BAB 2
LANDASAN TEORI

2.1. Android

Android adalah sistem operasi yang berbasis Linux untuk telepon seluler seperti telepon pintar dan komputer tablet. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak.Awalnya, Google Inc. membeli Android Inc., pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan Android, dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia. Pada saat perilisan perdana Android, 5 November 2007, Android bersama Open Handset Alliance menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, Google merilis kode–kode Android di bawah lisensi Apache, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler. Di dunia ini terdapat dua jenis distributor sistem operasi Android.Pertama yang mendapat dukungan penuh dari Google atau Google Mail Services (GMS) dan kedua adalah yang benar–benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai Open Handset Distribution (OHD).

Pada Juli 2005, Google bekerjasama dengan Android Inc., perusahaan yang berada di Palo Alto, CaliforniaAmerika Serikat.Para pendiri Android Inc. bekerja pada Google, di antaranya Andy Rubin, Rich Miner, Nick Sears, dan Chris White.Saat itu banyak yang menganggap fungsi Android Inc. hanyalah sebagai perangkat lunak pada telepon seluler.Sejak saat itu muncul rumor bahwa Google hendak memasuki pasar telepon seluler. Di perusahaan Google, tim yang dipimpin Rubin bertugas mengembangkan program perangkat seluler yang didukung oleh kernel Linux. Hal ini menunjukkan indikasi bahwa Google sedang bersiap menghadapi persaingan dalam pasar telepon seluler.

Pada 9 Desember 2008, diumumkan anggota baru yang bergabung dalam program kerja Android ARM Holdings, Atheros Communications, diproduksi oleh Asustek Computer Inc, Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, dan Vodafone Group Plc. Seiring pembentukan Open Handset Alliance, OHA mengumumkan produk perdana mereka, Android, perangkat bergerak (*mobile*) yang merupakan modifikasi kernel Linux 2.6. Sejak Android dirilis telah dilakukan berbagai pembaruan berupa perbaikan *bug* dan penambahan fitur baru. Telepon pertama yang memakai sistem operasi Android adalah HTC Dream, yang dirilis pada 22 Oktober 2008. Pada penghujung tahun 2009 diperkirakan di dunia ini paling sedikit terdapat 18 jenis telepon seluler yang menggunakan Android [1].

1. Android Versi 1.1

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

2. Android Versi 1.5 (Cupcake)

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

3. Android versi 1.6 (Donut)

   Donut (versi 1.6) dirilis pada September dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai indikator dan kontrol applet VPN. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus, *camcorder* dan galeri yang dintegrasikan; CDMA / EVDO, 802.1x, VPN,

Gestures, dan *Text-to-speech engine*; kemampuan dial kontak; teknologi *text to change speech* (tidak tersedia pada semua ponsel; pengadaan resolusi VWGA.

4. Android versi 2.0/2.1 (Éclair)

   Pada 3 Desember 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1 (Eclair), perubahan yang dilakukan adalah pengoptimalan *hardware*, peningkatan Google Maps 3.1.2, perubahan UI dengan *browser* baru dan dukungan HTML5, daftar kontak yang baru, dukungan *flash* untuk kamera 3,2 MP, digital Zoom, dan Bluetooth 2.1. Untuk bergerak cepat dalam persaingan perangkat generasi berikut, Google melakukan investasi dengan mengadakan kompetisi aplikasi *mobile* terbaik (*killer apps* - aplikasi unggulan).Kompetisi ini berhadiah $25,000 bagi setiap pengembang aplikasi terpilih.Kompetisi diadakan selama dua tahap yang tiap tahapnya dipilih 50 aplikasi terbaik.Dengan semakin berkembangnya dan semakin bertambahnya jumlah *handset* Android, semakin banyak pihak ketiga yang berminat untuk menyalurkan aplikasi mereka kepada sistem operasi Android.Aplikasi terkenal yang diubah ke dalam sistem operasi Android adalah Shazam, Backgrounds, dan WeatherBug.Sistem operasi Android dalam situs internet juga dianggap penting untuk menciptakan aplikasi Android asli, contohnya oleh MySpace dan Facebook.

5. Android 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, intergrasi V8 JavaScript*engine* yang dipakai Google Chrome yang mempercepat kemampuan rendering pada browser, pemasangan aplikasi dalam SD Card, kemampuan WiFi Hotspot *portable*, dan kemampuan *auto update* dalam aplikasi Android Market.

6. Android 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 *formatvideo* VP8 dan WebM, efek audio baru (*reverb, equalization, headphone virtualization,* dan *bass boost*), dukungan kemampuan *Near Field Communication* (NFC), dan dukungan jumlah kamera yang lebih dari satu.

7. Android 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. *Tablet* pertama yang dibuat dengan menjalankan Honeycomb adalah Motorola Xoom. Perangkat *tablet* dengan *platform*Android 3.0 akan segera hadir di Indonesia. Perangkat tersebut bernama Eee Pad Transformer produksi dari Asus.Rencana masuk pasar Indonesia pada Mei 2011.

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, perangkat tambahan fotografi, mencari email secara *offline*, dan berbagi informasi dengan menggunakan NFC.

2.1.1. Fitur

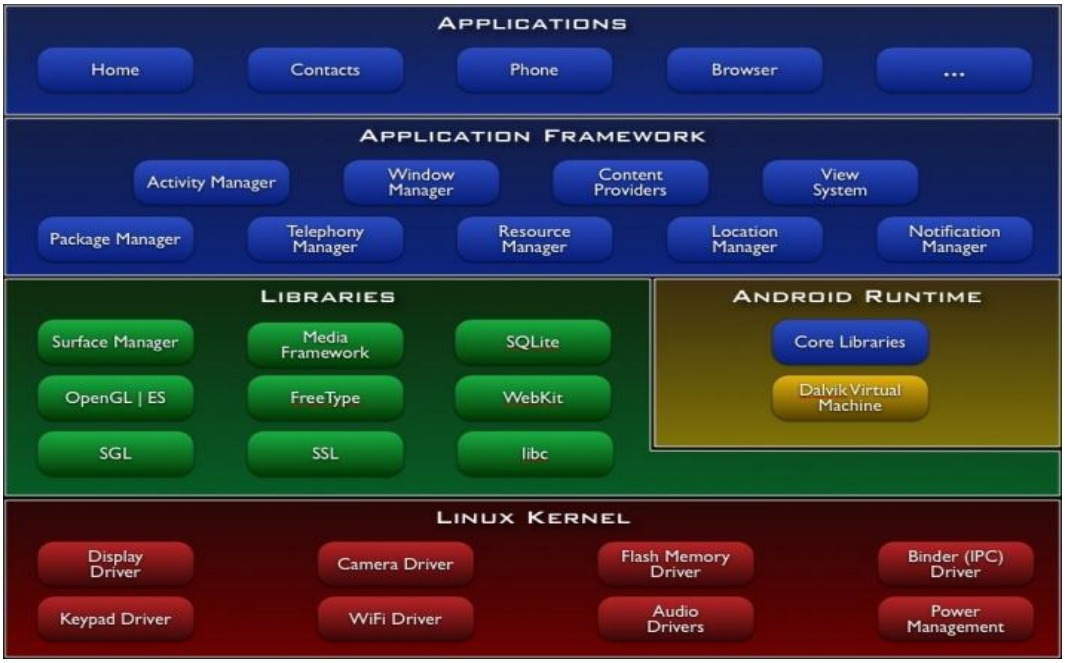
Fitur yang tersedia pada platform Android saat ini [2], antara lain :

a. *Framework* aplikasi yang mendukung penggantian komponen dan *reusable*.

b. Dalvik mesin virtual: mesin virtual dioptimalkan untuk perangkat *mobile*.
c. Grafik: grafik di 2D dan grafis 3D berdasarkan pustaka OpenGL.
d. SQLite: untuk penyimpanan data.
e. Mendukung media: *audio*, *video*, dan berbagai format gambar (MPEG4, H.264, MP3,AAC, AMR, JPG, PNG, GIF).
f. GSM, Bluetooth, EDGE, 3G, dan WiFi (*hardware dependent*).
g. Kamera, *Global Positioning System* (GPS), kompas, dan *accelerometer* (tergantung hardware).

2.1.2. Arsitektur Android

Dalam paket sistem operasi android tediri dari beberapa unsur seperti tampak pada gambar di bawah. Secara sederhana arsitektur android merupakan sebuah kernel Linux dan sekumpulan pustaka C / C++ dalam suatu *framework* yang menyediakan dan mengatur alur proses aplikasi.[Google IO, Android Anatomy and Physiology] [2]. Gambar arsitektur android dapat dilihat pada gambar 2.1.



