BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Beberapa referensi yang berkaitan dengan objek pembahasan-pembahasan dalam penelitian ini diantaranya sebagai berikut :

Penelitian yang berjudul ”Implementasi *Cloud Storage* menggunakan *OwnCloud* yang *High-Avaibility*” (Ikhwan Ar-Razy, Rinta Kridalukmana, Eko Didik Widianto, :2016). Pada penelitian ini, Metode penelitian yang digunakan adalah *Network Development Life Cycle* (NDLC), karena hasil penelitian berupa desain (perancangan) yang membutuhkan *planning* (perencanaan) yang melibatkan analisa kebutuhan perangkat keras dan perangkat lunak, analisa kebutuhan pengguna serta analisa topologi jaringan tempat penelitian.

Penelitian yang berjudul ” Penerapan Ojs Dalam *Mobile/Android* Yang Diperuntukkan Bagi Pembaca Dan *Author*” (Sudaryanto, Muhammad Ikhwandi : 2017). Pada penelitian ini, *Smartphone* berbasis *android* ini dapat digunakan untuk mengakses OJS yang terpasang pada sebuah *server* dengan syarat harus memiliki konektifitas dengan jaringan *internet*. Beberapa perangkat lunak *browser* dapat juga dijalankan pada *smartphone* ini dengan kondisi layar yang lebih kecil dibandingkan dengan layar komputer, dengan keterbatasan ukuran layar ini maka OJS tersebut dapat juga diakses

menggunakan perangkat lunak yang dibuat dan dibangun secara khusus menggunakan *Android Studio*.

Penelitian yang berjudul ”Perancangan Aplikasi Jurnal *Online* Berbasis *Android*” (Muhammad Juniardi, Dra. Suriati, Marischa Elveny : 2017). Pada penelitian ini, Dengan adanya aplikas ini, aplikasi akan sangat membantu serta mempermudah proses *browser jurnal online* langsung melalui sebuah *smartphone* tanpa menggunakan bantuan dari aplikasi kedua berupa *browser*. Untuk dapat melakukan implementasi sistem pada *smartphone*, sebelumnya persiapan yang harus dilakukan ialah pengguna harus menginstal aplikasi berupa pemindahan *file apk* pada komputer ke dalam *sdcard* pada *smartphone* menggunakan media kabel data.

Penelitian yang berjudul ”Aplikasi *Mobile Driver Online* Berbasis *Android* Untuk Perusahaan Rental Kendaraan” (Surawijaya Surahman1, Eko Budi Setiawan, : 2017). Pada penelitian ini, Adapun *platform* yang digunakan sebagai *pilot project* perangkat lunak ini adalah *android* dengan pertimbangan bahwa berdasarkan laporan kuartal II yang disusun oleh biro marketing bernama Waiwai Marketing, total pengguna *smartphone platform android* adalah 94% dari pangsa pasar di kawasan Asia Tenggara.

Penelitian yang berjudul ”Rancang Bangun Aplikasi *Chat* pada Platform Android dengan Media *Input* berupa *Canvas* dan *Shareable Canvas* untuk Bekerja Dalam Satu *Canvas* secara *Online*” (Luffi Aditya Sandy, Rizky Januar ,dan Ridho Rahman Hariadi : 2017). Pada penelitian ini, aplikasi sudah memenuhi semua fungsionalitas yang diharapkan. Selain itu, Pengujian

yang dilakukan terhadap pengguna menghasilkan kesimpulan dalam berupa rata-rata yang menunjukkan persetujuan pengguna terhadap kesesuaian, kemudahan, dan ketertarikan fitur yang disediakan aplikasi. Nilai rata-rata yang telah melebihi angka 3 atau memiliki arti "setuju" menunjukkan bahwa aplikasi sudah nyaman dan mudah digunakan serta fitur pada aplikasi telah memberikan pengalaman baru kepada pengguna dalam berinteraksi dengan pengguna lain.

