

BAB 2

TINJAUAN PUSTAKA

2.1 Landasan Teori

Landasan teori merupakan penjelasan berbagai konsep dasar dan teori yang berkaitan dengan pembangunan aplikasi chatbot menggunakan *dialogflow API* untuk jadwal misa di gereja Katolik berbasis Adroid dengan *Natural Language Processing*.

2.2 Bahasa Alami (*Natural Language*)

Bahasa alami (*natural language*) adalah suatu bentuk *representasi* dari suatu pesan yang ingin dikomunikasikan antar manusia. Bentuk utama respresntasinya adalah berupa suara/ucapan (*spoken language*), tetapi sering pula dinyatakan dalam bentuk tulisan.

Bahasa dapat dibedakan menjadi Bahasa Alami dan Bahasa Buatan. Bahasa alami merupakan bahasa yang biasa digunakan untuk berkomunikasi antar manusia, misalnya bahasa Indonesia, Sunda, Jawa, Melayu, Inggris, China, Jepang dan lain-lain.

Bahasa buatan adalah bahasa yang dibuat secara khusus untuk memenuhi kebutuhan tertentu, misalnya bahasa pemodelan atau bahasa pemrograman komputer.

Chomsky adalah orang yang pertama kali merepresentasikan bahasa sebagai rangkaian simbol. *Chomsky* berhasil memperlihatkan bahwa bahasa apapun dapat direpresentasikan dengan suatu cara yang *universal*. Pemikiran *Chomsky* yang merepresentasikan bahasa sebagai kumpulan simbol-simbol tersebut telah membuka peluang untuk melakukan pemrosesan bahasa secara simbolik dengan teknologi komputer, sehingga melahirkan bidang ilmu *Natural Language Processing (NLP)*.

Linguistik adalah bidang ilmu yang khusus mengkaji tentang bagaimana bahasa distrukturkan dan digunakan. Bidang ilmu ini salah satunya membahas tentang perbendaharaan kata dan *leksikon*.

Perbendaharaan kata adalah sekumpulan kata-kata dan *frase-frase* yang digunakan dalam bahasa tertentu. Sebagai bagian dari pengkajian bahasa, *linguis* mendefinisikan semua kata-kata dan *frase-frase* yang digunakan secara umum demikian mengorganisasikannya ke dalam sebuah *leksikon*. *Leksikon* adalah kamus yang mendaftarkan kata-kata bahasa itu secara alfabet [2].

2.2.1 Pengelolaan Bahasa Alami (*Natural Language Processing*)

Pengolahan bahasa alami (*NLP*) adalah bidang ilmu komputer dan *linguistik* berkaitan dengan interaksi antara komputer dan manusia. Seluler generasi bahasa sistem komputer yang mengubah informasi dari *database* ke dalam bahasa manusia yang dapat dibaca.

Natural Language Processing, biasanya disingkat dengan *NLP*, mencoba membuat komputer mampu memahami suatu perintah yang dituliskan dalam bentuk bahasa sehari-hari dan diharapkan komputer juga merespon dalam bahasa yang mirip dengan bahasa natural. Setelah komputer biasa memahami perintah dalam bahasa natural, maka diharapkan sistem komputer juga dapat memberikan respon dalam bahasa natural pula.

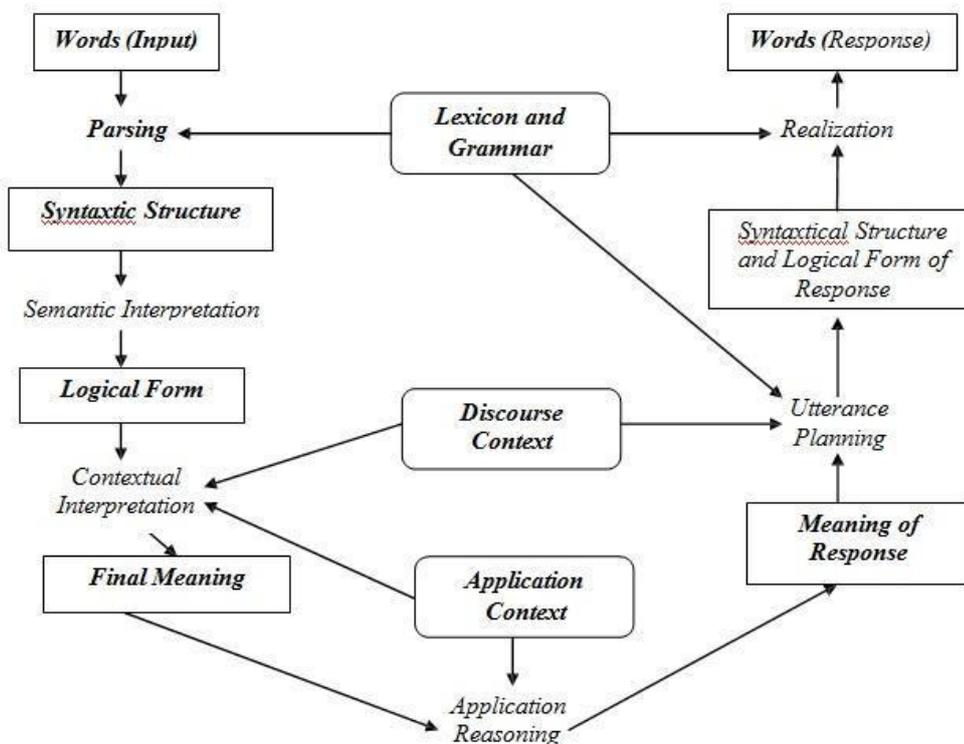
Sebuah sistem *natural language* harus memperhatikan pengetahuan terhadap bahasa itu sendiri baik dari segi kata yang digunakan bagaimana kata-kata tersebut digabung untuk menghasilkan suatu kalimat dan sebagainya. *Natural Language Processor* tidak memperdulikan bagaimana suatu kalimat diinputkan ke komputer. Tugasnya adalah mengekstrak informasi dari kalimat. Inti dari sistem *NLP* adalah *parser*. *Parser* adalah bagian dari program atau sistem yang membaca setiap kalimat, kata demi kata, untuk menentukan “*what is what*”.

NLP tidak bertujuan untuk mentransformasikan bahasa yang diterima dalam bentuk teks atau suara menjadi data digital dan/atau sebaliknya pula, melainkan bertujuan untuk memahami arti dari kalimat yang diberikan dalam

bahasa alami dan memberikan respon yang sesuai, misalnya dengan melakukan suatu aksi tertentu atau menampilkan data tertentu.

Untuk mencapai tujuan ini dibutuhkan tiga tahap proses. Proses yang pertama ialah *parsing* atau analisa *sintaksis* yang memeriksa kebenaran struktur kalimat berdasarkan suatu *grammar* (tata bahasa) dan *lexicon* (kosa kata) tertentu. Proses kedua ialah *semantic interpretation* atau *interpretasi* semantik yang bertujuan untuk merepresentasikan arti dari kalimat secara *context-independent* bertujuan untuk keperluan lebih.

Sedangkan proses ketiga ialah *contextual interpretation* atau interpretasi kontekstual yang bertujuan untuk merepresentasikan arti secara *context dependent* dan menentukan maksud dari penggunaan kalimat. Gambaran sebuah organisasi sistem *NLP* dapat dilihat pada gambar 2.1 [2].



