tekan **Finish**.
9. Anda dapat menerima instalasi dengan menekan **Install All**.
10. Restart Eclips.
11. Setelah restart, **update your Eclips preferences**. Untuk menunjuk ke direktori SDK:
    a. Pilih **Window > Preference**… untuk membuka panel Preference.
    b. Pilih **Android** dari panel kiri.
    c. Untuk lokasi SDK pada panel utama, tekan **Browse**… dan cari direktori SDK.

d. Tekan **Apply** kemudian **OK**.

2.8.3 Meng-*update* Plugin ADT

Mengupdate plugin ADT mengikuti prosedur standar upgrade plugin Eclips umum, yaitu:

a. Pilih **Help > Software Update > Find and Install…**.

b. Pilih **Search for updates of the currently installed features** dan tekan **Finish**.

c. Jika ada update untuk ADT tersedia, pilih dan instal.

2.9 Definisi Web Service

Web service adalah suatu sistem perangkat lunak yang dirancang untuk mendukung interoperabilitas dan interaksi antar sistem pada suatu jaringan. Web service digunakan sebagai suatu fasilitas yang disediakan oleh suatu web site untuk menyediakan layanan (dalam bentuk informasi) kepada sistem lain, sehingga sistem lain dapat berinteraksi dengan sistem tersebut melalui layanan-layanan (service) yang disediakan oleh suatu sistem yang menyediakan web service. Web service menyimpan data informasi dalam format XML, sehingga data ini dapat diakses oleh sistem lain walaupun berbeda platform, sistem operasi, maupun bahasa compiler. Web service bertujuan untuk meningkatkan kolaborasi antar pemrogram dan perusahaan, yang memungkinkan sebuah fungsi di dalam Web Service dapat dipinjam oleh aplikasi lain tanpa perlu mengetahui detil pemrograman yang terdapat di dalamnya. Beberapa alasan mengapa digunakannya web service adalah sebagai berikut:

1. Web service dapat digunakan untuk mentransformasikan satu atau beberapa bisnis logic atau class dan objek yang terpisah dalam satu ruang lingkup yang menjadi satu, sehingga tingkat keamanan dapat ditangani dengan baik.

2. Web service memiliki kemudahan dalam proses deployment-nya, karena tidak memerlukan registrasi khusus ke dalam suatu sistem operasi. Web service cukup di- upload ke web server dan siap diakses oleh pihak-pihak yang telah diberikan otorissi.

3. Web service berjalan di port 80 yang merupakan protokol standar HTTP, dengan demikian web service tidak memerlukan konfigurasi khusus di sisi firewall.

2.9.1 Arsitektur *Web Service*

Web service memiliki tiga entitas dalam arsitekturnya, yaitu:

1. *Service Requester* (peminta layanan)

   Peminta layanan yang mencari dan menemukan layanan yang dibutuhkan serta menggunakan layanan tersebut.

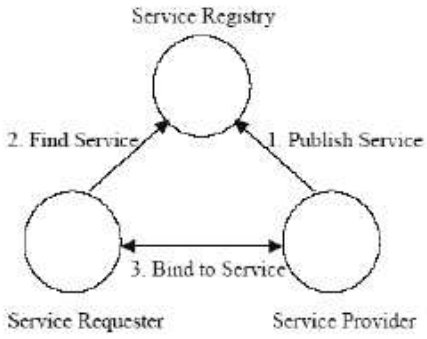
2. *Service Provider* (penyedia layanan)

   Berfungsi untuk menyediakan layanan/*service* dan mengolah sebuah registry agar layanan-layanan tersebut dapat tersedia.

3. *Service Registry* (daftar layanan)

   Berfungsi sebagai lokasi central yang mendeskripsikan semua layanan/*service* yang telah di-register.

