BAB 2
LANDASAN TEORI

Pada bab ini menjelaskan mengenai konsep dasar dan teori-teori yang mendukung pembangunan aplikasi. Selain itu, pada bab ini juga dijelaskan *tools-tools* yang digunakan dalam pembangunan aplikasi.

2.1. Pemandu Wisata

Menurut Kamus Besar Bahasa Indonesia (KBBI) pemandu wisata atau pramuwisata adalah petugas pariwisata yang berkewajiban memberi petunjuk dan informasi yg diperlukan wisatawan [3]. Pemandu wisata selain sebagai penunjuk jalan juga bertugas memberikan informasi mengenai lokasi yang dikunjungi. Pemandu wisata memahami informasi mengenai lokasi yang dikunjungi wisatawan.

2.2. Aplikasi

Aplikasi adalah sebuah perangkat lunak yang menjadi *frontend* dalam sebuah sistem yang digunakan untuk mengolah data menjadi suatu informasi yang berguna orang-orang dan sistem yang bersangkutan [Sri Widianti : 2000].

Istilah aplikasi berasal dari bahasa inggris *"application"* yang berarti penerapan, lamaran ataupun penggunaan. Sedangkan secara istilah, pengertian aplikasi adalah suatu program yang siap untuk digunakan yang dibuat untuk melaksanakan suatu fungsi bagi pengguna jasa aplikasi serta penggunaan aplikasi lain yang dapat digunakan oleh suatu sasaran yang akan dituju. Menurut kamus komputer eksekutif, aplikasi mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu tehnik pemrosesan data aplikasi yang biasanya berpacu pada sebuah komputansi yang diinginkan atau diharapkan maupun pemrosesan data yang diharapkan.

2.3. Data dan Informasi

Dalam memahami suatu sistem secara utuh, diperlukannya suatu pemahaman yang tepat mengenai konsep data dan informasi. Data dan informasi merupakan dua hal yang sangat erat keterkaitannya. Data merupakan dasar dari sebuah informasi sdangkan informasi merupakan elemen yang dihasilkan dari suatu pengolahan data.

2.3.1. Data

Menurut Al-Bahra Bin Ladjamudin, data dapat didefinisikan sebagai deskripsi dari suatu dan kejadian yang kita hadapi. Data dapat berupa catatan-catatan dalam kertas, buku, atau tersimpan sebagai file dalam database. Data akan menjadi bahan dalam suatu proses pengolahan data. Oleh karena itu, suatu data belum dapat berbicara banyak sebelum diolah lebih lanjut [4].

Adapun definisi dari kata data adalah suatu istilah majemuk dari *datum* yang berarti fakta atau bagian dari kata yang mengandung arti, yang berhubungan dengan kenyataan, simbol-simbol, gambar-gambar, kata-kata angka-angka, huruf-huruf atau simbol-simbol yang menunjukkan ide, objek, kondisi atau situasi. Jelasnya data itu dapat berupa apa saja dan dapat ditemui dimana saja. Kegunaan data adalah sebagai bahan dasar yang objektif dalam proses penyusunan kebijakan dan keputusan. Dalam kaitannya dengan pengolahan data dengan komputer, pengertian data dapat dibatasi pada fakta-fakta yang dapat direkam. Dalam setiap pengolahan data, data merupakan sumber informasi yang dapat dihasilkan.

2.3.2. Informasi

Menurut George R. Terry, Ph.D, informasi adalah data yang penting yang memberikan pengetahuan yang berguna. Kata informasi berasal dari kata Perancis kuno *informacion* (tahun 1387) yang diambil dari bahasa latin *informationem* yang berarti “garis besar, konsep, ide”. Informasi merupakan kata benda dari *informare* yang berarti aktivitas dalam “pengetahuan yang dikomunikasikan” [Online Etymology Dictionary: Information].

2.4. Basis Data

Menurut Ramakrishnan dan Gehrke (2003) basis data sebagai kumpulan data, umumnya mendeskripsikan aktivitas satu organisasi atau lebih yang berhubungan.

Basis data adalah suatu aplikasi terpisah yang menyimpan suatu koleksi data. Suatu basis data bisa terkomputerisasi atau tidak terkomputerisasi. Suatu basis data yang terkomputerisasi menawarkan keuntungan yang lebih banyak jika dibandingkan dengan basis data yang tidak terkomputerisasi. Keuntungan penggunaan basis data terkomputerisasi mencakup [9] :

1. Terpusat dan berbagi data

   Basis data terkomputerisasi dapat mengurangi penggunaan kertas, file, folder dan juga kehilangan atau kesalahan saat menenmpatkan data. Data dsimpan dalam komputer dan banyak pengguna yang dapat mengaksesnya.

2. *Current data*

   Pengguna bisa secara cepat memperbaharui data yang siap digunakan.

3. Akurasi dan konsistenssi

   Basis data terkomputerisasi bisa dirancang untuk memvalidasi masukan data serta memastikan data tersebut valid dan konsisten.

4. Analisis

   Basis data terkomputerisasi bisa menyimpan mnjejaki dan memproses isi data yang besar dari sumber yang berbeda.

5. Keamanan

   Basis data terkomputerisasi dapat diproteksi dengan penetapan sandi *(password)* dan identifikasi pegguna yang diotorisasi.

6. Pemulihan

   Basis data terkomputerisasi dapat lebih mudah dilakukan pemulihan sistem jika terjadi kegagalan sistem *(crash).*

7. Transaksi

   Transaksi memastikan bahwa perubahan basis data yang terkait selalu dijadikan unit di mana semua prubahan dilaksanakan atau tidak sama sekali.

2.5. *Web Services*

Web service adalah sebuah *service* yang tersedia dalam *internet* yang menggunakan sistem pesan *XML* terstandarisasi [12]. *Web service* tidak terikat pada sistem operasi maupun bahasa pemrograman. Ada beberapa alternatif dalam pertukaran pesan *XML*. Contohnya, *XML Remote Procedure Calls (XML-RPC)* atau *SOAP* dapat digunakan dalam pertukaran pesan. Alternatif lain adalah dengan hanya menggunakan *HTTP GET/POST* untuk mengirimkan pesan *XML*.

Meski tidak harus ada, sebuah *web service* juga diharapkan memiliki dua buah properti tambahan [12] :

1. Sebuah *web service* harus dapat mendefinisikan dirinya sendiri.

   Antarmuka publik *web service* harus dipublikasikan bersamaan dengan publikasi *web service*. *Service* paling tidak harus menyediakan dokumen yang dapat dibaca oleh pengembang lain sehingga mudah untuk digunakan. *SOAP service* yang dibuat sebaiknya juga menyertakan antarmuka publik yang ditulis dalam bahasa *XML* yang umum. *XML* tersebut dapat digunakan mengidentifikasi semua *public method, method argument,* dan *return values*.

2. Sebuah *web service* harus dapat ditemukan.

   Web service yang dibuat harus disertai dengan mekanisme sederhana untuk mempublikasikannya. Mekanisme tersebut memungkinkan pihak-pihak yang berkepentingan untuk menemukan *service* dan mendapatkan lokasi antarmuka publiknya. Mekanisme tersebut dapat berupa sistem terdesentralisasi atau sistem *registry* yang terpusat.

Web service yang lengkap adalah semua *service* yang memiliki [12] :

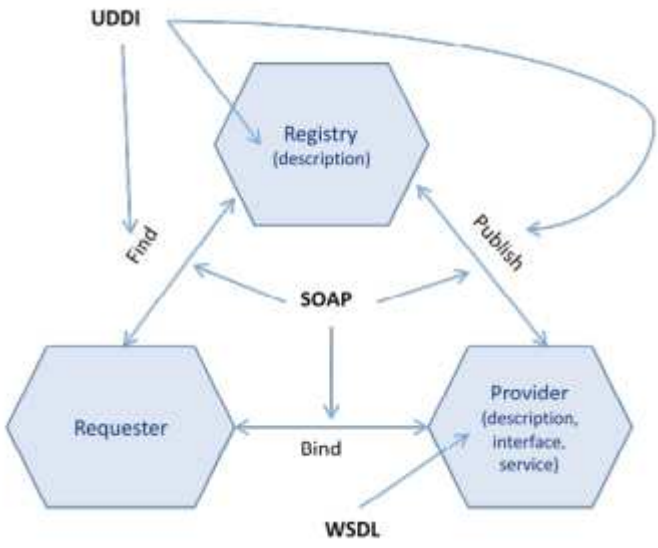
1. Tersedia melalui *internet* atau jaringan *private (intranet).*
2. Menggunakan sistem pesan *XML* yang terstandarisasi.
3. Tidak terikat pada sistem operasi dan bahasa pemrograman
4. Mendefinisikan diri sendiri melalui tata bahasa standar *XML*.
5. Dapat dicari dengan mekanisme pencarian sederhana.