Gambar 2.1 Organisasi Sebuah Sistem NLP

Jenis-jenis yang bisa dibuat pada bidang-bidang *natural language* adalah *text-based application* dan *dialogue-based applications*.

1. *Text-based application*

Text-based application ini adalah mencakup segala macam aplikasi yang melakukan proses terhadap teks tertulis seperti misalnya buku, berita di surat kabar, *email* dan lain sebagainya. Contoh penggunaan dari *text-based application* ini adalah :

- a. Mencari topik tertentu dari buku yang ada pada perpustakaan.
- b. Memberikan
- c. Respon atas *input* yang diberikan.
- d. Mencari isi dari surat atau *email*.
- e. Menterjemahkan dokumen dari satu bahasa ke bahasa yang lain.

2. *Dialogue-based application*

Dialogue-based application adalah idealnya pendekatan ini melibatkan bahasa lisan atau pengenalan suara, akan tetapi bidang ini juga memasukkan interaksi dengan cara memasukkan teks pertanyaan melalui *keyboard*. Aplikasi yang sering ditemui untuk bidang ini adalah :

- a. Sistem tanya jawab, dimana *natural language* digunakan dalam mendapatkan informasi dari suatu database.
- b. Sistem otomatis pelayanan melalui telepon.
- c. Kontrol suara pada peralatan sistem.
- d. Sistem *problem solving* yang membantu untuk melakukan penyelesaian masalah yang umum dihadapi dalam suatu pekerjaan [2].

2.3 *Text Mining*

Text mining merupakan salah satu aplikasi dari data *mining*. *Text mining* juga sering disebut sebagai *Text Data Mining* (TDM) dan *knowledge Discovery in Textual Databases* (KDT) [3]. *Text mining* merupakan proses mengesktrak *patterns* dan *knowledge* yang bersifat menarik dan nontrivial (penting) dari dokumen-dokumen teks. Pada intinya proses kerja *text mining* sama dengan

proses kerja *data mining* pada umumnya hanya saja data yang di *mining* merupakan *text databases*.

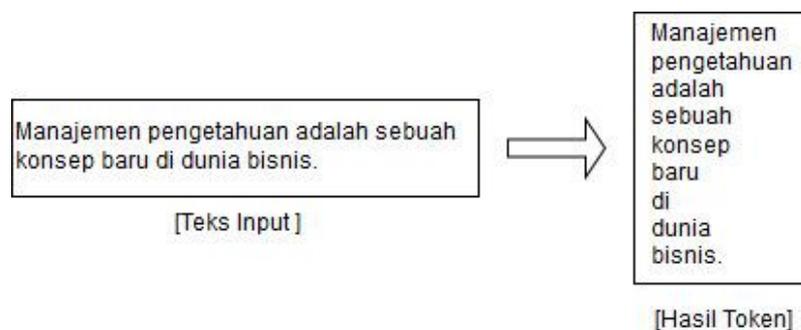
Data teks akan diproses menjadi data numerik agar dapat dilakukan proses lebih lanjut. Sehingga dalam *text mining* ada istilah *preprocessing* data, yaitu proses pendahulu yang diterapkan terhadap data teks yang bertujuan untuk menghasilkan data numerik.

Pada proses *preprocessing* merupakan tahap dimana deskripsi ditangani untuk dapat siap diproses memasuki tahap *text mining*. Tahap-tahap tersebut adalah:

1. *Parsing/Tokenizing*
2. *Stopwords Removal/Filtering*
3. *Stemming*

2.3.1 *Parsing/Tokenizing*

Parsing adalah sebuah proses yang dilakukan seseorang untuk menjadikan sebuah kalimat menjadi lebih bermakna atau berada dengan cara memecah kalimat tersebut menjadi kata-kata atau frase-frase ("*Parsing*"). *Parsing* di dalam pembuatan aplikasi *text mining* ini merupakan proses penguraian deskripsi yang semula berupa kalimat-kalimat berisi kata-kata dan tanda pemisah antara kata seperti titik(.), koma(,), spasi dan tanda pemisah lain menjadi kata-kata saja baik itu berupa kata-kata penting maupun kata-kata tak penting. Secara sederhana proses *parsing* ini terlihat sebagai proses pengambilan kata jika ketemu tanda spasi namun pada kenyataannya tidak sesederhana itu. Contoh tahap ini dapat dilihat pada gambar 2.2 [3].



Gambar 2.2 Tahap *Parsing/Tokenizing*

2.3.2 Filtering/Stop Removal

Kebanyakan bahasa resmi di berbagai negara memiliki kata fungsi dan kata sambung seperti artikel dan preposisi yang hampir selalu muncul pada dokumen-dokumen teks. Biasanya kata-kata ini memiliki arti yang lebih di dalam memenuhi kebutuhan seorang *searcher* di dalam mencari informasi. Kata-kata tersebut (misalnya *a, an, the on* pada bahasa Inggris) disebut sebagai *stopwords*. Di dalam bahasa Indonesia *stopwords* dapat disebut sebagai kata tidak penting misalnya “di”, “oleh”, “pada”, “sebuah”, “karena”. Sebelum proses *stopwords removal* dilakukan, terlebih dulu dibuat daftar *stopwords* (*stoplist*). Preposisi, kata hubung dan partikel biasanya merupakan kandidat *stoplist*.

Stopwords removal merupakan proses penghilangan kata tidak penting pada deskripsi melalui pengecekan kata-kata hasil parsing deskripsi apakah termasuk di dalam daftar kata tidak penting (*stoplist*) atau tidak. Jika termasuk di dalam *stoplist* maka kata-kata tersebut akan di-*remove* dari deskripsi sehingga kata-kata yang tersisa di dalam deskripsi dianggap sebagai kata-kata penting atau *keywords*. Tahap *filtering* dapat dilihat pada gambar 2.3 [3].



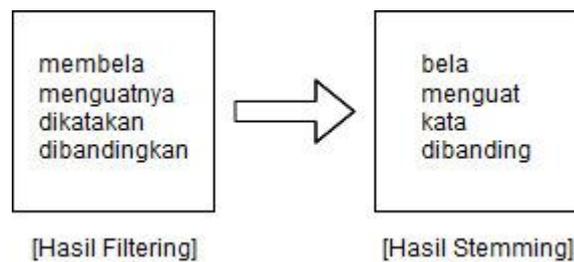
Gambar 2.3 Tahap Filtering

2.3.3 Stemming

Stemming adalah proses pemetaan dan penguraian berbagai bentuk (*variants*) dari suatu kata menjadi bentuk kata dasarnya (*stem*). Proses ini juga disebut sebagai *conflation*. Proses *stemming* secara luas sudah digunakan di dalam *Information retrieval* (pencarian informasi) untuk meningkatkan kualitas

informasi yang didapatkan. Kualitas informasi yang dimaksud misalnya untuk mendapatkan hubungan antara *variant* kata yang satu dengan yang lainnya. Sebagai contoh kata “diculik”, “menculik” (melakukan tindakan menculik) dan “penculik” (orang yang menculik) yang semula mengandung arti yang berbeda dapat di-*stem* menjadi sebuah kata “culik” yang memiliki arti yang sama sehingga kata-kata di atas saling berhubungan.

