BAB II
LANDASAN TEORI

II.1 Pengertian *Game*

Game dalam kamus bahasa Indonesia adalah permainan. Sebuah *game* adalah sebuah sistem dimana pemain terlibat dalam konflik buatan, disini pemain berinteraksi dengan sistem dan konflik dalam permainan merupakan rekayasa atau buatan, dalam permainan terdapat peraturan yang bertujuan untuk membatasi perilaku pemain dan menentukan permainan. *Game* bertujuan untuk menghibur, biasanya *game* banyak disukai oleh anak–anak hingga orang dewasa. *Game* sebenarnya penting untuk perkembangan otak, untuk meningkatkan konsentrasi dan melatih untuk memecahkan masalah dengan tepat dan cepat karena dalam *game* terdapat berbagai konflik atau masalah yang menuntut penggunanya untuk menyelesaikannya dengan cepat dan tepat.

Berdasarkan *genre*-nya *game* dibagi menjadi beberapa jenis, yaitu [4]:

1. *First Person Shooter* (FPS)

   Game jenis ini adalah jenis permainan tembak-tembakan dengan tampilan pemain adalah tokoh yang dimainkan. Biasanya berupa misi untuk suatu tujuan tertentu. Ciri khas *game* ini adalah penggunaan senjata jarak jauh.

   Contoh : Counter-Strike, Call of Duty, Ghost Recon, Point Blank, Crossfire



Gambar II.1 Tampilan *Game* Point Blank

2. *Real Time Strategy* (RTS)

   Game jenis ini adalah jenis permainan strategi, taktik, dan logika. Biasanya berupa perang yang tiap pemainnya memiliki suatu pasukan atau negara. Dalam RTS permainan dapat berupa sejarah, fantasi, dan fiksi ilmiah.

   Contoh : Age of Empires, Warcraft, Command & Concuer, Rise of Nations, Stronghold



Gambar II.2 Tampilan *Game* Age Of Empire

3. *Role Playing Game* (RPG)

   Game jenis ini adalah *game* yang para pemainnya memainkan peran tokoh-tokoh khayalan dan berkolaborasi untuk merajut sebuah cerita.

   Ada pula jenis lain game RPG ini, diantaranya :

   a. LARP (*Live Action Role Playing*) adalah *game* RPG dimana para pemain bisa melakukan gerakan fisik tokohnya. Biasanya pemain menggunakan kostum dan menggunakan alat-alat yang sesuai dengan tokoh, dunia dan cerita yang dia mainkan. Contoh : Final Fantasy



Gambar II.3 Tampilan *Game* Final Fantasy

   b. MMORPG (*Massively Multiplayer Online Role-Playing Game*) adalah *game* RPG yang melibatkan ribuan pemain untuk bermain *game* bersama dalam dunia maya. Contoh : DotA, Ragnarok



Gambar II.4 Tampilan *Game* DotA

4. *Construction and Management Simulation Games*

   Game jenis ini adalah *game* yang mensimulasikan proyek membangun dan mengelola. *Game* ini jarang yang melibatkan konflik dan eksplorasi, dan hampir tidak pernah meliputi tantangan fisik.

   Contoh : Sim City, Roller Coster Tycoon, Caesar



Gambar II.5 Tampilan *Game* Sim City

5. *Vehicle Simulation*

   Jenis permainan ini mensimulasi pengoperasian beberapa kendaraan, kendaraan bisa berupa pesawat terbang, pesawat tempur, kereta, kendaraan perang, maupun kendaraan konstruksi.

   Contoh : Train Simulator, Garbage Truck Simulator, FlightGear, Tram, Orbiter



Gambar II.6 Tampilan Game Garbage Truck Simulator

6. *Adventure Games*

Game jenis ini adalah *game* yang mengutamakan masalah eksplorasi dan pemecahan teka-teki. Namun terkadang meliputi masalah konseptual yaitu menyelesaikan misi yang sudah ditetapkan untuk menyelesaikan permainan, dan terdapat tantangan fisik.

Contoh : Indiana Jones, God of War, Tomb Raider, Assasin's Creed



Gambar II.7 Tampilan *Game* Assassin's Creed

7. *Fighting*

Game jenis *Fighting* adalah *game* yang memerlukan kecepatan refleks dan koordinasi mata-tangan, inti dari game ini adalah pemain beradu kemampuan dalam penguasaan jurus dari karakter yang terdapat di dalam *game*, pengenalan karakter dan *timing* sangatlah penting, permainan selesai jika lawan sudah kalah. Dan berbeda seperti *game* pada umumnya, yang umumnya hanya melawan *Artificial Intellegence* atau istilah umumnya melawan komputer saja, pemain jenis *game* ini baru teruji kemampuan sesungguhnya dengan melawan pemain lainnya.

Contoh : Street Fighter, Tekken.

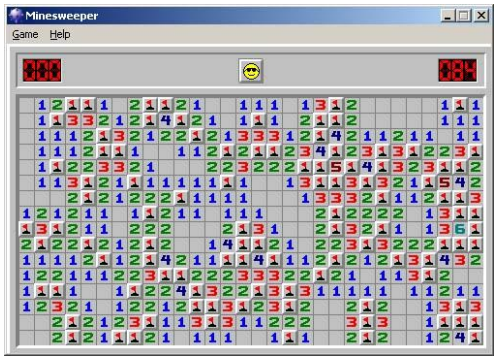


Gambar II.8 Tampilan *Game* Street Fighter

8. *Puzzle*

   Game jenis *puzzle* berintikan mengenai pemecahan teka-teki, baik itu menyusun balok, menyamakan warna bola, memecahkan perhitungan matematika, melewati labirin, sampai mendorong-dorong kota masuk ke tempat yang seharusnya, itu semua termasuk dalam jenis *puzzle*.

   Contoh : Tetris, Minesweeper, Bejeweled, Sokoban dan Bomberman.



Gambar II.9 Tampilan *Game* Minesweeper

9. *Sports*

   Game jenis *sports* merupakan jenis *game* yang bertema tentang olahraga, biasanya tampilan *game* diusahakan serealistik mungkin walau kadang ada yang menambah unsur fiksi.

   Contoh : Pro Evolution Soccer, seri NBA, seri FIFA



Gambar II.10 Tampilan *Game* Pro Evolution Soccer

10. *Racing*

Game jenis *racing* adalah *game* dengan tema kompetisi kecepatan. Terkadang dalam permainannya pemain dapat memilih kendaraan, mendandani, *upgrade* mesin bahkan mengecatnya.

Contoh: Grand Turismo, Need For Speed.



Gambar II.11 Tampilan *Game* Need For Speed

II.2 Media Transmisi

Media Transmisi adalah media yang digunakan sebagai penghubung antara pengirim dan penerima, untuk melintaskan isyarat, dan isyarat inilah yang akan dimanipulasi dengan berbagai macam cara dan akan diubah kembali menjadi data.

II.2.1 *Bluetooth*

Sebagian besar produsen *handphone* selalu melengkapi *handphone* buatannya dengan fasilitas *Bluetooth*. Fasilitas ini memungkinkan sebuah perangkat dapat saling berhubungan dengan perangkat lainnya tanpa menggunakan kabel. *Bluetooth* adalah salah satu media yang dapat digunakan untuk melakukan pertukaran (transfer) data dari satu perangkat ke perangkat lainnya yang juga memiliki fasilitas *Bluetooth* didalamnya.

II.2.2 Sejarah *Bluetooth*

Nama *Bluetooth* berasal dari nama raja di akhir abad sepuluh, Harald Blatand (Abad 10) yang di Inggris juga dijuluki Harald Bluetooth kemungkinan karena memang giginya berwarna gelap. Ia adalah raja Denmark yang telah berhasil menyatukan suku-suku yang sebelumnya berperang, termasuk suku dari wilayah yang sekarang bernama Norwegia dan Swedia. Bahkan wilayah Scania di Swedia, tempat teknologi *Bluetooth* ini ditemukan juga termasuk daerah kekuasaannya. Kemampuan raja itu sebagai pemersatu juga mirip dengan

teknologi *Bluetooth* sekarang yang bisa menghubungkan berbagai peralatan seperti komputer personal dan *handphone*.