Web service menyimpan data informasi dalam format *XML*, sehingga data ini dapat diakses oleh sistem lain walaupun berbeda *platform*, sistem operasi, maupun bahasa *compiler*. Namun dalam pengembangannya *XML* tidak lagi menjadi satu-satunya format pertukaran data, *JSON (Java Script Object Notation)* merupakan salah satu format pertukaran data yang dapat diimplementasikan pada *web service* selain *XML*.

Web service bertujuan untuk meningkatkan kolaborasi antar pemrogram dan perusahaan, yang memungkinkan sebuah fungsi di dalam *web service* dapat dipinjam oleh aplikasi lain tanpa perlu mengetahui detail pemrograman yang terdapat di dalamnya.

2.5.1. Arsitektur *Web Service*

Web service dibangun dari tiga komponen utama yaitu *service provider*, *service registry*, dan *service requestor* [13]. Komponen-komponen tersebut saling berinteraksi melalui artifak *web service*, yang berupa deskripsi dan implementasi layanan. Terdapat tiga macam operasi yang memungkinkan komponen-komponen tersebut untuk dapat saling berinteraksi, yaitu *publish, find,* dan *bind* [13].



Gambar 2. 1 Arsitektur Web Service [13].

Pada gambar diatas terdapat 3 komponen utama yang terdapat dalam *web service* yaitu [13] :

1. *Service Provider*.

   Merupakan komponen yang menyediakan akses operasi layanan *web service*.

2. *Service Requestor*.

   Platform yang bertindak sebagai *client* dari *web service* dan memberikan akses layanan.

3. *Service Registry*.

   Merupakan tempat dimana *service provider* mempublikasikan layanannya. Pada arsitektur *web service*, *service registry* bersifat optional. Teknologi *web service* memungkinkan kita dapat menghubungkan berbagai jenis *software* yang memiliki *platform* dan sistem operasi yang berbeda.

2.6. *Location Based Services (LBS)*

LBS adalah layanan informasi yang dapat diakses menggunakan piranti *mobile* melalui jaringan *internet* dan *seluler* serta memanfaatkan kemampuan penunjuk lokasi pada piranti *mobile* [Virrantasu, et al, 2001].

2.6.1. Komponen *LBS*

Terdapat empat komponen pendukung utama dalam teknologi *Location Based Services*, antara lain [Steiniger, et al 20] :

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* (*Personal Digital Assistant*), *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)

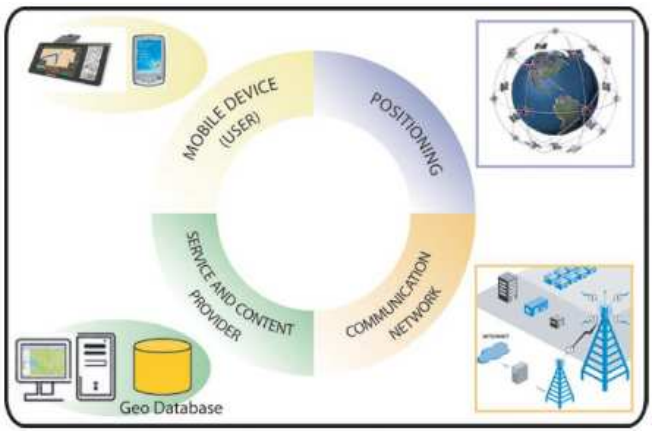
Setiap layanan yang diberikan oleh penyedia layanan biasanya akan berdasarkan pada posisi pengguna yang meminta layanan tersebut. Oleh karena itu diperlukan komponen yang berfungsi sebagai pengolah/pemroses yang akan menentukan posisi pengguna layanan saat itu. Posisi pengguna tersebut bisa didapatkan 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. Secara lengkap keempat komponen pendukung utama *LBS* tersebut dapat dilihat pada gambar berikut.



Gambar 2. 2 Komponen Pendukung Utama Teknologi LBS [Steiniger, et al 20].

2.7. *Global Positioning System (GPS)*

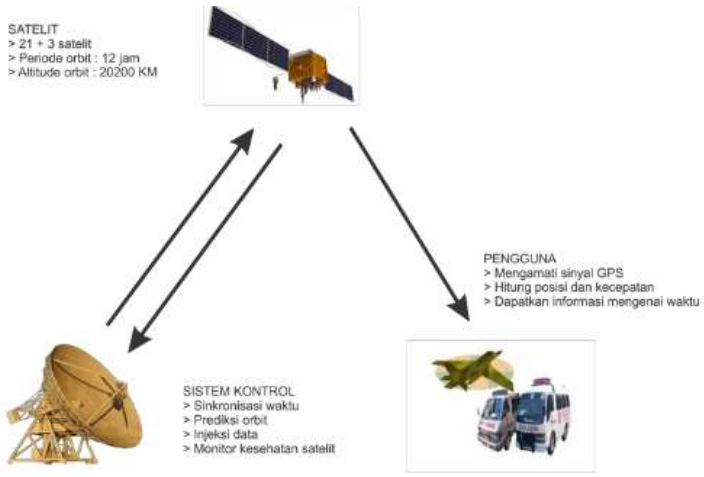
Pembahasan mengenai *GPS* akan dijelaskan pada sub bab berikut.

2.7.1. Pengertian *GPS*

GPS adalah sebuah sistem navigasi berbasis radio yang menyediakan informasi berupa koordinat posisi, kecepatan dan waktu kepada pengguna dengan bantuan sinkronisasi satelit. Sistem ini didesain untuk memberi kan posisi dan kecepatan tiga dimensi serta informasi mengenai waktu, secara kontinyu di seluruh dunia tanpa tergantung waktu dan cuaca, kepada banyak orang secara simultan. Sistem ini menggunakan satelit yang berfungsi sebagai pengirim sinyal yang berisi informasi koordinat lokasi, kecepatan, arah dan waktu pada alat penerima sinyal *GPS* (*receiver*) dipermukaan bumi [5].

2.7.2. Arsitektur *GPS*

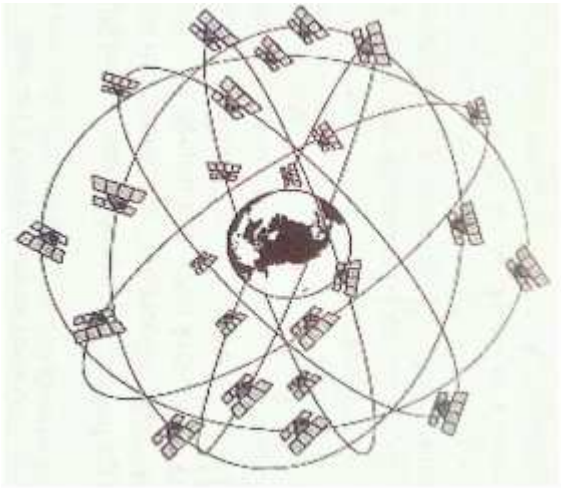
Sistem ini menggunakan sejumlah satelit yang berada di orbit bumi, yang memancarkan sinyal ke bumi dan ditangkap oleh alat penerima di permukaan bumi. Arsitektur *GPS* terdiri dari tiga bagian yaitu *Space Segment*, *Control Segment* dan *User Segment* [5].



Gambar 2. 3 Skema Sistem Penentuan Posisi Global [5].

1. *Space Segment*

Space Segment adalah bagian yang terdiri dari kumpulan-kumpulan satelit diluar angkasa yang berada pada orbit bumi. Jarak antara satelit dengan permukaan bumi biasanya sekitar 20.000 Km diatas permukaan bumi. Satelit-satelit ini diatur sedemikian rupa sehingga *GPS receiver* menerima sedikitnya data dari 6 satelit yang berbeda, namun berada dalam jangkauan orbitnya.



Gambar 2. 4 Space Segment GPS [5].