Gambar 2.1 Arsitektur Android

2.1.2.1 Linux Kernel

Android dibangun di atas kernel Linux 2.6.Namun secara keseluruhan android bukanlah linux, karena dalam android tidak terdapat paket standar yang dimiliki oleh linux lainnya. Linux merupakan sistem operasi terbuka yang handal dalam manajemen memori dan proses. Oleh karenanya pada android hanya terdapat beberapa layanan yang diperlukan seperti keamanan, manajemen memori, manajemen proses, jaringan dan *driver*.Kernel linux menyediakan *driver*layar, kamera, *keypad*, WiFi, *Flash Memory*, *audio*, dan IPC (*Interprocess Communication*) untuk mengatur aplikasi dan lubang keamanan.

2.1.2.2 *Libraries*

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. 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. SSl dan WebKit untuk browser dan keamanan internet.

Pustaka-pustaka tersebut bukanlah aplikasi yang berjalan sendiri, namun hanya dapat digunakan oleh program yang berada di level atasnya.Sejak versi Android 1.5, pengembang dapat membuat dan menggunakan pustaka sendiri menggunakan *Native Development Toolkit* (NDK).

2.1.2.3 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. Dalam Android *Runtime* terdapat 2 bagian utama, diantaranya:

1. **Pustaka Inti**, 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 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.

2.1.2.4 *Framework* Aplikasi

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

a. *Activity Manager*, berfungsi untuk mengontrol siklus hidup aplikasi dan menjaga keadaan ”*Backstack*“ untuk navigasi penggunaan.
b. *Content Providers*, berfungsi untuk merangkum data yang memungkinkan digunakan oleh aplikasi lainnya, seperti daftar nama.
c. *Resuource Manager*, untuk mengatur sumber daya yang ada dalam program. Serta menyediakan akses sumber daya diluar kode program, seperti karakter, grafik, dan file layout.
d. *Location Manager*, berfungsi untuk memberikan informasi detail mengenai lokasi perangkat android berada.

e. *Notification Manager*, mencakup berbagai macam peringatan seperti, pesan masuk, janji, dan lain sebagainya yang akan ditampilkan pada status bar.

2.1.2.5 *Application Layer*

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.

Lapisan aplikasi android sangat berbeda dibandingkan dengan sistem operasi lainnya. Pada android semua aplikasi, baik aplikasi inti (*native*) maupun aplikasi pihak ketiga berjalan diatas lapisan aplikasi dengan menggunakan pustaka API (*Application Programming Interface*) yang sama.

2.2. Java

Java adalah bahasa pemrograman berorientasi objek yang dikembangkan oleh Sun Microsystems sejak tahun 1991.Bahasa ini dikembangkan dengan model yang mirip dengan bahasa C++ dan Smalltalk, namun dirancang agar lebih mudah dipakai dan *platform independent*, yaitu dapat dijalankan di berbagai jenis sistem operasi dan arsitektur komputer [8].

2.2.1. Sejarah Java

Bahasa pemrograman Java terlahir dari The Green Project, yang berjalan selama 18 bulan, dari awal tahun 1991 hingga musim panas 1992. Proyek tersebut belum menggunakan versi yang dinamakan Oak.Proyek ini dimotori oleh Patrick Naughton, Mike Sheridan, James Gosling dan Bill Joy, beserta sembilan pemrogram lainnya dari Sun Microsystems. Salah satu hasil proyek ini adalah maskot Dukeyang dibuat oleh Joe Palrang.

Pertemuan proyek berlangsung di sebuah gedung perkantoran Sand Hill Road di Menlo Park. Sekitar musim panas 1992 proyek ini ditutup dengan menghasilkan sebuah program Java Oak pertama, yang ditujukan sebagai pengendali sebuah peralatan dengan teknologi layar sentuh (*touch screen*), seperti pada PDA sekarang ini. Teknologi baru ini dinamai "*7" (*Star Seven*).

Setelah era *Star Seven* selesai, sebuah anak perusahaan Tv kabel tertarik ditambah beberapa orang dari proyek The Green Project. Mereka memusatkan kegiatannya pada sebuah ruangan kantor di 100 Hamilton Avenue, Palo Alto.

Perusahaan baru ini bertambah maju: jumlah karyawan meningkat dalam waktu singkat dari 13 menjadi 70 orang. Pada rentang waktu ini juga ditetapkan pemakaian Internet sebagai medium yang menjembatani kerja dan ide di antara mereka.Pada awal tahun 1990-an, Internet masih merupakan rintisan, yang dipakai hanya di kalangan akademisi dan militer.

Mereka menjadikan perambah (*browser*) Mosaic sebagai landasan awal untuk membuat perambah Java pertama yang dinamai Web Runner, terinsipirasi dari film 1980-an, Blade Runner. Pada perkembangan rilis pertama, Web Runner berganti nama menjadi Hot Java.

Pada sekitar bulan Maret1995, untuk pertama kali kode sumber Java versi 1.0a2 dibuka. Kesuksesan mereka diikuti dengan untuk pemberitaan pertama kali pada surat kabar *San Jose Mercury News* pada tanggal 23 Mei1995.

Sayang terjadi perpecahan di antara mereka suatu hari pada pukul 04.00 di sebuah ruangan hotel Sheraton Palace.Tiga dari pimpinan utama proyek, Eric Schmidt dan George Paolini dari Sun Microsystems bersama Marc Andreessen, membentuk Netscape.

Nama Oak, diambil dari pohon oak yang tumbuh di depan jendela ruangan kerja "Bapak Java", James Gosling. Nama Oak ini tidak dipakai untuk versi release Java karena sebuah perangkat lunak lain sudah terdaftar dengan merek dagang tersebut, sehingga diambil nama penggantinya menjadi "Java". Nama ini diambil dari kopi murni yang digiling langsung dari biji (kopi tubruk) kesukaan Gosling.Konon kopi ini berasal dari Pulau Jawa. Jadi nama bahasa pemrograman Java tidak lain berasal dari kata Jawa (bahasa Inggris untuk Jawa adalah Java).

2.2.2. Versi Awal

Versi awal Java ditahun 1996 sudah merupakan versi release sehingga dinamakan Java Versi 1.0. Java versi ini menyertakan banyak paket standar awal yang terus dikembangkan pada versi selanjutnya:

1. java.lang: Peruntukan kelas elemen-elemen dasar.
2. java.io: Peruntukan kelas *input* dan *output*, termasuk penggunaan berkas.
3. java.util: Peruntukan kelas pelengkap seperti kelas struktur data dan kelas kelas penanggalan.
4. java.net: Peruntukan kelas TCP/IP, yang memungkinkan berkomunikasi dengan komputer lain menggunakan jaringan TCP/IP.
5. java.awt: Kelas dasar untuk aplikasi antarmuka dengan pengguna (GUI)
6. java.applet: Kelas dasar aplikasi antar muka untuk diterapkan pada penjelajah web.

2.2.3. Kelebihan

1. *Multiplatform*. Kelebihan utama dari Java ialah dapat dijalankan di beberapa *platform* / sistem operasi komputer, sesuai dengan prinsip tulis sekali, jalankan di mana saja. Dengan kelebihan ini pemrogram cukup menulis sebuah program Java dan dikompilasi (diubah, dari bahasa yang dimengerti manusia menjadi bahasa mesin / *bytecode*) sekali lalu hasilnya dapat dijalankan di atas beberapa platform tanpa perubahan. Kelebihan ini memungkinkan sebuah program berbasis java dikerjakan diatas *operatingsystem* Linux tetapi dijalankan dengan baik di atas Microsoft Windows. *Platform* yang didukung sampai saat ini adalah Microsoft Windows, Linux, Mac OS dan Sun Solaris. Penyebanya adalah setiap sistem operasi menggunakan programnya sendiri-sendiri (yang dapat diunduh dari situs Java) untuk meninterpretasikan *bytecode* tersebut.
2. OOP (*Object Oriented Programming* - Pemrogram Berorientasi Objek)

3. Perpustakaan Kelas Yang Lengkap, Java terkenal dengan kelengkapan *library*/perpustakaan (kumpulan program program yang disertakan dalam pemrograman java) yang sangat memudahkan dalam penggunaan oleh para pemrogram untuk membangun aplikasinya. Kelengkapan perpustakaan ini ditambah dengan keberadaan komunitas Java yang besar yang terus menerus membuat perpustakaan-perpustakaan baru untuk melingkupi seluruh kebutuhan pembangunan aplikasi.
4. Bergaya C++, memiliki sintaks seperti bahasa pemrograman C++ sehingga menarik banyak pemrogram C++ untuk pindah ke Java. Saat ini pengguna Java sangat banyak, sebagian besar adalah pemrogram C++ yang pindah ke Java. Universitas-universitas di Amerika Serikat juga mulai berpindah dengan mengajarkan Java kepada murid-murid yang baru karena lebih mudah dipahami oleh murid dan dapat berguna juga bagi mereka yang bukan mengambil jurusan komputer.
5. Pengumpulan sampah otomatis, memiliki fasilitas pengaturan penggunaan memori sehingga para pemrogram tidak perlu melakukan pengaturan memori secara langsung (seperti halnya dalam bahasa C++ yang dipakai secara luas).

