

## BAB II

### LANDASAN TEORI

#### 2.1. Tinjauan Pustaka

Berikut beberapa jurnal terkait permasalahan yang diangkat dalam Tugas Akhir ini:

- 1) **Aplikasi Penerima Radio *Streaming Online* Pada *Smartphone* Berbasis Java** (Parsumo Raharjo dan Endah Tri Utami, 2012). Pada jurnal ini, menerangkan Tugas Akhir terkait perancangan sebuah aplikasi radio *streaming online* untuk *smartphone* khususnya *blackberry*. Tujuan dari pembuatan tugas akhir dalam jurnal tersebut adalah untuk membuat aplikasi radio *streaming online* pada *smartphone* dengan menggunakan bahasa pemrograman java sehingga aplikasi ini dapat terselesaikan dan berwujud aplikasi yang sesungguhnya. Pengembangan sistem aplikasi radio *streaming* ini dibangun berdasarkan metode *waterfall*. Pengujian sistem dilakukan dengan menginstal aplikasi ke dalam *smartphone blackberry* yang mendukung Java dan memiliki akses jaringan internet dengan kecepatan data tinggi (EDGE, 2G, 3G). Selain itu pengujian juga dilakukan untuk pencatatan waktu akses tiap operator dan operator XL memiliki waktu akses yang lebih cepat yaitu 5 detik dengan kecepatan rata-rata akses 300kbps. Pada pengujian tingkat persepsi kepuasan pengguna *smartphone* didapatkan hasil 75,6%. Dengan penerapan aplikasi ini, pengguna dapat mendengarkan siaran radio *streaming* secara *online*.
  
- 2) **Aplikasi Radio *Streaming* dengan Basis *Client* Android di Radio Dista FM IAIN Surakarta** (Tri Susilo, 2013). Karya Ilmiah tersebut menjelaskan, Dista FM merupakan radio komunitas milik IAIN Surakarta, jangkauan pancar siaran hanya sekitar 2,5 Km. Penelitian ini memiliki tujuan untuk mengatasi kekurangan tersebut dengan membangun sebuah layanan radio *streaming* dengan basis *client*

Android pada radio tersebut. Penelitian ini menggunakan metode studi literatur, eksperimen, *coding*, observasi, dan *sampling*. Tujuan dari penelitian ini adalah untuk membangun sebuah layanan *streaming* radio menggunakan *software winamp 5.58* dan *shoutcast DSP 2.2.3* serta membuat aplikasi Android untuk mendengarkan siaran radio *streaming* Dista FM menggunakan Eclipse 3.8, ADT 21.1.0.2013, SDK 4.2.2. Dan desain gambar menggunakan Adobe Photoshop CS 5. Dengan adanya layanan *streaming* radio Dista FM dengan basis *client* android ini, jangkauan siaran Dista FM sudah tidak lagi menjadi kendala bagi radio tersebut, terlebih dengan adanya aplikasi android *streaming* Dista FM memungkinkan pengguna *smartphone* android dapat mendengarkan siaran radio melalui perangkat android mereka. Dengan *encoder* dari *streaming* Dista FM sebesar 40 kbs, maka pendengar kecepatan *download* berkisar 40 – 60 kbps sudah bisa mendengarkan siaran radio *streaming* Dista FM. Selisih antara siaran pemancar Dista FM dengan *streaming* Dista FM sebesar 5 – 10 detik dengan kondisi sinyal *client* stabil.

- 3) **Pembangunan Aplikasi *Streaming* Radio Berbasis Windows Phone 8** (Andreas Chandra Yogyaswara Budiono, 2013). Tugas Akhir tersebut menjelaskan pembuatan aplikasi klien *streaming* radio berbasis Windows Phone 8 dengan *tool* pembangunan *Microsoft Visual Studio 2012* dengan bahasa pemrograman C# dan jaringan yang terhubung kepada sebuah penyedia radio *online*, JOGJASTREAMERS. Penelitian dilakukan dengan melakukan analisa kebutuhan untuk suatu aplikasi *streaming* agar radio-radio dalam jaringan JOGJASTREAMERS dapat diakses menggunakan Windows Phone 8. Aplikasi yang dibangun ini juga dapat dihubungkan dengan jejaring sosial *Facebook* dan *Twitter* untuk menyampaikan informasi tentang pengguna yang sedang mendengarkan radio. Dari survei yang dilakukan terhadap 31 responden yang terdiri dari 22 pria dan 9 wanita, dengan rentang usia 17 sampai 36 tahun, dengan 18 orang mempunyai profesi sebagai mahasiswa dan 13 orang karyawan, 63,7% menyatakan aplikasi mempunyai tampilan yang bagus, 67,2% menyatakan