Sinyal satelit dapat melewati awan, kaca maupun plastik, tetapi tidak dapat melewati gedung dan gunung. Satelit mempunyai jam atom, dan juga akan memancarkan informasi berupa waktu/jam saat ini. Data ini

dipancarkan dengan kode *pseudo-random*. Masing-masing satelit memiliki kodenya sendiri-sendiri. Nomor kode ini biasanya akan ditampilkan pada alat navigasi atau *GPS receiver*, maka dengan nomor kode tersebut pengguna dapat mengidentifikasi sinyal satelit yang sedang diterima alat tersebut. Data ini berguna bagi alat navigasi untuk mengukur jarak antara alat navigasi dengan satelit, yang akan digunakan untuk mengukur koordinat lokasi. Kekuatan sinyal satelit juga akan membantu alat dalam penghitungan.

Kekuatan sinyal ini lebih dipengaruhi oleh lokasi satelit, sebuah alat akan menerima sinyal lebih kuat dari satelit yang berada tepat diatasnya dibandingkan dengan satelit yang berada di garis cakrawala. Ada dua jenis gelombang yang saat ini dipakai untuk alat navigasi berbasis satelit. Pada umumnya, yang pertama lebih dikenal dengan sebutan L1 pada 1575.42 MHz. Sinyal L1 ini yang akan diterima oleh alat navigasi. Satelit juga mengeluarkan gelombang L2 pada frekuensi 1227.6 Mhz. Gelombang L2 ini digunakan untuk tujuan militer dan bukan untuk umum.

2. *Control Segment*

Segment sistem control *GPS* berfungsi mengontrol dan memantau operasional semua satelit *GPS* dan memastikan bahwa semua satelit berfungsi sebagai mana mestinya. Secara lebih spesifik tugas utama dari *segment system control GPS* adalah :

1) Secara kontinyu memantau dan mengontrol sistem satelit.
2) Menentukan dan menjaga waktu system *GPS*.
3) Memprediksi empheris satelit serta karakteristik jam satelit.
4) Secara periodik meremajakan (*update) navigation message* dari setiap satelit.
5) Melakukan manuver satelit agar tetap berada dalam orbitnya, atau melakukan relokasi untuk menggantikan satelit yang tidak ”sehat”, seandainya diperlukan.

3. *User Segment*

Segmen pengguna ini terdiri dari para pengguna satelit *GPS*, baik di darat, laut, udara, maupun di angkasa. Dalam hal ini, alat penerima sinyal *GPS* (*GPS receiver*) diperlukan untuk menerima dan memperoses sinyal dari satelit GPS untuk digunakan dalam penentuan posisi, kecepatan, waktu maupun parameter turunan lainnya.

Satelit akan memancarkan data almanak dan *ephemeris* yang akan diterima oleh alat navigasi secara teratur. Data almanak berisikan perkiraan lokasi (*approximate location*) satelit yang dipancarkan terus menerus oleh satelit. Data *ephemeris* dipancarkan oleh satelit, dan valid untuk sekitar 4-6 jam. Untuk menunjukkan koordinat sebuah titik (dua dimensi), alat navigasi memerlukan paling sedikit sinyal dari 3 buah satelit. Untuk menunjukkan data ketinggian sebuah titik (tiga dimensi), diperlukan tambahan sinyal dari 1 buah satelit lagi. Dari sinyal-sinyal yang dipancarkan oleh kumpulan satelit tersebut, alat navigasi akan melakukan perhitungan-perhitungan, dan hasil akhirnya adalah koordinat posisi yang akan diteralat tersebut. Makin banyak jumlah sinyal satelit yang diterima oleh sebuah alat, akan membuat alat tersebut menghitung koordinat posisinya dengan lebih tepat.

Karena alat navigasi ini bergantung penuh pada satelit, maka sinyal satelit menjadi sangat penting. Alat navigasi berbasis satelit ini tidak dapat bekerja maksimal ketika ada gangguan pada sinyal satelit.

2.8. *Augmented Reality*

Berikut pembahasan mengenai teknologi *Augmented Reality*.

2.8.1. Pengertian *Augmented Reality*

Realitas tertambah, atau kadang dikenal dengan singkatan bahasa Inggrisnya *AR* (*augmented reality*), adalah teknologi yang menggabungkan benda maya dua dimensi dan ataupun tiga dimensi ke dalam sebuah lingkungan nyata tiga dimensi lalu memproyeksikan benda-benda maya tersebut dalam waktu nyata.

Ronald T. Azuma (1997) mendefinisikan *augmented reality* sebagai penggabungan benda-benda nyata dan maya di lingkungan nyata, berjalan secara interaktif dalam waktu nyata, dan terdapat integrasi antarbenda dalam tiga dimensi, yaitu benda maya terintegrasi dalam dunia nyata. Penggabungan benda nyata dan maya dimungkinkan dengan teknologi tampilan yang sesuai, interaktivitas dimungkinkan melalui perangkat-perangkat input tertentu, dan integrasi yang baik memerlukan penjejakan yang efektif.

Benda-benda maya menampilkan informasi yang tidak dapat diterima oleh pengguna dengan inderanya sendiri. Hal ini membuat realitas tertambah sesuai sebagai alat untuk membantu persepsi dan interaksi penggunanya dengan dunia nyata. Informasi yang ditampilkan oleh benda maya membantu pengguna melaksanakan kegiatan-kegiatan dalam dunia nyata.

Realitas tertambah dapat diaplikasikan untuk semua indera, termasuk pendengaran, sentuhan, dan penciuman. Selain digunakan dalam bidang-bidang seperti kesehatan, militer, industri manufaktur, realitas tertambah juga telah diaplikasikan dalam perangkat-perangkat yang digunakan orang banyak, seperti pada telepon genggam.

Selain menambahkan benda maya dalam lingkungan nyata, realitas tertambah juga berpotensi menghilangkan benda-benda yang sudah ada. Menambah sebuah lapisan gambar maya dimungkinkan untuk menghilangkan atau menyembunyikan lingkungan nyata dari pandangan pengguna. Misalnya, untuk menyembunyikan sebuah meja dalam lingkungan nyata, perlu digambarkan lapisan representasi tembok dan lantai kosong yang diletakkan di atas gambar meja nyata, sehingga menutupi meja nyata dari pandangan pengguna.

2.8.2. Konsep *Augmented Reality*

Milgram dan Kishino (1994) merumuskan kerangka kemungkinan penggabungan dan peleburan dunia nyata dan dunia maya ke dalam sebuah kontinum virtualitas. Sisi yang paling kiri adalah lingkungan nyata yang hanya berisi benda nyata, dan sisi paling kanan adalah lingkungan maya yang berisi benda maya.

Dalam realitas tertambah, yang lebih dekat ke sisi kiri, lingkungan bersifat nyata dan benda bersifat maya, sementara dalam *augmented virtuality* atau *virtualitas* tertambah, yang lebih dekat ke sisi kanan, lingkungan bersifat maya dan benda bersifat nyata. Realitas tertambah dan virtualitas tertambah digabungkan menjadi *mixed reality* atau realitas campuran.



Gambar 2. 5 Konsep Kerangka Augmented Reality.

2.8.3. Sejarah *Augmented Reality*

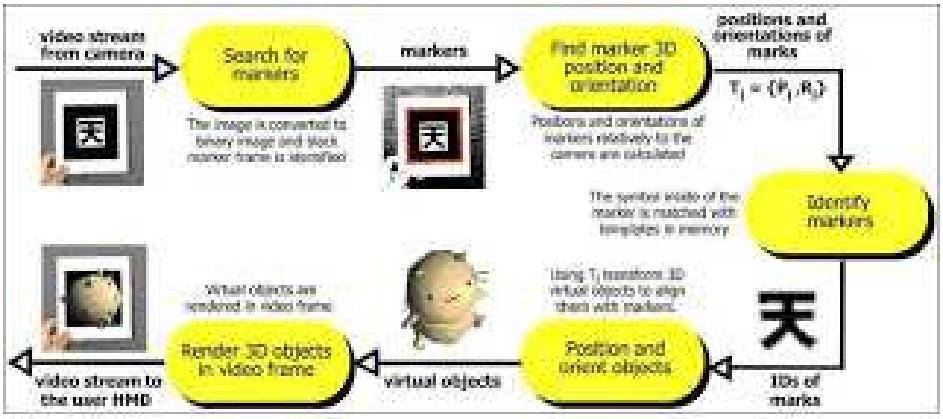
Konsep pertama *augmented reality* dikenalkan oleh Morton Heilig, seorang *cinematographer* pada tahun 1950an. Ketika itu *Augmented Reality* membutuhkan sebuah alat yang besar sebagai alat *output*. Alat *output* dapat berupa yang dipasang ditubuh kita (dikenal dengan nama *HMD, Head Mounted Device*), ada juga yang berupa *monitor*, seperti *monitor TV, LCD, monitor* ponsel, dll. Alat *HMD* pertama kali ditemukan pada tahun 1968 oleh Ivan Sutherland dari Harvard University. *Augmented reality* dengan *input* berupa sensor *GPS* diperkenalkan pada tahun 2003 dari hasil penelitian Loomis, dkk pada karya ilmiahnya *Personal guidance system for the visually impaired using GPS, GIS, and VR technologies*, pada tahun 1994.