   Pada gambar disamping ini merupakan arsitektur *web service.*



Gambar 2.6 Arsitektur Webservice

2.9.2 Operasi Web Service

Secara umum, web service memiliki tiga operasi yang terlibat di dalamnya, yaitu:

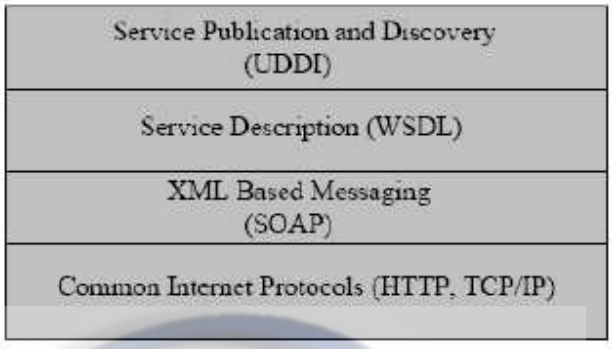
1. Publish/Unpublish : Menerbitkan/menghapus layanan ke dalam atau dari registry.
2. Find : Service requestor mencari dan menemukan layanan yang dibutuhkan.
3. Bind : Service requestor setelah menemukan layanan yang dicarinya, kemudian melakukan binding ke service provider untuk melakukan interaksi dan mengakses layanan/service yang disediakan oleh service provider.

2.9.3 Komponen Web Service

Web service secara keseluruhan memiliki empat layer komponen seperti pada gambar di bawah, yaitu:

1. Layer 1 : Protokol internet standar seperti HTTP, TCP/IP
2. Layer 2 : *Simple Object Access Protocol* (SOAP), merupakan protokol akses objek berbasis XML yang digunakan untuk proses pertukaran data/informasi antar layanan.
3. Layer 3 : *Web Service Definition Language* (WSDL), merupakan suatu standar bahasa dalam format XML yang berfungsi untuk mendeskripsikan seluruh layanan yang tersedia.

Pada gambar dibawah ini merupakan komponen *web service.*



Gambar 2.7 Komponen *WebService*

2.10 HyperText Transfer Protocol (HTTP)

HyperText Transfer Protocol (HTTP) adalah sebuah protokol jaringan lapisan aplikasi yang digunakan untuk sistem informasi terdistribusi, kolaboratif, dan menggunakan hipermedia. Penggunaannya banyak pada pengambilan sumber daya yang saling terhubung dengan tautan, yang disebut dengan dokumen hiperteks, yang kemudian membentuk *World Wide Web* pada tahun 1990oleh fisikawan Inggris, Tim Berners-Lee. Hingga kini, ada dua versi mayor dari protokol HTTP, yakni HTTP/1.0 yang menggunakan koneksi terpisah untuk setiap dokumen, dan HTTP/1.1 yang dapat menggunakan koneksi yang sama untuk melakukan transaksi. Dengan demikian, HTTP/1.1 bisa lebih cepat karena memang tidak perlu membuang waktu untuk pembuatan koneksi berulang-ulang..

HTTP adalah sebuah protokol meminta/menjawab antara klien dan *server*. Sebuah klien HTTP (seperti *web browser* atau robot dan lain sebagainya), biasanya memulai permintaan dengan membuat hubungan ke port tertentu di sebuah *server Webhosting* tertentu (biasanya *port* 80). Klien yang mengirimkan permintaan HTTP juga dikenal dengan *user agent*. *Server* yang meresponnya, yang menyimpan sumber daya seperti berkas HTML dan gambar, dikenal juga sebagai *origin server*.

Di antara *user agent* dan juga *origin server*, bisa saja ada penghubung, seperti halnya *proxy*, *gateway*, dan juga *tunnel*.

HTTP tidaklah terbatas untuk penggunaan dengan TCP/IP, meskipun HTTP merupakan salah satu protokol aplikasi TCP/IP paling populer melalui Internet. Memang HTTP dapat diimplementasikan di atas protokol yang lain di atas Internet atau di atas jaringan lainnya. seperti disebutkan dalam *"implemented on top of any other protocol on the Internet, or on other networks."*, tapi HTTP membutuhkan sebuah protokol lapisan *transport* yang dapat diandalkan. Protokol lainnya yang menyediakan layanan dan jaminan seperti itu juga dapat digunakan.