Sedangkan logo *Bluetooth* berasal dari penyatuan dua huruf Jerman yang analog dengan huruf H dan B (singkatan dari Harald Bluetooth), yaitu ᚼ(Hagall) dan ᛒ(Blatand) yang kemudian digabungkan [5].



Gambar II.12 Logo *Bluetooth*

II.2.3 Aplikasi dan Layanan *Bluetooth*

Teknologi *Bluetooth* adalah sebuah teknologi *wireless* yang mampu menyediakan layanan komunikasi data dan suara dengan jarak jangkauan yang terbatas. *Bluetooth* adalah sebuah teknologi komunikasi *wireless* (tanpa kabel) yang beroperasi dalam pita frekuensi 2,4 GHz *unlicensed* ISM (*Industrial, Scientific and Medical*) dengan menggunakan sebuah *frequency hopping tranceiver* yang mampu menyediakan layanan komunikasi data dan suara secara *real-time* antara *host-host bluetooth* dengan jarak jangkauan layanan yang terbatas.

Bluetooth sendiri dapat berupa *card* yang bentuk dan fungsinya hampir sama dengan *card* yang digunakan untuk *wireless local area network* (WLAN) dimana menggunakan frekuensi radio standar IEEE 802.11, hanya saja pada

Bluetooth mempunyai jangkauan jarak layanan yang lebih pendek dan kemampuan transfer data yang lebih rendah.

Bluetooth menggunakan salah satu dari dua jenis frekuensi *Spread Specturm Radio* yang digunakan untuk kebutuhan *wireless*. Jenis frekuensi yang digunakan adalah *Frequency Hopping Spread Spectrum* (FHSS), sedangkan yang satu lagi yaitu *Direct Sequence Spread Spectrum* (DSSS) digunakan oleh IEEE802.11xxx. *Transceiver* yang digunakan oleh *Bluetooth* bekerja pada frekuensi 2,4 GHz *unlicensed* ISM (*Industrial, Scientific, and Medical*) [5].

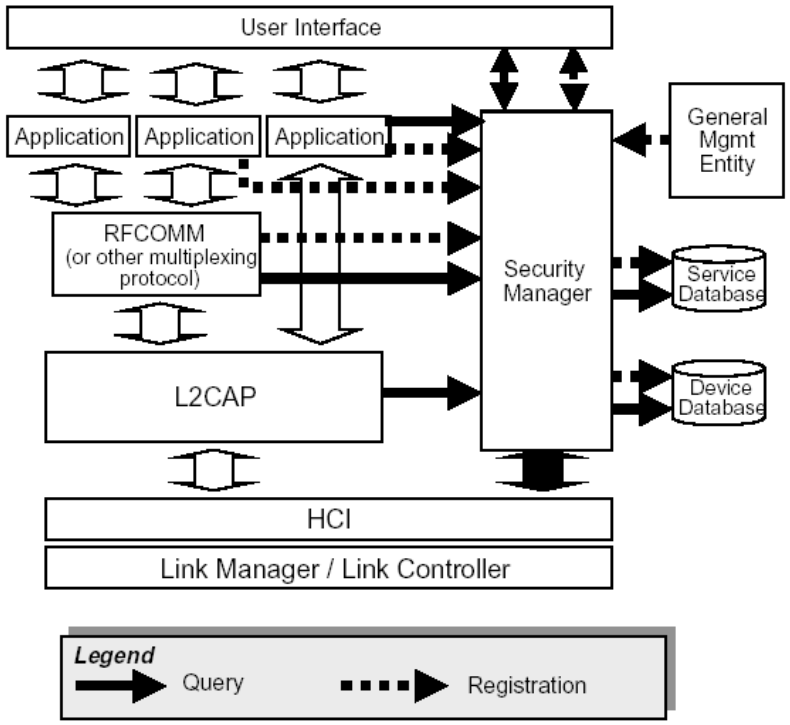
Media *Bluetooth* memungkinkan komunikasi antara dua *smartphone* dalam jarak pendek. Untuk melindungi penggunaan dan keamanan informasi, sistem *Bluetooth* memberikan keamanan pada layer aplikasi dan layer sambungan (*link*). Sistem keamanan dirancang untuk lingkungan berpasangan artinya ketika dua buah perangkat *smartphone* akan dihubungkan maka akan terjadi proses *authorization* dan *authentication*, proses tersebut digunakan untuk mengetahui siapa pengguna, apa perangkatnya dan apa kewenangannya. Berikut penjelasan mengenai proses yang terjadi :

1. *Authentication*

   Proses untuk verifikasi ‘siapa’ pada sisi lain koneksi. *Authentication* dilakukan pada perangkat. *Authentication* pada *Bluetooth* berdasarkan kunci sambungan atau dengan pencocokan dengan memasukan nomor PIN.

2. *Authorization*

Proses ini adalah proses yang menentukan apakah suatu perangkat X diijinkan untuk mengakses layanan Y. Pada tahap ini konsep "terpercaya" digunakan. *Bluetooth* menggunakan keamanan tingkat sambungan (*link*) dimana setiap koneksi diberikan kunci autentifikasi unik rahasia dan kunci enkripsi yang diturunkan dari kunci autentifikasi. Komunikasi antar perangkat *Bluetooth* menggunakan teknik *fast frequency-hoping spread spectrum* (FHSS) yang menggunakan 79 kanal radio berbeda untuk mencegah interferensi oleh perangkat rumah tangga seperti *oven microwave* sebab frekuensi yang dihasilkan perangkat rumah tangga tersebut sama dengan frekuensi *bluetooth*. FHSS dapat mencegah interferensi sebab merubah frekuensi transmisi sebanyak 1600 kali per detik dan kalaupun terjadi interferensi pada satu frekuensi maka hanya transmisi yang dilakukan pada frekuensi itu saja yang terpengaruhi. FHSS juga menambah keamanan pada transmisi data antar perangkat karena membuat perangkat lebih sulit untuk disadap. Daya transmisi yang dipancarkan rendah untuk mencegah transmisi yang dipancarkan terpropagasi jauh dan membuat lebih sulit memutus hubungan antar transmisi.



