BAB II

Landasan Teori

2.1. *Operating System Android*

Android adalah sebuah sistem operasi yang dikembangkan untuk perangkat *mobile* dan memiliki struktur atau merupakan modifikasi dari Linux (Li, 2011). Sampai saat ini android adalah salah satu sistem operasi yang paling sering melakukan pembaharuan atau *update.*

Awal berdirinya android sebenarnya bukan merupakan bagian dari Google Inc. Google membeli sebuah perusahaan *startup* yang memiliki nama sama yaitu Android Inc. pembelian Android, Inc. oleh Google pada tahun 2005 itu adalah salah satu strategi Google yang memiliki keinginan untuk memasuki pasar *mobile* yang dianggap cukup memiliki prospek.

Pembelian Android tersebut juga termasuk dengan didalamnya *team-developer* dari Android, Inc. itu sendiri. Dengan pembelian tersebut, Google ingin agar Android menjadi sebuah sistem operasi yang *Open* dan *free.* Dengan diberikannya keleluasaan untuk melakukan pengembangan terhadap Android tersebut, maka memberikan keuntungan lebih bagi *vendor* (khususnya *hardware manufaturers*) untuk dapat membuat serta menambahkan *proprietary extensions* milik mereka sendiri, hal ini penting agar membedakan produk mereka dengan vendor lainnnya sesama pengguna sistem operasi Android.

Keuntungan lain yang didapat apabila menggunakan sistem operasi ini adalah bahwa pengembang hanya perlu membuat aplikasi berbasis Android serta memastikan aplikasi tersebut berjalan baik, maka sudah dapat digunakan diberbagai macam *device* atau perangkat yang mengusung Android sebagai sistem operasinya.

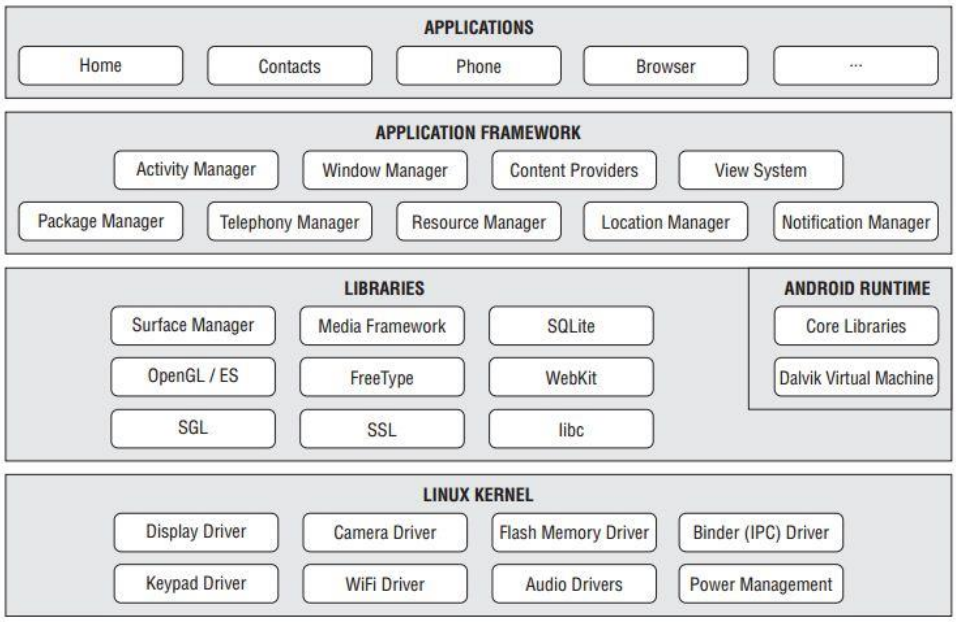
2.2. Fitur Android

Pada bukunya (Li, 2011) mengatakan bahwa Android merupakan sistem operasi terbuka (*Open source*) dan gratis (*free*) serta memberikan keleluasaan untuk melakukan kostumisasi, dimana tidak ada konfigurasi *hardware* dan *software* yang tetap. Android sendiri antara lain memberikan fitur seperti:

- *Storage* → menggunakan SQLite, relational database, data storage
- *Connectivity* → *support* GSM/Edge, CDMA, EV-DO,UMTS, Bluetooth, Wifi, LTE, dan WiMax
- *Messaging* → support SMS dan MMS
- *Web Browser*
- *Media Support* → 3gp, MP4, MP3, MIDI, JPEG, PNG, GIF, *etc*
- *Hardware Support* → Accelerometer Sensor, *Camera*, Digital Compass, Proximity Sensor dan GPS
- *Multi – Touch* → layar sudah mendukung *Multi – Touch*
- *Multi – Tasking* → Aplikasi yang ada sudah support *Multi – Tasking*
- *Flash Support* → Android versi 2.3 sudah *support* Flash 10.1.
- *Tethering* → sudah bisa melakukan *sharing* koneksi dengan menjadikan perangkat Android menjadi *wired/wireless hotspot*

2.3. Arsitektur Android

Pada bagian ini akan diberikan gambaran beberapa *Layers* yang ada pada Sistem Operasi Android.



Gambar 2.1. Arsitektur Android

Seperti pada gambar, android dibagi kedalam lima bagian dalam empat lapisan utama (*Layers*).

- **Linux Kernel**, dasar tumpuan atau *based* dari Android adalah kernel. Berisi berbagai macam *driver low level* dari komponen perangkat keras Android.
- ***Libraries***, terdiri dari satu set dari *library-library* yang ditulisan dalam bahasa C/C++ dimana digunakan oleh berbagai komponen yang ada pada Sistem Android. Sebagai contoh SQLite menyediakan database support sehingga aplikasi bisa menggunakannya untuk menyimpan data.
- **Android *Runtime*** berisi Core Libraries dan Dalvik Virtual Machine.
    - **Core Libraries** berisi inti *library* Java. Berisi sebagian besar fungsi dalam bahasa pemrograma Java.
    - **Dalvik** adalah *Virtual machine* yang didesain khusus untuk Android dan mengoptimalisasi kekuatan baterai dengan penggunaan *memory* dan CPU yang terbatas.

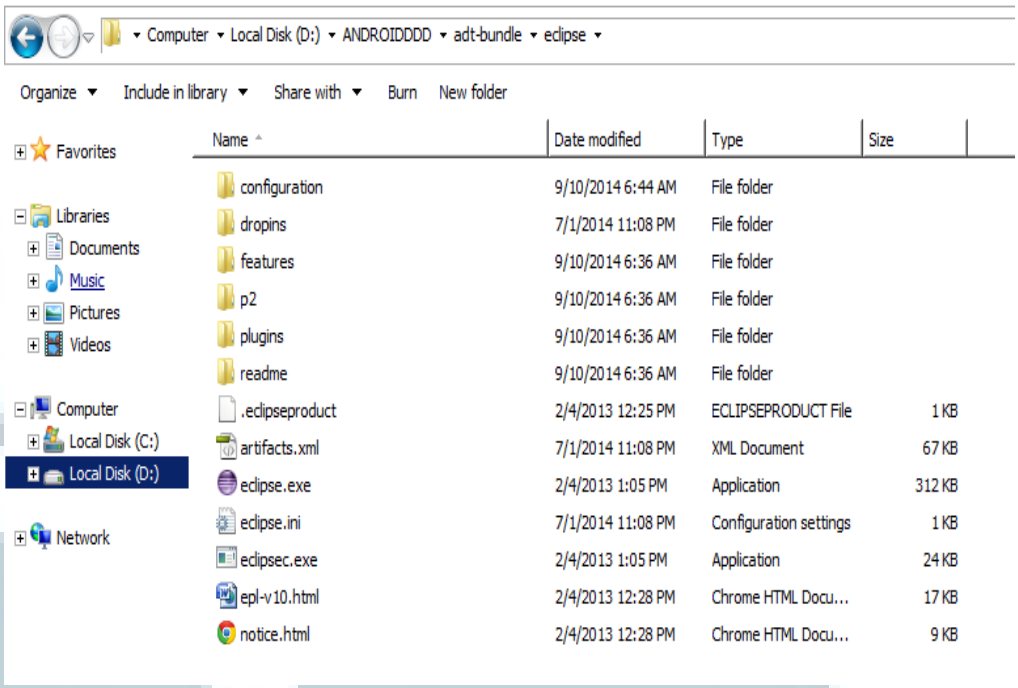
- ***Application Framework,*** berisi program yang berfungsi untuk mengatur fungsi-fungsi dasar dari perangkat Android.