Pada tahun 1996, Rekimoto dalam karya ilmiahnya *Augmented Reality Using the 2D Matrix Code. In Proceedings of the Workshop on Interactive Systems and Software* memperkenalkan *marker 2D* untuk pertama kalinya. Dua tahun kemudian *ARtoolkit, augmented reality library* pertama kali diluncurkan oleh Kato.

Pada tahun 2009 Lab MIT (Mistry, dkk) meneliti *sixth sense project* dan *Wear Ur World-A Wearable Gestural Interface* dimana *augmented reality* di implementasikan pada kehidupan sehari-hari.

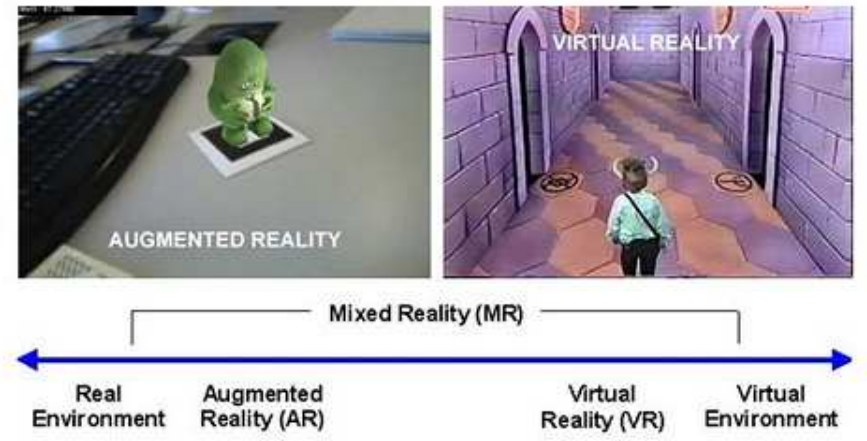
2.8.4. Prinsip Kerja *Augmented Reality*

Sistem *Augmented Reality* bekerja berdasarkan deteksi citra dan citra yang digunakan adalah *marker*. Prinsip kerjanya sebenarnya cukup sederhana. Kamera yang telah dikalibrasi akan mendeteksi *marker* yang diberikan, kemudian setelah mengenali dan menandai pola *marker*, kamera akan melakukan perhitungan apakah *marker* sesuai dengan *database* yang dimiliki. Bila tidak, maka informasi *marker* tidak akan diolah, tetapi bila sesuai maka informasi *marker* akan digunakan untuk me-*render* dan menampilkan objek 3D atau animasi yang telah dibuat sebelumnya.



Gambar 2. 6 Prinsip Kerja Augmented Reality.

2.8.5. Perbedaan *Augmented Reality* dengan *Virtual Reality*



Gambar 2. 7 Augmented Reality VS Virtual Reality.

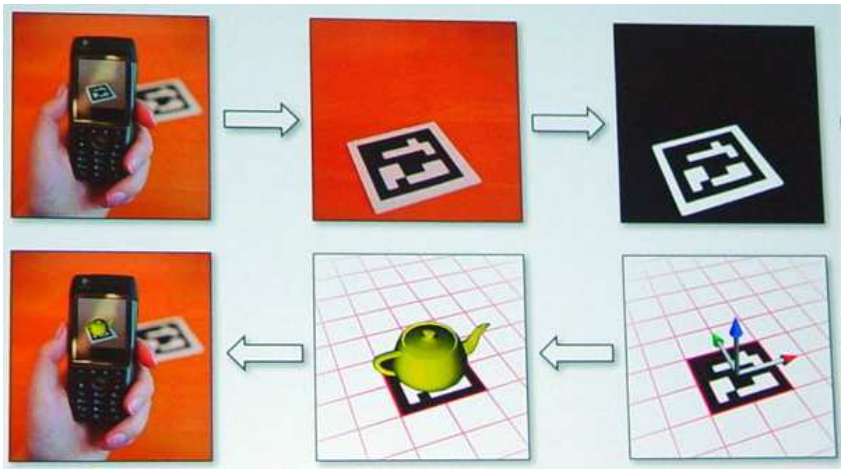
Virtual Reality mengacu pada penggabungan dari objek dunia nyata ke dunia digital/maya. *Augmented Reality* merupakan kebalikan dari *Virtual reality* yang berarti integrasi elemen-elemen digital yang ditambahkan ke dalam dunia nyata secara realtime dan mengikuti keadaan lingkungan yang ada di dunia nyata.

2.8.6. Metode *Augmented Reality*

Berikut penjelasan mengenai metoda yang digunakan pada *Augmented Reality* dalam menangkap dan mengenali tanda yang dijadikan acuan, berdasarkan ada tidaknya penanda tersebut :

1. *Marker Augmented Reality* (*Marker Based Tracking*)

   Aplikasi *augmented* ini berjalan dengan memindai tanda atau yang lebih sering disebut sebagai *marker*. *Marker* biasanya merupakan ilustrasi hitam dan putih persegi dengan batas hitam tebal dan latar belakang putih. Komputer akan mengenali posisi dan orientasi *marker* dan menciptakan dunia virtual 3D yaitu titik (0,0,0) dan 3 sumbu yaitu X,Y,dan Z.



Gambar 2. 8 Marker Based Tracking.

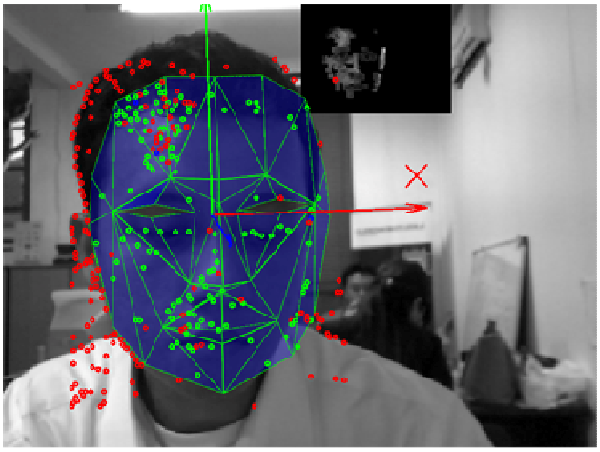
2. *Markerless Augmented Reality*

   Salah satu metode *Augmented Reality* yang saat ini sedang berkembang adalah metode "*Markerless Augmented Reality*", dengan metode ini pengguna tidak perlu lagi menggunakan sebuah *marker* untuk menampilkan elemen-elemen digital. Sekalipun dinamakan dengan

markerless namun aplikasi tetap berjalan dengan melakukan pemindaian terhadap *object*, namun ruang lingkup yang dipindai lebih luas dibanding dengan *marker* AR. Teknik *Markerless Tracking* sebagai teknologi meliputi *Face Tracking*, *3D Object Tracking*, *Motion Tracking* dan GPS *Based Tracking*

1) *Face Tracking*.

Ciri pada wajah setiap manusia berbeda-beda, namun pasti setiap manusia memilki mata, hidung dan mulut. Penentuan titik koordinat bentuk pola wajah menggunakan algoritma *Viola-Jones*. Dengan algoritma *Viola-Jones*, teknik ini dapat mengenali pola pada mata, hidung dan mulut dan mengabaikan objek sekitarnya sehingga objek digital dapat diimpelementasikan pada wajah melalui sebuah kamera.



Gambar 2. 9 Face Tracking.

2) *3D Object Tracking*

Berbeda dengan *Face Tracking* yang hanya mengenali pola wajah secara umum, teknik *3D Object Tracking* dapat mengenali semua bentuk benda yang ada disekitarnya seperti televisi, *handphone*, mobil, rumah dan lain-lain. Teknik ini dapat memvisualisasikan hal-hal umum.



Gambar 2. 10 Object Tracking.

3) *Motion Tracking*

Metode *markerless* dengan teknik ini yaitu dengan menangkap gerakan yang dilakukan pengguna. Salah satu contoh *Augmented Reality* dengan menggunakan teknik ini yaitu pada film Avatar. Dunia film sudah menggunakan teknologi ini dengan sangat ekstensif karena dapat mensimulasikan gerakan secara real-time.

