BAB II

LANDASAN TEORI

2.1. Internet of Things

Kevin Ashton seorang pelopor teknologi yang juga membuat sistem standar global untuk *RFID* dan sensor lainnya mengatakan bahwa hampir semua data yang beredar di *internet* berasal dari hasil *input* atau hasil *capture* yang dilakukan oleh manusia ke dalam sistem. Dari sudut pandang sistem, manusia adalah obyek yang lambat, rawan kesalahan, pengantar data yang tidak efisien dan memiliki batasan dalam hal kualitas dan kuantitas, bahkan kadang mencoba menterjemahkan dan mengubah data tersebut. Sebagai alternatif akan lebih efisien jika sistem dapat terkoneksi dengan sensor yang dapat menterjemahkan kejadian di dunia nyata secara langsung. Jadi, di masa depan, sistem tidak memerlukan perantara manusia dan tersambung secara langsung ke sensor dan internet untuk mencatat data yang diambil dari dunia nyata. Sehingga bisa dikatakan bahwa *Internet of Things* (IoT) adalah ketika kita menyambungkan sesuatu (*things*), yang tidak dioperasikan oleh manusia, ke internet. (Peter Waher, 2015)

2.2. Raspberry Pi

Raspberry Pi merupakan sebuah komputer sebesar kartu kredit yang dikembangkan di Inggris oleh Raspberry Pi Foundation. Gagasan di balik sebuah komputer kecil dan murah untuk anak-anak muncul pada tahun 2006.

Ide ini muncul ketika beberapa mahsiswa Laboratorium Komputer di Unversitas Cambridge, yakni Eben Upton, Rob Mullins, Jack Lang, dan Alan Mycroft. Nama Raspberry Pi diambil dari nama buah, yaitu buah Raspberry, sedangkan Pi diambil dari kata Python, yaitu nama dari sebuah bahasa pemrograman. Python dijadikan bahasa pemrograman utama dari Raspberry Pi, namun tidak tertutup kemungkinan untuk menggukan bahasa pemrograman lain pada Raspberry Pi.

Raspberry Pi memiliki komponen yang hampir serupa dengan PC pada umumnya, seperti CPU, GPU, RAM, Port USB, Audio Jack, HDMI, Ethernet, dan GPIO. Untuk tempat penyimpanan data dan Sistem Operasi Raspberry Pi tidak menggukan *harddisk drive (HDD)* melainkan menggukan *Micro SD* dengan kapasitas paling tidak 4GB, sedangkan untuk sumber tenaga berasal dari *micro USB* power dengan sumber daya yang direkomendasikan yaitu sebesar 5V dan minimal arus 700 mA. Raspberry Pi dapat digunakan layaknya PC konvensional, seperti untuk mengetik dokumen atau sekedar *browsing*, namun Raspberry Pi dapat juga digunakan untuk membuat ide-ide inovatif seperti membuat robot yang dilengkapi dengan Raspberry Pi dan kamera, atau mungkin dapat membuat sebuah super computer yang dibuat dari beberapa buah Raspberry Pi. (Edi Rakhman, dkk., 2014)

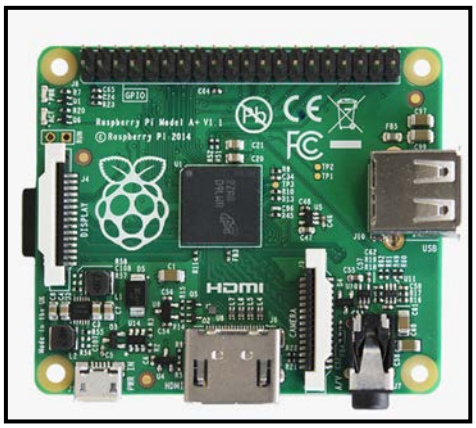
2.2.1. Model Raspberry Pi

Raspberry Pi memiliki beberapa model produk yang saat ini beredar luas di pasaran, ukuran dari tiap model tidak jauh berbeda, yaitu memiliki

ukuran sebesar kartu kredit, yang membedakan dari tiap model adalah spesifikasi dan perangkat yang terpasang pada *Raspberry Pi* tersebut. Beberapa model yang beredar di pasaran saat ini adalah model *Raspberry Pi 1* model A+, *Raspberry Pi 1* model B+, dan *Raspberry Pi 2* model B.