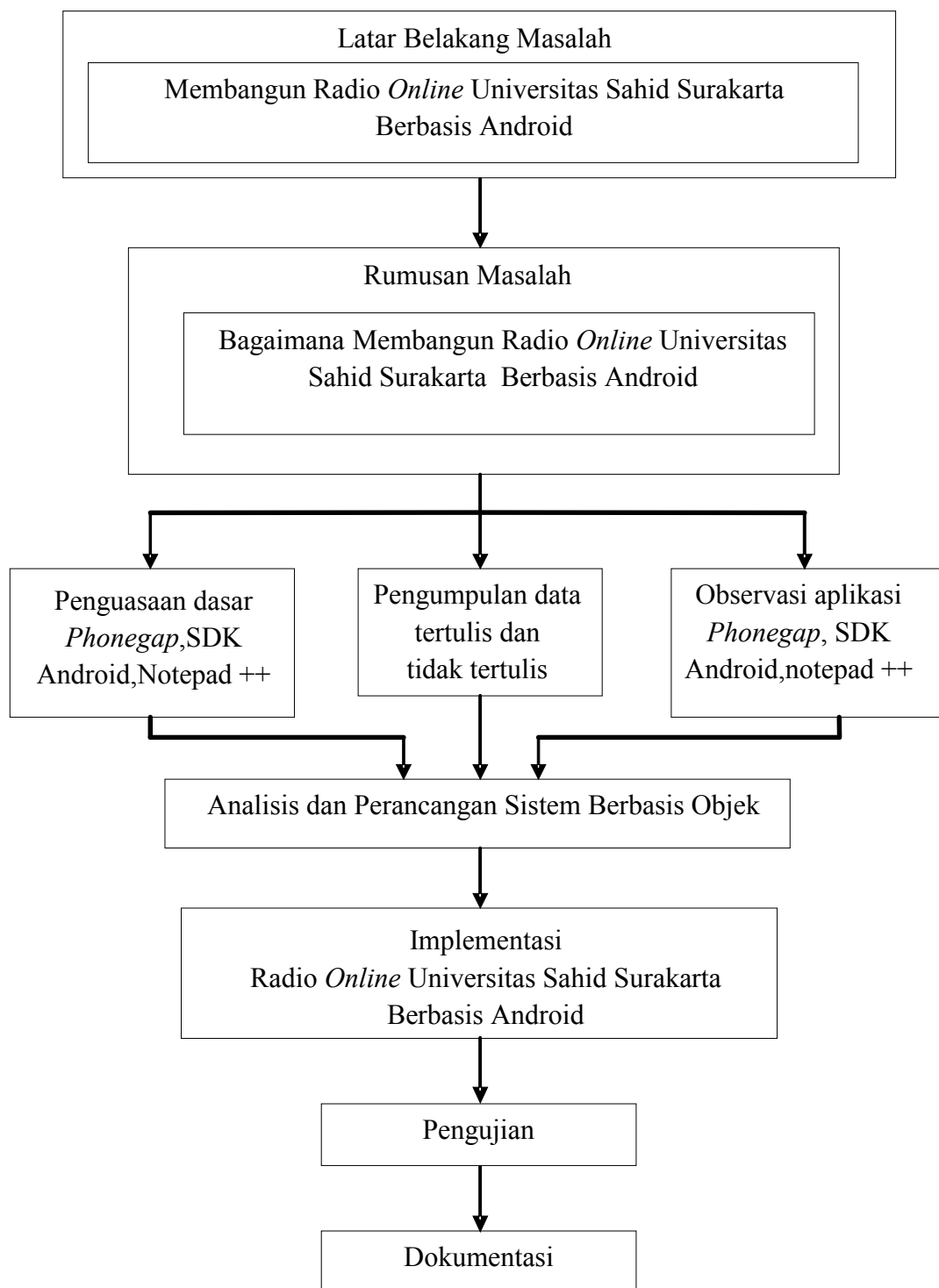
fungsi-fungsi yang dimiliki aplikasi mudah untuk dijalankan, dan 90,32% menyatakan aplikasi sudah bagus bila dibandingkan dengan aplikasi yang sudah ada sebelumnya (website JOGJASTREAMERS, aplikasi JOGJASTREAMERS untuk Android, Blackberry, dan iOS).