Selain itu *stemming* juga dapat digunakan untuk mengurangi ukuran dari suatu ukuran *index file*. Misalnya dalam suatu deskripsi terdapat *variant* kata “memberikan”, “diberikan”, “memberi” dan “diberi” hanya memiliki akar kata (*stem*) yaitu “beri”. Ukuran *file* daftar *index* yang semula berjumlah lima *record* akan di-*reduce* sehingga menjadi satu *record* saja. Gambaran tahap *stemming* dapat dilihat pada gambar 2.4 [4].



Gambar 2.4 Tahap *Stemming*

2.4 *Chatbot*

Chatbot adalah sebuah program komputer yang diprogram untuk dapat berkomunikasi dengan manusia atau *user* dengan menggunakan bahasa manusia itu sendiri. Salah satunya contoh kongkritnya adalah *Help Bot* pada Yahoo! Mesenger dan *ALICE (Artificial Linguistic Internet Computer Entity)* yang dikembangkan oleh Dr. Richard S. Wallace.

Atau dapat disebut juga dengan *chatbot (chatterbot atau bots)* merupakan sebuah program atau aplikasi komputer yang dirancang untuk menstimulasikan percakapan intelektual dengan satu atau lebih manusia baik secara audio maupun teks.

Pada mulanya, program komputer (*bots*) ini diuji melalui *Turing Test* yaitu dengan merahasiakan identitasnya sebagai mesin sehingga dapat melabui orang yang bercakap-cakap dengannya. Jika pengguna tidak dapat mengidentifikasi *bots* sebagai suatu program komputer, maka *chatbot* tersebut dikategorikan sebagai kecerdasan buatan (*artificial intelligence*). Zaman sekarang ini, *chatbot* telah dimanfaatkan untuk tujuan praktis seperti bantuan online, layanan personal, atau akuisis informasi, dalam hal ini dapat dilihat fungsi program sebagai suatu jenis agen percakapan (*conversational agent*). Yang membedakan *chatbot* dengan sistem pemrosesan bahasa alami (*natural language processing system*) adalah kesederhanaan algoritma yang digunakan.

Meskipun banyak *bots* yang tampaknya dapat menginterpretasikan dan menanggapi masukan dari manusia, sebenarnya *bots* tersebut hanya menganalisis kata kunci dalam masukan dan membalasnya dengan kata kunci yang paling cocok, atau pola kata-kata yang paling mirip dari basis data tekstual.

Chatbot juga merupakan *QA system* atau *question-answering system*, yaitu memberikan kemampuan pada sebuah mesin (komputer) untuk menginterpretasikan bahasa alami untuk melakukan dialog dengan pengguna hampir seperti dialog antara dua orang manusia dalam bahasa sehari-hari.

Baru-baru ini telah muncul ide-ide baru di dalam pengembangan *ALICE* di antaranya adalah kemampuan untuk menjaga alur pembicaraan untuk setiap *user* yang melakukan pembicaraan dengannya. Selain itu juga mulai telah dipikirkan suatu cara agar nantinya program tersebut dapat memprediksi atau mengidentifikasi umur, jenis kelamin, lokasi geografis dan pekerjaan dari *client* (*user*) yang melakukan perbincangan dengannya [2].

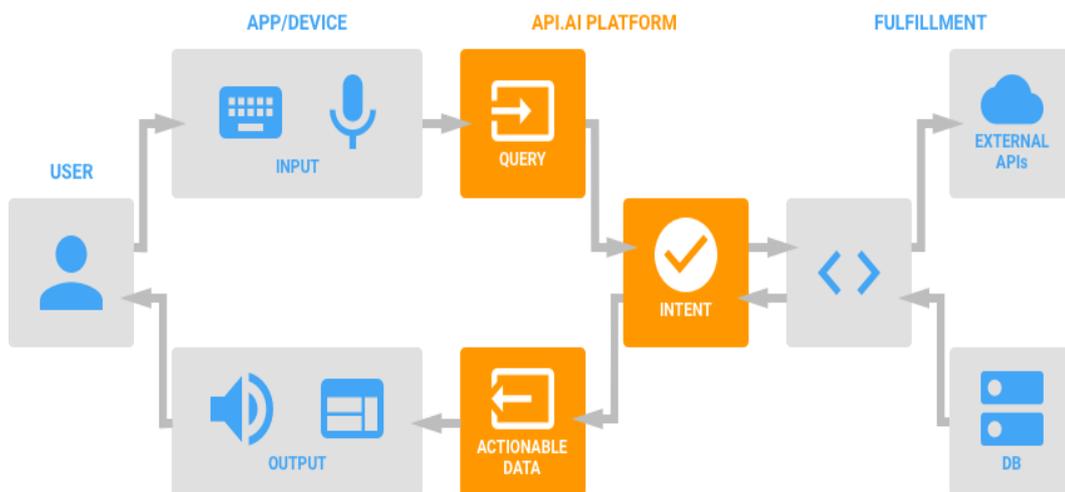
2.4.1 Penggunaan *Brainfile* pada *Chatbot*

Brainfile adalah pusat atau otak dari *chatbot* itu sendiri yang menentukan bagaimana cara *chatbot* berpikir dan akan memberikan respon. *Brainfile* biasanya cara *chatbot* berupa *file plain text*. *Brain file* berfungsi sebagaimana tabel informasi pada kompilator bahasa pemrograman tingkat tinggi. Di dalam *brain file* inilah disimpan semua kosakata, kepribadian, dan pengetahuan (*knowledge*)

dari *chat bot*. Semakin banyak pengetahuan yang dimiliki *chat bot* maka akan semakin besar ukuran *file* dari *brain file* tersebut [2].

2.5 *Dialogflow*

Dialogflow (dulunya *Api.ai*) adalah teknologi interaksi manusia dengan komputer yang berbasis pada percakapan suara dan teks yang didukung *Artificial Intelegent*. *Dialogflow* juga merupakan milik Google teknologi pengembangan interaksi manusia dengan komputer berdasarkan percakapan bahasa alami. *Dialogflow* menyediakan *platform* yang memungkinkan pengembang untuk merancang dan mengimplementasikan antarmuka percakapan yang dapat disematkan aplikasi eksternal seperti *bot*. Aliran data dasar dalam sistem *Dialogflow* ditampilkan pada gambar 2.5 [5] :



Gambar 2.5 Data Flow in Dialogflow [6]

Dialogflow mendukung 14 *platform* berbeda termasuk *Skype*, *Telegram*, *Slack*, *Cortana*, *Alexa* dan *Facebook Messenger*. Namun, meskipun aplikasi *bot* dibuat dengan *Dialogflow* secara teknis memungkinkan berintegrasi dengan *Azure Bot* layanan; itu akan menuntut persiapan tambahan, karena *bot Dialogflow* dirancang untuk bekerja dengan *Google sendiri Platform Cloud* [5].

2.5.1 Komponen-komponen *Dialogflow*

Di dalam *dialogflow* terdapat beberapa komponen yaitu sebagai berikut :

1. *Agent*

Merupakan modul NLU (*Natural Language Understanding*). Digunakan untuk manajemen *flow* percakapan.

2. *Intent*

Merupakan *mapping* antara percakapan *user* dan *action* dari percakapan tersebut, sebagai contoh :

- *User* : Hai
- *Bot* : Hai juga, apa kabar? Sibuk apa hari ini?

3. *Entities*

Fitur yang digunakan untuk ekstrak parameter data dari *natural language*.