1. *Raspberry Pi 1* Model A+

   Model ini mulai beredar pada bulan November 2014, untuk menggantikan model sebelumnya yaitu *Raspberry Pi* model A. Model ini direkomendasikan untuk digunakan dalam *project* yang tidak membutuhkan konektifitas *Ethernet* dan konsumsi listrik yang kecil. Keunggulan dari model ini adalah bentuknya yang relatif lebih kecil ketimbang dua model lain.

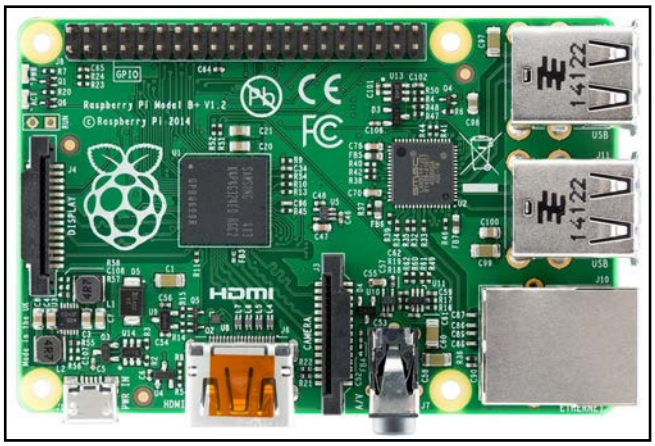


Gambar 2.1 *Raspberry Pi 1 Model A+*

2. *Raspberry Pi 1* Model B+

   Model B+ diluncurkan pada bulan Juli 2014, memiliki spesifikasi 4 buah *port USB*, 40 *pin GPIO*, *Ethernet*, *HDMI*, *port* audio dan *micro sd* sebagai tempat penyimpanan sistem operasi. Model ini direkomendasikan untuk para pemula yang baru mulai belajar untuk

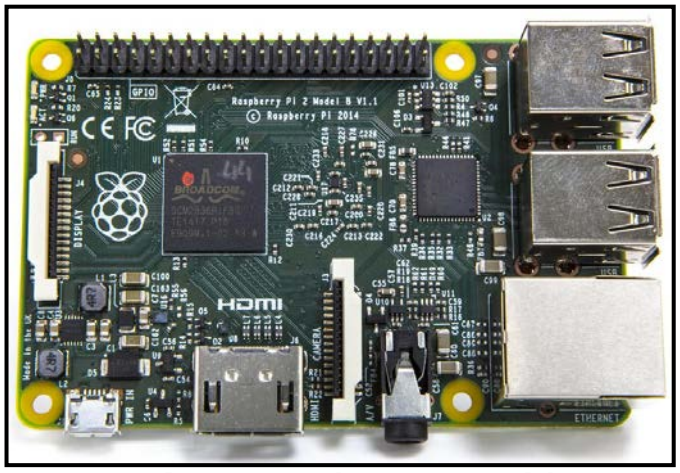
membuat *project* menggunakan *Raspberry Pi*. Dibanding *Raspberry Pi model A+*, model ini memiliki kapasitas RAM yang lebih besar, yaitu 512 MB, dan juga memiliki sebuah *port Ethernet* dan *port USB* yang lebih banyak.



Gambar 2.2 *Raspberry Pi 1 Model B+*

3. *Raspberry Pi 2* Model B

Raspberry Pi 2 adalah generasi kedua dari *Raspberry Pi*. Model ini dirilis pada bulan Februari 2015 menggantikan *Raspberry Pi 1* Model B+. Dibanding pendahulunya, model ini memiliki prosesor yang lebih mumpuni, yaitu prosesor *Quad-Core ARM Cortex-A7 CPU* dengan kecepatan 900MHz, dan juga dilengkapi dengan 1GB RAM. Untuk bentuk dan fitur, model ini memiliki kemiripan dengan model *Raspberry Pi 1* model B+.