- 4) **Perancangan dan Implementasi Radio *Streaming* Berbasis Android** (Irfan Sa'di, 2014). Jurnal tersebut menjelaskan aplikasi radio *streaming* berbasis Android yang dibuat merupakan sebuah aplikasi yang memutar radio-radio yang tersedia melalui *internet*. Aplikasi ini juga menyediakan beberapa fitur antara lain daftar radio, penambahan daftar radio, penghapusan daftar radio, pengeditan data radio, halaman berita, deretan tangga lagu dan fitur pesan singkat. Pada aplikasi ini didapat kesimpulan bahwa untuk mendapatkan kualitas suara yang jernih dibutuhkan akses *internet* yang stabil.

Berdasar empat tinjauan pustaka tersebut dapat dilihat bahwa teknologi radio *streaming* di *internet* sudah sangat berkembang dan memberikan solusi bagi keterbatasan jarak siar radius konvensional. *Platform* dan alat yang digunakan dalam mengembangkan aplikasi ini juga sangat beragam tergantung kondisi studi kasus masing-masing. Sistem operasi Android juga menjadi pilihan dalam beberapa pengembangan. Perbedaan dalam Tugas akhir ini bila dibandingkan dengan tinjauan pustaka tersebut ialah fokus pada implementasi teknologi radio *streaming* di radio Universitas Sahid Surakarta, sebagai solusi masalah keterbatasan jangkauan siar radio ini.

## 2.2. Kerangka Pemikiran

Pada sub bab ini akan di jelaskan gambaran kerangka pemikiran dalam pembuatan Tugas Akhir ini yang ditunjukkan pada gambar 2.1. Dimulai dari latar belakang, rumusan masalah, pengambilan data, analisis perancangan sistem, implementasi, pengujian, sampai dengan dokumentasi.



Gambar 2.1 Diagram Kerangka Pemikiran

### 2.3. Teknologi Radio *Streaming*

*Streaming* adalah proses mentransfer data melalui *channel* ke tujuan dengan karakteristik *real time*, di mana ia diterjemahkan dan dikonsumsi melalui pengguna / perangkat secara *real time*, yaitu sebagai data yang disampaikan dengan cepat. Ini berbeda dari proses *non-streaming* yang tidak memerlukan *channel* data yang akan diunduh sepenuhnya sebelum dapat dilihat atau digunakan. *Streaming is never a property of the data that is being delivered*, ini berarti bahwa secara teknis sebagian besar media dapat dialirkan (Arun Kumar B. R, dkk, 2008). Berikut hal-hal terkait teknologi ini.

Ajinkya Patil, dkk (2011) menjelaskan dalam jurnalnya, secara konten audio dapat disampaikan melalui jaringan dalam dua cara. Pertama *file audio* dapat diunduh dan kemudian diputar dari *hard disk* lokal dari klien. Kedua konten *audio* dapat dialirkan dari *server* ke klien yang *decode* paket yang diterima secara *real time*.

Keuntungan dan kerugian dari kedua metode ialah, sebagai keuntungan unduh data bekerja dengan data *rate* dan memungkinkan salah satu kualitas audio terbaik. *File* tersebut ditransmisikan bebas dari kesalahan sehingga tidak ada penurunan kualitas terjadi selama transmisi. Mengunduh data menyimpan permanen data, sehingga data kembali sesuai permintaan. Kekurangan unduh data, pengambilan data sangat panjang dengan audio yang kualitas tinggi disediakan melalui jaringan *bandwidth* rendah. Misalnya, klip musik lima menit panjang dikodekan dengan 128 kbps dapat berlangsung lebih dari 20 menit untuk mengunduh lebih dari akses *internet* pribadi dengan *bandwidth* 20 kbps, untuk mengurangi waktu *download*, kualitas *audio* harus dikurangi. Pengguna harus selesai mengunduh *file* sebelum ia dapat mendengarkan setiap bagian dari itu. Dia tidak dapat melihat pratinjau *file* untuk memutuskan apakah ia tertarik pada konten. Tidak ada kemungkinan untuk interaksi penyiar dan pendengar.

Keuntungan *streaming*, pengguna dapat mendengarkan konten segera setelah ia memilih *channel*. Pengguna tidak perlu menunggu selesai mengunduh untuk mendengarkan siaran. Hal ini dimungkinkan untuk menyediakan layanan "*live*"

dengan pengkodean sinyal *audio* secara *real time* dan mengirim dihasilkan aliran data *audio* langsung ke klien.

