BAB II
TINJAUAN PUSTAKA

Berikut ini adalah beberapa tinjauan pustaka yang menjadi pokok bahasan dalam pembuatan tugas akhir yaitu *Smart Home*, sistem operasi Android, pendekatan berorientasi objek, *Unified Modelling Language* (UML), pemrograman java, Android studio serta pengujian *Black Box.*

2.1 Smart Home

Definisi rumah pintar (*Smart Home)* adalah sebuah tempat tinggal atau kediaman yang menghubungkan jaringan komunikasi dengan peralatan listrik untuk memungkinkan dikontrol, dimonitor, atau diakses dari jarak jauh [1]. Dalam operasinya, *Smart Home* dibantu oleh komputer atau *smartphone* untuk memberikan fasilitas-fasilitas yang diinginkan secara otomatis dan sudah teprogram. Perintah dan sistem kendali *Smart Home* dapat dilakukan dengan suara, remote kontrol dengan kendali jarak jauh, tepukan tangan, sensor, dan sebagainya.



Gambar 2.1 *Smart Home*

2.2 Sistem Operasi Android

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis Linux yang mencakup sistem operasi, dan aplikasi. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Kemudian untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, sebuah konsorsium dari 34 perusahaan perangkat keras, perangkat lunak, dan telekomunikasi, termasuk Google. Pada saat perilisan perdana Android 5 November 2007, Android Bersama Open Handset Alliance menyatakan

mendukung pengembangan *open source* pada perangkat mobile. Android dipuji sebagai "*Platfrom* mobile pertama yang lengkap, terbuka, dan bebas".

- Lengkap (*Complete Platform*) : Para desainer dapat melakukan pendekatan yang komprehensif ketika mereka sedang mengembangkan platform Android.
- Terbuka (*Open Source Platform*) : Platform android disediakan melalui lisensi open source. Pengembangan dapat dengan bebas untuk mengembangkan aplikasi.
- Free (*Free Platform*) : android adalah aplikasi yang bebas untuk *develope.* Tidak ada lisensi atau biaya royaliti untuk dikembangkan pada *Platform* Android.

Telepon pertama yang menggunakan sistem operasi Android adalah HTC Dream, yang dirilis pada 22 Oktober 2008. Pada penghujung tahun 2010 diperkirakan hampir semua vendor seluler didunia menggunakan Android sebagai sistem operasi. Tabel 2.1 menunjukkan berbagai versi Android[2].

Tabel 2.1 Versi Android

Versi Android	Tanggal Rilis	Nama Kode
1.0	23 September 2008	-
1.1	9 Februari 2009	-
1.5	30 April 2009	Cupcake
1.6	15 September 2009	Donut
2.0/2.1	26 Oktober 2009	Éclair
2.2	20 Mei 2010	Froyo
2.3	6 Desember 2010	Gingerbread
3.0/3.1/3.2	22 Februari 2011	Honeycomb
4.0	19 Oktober 2011	Ice Cream Sandwich
4.1/4.2/4.3	9 Juli 2012	Jelly Bean
4.4	31 Oktober 2013	Kitkat
5.0	3 November 2014	Lolipop
6.0	September 2015	Marshmallow
7.0	-	Noughat

2.3 Pemrograman Berorientasi Objek (Object Oriented Programming)

Pemrograman Berorientasi Objek (*Object Oriented Programming*) merupakan suatu paradigma pemrograman yang secara mendasar berbeda dengan pemrograman prosedural. Pemrograman Berorientasi Objek (selanjutnya disingkat PBO) memiliki fokus pada konsep pemrograman objek yang memiliki atribut dan fungsi yang digunakan untuk melakukan manipulasi terhadap atribut tersebut.