Motion tracking lebih dekat ke dunia *virtual reality*, karena di sini hasil yang tampak hampir keseluruhan dunia *virtual*. Dalam penerapannya, *motion tracking* memerlukan sebuah alat pendeteksi gerakan dari tubuh penggunanya. Salah satu alat tersebut adalah *Head Mounted Display* (*HMDs*), alat ini berfungsi sebagai indra penglihatan dunia *virtual*. Metode pengembangan yang dipakai dalam motion tracking yaitu metode *TDA*. Metode ini dibutuhkan karena inertial orientation sensors yang menjadi solusi dalam menangkap sebuah gerakan.



Gambar 2. 11 Motion Tracking.

4) *GPS Based Tracking*

Pengembangan teknik ini lebih diarahkan pada *smartphone*, karena teknologi *GPS* dan kompas yang tertanam pada *smartphone* tersebut. Dengan memanfaatkan fitur *GPS* yang berfungsi sebagai penentu lokasi pengguna pada saat itu berada sehingga lokasi terdekat yang ingin dituju dapat dilihat melalui implementasi *augmented reality*.

Teknik ini berguna sebagai pemandu selayaknya fungsi *GPS*, namun dilengkapi dengan *marker* informasi arah yang dituju. Dalam implementasinya, teknik ini mengharuskan tersambungnya koneksi *GPS* dan kebutuhan paket data yang ada pada smartphone, karena data-data lokasi yang dimiliki *GPS* memiliki akses langsung dari satelit agar cepat mendeteksi wilayah yang telah dijadikan sebuah objek *marker* informasi pada *Augmented Reality*. Akses *internet* memiliki fungsi sebagai pemanggilan data-data berupa *latitude, longitude,* serta informasi yang mendukung setiap lokasi yang disimpan pada *server* sehingga beban ukuran aplikasi dapat diminimalisir.

Teknik *GPS based tracking* sebenarnya membutuhkan peran kompas dan akselerometer sebagai pengatur ukuran layar secara horizontal dan vertikal agar *marker* lokasi dapat dilihat ketika kamera *handset* berada posisi yang sesuai dengan lokasi tersebut.

Namun ketika *handset* tidak berada dalam sudut pandang lokasi tersebut maka *marker* tersebut tidak akan tampak.

Kebanyakan teknik *GPS based tracking* sudah memiliki *engine* pembantu (*AR browser*) yang telah dikembangkan oleh beberapa perusahaan sehingga mempermudah untuk mengembangkan teknik ini sesuai dengan keinginan. Pada tugas akhir ini digunakan *AR Junaio Browser* sebagai engine untuk mengembangkan aplikasi menjadi lebih mudah digunakan.



Gambar 2. 12 GPS Based Tracking.

2.8.7. Manfaat *Augmented Reality*

Dengan teknologi *Augmented Reality*, banyak manfaat yang diperoleh dari berbagai macam bidang, seperti berikut ini:

1. Kesehatan

   Bidang ini merupakan salah satu bidang yang paling penting bagi sistem realitas tertambah. Contoh penggunaannya adalah pada pemeriksaan sebelum operasi, seperti *CT Scan* atau *MRI*, yang memberikan gambaran kepada ahli bedah mengenai anatomi internal pasien. Dari gambar-gambar ini kemudian pembedahan direncanakan. Realitas tertambah dapat diaplikasikan sehingga tim bedah dapat melihat data *CT Scan* atau *MRI* pada pasien saat pembedahan berlangsung. Penggunaan lain adalah untuk pencitraan *ultrasonik*, di mana *teknisi*

ultrasonik dapat mengamati pencitraan *fetus* yang terletak di *abdomen* wanita yang hamil.

2. Manufaktur dan Reparasi

Bidang lain di mana realitas tertambah dapat diaplikasikan adalah pemasangan, pemeliharaan, dan reparasi mesin-mesin berstruktur kompleks, seperti mesin mobil. Instruksi-instruksi yang dibutuhkan dapat dimengerti dengan lebih mudah dengan realitas tertambah, yaitu dengan menampilkan gambar-gambar tiga dimensi di atas peralatan yang nyata. Gambar-gambar ini menampilkan langkah-langkah yang harus dilakukan untuk menyelesaikannya dan cara melakukannya. Selain itu, gambar-gambar tiga dimensi ini juga dapat dianimasikan sehingga instruksi yang diberikan menjadi semakin jelas.

Beberapa peneliti dan perusahaan telah membuat beberapa *prototipe* di bidang ini. Perusahaan pesawat terbang Boeing tengah mengembangkan teknologi realitas tertambah untuk membantu teknisi dalam membuat kerangka kawat yang membentuk sebagian dari sistem elektronik pesawat terbang. Kini, untuk membantu pembuatannya teknisi masih menggunakan papan-papan besar yang perlu disimpan di beberapa gudang penyimpanan yang terpisah. Menyimpan instruksi-instruksi pembuatan kerangka kawat ini dalam bentuk elektronik dapat menghemat tempat dan biaya secara signifikan.

3. Hiburan

Bentuk sederhana dari realitas tertambah telah dipergunakan dalam bidang hiburan dan berita untuk waktu yang cukup lama. Contohnya adalah pada acara laporan cuaca dalam siaran televisi di mana wartawan ditampilkan berdiri di depan peta cuaca yang berubah. Dalam studio, wartawan tersebut sebenarnya berdiri di depan layar biru atau hijau. Pencitraan yang asli digabungkan dengan peta buatan komputer menggunakan teknik yang bernama *chroma-keying*.

Princeton Electronic Billboard telah mengembangkan sistem realitas tertambah yang memungkinkan lembaga penyiaran untuk memasukkan iklan ke dalam area tertentu gambar siaran. Contohnya, ketika menyiarkan sebuah pertandingan sepak bola, sistem ini dapat menempatkan sebuah iklan sehingga terlihat pada tembok luar stadium.

4. Pelatihan Militer

Kalangan militer telah bertahun-tahun menggunakan tampilan dalam kokpit yang menampilkan informasi kepada pilot pada kaca pelindung kokpit atau kaca depan helm penerbangan mereka. Ini merupakan sebuah bentuk tampilan realitas tertambah. *SIMNET*, sebuah sistem permainan simulasi perang, juga menggunakan teknologi realitas tertambah. Dengan melengkapi anggota militer dengan tampilan kaca depan helm, aktivitas unit lain yang berpartisipasi dapat ditampilkan.

Contohnya, seorang tentara yang menggunakan perlengkapan tersebut dapat melihat helikopter yang datang. Dalam peperangan, tampilan medan perang yang nyata dapat digabungkan dengan informasi catatan dan sorotan untuk memperlihatkan unit musuh yang tidak terlihat tanpa perlengkapan ini.

5. Navigasi Telepon Genggam

Dalam kurun waktu 1 tahun terakhir ini, telah banyak *integrasi Augmented Reality* yang dimanfaatkan pada telepon genggam. Saat ini ada *Operating System* telepon genggam yang secara langsung memberikan dukungan terhadap teknologi *Augmented Reality* melalui antarmuka pemrograman aplikasinya masing-masing. Dengan menggunakan kamera sebagai sumber aliran data visual, maka sistem operasi tersebut harus mendukung penggunaan kamera dalam modus *video*.

Augmented Reality adalah sebuah presentasi dasar dari aplikasi-aplikasi navigasi. Dengan menggunakan *GPS* maka aplikasi pada telepon genggam dapat mengetahui keberadaan penggunanya pada setiap waktu.

Berbagai macam aplikasi telah menggunakan teknologi *Augmented Reality* digabungkan dengan lokasi sebagai presentasi untuk menampilkan titik-titik di sekitar dengan radius tertentu. Hal ini memungkinkan pengembang aplikasi untuk membuat *fitur* pemberian arah (dalam bahasa inggrisnya disebut *turn-by-turn*) lalu menampilkan dan atau menyuarakan kepada penggunanya untuk membelokkan arah.

Khusus untuk sistem operasi *iPhone* dan *Android*, ada 2 pengembang terbesar (*Windows* dan *Wikitude*) di dunia Realitas Tertambah yang telah membuka antarmuka pemrograman aplikasi mereka untuk dapat dipergunakan secara gratis dengan syarat dan prasyarat tertentu.