Kekurangan *streaming*, inti dari *streaming* ialah *real time*, maka *channel* transmisi harus menyediakan *bandwidth* penuh selama periode transmisi secara keseluruhan. Kualitas *audio* sangat membebankan pada *bandwidth*. Aliran yang ditransmisikan sangat sensitif terhadap beban yang dapat menyebabkan yang hilang atau tertunda paket data jaringan. Hal ini menyebabkan putusnya *output audio* pada klien. Proses kontrol *bandwidth* memerlukan tambahan perangkat lunak di *server*.

Implementasi dilapangan, jika klien pada jaringan memiliki memori terbatas, misalnya perangkat *mobile*, maka salah satu memilih dari dua metode di atas membuat banyak perbedaan sehubungan dengan biaya dan pengalaman pengguna. Pengguna perangkat *mobile* apabila menggunakan metode mengunduh akan sangat terbebani. *Streaming* di sisi lain, membentuk solusi yang lebih baik. Pertimbangan keuntungan utama dari *download*, mungkin tingginya kualitas *audio* bahkan melalui jaringan *bandwidth* rendah. Kekurangan dari *streaming*, bagaimanapun, dapat dikurangi dengan dinamis menyesuaikan jumlah data yang dikirimkan ke kapasitas sebenarnya dari koneksi jaringan. Untuk melakukan hal ini, data *audio* harus baik disimpan dalam format yang memungkinkan menjatuhkan bagian dari data yang mengakibatkan data tidak lengkap atau mereka harus disimpan dalam format yang berbeda bahwa *server* dapat memilih format yang paling cocok untuk *bandwidth* diberikan koneksi. Sejauh ini aplikasi *audio* pada ponsel menggunakan metode mengunduh bukan merupakan pilihan yang layak karena ponsel tidak memiliki memori yang cukup untuk mengunduh data *audio* dan kemudian memainkannya. Oleh karena itu *streaming* satu-satunya pilihan untuk ponsel.

#### **2.4. Sistem Operasi Android**

Suhas Holla dan Mahima M Katti (2012) dalam jurnal mereka menjelaskan, Android ialah generasi baru sistem operasi *mobile* yang berjalan pada Kernel Linux. Pengembangan aplikasi *mobile* Android berdasarkan kode bahasa Java. Kode ini

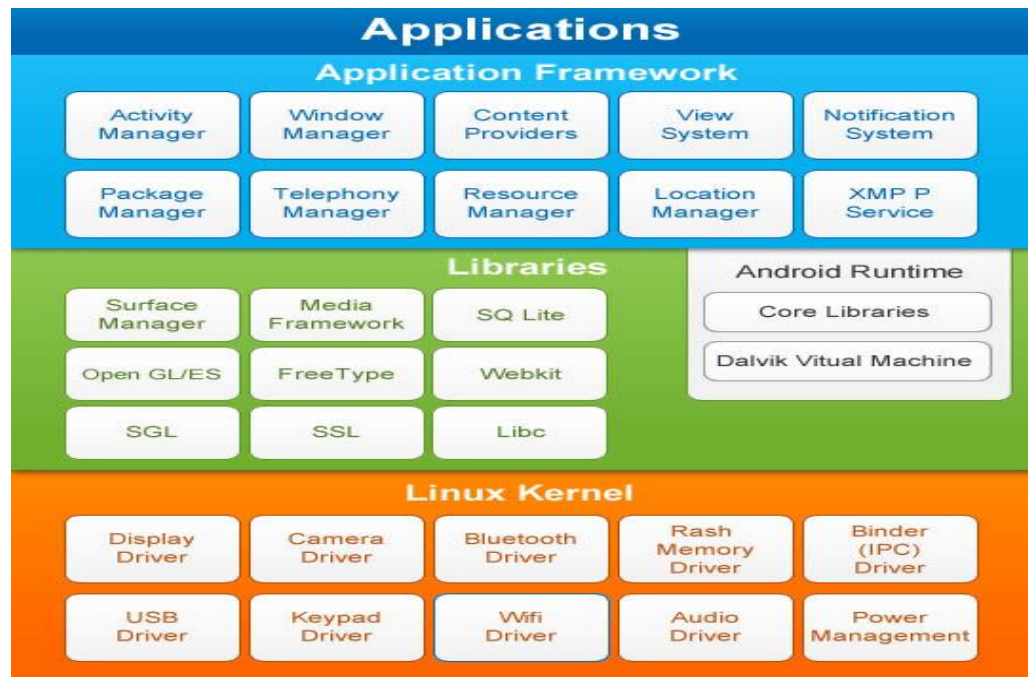
dapat mengontrol perangkat *mobile* melalui *library Java Google-enabled*. Ini adalah *platform* penting untuk mengembangkan aplikasi *mobile* menggunakan *software stack* yang disediakan di Android SDK Google. OS Android menyediakan lingkungan yang fleksibel untuk pengembangan aplikasi *mobile*. Pengembang tidak bisa hanya menggunakan *library Java* tetapi juga memungkinkan untuk menggunakan IDE Java normal. Para pengembang *software* di *Mobile Development India* memiliki keahlian dalam mengembangkan aplikasi berbasis *library Jawa* dan alat-alat penting lainnya. *Android Mobile Application* dapat digunakan untuk membuat aplikasi pihak ketiga yang inovatif dan dinamis. Pengembang ponsel India telah bekerja secara luas pada proyek-proyek mulai dari *software game, organizers, media player, picture editor* untuk digunakan perangkat lain.