Gambar II.13 Arsitektur *Bluetooth*

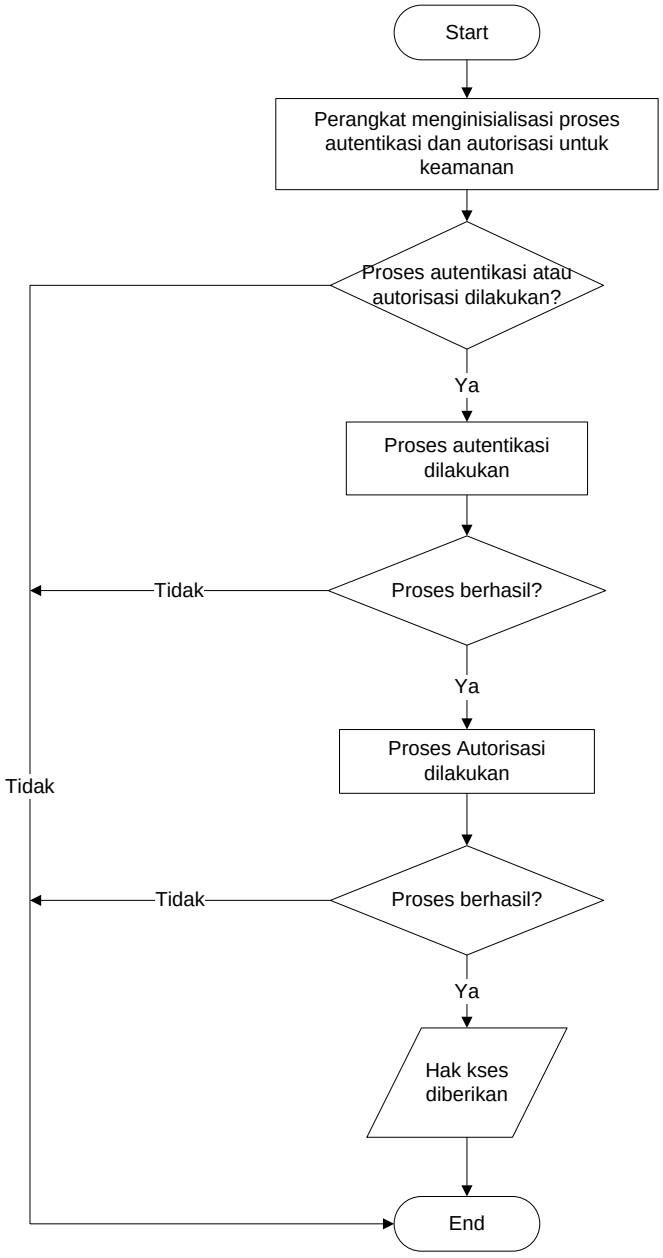
Arsitektur *Bluetooth* ditunjukan pada Gambar 3.9. Manajer keamanan (*security manager*) menyimpan informasi tentang keamanan layanan dan perangkat. Manajer *(General Mgmt Entity)* memutuskan apakah menerima akses atau memutuskan hubungan dan meminta autentifikasi dan enkripsi jika dibutuhkan. Manajer keamanan juga memulai hubungan terpercaya dan menanyakan dan mencocokan kode PIN dari pengguna.

Bluetooth memiliki beberapa tingkat keamanan yang berbeda yang dapat didefinisikan untuk berbagai perangkat dan layanan. Setiap perangkat mendapat status saat perangkat tersebut untuk pertama kali terhubung dengan perangkat lain.

Perangkat dapat memiliki dua tingkat kepercayaan (*trust*) , terpercaya (*trusted*) dan tidak terpecaya (*untrusted*). Tingkat trusted membutuhkan hubungan

yang tetap dan terpercaya dan tingkat ini memiliki akses ke semua layanan. Perangkat harus diautentifikasi terlebih dahulu. Perangkat dengan status untrusted tidak memiliki hubungan yang tetap dan akses layanannya terbatas. Perangkat untrusted dapat juga memiliki hubungan yang tetap tetapi tidak dikenali sebagai trusted. Perangkat yang baru terhubung ditandai sebagai perangkat tidak dikenal dan selalu untrusted. *Bluetooth* memiliki 3 mode keamanan berbeda antara lain:

1. Security mode1 : Mode tidak aman (*Non secure mode).* Perangkat tidak akan menginisialisasi keamanan.
2. Security mode 2 : Keamanan tingkat layanan (*service level enforced security mode*). Perangkat tidak menginisialisasi prosedur keamanan sebelum kanal terhubung pada level L2CAP*(Logical Link Control and Adaptaion Layer Protocol)* . Mode ini mengijinkan akses yang bermacam-macam dan fleksibel untuk aplikasi terutama untuk aplikasi yang berjalan dengan berbagai permintaan keamanan yang paralel.
3. Security mode 3 : Keamanan tingkat sambungan (*link level enforced security mode*). Perangkat menginisialisasi prosedur keamanan sebelum hubungan dibangun pada tingkat LPM (*Link Manager Protocol*) selesai.



Gambar II.14 *Flowchart* Pengecekan Akses

II.3 Smartphone

Hampir sama dengan ponsel, *smartphone* juga berorientasi ke arah telekomunikasi. Hanya saja terdapat sedikit perbedaan pada *smartphone*, selain fungsi utamanya sebagai perangkat telekomunikasi, penciptaan awal dari

smartphone juga difungsikan sebagai PC atau komputer *desktop* yang bersifat *mobile*[6].

II.3.1 Pengertian *Smartphone*

Belum ada kesepakatan dalam industri telekomunikasi mengenai definisi dari *smartphone* dan ketentuan-ketentuan apa saja yang dapat membuat sebuah perangkat telekomunikasi tergolong ke dalam jenis *smartphone*, dan pengertian dari *smartphone* itu pun berubah mengikuti waktu. *Smartphone* dapat dibedakan dengan *handphone* biasa dengan dua cara,yaitu : bagaimana mereka dibuat dan apa yang mereka bisa lakukan[2].

Kebanyakan alat yang dikategorikan sebagai *smartphone* jika alat tersebut menggunakan sistem operasi dan di dalamnya terdapat fitur-fitur yang mendukung sepenuhnya fasilitas surat elektronik dengan fungsi pengaturan yang lengkap. Fungsi lainnya dapat menyertakan miniatur papan ketik QWERTY, layar sentuh, kamera, pengaturan daftar nama, aplikasi navigasi, kemampuan membaca dokumen bisnis, pemutar musik, penjelajah foto dan melihat klip video, penjelajah internet dan lain-lain. Fitur yang paling sering ditemukan dalam *smartphone* adalah kemampuannya menyimpan daftar nama sebanyak mungkin, tidak seperti *handphone* biasa yang mempunyai batasan maksimum penyimpanan daftar nama.

II.3.2 Sistem Operasi Pada *Smartphone*

Sistem operasi *mobile* dikenal dengan sebutan *platform*. Ada berbagai jenis sistem operasi (OS) yang dijalankan perangkat keras yang beredar pasaran.

Beberapa ada yang akrab terdengar oleh kita seperti iOS, Android OS, BlackBerry OS dan sebagainya. Berikut ini adalah beberapa sistem operasi (OS) yang dijalankan oleh vendor-vendor dan perusahaan-perusahaan besar di dunia[7].

1. Apple iOS

   iOS adalah sistem operasi *mobile* milik Apple yang dikembangkan dan hanya diaplikasikan untuk perangkat Apple Inc seperti iPhone, iPod Touch, iPad dan Apple TV. iOS menjadi *leading* bagi sistem operasi *mobile* yang lain dengan mengusung *interface multitouch* yang bisa digeser (*slider*), ditekan (*switch*), dicubit dan tombol. Perangkat Apple masih menjadi nomor satu dalam hal teknologi dan angka penjualan di dunia. Produk *smartphone* Apple yang terbaru adalah iPhone 4 dan iOS5 adalah versi terbaru.



Gambar II.15 Perangkat Dengan Sistem Operasi iOS

2. Android OS

   Android OS adalah sebuah sistem operasi *mobile* yang diperuntukan bagi *smartphone* dan komputer tablet. Android OS yang bebasis linux kernel ini dikembangkan oleh Open Handset Alliance dibawah bendera

Google. Banyak *developer* independen di seluruh dunia (termasuk Indonesia) yang mengembangkan aplikasi-aplikasi Android dan menguploadnya di Android Market, dengan tujuan aplikasi-aplikasi tersebut dapat didownload oleh para pengguna *smartphone* yang berlogo robot hijau ini. Android terdaftar sebagai *smartphone* dengan angka penjualan terbaik pada kuartal 2010 tercatat lebih dari 200 juta perangkat Android berhasil dijual. Dengan banyaknya *developer* independen membuat teknologi Android OS berkembang dengan pesat. Dalam empat tahun terakhir (rilis tahun 2008) Android berhasil merilis versi terbaru mulai dari : Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb dan yang terbaru Ice Cream Sandwich. *Smartphone* yang menggunakan Android OS adalah Samsung Galaxy, Samsung Galaxy Nexus dan lain sebagainya.