- ***Application,*** ini adalah *top layer* atau lapisan teratas dari aplikasi itu sendiri. Contoh dari *application* dalam perangkat Android seperti *Contact, Web Browser,* SMS, dan aplikasi lainnya yang masih bisa diunduh dan diinstal dari Android *Market.*

2.4. Eclipse

Sebelum memulai untuk melakukan pengembangan aplikasi berbasis android, yang harus dilakukan pertama kali adalah kita harus memiliki Integrated Development Environment (IDE). Untuk kasus aplikasi berbasis Android, IDE yang direkomendasikan untuk pengembangan Android adalah menggunakan Eclipse. Eclipse adalah sebuah *software development environment* yang bisa berbagai macam bahasa pemrograman, seperti Java, Ada, C++, C, COBOL, Python, dan lain-lain.

Eclipse sebenarnya tidak bisa langsung digunakan untuk mendevelop aplikasi android, agar bisa digunakan untuk membuat aplikasi Android dibutuhkan sebuah Plugin bernama Android Development Tool (ADT). Apabila ADT dan Eclipse telah terpasang, maka Eclipse sudah bisa untuk digunakan, akan tetapi belum dilengkapi dengan Library dan Emulator Android untuk mencoba menjalankan aplikasi di PC atau Laptop, untuk itu masih diperlukan untuk dilakukan proses *Install* SDK Android (Software Development Kit).



Gambar 2.2. Isi *Folder* Eclipse

2.5. *Global Positioning System* (GPS)

Global Positioning System adalah sebuah sistem navigasi untuk menentukan posisi dengan bantuan satelit. GPS pertama kali dikembangkan oleh Militer Amerika Serikat yaitu bagian Departemen Pertahanan (*Department of Defense*) pada awal tahun 1970. Pada awalnya teknologi ini hanya digunakan oleh kalangan militer, namun sekarang sudah digunakan oleh kalangan sipil juga oleh kalangan Militer.

GPS memberikan informasi berupa posisi dan waktu secara terus menerus. GPS adalah sistem satu arah, dimana pengguna hanya bisa menerima sinyal yang dipancarkan oleh satelit. Untuk bisa digunakan secara optimal dan memperoleh *detail* posisi yang akurat, GPS sebaiknya digunakan di ruangan terbuka. Penggunaan GPS di tempat yang terhalang gedung tinggi atau pepohonan di dalam hutan juga membuat kerja GPS menjadi kurang akurat.

2.6. *Location Based Services*

Location Based Services (LBS) adalah sebuah layanan yang diberikan oleh operator seluler berisi informasi geografis untuk memberikan informasi terkait lokasi, tempat, atau suatu titik tertentu dimana posisi perangkat *mobile* berada (Kupper, 2005).

LBS memiliki kemungkinan untuk terjadinya komunikasi dua arah, hal ini terjadi karena pengguna meminta kepada penyedia layanan untuk mendapatkan informasi yang dibutuhkan dengan menggunakan posisi dan referensi yang diberikan oleh pengguna tersebut.

LBS dapat dibagi menjadi dua kategori, yaitu *reactive* dan *proactive* LBS. *Reactive* LBS terjadi pertama kali karena pengguna memanggil *service* dan membuat *session service*, baik melalui perangkat *mobile* ataupun PC. Kemudian pengguna melakukan *request* informasi tertentu, sedangkan LBS sendiri mengumpulkan data kemudian memprosesnya dan mengembalikan hasilnya tergantung dari lokasi pengguna sendiri misal lokasi bengkel Mobil atau Motor terdekat dari lokasi.

Proactive services terjalin meskipun tanpa *request* dari user tetapi interaksi yang terjadi adalah *asynchronously*. *Proactive service* secara terus menerus melacak lokasi dari pengguna.

2.7. Google Maps

Google maps adalah sebuah layanan peta Global secara gratis dan tersedia secara *online* oleh Google serta dapat diakses dengan menuliskan alamat URL http://maps.google.com/.

Google maps API merupakan aplikasi yang menggunakan basis dari pemrograman Javascript untuk dapat digunakan atau apabila ingin ditampilkan pada sebuah website atau *mobile*.