II.2. Studi Literatur

II.2.1 Sistem

Sistem adalah merupakan sekumpulan ide yang berhubungan satu dengan lainya namun mempunyai sekumpulan elemen yang saling terkait atau terpadu yang di maksudkan untuk mencapai suatu kesatuan yang terdiri dari dua atau lebih komponen atau subsistem yang berinteraksi untuk mencapai suatu tujuan (Supriyono, H, 2016).

II.2.2. Karakteristik Sistem

Suatu sistem mempunyai karakteristik atau sifat-sifat tertentu, yaitu mempunyai komponen-komponen (*component*), batas sistem (*boundary*), lingkungan luar system (*environments*), penghubung (*interface*), masukan (*input*), keluaran (*output*), pengolah (*process*) dan sasaran (*objectives*) atau tujuan (*goal*). (Supriyono, H, 2016).

Sebuah sistem terdiri dari berbagai unsur yang saling melengkapi dalam mencapai tujuan atau sasaran. Unsur-unsur yang saling melengkapi tersebut terdapat di dalam sistem yang disebut dengan nama subsistem. Subsistem-subsistem tersebut harus selalu berhubungan dan berinteraksi melalui komunikasi yang relevan sehingga sistem dapat bekerja secara efektif.

II.2.3. Data

Data adalah deskripsi tentang benda, kejadian, aktifitas, dan transaksi yang tidak mempunyai makna atau tidak berpengaruh langsung kepada pemakai. Data dapat berupa nilai terformat, *teks*, citra, *audio* dan *video.*

Data yang terformat adalah data dengan suatu format tertentu. Misalnya, data yang menyatakan tanggal atau jam, atau menyatakan nilai mata uang.

Teks adalah sederetan huruf, angka, dan simbol-simbol khususnya (misalnya “+” dan “$”) yang kombinasinya tidak tergantung pada masing-masing item secara individual Contoh teks adalah koran.

Citra *(image)* adalah data dalam bentuk gambar. Citra dapat berupa grafik, foto, hasil rontgen, dan tanda tangan ataupun gambar yang lain.

Audio adalah data dalam bentuk suara. Instrumen musik, suara orang atau suara binatang, gemercik air, detak jantung merupakan beberapa contoh data *audio.*

Video menyatakan data dalam bentuk sejumlah gambar yang bergerak dan bisa saja dilengkapi dengan suara. *Video* dapat digunakan untuk mengabadikan suatu kejadian atau aktivitas. (Supriyono, H, 2016)

II.2.4 Aplikasi

Secara istilah pengertian aplikasi adalah suatu program yang siap untuk digunakan yang dibuat untuk melaksanankan suatu fungsi bagi pengguna jasa aplikasi serta penggunaan aplikasi lain yang dapat digunakan oleh suatu sasaran yang akan dituju. Menurut kamus computer eksekutif, aplikasi mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu tehnik pemrosesan data aplikasi yang biasanya berpacu pada sebuah komputansi yang diinginkan atau diharapkan maupun pemrosesan data yang di harapkan.

Pengertian aplikasi menurut Kamus Besar Bahasa Indonesia, "Aplikasi adalah penerapan dari rancang sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu". (Andi Juansyah, 2015)

II.2.5 *Android Studio*

Android studio adalah IDE (Integrated Development Environment) resmi untuk pengembangan aplikasi Android dan bersifat open source atau gratis. Peluncuran Android Studio ini diumumkan oleh Google pada 16 mei 2013 pada event Google I/O Conference untuk tahun 2013. Sejak saat itu, Android Studio mengantikan Eclipse sebagai IDE resmi untuk mengembangkan aplikasi Android.

Android studio sendiri dikembangkan berdasarkan IntelliJ IDEA yang mirip dengan Eclipse disertai dengan ADT plugin (Android Development Tools). Android studio memiliki fitur :

a. Projek berbasis pada Gradle Build
b. Refactory dan pembenahan bug yang cepat
c. Tools baru yang bernama “Lint” dikalim dapat memonitor kecepatan, kegunaan, serta kompetibelitas aplikasi dengan cepat.
d. Mendukung Proguard And App-signing untuk keamanan.
e. Memiliki GUI aplikasi android lebih mudah
f. Didukung oleh Google Cloud Platfrom untuk setiap aplikasi yang dikembangkan.