Gambar II.16 Perangkat Dengan Sistem Operasi Android

3. Symbian OS

Symbian OS adalah sebuah sistem operasi yang digunakan untuk operasi standar ponsel pada perangkat *smartphone*. Symbian OS menetapkan

persyaratan bahwa sistem operasi ini hanya dapat digunakan oleh ponsel dengan fitur 2.5G dan 3G. Ada beberapa vendor terbesar didunia yang berkolaborasi menanamkan saham di "proyek" Symbian OS ini, seperti Nokia, Motorola, Sonny Ericsson. Dan yang memegang saham terbesar dimiliki oleh Nokia, jadi tidak mengherankan produk-produk Nokia didominasi dengan OS Symbian. Versi terbaru dari Symbian OS adalah Symbian 3, setidaknya ada sepuluh *smartphone* yang menggunakan sistem operasi ini diantaranya Nokia N8, Nokia C6-01, Nokia E7-00 dan sebagainya.



Gambar II.17 Perangkat Dengan Sistem Operasi Symbian

4. BlackBerry OS

BlackBerry menggunakan BlackBerry OS sebagai sistem operasinya. BlackBerry OS, dikembangkan oleh *Reseach in Motion* (RIM), untuk BlackBerry, perusahaan telekomunikasi asal Kanada ini. Diawal kemunculannya BlackBerry booming dengan layanan *push-email* dan sebagai *smartphone* yang sukses dipasaran. Versi terakhir dari BlackBerry OS adalah BlackBerry OS 7, salah satu *smartphone* andalannya BlackBerry Curve 9380. Indonesia dinilai sebagai pasar

terbesar bagi penjualan BlackBerry, namun sayangnya pihak RIM belum mau berinvestasi di lndonesia dan tetap membangun pabriknya di Malaysia yang notabene angka penjualan BlackBerry jauh dibawah Indonesia.



Gambar II.18 Perangkat Dengan Sistem Operasi BlackBerry

II.4 Sistem Operasi Android

Android merupakan *platform* yang berjalan diatas sistem operasi Linux. Android pertama kali dikembangkan Google dan Open Handset Alliance. Bahasa pemrograman dasar yang dipakai untuk pengembangan aplikasi di atas Android adalah Java dengan pemanfaatan *Google developed Java libraries*.



Gambar II.19 Gambar Logo Android

II.4.1 Perkembangan Sistem Operasi Android

Sejak Android dirilis telah dilakukan berbagai pembaruan berupa perbaikan *bug* dan penambahan fitur baru, sehingga Sistem Operasi Android terdiri dari beberapa versi, antara lain [8]:

1. Android versi 1.1

   Pada 9 Februari 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.



Gambar II.20 Logo Android Versi 1.1

2. Android versi 1.5 (Cupcake)

   Pada tanggal 30 April 2009, Google kembali merilis telepon seluler dengan menggunakan Android dan SDK (*Software Development Kit*) dengan versi 1.5 (Cupcake). Terdapat beberapa pembaruan termasuk juga penambahan beberapa fitur dalam seluler versi ini.



Gambar II.21 Logo Cupcake

3. Android versi 1.6 (Donut)

   Donut (versi 1.6) dirilis pada 15 September 2009 dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai indikator dan kontrol *applet* VPN.



Gambar II.22 Logo Donut

4. Android versi 2.0/2.1 (Eclair)

   Pada 26 Oktober 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1 (Eclair), perubahan yang dilakukan adalah pengoptimalan *hardware*, peningkatan Google Maps 3.1.2, perubahan UI dengan *browser* baru dan dukungan HTML5, daftar kontak yang baru, dukungan flash untuk kamera 3,2 MP, digital Zoom, dan *Bluetooth* 2.1.



Gambar II.23 Logo Éclair

5. Android versi 2.2 (Froyo: Frozen Yoghurt)

   Pada 20 Mei 2010, Android versi 2.2 (Froyo) diluncurkan. Perubahan-perubahan umumnya terhadap versi-versi sebelumnya antara lain dukungan Adobe Flash 10.1, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, pemasangan aplikasi dalam SD Card, dan kemampuan *auto update* dalam aplikasi Android Market.



Gambar II.24 Logo Froyo

6. Android versi 2.3 (Gingerbread)

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



Gambar II.25 Logo Gingerbread

7. Android versi 3.0/3.1 (Honeycomb)

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



Gambar II.26 Logo Honeycomb

8. Android versi 4.0 (ICS :Ice Cream Sandwich)

   Diumumkan pada tanggal 19 Oktober 2011, membawa fitur Honeycomb untuk *smartphone* dan menambahkan fitur baru termasuk membuka kunci dengan pengenalan wajah, jaringan data pemantauan penggunaan dan kontrol, terpadu kontak jaringan sosial, dan berbagi informasi dengan menggunakan NFC.



.

Gambar II.27 Logo Ice Cream Sandwich

II.4.2 Arsitektur Sistem Operasi Android

Sistem operasi Android memiliki 4 lapisan (*layer*) yang merupakan komponen sistem Android. Setiap lapisan tersebut menghimpun beberapa program yang mendukung fungsi-fungsi spesifik dari sistem operasi. Lebih jelasnya terlihat pada gambar berikut :



Gambar II.28 Arsitektur *Platform* Android [9]

II.4.2.1 *Linux Kernel* Dalam Android OS

Tumpukan paling bawah pada Arsitektur Android ini adalah kernel. Google menggunakan kernel Linux versi 2.6 untuk membangun sistem Android, yang mencakup *memory management*, *security setting*, *power management*, dan beberapa *driver hardware*. Kernel linux menyediakan *driver* layar, kamera, *keypad*, *WiFi*, *Flash Memory*, audio, dan IPC (*Interprocess Communication*). Kernel berperan sebagai *abstraction layer* antara *hardware* dan keseluruhan *software.* Sebagai contoh, HTC GI dilengkapi dengan kamera. Pada kernel Android terdapat *driver* kamera, dan *driver* tersebut mengirimkan perintah kepada *hardware* kamera ketika *software* kamera digunakan.[2]

II.4.2.2 *Android Runtime* Dalam Android OS

Lapisan setelah *Linux Kernel* adalah *Android Runtime*. Pada Android tertanam paket pustaka inti yang menyediakan sebagian besar fungsi Android. Inilah yang membedakan Android dibandingkan dengan sistem operasi lain yang juga mengimplementasikan Linux. *Android Runtime* merupakan mesin virtual yang membuat aplikasi Android menjadi lebih tangguh dengan paket pustaka yang telah ada [2]. Dalam *Android Runtime* terdapat 2 bagian utama, diantaranya:

1. Pustaka Inti (*Core Libraries*), Android dikembangkan melalui bahasa pemrograman Java, tapi *Android Runtime* bukanlah mesin virtual Java. Pustaka inti Android menyediakan hampir semua fungsi yang terdapat pada pustaka Java serta beberapa pustaka khusus Android.
2. Mesin Virtual Dalvik (*Dalvik Virtual Machine*), Dalvik merupakan sebuah mesin virtual yang dikembangkan oleh Dan Bornstein yang terinspirasi dari nama sebuah perkampungan yang berada di Iceland. Dalvik hanyalah interpreter mesin virtual yang mengeksekusi *file* dalam format *Dalvik Executable* (*.dex). Dengan format ini Dalvik akan mengoptimalkan efisiensi penyimpanan dan pengalamatan memori pada *file* yang dieksekusi. Dalvik berjalan di atas kernel Linux 2.6, dengan fungsi dasar seperti *threading* dan manajemen memori yang terbatas.

II.4.2.3 *Libraries* Dalam Android OS

Android menggunakan beberapa paket pustaka yang terdapat pada C/C++ dengan standar Berkeley Software Distribution (BSD) hanya setengah

dari yang aslinya untuk tertanam pada kernel Linux [2]. Beberapa pustaka diantaranya:

1. *Media Library* untuk memutar dan merekam berbagai macam format audio dan video.
2. *Surface Manager* untuk mengatur hak akses *layer* dari berbagai aplikasi.
3. *Graphic Library* termasuk didalamnya SGL dan OpenGL, untuk tampilan 2D dan 3D.
4. *SQLite* untuk mengatur relasi *database* yang digunakan pada aplikasi.
5. *LibWebCore* untuk *browser* dan keamanan internet.