2.2.4. Kekurangan

1. Tulis sekali, jalankan di mana saja - Masih ada beberapa hal yang tidak kompatibel antara *platform* satu dengan *platform* lain. Untuk J2SE, misalnya *SWT-AWT bridge* yang sampai sekarang tidak berfungsi pada Mac OS X.
2. Mudah didekompilasi. Dekompilasi adalah proses membalikkan dari kode jadi menjadi kode sumber. Ini dimungkinkan karena kode jadi Java merupakan *bytecode* yang menyimpan banyak atribut bahasa tingkat tinggi, seperti nama-nama kelas, metode, dan tipe data. Hal yang sama juga terjadi pada Microsoft .NET *Platform*. Dengan demikian, algoritma yang digunakan program akan lebih sulit disembunyikan dan mudah dibajak/di-*reverse-engineer*.

3. Penggunaan memori yang banyak. Penggunaan memori untuk program berbasis Java jauh lebih besar daripada bahasa tingkat tinggi generasi sebelumnya seperti C/C++ dan Pascal (lebih spesifik lagi, Delphi dan Object Pascal). Biasanya ini bukan merupakan masalah bagi pihak yang menggunakan teknologi terbaru (karena trend memori terpasang makin murah), tetapi menjadi masalah bagi mereka yang masih harus berkutat dengan mesin komputer berumur lebih dari 4 tahun.

2.2.5. Tahap Kompilasi Java

1. Tulis / Ubah. Pemrogram menulis program dan menyimpannya di media dalam bentuk berkas '.java'.
2. Kompilasi. Pengkompilasi membentuk *bytecodes* dari program menjadi bentuk berkas '.class'.
3. Muat. Pemuat kelas memuat *bytecodes* ke memori.
4. Verifikasi. Peng-verifikasi memastikan *bytecodes* tidak mengganggu sistem keamanan Java.
5. Jalankan. Penerjemah menerjemahkan *bytecodes* ke bahasa mesin.

2.3. XML

***XML (Extensible Markup Language*)** adalah bahasa markup untuk keperluan umum yang disarankan oleh W3C untuk membuat dokumen markup keperluan pertukaran data antar sistem yang beraneka ragam.XML merupakan kelanjutan dari HTML (*HyperText Markup Language*) yang merupakan bahasa standar untuk melacak Internet.

XML didesain untuk mempu menyimpan data secara ringkas dan mudah diatur.Kata kunci utama XML adalah data (jamak dari datum) yang jika diolah bisa memberikan informasi. XML menyediakan suatu cara terstandarisasi namun bisa dimodifikasi untuk menggambarkan isi dari dokumen. Dengan sendirinya, XML dapat digunakan untuk menggambarkan sembarang *view database*, tetapi dengan suatu cara yang standar.

XML memiliki tiga tipe *file,* diantaranya :

1. XML, merupakan standar format dari struktur berkas (*file*).
2. XSL, merupakan standar untuk memodifikasi data yang diimpor atau diekspor.
3. XSD, merupakan standar yang mendefinisikan struktur *database* dalam XML.

Keunggulan XML bisa diringkas sebagai berikut :

1. Pintar (*Intelligence*). XML dapat menangani berbagai tingkat (*level*) kompleksitas.
2. Dapat beradaptasi. Dapat mengadaptasi untuk membuat bahasa sendiri. Seperti Microsoft membuaat bahasa MSXML atau Macromedia mengembangkan MXML.
3. Mudah pemeliharaannya.
4. Sederhana. XML lebih sederhana.
5. Mudah dipindah-pindahkan (*Portability*). XML mempunyai kemudahan perpindahan (*portabilitas*) yang lebih bagus.

2.4. JSON

JSON (*JavaScript Object Notation*) merupakan format yang ringan untuk memasukan data ke dalam sebuah variabel.Sangat mudah dimengerti dan diimplementasikan oleh manusia, dan mudah juga untuk komputer dalam melakukan parsingnya.

JSON merupakan bagian dari bahasa pemrograman JavaScript (Standard ECMA-262 3rd Edition – December 1999).JSON merupakan format teks yang sepenuhnya independen tetapi menggunakan konvensi yang familiar dengan bahasa pemrograman dari keluarga-C, termasuk C, C++, C#, Java, JavaScript, Perl, Python, dan sebagainya.Kelebihan inilah yang membuat JSON menjadi sebuah bahasa *data-interchange* yang ideal.

JSON dibangun dalam dua struktur:

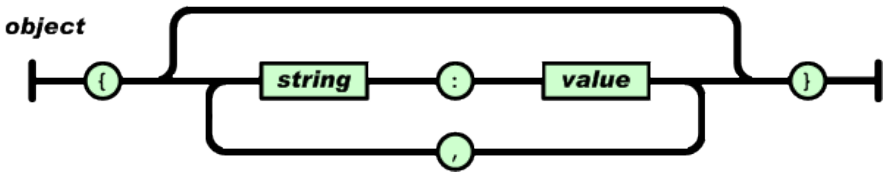
1. Beberapa pasangan dari nama/nilai. Dalam beberapa bahasa perograman biasa disebut dengan istilah *object*, *record*, *struct*, tabel *hash*, *key list* atau *associative array*.
2. Nilai-nilai yang terusun secara *ordered list*. Biasa disebut dengan *array*, *vector*,*list* atau daftar dalam bahasa pemrograman.

JSON adalah struktur data yang universal, dalam artian bisa digunakan dalam berbagai bahasa pemrograman.Hampir semua bahasa pemrograman mendukung penuh JSON dalam berbagai format.Hal ini memungkinkan format data yang dapat dipertukarkan menggunakan bahasa pemrograman juga menggunakan dasar dari struktur JSON [10].

Format data JSON mempunyai aturan sebagai berikut:

1. Object

Object adalah satu set nama/nilai yang tidak terurut (*An object is an unordered set of name/value pairs*). Penulisan object dimulai dengan tanda { (*left brace*) dan diakhiri dengan tanda } (*right brace*). Setiap nama diikuti oleh tanda : (*colon*) dan pasangan nama/nilai dipisahkan dengan tanda , (*comma*). Ilustrasi dari object JSON dapat dilihat pada gambar 2.2.

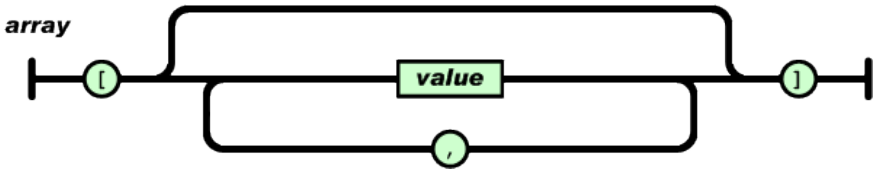


Gambar 2.2 *Object* JSON

2. Array

Array adalah sekumpulan nilai yang teratur (*An array is an ordered collection of values*). Penulisan sebuah array dimulai dengan tanda [(*left bracket*) dan diakhiri dengan tanda] (*right bracket*). Nilai dipisahkan

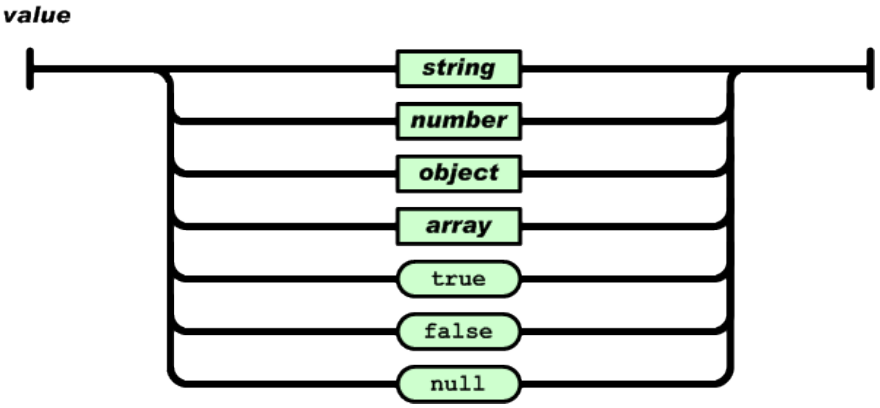
menggunakan tanda , (*comma*). Ilustrasi dari array JSON dapat dilihat pada gambar 2.3.