II.2.6 *Java Development Kit (JDK)*

Java Development Kit (JDK) adalah sekumpulan perangkat lunak yang dapat kamu gunakan untuk mengembangkan perangkat lunak yang berbasis Java, sedangkan JRE adalah sebuah implementasi dari Java Virtual Machine yang benar-benar digunakan untuk menjalankan program java. Baisanya, setiap JDK berisi satu atau lebih JRE dan berbagai alat pengembangan lain

seperti sumber compiler java, bundling, debuggers, development libraries dan lain sebagainya.

II.2.7 *Wireless*

Wireless adalah *transfer* informasi antara dua atau lebih titik yang tidak terhubung secara fisik. Jarak bisa pendek, seperti beberapa meter untuk *remote control* televisi, atau sejauh ribuan atau bahkan jutaan kilometer untuk ruang-dalam *komunikasi radio.Wireless* pertama pada tahun 1970-an. didahului oleh IBM dengan rancangan teknologi RI, dan perusahaan HP, dengan ISM band yaitu 902-908 Mhz, 2400+2483 dan5725-5850 Mhz, pada tahun 1990 dipasarkan dengan teknik *spektrum* tersebar (SS) pada pita ISM, terlisensi *frekuensi* 18-19 Ghz, pada tahun 1997 *IEEE* membuat *standar WLAN* dengan kode 802.11 dapat bekerja pada *frekuensi* 2.4 Ghz kecepatan 2 Mbps ,pada juli 1999 *IEEE* kembali mengeluarkan kode 802.11b dengan kecepatan 11 Mbps dan pada waktu hampir bersamaan *IEEE* juga mengeluarkan 802.11a menggunakan *frekuensi 5 Ghz*, dan kecepatan data hingga 54Mbps. Tahun 2002 *IEEE* menggabungkan kelebihan 802.11b dan 802.11a yakni 802.11g bekerja pada f*rekuensi 2.4 Ghz* hingga 54Mbps. Yang terkhir tahun 2006 *IEEE* mengeluarkan teknologi 802.11n dikembangkan dengan menggabungkan 802.11b dan 802.11g sehingga menghasilakan peningkatan *throughput* dengan kecepatan 108Mbps (James, 2009). (Dikka Pratama, dkk, 2016)

II.2.8 *Android*

Android adalah sebuah sistem operasi untuk perangkat lunak *mobile* berbasis *linux* yang mencakup sistem operasi, *middleware* dan aplikasi. *Android* menyediakan *platform* terbuka bagi para pengembang untuk

menciptakan aplikasi mereka. Awalnya, *Google Inc*. membeli *Android Inc*. yang merupakan pendatang baru yang membuat peranti lunak untuk *ponsel/smartphone.* Kemudian untuk mengembangkan *Android*, dibentuklah *Open Handset Alliance, konsorsium* dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk *Google, Htc*, *Inte*l, *Motorola*, *Qualcomm*, *T-Mobile*, dan Nvidia. Pada saat perilisan perdana *Android,* 5 November 2007, *Android* bersama *Open Handset Alliance* menyatakan mendukung pengembangan *open source* pada perangkat *mobile*. Di lain pihak, *Google* merilis kode-kode *android* dibawah lisensi *Apache*, sebuah lisensi perangkat lunak dan *open platform* perangkat seluler.

Peta berbasis komputer (digital) lebih serba guna dan dinamis karena bias menunjukkan banyak *view* yang berbeda dengan subjek yang sama. Peta ini juga memungkinkan perubahan skala, animasi gabungan, gambar, suara, dan bisa terhubung ke sumber informasi tambahan melalui *internet.* Peta digital dapat diupdate ke peta tematik baru dan bisa menambahkan detail informasi geografi lainnya. (Irma Agtrisari, 2018).

Tidak hanya menjadi sistem operasi di *smartphone,* saat ini *android* menjadi pesaing utama dari *Apple* pada sistem operasi *Table* PC. Pesatnya pertumbuhan *Android* selain faktor yang disebutkan diatas adalah karena *android* itu sendiri adalah *platform* sangat lengkap baik itu sistem operasinya, aplikasi dan *Tool Development*.