2.9. *Android*

Pembahasan mengenai *android* dan *tools* yang digunakan dalam pembuatan aplikasi, akan dijelaskan pada sub bab berikut :

2.9.1. Pengertian *Android*

Android adalah sistem operasi berbasis *Linux* yang dirancang untuk perangkat seluler layar sentuh seperti telepon pintar dan komputer tablet. *Android* adalah sistem operasi *open source*, dan *Google* merilis kodenya di bawah lisensi *Apache*. Kode *open source* dan lisensi perizinan pada *Android* memungkinkan perangkat lunak untuk dimodifikasi secara bebas dan didistribusikan oleh para pembuat perangkat, operator nirkabel, dan pengembang aplikasi.

2.9.2. Sejarah *Android*

Android, Inc. didirikan di Palo Alto, California, pada bulan Oktober 2003 oleh Andy Rubin, Rich Miner, Nick Sears, dan Chris White untuk mengembangkan "perangkat seluler pintar yang lebih sadar akan lokasi dan preferensi penggunanya". *Google* mengakuisisi *Android Inc.* pada tanggal 17 Agustus 2005, menjadikannya sebagai anak perusahaan yang sepenuhnya dimiliki oleh *Google.* Pendiri *Android Inc*. seperti Rubin, Miner dan White tetap bekerja di perusahaan setelah diakuisisi oleh *Google*.

Pada tanggal 5 November 2007, *Open Handset Alliance* (*OHA*) didirikan. *OHA* adalah konsorsium dari perusahaan-perusahaan teknologi seperti *Google*, produsen perangkat seluler seperti *HTC*, *Sony* dan *Samsung*, operator nirkabel seperti *Sprint Nextel* dan *T-Mobile*, serta produsen *chipset* seperti *Qualcomm* dan *Texas Instruments*. *OHA* sendiri bertujuan untuk mengembangkan standar terbuka bagi perangkat seluler. Saat itu, *Android* diresmikan sebagai produk pertamanya; sebuah *platform* perangkat seluler yang menggunakan *kernel Linux* versi 2.6. Telepon seluler komersial pertama yang menggunakan sistem operasi *Android* adalah *HTC Dream*, yang diluncurkan pada 22 Oktober 2008.

Sejak tahun 2008, *Android* secara bertahap telah melakukan sejumlah pembaruan untuk meningkatkan kinerja sistem operasi, menambahkan fitur baru, dan memperbaiki *bug* yang terdapat pada versi sebelumnya. Setiap versi utama yang dirilis dinamakan secara alfabetis berdasarkan nama-nama makanan pencuci mulut atau cemilan bergula; misalnya, versi 1.5 bernama *Cupcake*, yang kemudian diikuti oleh versi 1.6 *Donut*. Versi terbaru adalah 4.3 *Jelly Bean*. Pada tahun 2010, *Google* merilis seri *Nexus*, perangkat telepon pintar dan *tablet* dengan sistem operasi *Android* yang diproduksi oleh mitra produsen telepon seluler seperti *HTC*, *LG*, dan *Samsung*. *HTC* bekerjasama dengan *Google* dalam merilis produk telepon pintar *Nexus* pertama, yakni *Nexus One*. Seri ini telah diperbarui dengan perangkat yang lebih baru, misalnya telepon pintar *Nexus* 4 dan *tablet Nexus* 10 yang diproduksi oleh *LG* dan *Samsung*.

Pada 13 Maret 2013, Larry Page mengumumkan dalam postingan blognya bahwa Andy Rubin telah pindah dari divisi *Android* untuk mengerjakan proyek-proyek baru di *Google*. Ia digantikan oleh Sundar Pichai, yang sebelumnya menjabat sebagai kepala divisi *Google Chrome* yang mengembangkan *Chrome OS*.

2.9.3. 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].



Gambar 2. 13 Arsitektur Android.

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 servis 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. *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*).

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.

4. *Application Framework*

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 :

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, janji, dan lain sebagainya yang akan ditampilkan pada *status bar.*

5. *Aplication 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.

2.10. *Unified Modelling Language (UML)*

Unified Modeling Language (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OO) [6].

UML merupakan alat komunikasi yang konsisten dalam men-*support* para pengembang sistem saat ini. Sebagai perancang sistem mau tidak mau pasti akan menjumpai *UML*, baik kita sendiri yang membuat atau sekedar membaca diagram *UML* buatan orang lain [Pilone, 2005 : Bab 1].

UML 2.0 terdiri dari 13 jenis diagram yaitu *Activity, Class, Communication, Component, Composite Structure, Deployment, Interaction Overview, Object, Package, Sequence, State Machine (Statechart), Timing* dan *Use Case*. Namun dalam pembangunan aplikasi ini yang akan digambarkan hanya 7 diagram yaitu *Use Case, Activity, Class, Sequence, Statechart (State Machine), Component* dan *Deployment*.

2.10.1. Use Case Diagram

Use case diagram merupakan diagram yang memperlihatkan himpunan *use case* dan aktor-aktor (suatu jenis khusus dari kelas). Komponen pembentuk diagram meliputi aktor *(actor), use case* (aktifitas/sarana sistem), dan hubungan *(link)*. *Use case* menggambarkan fungsi tertentu dalam suatu sistem berupa komponen, kejadian atau kelas [Pilone, 2005 : Bab 7.1]. Sedangkan (Whitten, 2004 : 258) mengartikan *use case* sebagai urutan langkah-langkah yang secara tindakan saling terkait *(scenario)*, baik terotomatisasi maupun secara manual, untuk tujuan melengkapi satu tugas bisnis tunggal. *Use case* digambarkan dalam bentuk ellips/oval.

2.10.2. Activity Diagram

Activity Diagram adalah teknik menggambarkan logika prosedural, proses bisnis dan jalur kerja. Dalam beberapa hal, diagram ini memainkan peran mirip sebuah diagram alir, tetapi perbedaan prinsip antara diagram ini dan notasi diagram alir adalah diagram ini mendukung *behavior* paralel [6].

Activity Diagram lebih memfokuskan diri pada eksekusi dan alur sistem dari pada bagaimana sistem itu dirakit. Diagram ini menunjukkan aktifitas sistem dalam bentuk kumpulan aksi-aksi [7].

2.10.3. Class Diagram

Class adalah suatu set objek yang memiliki atribut dan perilaku yang sama. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi [7]. *Class diagram* merupakan diagram yang mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat diantaranya [6]. 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. *Class diagram* sangat membantu dalam visualisasi struktur kelas dari suatu sistem.

2.10.4. Sequence Diagram

Sequence diagram secara khusus menjabarkan *behavior* sebuah skenario tunggal. Diagram tersebut menunjukkan sejumlah objek contoh dan pesan-pesan yang melewati objek-objek ini di dalam *use case* [6]. Diagram ini bersifat dinamis dan merupakan diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu [7].

2.10.5. Statechart Diagram

Statechart diagram merupakan diagram yang memperlihatkan keadaan-keadaan pada sistem, memuat status *(state),* transisi, kejadian serta aktifitas. Diagram ini untuk memperlihatkan sifat dinamis dari antarmuka *(interface)*, kelas, kolaborasi dan terutama pada pemodelan sistem-sistem yang reaktif [7]. Diagram ini bermanfaat juga dalam menyediakan cara yang baik dalam memodelkan komunikasi yang terjadi dengan entitas luar via protokol atau sistem dasarnya [Pilone, 2005 : Bab 8].

2.10.6. Component Diagram

Component diagram memperlihatkan organisasi serta kebergantungan sistem/perangkat lunak pada komponen-komponen yang telah ada sebelumnya. Diagram ini berhubungan dengan diagram kelas di mana komponen secara tipikal dipetakan ke dalam satu atau lebih kelas-kelas, antarmuka-antarmuka serta kolaborasi-kolaborasi [7].

2.10.7. Deployment Diagram

Deployment diagram memperlihatkan konfigurasi saat aplikasi dijalankan *(run-time)*. Diagram ini berhubungan erat dengan diagram komponen dimana diagram ini memuat satu atau lebih komponen-komponen. Diagram ini sangat berguna saat aplikasi berlaku sebagai aplikasi yang dijalankan pada banyak mesin *(distributed computing)* [7].

2.11. *MySQL*

MySQL adalah sebuah perangkat lunak sistem 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 di bawah lisensi *GNU General Public License (GPL)*, tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan *GPL* [10].

Tidak seperti *PHP* atau *Apache* yang merupakan *software* yang dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh penulisnya masing-masing, *MySQL* dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia yaitu *MySQL AB*. *MySQL AB* memegang penuh hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan *MySQL AB* adalah : David Axmark, Allan Larsson, dan Michael "Monty" Widenius.