Gambar 2.3 Array JSON

3. Nilai

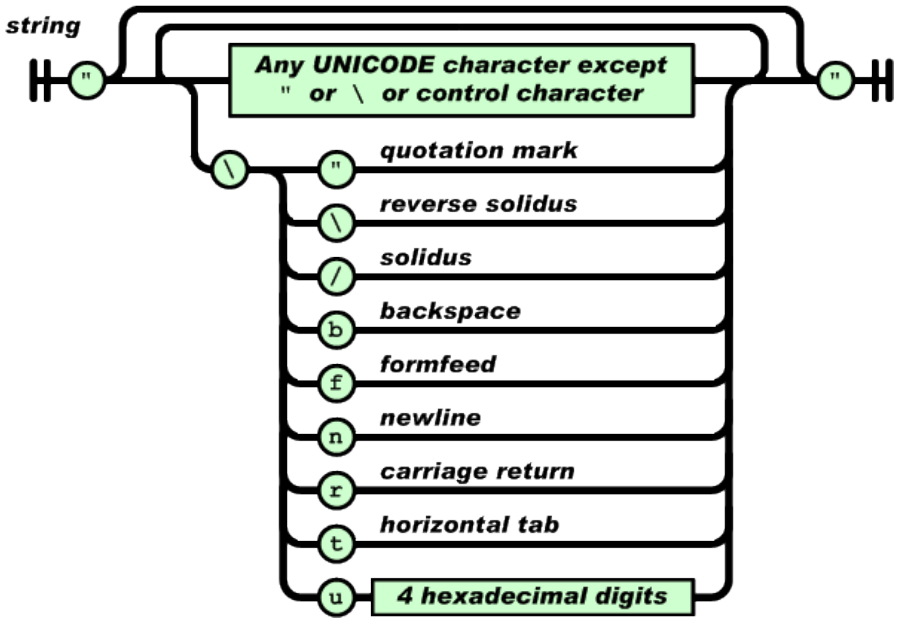
Nilai bisa berupa *string* dalam tanda kutip, atau *number* (angka), *TRUE* atau *FALSE* atau *NULL*, sebuah *object* atau sebuah *array*.Struktur ini dapat ditulis menggunakan metode bersarang.Ilustrasi dari nilai JSON dapat dilihat pada gambar 2.4.



Gambar 2.4 Nilai JSON

4. String

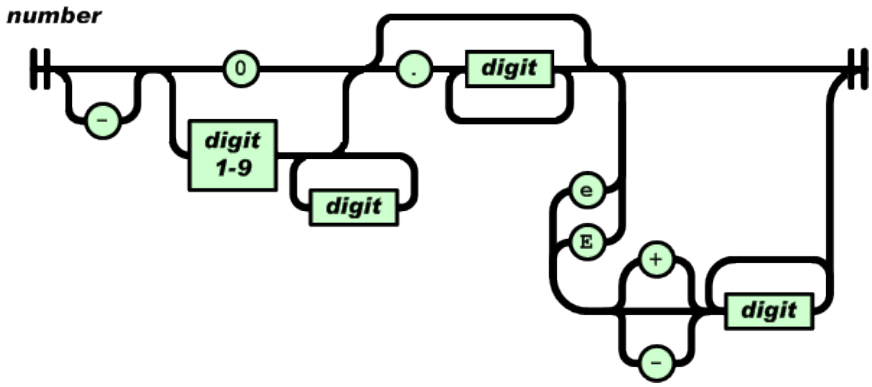
String adalah rangkaian atau urutan karakter unicode yang berada daslam tanda kutip, bisa juga hanya berisi karakter kosong, menggunakan tanda \ (*backslash*) untuk *escape*.Karakter di representasikan sebagai *string* tunggal. Tipe data string pada JSON sangat mirip dengan definisi pada bahasa C atau Java. Ilustrasi dari *string* JSON dapat dilihat pada gambar 2.5.



Gambar 2.5 *String* JSON

5. Number

Number sangat mirip dengan definisi pada bahasa C atau Java, hanya saja tipe bilangan oktal dan heksadesimal tidak digunakan.Ilustrasi dari *number* JSON dapat dilihat pada gambar 2.6.



Gambar 2.6 *Number* JSON

6. Ruang Kosong (*White Space*)

Ruang kosong (*Whitespace*) dapat diselipkan diantara pasangan atau token apa saja. Kecuali *detail encoding*, yang secara jelas akan berfungsi untuk menjelaskan bahasa yang digunakan.

2.5. Bioskop

Kata BIOSKOP berasal dari kata ***BOSCOOP*** (bahasa Belanda yang juga berasal dari Bahasa Yunani) yang artinya gambar hidup.Bioskop sendiri adalah tempat untuk menonton pertunjukan film dengan menggunakan layar lebar, gambar film dengan menggunakan *proyektor*.Bioskop pertama kali di Indonesia berdiri pada bulan Desember 1900, di jalan Tanah Abang 1, Jakarta Pusat.Dan uniknya harga karcis kelas satu pada saat itu adalah dua gulden (perak) dan harga karcis kelas dua, setengah gulden (perak).

2.6. Film

Film dapat dikatakan sebagai suatu penemuan teknologi modernpaling spektakuler yang melahirkan berbagai kemungkinan. Pertama,dalam pengertian kimia fisik dan teknik, film berarti selaput halus.Pengertian ini dapat dicontohkan, misalnya pada selaput tipis cat atau padalapisan tipis yang biasa dipakai untuk melindungi benda-benda sepertidokumen (laminasi).Dalam fotografi dan sinematografi film berarti bahanyang dipakai untuk segala sesuatu yang berkaitan dengan foto.Kedua, film juga mempunyai pengertian paling umum, yaitu untuk menamakanserangkaian gambar yang diambil dari obyek yang bergerak. Gambar obyek itu memperlihatkan suatu serial gerakan atau momen yangberlangsung secara terus-menerus, kemudian diproyeksikan ke dalamsebuah layer dengan memutarnya dalam kecepatan tertentu sehinggamenghasilkan sebuah gambar hidup.

Film atau gambar hidup merupakan gambar-gambar dalam frame di mana frame demi frame diproyeksikan melalui lensa proyektor secara mekanis sehingga pada layer terlihat gambar itu hidup.Film itu bergerak dengan cepat dan bergantian sehingga memberikan visual yang kontinyu.Film juga merupakan

serangkaian gambar-gambar yang diambil dari obyek yang bergerak memperlihatkan suatu serial peristiwa-peristiwa gerakan yang berlaku secara berkesinambungan, yang berfungsi sebagai media hiburan, pendidikan, dan penerangan. Sebagai salah satu media informasi maka film secara otomatis akan membawa dampak (*side effect*), baik itu positif maupun negatif kepada penontonnya.

Menurut Dr. Phil Astrid Susanto, film adalah gambar yang bergerak dikenal dengan gambar hidup dan memang gerakan itu merupakan unsur pemberi hidup kepada suatu gambar, namun betapapun sempurnanya dan modernnya teknik yang dipergunakan belum mendekati kenyataan hidup sehari-hari sebagaimana film. Untuk meningkatkan kesan dan dampak dari film, suatu film diiringi suara yang dapat berupa dialog atau musik sehingga dialog atau musik merupakan alat bantu penguat ekspresi, di samping suara musik, warna yang mempertinggi tingkat nilai kenyataan pada film sehingga unsur sungguh-sungguh terjadi sedang dialami oleh khalayak pada saat film diputar makin terpenuhi.

2.7. Internet

Pengertian Internet, internet (*interconnected computer networks*) bisa didefinisikan *network* komputer tiada batas yang menjadi penghubung pengguna komputer dengan pengguna komputer lainnya serta dapat berhubungan dengan komputer di sebuah wilayah ke wilayah di penjuru dunia, dimana di dalam jaringan tersebut mempunyai berbagai macam informasi serta fasilitas layanan internet *browsing* atau *surfing*. Istilah ini lebih dikenal dengan "*online*" di internet.Pekerjaan ini bisa diibaratkan seperti kita berjalan-jalan di tempat hiburan sembari melihat-lihat ke toko-toko namun tidak membeli jualan tersebut.

2.8. *Tools* Implementasi

Tools implementasi merupakan peralatan-peralatan yang dibutuhkan dalam membangun perangkat lunak. Adapun *tools* implementasi yang digunakan untuk membuat aplikasi informasi bioskop berbasis Android adalah JDK sebagai perangkat pembangun Java, Eclipse IDE sebagai lingkungan pengembangan

perangkat lunak, MySQL sebagai sistem manajemen basis data, dan XAMPP sebagai server pada localhost.

2.8.1. JDK

JDK (*Java Development Kit*) adalah sebuah perangkat peralatan yang digunakan untuk membangun perangkat lunak dengan menggunakan bahasa pemrograman Java.JDK berjalan diatas sebuah *virtual machine* yang dinamakan JVM (Java *Virtual Machine*).

Dokumentasi JDK berisi spesifikasi API, deskripsi fitur, panduan pengembang, referensi halaman untuk perkakas JDK dan utilitas, demo, dan *link* ke informasi terkait.Dokumentasi ini juga tersedia dalam bundel yang dapat diunduh dan diinstal pada komputer.

2.8.2. Ecipse IDE

Eclipse merupakan komunitas *open source* yang bertujuan menghasilkan *platform* pemrograman terbuka. Eclipse terdiri dari *framework* yang dapat dikembangkan lebih lanjut, peralatan bantu untuk membuat dan me-*managesoftware* sejak awal hingga diluncurkan. *Platform* Eclipse didukung oleh ekosistem besar yang terdiri dari vendor teknologi, *start-up* inovatif, universitas, riset institusi serta individu.