Gambar 2.3 *Raspberry Pi 2 Model B*

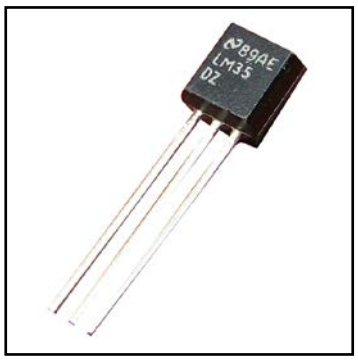
2.3. Sensor

Sensor adalah komponen elektronik yang berfungsi sebagai perangkat masukan (*input Devices*). Tidak semua *input* merupakan sensor, tetapi hampir semua *input* menggunakan sensor. Misalnya saja *mouse* pada komputer atau *trackpad*, *keyboard*, atau bahkan *webcam*; perangkat-perangkat tersebut bukanlah merupakan sensor, tetapi sudah pasti salah satu komponen dalam perangkat tersebut merupakan sensor. Secara abstrak sensor adalah sebuah komponen untuk mengukur stimulus yang berada di luar sistem, kemudian data yang dihasilkan adalah berdasarkan pengukuran yang telah dilakukan. (Kimmo dan Tero Karvinen, 2014)

2.3.1. Sensor Suhu LM35

LM35 adalah komponen sensor suhu terkalibrasi yang berukuran kecil dengan tegangan keluaran terskala linear dengan suhu terukur, yakni 10 mV per 1° Celsius. komponen ini mampu mengukur suhu hingga 100° Celsius. LM35 disuplai dengan tegangan mulai dari 4V hingga 30V DC

dengan arus pengurasan sebesar 60 μA. LM35 memiliki tingkat *self-heating* yang rendah, yakni kurang dari 0,1° Celsius pada keadaan udara normal.



Gambar 2.4 Sensor suhu (LM35)

2.4. Internet

Internet adalah kumpulan dari jaringan yang menghubungkan komputer di seluruh dunia. Komputer terkoneksi ke internet dan berkomunikasi dengan menggunakan *internet protocol* (IP), yang membelah informasi menjadi *packet* (potongan data yang ditransmisikan secara terpisah) dan mengarahkannya ke tujuannya. Bersamaan dengan IP hampir semua komputer yang tersambung ke internet berkomunikasi dengan *Transmission Control Protocol (TCP),* dan kombinasi keduanya dinamakan TCP/IP.

2.5. Web Site

Web site adalah kumpulan dari banyak halaman, biasanya dalam format HTML (Hypertext Markup Language), yang berisi teks, grafis, dan elemen multimedia seperti *flash*, audio, ataupun video. Halaman utama dari sebuah *site* biasanya disebut dengan *Home Page*, berisi tautan menuju ke dokumen

lain di *site* tersebut dengan menggunakan *Hyperlinks*. Semua halaman web disimpan pada sebuah *web server*. (Richard Wagner, 2011)

2.5.1. HTML

Sebuah halaman *web* ditulis dalam *Hypertext Markup Language (HTML)*, sebuah bahasa berbasis *tag* yang digunakan untuk menyajikan informasi. HTML terdiri dari dua tipe data:

a. *Content*: teks dan grafis yang akan ditampilkan pada halaman *web*
b. *Instructions*: elemen format yang terdefinisi, atau *tags*, untuk menentukan bagaimana teks dan grafis akan ditampilkan dan disusun pada sebuah halaman. Instruksi ini tidak akan tampak saat halaman *web* ditampilkan pada *web browser*. (Richard Wagner, 2011)

2.5.2. Cascading Style Sheet (CSS)

Cascading style sheet (CSS) adalah sebuah bahasa untuk mendeskripsikan tampilan dari sebuah dokumen yang ditulis dalam bahasa *markup* seperti HTML misalnya. Dengan CSS maka akan dapat mengatur warna tulisan, bentuk huruf, jarak antar paragraf, ukuran kolom, gambar latar atau warna latar yang akan digunakan, dan berbagai variasi efek visual. Salah satu keuntungan utama dari CSS adalah satu CSS yang sama dapat digunakan lebih dari satu halaman, yang artinya keseluruhan *style* dari satu *website* dapat diatur tanpa harus mengganti di tiap halamannya. CSS biasanya digunakan untuk mengatur tampilan visual dari sebuah halaman *web*, dan dikombinasikan dengan HTML atau XHTML (digunakan untuk

mendeskripsikan konten) dan *JavaScript* (yang digunakan untuk menambahkan kesan interaktif pada sebuah halaman). (Ian Pouncey dan Richard York, 2011)