2.11.1. Fitur-Fitur *MySQL*

Fitur-fitur *MySQL* antara lain [10] :

1. *Relational Database System.*

   Seperti halnya *software database* lain yang ada di pasaran, *MySQL* termasuk *RDBMS (Relational Database Management System).*

2. Arsitektur *Client-Server.*

   MySQL memiliki arsitektur *client-server* dimana *server database MySQL* terinstal di *server*. *Client MySQL* dapat berada di komputer yang sama dengan *server*, dan dapat juga di komputer lain yang berkomunikasi dengan *server* melalui jaringan bahkan *internet*.

3. Mengenal perintah *SQL* standar.

   SQL (Structured Query Language) merupakan suatu bahasa standar yang berlaku di hampir semua *software database*. *MySQL* mendukung *SQL* versi *SQL*:2003.

4. Mendukung *sub select*.

   Mulai versi 4.1 MySQL telah mendukung *select* dalam *select (sub select).*

5. Mendukung *Views*.

   MySQL mendukung *views* sejak versi 5.0.

6. Mendukung *Stored Prosedured (SP).*

   MySQL mendukung *SP* sejak versi 5.0.

7. Mendukung *Triggers*.

   MySQL mendukung *trigger* pada versi 5.0 namun masih terbatas. Pengembang *MySQL* berjanji akan meningkatkan kemampuan *trigger* pada versi 5.1.

8. Mendukung *replication*.
9. Mendukung transaksi.
10. Mendukung *foreign key*.
11. Tersedia fungsi *GIS*.
12. *Free* (bebas didownload).
13. Stabil dan tangguh.
14. Fleksibel dengan berbagai pemrograman.
15. *Security* yang baik.
16. Dukungan dari banyak komunitas.
17. Perkembangan *software* yang cukup cepat.

2.12. *Hypertext Preprocessor (PHP)*

PHP (PHP: Hypertext Preprocessor) adalah sebuah bahasa pemogramaman di sisi *server* [14]. Ketika mengakses sebuah *URL*, maka *web browser* akan melakukan *request* ke sebuah *web server*. Misalnya akan me-*request* sebuah file *PHP* http://www.koder.web/index.php, maka *web server* akan melakukan *parsing* terhadap *file PHP* tersebut. *PHP parser* yang menjalankan kode-kode *PHP* yang terdapat pada *file index.php* lalu mengirimkan hasilnya ke *web browser*.

2.12.1. Sejarah *PHP*

Pada tahun 1994, Rasmus Lerdorf mengembangkan sebuah perkakas yang digunakan sebagai *engine* parsing sebagai penerjemah/*interpreter* beberapa *macro*. Ia menamai *engine parser* tersebut dengan nama *PHP/FI*. Sampai pada tahun 1997, lebih dari 500.000 *website* di dunia menggunakan *PHP/FI* untuk menyelesaikan masalah seperti koneksi ke *database*, menampilkan *content* yang dinamis dan lain-lain. Sampai pada tahun 1997, lebih dari 500.000 *website* di dunia menggunakan *PHP*/*FI* untuk menyelesaikan masalah seperti koneksi ke *database*, menampilkan *content* yang dinamis dan lain-lain [14].

2.12.2. Kelebihan *PHP*

Menurut Sukarno (2006, p10), kelebihan *PHP* adalah :

1. *PHP* merupakan sebuah bahasa *script* yang tidak melakukan kompilasi dalam penggunaanya. Tidak seperti halnya bahasa pemograman aplikasi seperti *Visual Basic* dan sebagainya.
2. *PHP* dapat berjalan pada *web server* yang diliris oleh *Microsoft, Script IIS* atau *PWS* juga pada *Apache* yang bersifat *open source.*
3. Karena sifatnya *open source*, maka perubahan dan perkembangan intrepreter pada *PHP* lebih cepat dan mudah, karena banyak milis-milis dan *developer* yang siap membantu pengembanganya.
4. Jika dilihat dari segi pemahaman, *PHP* memiliki referensi yang begitu banyak sehingga sangat mudah untuk dipahami.
5. *PHP* dapat berjalan pada 3 *operating system*, yaitu *Linux*, *Unix* dan *Windows*, dan juga dapat dijalankan secara *runtime* pada suatu *console*.

2.13. *Wamp Server 2.0*

Wamp Server merupakan paket *web server* yang bekerja secara pada lingkungan *localhost* yang di instal pada sistem operasi *Windows*. *WAMP* adalah merupakan singkatan dari dari *Windows and the principal components of the package*: *Apache*, *MySQL* and *PHP* (*Perl*/*Python*). *Apache* sebagai *web server*, *MySQL* sebagai *database*, *PHP* sebagai bahasa *scripting* yang dapat

memanipulasi informasi yang dibuat di *database* dan menghasilkan halaman *web.* Pada *wamp* juga terdapat *phpMyAdmin* yang menyediakan antarmuka pengguna grafis untuk mengolah *database MySQL*.

2.14. 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* [id.wikipedia : Eclipse (Perangkat Lunak)] :

1. *Multi-platform.*

   Target sistem operasi *Eclipse* adalah *Microsoft Windows*, *Linux*, *Solaris*, *AIX*, *HP-UX* dan *Mac OS X*.

2. *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.

3. *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.*

2.14.1. Sejarah *Eclipse*

Eclipse awalnya dikembangkan oleh *IBM* untuk menggantikan perangkat lunak *IBM Visual Age for Java 4.0*. Produk ini diluncurkan oleh *IBM* pada tanggal 5 November 2001, yang menginvestasikan sebanyak US$ 40 juta untuk

pengembangannya. Semenjak itu konsursium *Eclipse Foundation* mengambil alih untuk pengembangan *Eclipse* lebih lanjut dan pengaturan organisasinya [id.wikipedia : Eclipse (Perangkat Lunak)].

2.14.2. 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 [id.wikipedia : Eclipse (Perangkat Lunak)] :

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

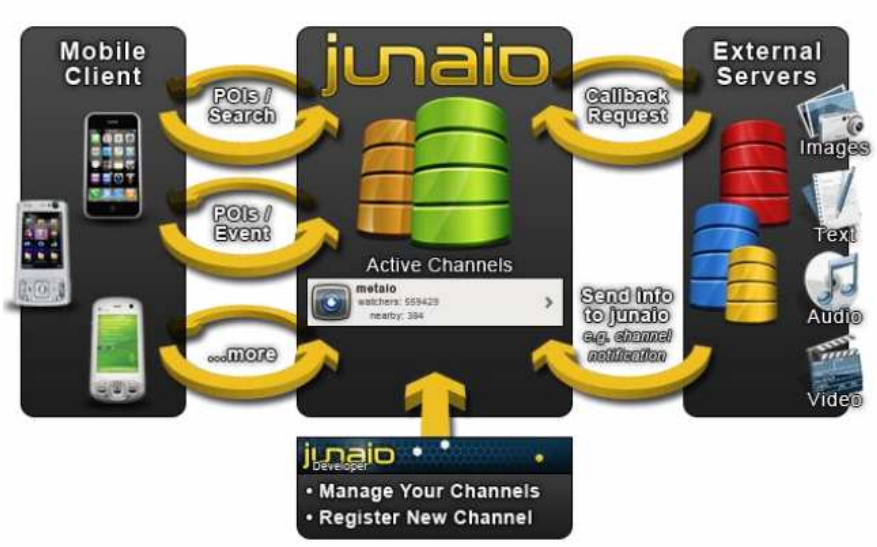
Secara standar *Eclipse* selalu dilengkapi dengan *JDT* (*Java Development Tools*), *plug-in* yang membuat *Eclipse* kompatibel untuk mengembangkan program *Java*, dan *PDE* (*Plug-in Development Environment*) untuk mengembangkan *plug-in* baru. *Eclipse* beserta *plug-in*-nya diimplementasikan dalam bahasa pemrograman *Java*.