II.4.2.4 *Application Framework* Dalam Android OS

Kerangka aplikasi menyediakan kelas-kelas yang dapat digunakan untuk mengembangkan aplikasi Android. Selain itu, juga menyediakan abstraksi generik untuk mengakses perangkat, serta mengatur tampilan *User Interface* dan sumber daya aplikasi. Bagian terpenting dalam kerangka aplikasi Android adalah sebagai berikut[2]:

1. *Activity Manager*, berfungsi untuk mengontrol siklus hidup aplikasi dan menjaga keadaan *Backstack* untuk navigasi penggunaan.
2. *Content Providers*, berfungsi untuk merangkum data yang memungkinkan digunakan oleh aplikasi lainnya, seperti daftar nama.
3. *Resuource Manager*, untuk mengatur sumber daya yang ada dalam program. Serta menyediakan akses sumber daya diluar kode program, seperti karakter, grafik, dan *file layout*.

4. *Location Manager*, berfungsi untuk memberikan informasi detail mengenai lokasi perangkat Android berada.
5. *Notification Manager*, mencakup berbagai macam peringatan seperti, pesan masuk dan lain sebagainya yang akan ditampilkan pada status bar.

II.4.2.5 *Applications* Dalam Android OS

Puncak dari diagram arsitektur android adalah lapisan aplikasi dan *widget*. Lapisan aplikasi merupakan lapisan yang paling tampak pada pengguna ketika menjalankan program. Pengguna hanya akan melihat program ketika digunakan tanpa mengetahui proses yang terjadi dibalik lapisan aplikasi. Lapisan ini berjalan dalam *Android runtime* dengan menggunakan kelas dan *service* yang tersedia pada *Framework* aplikasi [2].

II.4.3 *Android Software Development Kit*

Android SDK adalah *tools* API (*Application Programming Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. Android merupakan *subset* perangkat lunak untuk ponsel yang meliputi sistem operasi, *middle ware* dan aplikasi kunci yang di *release* oleh Google [2].

II.4.4 *Android Development Tools*

Android Development Tools (ADT) adalah *plugin* untuk Eclipse IDE yang dirancang untuk memberikan lingkungan yang terintegrasi yang kuat di

mana untuk membangun aplikasi Android. ADT memperluas kemampuan Eclipse untuk membiarkan *developer* cepat dalam membuat proyek baru Android, membuat UI aplikasi, tambahan paket berdasarkan kerangka Android API, *debug* aplikasi menggunakan *tools* Android SDK [2].

II.5 Pemrograman Berorientasi Objek

Pengembangan sistem berorientasi objek berbeda dari pengembangan sistem konvensional (aliran data) yang memandang perangkat lunak sebagai fungsi dan data yang tersimpan. Perbedaan pengembangan sistem berorientasi objek dengan pengembangan sistem konvensional yaitu pada pengembangan sistem secara konvensional, kebanyakan berfokus pada fungsi, tetapi terdapat pendekatan yang berfokus pada data terutama pada basis data dan pemodelan informasi, sedangkan pengembangan berorientasi objek berpusat pada objek yang mengkombinasikan data dan fungsi keduanya sekaligus [10].

Pengembangan sistem berorientasi objek mempunyai kelebihan sebagai berikut:

1. Pendekatan objek menuntut penggunaan ulang komponen-komponen program sebelumnya.
2. Perangkat lunak yang dikembangkan dengan berorientasi objek mempermudah pemeliharaan.
3. Sistem berorentasi objek lebih mudah diadaptasi menjadi sistem yang lebih besar dibuat dengan cara membangun subsistem-subsistem yang dapat digunakan.

II.6 Bahasa Pemrograman

Bahasa pemrograman atau sering diistilahkan sebagai bahasa komputer, adalah teknik komando/instruksi standar untuk memerintah komputer. Bahasa pemrograman merupakan himpunan atau kumpulan dari aturan sintaks dan semantik yang dipakai untuk mendefinisikan program komputer [10].

Menurut tingkat kedekatannya dengan mesin komputer, bahasa pemrograman terdiri dari [10]:

1. Bahasa Mesin, yaitu memberikan perintah kepada komputer dengan memakai kode bahasa biner, contohnya 01100101100110
2. Bahasa Tingkat Rendah, atau dikenal dengan istilah bahasa rakitan (*Assembly*), yaitu memberikan perintah kepada komputer dengan memakai kode-kode singkat (kode mnemonic), contohnya MOV, SUB, CMP, JMP, JGE, JL, LOOP, dsb.
3. Bahasa Tingkat Menengah, yaitu bahasa komputer yang memakai campuran instruksi dalam kata-kata bahasa manusia (lihat contoh Bahasa Tingkat Tinggi di bawah) dan instruksi yang bersifat simbolik, contohnya {, }, ?, <<, >>, &&, ||, dsb.
4. Bahasa Tingkat Tinggi, yaitu bahasa komputer yang memakai instruksi berasal dari unsur kata-kata bahasa manusia, contohnya begin, end, if, for, while, and, or, dsb.

Sebagian besar bahasa pemrograman digolongkan sebagai Bahasa Tingkat Tinggi, salah satunya bahasa pemrograman Java, hanya bahasa C yang

digolongkan sebagai Bahasa Tingkat Menengah dan *Assembly* yang merupakan Bahasa Tingkat Rendah [10].

Bahasa pemrograman orientasi objek telah menjadi aliran utama (*mainstream*), Java berorientasi objek sejati melebihi C++. Segala sesuatu di Java kecuali sedikit tipe dasar (int, float, double, char) adalah objek [10].

Dibanding bahasa C++, Java lebih memberi kemudahan, antara lain:

1. Perancang Java menghilangkan keperluan dealokasi *manual*, Java dilengkapi *garbage collector* yang bertugas mendealokasi memori yang tidak diperlukan. Tidak ada lagi upaya pemrogram untuk dispose(). Pemrogram tidak dibebani urusan korupsi dan sampah memori.
2. Java menerapkan *array* sebenarnya, menghilangkan keperluan aritmatika *pointer* yang berbahaya dan berpeluang besar mengakibatkan kesalahan.
3. Menghilangkan keniscayaan operasi penugasan (*assignment*) sebagai pengujian atas kesamaan di kalimat bersyarat. Di C/ C++, pernyataan seperti

   (nPersons=3)…

   sering menjadi penyebab masalah, memaksudkan pengujian namun yang terjadi adalah pemberian nilai. Pada kasus ini, kompilator tidak berhak memberi peringatan karena pernyataan itu merupakan pernyataan yang sah.
4. Menghilangkan pewarisan jamak (*multiple inheritance*) diganti fasilitas *interface*. *Interface* memberi banyak manfaat antara lain untuk pewarisan

jamak tanpa dibebani kompleksitas yang muncul dari pengelolaan hirarki pewarisan jamak.

II.6.1 Bahasa Pemrograman Java

Bahasa Java merupakan karya Sun Microsystem Inc. Rilis resmi level beta dilakukan pada November 1995. Dua bulan berikutnya Netscape menjadi perusahaan pertama yang memperoleh lisensi bahasa Java dari Sun [10].



Gambar II.29 Logo Java

Pada 1996, Sun mengeluarkan JSDK (*Java Software Development Kit*), kemudian secara berturut-turut [10]:

1. Versi 1.02 yang mendukung konektivitas basis data dan objek-objek tersebar.
2. Versi 1.1 pada 1997 ditambah model kejadian (*event model*) yang andal, *internazionalization*, dan model komponen JavaBeans.
3. Versi 1.2 dirilis pada 1998 mempunyai banyak peningkatan diantaranya *user interface toolkit* "Swing" yang memungkinkan pemrograman membuat aplikasi berbasis GUI yang sepenuhnya portabel. Sejak ini disebut sebagai Java 2.
4. Versi 1.3 dirilis pada 2000, dengan banyak peningkatan.