2.5.3. PHP

PHP adalah bahasa pemrograman untuk pengembangan *web* di sisi server paling populer, diperkirakan digunakan oleh paling tidak 78.9% dari semua *website* yang beredar.

PHP dibuat oleh Rasmus Lerdorf di tahun 1995, dan awalnya PHP adalah akronim dari "Personal Home Page (Tools)", meskipun sekarang lebih dikenal sebagai akronim rekursif dari "PHP: Hypertext Preprocessor". Bahasa pemrograman PHP dikelola, diawasi, dan dikembangkan oleh sebuah kelompok pengembang yang dikenal dengan "The PHP Group", yang terus mendistribusikan bahasa pemrogramannya secara gratis lewat website resmi PHP.

Kode PHP pada umumnya diinterpretasikan, diproses, dan diterjemahkan menggunakan *web server* dengan sebuah *module* PHP yang sudah terpasang pada server tersebut, sehingga memungkinkan PHP untuk ditanamkan pada sebuah dokumen markup HTML dengan ekstensi ".**php"**.

PHP digunakan untuk menangani pemrosesan data yang kompleks agar data dinamis dapat muncul di halaman *web*, misalnya seperti kalkulasi matematis, dan interaksi dengan *database*. PHP membuat developer dapat menjadikan HTML yang awalnya hanya berisi konten statis menjadi sebuah

halaman yang responsif terhadap permintaan pengguna. (Callum Hopkins, 2013)

2.5.4. JavaScript

JavaScript adalah sebuah bahasa pemrograman yang dapat menambahkan animasi, interaksi, dan efek visual dinamis ke HTML. *JavaScript* dapat membuat halaman *web* menjadi lebih berarti dengan menambahkan umpan balik yang cepat. Sebagai contoh, *JavaScript* dapat memberikan pesan kesalahan dengan segera saat pengguna melakukan *submit* pada sebuah *web form* namun ada salah informasi yang tidak lengkap. (David Sawyer McFarland, 2011)

2.5.5. Bootstrap

Bootstrap adalah *front-end framework* yang di dalamnya terdapat CSS dan *JavaScript* untuk mempermudah pengembang dalam memulai pengembangan sebuah *web*. Pengembang yang beralih ke pengembangan *front-end* dari bahasa pemrograman untuk sisi server seperti *Java* atau PHP akan merasakan kesulitan saat harus berurusan dengan CSS dan *JavaScript*; namun dengan *bootstrap* para pengembang dapat berkonsentrasi hanya pada penulisan HTML yang baik, dan menyerahkan CSS dan *JavaScript* kepada *Bootstrap*.

Bootstrap dikembangkan pada tahun 2011 oleh dua orang pengembang *web* di *twitter*, Mark Otto dan Jacob Thornton. Tujuan utamanya adalah untuk menjaga konsistensi dan mempermudah perawatan pada kode yang mereka buat. (Syed Fazle Rahman, 2014)

2.6. Database MySQL

Database adalah sebuah koleksi dokumen yang terstruktur atau data yang tersimpan pada sebuah sistem komputer dan diatur sedemikian rupa sehingga dapat dicari dengan cepat dan informasi bisa segera didapatkan.