Sumber daya yang hendak diakses dengan menggunakan HTTP diidentifikasi dengan menggunakan *Uniform Resource Identifier* (URI), atau lebih khusus melalui *Uniform Resource Locator* (URL), menggunakan skema URI http: atau https:

HTTP menetapkan sembilan metode (kadang disebut *"verbs"*) yang menunjukkan tindakan yang ingin dilakukan terhadap sumber teridentifikasi. Hal yang diwakili sumber ini, berupa data yang sudah ada atau data yang diciptakan secara dinamis, bergantung pada implementasi peladen. Biasanya sumber ini berkaitan dengan berkas atau keluaran dari berkas pelaksana yang menetap di peladen.

a) HEAD

   Meminta tanggapan yang identik dengan tanggapan yang sesuai dengan permintaan GET, namun tanpa badan tanggapan. Ini berguna untuk mengakses informasi meta yang tertulis dalam kepala tanggapan tanpa perlu mengangkut seluruh konten.

b) GET

   Meminta representasi sumber tertentu. Permintaan menggunakan GET (dan beberapa metode HTTP lain) "tidak boleh memiliki kepentingan melakukan

tindakan selain pengaksesan".[3] W3C telah menerbitkan prinsip panduan mengenai perbedaan ini dengan menyatakan, "desain aplikasi web harus mematuhi prinsip di atas, serta batasan sejenis."[4]

c) POST

Mengirimkan data untuk diproses (misalnya dari bentuk HTML) ke sumber teridentifikasi. Data dimasukkan dalam badan permintaan. Ini dapat menghasilkan pembentukan sumber baru atau pemutakhiran sumber yang sudah ada atau keduanya.

d) PUT

Mengunggah representasi sumber tertentu.

e) DELETE

Menghapus sumber tertentu.

f) TRACE

Menggaungkan kembali permintaan yang diterima, sehingga klien dapat melihat perubahan atau tambahan yang dilakukan oleh peladen perantara.

g) OPTIONS

Mengembalikan metode HTTP yang didukung peladen untuk URL tertentu. Ini dapat digunakan untuk memeriksa fungsionalitas peladen web dengan meminta '*' daripada fungsionalitas sumber tertentu.

h) CONNECT

Menukarkan koneksi permintaan dengan terowongan TCP/IP transparan, biasanya untuk memfasilitasi komunikasi terenkripsi SSL(HTTPS) melalui proksi HTTP tak terenkripsi.[5]

i) PATCH

Menerapkan modifikasi parsial terhadap sumber.[6]

2.10.1 *Hypertext Preprocessor* (PHP)

PHP (akronim dari PHP *Hypertext Preprocessor*) yang merupakan bahasa pemrograman berbasis web yang memiliki kemampuan untuk memproses data dinamis.

PHP dikatakan sebagai sebuah *server-side embedded script language* artinya sintaks-sintaks dan perintah yang kita berikan akan sepenuhnya dijalani oleh *server* tetapi disertakan pada halaman HTML biasa. Aplikasi-aplikasi yang dibangun oleh PHP pada umumnya akan memberikan hasil pada *web browser*, tetapi prosesnya secara keseluruhan dijalankan oleh *server*.

Pada prinsipnya *server* akan bekerja apabila ada permintaan dari *client*. Dalam hal ini *client* menggunakan kode-kode PHP untuk mengirimkan permintaan ke *server*. Ketika menggunakan PHP sebagai *server side embedded script language* maka *server* akan melakukan hal-hal sebagai berikut:

PHP memiliki kelebihan-kelebihan dibandingkan bahasa programming web lainnya. Kelebihan-kelebihan PHP antara lain :

1. Kemudahan sintax programming

   Salah satu tujuan programming *web* adalah menghasilkan kode-kode html. Secara teknis kode-kode mempunyai tipe string. Dengan demikian kita akan banyak berhubungan variable string.Berkaitan dengan variable string ini, menggabungkan string paling mudah dilakukan oleh PHP.Misal ada variabel $company dengan isi 'proweb indonesia'.Kemudian ada variable $place yang diisi dengan 'jakarta'. Kita ingin menampilkan gabungan antara 'perusahaan ' dan$company dan "di " dan $place.Dengan bahasa programming web lain misalnya ASP sintaxnya akan seperti*"perusahaaan" & company & "di "*

*&place*Bandingkan dengan sintax PHP *"perusahaan $company di $place"*.Dengan demikian sintax php sangat mudah dikerjakan dan dimengerti

2. Dapat dijalankan di berbagai sistem operasi(*operating system*)

   PHP dapat dijalankan di berbagai platform seperti Windows, Linux dan Unix.Dengan demikian programmer tidak perlu memikirkan di mana programnya akan diinstall karena php bisa dijalankan di banyak platform

3. Dokumentasnya mudah, lengkap dam sederhana

   Manual PHP dengan mudah didownload di situsnya yaitu www.php.net dan ujurannya hanya beberapa mega bytes saja. Bandingkan dengan ASP yang dokumentasinya bisa lebih dari 3 cd dan tentu akan sangat merepotkan.