5. Versi 1.4 ditambahkan fasilitas asersi untuk dukungan *design-by-contact.*
6. Versi 1.5 ditambahkan fitur-fitur baru di level bahasa diantaranya *generics* (*parameterized types*), *enumeration*,dan *metadata*.Versi ini disebut JDK 5.
7. Versi 1.5 ditambahkan *lightweight database system* yaitu *Derby. Derby* merupakan hasil pengembangan dari proyek basis data Apache. *Derby* mulanya merupakan CloudScape dari IBM.

II.6.2 Edisi Java

Sebagian besar pemrograman *modern* berdiri diatas pustaka-pustaka kelas yang telah ada untuk mendukung fungsionalitas. Pada bahasa Java, kelompok-kelompok kelas yang berkaitan erat dimasukan di satu paket, bervariasi sesuai dengan edisi Java. Masing-masing paket untuk maksud tertentu: *applet*, aplikasi standar, skala *enterprise*, dan produk konsumer [10].

Java adalah bahasa yang dapat dijalankan di berbagai *platform*, di beragam lingkungan: *internet*, *consumer electronics products*, dan *computer applications. The Java 2 Platform* tersedia dalam tiga edisi untuk keperluan berbeda sebagai berikut[10]:

1. Java 2 *Standard Edition* (J2SE)

   The Java 2 Platform, Standard Edition (J2SE) menyediakan lingkungan pengembangan yang kaya fitur, stabil, aman, dan *cross-platform*. Edisi ini mendukung konektivitas basis data, rancangan antarmuka pemakai,

masukan/keluaran, dan pemrograman jaringan dan termasuk sebagai paket-paket dasar bahasa Java.

2. Java 2 *Enterprise Edition* (J2EE)

   The Java 2 Platform, Enterprise Edition (J2EE) menyediakan kakas untuk membangun dan menjalankan *multitier enterprise applications*. J2EE berisi paket-paket di J2SE, ditambah paket-paket untuk mendukung pengembangan *Enterprise JavaBeans*, *Java Servlets*, *JavaServer Pages*, XML, dan kendali transaksi yang fleksibel.

3. Java 2 *Micro Edition* (J2ME)

   The Java 2 Platform, Micro Edition (J2ME) untuk beragam *consumer electronic product*, seperti *pager*, *smart card*, *cell phone*, *handheld* PDA, dan *set-top box*. J2ME sembari menyediakan bahasa Java yang sama, unggul dalam probabilitas, kemampuan dijalankan di mana pun dan *safe network delivery* seperti J2SE dan J2EE. J2ME menggunakan sekumpulan paket lebih kecil. J2ME berisi *subset* paket J2SE ditambah paket spesifik *Micro Edition* javax.microedition.io. Aplikasi J2ME dapat diskala agar juga dapat bekerja dengan J2SE dan J2EE.

II.6.3 *Java Development Kit*

Java Development Kit (JDK) berisi sekumpulan kakas baris perintah (command-line tool) untuk menciptakan program Java [10].

Berikut adalah daftar komponen utama Java Development Kit (JDK) :

1. Kompilator (**`javac`**)

2. *Interpreter* program java (**`java`**)
3. *Applet viewer* (**`appletviewer`**)
4. *Debugger* (**`jdb`**)
5. *Class file disassembler* (**`javap`**)
6. *Header and stub file generator* (**`javah`**)
7. *Documentation generator* (**`javadoc`**)
8. *Applet* demo
9. Kode sumber Java API

II.7 *Integrated Development Environment*

IDE (*Integrated Development Environment*) adalah program komputer yang memiliki beberapa fasilitas yang diperlukan dalam pembangunan perangkat lunak. Tujuan dari IDE adalah untuk menyediakan semua utilitas yang diperlukan dalam membangun perangkat lunak [10].

Sebuah IDE, atau secara bebas dapat diterjemahkan sebagai Lingkungan Pengembangan Terpadu, setidaknya memiliki fasilitas:

1. *Editor*, yaitu fasilitas untuk menuliskan kode sumber dari perangkat lunak.
2. *Compiler*, yaitu fasilitas untuk mengecek sintaks dari kode sumber kemudian mengubah dalam bentuk binari yang sesuai dengan bahasa mesin.
3. *Linker*, yaitu fasilitas untuk menyatukan data binari yang beberapa kode sumber yang dihasilkan *compiler* sehingga data-data binari tersebut

menjadi satu kesatuan dan menjadi suatu program komputer yang siap dieksekusi.

4. *Debuger*, yaitu fasilitas untuk mengetes jalannya program, untuk mencari *bug*/kesalahan yang terdapat dalam program.

Berikut ini adalah daftar IDE yang biasa dipakai dalam industri teknologi informasi:

1. Basic: MS Visual Basic
2. C: MS Visual C++
3. Delphi: Borland Delphi
4. Java: IntelliJ IDEA, Eclipse, NetBeans
5. Pascal: Turbo Pascal

II.7.1 Eclipse

Eclipse adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua *platform* (*platform-independent*). Berikut ini adalah sifat dari Eclipse[11]:

a. *Multi-platform*: Target sistem operasi Eclipse adalah Microsoft Windows, Linux, Solaris, AIX, HP-UX dan Mac OS X.
b. *Multi-language*: Eclipse dikembangkan dengan bahasa pemrograman Java, akan tetapi Eclipse mendukung pengembangan aplikasi berbasis bahasa pemrograman lainnya, seperti C/C++, Cobol, Python, Perl, PHP, dan lain sebagainya.

c. *Multi-role*: Selain sebagai IDE untuk pengembangan aplikasi, Eclipse pun bisa digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, test perangkat lunak, pengembangan web, dan lain sebagainya.

Eclipse pada saat ini merupakan salah satu IDE favorit dikarenakan gratis dan *open source*, yang berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Selain itu, kelebihan dari Eclipse yang membuatnya populer adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan komponen yang dinamakan plug-in.

II.7.2 Arsitektur Eclipse

Sejak versi 3.0, Eclipse pada dasarnya merupakan sebuah kernel, yang mengangkat *plug-in*. Apa yang dapat digunakan di dalam Eclipse sebenarnya adalah fungsi dari *plug-in* yang sudah diinstal. Ini merupakan basis dari Eclipse yang dinamakan *Rich Client Platform* (RCP). Berikut ini adalah komponen yang membentuk RCP[11]:

1. *Core platform*
2. OSGi
3. SWT (*Standard Widget Toolkit*)
4. Jface
5. Eclipse Workbench

Secara standar Eclipse selalu dilengkapi dengan JDT (*Java Development Tools*), *plug-in* yang membuat Eclipse kompatibel untuk mengembangkan

program Java, dan PDE (*Plug-in Development Environment*) untuk mengembangkan *plug-in* baru. Eclipse beserta *plug-in*-nya diimplementasikan dalam bahasa pemrograman Java.

Konsep Eclipse adalah IDE yang terbuka , mudah diperluas (*extensible*) untuk apa saja, dan tidak untuk sesuatu yang spesifik. Jadi, Eclipse tidak saja untuk mengembangkan program Java, akan tetapi dapat digunakan untuk berbagai macam keperluan, cukup dengan menginstal *plug-in* yang dibutuhkan. Apabila ingin mengembangkan program C/C++ terdapat *plug-in* CDT (C/C++ *Development Tools*). Selain itu, pengembangan secara visual bukan hal yang tidak mungkin oleh Eclipse, *plug-in* UML 2 tersedia untuk membuat diagram UML. Dengan menggunakan PDE setiap orang bisa membuat *plug-in* sesuai dengan keinginannya[11].