2.8. SQLite

SQLite adalah sebuah *software package* yang menyediakan *relational database system* atau RDBMS. RDBMS ini digunakan untuk

melakukan penyimpanan data yang dilakukan user sekalipun dalam jumlah yang besar. Selain itu untuk penyimpanan *data* dan mengatur *data*, *database engine* ini dapat digunakan untuk menglakukan pengolahan *query* secara kompleks dari banyak *table* dan bisa menampilkan hasilnya dalam bentuk ringkasan *data* (Kreibich, 2010).

Contoh produk RDBMS lain yang cukup terkenal antara lain Oracle *Database*, IBM DB2, Microsoft SQL Server.

Kata "Lite" pada SQLite ini tidak memcerminkan kemampuan seperti pada kemapuan RDBMS yang asli. Untuk itu SQLite disini lebih fokus pada kemampuan untuk melakukan *multiple process*, *file I/O*, *caches*, *query optimization* dan *query processing* pada perangkat *mobile.*
SQLite sendiri memiliki fitur antara lain:

- *Serverless*

  SQLite tidak memerlukan server yang terpisah untuk melakukan sebuah operasi. *Library* langsung diakses dari *Storage file* SQLite.
- *Zero Configuration*

  Kerena tidak ada server maka tidak perlu melakukan konfigurasi pada server.
- *Cross – Platform*

  Seluruh *database* yang ada disimpan dalam sebuah file yang dapat diakses oleh *platform* yang berbeda
- *Self-Contained*

  Sebuah *Library* berisi keseluruhan *database system*, dimana langsung terintegrasi ke sebuah *host applications*
- *Small Runtime Footprint*

  Program *default* berukuran kurang dari satu megabyte yang bisa membantu melakukan pengurangan *memory* secara seignifikan
- *Transactional*

  SQLite memberikan izin secara penuh untuk dapat mengakses beberapa proses atau *Threads* secara bersamaan.
- *Full-Featured*

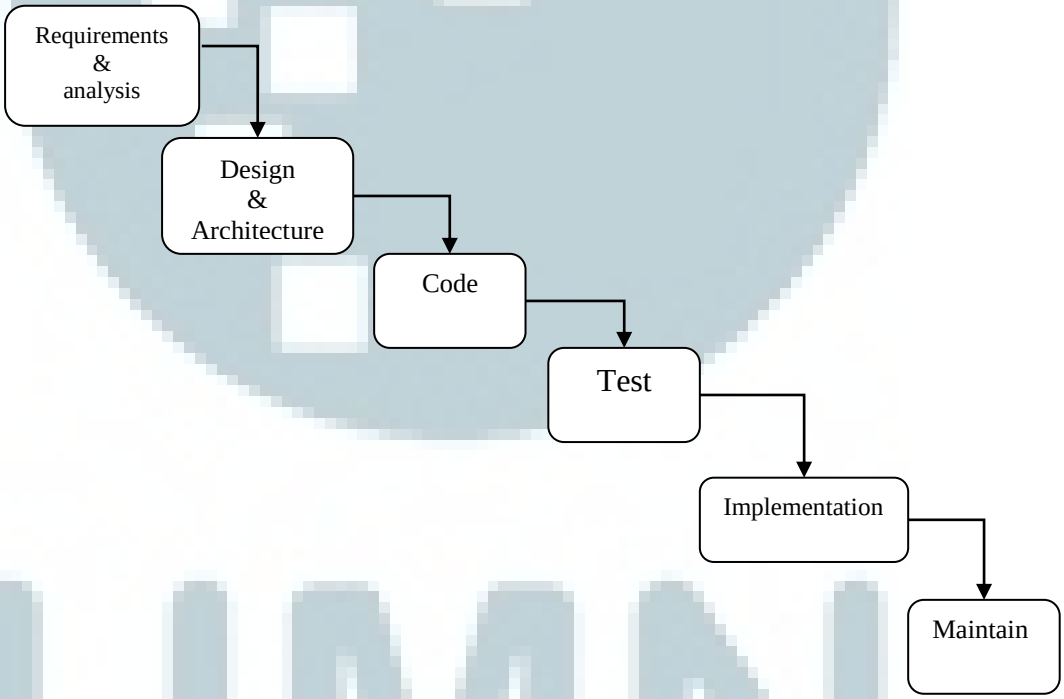
SQLite mendukung hampir semua bahasa *query*, sesuai dengan yang ada pada standar SQL92 (SQL2).