- *System* : Parameter yang disediakan oleh *Dialogflow*
- *Developer* : Parameter yang di-*difine* oleh *Developer*
- *User* : Parameter yang di-*redefine* oleh user (session ID level)

4. *Context*

Merupakan *parsing* parameter dari percakapan sebelumnya. Digunakan untuk membedakan percakapan, dan menentukan *flow* percakapan.

5. *Fulfillment*

Webhook yang di-*trigger* dari *Dialogflow*

2.6 *Smartphone*

Smartphone atau ponsel cerdas merupakan kombinasi dari PDA dan ponsel, namun lebih berfokus pada bagian ponselnya. *Smartphone* ini mengintegrasikan kemampuan ponsel dengan fitur komputer-PDA. *Smartphone* mampu menyimpan informasi, *email* dan instalasi program, seperti menggunakan *mobile phone* dalam satu *device*. *Smartphone* biasanya berorientasi pada fitur ponsel dibanding dengan fitur PDA. Sebagian besar perangkat *mobile* yang melebihi kemampuan ponsel dapat dikategorikan sebagai *smartphone*. Banyak

yang mendefinisikan *smartphone* adalah ponsel yang di dalamnya berisi inovasi *gadget* termutakhir [7].

2.7 Android

Android adalah istilah dalam bahasa Inggris yang berarti “Robot yang menyerupai manusia”. Logo “Android” sendiri, dicerminkan seperti sebuah robot berwarna hijau, yang mengacu kepada arti kata Android.



Gambar 2.6 Logo Adroid

Android adalah sebuah sistem operasi untuk *Smartphone* dan Tablet. Sistem operasi dapat diilustrasikan sebagai ‘jembatan’ antara piranti (*device*) dan penggunaanya, sehingga pengguna bisa berinteraksi dengan *devicenya* dan menjalankan aplikasi-aplikasi yang tersedia pada *device*. Di dunia personal komputer, sistem operasi yang banyak dipakai adalah *Windows*, *MAC*, dan *Linux*.

Di dunia *mobile devices* (*smartphone* dan tablet), sistem operasional yang menguasai pasar saat ini adalah Android. Menurut data *market sharei* dari Gartner, Inv. Pada perengahan awal tahun 2013, Android memegang 79% *market share smartphone* di seluruh dunia, *iOS* yang merupakan sistem operasi dari *iPhone* di peringkat ketiga dengan 3,3 %, dan *Blackberry* di peringkat keempat dengan 2,7 % *market share*. Di dunia tablet, Android juga mendominasi dengan 56,5% *market share*, *iOS* dengan *iPad*-nya kembali menduduki peringkat kedua dengan 40% *market share* [8].

Tabel 2.1 Market Share Smartphone Seluruh Dunia Tahun 2013 [8]

Operating System	2Q13	2Q13 Market Share (%)	2Q12	2Q12 Market Share (%)
	Units		Units	
Android	177,898.2	79.0	98,664.0	64.2
iOS	31,899.7	14.2	28,935.0	18.8
Microsoft	7,407.6	3.3	4,039.1	2.6
BlackBerry	6,180.0	2.7	7,991.2	5.2
Bada	838.2	0.4	4,208.8	2.7
Symbian	630.8	0.3	9,071.5	5.9
Others	471.7	0.2	863.3	0.6
Total	225,326.2	100.0	153,772.9	100.0

Source: Gartner (August 2013)

Android adalah sistem operasi yang bersifat *open source* (sumber terbuka). Disebut *open source* karena *source code* (kode sumber) dari sistem operasi Android dapat dilihat, *download*, dan dimodifikasi secara bebas. Paradigma *open source* ini memudahkan pengembangan teknologi Android, karena semua pihak yang tertarik dapat memberikan kontribusi, baik pada pengembangan sistem operasi maupun aplikasi. Pada awal Oktober 2013, tercatat ada lebih dari 850.000 aplikasi Android yang tersedia di *Google Play* (dulu bernama Android Market) [8].

2.7.1 Versi dan Jenis-jenis Android

Pengembangan Android dimulai dengan berdirinya Android, Inc. pada Oktober 2013 dengan tujuan membuat *mobile device* yang lebih *smart* untuk menyaingi *Symbian* dan *Windows Mobile* yang populer saat itu (*iPhone* dan *Blackberry* belum dirilis). Pada tahun 2005, Android, Inc. diakuisisi oleh Google. Pengembangan terus dilanjutkan sampai Android versi beta diluncurkan pada tanggal 5 November 2007, bersamaan dengan berdirinya OHA (*Open Handset Alliance*). Sampai saat ini tanggal 5 November diperingati sebagai hari jadi

Android SDK (*Software Development Kit*) diluncurkan, sehingga pengguna dapat membuat dan mengembangkan aplikasi Android mereka sendiri.

Pada tanggal 23 September 2008, sistem operasi Android versi 1.0 resmi diluncurkan. Sekitar sebulan berikutnya, pada tanggal 22 Oktober 2008, *smartphone* pertama yang menjalankan Android 1.0 itu, yaitu HTC *Dream*, diluncurkan ke pasar.

Pada tanggal 9 Februari 2009, Android versi 1.1 diluncurkan untuk memperbaiki bug dari versi sebelumnya dan menambahkan fitur yang tersedia. Setelah versi 1.1, dirilis Android berikutnya memakai nama makanan manis dengan urutan alfabetik, dimulai dengan Android 1.5 *Cupcake* yang diluncurkan pada tanggal 30 April 2009.

Rilis-rilis Android selanjutnya, yaitu *Donut*, *Eclair*, *Froyo*, dan *Gingerbread* semua dibuat untuk *smartphone*. Namun *Apple* meluncurkan *iPad* pada tahun 2010 dan meningkatkan ketertarikan masyarakat luas kepada komputer tablet. Beberapa pengembang Android mencoba mengembangkan tablet Android untuk menyaingi *iPad*, seperti Samsung Galaxy Tab yang menggunakan *Gingerbread* yang dikustomisasi. Pada tanggal 22 Februari 2011, Android *Honeycomb* diluncurkan ke pasar, dan pada tanggal 24 Februari 2011, tablet pertama yang menggunakan *Honeycomb*, yaitu *Motorola Xoom*, diluncurkan ke pasar.

Pada tanggal 19 Oktober 2011, Android meluncurkan *IceCream Sandwich*. Versi ini dapat bekerja secara optimal baik di *Smartphone* maupun di Tablet. Rilis Android berikutnya, yaitu *Jelly Bean*, bertujuan untuk semakin meningkatkan apa yang sudah tersedia di *Ice Cream Sandwich*, dengan memperbaiki *bug-bug* dan menambahkan fitur-fitur.

Pada tanggal 19 Oktober 2013, diumumkan versi Android selanjutnya adalah Android 4.4 *Kit Kat*. Android sudah mendapatkan izin dari *Nestle* dan *Hershey* selaku pemilik merek dagang *Kit Kat*. Sebelumnya pengumuman ini, banyak yang berspekulasi bahwa versi Android berikutnya akan diberi nomor 5.0 dengan nama *Key Lime Pie* [8].