Banyak orang mengenal Eclipse sebagai IDE (integrated development environment) untuk bahasa Java, tapi Eclipse lebih dari sekedar IDE untuk Java.

Komunitas Eclipse memiliki lebih dari 60 proyek *open source*. Proyek-proyek ini secara konsep terbagi menjadi 7 categori :

1. Enterprise Development
2. Embedded and Device Development
3. Rich Client Platform
4. Rich Internet Applications
5. Application Frameworks
6. Service Oriented Architecture (SOA)

Secara umum Eclipse digunakan untuk membangun software inovatif berstandar industri, dan alat bantu beserta *framework*-nya membantu pekerjaan menjadi lebih mudah.

2.8.3. MySQL

MySQL adalah sebuah perangkat lunak iasm manajemen basis data SQL (bahasa Inggris: *database management system*) atau DBMS yang *multithread*, *multi-user*, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL.

Tidak sama dengan proyek-proyek seperti Apache, dimana perangkat lunak dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh penulisnya masing-masing, MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia MySQL AB, dimana memegang hak cipta iasm atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan MySQL AB adalah: David Axmark, Allan Larsson, dan Michael "Monty" Widenius.

MySQL adalah *Relational Database Management System* (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public License*).Dimana setiap orang bebas untuk menggunakan MySQL, namun tidak boleh dijadikan produk turunan yang bersifat komersial.MySQL sebenarnya merupakan turunan salah satu konsep utama dalam *database* sejak lama, yaitu SQL (*Structured Query Language*).SQL adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis. Keandalan suatu iasm *database* (DBMS) dapat diketahui dari cara kerja *optimizer*-nya dalam melakukan proses perintah-perintah SQL, yang dibuat oleh user maupun program-program aplikasinya. Sebagai *database server*, MySQL dapat dikatakan lebih unggul dibandingkan *database server* lainnya dalam *query* data. Hal ini terbukti untuk *query* yang dilakukan oleh *single user*, kecepatan

query MySQL iasm sepuluh kali lebih cepat dari PostgreSQL dan lima kali lebih cepat dibandingkan Interbase.

2.8.4. XAMPP

XAMPP merupakan singakatan dari X (empat sistem operasi apapun), Apache, MySQL, PHP, Perl.XAMPP merupakan tool yang menyediakan paket perangkat lunak ke dalam satu buah paket. Dalam paketnya sudah terdapat Apache (*web server*), MySQL (*database*), PHP (*server side scripting*), Perl, FTP *Server*, phpMyAdmin dan berbagai pustaka bantu lainnya. Dengan menginstall XAMPP maka tidak perlu lagi melakukan instalasi dan konfigurasi *web server* Apache, PHP dan MySQL secara manual. XAMPP akan menginstalasi dan mengkofigurasikannya secara otomatis.

2.9. Unified Modelling Language (UML)

UML (Unified Modeling Language) adalah sebuah bahasa untuk menetukan, visualisasi, kontruksi, dan mendokumentasikan artifact (bagian dari informasi yang digunakan atau dihasilkan dalam suatu proses pembuatan perangkat lunak.Artifact dapat berupa model, deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan system non perangkat lunak lainnya.

UML merupakan suatu kumpulan teknik terbaik yang telah terbukti sukses dalam memodelkan sistem yang besar dan kompleks. UML tidak hanya digunakan dalam proses pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan.

Bagian-bagian utama dari UML adalah *view*, *diagram*, *model element*, dan *general mechanism* :

A. View

View digunakan untuk melihat sistem yang dimodelkan dari beberapa aspek yang berbeda. *View* bukan melihat grafik, tapi merupakan suatu abstraksi yang berisi sejumlah *diagram*.

Beberapa jenis *view* dalam UML antara lain: *use case view*, *logical view*, *component view*, *concurrency view*,dan *deployment view*.

B. Use Case View

Mendeskripsikan fungsionalitas sistem yang seharusnya dilakukan sesuai yang diinginkan *external actors*. *Actor* yang berinteraksi dengan sistem dapat berupa *user* atau sistem lainnya.

View ini digambarkan dalam *use casediagrams* dan kadang-kadang dengan *activity diagrams*. *View* ini digunakan terutama untuk pelanggan, perancang (*designer*), pengembang (*developer*), dan penguji sistem (*tester*).

C. Logical View

Mendeskripsikan bagaimana fungsionalitas dari sistem, struktur statis (*class*,*objec*t,dan*relationship*) dan kolaborasi dinamis yang terjadi ketika object mengirim pesan ke object lain dalam suatu fungsi tertentu.

View ini digambarkan dalam *class* diagrams untuk struktur statis dan dalam *state*, *sequence*, *collaboration*, dan *activitydiagram* untuk model dinamisnya. *View* ini digunakan untuk perancang (*designer*) dan pengembang (*developer*).

D. Component View

Mendeskripsikan implementasi dan ketergantungan modul.Komponen yang merupakan tipe lainnya dari *code module* diperlihatkan dengan struktur dan ketergantungannya juga alokasi sumber daya komponen dan informasi *administrative* lainnya.

View ini digambarkan dalam *component view* dan digunakan untuk pengembang (*developer*).

E. Concurrency View

Membagi sistem ke dalam proses dan prosesor. *View* ini digambarkan dalam diagram dinamis (*state*, *sequence*, *collaboration*, dan *activity*

diagrams) dan diagram implementasi (*component* dan *deployment diagrams*) serta digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

F. Deployment View

Mendeskripsikan fisik dari sistem seperti komputer dan perangkat (*nodes*) dan bagaimana hubungannya dengan lainnya.

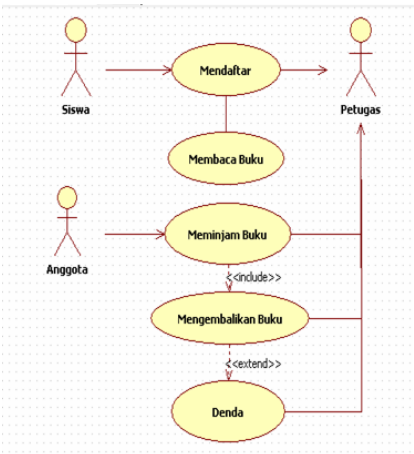
View ini digambarkan dalam deployment diagramsdan digunakan untuk pengembang (*developer*), pengintegrasi (*integrator*), dan penguji (*tester*).

G. Diagram

Diagram berbentuk grafik yang menunjukkan simbol elemen model yang disusun untuk mengilustrasikan bagian atau aspek tertentu dari sistem. Sebuah diagram merupakan bagian dari suatu view tertentu dan ketika digambarkan biasanya dialokasikan untuk *view* tertentu. Adapun jenis diagram antara lain :

1. ***Use Case Diagram***

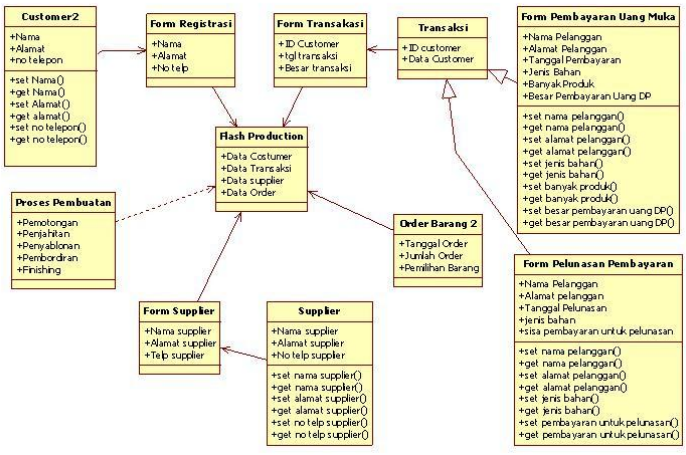
   Use case adalah abstraksi dari interaksi antara system dan actor. *Use case* bekerja dengan cara mendeskripsikan tipe interaksi antara *user* sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. *Use case* merupakan konstruksi untuk mendeskripsikan bagaimana sistem akan terlihat di mata *user*. Sedangkan *use case diagram* memfasilitasi komunikasi diantara analis dan pengguna serta antara analis dan *client*. Contoh *use case diagram* dapat dilihat pada gambar 2.7.