II.8 Pemodelan Sistem

Dalam pengembangan sistem berorientasi objek terdapat suatu alat atau *tool* pemodelan yang digunakan untuk membangun suatu perangkat lunak yang berorientasi objek yaitu *Unified Modeling Language* (UML).

II.8.1 Pemodelan Dengan Menggunakan UML

UML (*Unified Modeling Language*) adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma berorientasi objek. Pemodelan (*modeling*) sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kampleks sedemikian rupa sehingga lebih mudah dipelajari

dan dipahami. Tujuan pemodelan (dalam kerangka pengembangan sistem/perangkat lunak aplikasi) adalah sebagai sarana analisis, pemahaman, visualisasi, dan komunikasi antaranggota tim pengembang (saat seorang analis/perancang perangkat lunak bekerja dalam tim yang beranggotakan beberapa/banyak orang), serta sebagai sarana dokumentasi (yang bermanfaat untuk menelaah perilaku perangkat lunak secara seksama serta bermanfaat untuk melakukan pengujian terhadap perangkat lunak yang telah selesai dikembangkan)[12].

II.8.2 Diagram dan Teknik Pemodelan

Pemodelan dengan UML terdiri dari beberapa tipe diagram yang berbeda untuk memodelkan sistem perangkat lunak. Masing-masing diagram UML didesain untuk menunjukkan satu sisi dari bermacam-macam sudut pandang (perspektif) dan terdiri dari tingkat abstraksi yang berbeda, diagram-diagram tersebut antara lain [12]:

1. *Use case Diagram*
2. *Class Diagram*
3. *Statechart Diagram*
4. *Activity Diagram*
5. *Sequence Diagram*
6. *Collaboration diagram*
7. *Component diagram*
8. *Deployment diagram*

II.8.2.1 *Use case Diagram*

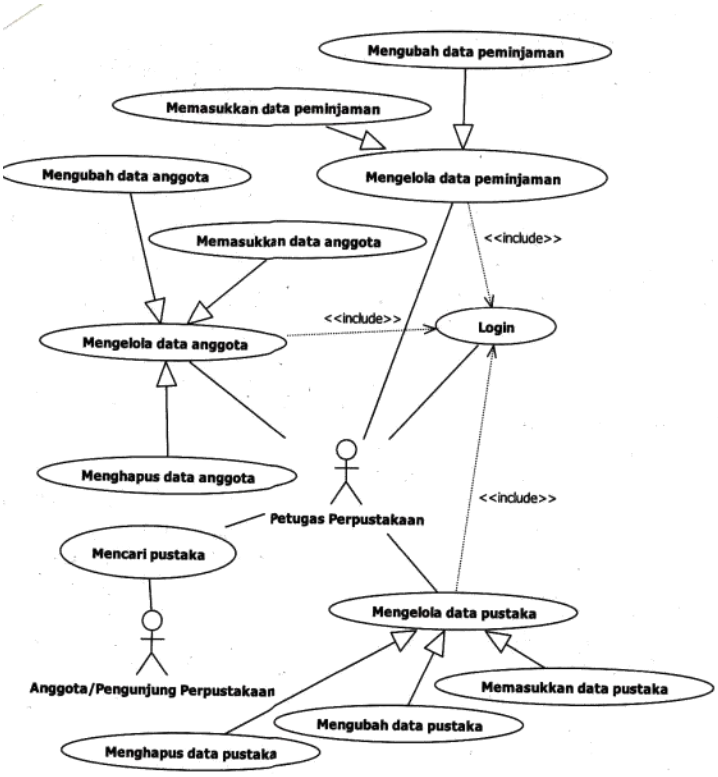
Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu [12].

Use case diagram dapat sangat membantu bila sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem.

Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common* [12].

Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

Contoh use case diagram :



Gambar II.30 Contoh *Use Case Diagram* [13]

II.8.2.2 *Class Diagram*

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi) [12].

Class diagram menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

Class memiliki tiga area pokok :

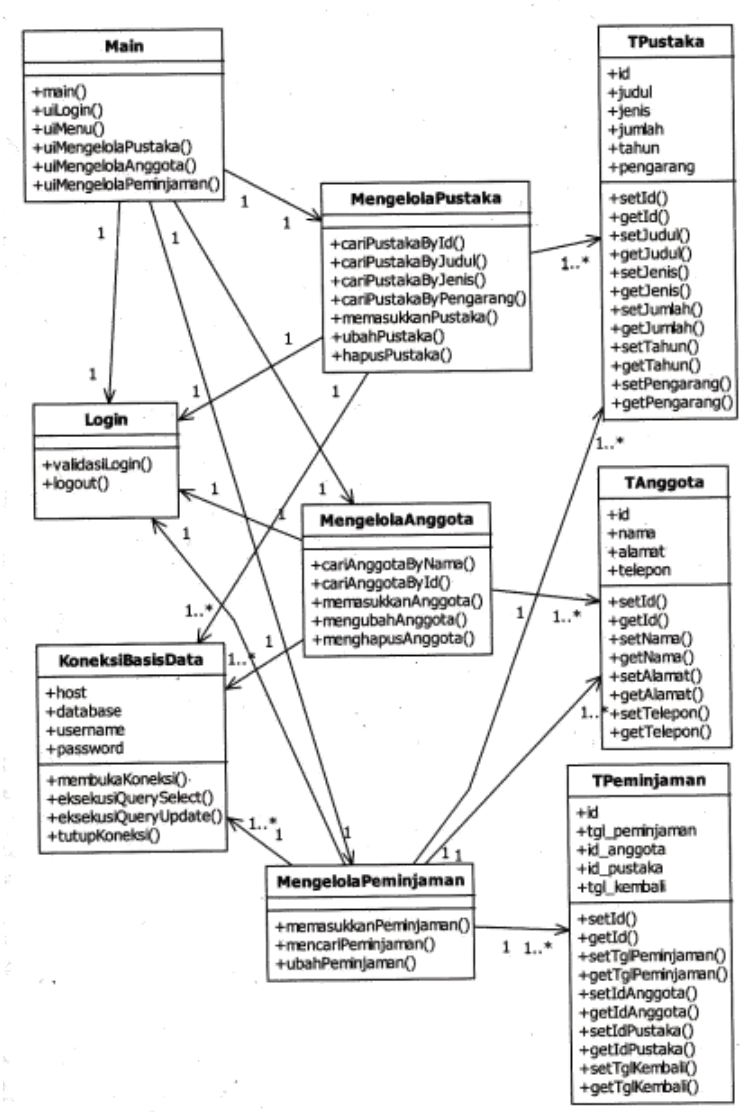
1. Nama
2. Atribut
3. Metoda

Atribut dan metoda dapat memiliki salah satu sifat berikut :

a. *Private,* tidak dapat dipanggil dari luar *class* yang bersangkutan.
b. *Protected,* hanya dapat dipanggil oleh *class* yang bersangkutan dan anak-anak yang mewarisinya.
c. *Public*, dapat dipanggil oleh siapa saja.

Class dapat merupakan implementasi dari sebuah *interface,* yaitu *class* abstrak yang hanya memiliki metoda. *Interface* tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class.* Dengan demikian *interface* mendukung resolusi metoda pada saat *run-time*.