2.8 Perangkat Lunak

Rekayasa perangkat lunak adalah sebuah profesi yang dilakukan oleh seorang perancang perangkat lunak yang berkaitan dengan pembuatan dan pemeliharaan aplikasi perangkat lunak dengan menerapkan teknologi dan praktik dari ilmu komputer, manajemen proyek dan bidang-bidang lainnya. Perangkat lunak adalah intruksi langsung komputer untuk melakukan pekerjaan dan dapat ditemukan di setiap aspek kehidupan modern dari aplikasi yang kritis untuk hidup (*life-critical*), seperti perangkat pemantauan medis dan pembangkit tenaga listrik sampai hiburan, seperti video *game*. Banyak produk perangkat lunak berisi jutaan baris kode yang diharapkan dapat melakukan pekerjaan dengan baik dalam menghadapi perubahan kondisi. Semua perangkat lunak juga membutuhkan keandalan yang tinggi dan harus dihasilkan secara ekonomis.

Rekayasa atau teknik adalah penerapan ilmu dan teknologi untuk menyelesaikan permasalahan manusia. Hal ini diselesaikan lewat pengetahuan, matematika dan pengalaman praktis yang diterapkan untuk mendesain objek atau proses yang berguna. Para praktisi teknik profesional disebut perancang (sarjana teknik).

Menurut sejarahnya, banyak para ahli yang meyakini kemampuan teknik manusia sudah tertanam secara alami. Hal ini ditandai dengan kemampuan manusia purba untuk membuat peralatan-peralatan dari batu. Dengan kata lain, teknik pada mulanya didasari dengan metode coba dan ralat (*trial and error*). Seiring dengan berjalannya waktu, ilmu pengetahuan mulai berkembang dan mulai mengubah cara pandang manusia terhadap bagaimana alam bekerja [9].

Pendefinisian istilah “rekayasa perangkat lunak” telah dilakukan oleh banyak pihak, antara lain:

Menurut Ian Sommerville :

Rekayasa perangkat lunak berkaitan dengan teori, metode dan kaidah yang diperlukan dalam mengembangkan perangkat lunak untuk komputer.

Menurut Fritz Bauer :

Rekayasa perangkat lunak adalah penetapan dan penggunaan prinsip-prinsip rekayasa yang tangguh/teruji dalam upaya memperoleh perangkat lunak secara ekonomis, handal dan bekerja efisien di mesin nyata.

Menurut Mynatt :

Rakayasa perangkat lunak secara sederhana dinyatakan sebagai pendekatan sistematis untuk penciptaan dan pemeliharaan perangkat lunak.

Menurut IEEE : [IEEE Standard Glossary of SE Terminology (IEE 83)] :

Rekayasa perangkat lunak adalah pendekatan sistematis untuk pengembangan, operasi, pemeliharaan, dan pemberhentian pemakaian perangkat lunak.

Rekayasa perangkat lunak adalah pendekatan yang sistematis, berdisiplin dan dapat dikuantifikasi untuk pengembangan, pengoperasian dan pemeliharaan perangkat lunak, yaitu penerapan rekayasa terhadap perangkat lunak.

Rekayasa perangkat lunak merupakan studi menyeluruh terhadap teori-teori, proses-proses, metode-metode, dan teknik-teknik untuk membangun perangkat lunak berkualitas tinggi dengan cara-cara yang *cost-effective*.

Rekayasa perangkat lunak melibatkan orang (*people*), proses (*process*), proyek (*project*), dan produk (*product*) [10].

2.8.1 Bahasa Pemrograman Java

Java adalah nama sebuah bahasa pemrograman yang sangat terkenal. Sebagai bahasa pemrograman, *Java* dapat digunakan untuk menulis program. Sebagaimana diketahui, program adalah kumpulan instruksi yang ditujukan untuk komputer. Melalui program, komputer dapat diatur agar melaksanakan tugas tertentu sesuai yang ditentukan oleh pemrogram (orang yang membuat program). Sebagai contoh, Anda mungkin terbiasa menggunakan pengolahan kata seperti *Word*. Nah, *Word* adalah contoh aplikasi yang dibuat dengan menggunakan bahasa pemrograman.

Bahasa *Java* dikembangkan di *Sun Microsystems* dan mulai diperkenalkan kepada publik pada tahun 1995. Seperti halnya *C++*, *Java* juga merupakan bahasa yang berorientasi objek. Dengan demikian, *Java* juga memudahkan dalam pembuatan aplikasi yang berskala besar.

Sebagai bahasa yang beraras tinggi, yang menggunakan perintah-perintah yang mudah dimengerti oleh orang, *Java* mempunyai keunggulan yakni bersifat universal. Sebagai bahasa yang universal, *Java* bisa dijumpai *platform* (*Linux*, *UNIX*, *Windows*, *Mac* dan lain-lain). Artinya, jika Anda menguasai *Java* di *platform* PC sangat mudah untuk berpindah di *Linux* ataupun sistem operasi yang lain. Hal yang menarik lagi, hasil kompilasi *Java* yang dinamakan *bytecode* dapat dijalankan di berbagai *platform* sepanjang di sistem target memiliki *Java Runtime Environment* (JRE) [11].

Java dikembangkan dengan mengadaptasi *C++*. Bahasa *Java* distandarkan secara terbuka oleh *Sun Microsystems*. Hasil kompilasi bahasa *Java* berupa *bytecode* yang akan dijalankan di *Java Virtual Machine*. Hasil kompilasi ini dapat dijalankan di *platform*. *Bytecode* distandarkan secara terbuka oleh *Sun Microsystems* agar pembuat *Java Virtual Machine* dapat mengimplementasikannya secara seragam sehingga *bytecode* hasil kompilasi di *platform* manapun dapat dijalankan di sembarang *platform*.

Perkembangan bahasa *Java* selanjutnya adalah sebagai bahasa untuk menyaingi gagasan *Microsoft* dalam *Distributed (Objects) Application* pada proyek DNA (*Distributed interNet Application*), *Sun* (dibantu perusahaan-perusahaan pesaing *Microsoft* lain) telah mengeluarkan J2EE (*Java Platform 2nd Enterprise Edition*). *Java* telah dikembangkan oleh bahasa pengembangan aplikasi terbesar skala *enterprise* secara terpadu. *Java* dimaksudkan sebagai bahasa Inggris untuk pengembangan *Java Applet*, *Java Script*, *Java Servlet*, *Java Server Pages* (*JSP*), khusus pada arsitektur N-tier yang dikembangkan untuk aplikasi tersebar skala *enterprise* [10].

2.8.2 Database

Database adalah sebuah struktur yang umumnya terbagi dalam dua hal, yaitu sebuah *database flat* dan sebuah *database* relasional. *Database* relasional lebih mudah dipahami daripada *database flat* karena *database* relasional mempunyai bentuk yang sederhana serta mudah dilakukan operasi data. *MySQL* sendiri adalah sebuah *database* relasional. *Database* yang memiliki struktur relasional terdapat tabel-tabel untuk menyimpan data. Pada setiap tabel terdiri dari kolom dan baris serta sebuah kolom untuk mendefinisikan jenis informasi apa yang harus disimpan [12].

2.8.3 Perancangan Antar Muka

Perancangan antar muka atau perancangan *interface* secara detail dibahas di disiplin ilmu HCI (*Human Computer Interaction*) atau IMK (Interaksi Manusia dan Komputer). Meski demikian, akan dibahas secara sekilas mengenai aspek *interface* dalam sebuah tahapan perancangan perangkat lunak.

Interaksi manusia dan komputer bertujuan untuk mengembangkan keamanan, utilitas, efisiensi dan *usability* dari sistem yang memakai komputer serta memberikan pedoman bagi para desainer dalam mendesain sistem yang *usabel*. Dalam kaitannya, interaksi manusia dan komputer merupakan pendukung utama dari perancangan antar muka. Sebab perancangan antar muka bertujuan untuk menjembatani antara kepentingan pengguna dengan perangkat lunak yang akan dibangun.