Gambar 2.7 *Use Case Diagram*

2. *Class Diagram*

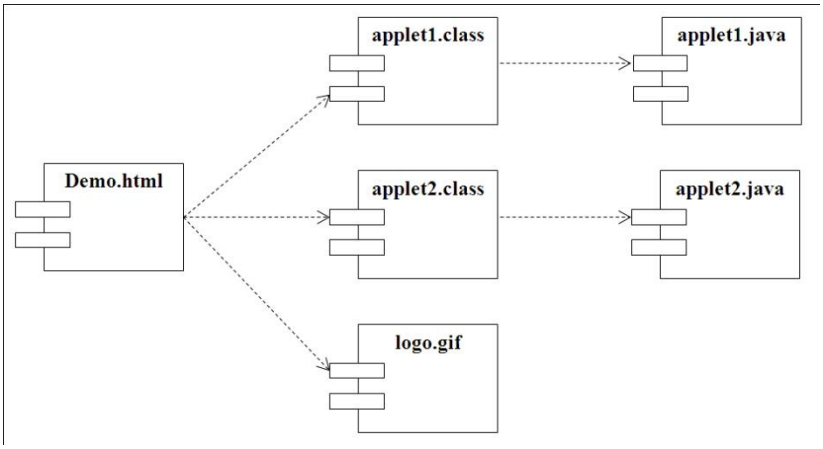
Class adalah dekripsi kelompok objek-objek dengan *property*, perilaku (operasi) dan relasi yang sama. Sehingga dengan adanya *class diagram* dapat memberikan pandangan global atas sebuah sistem. Hal tersebut tercermin dari *class-class* yang ada dan relasinya satu dengan yang lainnya.Sebuah sistem biasanya mempunyai beberapa *class diagram*. *Class diagram* sangat membantu dalam visualisasi struktur kelas dari suatu sistem. Contoh dari *class diagram* dapat dilihat pada gambar 2.8.



Gambar 2.8 *Class Diagram*

3. *Component Diagram*

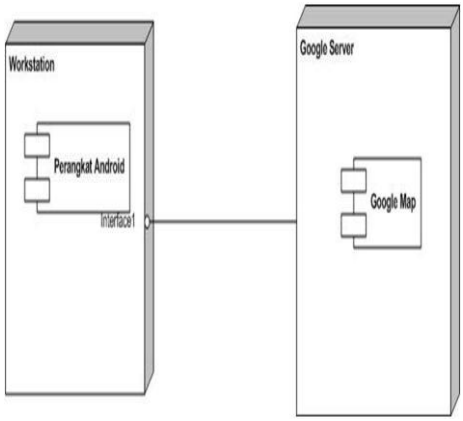
Component software merupakan bagian fisik dari sebuah sistem, karena menetap di komputer tidak berada di benak para analis.Komponent merupakan implementasi *software* dari sebuah atau lebih class.Komponent dapat berupa *source code*, komponent biner, atau *executable component*.Sebuah komponent berisi informasi tentang *logic class* atau *class* yang diimplementasikan sehingga membuat pemetaan dari *logical view* ke *component view*.Sehingga *component diagram* merepresentasikan dunia riil yaitu *component software* yang mengandung *component*, *interface* dan *relationship*.Contoh *component diagram* dapat dilihat pada gambar 2.9.



Gambar 2.9 *Component Diagram*

4. *Deployment Diagram*

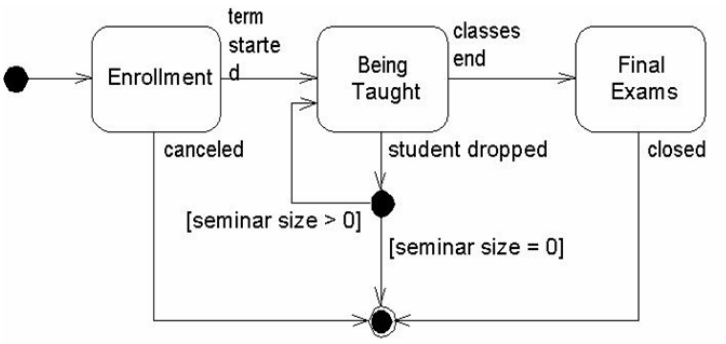
Menggambarkan tata letak sebuah sistem secara fisik, menampakkan bagian-bagian *software* yang berjalan pada bagian-bagian *hardware*, menunjukkan hubungan komputer dengan perangkat (*nodes*) satu sama lain dan jenis hubungannya. Di dalam *nodes*, *executeable component* dan *object* yang dialokasikan untuk memperlihatkan unit perangkat lunak yang dieksekusi oleh *node* tertentu dan ketergantungan komponen. Contoh *deployment diagram* dapat dilihat pada gambar 2.10.



Gambar 2.10 *Deployment Diagram*

5. State Diagram

Menggambarkan semua *state* (kondisi) yang dimiliki oleh suatu object dari suatu *class* dan keadaan yang menyebabkan *state* berubah. Kejadian dapat berupa *object* lain yang mengirim pesan. *Stateclass* tidak digambarkan untuk semua *class*, hanya yang mempunyai sejumlah *state* yang terdefinisi dengan baik dan kondisi *class* berubah oleh *state* yang berbeda. Contoh dari *state diagram* dapat dilihat pada gambar 2.11.

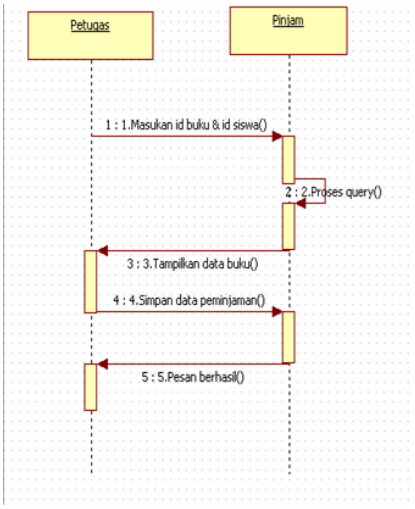


Gambar 2.11 *State Diagram*

6. Sequence Diagram

Sequence Diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Kegunaannya untuk menunjukkan rangkaian pesan yang dikirim antara *object* juga interaksi antara *object*, sesuatu yang

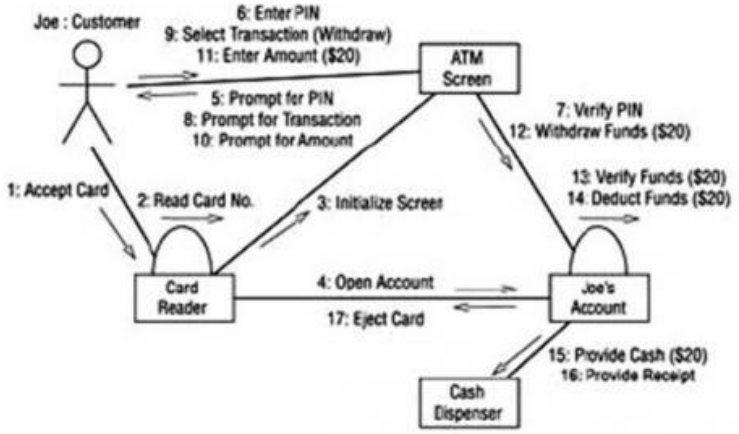
terjadi pada titik tertentu dalam eksekusi sistem. Contoh dari *sequence diagram* dapat dilihat pada gambar 2.12.



Gambar 2.12 *Sequence Diagram*

7. *Collaboration Diagram*

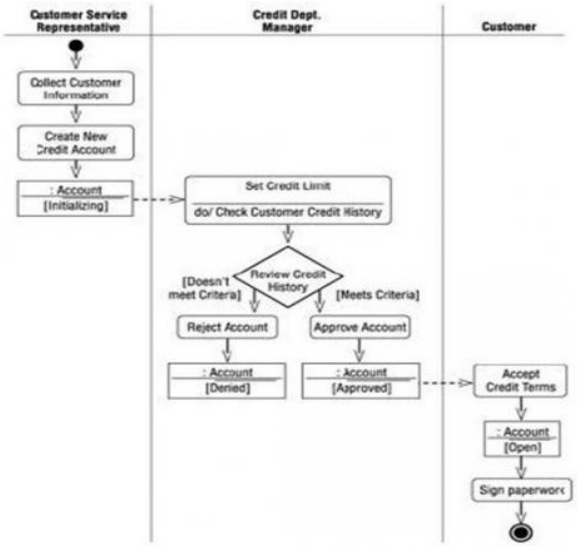
Menggambarkan kolaborasi dinamis seperti *sequence diagrams*. Dalam menunjukkan pertukaran pesan, *collaboration diagrams* menggambarkan *object* dan hubungannya (mengacu ke konteks). Jika penekannya pada waktu atau urutan gunakan *sequence diagrams*, tapi jika penekanannya pada konteks gunakan *collaboration diagram*. Contoh *collaboration diagram* dapat dilihat pada gambar 2.13.



Gambar 2.13 *Collaboration Diagram*

8. Activity Diagram

Menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktifitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *use case* atau interaksi. Contoh dari *activity diagram* dapat dilihat pada gambar 2.14.