4. Fungsi-fungsi yang lengkap

   Fungsi-funsinyanya sangat lengkap termasuk dukungan/support terhadap OOP (*Object Oriented Programming*). Dengan support terhadap OOP ini melahirkan framework-framework PHP seperti Code Igniter, Cakephp, Yii dan lain-lain. PHP juga mendukung banyak database seperty MySQL. MSSql, Oracle dan lain-lain

2.10.2 *JavaScript Object Notation* (JSON)

JSON (*JavaScript Object Notation)* adalah standar terbuka berbasis text yang ringan dan dirancang untuk pertukaran data yang bersifat *human-readable*. JSON berasal dari bahasa pemrograman Javascript untuk mempresentasikan struktur data sederhana dan array asosiatif yang disebut dengan objek. Walaupun hubungannya dengan javascript nyatanya JSON adalah independen dengan parser yang tersedia untuk hampir semua bahasa pemrograman C, C++, C#, Java, JavaScript, Perl, Python. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaan-data.

JSON terdiri dari dua struktur:

- 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 (*keyed list*), atau *associative array*.
- 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 moderen 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.

Pada gambar dibawah ini merupakan contoh dalam penulisan JSON.

```
{
    "namaDepan":"Edy",
    "namaBelakang":"Vanhoten",
    "umur":95,
    "alamat":
    {
        "jalan":"Jalan Canjadi No 23",
        "kota":"Bandung",
        "propinsi":"Jawa Barat",
        "kodePos":"45363"
    },
    "telepon":
    [
        {
            "jenis":"rumah",
            "nomor":"022-7123456"
        },
        {
            "jenis":"mobile",
            "nomor":"081234567"
        }
    ],
    "hobi":"tidur"
}
```

Gambar 2.8 Contoh JSON

2.10.3 *Extensible Markup Language*(XML*)*

XML adalah bahasa markup untuk keperluan umum yang disarankan oleh W3C (*World Wide Web Consortium*) untuk membuat dokumen markup keperluan pertukaran data antar sistem yang beraneka ragam. XML merupakan kelanjutan dari HTML yang merupakan bahasa standar untuk Internet. Keunggulan XML:

1. Pintar. XML dapat menangani berbagai tingkat kompleksitas.
2. Dapat beradaptasi. Dapat mengadaptasi untuk membuat bahasa sendiri, seperti Microsoft membuat MSXML atau Macromedia mengembangkan MXML.
3. Sederhana.
4. Mudah dipindahkan. XML mempunyai kemudahan perpindahan yang bagus.

```
<myuser>
<user>
<msisdn>628111111111</msisdn>
<account>13435140</account>
</user>
<user>
<msisdn>628222222222</msisdn>
<account>13102743</account>
</user>
<user>
<msisdn>628333333333</msisdn>
<account>13361518</account>
</user>
</myuser>
```

Gambar 2.9 Contoh XML

2.11 MySQL

MySQL adalah sebuah implementasi dari sistem manajemen basisdata relasional (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public License*). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basisdata yang telah ada sebelumnya; SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian basisdata, terutama untuk pemilihan atau seleksi

dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Kehandalan suatu sistem basisdata (DBMS) dapat diketahui dari cara kerja pengoptimasi-nya dalam melakukan proses perintah-perintah SQL yang dibuat oleh pengguna maupun program-program aplikasi yang memanfaatkannya. Sebagai peladen basis data, MySQL mendukung operasi basisdata transaksional maupun operasi basisdata non-transaksional. Pada modus operasi non-transaksional, MySQL dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak peladen basisdata kompetitor lainnya. Namun demikian pada modus non-transaksional tidak ada jaminan atas reliabilitas terhadap data yang tersimpan, karenanya modus non-transaksional hanya cocok untuk jenis aplikasi yang tidak membutuhkan reliabilitas data seperti aplikasi blogging berbasis web (Wordpress), CMS, dan sejenisnya. Untuk kebutuhan sistem yang ditujukan untuk bisnis sangat disarankan untuk menggunakan modus basisdata transaksional, hanya saja sebagai konsekuensinya unjuk kerja MySQL pada modus transaksional tidak secepat unjuk kerja pada modus non-transaksional.