Yang dimaksud dengan istilah *interface* atau antar muka sendiri adalah *the part of the system that you see, hear and feel*. Jadi antar muka adalah bagian dari perangkat lunak yang dapat disarankan oleh panca indera pengguna baik dari sisi penglihatan pendengaran maupun diraba bahkan juga dapat dicium (untuk perangkat lunak tertentu). [13]

2.9 Object Oriented Programming (OOP)

Object Oriented Programming (OOP) merupakan sebuah pendekatan untuk pengembangan atau *development* suatu *software* dimana dalam struktur

software tersebut didasarkan kepada interaksi *object* dalam penyelesaian suatu proses atau tugas. Interaksi tersebut mengambil *form* dari pesan-pesan dan mengirimkan kembali antar *object* tersebut. *Object* akan merespon pesan tersebut menjadi sebuah tindakan atau *action* atau metode.

Object-oriented program tersiri dari *object* yang berinteraksi satu sama lainnya untuk menyelesaikan sebuah tugas. Seperti dunia nyata, *users* dari *software* program dilibatkan dari logika proses untuk menyelesaikan tugas. Adapun karakteristik yang dimiliki oleh OOP sebagai berikut :

a. *Object*

Object merupakan sebuah struktur yang menggabungkan data dan prosedur untuk bekerja bersama-sama.

b. *Abstraction*

Abstraction merupakan interaksi antara manusia dengan objek-objek di dunia ini, manusia seiring hanya konsentrasi dengan sebuah bagian dari propertiesnya.

Ketika membangun objek dalam aplikasi OOP, adalah penting untuk menggabungkan konsep abstraksi ini. Jika membangun aplikasi *shipping*, maka harus membangun objek produk dengan atribut seperti ukuran dan berat. Warna adalah contoh informasi yang tidak ada hubungannya dan harus dibuang. Tetapi ketika akan membangun *order-entry application*, warna menjadi penting dan harus termasuk atribut objek produk.

c. *Encapsulation*

Ciri penting lainnya dari OOP adalah *encapsulation*. *Encapsulation* ialah sebuah proses dimana tidak ada akses langsung ke data yang diberikan, bahkan *hidden*. Jika ingin mendapatkan data, harus berinteraksi dengan objek yang bertanggung jawab atas data tersebut.

d. *Polymorphism*

Merupakan kemampuan dua buah objek yang berbeda untuk merespon pesan permintaan yang sama dalam suatu cara yang unik. Dalam OOP, diterapkan tipe *polymorphism* melalui proses yang disebut *overloading*. Dapat dilakukan

dalam implementasi metode yang berbeda pada sebuah objek yang mempunyai nama yang sama.

e. *Inheritance*

Penggunaan *inheritance* dalam OOP untuk mengklasifikasi objek dalam program sesuai karakter umm dan fungsinya [13].

2.9.1 *Unified Modeling Language (UML)*

UML (*Unified Modeling Language*) adalah bahasa grafis untuk mendokumentasikan, mespesifikasikan dan membangun sistem perangkat lunak. UML beroorientasi objek, menerapkan banyak level abstraksi, tidak bergantung proses pengembangan, tidak bergantung bahasa dan teknologi, pemanduan beberapa notasi di beragam metodologi, usha bersama dari pihak , didukung oleh kakas-kakas yang diintegrasikan lewat XML (XMI). Standar UML dikelola oleh OMB (*Object Management Group*).

UML adalah bahasa pemodelan untuk menspesifikasikan, memvisualisasikan, membangun dan mendokumentasikan artifak-artifak dari sistem [10].

2.9.2 **Diagram dan Teknik Pemodelan**

Diagram mengemukakan banyak hal, penggunaan notasi yang terdefinisi bnaik dan ekspresif adalah penting pada proses pengembangan perangkat lunak, yaitu :

1. Notasi standar memungkinkan pengembangan mendeskripsikan skenario atau rumusan arsitektur dan kemudian mengkomunikasikan secara tidak ambigu.
2. Notasi yang bagus membebaskan otak untuk berkonsentrasi pada masalah-masalah yang lebih lanjut.
3. Notasi yang baik memungkinkan mengeliminasi keperluan pemeriksaan konsistensi dan kebenaran keputusan-keputusan dengan menggunakan *tool* terotomatisasi [10].

2.9.3 Diagram Struktur

Diagram ini untuk memvisualisasi, mespesifikasikan, membangun dan mendokumentasikan aspek statik dari sistem. Diagram struktur di UML terdiri dari :

1. *Class Diagram* (Diagram Kelas)

Diagram ini menunjukkan sekumpulan kelas, *interface* dan kolaborasi dan keterhubungannya. Diagram kelas ditujukan untuk pandangan statik terhadap sistem.

2. *Object Diagram* (Diagram Objek)

Diagram ini menunjukkan sekumpulan objek dan keterhubungannya. Diagram ini menunjukkan potongan statik dari instan-instan yang ada di diagram kelas. Diagram ini untuk memperlihatkan suatu protipe atau kasus tertentu yang mungkin terjadi.

3. *Use Case Diagram* (Diagram *Use Case*)

Diagram ini menunjukkan sekumpulan kasus fungsional dan aktor (jenis kelas khusus) dan keterhubungannya.

4. *Sequance Diagram* (Diagram sekuen)

Diagram ini menunjukkan interaksi yang terjadi antar objek. Diagram ini merupakan pandangan dinamis terhadap sistem. Diagram ini menekankan pada basis keberurutan waktu dari pesan-pesan yang terjadi [10].

2.9.4 Diagram Perilaku

Diagram ini untuk memvisualisasi, menspesifiasikan, membangun dan mendokumentasikan aspek dinamis dari sistem. Diagram perilaku di UML terdiri dari:

5. *Collaboraton Diagram* (Diagram Kolaborasi)

Diagram ini juga merupakan diagram intreaksi. Diagram ini menekankan pada organisasi struktur dari objek-objek yang mengirim dan menerima pesan.

6. *Statechart Diagram* (Diagram *Statechart*)

Diagram ini adalah *state-machine diagram*, berisi *state*, transisi, kejadian dan aktivitas. *Statechart* merupakan padangan dinamis dari sistem. Diagram ini penting dalam memodelkan perilaku antarmuka, kelas, kolaborasi dan menekankan pada urutan kejadian. Penting untuk sistem reaktif yang dipicu kejadian di dunia nyata.

7. *Activity Diagram* (Diagram Aktivitas)

Diagram ini untuk menunjukkan aliran aktivitas di sistem. Diagram ini adalah pandangan dinamis terhadap sistem. Diagram ini penting untuk memodelkan fungsi sistem dan menekankan pada aliran kendali di antara objek-objek.

8. *Component Diagram* (Diagram Komponen)

Diagram ini menunjukkan organisasi dan kebergantungan di antara sekumpulan komponen. Diagram ini merupakan pandangan statik terhadap implementasi sistem.

9. *Deployment Diagram* (Diagram *Deployment*)

Diagram ini menunjukkan konfigurasi pemrosesan saat jalan dan komponen-komponen yang terdapat di dalamnya. Diagram ini merupakan pandangan statik dari arsitektur [10].