Konsep *Eclipse* adalah *IDE* yang terbuka *(open),* 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 UML2* tersedia untuk membuat diagram *UML*. Dengan menggunakan PDE setiap orang bisa membuat plug-in sesuai dengan keinginannya. Salah satu situs yang menawarkan *plug-in* secara gratis seperti *Eclipse downloads by project.*

2.15. *Adobe Dreamweaver CS5*

Merupakan sebuah aplikasi *software* yang berguna membantu dalam membuat, merancang, men-*design* sebuah *website* yang dapat dijalankan secara *offline*, *software* ini mempunyai fitur yang lengkap dengan pengecekkan *error* pada *coding PHP, HTML, JavaScript* dan *CSS* yang dapat memaksimalkan dalam pembuatan *website. Software* ini juga mendukung pembuatan *website* dengan tampilan *GUI.* Yang sangat membantu pengguna yang belum memahami pengkodean *website.*

2.16. *Junaio Browser*



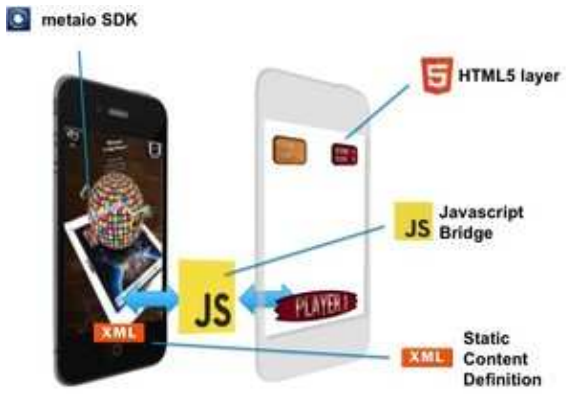
Gambar 2. 14 Arsitektur Junaio Browser [Junaio : Doc].

Junaio browser merupakan browser yang digunakan untuk menampilkan *AR* pada perangkat *mobile.* Pada dasarnya *browser* ini di dalam implementasinya memerlukan *backend server junaio* itu sendiri. *Server* tersebut digunakan untuk men-*generate channel* yang sudah di daftarkan. Kelebihannya dengan menerapkan sistem ini memudahkan dalam pengkodean dan tidak terlalu membebankan kepada perangkat yang digunakan. Namun *browser* ini juga tentunya memiliki kekurangan yaitu aplikasi yang dibangun sangat bergantung pada status *server backend junaio*.

2.17. *AREL (Augmented Reality Experience Language)*

Dalam komunikasi antara *client, server junaio, server hosting, junaio* menggunakan teknologi bahasa pemrograman *AREL (Augmented Reality Experience Language)* untuk membaca dan menggambarkan posisi dalam bentuk *AR*. *AREL* merupakan komunikasi dari *Javascript* dengan *Metaio SDK API* dalam kombinasi dengan konten definisi *XML statis*.

Bagian *XML* mendefinisikan konten mana yang harus diambil, seperti model *3D* atau *billboard* serta mendefinisikan sifat awal objek tersebut seperti ukuran, sistem koordinat, transformasi, dan lain-lain. Lapisan *HTML5* menyediakan antarmuka pengguna grafis dan berinteraksi menggunakan *JavaScript* dengan *Metaio SDK*. Semua *callback* dari *metaio SDK* akan diteruskan ke logika aplikasi *JavaScript*.



Gambar 2. 15 Gambaran AREL [AREL : Overview].

2.18. *JSON (JavaScript Object Notation)*

JSON adalah *format* pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer [json.org]. *JSON* pada aplikasi digunakan sebagai pertukaran data multimedia yang di-*request* oleh perangkat *mobile (client)* dengan *server*.

Format ini dibuat berdasarkan bagian dari bahasa pemrograman *JavaScript*, *Standar ECMA-262* Edisi ke-3– Desember 1999. *JSON* merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh *programmer* keluarga

C termasuk *C*, *C++*, *C#*, *Java*, *JavaScript*, *Perl*, *Python* dan lain-lain. Oleh karena sifat-sifat tersebut, menjadikan *JSON* ideal sebagai bahasa pertukaran data.

JSON terbuat dari dua struktur [json.org] :

1. Kumpulan pasangan nama/nilai.

   Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel *hash* (*hash table*), daftar berkunci (*keyed list*), atau *associative array*.

2. Daftar nilai terurutkan *(an ordered list of values).*

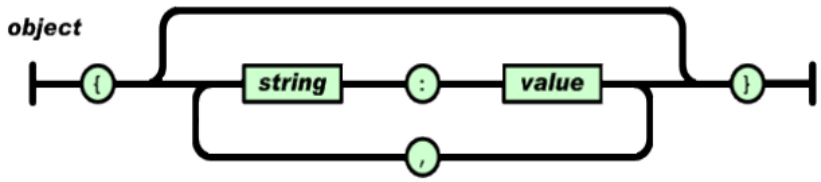
   Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

Struktur-struktur data ini disebut sebagai struktur data *universal*. Pada dasarnya, semua bahasa pemrograman modern mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemrograman yang juga berdasarkan pada struktur data ini.

JSON menggunakan bentuk sebagai berikut [json.org] :

1. Objek adalah sepasang nama/nilai yang tidak terurutkan.

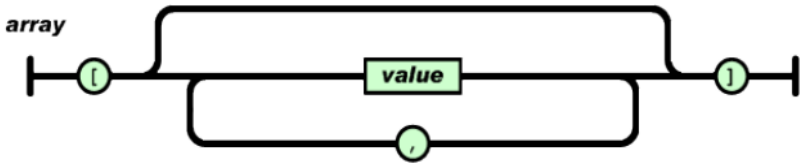
   Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma).



Gambar 2. 16 Objek JSON.

2. Larik adalah kumpulan nilai yang terurutkan.

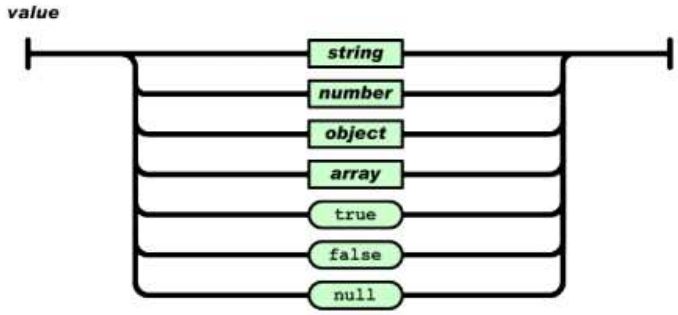
Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma).



Gambar 2. 17 Larik JSON.

3. Nilai *(value)*.

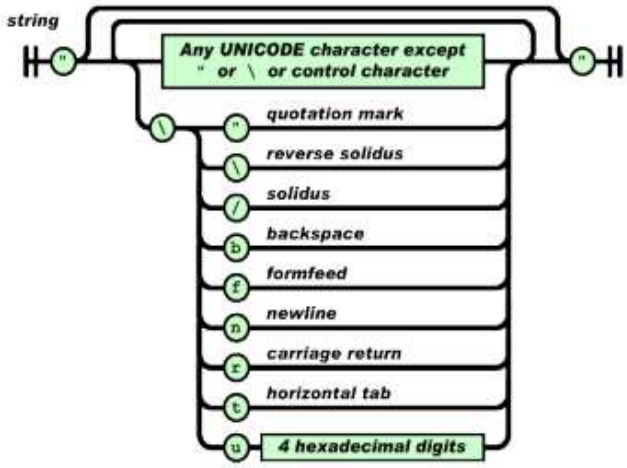
Dapat berupa sebuah string dalam tanda kutip ganda, atau angka, atau true atau false atau null, atau sebuah objek atau sebuah larik. Struktur-struktur tersebut dapat disusun bertingkat.



Gambar 2. 18 Nilai JSON.

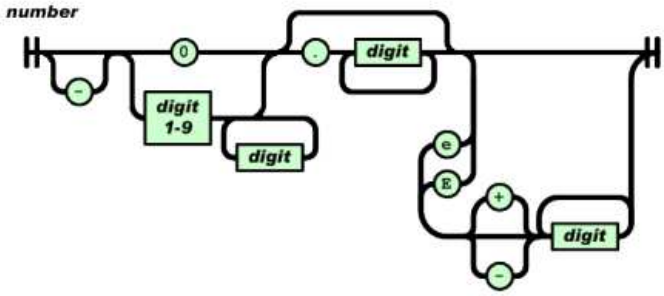
4. *String* adalah kumpulan dari nol atau lebih karakter *unicode*, yang dibungkus dengan tanda kutip ganda.

Di dalam string dapat digunakan *backslash escapes* "\" untuk membentuk karakter khusus. Sebuah karakter mewakili karakter tunggal pada *string*. *String* sangat mirip dengan string *C* atau *Java*.



Gambar 2. 19 String JSON.

5. Angka adalah sangat mirip dengan angka di *C* atau *Java*, kecuali format *oktal* dan *heksadesimal* tidak digunakan.



Gambar 2. 20 Angka JSON.