#### **2.4.1. Dasar Aplikasi Android**

Aplikasi Android ditulis dalam bahasa pemrograman Java. Namun, penting untuk diingat bahwa ini tidak dieksekusi menggunakan standar Java *Virtual Machine* (JVM). Sebaliknya, Google telah menciptakan kustom VM disebut Dalvik yang bertanggung jawab untuk mengubah dan mengeksekusi kode *byte Java*. Semua kelas Java kustom harus dikonversi (ini dilakukan secara otomatis tetapi juga dapat dilakukan secara manual) menjadi instruksi Dalvik kompatibel ditetapkan sebelum dieksekusi ke sistem operasi Android. Dalvik VM mengambil *file* kelas Java yang dihasilkan dan menggabungkan mereka ke dalam satu atau lebih *Dalvik Executable* (Dex) *file*. Menggunakan duplikat informasi dari beberapa file kelas, efektif mengurangi kebutuhan ruang (terkompresi) oleh setengah dari file *.jar* tradisional. Dalvik diciptakan untuk mendukung sifat sistem operasi mobile ringan ini dibutuhkan karena kemampuan perangkat keras terbatas dibandingkan dengan desktop konvensional atau laptop (Suhas Holla dan Mahima M Katti, 2012).

### 2.4.2. Arsitektur Android

Android adalah *software stack* untuk perangkat mobile yang mencakup sistem operasi, *middleware* dan aplikasi kunci. Android SDK menyediakan alat dan API diperlukan untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. Android berbasis Linux versi 2.6. Layanan sistem seperti keamanan, manajemen memori, manajemen proses dikendalikan oleh Linux (Suhas Holla dan Mahima M Katti, 2012). Gambar 2.2 menunjukkan arsitektur android.



Gambar 2.2 Arsitektur Android.

### 2.4.3. Mengembangkan Aplikasi Android

Android SDK menyediakan serangkaian *Application Programming Interfaces* (API) yang bersifat modern dan kuat. layanan sistem inti Android yang terkena dan dapat diakses oleh semua aplikasi. Ketika diberikan izin yang sesuai, aplikasi Android dapat berbagi data di antara satu sama lain dan berbagi akses sumber daya



pada sistem aman. Aplikasi Android ditulis dalam bahasa pemrograman Java (Suhas Holla dan Mahima M Katti, 2012).

#### 2.4.4. Kerangka Aplikasi Android

Dengan menyediakan sebuah *platform* pengembangan terbuka, Android menawarkan pengembang kemampuan untuk membangun aplikasi yang sangat kaya dan inovatif. Pengembang bebas untuk mengambil keuntungan dari perangkat keras, informasi lokasi akses, *background services*, set *alarm*, tambahkan pemberitahuan ke status bar, dan masih banyak lagi.

Pengembang memiliki akses penuh ke API kerangka yang sama digunakan oleh aplikasi inti. Arsitektur aplikasi dirancang untuk menyederhanakan penggunaan kembali komponen. Aplikasi apapun dapat mempublikasikan aplikasi lainnya, maka dapat menggunakan kemampuannya (tunduk pada batasan keamanan *framework*). Mekanisme yang sama memungkinkan komponen untuk diganti oleh pengguna. Berikut merupakan kerangka dasar aplikasi Android:

- 1) Kerangka *views* yang dapat digunakan untuk membangun aplikasi, termasuk daftar, *grids*, kotak *teks*, tombol, dan bahkan sebuah *browser web embeddable*.
- 2) *Content Providers* yang memungkinkan aplikasi untuk mengakses data dari aplikasi lain (seperti Kontak), atau untuk berbagi data mereka sendiri.
- 3) *Resource Manager*, menyediakan akses ke sumber daya *non-kode* seperti string lokal, grafis, dan *file layout*.
- 4) *Notification Manager* yang memungkinkan semua aplikasi untuk menampilkan peringatan kebiasaan di status bar.
- 5) *Activity Manager* yang mengelola siklus hidup aplikasi dan menyediakan navigasi umum *backstack* (Suhas Holla dan Mahima M Katti, 2012).