Gambar 2.14 *Activity Diagram*

2.10. Google Maps

Google Mapsadalah sebuah jasapetaglobe virtualgratisdan*online*disediakan oleh Googledapat ditemukan di http://maps.google.com.Ia menawarkan peta yang dapat diseret dan gambar satelit untuk seluruh dunia dan baru-baru ini, Bulan, dan juga menawarkan perencana rute dan pencari letak bisnis di U.S., Kanada, Jepang, Hong Kong, Cina, UK, Irlandia (hanya pusat kota) dan beberapa bagian Eropa. Google Maps masih berada dalam tahap beta ada juga aplikasi dari google yaitu google earth.

Google Earth adalah sebuah program interaktif yang memungkinkan pengguna untuk melihat semua tempat di bumi atau di angkasa. Tampilan bukan sebuah gambar rekayasa tapi gambar nyata yang telah diambil oleh fotografi, jadi apa yang anda lihat adalah foto yang sebenarnya.Cara menggunakan Google Earth tidak begitu sulit, tapi dibutuhkan kebiasaan untuk

menggunakannya.Google Earth adalah cara yang luar biasa untuk melihat dunia dan merupakan alat pendidikan yang sangat bermanfaat.

2.11. Location Based Service

Definisi *Location Based Service* atau disingkat LBS adalah kemampuan untuk mencari lokasi geografis dari *mobile device* dan menyediakan layanan berdasarkan lokasi yang diperolehnya.Konsep LBS ini menghasilkan layanan informasi yang dapat dicocokkan dengan lokasi keberadaan calon penerima informasi.Hal ini menyebabkan peningkatan nilai informasi dikarenakan penerima dapat mengasosiasikan pengetahuan atau informasi yang didapat keberadaannya.

Terdapat empat komponen pendukung utama dalam teknologi Layanan Berbasis Lokasi, antara lain:

1. Piranti *Mobile*.

   Piranti *Mobile* adalah salah satu komponen penting dalam LBS. Piranti ini berfungsi sebagai alat bantu (*tool*) bagi pengguna untuk meminta informasi. Hasil dari informasi yang diminta dapat berupa teks, suara, gambar dan lain sebagainya.Piranti *mobile* yang dapat digunakan bisa berupa PDA, *smartphone*, *laptop*.Selain itu, piranti *mobile* dapat juga berfungsi sebagai alat navigasi di kendaraan seperti halnya alat navigasi berbasis GPS.

2. Jaringan Komunikasi.

   Komponen kedua adalah jaringan komunikasi.Komponen ini berfungsi sebagai jalur penghubung yang dapat mengirimkan data-data yang dikirim oleh pengguna dari piranti *mobile*-nya untuk kemudian dikirimkan ke penyedia layanan dan kemudian hasil permintaan tersebut dikirimkan kembali oleh penyedia layanan kepada pengguna.

3. Komponen *Positioning* (Penunjuk Posisi/Lokasi).

   Setiap layanan yang diberikan oleh penyedia layanan biasanya akan berdasarkan pada posisi pengguna yang meminta layanan tersebut. Oleh karena itu diperlukan komponen yang berfungsi sebagai pengolah/pemroses yang akan menentukan posisi pengguna layanan

saat itu. Posisi pengguna tersebut bisa didapatkan melalui jaringan komunikasi *mobile* atau juga menggunakan *Global Positioning System* (GPS).

4. Penyedia layanan dan aplikasi.

   Penyedia layanan merupakan komponen LBS yang memberikan berbagai macam layanan yang bisa digunakan oleh pengguna. Sebagai contoh ketika pengguna meminta layanan agar bisa tahu posisinya saat itu, maka aplikasi dan penyedia layanan langsung memproses permintaan tersebut, mulai dari menghitung dan menentukan posisi pengguna, menemukan rute jalan, mencari data di Yellow Pages sesuai dengan permintaan, dan masih banyak lagi yang lainnya.

5. Penyedia data dan konten.

   Penyedia layanan tidak selalu menyimpan seluruh data dan informasi yang diolahnya.Karena bisa jadi berbagai macam data dan informasi yang diolah tersebut berasal dari pengembang/pihak ketiga yang memang memiliki otoritas untuk menyimpannya.Sebagai contoh basis data geografis dan lokasi bisa saja berasal dari badan-badan milik pemerintah atau juga data-data perusahaan/bisnis/industri bisa saja berasal dari Yellow Pages, maupun perusahaan penyedia data lainnya.

2.12. Web Service

Web service adalah sebuah *software* aplikasi yang tidak terpengaruh oleh *platform*, ia akan menyediakan *method-method* yang dapat diakses oleh *network*. Ia juga akan menggunakan XML untuk pertukaran data, khususnya pada dua entitas bisnis yang berbeda. Sistem lain berinteraksi dengan *Web service* dalam cara ditentukan oleh deskripsi dengan menggunakan pesan SOAP, biasanya disampaikan menggunakan HTTP dengan serialisasi XML dalam hubungannya dengan Web lainnya yang terkait standar.

Dalam pengertian yang sederhana , XML *Web Services* dapat di definisikan sebagai aplikasi yang diakses oleh aplikasi yang lain. Mungkin orang berpendapat

itu semacam *web site*, tetapi itu bukan demikian.Ada perbedaan – perbedaan yang membedakan dengan *web site* [9]. Perbedaan tersebut dapat dilihat dibawah ini :

Web Site :

1. Memiliki *web interface*
2. Dibuat untuk berinteraksi langsung dengan *user*
3. Dibuat untuk bekerja pada *web browser*

Web Service :

1. Tidak memiliki *interface* yang bagus
2. Dibuat untuk berinteraksi langsung dengan aplikasi yang lain baik yang berbeda sistem operasi / konsep sekalipun
3. Dibuat untuk bekerja pada semua tipe *client* aplikasi / perangkat *device*

Adapun beberapa karakteristik dari *web service*adalah :

- *Message-based*
- *Standards-based*
- *Programming language independent*
- *Platform-neutral*

Beberapa *key standard* didalam *web service* adalah: XML, SOAP, WSDL and UDDI.

SOAP (*Simple Object Access Protocol*) adalah sebuah XML-*based mark-up language* untuk pergantian pesan diantara aplikasi-aplikasi. SOAP berguna seperti sebuah amplop yang digunakan untuk pertukaran data *object* didalam *network*. SOAP mendefinisikan empat aspek didalam komunikasi: *Message envelope, Encoding, RPC call convention*, dan bagaimana menyatukan sebuah *message* didalam protokol *transport*.

Sebuah SOAP *message* terdiri dari SOAP *Envelop* dan bisa terdiri dari *attachments* atau tidak memiliki *attachment*. SOAP *envelop* tersusun dari SOAP *header* dan SOAP *body*, sedangkan SOAP *attachment* membolehkan *non*-XML

data untuk dimasukkan kedalam SOAP *message*, di-*encoded*, dan diletakkan kedalam SOAP *message* dengan menggunakan MIME-*multipart*.

WSDL (*Web Services Description Language*) adalah sebuah XML-*based language* untuk mendeskripsikan XML.WSDL menyediakan *service* atau layananyang mendeskripsikan *service request* dengan menggunakan protokol-protokol yang berbeda dan juga *encoding*.WSDL memfasilitasi komunikasi antar aplikasi. WSDL akan mendeskripsikan apa yang akan dilakukan oleh *web service*, bagaimana menemukannya dan bagaimana untuk mengoperasikannya.

Spesifikasi WSDL mendefinisikan tujuh tipe *element*:

1. *Types* – elemen untuk mendefinisikan tipe data. Mereka akan mendefinisikan tipe data (seperti *string* atau *integer*) dari elemen didalam sebuah *message*.
2. *Message* – *abstract*, pendefinisian tipe data yang akan dikomunikasikan.
3. *Operation* – sebuah deskripsi *abstract* dari sebuah tindakan yang didukung oleh *service*.
4. *Port Type* – sebuah koleksi *abstract* dari operasi - operasi yang didukung oleh lebih dari satu *endpoint*.
5. *Binding* – mendefinisikan penyatuan dari tipe *port* (koleksi dari operasi - operasi) menjadi sebuah protokol transport dan data format (ex. SOAP 1.1 pada HTTP). Ini adalah sebuah protokol konkret dan sebuah spesifikasi data format didalam tipe *port* tertentu.
6. *Port* – mendefinisikan sebuah komunikasi *endpoint* sebagai kombinasi dari *binding* dan alamat *network*. Bagi protokol HTTP, sebuah bentuk dari URL sedangkan bagi protokol SMTP, ini adalah sebuah *form* dari *email address*.
7. *Service* – satu set *port* yang terkorelasi atau suatu *endpoints*.

WSDL mendefinisikan *service* sebagai sebuah koleksi dari *endpointsnetwork*. Sebuah definisi abstrak dari *endpoints* dan *messages* adalah ia bersifat terpisah dari pembangunan *network* atau penyatuan data format.