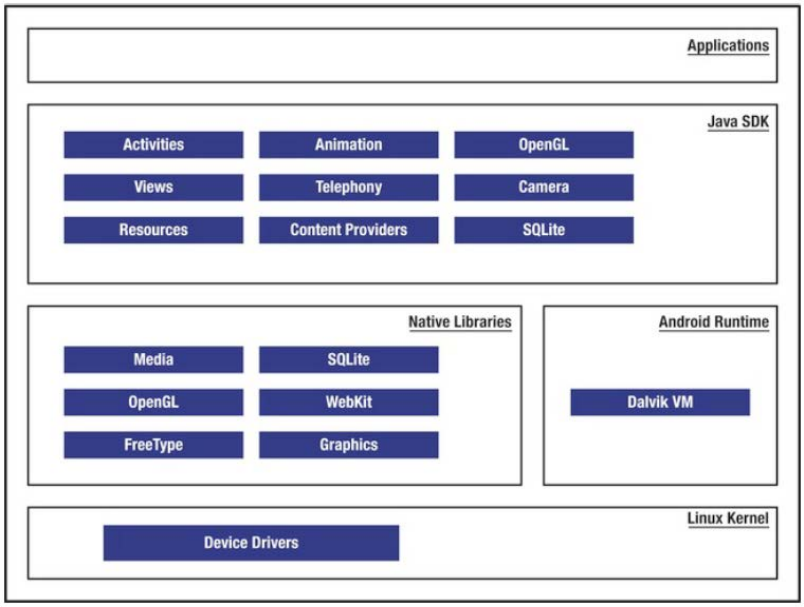
SQL pada *MySQL* adalah akronim dari *Structured Query Language*. Sebuah *database* MySQL berisi satu atau banyak *table*, setiap *table* berisi *record* (data) atau *row* (baris). Dalam baris ini terdapat berbagai *column* (kolom) atau *fields* yang berisi data.

2.7. Android

Android merupakan *Operationg System* (OS) mobile *open source* yang tumbuh di tengah OS lainnya yang berkembang dewasa ini. Android menawarkan sebuah lingkungan yang berbeda untuk pengembang. Android merangkul semua ide mengenai komputasi serbaguna untuk perangkat genggam. Android merupakan *Platform* yang lengkap dimana OS berbasis linux menangani pengaturan kerja perangkat, memory, dan proses. Sementara *Java Libraries* Android menanangani proses *telephony, video, speech, graphic, connectivity, UI programming,* dan beberapa aspek lain dari perangkat genggam tersebut. (Satya Komatineni dan Dave MacLean, 2012)

2.7.1. Arsitektur Android

Sistem operasi Android dibangun berdasarkan kernel Linux dan memiliki arsitektur seperti pada Gambar 2.2



Gambar 2.5 Arsitektur Android (Satya dan Dave, 2012)

Inti dari *Platform* Android adalah sebuah *Linux Kernel* yang bertanggung jawab atas *Driver* dari suatu perangkat Android, akses *Resource*, *Power Management*, dan tugas OS lainnya. Kernel juga bertindak sebagai lapisan antara hardware dan seluruh software.

Menuju ke tingkat selanjutnya, yaitu tempat di mana fitur-fitur Android berada, berisi kumpulan *libraries* dalam bahasa C/C++ . Biasanya para pembuat aplikasi mengakses libraries untuk menjalankan aplikasinya.

Selanjutnya adalah Android *Runtime*, berisi satu set *libraries* inti yang menyediakan sebagian besar fungsi yang tersedia di *libraries* inti dari bahasa pemrograman Java. Setiap aplikasi akan berjalan sebagai proses sendiri pada *Dalvik Virtual Machine* (DVM).

Pengembang Aplikasi memiliki akses penuh ke Android sama dengan aplikasi inti yang telah tersedia. Pengembang dapat dengan mudah mengakses informasi lokasi, mengatur alarm, menambah pemberitahuan ke status bar dan lain sebagainya. Arsitektur aplikasi ini dirancang untuk menyederhanakan penggunaan komponen jika ingin digunakan lagi, aplikasi apa pun dapat menggunakan kemampuan mereka sesuai batasan keamanan.

Aplication adalah lapisan paling atas, berisi serangkaian aplikasi yang terdapat pada perangkat mobile. Aplikasi-aplikasi ini ditulis dengan bahasa pemrograman Java. Kalender, kontak, SMS, Alarm, dan lainnya adalah merupakan beberapa contoh aplikasi inti yang ada pada Android. (Stephanus Hermawan S, 2011)

2.7.2. Komponen Dasar

Aplikasi Android ditulis dalam bahasa pemrograman Java, Java mengompilasi kode bersama dengan data *resources* dan file yang dibutuhkan oleh aplikasi di-*bundle* ke dalam paket Android, file arsip ditandai dengan '.apk'.