#### **2.4.5. Android Runtime**

Android termasuk satu set *library* inti yang menyediakan sebagian besar fungsi yang tersedia di *library* inti dari bahasa pemrograman Java. Setiap aplikasi Android berjalan dalam prosesnya sendiri dan dengan contoh sendiri dari VM Dalvik. Dalvik telah ditulis sehingga perangkat dapat menjalankan beberapa VM secara efisien. VM Dalvik *file* dieksekusi dalam Dalvik executable (Dex) format yang dioptimalkan untuk minimal jejak memori. VM menjalankan kelas dikompilasi oleh *compiler* bahasa Java yang telah diubah menjadi format Dext. VM Dalvik bergantung pada kernel Linux untuk fungsionalitas yang mendasari seperti *threading* dan manajemen memori tingkat rendah (Suhas Holla dan Mahima M Katti, 2012).

### **2.5. Perancangan Sistem dengan UML**

*Unified Modelling Language* (UML) adalah suatu alat untuk memvisualisasikan dan mendokumentasikan hasil analisa dan desain yang berisi sintak dalam memodelkan sistem secara visual. Juga merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem *software* yang terkait dengan objek. Menganalisa dan mendesain atau memodelkan suatu sistem karena UML memiliki seperangkat aturan dan notasi dalam bentuk grafis yang cukup spesifik. Komponen atau notasi UML diturunkan dari 3 (tiga) notasi yang telah ada sebelumnya yaitu Grady Booch OOD (*Object-Oriented Design*), Jim Rumbaugh OMT (*Object Modelling Technique*), dan Ivar Jacobson OOSE (*Object-Oriented Software Engineering*). Pada UML versi 2 terdiri atas tiga kategori dan memiliki 13 jenis diagram (Haviluddin, 2011).

#### **2.5.1. Struktur Diagram**

Haviluddin (2011) menjelaskan menggambarkan elemen dari spesifikasi dimulai dengan kelas, obyek, dan hubungan mereka, dan beralih ke dokumen arsitektur logis dari suatu sistem. Struktur diagram dalam UML terdiri atas :


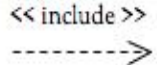
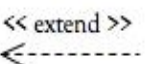



- 1) *Class diagram* menggambarkan struktur statis dari kelas dalam sistem anda dan menggambarkan atribut, operasi dan hubungan antara kelas. *Class diagram* membantu dalam memvisualisasikan struktur kelas-kelas dari suatu sistem dan merupakan tipe diagram yang paling banyak dipakai. Selama tahap desain, *class diagram* berperan dalam menangkap struktur dari semua kelas yang membentuk arsitektur sistem yang dibuat. *Class* memiliki tiga area pokok yaitu: Nama (dan *stereotype*), Atribut, dan Metoda.
- 2) *Object diagram* menggambarkan kejelasan kelas dan warisan dan kadang - kadang diambil ketika merencanakan kelas, atau untuk membantu pemangku kepentingan *non-program* yang mungkin menemukan *diagram* kelas terlalu abstrak..
- 3) *Component diagram* menggambarkan struktur fisik dari kode, pemetaan pandangan logis dari kelas proyek untuk kode aktual di mana logika ini dilaksanakan.
- 4) *Deployment diagram (Collaboration diagram in version 1.x)* merupakan gambaran dari arsitektur fisik perangkat lunak, perangkat keras, dan artefak dari sistem. *Deployment diagram* dapat dianggap sebagai ujung spektrum dari kasus penggunaan, menggambarkan bentuk fisik dari sistem yang bertentangan dengan gambar konseptual dari pengguna dan perangkat berinteraksi dengan sistem.
- 5) *Composite structure diagram* Sebuah diagram struktur komposit mirip dengan diagram kelas, tetapi menggambarkan bagian individu, bukan seluruh kelas. Kita dapat menambahkan konektor untuk menghubungkan dua atau lebih bagian dalam atau ketergantungan hubungan asosiasi.
- 6) *Package diagram*, biasanya digunakan untuk menggambarkan tingkat organisasi yang tinggi dari suatu proyek *software*. Dengan kata lain untuk menghasilkan diagram ketergantungan paket untuk setiap paket dalam Pohon Model.