Secara spesifik, pengertian berorientasi objek berarti bahwa mengorganisasi perangkat lunak sebagai kumpulan objek tertentu yang memiliki struktur data dan perilakunya. Hal ini yang membedakan dengan pemrograman konvensional dimana struktur data dan perilaku hanya berhubungan secara terpisah[3]. Terdapat beberapa cara untuk menentukan karakteristik dalam pendekatan berorientasi objek, tetapi secara umum mencakup empat hal, yaitu:

1. Identifikasi
2. Klasifikasi
3. *Polymorphism* (polimorfisme)
4. *Inheritance* (pewarisan)

2.4 UML

Unified Modelling Language (UML) adalah bahasa grafis untuk mendokumentasi, menspesifikasikan, dan membangun sistem perangkat lunak. UML berorientasi objek, menerapkan banyak level abstraksi, tidak bergantung proses pengembangan, tidak bergantung bahasa dan teknologi, pemaduan beberapa notasi di beragam metodologi, usaha bersama dari banyak pihak, serta didukung oleh kakas-kakas yang diintegrasikan lewat XML[4]. Standar UML dikelola oleh OMG (*Object Management Group*).

Tujuan utama dari perancangan UML adalah:

1. Menyediakan bahasa pemodelan yang visual yang ekspesif dan siap pakai untuk mengembangkan dan pertukaran model-model yang berarti.
2. Menyediakan mekanise perluasan dan spesialisasi untuk memperluas konsep-konsep inti.
3. Mendukung spesifikasi independen bahasa pemrograman dan proses pengembangan tertentu.
4. Menyediakan basis formal untuk pemahaman bahasa pemodelan.

5. Mendorong pertumbuhan pasar kakas berorientasi objek.
6. Mendukung konsep-konsep pengembangan level lebih tinggi seperti komponen, kolaborasi, *framework* dan *pattern*.

UML menyediakan diagram-diagram yang sangat kaya dan dapat diperluas sesuai kebutuhan kita. Diagram adalah representasi secara grafis dari elemen-elemen tertentu beserta hubungan-hubungannya. Diagram penting karena diagram menyediakan representasi secara grafis dari sistem (atau bagiannya). Representasi grafis sangat memudahkan pemahaman terhadap sistem.

Diagram UML terbagi menjadi dua diantaranya:

1. Diagram Struktur
   a. Diagram Kelas (*Class Diagram*)

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

   b. Diagram Objek (*Object Diagram*)

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

   c. Diagram Komponen (*Component Diagram*)

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

   d. Diagram *Deployment* (*Deployment Diagram*)

      Diagram ini menunjukan konfigurasi pemrosesan saat jalan dan komponen-komponen yang terdapat di dalamnya. Diagram ini merupakan pandangan statik dari arsitektur.

2. Diagram Perilaku
   a. Diagram *Use Case*

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

b. Diagram Sekuen (*Sequence Diagram*)

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.

c. Diagram Kolaborasi (*Collaboration Diagram*)

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

d. Diagram *Statechart* (*Statechart Diagram*)

Diagram ini adalah *state-machine* diagram, berisi *state*, transisi, kejadian, dan aktivitas. *Statechart* merupakan pandangan 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.

e. Diagram Aktivitas (*Activity Diagram*)

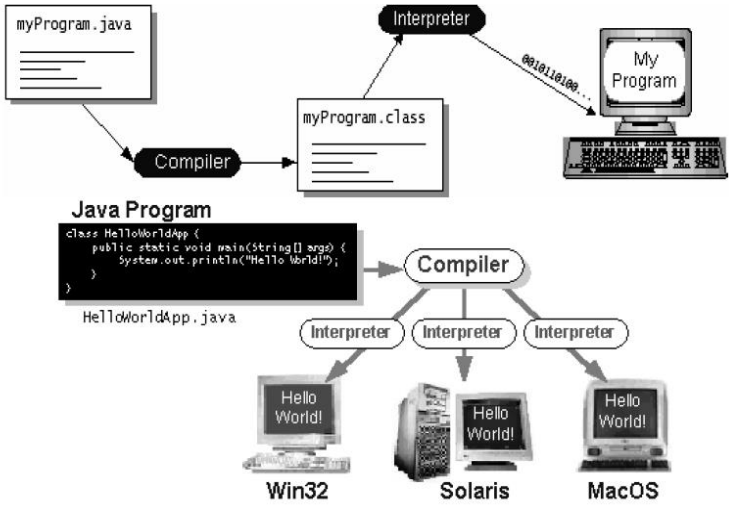
Diagram ini menunjukan aliran aktivitas di sistem. Diagram ini adalah pandangan dinamis terhadap sistem. Diagram ini penting untuk memodelkan fungsi sistem dan menekankan pada aliran kendali di antara objek-objek.