Komponen aplikasi pada Android terdiri dari 4 komponen utama, yaitu:

a. Activities

Activities merupakan potongan kode *executable* yang menyajikan UI secara visual dimulai oleh pengguna maupun sistem operasi dan berjalan selama diperlukan. Masing-masing *Activities* menunjukkan

satu layar untuk pengguna. *Activities* yang tidak aktif akan dimatikan oleh sistem operasi untuk menghemat memori.

b. Services

Services tidak memiliki visual *User Interface* (UI), melainkan berjalan di *Background* untuk waktu yang tidak terbatas. Contoh dari *Services* adalah MP3 player yang akan terus memainkan file MP3 sesuai urutan file, walaupun pengguna menggunakan aplikasi lain.

c. Broadcast Receiver

Broadcast Receiver merupakan komponen yang menerima dan bereaksi untuk menyiarkan *Notification*. Contoh dari *Notification* tersebut adalah seperti pemberitahuan bahwa daya baterai tinggal sedikit, file telah berhasil dipindahkan dan lain sebagainya.

d. Content Provider

Content Provider diciptakan untuk berbagi data dengan *Activities* lain atau *Services*. Sebuah *Content Provider* menggunakan antarmuka standar dalam bentuk URI untuk memenuhi permintaan data dari aplikasi lain.

2.8. Extensible Markup Language (XML)

XML adalah singkatan dari *Extensible markup language*, pada awal kemunculannya XML kurang populer, namun setelah beberapa tahun, XML menjadi pusat perhatian dan mulai merambah dunia *data exchange*.

Ada dua kegunaan utama dari XML: yang pertama adalah untuk merepresentasikan *low-level data*, contohnya *file* konfigurasi. Kegunaan kedua adalah untuk menambahkan *metadata* ke suatu dokumen; contohnya menambahkan *style bold* atau *italic* ke suatu baris dalam sebuah laporan. Mudahnya kegunaan yang kedua mirip dengan HTML. (Joe Fawcett, dkk., 2012)

2.9. Unified Modeling Language

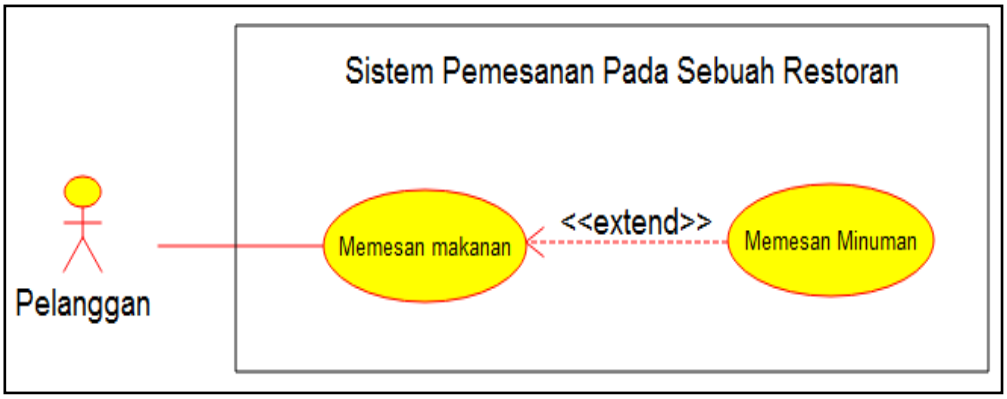
Unified Modeling Language adalah standar bahasa pemodelan untuk pengembangan perangkat lunak dan sistem. Sebuah Model adalah abstraksi dari hal yang sebenarnya. Ketika melakukan pemodelan, abstrak yang dibuat akan jauh dari bagian yang tidak relevan atau mungkin berpotensi menjadi hal yang membingungkan. Model adalah penyederhanaan dari sistem yang sebenarnya, sehingga memungkinan untuk dapat memahami, mengevaluasi, dan mencari celah dari sebuah desain dan sistem lebih cepat daripada menelusuri sistem yang sebenarnya.