### **2.5.2. Behavior Diagram**

Menggambarkan ciri-ciri *behavior/metode/* fungsi dari sebuah sistem atau business process. *Behavior diagram* dalam UML terdiri atas :






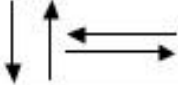
1) *Use Case diagram* ialah diagram yang menggambarkan *actor*, *use case* dan relasinya sebagai suatu urutan tindakan yang memberikan nilai terukur untuk aktor. Sebuah *use case* digambarkan sebagai *elips horizontal* dalam suatu diagram UML *use case*. *Use case* memiliki dua istilah yaitu: *System use case* (interaksi dengan sistem), dan *Business use case* (interaksi bisnis dengan konsumen atau kejadian nyata). Bagian-bagian dari *Use Case diagram* ditunjukkan pada Tabel 2.1.

Tabel 2.1 Bagian-bagian *use case* diagram.

NO	GAMBAR	NAMA	KETERANGAN
1		Actor	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan <i>use case</i> .
2		Include	Menspesifikasikan bahwa <i>use case</i> sumber secara eksplisit.
3		Extend	Menspesifikasikan bahwa <i>use case</i> target memperluas perilaku dari <i>use case</i> sumber pada suatu titik yang diberikan.
4		Association	Apa yang menghubungkan antara objek satu dengan objek lainnya.
5		System	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
6		Use Case	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor

2) *Activity diagram* menggambarkan aktifitas-aktifitas, obyek, *state*, *transisi state* dan *event*. Dengan kata lain kegiatan diagram alur kerja menggambarkan perilaku sistem untuk aktivitas. Bagian-bagian *Activity diagram* ditunjukna pada Tabel 2.2.

Tabel 2.2 Bagian-bagian *Activity* diagram.

NO	GAMBAR	NAMA	KETERANGAN
1		Activity	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		Action	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		Initial Node	Bagaimana objek dibentuk atau diawali.
4		Activity Final Node	Bagaimana objek dibentuk dan diakhiri
5		Decision	Digunakan untuk menggambarkan suatu keputusan / tindakan yang harus diambil pada kondisi tertentu
6		Line Connector	Digunakan untuk menghubungkan satu simbol dengan simbol lainnya

3) *State Machine* diagram (*State chart* diagram in version 1.x) Menggambarkan *state*, *transisi state* dan *event*.

### 2.5.3. *Interaction diagram*

Bagian dari *behavior* diagram yang menggambarkan interaksi obyek. *Interaction* diagram dalam UML terdiri atas :

- 1) *Communication diagram* serupa dengan *sequence* diagram, tetapi diagram komunikasi juga digunakan untuk memodelkan perilaku dinamis dari *use case*. Bila dibandingkan dengan *Sequence diagram*, diagram komunikasi lebih terfokus pada menampilkan kolaborasi benda daripada urutan waktu.
- 2) *Interaction Overview* diagram *Interaksi overview* diagram berfokus pada gambaran aliran kendali interaksi dimana *node* adalah interaksi atau kejadian interaksi.

- 3) *Sequence diagram* menjelaskan interaksi objek yang disusun berdasarkan urutan waktu. Secara mudahnya *sequence diagram* adalah gambaran tahap demi tahap, termasuk kronologi (urutan) perubahan secara logis yang seharusnya dilakukan untuk menghasilkan sesuatu sesuai dengan *use case diagram*.
- 4) *Timing diagram* di UML didasarkan pada diagram waktu *hardware* awalnya dikembangkan oleh para insinyur listrik.

## 2.6. Alat Pengembangan

- 1) SDK (*Software Development Kit*) menurut Nasruddin Safaat H (2012), adalah alat bantu dan API dalam mengembangkan aplikasi dalam *platform* android menggunakan bahasa pemrograman Java.
- 2) Notepad ++ adalah editor kode sumber yang berjalan di *Windows*. Notepad++ menggunakan komponen *Scintilla* untuk dapat menampilkan dan menyuntingan teks dan berkas kode sumber berbagai bahasa pemrograman. Notepad++ mendukung banyak bahasa pemrograman (Notepad ++, 2016).
- 3) *Shoutcast* adalah lebih dari sekedar perangkat lunak, ini adalah paket lengkap dari produk tanpa biaya radio daya internet ke masa depan. *Shoutcast* memungkinkan pengguna mengirimkan audio untuk pendengar di seluruh dunia (*Shoutcast*, 2015).
- 4) *PhoneGap* Membangun aplikasi untuk setiap *platform* iPhone, Android, *Windows* dan lain-lain membutuhkan kerangka kerja dan bahasa yang berbeda. *PhoneGap* memecahkan ini dengan menggunakan standar berbasis teknologi web untuk menjembatani aplikasi *web* dan perangkat *mobile*. *PhoneGap* aplikasi yang memenuhi persyaratan standar untuk bekerja dengan *browser* yang berevolusi. *PhoneGap* telah *download* jutaan kali dan sedang digunakan oleh ratusan ribu pengembang. Ribuan aplikasi yang dibangun menggunakan *PhoneGap* yang tersedia di toko aplikasi *mobile* dan direktori. Kode *PhoneGap* disumbangkan untuk Apache *Software Foundation* (ASF) dengan nama *Apache Cordova* dan lulus ke tingkat atas status proyek pada bulan Oktober 2012. Melalui ASF, pengembangan *PhoneGap* masa depan akan memastikan pengelolaan terbuka