2.5 Pemrograman Java

Saat ini Java merupakan salah satu bahasa pemrograman paling populer untuk segala hal mulai dari permainan pemrograman untuk menciptakan aplikasi *mission-critical.* Tujuannya adalah untuk mempermudah dan menjadi platform independen alternative menggunakan C++. Java adalah program yang sangat populer di lingkungan pendidikan saat ini, hampir di seluruh kampus memiliki satuan ajar pemrograman ini. *Java Virtual Machine* (JVM) atau *Java Runtime Environment* (JRE) merupakan fitur Java yang membuatnya dapat berjalan pada semua *platform* sistem operasi[5]. Apabila membuat Java pada Windows, dapat pula dijalankan pada sistem operasi Linux, Mac OS, dan lainnya.

Hal yang paling mendasar dan harus dikuasai oleh Programer Java adalah memasang library (kelas dalam bentuk biner). Karena java yang dirancang secara orientasi objek, tentunya program yang terdiri atas kesatuan antar objek yang membentuk aplikasi. Sehingga sebuah objek tertentu bisa digunakan dalam

beberapa aplikasi yang berbeda. Objek ini biasanya bersifat umum penggunaannya, misalkan objek kalender, objek *driver* basis data, objek *report* dan lain-lain.



Gambar 2.2 Cara Kerja Pemrograman Java

2.6 Android Studio

Android studio merupakan sebuah *Intregated Development Environment* (IDE) untuk platform Android. Android Studio merupakan pengembangkan dari Eclipse IDE, dan dibuat berdasarkan IDE Java populer, yaitu IntelliJ IDEA. Android Studio merupakan IDE resmi untuk pengembangan aplikasi Android. Sebagai pengembangan dari Eclipse, Android Studio mempunyai banyak fitur-fitur baru dibandingkan dengan Eclipse IDE. Android studio bersifat free dibawah *Apache License* 2.0. Android studio awalnya dimulai dengan versi 0.1. Berbasisikan JetBrainns' Intelej Idea, Studio di desain khususu untuk *Android* Development. Berbasiskan JetBrainns' Intelej Idea, Studio di desain khusus untuk *Android Development.* Android Studio adalah lingkungan pengembangan baru dan terintegrasi penuh, yang baru saja dirilis oleh Google untuk sistem operasi Android. Android Studio dirancang untuk menjadi peralatan baru dalam pengembangan aplikasi dan juga memberi alternatif lain selain Eclipse yang saat ini menjadi IDE yang paling banyak dipakai. Saat memulai proyek baru dengan Android Studio, struktur proyek akan muncul bersama dengan hampir semua berkas yang ada di dalam direktori SDK, peralihan ke sistem manajemen berbasis *Gradle* ini memberikan fleksibilitas yang lebih besar pada proses pembangunannya[6].

2.7 Pengujian Black Box

Pengujian *black box* fokus pada keperluan penelusuran kesalahan fungsional dari *software*[7]. Uji coba *black box* berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya:

1. Fungsi-fungsi yang salah atau hilang.
2. Kesalahan *interface*.
3. Kesalahan dalam struktur data.
4. Kesalahan Performa.
5. Kesalahan inisialisasi dan terminasi