Bahasa pemodelan dapat berupa *Pseudo-code*, *actual code*, gambar, diagram, atau mungkin berupa tulisan berupa deskripsi panjang; intinya apa saja yang dapat membantu untuk mendeskripsikan sistem yang akan dibuat.

2.9.1. Use Case

Use Case diagram adalah sebuah kejadian (atau situasi) dimana sistem akan memenuhi kebutuhan penggunanya. *Use case* adalah jantung dari model yang dibuat, dimana akan sangat mempengaruhi dan menuntun

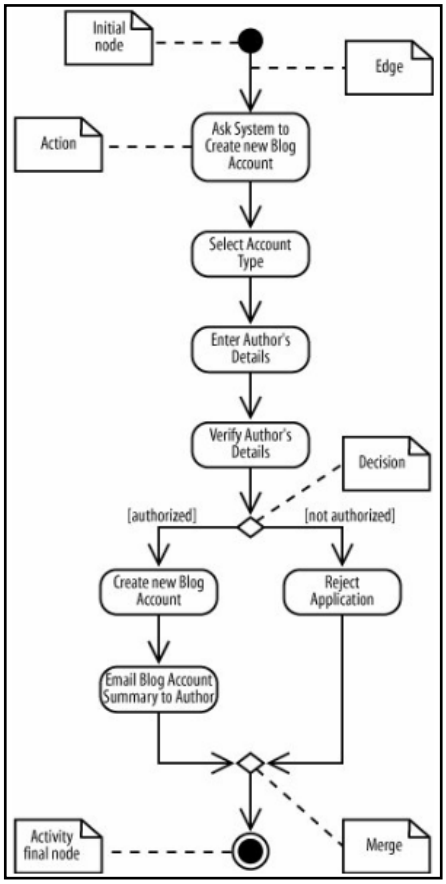
seluruh elemen lain saat pemodelan sistem. *Use case* adalah awalan yang sempurna untuk melakukan pengembangan, pemodelan, pengetesan, dan dokumentasi sitem berorientasi objek dari segala sisi. (Kim Hamilton dan Russell Miles, 2006)



Gambar 2.6 Contoh *Use Case diagram*

2.9.2. Activity Diagram

Activity diagram memungkinkan kita untuk menentukan bagaimana sistem akan akan menyelesaikan tujuannya. *Activity diagram* menampilkan *langkah-langkah tingkat tinggi* yang saling berhubungan untuk merepresentasikan sebuah proses yang berlangsung pada sistem. (Kim Hamilton dan Russell Miles, 2006)