Market aplikasi *android* serta dukungan yang sangat tinggi dari komunitas *Open Source* di dunia, sehingga *android* terus berkembang pesat dari segi teknologi maupun dari segi jumlah *device* yang ada didunia.

II.2.9 Fundamental Aplikasi

Aplikasi *android* ditulis dalam bahasa pemrograman *java, kode java* dikompilasi bersama data *file resource* yang dibutuhkan oleh aplikasi dimana prosesnya dipaket oleh *tools* yang dinamakan "*apt tools*" ke dalam paket *android* sehingga menghasilkan *file* dengan ekstensi apk. *File apk* itulah yang sebenarnya disebut dengan aplikasi yang dapat diinstal di perangkat *mobile* nantinya. Ada empat jenis komponen pada aplikasi *android* yaitu:

a. *Activities*

Suatu *activity* akan menyajikan *user interface (UI)* kepada pengguna, sehingga pengguna dapat melakukan interaksi. Sebuah aplikasi android bisa jadi hanya memiliki satu *activity*, tetapi umumnya aplikasi memiliki banyak *activity* tergantung pada tujuan aplikasi dan desain dari aplikasi tersebut. Satu *activity* biasanya akan dipakai untuk menampilkan aplikasi atau yang bertindak sebagai *user interface* (UI) saat aplikasi diperlihatkan kepada *user.* Untuk pindah dari satu *activity* ke *activity* lain dapat dilakukan dengan satu *even* misalnya *click* tombol, memilih opsi atau menggunakan *triggers* tertentu. Secara hirarki sebuah *window activity* dinyatakan dengan *method Activity*. *SetContentView*(). *Content View* adalah objek yang berada pada *root hirarki.*

b. *Service*

Service tidak memiliki *visual user interface (UI),* tetapi *service* berjalan secara *background,* sebagai contoh dalam memainkan *music, service* mungkin memainkan music atau mengambil data dari jaringan, tetapi setiap *service* haruslah berada dalam kelas induknya. Misalnya media *player* sedang memutar lagu dari *list* yang ada. Aplikasi ini akan memiliki dua atau lebih *activity* yang memungkinkan user untuk memilih lagu atau menulis sms sambil *player* sedang jalan untuk menjaga *music* tetapi dijalankan, *activity* player dapat menjalankan *service* untuk membuat aplikasi tetap berjalan. *Service* dijalankan pada *thread* utama dari proses aplikasi.

c. *Broadcast Receiver*

Broadcast Receiver berfungsi menerima dan bereaksi untuk menyampaikan notifikasi. Contoh *broadcast* seperti notifikasi zona waktu berubah, baterai *low*, gambar telah selesai diambil oleh kamera, atau pengubah referensi bahasa yang digunakan. Aplikasi juga dapat menginisiasi broadcast misalnya memberikan informasi pada aplikasi lain bahwa ada data yang telah didownload ke perangkat dan siap untuk digunakan *Broadcast Receiver* tidak memiliki *user interface* (UI) tetapi memiliki sebuah *activity* untuk merespon informasi yang mereka terima atau mungkin menggunakan *notification* manager untuk memberitahu kepada pengguna seperti lampu latar atau *viberating* (getaran) perangkat dan lain sebagainya.

d. *Content Provider*

Content *Provider* membuat kumpulan aplikasi data secara spesifik sehingga bisa digunakan oleh aplikasi lain. Data disimpan dalam *file* sistem seperti *database SQLite*.*Content Provider* menyediakan cara untuk mengakses data yang dibutuhkan oleh suatu *activity*, misalnya ketika menggunakan aplikasi yang membutuhkan peta (MAP) atau aplikasi yang membutuhkan untuk mengkases data kontak dan navigasi, maka disinilah fungsi *content provider.*

II.2.10 Metode Pengembangan *(Waterfall Model)*

Metode pengembangan sistem sekuensiallinier atau yang sering disebutdengan siklus kehidupan klasik atau model air terjun *(waterfall model)* memberikan sebuah pendekatan pengembangan sistem yang sistematik dan sekuensia, dimulai pada fase perencanaan sistem, analisis, desain, kode , pengujian dan pemeliharaan.