2.12 Google Map API

Google menyediakan layanan API (Application Programming Interface) memungkinkan aplikasi client untuk melihat, menyimpan dan memperbarui data peta dalam bentuk data API Google feed dengan menggunakan model data fitur (letak, garis dan bentuk) dalam peta. Aplikasi ini diberi namaGoogle map API. Peta yang ditampilkan diambil dari layanan Google Maps. Ada tiga jenis tampilan yang bisa dipilih dari GoogleMaps, yaitu:

a) Map, menampilkan peta dalam bentuk peta garis

b) *Satelite*, menampilkan peta dalam bentuk citra/foto satelit

c) *Earth*, merupakan gabungan dari Map dan Sattelite

Menampilkan peta dan menentukan bagian peta yang ditampilkan. Peta yang diambil dari Google API akan menampilkan peta dengan titik koordinat (-6.861326 sampai 107.594862) serta memiliki zoom level 13.

A. Elemen-elemen yang terdapat pada Google Map API adalah :

- *Marker*, symbol yang mennadakan suatu lokasi bangunan sejarah pada peta yang ditampilkan Google Maps.
- *Polyline*, shape yang digunakan untuk menandakan suatu jalur, jalan atau area. *Polyline* terdiri dari kumpulan titik koordinat.

B. Objek model yang terdapat pada Google Map API adalah:

- Inisialisasi Map

  Inisialisasi diproses dengan menggunakan method setCenter(). Method setCenter() membutuhkan GetLatLng koordinat dan zoom level, dan method ini harus segera dikirim sebelum ada pengoperasian lain pada peta, termasuk seting atribut peta itu sendiri

- Loading Google Maps API

  Koneksi script yang kita buat keserver Google Map API dengan menggunakan key yang anda dapatkan pada saat anda mendaftar ke Google Maps API.

- GMap2 – Elementary Object

  Class javascript yang membuat peta itu adalah class GMap2, Object dari class ini akan menyediakan sebuah peta dihalaman web, Variable map akan berikan nilai sebuah object dari class GMap2. Fungsi GMap2 adalah sebagai constructor dan definisinya.

- Loading Map

Ketika halaman HTML di render, document object model (DOM) sudah bisa digunakan, dan semua gambar external dan script diterima oleh object document. Untuk memastikan peta kita dimuat sesudah halaman selesai dimuat oleh browser

- Latitude and Longitude

  Objek GLatLng menentukan titik koordinat peta yang akan ditampilkan, parameternya terdiri dari lintang/latitude dan bujur/longitude

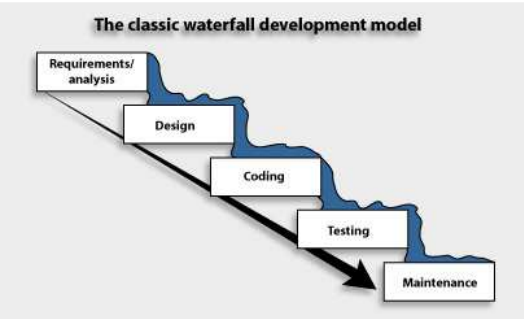
- Fungsi Gdirections

  Gdirection adalah utility yang menangani masalah pencarian rute optimal dan satu tempat ke tempat lain.

2.13 Perancangan Aplikasi dengan Metode *Waterfall*

Dalam perancangan aplikasi pada tugas akhir ini menggunakan metode *Waterfall*. Metode *Waterfall* adalah metode yang menyarankan sebuah pendekatanyang sistematis dan sekuensial melalui tahapan-tahapan yang ada pada SDLC untuk membangun sebuah perangkat lunak.

Gambar menjelaskan bahwa metode *Waterfall* menekankan pada sebuah keterurutan dalam proses pengembangan perangkat lunak. Metode ini adalah sebuah metode yang tepat untuk membangun sebuah perangkat lunak yang tidak terlalubesar dan sumber daya manusia yang terlibat dalam jumlah yang terbatas.



Gambar 2.10 Diagram Waterfall Model

Dari gambar 2.10 dapat dilihat bahwa tahapan pada metode *Waterfall* diawalioleh tahap analisis kebutuhan yang merupakan tahap awal pembangunan sebuahperangkat lunak. Tahap ini didefinisikan sebagai sebuah tahap yang menghasilkansebuah kondisi yang diperlukan oleh pengguna untuk menyelesaikan permasalahanataupun mencapai sebuah tujuan. Tahap ini bertujuan untuk mengumpulkan kebutuhan-kebutuhan pengguna dan kemudian mentransformasikan kedalam sebuah deskripsi yang jelas dan lengkap.