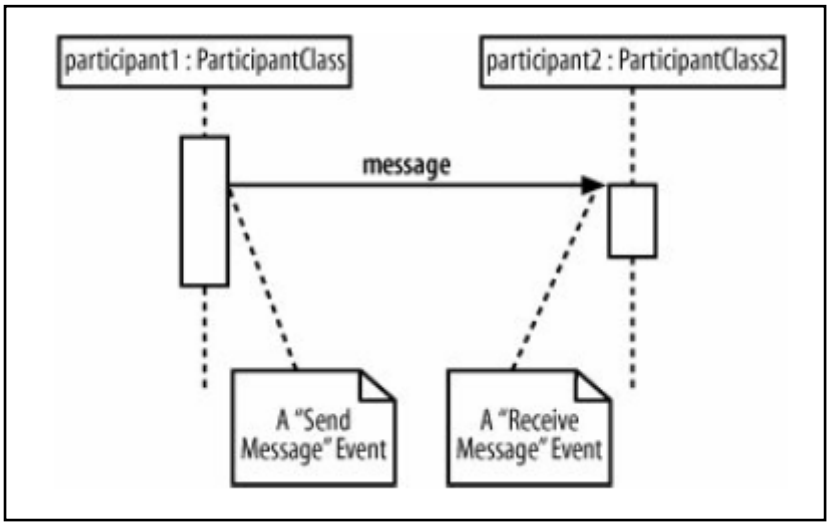


Gambar 2.7 Contoh *Activity diagram*

2.9.3. Sequence Diagram

Sequence diagram adalah salah satu anggota penting dalam kelompok diagram yang dikenal dengan nama *interaction diagram*. *Sequence diagram* berguna untuk memberi gambaran mengenai susunan dari interaksi antar bagian dari sebuah sistem. Dengan *sequence diagram*, maka akan dapat mendeskripsikan interaksi mana yang akan terpicu jika suatu perintah dijalankan. *Sequence diagram* dapat menampilkan banyak informasi lain tentang interaksi, tapi kegunaan utamanya adalah menampilkan secara sederhana dan mudah bagaimana sistem

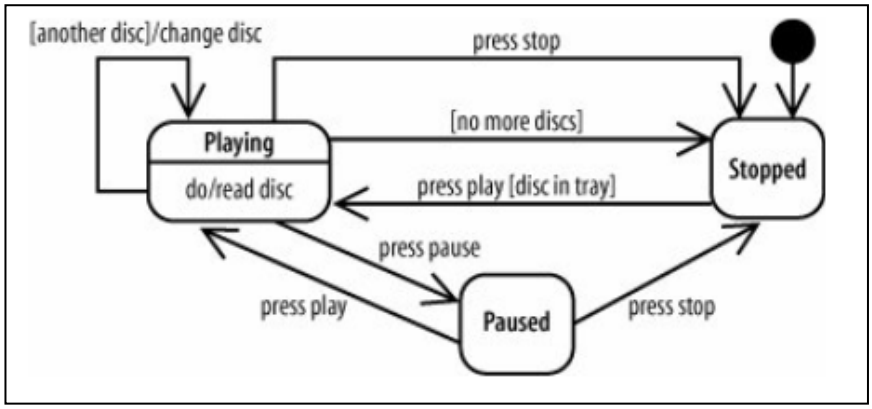
berkomunikasi secara tersusun sesuai urutan kejadian dalam sebuah interaksi. (Kim Hamilton dan Russell Miles, 2006)



Gambar 2.8 Contoh *Sequence diagram*

2.9.4. State Machine Diagram

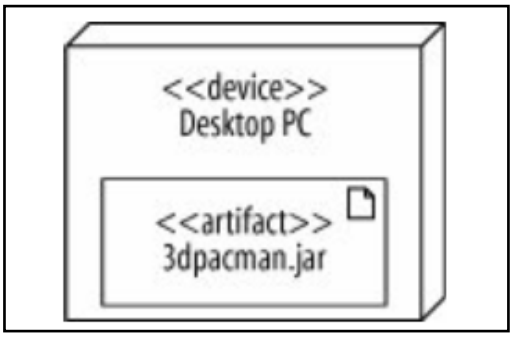
State machine diagram terdiri dari *states*, digambarkan dengan persegi dengan sisi bulat, dan *transistion* digambarkan dengan panah yang menyambungkan antar *states*. *Transition* merepresentasikan perubahan dari *states*, atau bagaimana untuk sampai ke suatu *state* dan kemudian berpindah ke *state* selanjutnya. Sebuah *state* akan aktif ketika masuk melaui sebuah *transition*, dan *state* akan menjadi tidak aktif saat keluar melalui *transition*. (Kim Hamilton dan Russell Miles, 2006)



Gambar 2.9 Contoh *State machine diagram*

2.9.5. Deployment Diagram

Deployment diagram menampilkan gambaran fisik dari sebuah sistem, membawa perangkat lunak ke dunia nyata dengan menampilkan bagaimana perangkat lunak dipasang di perangkat keras dan bagaimana tiap bagiannya berkomunikasi. *Deployment diagram* seharusnya menampilkan rincian dari sistem yang penting bagi pengguna. Jika merasa perlu untuk menampilkan perangkat keras, *firmware*, sistem operasi, *runtime environments*, atau mungkin *driver* perangkat dari sistem yang dibuat, maka komponen tersebut harus di masukkan ke dalam *deployment diagram.* (Kim Hamilton dan Russell Miles, 2006)



Gambar 2.10 Contoh *Deployment diagram*