- Highly Reliable

  SQLite sangat memperhatikan *code testing* dan *verification*.

2.9. *System Development Life Cycle*

SDLC adalah fase untuk melakukan pendekatan analisa dan perancangan yang mempengaruhi pengembangan sistem melalui siklus tertentu dan aktifitas pengguna (Kendall & Kendall, 2008). SDLC memiliki beberapa model, salah satunya adalah *Waterfall*, terdiri dari beberapa tahapan seperti pada gambar dibawah.



Gambar 2.3. Model *Waterfall* (Kendall & Kendall, 2008)

Requirements &Analysis : Mengumpulkan kebutuhan dan keinginan pengguna serta melakukan analisa proses.

Design : Membuat rancangan untuk tabel basis data dan juga rancangan layar.

Code : Membuat pemograman pada rancangan layar dan basis data.

Test	:	Melakukan testing pada program yang dibuat
Implement	:	Melakukan implementasi pada pihak terkait
Maintain	:	Melakukan pemeliharaan pada sistem

2.10 *Data flow diagram*

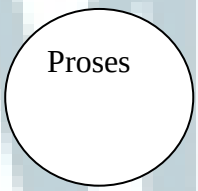
Diagram digunakan untuk melihat proses yang terjadi pada suatu proses. DFD adalah sebuah gambaran model dari objek, hubungan asosiasi dan aktifitas dengan mendeskripsikan bagaimana data mengalir diantara beberapa objek yang ada (Ralph & George, 2012).

Simbol DFD terdiri dari 4 simbol :

Simbol garis arus data, menggambarkan arah perpindaha data dan pergerakan dari elemen yang ada.

Gambar 2.4. Garis arus data

Simbol Proses, sebagai penanda sebuah fungsi yang bekerja pada suatu proses.

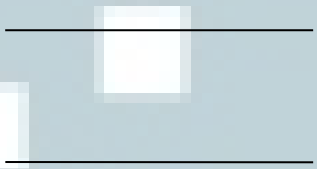


Gambar 2.5. Simbol Proses

Simbol entitas, menggambarkan darimana suatu data berasal atau sebagai penerima suatu hasil dari proses yang ada. Setiap entitas harus

terhubung dengan proses terlebih dahulu. Tidak bisa data berpindah tanpa diolah dari suatu entitas.

Tempat penyimpanan data, menggambarkan dimana data akan disimpan. Data disimpan biasanya dalam bentuk *table*.



Gambar 2.6. Simbol penyimpanan data

2.11. *Entity Relationship Diagram*

Diagram digunakan untuk menampilkan relasi antar entitas. Entitas merupakan suat sumber yang bisa digambarkan sebagai suatu individu ataupun objek. Diagram ini digunakan untuk menggambarkan basis data dari suatu organisasi dan menjadi dasar untuk pembuatan table yang ada pada basis data (Hall, 2011).

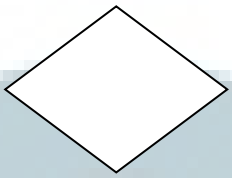
Simbol-simbol dalam ERD:

1. *Entity* (entitas)

Gambar 2.7. Entitas

Entitas adalah simbol yang menggambarkan pelaku atau suatu proses ataupun bisa juga sebagai laporan yang diharapkan oleh pengguna sistem.

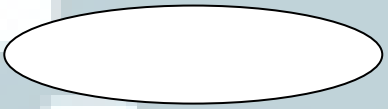
2. *Relationship* (relasi, hubungan antar entitas)



Gambar 2.8. Relasi

Relasi adalah hubungan yang terjadi antara entitas, simbol ini terbentuk dari kata kerja yang digunakan dalam sistem. Relasi ini bisa terjadi antara satu dengan satu atau satu dengan banyak. Tidak bisa antara banyak dengan banyak.

3. *Attribute* (Atribut)



Gambar 2.9. Atribut

Atribut adalah karakteristik dari suatu entitas atau relasi yang didalamnya terdapat penjelasan tentang entitas atau relasi tersebut. suatu entitas pasti terdapat banyak atribut.

4. Alur

Gambar 2.10. Alur

Alur memiliki fungsi untuk menghubungkan atribut dengan entitas dan entitas dengan relasi.