Tahapan kedua adalah tahap analisis sistem yang bertujuan untuk menjabarkan segala sesuatu yang nantinya akan ditangani oleh perangkat lunak. Tahapan ini adalah tahapan dimana pemodelan merupakan sebuah representasi dari object di dunia nyata. Untuk memahami sifat perangkat lunak yang akan dibangun, analis harus memahami domain informasi, dan tingkah laku yang diperlukan.

Tahap ketiga adalah tahap perancangan perangkat lunak yang merupakan proses multi langkah dan berfokus pada beberapa atribut perangkat lunak yang berbeda yaitu struktur data, arsitektur perangkat lunak, dan detil algoritma. Proses ini menerjemahkan kebutuhan ke dalam sebuah model perangkat lunak yang dapat diperkirakan kualitasnya sebelum dimulainya tahap implementasi.

Tahap implementasi adalah tahap yang mengkonversi apa yang telah dirancang sebelumnya ke dalam sebuah bahasa yang dimengerti komputer. Kemudian komputer akan menjalankan fungsi-fungsi yang telah didefinisikan sehingga mampu memberikan layanan-layanan kepada penggunanya.

Tahap selanjutnya adalah tahap pengujian. Terdapat dua metode pengujian perangkat lunak yang umum digunakan, yaitu metode black-box dan white-box.

- Pengujian dengan metode *black-box* merupakan pengujian yang menekankan pada fungsionalitas dari sebuah perangkat lunak tanpa harus mengetahui bagaimana struktur di dalam perangkat lunak tersebut. Sebuah perangkat lunak

yang diuji menggunakan metode black-box dikatakan berhasil jika fungsi-fungsi yang ada telah memenuhi spesifikasi kebutuhan yang telah dibuat sebelumnya.

- Sedangkan metode *white-box* menguji struktur internal perangkat lunak dengan melakukan pengujian pada algoritma yang digunakan oleh perangkat lunak. Tahap akhir dari metode Waterfall adalah tahap perawatan. Tahap ini dapat diartikan sebagai tahap penggunaan perangkat lunak yang disertai dengan perawatan dan perbaikan. Perawatan dan perbaikan suatu perangkat lunak diperlukan, termasuk di dalamnya adalah pengembangan, karena dalam prakteknya ketika perangkat lunak tersebut digunakan terkadang masih terdapat kekurangan ataupun penambahan fiturfitur baru yang dirasa perlu.

2.13.1 *Unified Modeling language (UML)*

UML merupakan bahasa modeling standar dalam *software* / *system development* yang berbasiskan konsep *object oriented*. UML lahir karena banyaknya *modeling language* sehingga sulit sekali untuk mengembangkan suatu desain *software* yang berbasis *object oriented*. Model yang satu dengan yang lain mempunyai standar notasi yang berbeda untuk maksud yang sama sehingga bisa membingungkan pada saat implementasi.

The objective of ***UML*** *is to provide system architects, software engineers and software developers with tools for the analysis, design and implementation of software-based systems, as well as for modeling business and similar processes.*

UML merupakan bahasa pemodelan yang memadukan tiga *modeling* terbesar saat itu, yaitu OMT, Booch, dan OOSE. Saat ini UML menjadi standar seluruh dunia dalam pemodelan sistem*/software* di bawah naungan Object Management Group (OMG), sebuah organisasi nirlaba yang mengawasi Ommon Object Request Broker Architecture (CORBA).

Karena UML merupakan bahasa pemodelan, maka untuk implementasinya tergantung dari metode dan kebutuhanya. Memang seringkali terjadi perselisihan persepsi dalam menggunakan UML, hal ini bisa dihindari dengan pemahaman UML dan menggunakan cara pandang (metode) yang sama. Beberapa contoh metode yang dapat digunakan antara lain IBM Rational Unified Process (RUP), Abstraction, Dynamic System Development, ICONIX Process, dan sebagainya.