1. Perencanaan atau rekayasa dan pemodelan sistem

   Pada fase ini dilakukan identifikasi sistem, studi kebutuhan pengguna, dan studi kelayakan sistem baik secara teknis maupun teknologi serta penjadwalan pengembangan sistem.

2. Analisis kebutuhan perangkat lunak

   Pada fase ini pengumpulan kebutuhan diidentifikasi dan difokuskan pada sistem yang akan dibangun meliputi identifikasi domain informasi, tingkah laku sistem, untuk kerja dan antar muka sistem.

Kebutuhan untuk sistem didokumentasikan dan dikonsultasikan lagi bagi pengguna.

3. Desain

   Fase ini difokuskan pada proses desain struktur data, arsitektur sistem, representasi *interface* dan algoritma program.

4. Kode

   Setelah proses desain selesai maka hasilnya harus diterjemahkan ke dalam bentuk program komputer yang kemudian menghasilkan suatu sistem

5. Pengujian

   Pengujian dilakukan untuk menemukan kesalahan-kesalahan yang memungkinkan terjadi pada proses pengkodean serta memastikan bahwa input yang dibatasi memberikan hail yang sesuai dengan kebutuhan.

6. Pemeliharaan dan Pengoperasian

   Ditandai dengan penyerahan perangkat lunak kepada pemesannya untuk dioperasikan. Dalam masa operasional, perangkat lunak masih memungkinkan untuk terjadi sesuatu kesalahan atau kegagalan dalam menjalankan fungsi, perangkat lunak tersebut masih membutuhkan proses (*maintenance*) dari waktu ke waktu.

II.2.11 *Unified Modeling Language* (UML)

Dalam konteks *UML*, konseptualisme dilakukan dengan pembuatan *use case diagram* yang sesungguhnya merupakan deskripsi peringkat tinggi bagaimana perungkat lunak. (aplikasi) akan digunakan oleh penggunaanya. Selanjutnya, *use case diagram* tidak hanya sangat penting pada tahap analisis, tetapi juga sangat penting untuk perancangan (*design*), untuk mencari (mencoba menemukan) kelas – kelas yang terlibat dalalm aplikasi, dan untuk melakukan pengujiaan atau (*testing*).

Saat kita akan mengembangkan use case diagram, hal yang pertma kali kita lakukan adalah mengenal *actor* untuk sistem/aplikasi ysng sedang kita kembangkan. Dalam hal ini ada beberapa karakteristik untuk para *actor*, yaitu *actor* ada diluar sistem yang sedang kita kembangkan dan *actor* berinteraksi dengan sistem yang sedang kita kembangkan. Maksud tertentu, biasanya antara lain untuk :

1. Merancang perangkat lunak.
2. Sarana komunikasi antara perangkat lunak dengan bisnis.
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasikan sistem yang ada, proses-proses dan organisasinya.

UML telah diaplikasikan dalam investasi perbankan,lembaga kesehatan, departemen pertahanan, sistem terdistribusi, sistem pendukung alat kerja, *retail*, sales, dan *supplier.*

Blok pembangunan utama *UML* adalah diagram. Beberapa diagram ada yang rinci (jenis *timing diagram*) dan lainya ada yang bersifat umum (misalnya diagram kelas). Para pengembang sistem berorentasikan objek menggunakan bahasa model untuk menggambarkan, membangun dan mendokumentasikan sistem yang mereka rancang. *UML* memungkinkan para anggota team untuk bekerja sama dalam mengaplikasikan beragam sistem. Intinya, *UML* merupakan alat komunikasi yang konsisten dalam mensupport para pengembang sistem saat ini. Sebagai perancang sistem mau tidak mau pasti menjumpai *UML*, baik kita sendiri yang membuat sekedar membaca diagram *UML* buatan orang lain (Adi Nugroho, 2009 : 7).

II.2.12 Diagram-*Diagram UML*

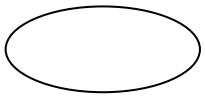
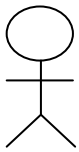
UML adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak. UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem.