2.10 Profil Keuskupan Bandung

Keuskupan Bandung adalah wilayah gerjani Gereja Katolik Roma yang terletak di bagian timur dan tengah tanah Parahyangan Jawa Barat dengan luas wilayah 23.315,31 Km² (*Up Date 25-6-2015*). Selain Kota dan Kabupaten Bandung, wilayah Keuskupan Bandung meliputi Kabupaten Purwakarta, dan Karawang di sebelah Barat; Subang, Pamanukan, Indramayu di sebelah Utara; Cirebon, Kuningan, Tasikmalaya-Ciamis di sebelah Timur ; dan Garut di sebelah Selatan.

Pada tanggal 12 November 2010, pukul 18.00 WIB diumumkan secara resmi oleh Tahta Suci bahwa Mgr. Johannes Pujasumarta, Uskup Diosesan Keuskupan Bandung, diangkat menjadi Uskup Diosesan Keuskupan Agung

Semarang dan Mgr. Ignatius Suharyo, Uskup Agung, Keuskupan Agung Jakarta diangkat menjadi Administrator Apostolik Keuskupan Bandung hingga terpilih Uskup yang baru.

Dengan penuh syukur, pada tanggal 3 Juni 2014, tepat pukul 12.00 waktu Roma (17.00 WIB), Bapa Paus Fransiskus mengumumkan pengangkatan **Mgr. Antonius Subianto Bunjamin, OSC**, dimana sebelumnya menjabat Provinsial Ordo Salib Suci, kini sebagai Uskup Keuskupan Bandung.

Di bawah kepemimpinan beliau wilayah gereja ini dibagi menjadi wilayah territorial yang disebut paroki. Keuskupan Bandung meliputi 24 Paroki dan 2 Kuasi; 13 Paroki dan 1 Kuasi diantaranya terletak di kota Bandung dan sekitarnya. 24 Paroki dan 2 Kuasi, kini menjadi tempat bernaung Umat Katolik Keuskupan Bandung yang saat ini berjumlah kurang lebih **96.281 orang** (*berdasarkan data SIMU 21-06-2017 pukul 09.00 WIB*).

Pelayanan dan penggembalaan di Paroki dipercayakan oleh Uskup kepada para Imam (Pastor), baik Imam diosesan maupun tarekat. Imam diosesan adalah pria yang ditahbiskan menjadi Imam Katolik dan dengan keataatannya kepada Uskup terikat seumur hidup untuk pelayanan di Keuskupan, dengan tetap terbuka untuk ditugaskan oleh Uskup di wilayah gerejani lain. Imam Tarekat adalah Imam yang menjadi anggota suatu tarekat/ordo/lembaga religious Gereja Katolik. Atas kesepakatan dan mandate dari Uskup, Imam Tarekat tersebut ditugaskan oleh pimpinannya untuk tugas pelayanan di Keuskupan. Tarekat Imam yang bekerja di Keuskupan Bandung saat ini ialah Ordo Salib Suci (OSC) dan Kongregasi Hati Kudus Yesus dan Maria (SS.CC).

Selain pelayanan parokial, keuskupan juga menjalankan pelayanan kategorial yang dilaksanakan baik oleh komisi ataupun yayasan keuskupan. Komisi adalah unit kerja keuskupan di bawah koordinasi Dewan Karya Pastoral (DKP) untuk menangani bidang pelayanan tertentu baik yang menyangkut kehidupan internal gereja maupun sosial kemasyarakatan, seperti Komisi Liturgi, Kitab Suci, Pengembangan Sosial Ekonomi Masyarakat, Pendidikan, Perburuan, Hubungan antar Agama dan Kepercayaan. Yayasan Keuskupan adalah yayasan pendidikan dan kesehatan. Melalui yayasan-yayasan pendidikan yang telah

didirikan, saat ini Keuskupan menjalankan 89 sekolah tersebar di seluruh wilayah keuskupan dari tingkat taman kanak-kanak sampai atas, 2 perguruan tinggi, dan 1 Balai Latihan Kerja. Bidang pelayanan kesehatan terdiri dari 4 rumah sakit dan 6 Balai Pengobatan Paroki.

Selain yayasan keuskupan, hadir pula atas izin Uskup Bandung, yayasan-yayasan yang didirikan baik oleh tarekat suster (biarawati) maupun sekelompok umat Katolik untuk membantu pelayanan social dan pendidikan di wilayah Keuskupan Bandung. Mereka menjalankan 26 sekolah dari taman kanak-kanak sampai menengah atas, 3 panti wredha, 1 pusat rehabilitasi narkoba. Saat ini 5 tarekat suster yang bekerja di Keuskupan Bandung : CB, PI, OP, OSU, dan Ursulin Somasca. Selain itu, terdapat pula pertapaan suster Karmel (OCD) di Lembang yang tentunya memberikan warna tersendiri bagi Keuskupan Bandung.