Keuntungan menngunakan UML, antara lain :

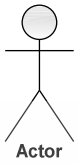
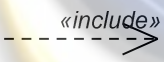
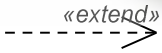
1. *Formal Languange*, memiliki definisi arti yang kuat sehingga membuat kita nyaman dalam memodelkan sistem karena mengurangi kesalahpahaman.
2. *Concise*, notasi yang sederhana dan mudah.
3. *Comprehensive*, menggambarkan semua aspek penting dari sistem.
4. *Scalable*, dapat disesuaikan dengan kebutuhan *project* besar atau kecil.
5. *Built on Lessons Learned*, puncak dari praktek-praktek terbaik komunitas *object oriented*.
6. *Standard*, dikontrol oleh grup *open stan*dard dengan kontribusi aktif dari vendor dan akademisi di seluruh dunia.

2.13.2 *Use Case Diagram*

Use Case adalah teknik untuk merekam fungsionalitas sebuah sistem. *Use case* mendeskripsikan interaksi tipikal antara para pengguna sistem dengan sistem itu sendiri dengan memberi sebuah narasi tentang bagaimana sistem tersebut digunakan.

Berikut ini merupakan simbol-simbol yang digunakan dalam menggambarkan *use case diagram*:

Tabel 2.4 Simbol-simbol pada *Use Case Diagram*

No.	Simbol	Keterangan
1.		Menggambarkan pengguna aplikasi. Actor membantu memberikan suatu gambaran jelas tentang apa yang harus dikerjakan aplikasi.
2.		Menggambarkan perilaku aplikasi, termasuk didalamnya interaksi antara actor dengan aplikasi tersebut.
3.		Relasi asosiasi
4.		Relasi include memungkinkan suatu use case untuk dapat menggunakan fungsionalitas dari use case lainnya.
5.		Relasi ekstend memungkinkan suatu use case memiliki kemungkinan untuk memperluas fungsionalitas yang disediakan use case lainnya.

2.13.3 *Use Case* Skenario

Skenario adalah rangkaian langkah-langkah yang menjabarkan sebuah interaksi antara seorang pengguna dengan sebuah sistem.

Berikut ini merupakan penjelasan informasi yang terdapat dalam *use case*skenario:

a. Pra-kondisi menjelaskan apa yang harus dipastikan bernilai true sebelum sistem memunginkan *use case* dimulai.

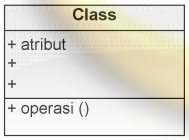
b. Pasca-kondisi menjelaskan tentang status sistem setelah sebuah *use case* selesai dikerjakan

2.13.4 *Class Diagram*

Class diagram mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat diantara mereka. *Class diagram* juga menunjukkan property dan operasi sebuah *class* dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut.

Berikut ini merupakan simbol-simbol yang digunakan dalam menggambarkan *class diagram*:

Tabel 2.5 Simbol-simbol pada *Class Diagram*

No.	Simbol	Keterangan
1.		Simbol *class*
3.		Relasi asosiasi

Dalam sistem objek oriented, terdapat beberapa tipe *class*, yaitu:

a. Entity class

Biasanya sesuai dengan item dalam kehidupan nyata dan mengandung informasi yang dikenal sebagai atribut.

b. Interfaces class

Penggunaberkomunikasidengansistemmelaluiantarmukapengguna, yang diimplementasikan dalam bentuk*interfaces class*.

c. Control class

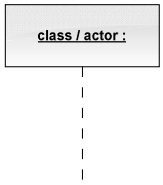
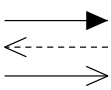
Control *class*mengimplementasikanaturanlogika bisnisatausistem bisnis. umumnya, setiapkasuspenggunaandiimplementasikandengansatuataulebih*control class*.

2.13.5 *Sequence Diagram*

Sequence diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram ini menunjukan sejumlah contoh objek dan *message* yang diletakkan di antara objek-objek di dalam *use case*. Komponen utama *sequence diagram* terdiri dari obyek yang di tuliskan dengan kotak segi empat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan progress vertikal.

Berikut ini merupakan simbol-simbol yang digunakan dalam menggambarkan *sequence diagram*:

Tabel 2.6 Simbol-simbol pada *Sequence Diagram*

No.	Simbol	Keterangan
1.		Simbol ini menggambarkan aktor atau tipe dari *class*.
2.		Message menggambarkan pesan yangdisampaikan dari satu *class* ke *class* yang lain.