Alat bantu yang digunakan dalam perancangan, berorientasi objek berbasiskan UML adalah sebagai berikut :

1. *Use Case Diagram*

   Use Case diagram merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use Case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Simbol-simbol yang digunakan dalam *use case* diagram, yaitu :

Tabel II.1. *Simbol Use Case*

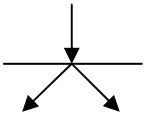
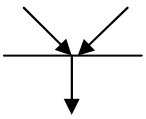
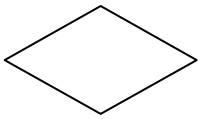
Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit bertukar pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja diawal nama *use case*.
	Aktor adalah abstraction dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan actor, harus ditentukan pembagian tenaga dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengindikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengindikasikan bila aktor berinteraksi secara pasif dengan sistem.
	Include, merupakan didalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	Extend, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber : Panduan Praktis Diagram UML, 2014)

2. Diagram Aktivitas (*Activity Diagram*)

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem ataau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram*, yaitu :

Tabel II.2. Simbol *Activity Diagram*

Gambar	Keterangan
	*Start point,* diletakkan pada pojik kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activities*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu.
	*Join*, (penggabungan) atau rake, digunakan untk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true*, *false*.
	*Swimlane,* pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

(Sumber : Panduan Praktis Diagram UML, 2014)

3. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas didalam model desain dari suatu sistem. *Class diagram* juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. Hubungan antar kelas mempunyaai keterangan yang disebut dengan *multiplicity* atau kardinaliti.

Tabel II.3. *Multiplicity Class Diagram*

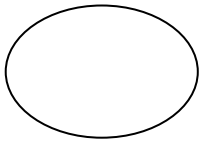
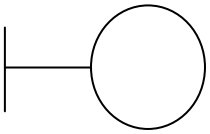
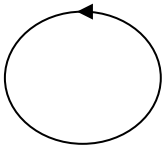
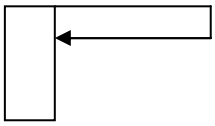
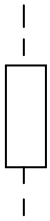
Multiplicity	Penjelasan
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Panduan Praktis Diagram UML, 2014)

4. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram* yaitu :

Tabel II.4. Simbol *Sequence Diagram*

Gambar	Keterangan
	*Entity Class,* merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih faktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control Class*, objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	Message, simbol mengirim pesan antar *class*.
	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation*, *activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Panduan Praktis Diagram UML, 2014)

II.2.13 *Android SDK*

Android SDK adalah *tools API (Application Programming Interface)* yang diperlukan untuk mulai mengembangkan aplikasi pada *platform android* menggunakan bahasa pemrograman *Java. Android* merupakan *subset* perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang di *release* oleh *Google.* Saat ini disediakan *Android SDK (Software Development Kit)* sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada *platform Android* menggunakan bahasa pemrograman *Java*. Sebagai *platform* aplikasi-netral, *android* member anda kesempatan untuk membuat aplikasi yang kita butuhkan yang bukan merupakan aplikasi bawaan *Handphone* atau *Smartphone*. Beberapa fitur-fitur *android* yang paling penting adalah :

a. *Framework* : Aplikasi yang mendukung pengganti komponen dan *reusable.*

b. *Dalvik Virtual Machine* dioptimalkan untuk perangkat *mobile.*

c. *Integrated Browser* berdasarkan *engine open source WebKit*.

d. Grafis yang dioptimalkan dan didukung oleh libraries grafis 2D, grafis 3D

e. berdasarkan spesifikasi opengl ES 1,0 (*Opsional Ekselerasi hardware*)

f. Media *Support* yang mendukung *audio, video,* dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PING, GIF), GSM *Telephony* (tergantung hardware)

g. *Bluetooth, EDGE, 3G,* dan *WiFi* (tergantung *hardware)*

h. Kamera, GPS, Kompas, dan *Accelerometer* (tergantung hardware)

i. Lingkungan Development yang lengkap dan termasuk perangkat *emulator*, *tools* untuk *debugging*, profil dan kinerja memori, dan *plugin* untuk IDE.