Contoh *class diagram* :



Gambar II.31 Contoh *Class Diagram* [13]

II.8.2.3 *Activity Diagram*

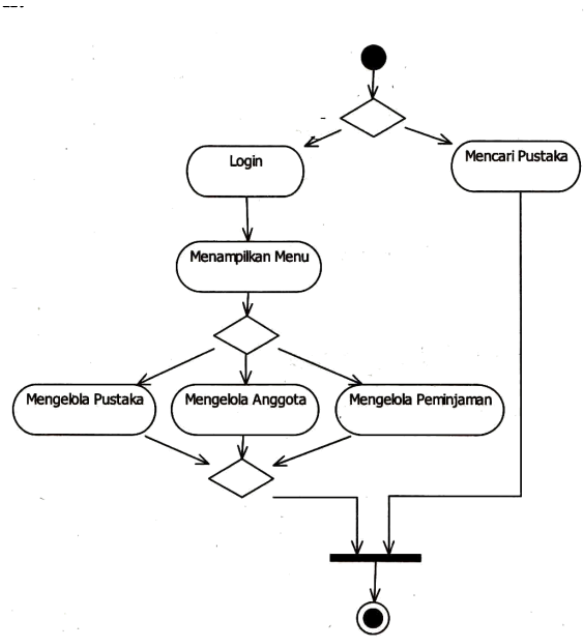
Activity diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi[12].

Activity diagram merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas.

Sama seperti *state*, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal.

Activity diagram dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.
Contoh *activity diagram* tanpa *swimlane*:



Gambar II.32 Contoh Activity Diagram tanpa Swimlane [13]

II.8.2.4 *Sequence Diagram*

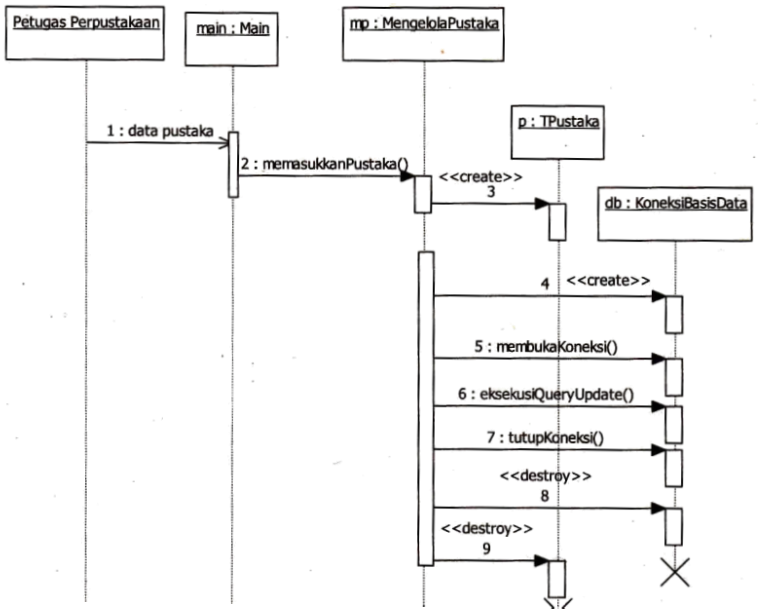
Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan [12].

Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal. *Message* digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, *message* akan dipetakan menjadi operasi/metoda dari

class. *Activation bar* menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah message. Untuk objek-objek yang memiliki sifat khusus, standar UML mendefinisikan *icon* khusus untuk objek *boundary*, *controller* dan *persistent entity*.

Contoh *sequence diagram* :



Gambar II.33 Contoh *Sequence Diagram* [13]

II.9 Skala Pengukuran

Skala pengukuran merupakan kesepakatan yang digunakan sebagai acuan untuk menentukan panjang pendeknya interval yang ada dalam alat ukur, sehingga alat ukur tersebut bila digunakan dalam pengukuran akan menghasilkan data kuantitatif. Dengan skala pengukuran ini, maka nilai variabel yang diukur 10 dengan instrumen tertentu dapat dinyatakan dalam bentuk angka, sehingga akan lebih akurat, efisien, dan komunikatif. Berbagai skala yang dapat digunakan untuk penelitian adalah [14]:

II.9.1 Skala *Likert*

Skala *Likert* digunakan untuk mengukur sikap, pendapat, dan persepsi seseorang atau sekelompok orang tentang fenomena sosial. Dengan skala *Likert*, maka variabel yang akan diukur dijabarkan menjadi indikator variabel. Kemudian indikator tersebut dijadikan sebagai titik tolak untuk menyusun item–item instrumen yang dapat berupa pernyataan atau pertanyaan[14]. Jawaban setiap item instrumen yang menggunakan skala likert mempunyai gradasi dari sangat positif sampai sangat negatif, yang dapat berupa kata–kata antara lain: sangat setuju dengan skor 5, setuju dengan skor 4, ragu–ragu dengan skor 3, tidak setuju dengan skor 2, sangat tidak setuju dengan skor1.

Instrument peneletian yang menggunakan skla likert dapat dibuat dalam bentuk *checklist*.

a. Contoh Bentuk *Checklist*

Berilah jawaban pertanyaan berikut sesuai dengan pernyataan anda, dengan cara memberi tanda (√) pada kolom yang tersedia.

Tabel II.1 Tabel Pertanyaan Skala *Likert*

No	Pertanyaan	Jawaban				
		SS	ST	RG	TS	STS
1	Prosedur kerja yang baru itu akan segera diterapkan di perusahaan anda.		√			
2	.....					

SS = Sangat Setuju diberi skor 5

ST = Setuju diberi skor 4

RG = Ragu-ragu	diberi skor	3
TS = Tidak Setuju	diberi skor	2
STS = Sangat Tidak Setuju	diberi skor	1

Kemudian dengan teknik pengumpulan data angket, maka instrument tersebut misalnya diberikan kepada 100 orang karyawan yang diambil secara *random*. Dari 100 orang pegawai setelah dilakukan analis misalnya:

25	Orang menjawab	SS
40	Orang menjawab	ST
5	Orang menjawab	RG
20	Orang menjawab	TS
10	Orang menjawab	STS

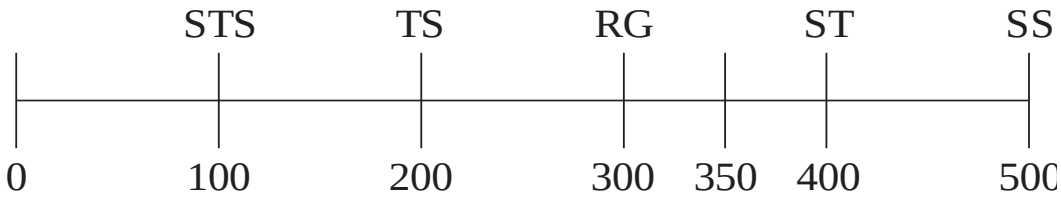
Berdasarkan data tersebut 65 orang (40+25) atau 65% karyawan menjawab setuju dan sangat setuju. Jadi kesimpulannya mayoritas karyawan setuju adanya metode kerja baru.

Data interval tersebut juga dapat dianalisis dengan menghitung rata-rata jawaban berdasarkan skoring setiap jawaban dari responden. Berdasarkan skor yang telah ditetapkan dapat dihitung sebagai berikut :

Jumlah skor untuk 25 orang yang menjawab SS	= 25 x 5= 125
Jumlah skor untuk 40 orang yang menjawab ST	= 40 x 4= 160
Jumlah skor untuk 5 orang yang menjawab RG	= 5 x 3 = 15
Jumlah skor untuk 20 orang yang menjawab TS	= 20 x 2= 40
Jumlah skor untuk 10 orang yang menjawab STS	= 10 x 1= 10
Jumlah total	= 350

Jumlah skor ideal (kriterium) untuk seluruh item = 5 x 100 = 500 (seandainya semua menjawab SS). Jumlah skor yang diperoleh dari penelitian 350. Jadi berdasarkan data tu maka tingkat persetujuan terhadap metode kerja baru itu = (350 : 500) x 100% = 70% dari yang diharapkan (100%)

Secara kontinum dapat digambarkan seperti berikut :



Gambar II.34 Gambar interpretasi skor persetujuan metode kerja baru

Jadi berdasarkan data yang diperoleh dari 100 responden maka rata-rata 350 terletak pada daerah setuju.