proyek. Ini akan selalu tetap *free* dan *open source* di bawah lisensi *Apache*, Versi 2.0 (Adobe System Inc, 2016).

- 5) *Apache Cordova* adalah kerangka pengembangan *mobile open source*. Hal ini memungkinkan untuk menggunakan teknologi *web* standar seperti HTML5, CSS3, dan JavaScript untuk pengembangan lintas-*platform*, menghindari bahasa pengembangan asli masing-masing *platform mobile*. Aplikasi dieksekusi dalam kerangka yang ditargetkan setiap *platform*, dan bergantung pada *standar-compliant API binding* untuk mengakses sensor, data, dan status jaringan masing-masing perangkat (*The Apache Software Foundation*, 2015).
- 6) *Ionic* adalah *Powerfull SDK HTML5* yang membantu membangun aplikasi *mobile* menggunakan teknologi *web* seperti HTML, CSS, dan Javascript. *Ionic* difokuskan terutama pada tampilan dan nuansa, dan interaksi UI dari aplikasi Anda. *Ionic* dibangun dengan *PhoneGap* dan *Javascript*. *Ionic* hanya cocok dengan proyek-proyek dalam rangka untuk menyederhanakan aplikasi. *Ionic* saat ini membutuhkan *AngularJS* untuk bekerja pada potensi penuh (*Ionic Framework*, 2015).

## 2.7. Pengujian Sistem

### 2.7.1. WebQual

Metode *Webqual* merupakan salah satu metode atau teknik pengukuran kualitas *website* berdasarkan persepsi pengguna akhir. Metode ini merupakan pengembangan dari SERQUAL (Zeithaml, 1990 di dalam Stuart J. Barnes dan Richard Vidgen, 2003) yang banyak digunakan sebelumnya pada pengukuran kualitas jasa. Instrument penelitian pada WebQual tersebut dikembangkan dengan metode *Quality Function Development* (QFD), yang bermakna terstruktur dan proses disiplin yang memberikan makna untuk mengidentifikasi dan membawa suara *customer* melalui tiap-tiap tahap dari produk dan atau pengembangan pelayanan dan penerapannya (Slabey, 1990 di dalam Stuart J. Barnes dan Richard Vidgen, 2003).

WebQual sudah mulai dikembangkan sejak tahun 1998 dan telah mengalami beberapa interaksi dalam penyusunan dimensi dan 23 butir pertanyaan (di dalam Stuart J. Barnes dan Richard Vidgen, 2003). WebQual 4.0 disusun berdasarkan penelitian pada tiga area yaitu;

- 1) Kualitas informasi dari penelitian sistem informasi (*Information Quality*)
- 2) *Information Quality* adalah mutu dari isi yang terdapat pada *site*, pantas tidaknya informasi untuk tujuan pengguna seperti akurasi, format dan keterkaitannya. (Stuart J. Barnes dan Richard Vidgen, 2003)
- 3) Interaksi dan kualitas layanan dari penelitian kualitas sistem informasi (*Service Interaction Quality*).
- 4) *Service Interaction Quality* adalah mutu dari interaksi pelayanan yang dialami oleh pengguna ketika mereka menyelidiki kedalam *site* lebih dalam, yang terwujud dengan kepercayaan dan empati, sebagai contoh isu dari keamanan transaksi dan informasi, pengantaran produk, personalisasi dan komunikasi dengan pemilik *site*. (Stuart J. Barnes dan Richard Vidgen, 2003)
- 5) *Usability* dari *human computer interaction*.
- 6) *Usability* adalah mutu yang berhubungan dengan rancangan *site*, sebagai contoh penampilan, kemudahan penggunaan, navigasi dan gambaran yang disampaikan kepada pengguna. (Stuart J. Barnes dan Richard Vidgen, 2003)