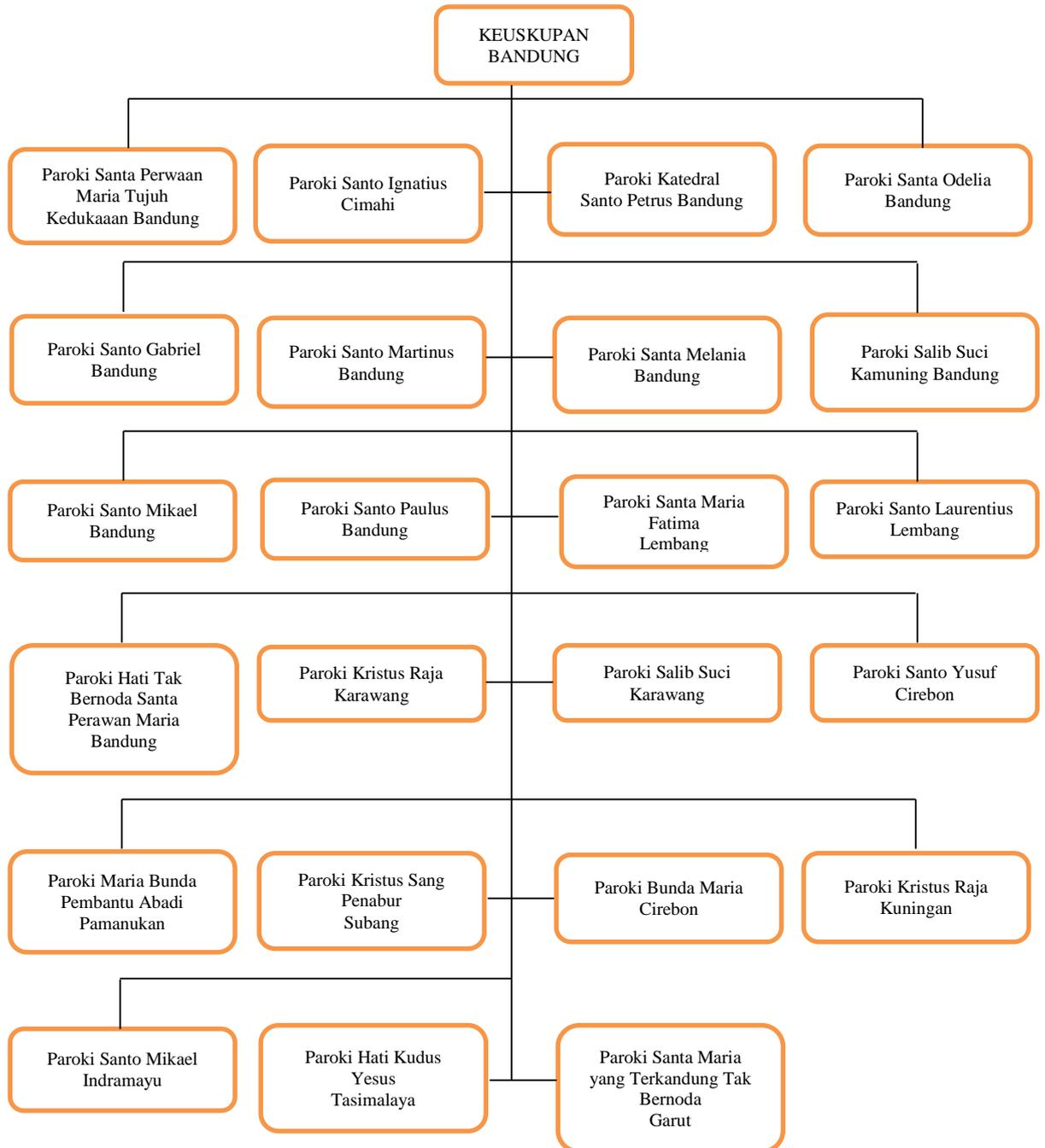
Gambar 2.7 Wisma Uskup Bandung

Dengan dinamika kehidupan dan pelayanan baik internal maupun eksternal, Gereja Keuskupan Bandung berkomitmen untuk menjadi bagian integral dan Masyarakat Jawa Barat dan Indonesia pada khususnya, Asia dan dunia pada umumnya.

Hal ini ditegaskan dalam Arah Dasar Pastoral 2010 yang menyatakan bahwa Gereja Keuskupan Bandung hendak menjadi komunitas yang hidup, mengakar, mekar dan berbuah bersama masyarakat Jawa Barat. Dengan demikian komitmen Gereja Katolik Keuskupan Bandung ialah bersama komunitas-komunitas lain lintas Agama, ras, suku, dan bahasa yang hadir di Jawa Barat membangun suatu masyarakat yang bermartabat dan manusiawi. Dilandasi oleh kemurahan hati dan kerelaan berbagi, Gereja Katolik Keuskupan Bandung hendak berperan aktif dalam peningkatan kesejahteraan material dan spiritual masyarakat [14].

2.10.1 Struktur Wilayah

Berikut ini adalah struktur wilayah Keuskupan Bandung.



Gambar 2.8 Struktur Wilayah

Dari gambar 2.8 terlihat ada 23 paroki yang berada di wilayah Keuskupan Bandung yakni 13 paroki yang berda di Kota Bandung dan 10 paroki berada di daerah lainnya.

2.10.2 Struktur Kuria Keuskupan (Struktur Organisasi)

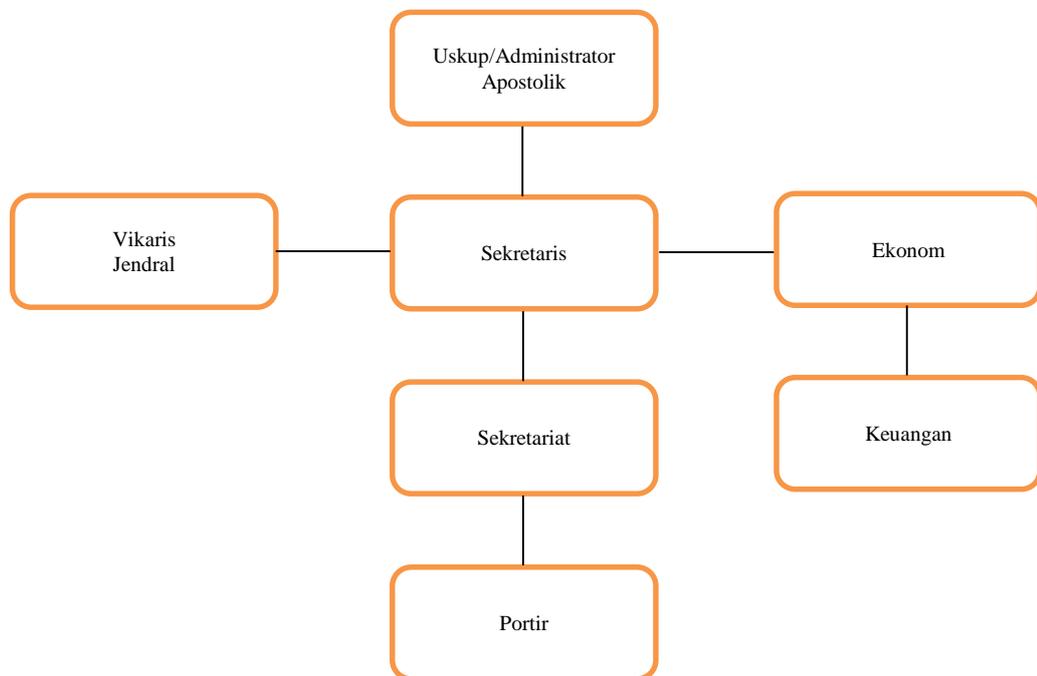
Kuria Keuskupa adalah tim kerja yang membantu Uskup dalam memimpin Keuskupan, terutama dalam mengarahkan karya pelayanan pastoral, melaksanakan administrasi Keuskupan dan dalam menjalankan kuasa yudisial.

Uskup : Mgr. Antonius Subianto Bunjamin, OSC.

Vikaris Jendral : R. D. Yustinus Hilman Pujiatmoko.

Sekretaris : R. P. Ignatius Eddy Putranto, OSC.

Ekonom : R. D. Antonius Sulastijana.



Gambar 2.9 Struktur Kuria Keuskupan Bandung

Berdasarkan gambar 2.3 berikut ini jobdes atau tugas dari masing-masing bagian :

Tabel 2.2 Jobdes

No.	Jabatan	Jobdes
1.	Uskup/Administrator Apostolik	Memimpin pengembalaan umat Katolik di Keuskupan Bandung.
2.	Vikaris Jendral	Mewakili sebagian tugas atau wewenang Uskup di Keuskupan Bandung.

3.	Sekretaris	Mengatur atau mengelola arsip-arsip atau keadministrasian Uskup dan Keuskupan Bandung.
4.	Ekonom	Mengelola harta-benda Keuskupan dibawah otoritas Uskup dan dari pendapatan Keuskupan yang telah ditetapkan melakukan pengeluaran-pengeluaran yang diperintahkan Uskup atau orang-orang lain yang ditugaskan dengan legitim olehnya.
5.	Sekretariat	Di bagian sekretarian ini terdiri dari 3 orang dimana masing-masing memiliki tugas seperti satu orang bertugas untuk membuat Surat Keputusan (SK) tugas penempatan para Pastor sesuai keputusan Uskup, mengelola data Gereja, mengelola data umat paroki, data sekolah, data yayasan, dta lembaga hidup bakti, dan lain-lain; satu orang mengelola data majalah komuni, mengelola data kendaraan se-Keuskupan dan pengurusnya.
6.	Keuangan	Membuat laporan keuangan Keuskupan dan paroki-paroki, juga komisi-komisi yang ada di Keuskupan Bandung.
7.	Portir	Menerima tamu, telepon, dan surat-surat untuk Kuria, juga mengirim surat dari Kuria.