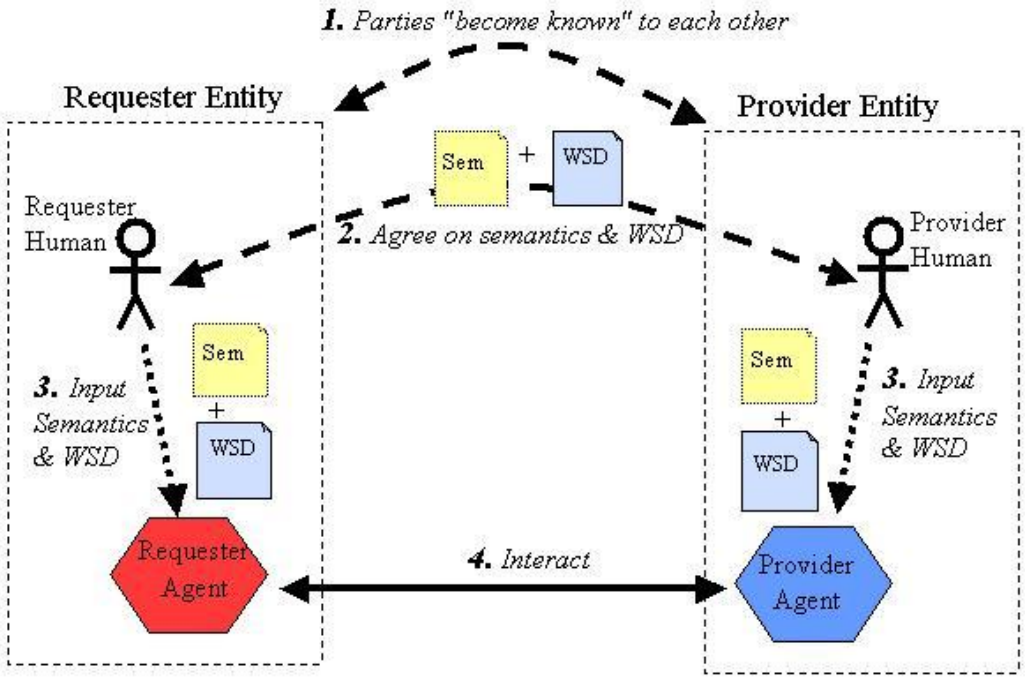
Pembagian ini menyebabkan penggunaan kembali *abstract description* dari data yang akan dipertukarkan (*message exchange*) dan *abstract collection* dari operasi (*ports*) Protokol konkret dan spesfikasi data format bagi tipe *port* tertentu menentukan *binding* yang dapat digunakan kembali(*reusable*). Sebuah *port* adalah sebuah *network address* yang dikombinasikan *reusable binding*; sebuah *service* adalah koleksi dari *port-port*.

Sedangkan **UDDI** (U*niversal Description, Discovery and Integration*) adalah sebuah *service registry* bagi pengalokasian *web service*.UDDI mengkombinasikan SOAP dan WSDL untuk pembentukan sebuah *registry* API bagi pendaftaran dan pengenalan *service*.Ia menyediakan sebuah area umum dimana sebuah organisasi dapat mengiklankan keberadaan mereka dan *service* yang diberikan (*web service*).

Semantik pada *Web service* adalah harapan bersama tentang perilaku layanan, khususnya dalam menanggapi pesan yang dikirim ke tujuan.Akibatnya, ini adalah "kontrak" antara entitas pemohon dan badan penyedia tentang tujuan dan konsekuensi dari interaksi. Meskipun kontrak ini merupakan keseluruhan perjanjian antara entitas peminta dan entitas penyedia tentang bagaimana dan mengapa masing-masing agen akan berinteraksi, itu belum tentu tertulis atau eksplisit dinegosiasikan. Ini mungkin eksplisit atau implisit, lisan atau tertulis, mesin *processable* atau manusia berorientasi, dan mungkin suatu perjanjian hukum atau kesepakatan informal (non-hukum).

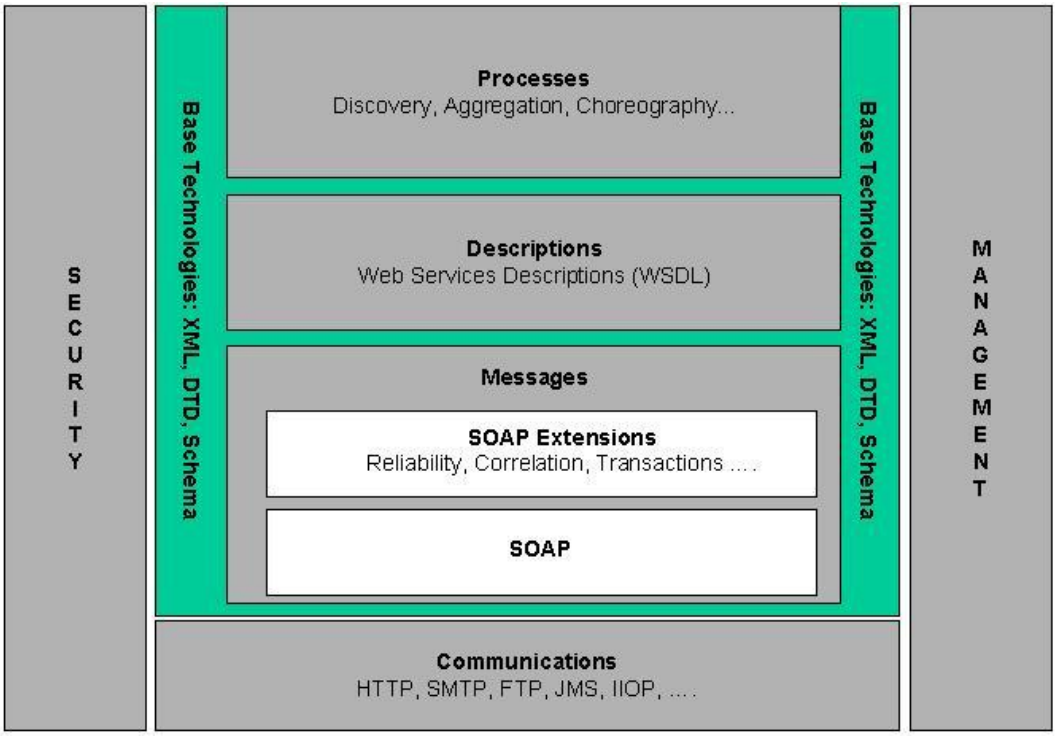
Ada banyak cara bahwa entitas peminta mungkin terlibat dan menggunakan *Web service*. Secara umum, langkah-langkah yang luas berikut yang diperlukan, seperti yang diilustrasikan pada Gambar 2.15. (1) pemohon dan penyedia entitas menjadi dikenal satu sama lain (atau setidaknya satu menjadi tahu untuk yang lain); (2) peminta dan penyedia entitas entah bagaimana setuju pada deskripsi layanan dan semantik yang akan mengatur interaksi antara pemohon dan agen penyedia; (3) deskripsi layanan dan semantik direalisasikan oleh pemohon dan agen penyedia, dan (4) pemohon dan agen penyedia bertukar pesan, sehingga melakukan beberapa tugas atas nama pemohon dan badan penyedia. (Ie, pertukaran pesan dengan agen penyedia merupakan wujud nyata dari berinteraksi

dengan layanan Web penyedia entitas.) Proses umum web service dapat dilihat pada gambar 2.15.



Gambar 2.15 Proses Umum *Web Service*

Arsitektur *Web service* melibatkan teknologi berlapis banyak dan saling terkait. Ada banyak cara untuk memvisualisasikan teknologi ini, seperti halnya ada banyak cara untuk membangun dan menggunakan *Web service*. Gambar 2.16 di bawah ini memberikan sebuah ilustrasi dari beberapa keluarga teknologi.



Gambar 2.16 Arsitektur *Web Service*

Web Services itu digunakan saat kita akan mentransformasi sebuat bisnis logik / sebuah *class* dan *object* yang terpisah dalam 1 ruang lingkup yang menjadi satu, sehingga tingkat keamanan dapat di tangani dengan baik. Selain itu *Web Service* juga lebih mudah dalam *processdeployment*-nya, karena tidak memerlukan registrasi khusus ke dalam sistem operasi.*Web Service* cukup diupload ke *Web Server* dan siap diakses oleh pihak-pihak yang telah diberikan otorisasi.*Web Service* berjalan di *port* 80 yang merupakan protokol standar HTTP, dengan demikian mengurangi resiko terblokir oleh *firewall*. Kendala arsitektur COM/DCOM adalah memerlukan konfigurasi khusus di sisi *firewall*, dan ini tidak perlu dilakukan untuk mengakses *Web Service*.

Beberapa vendor luar negeri mulai berkolaborasi satu sama lain dengan konsep *web services* , diantaranya : IBM, Microsoft , SUN , ORACLE Diantaranya contoh *web services* yang sudah jadi dan dipakai adalah *web services* keluaran Microsoft (Microsoft Passport) – *web services* untuk *user name* dan *password* yang sudah dipasang di *web site* Microsoft dan HOTMAIL.