BAB II

TINJAUAN PUSTAKA

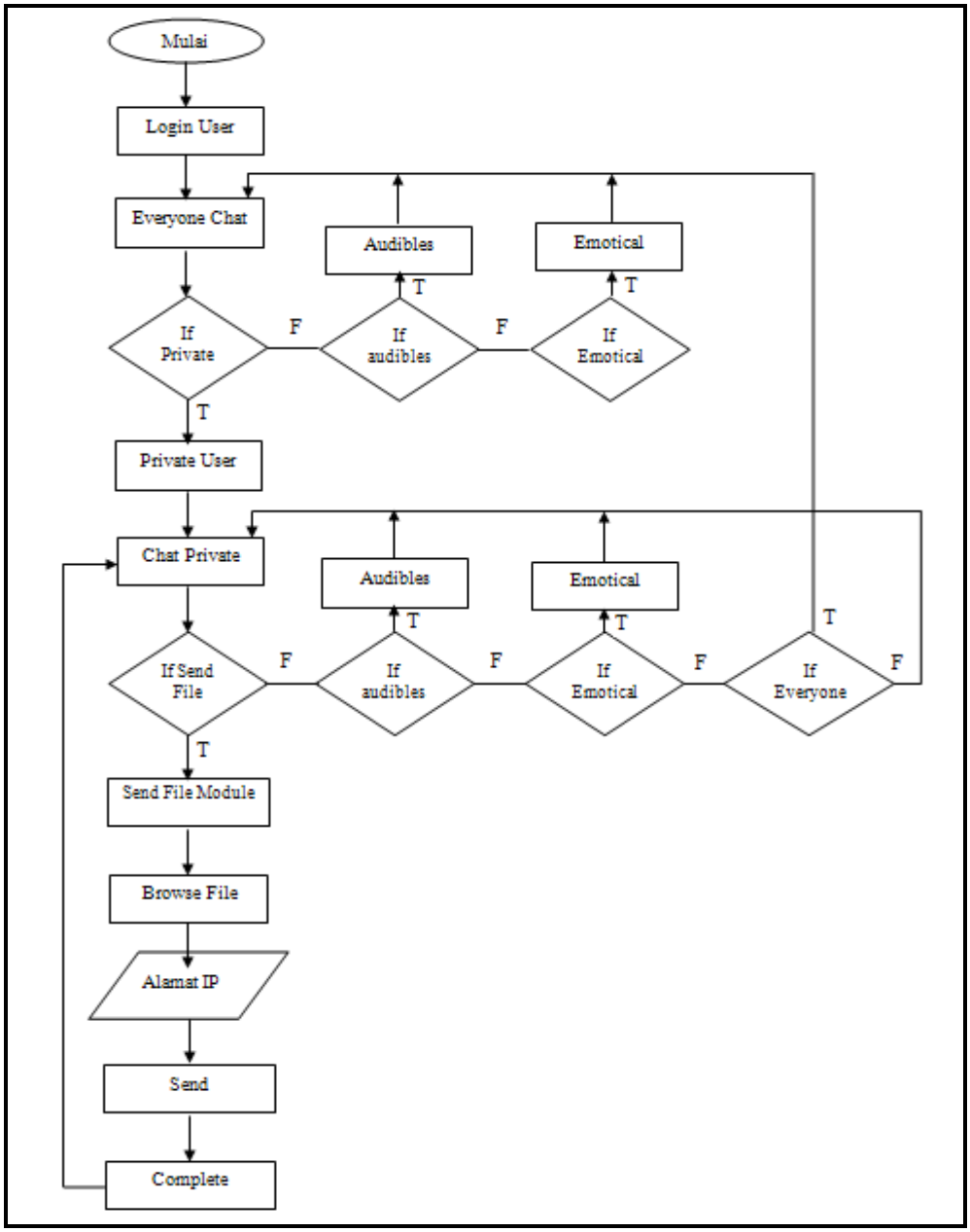
II.1. Aplikasi

Aplikasi adalah suatu *sub* kelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Contoh utama aplikasi adalah pengolah kata, lembar kerja, memanipulasi foto dan pemutar media. Beberapa aplikasi yang digabung bersama menjadi suatu paket disebut sebagai suatu paket atau *suite* aplikasi (*application suite*). Contohnya adalah *Microsoft Office* dan *OpenOffice.org*, yang menggabungkan suatu aplikasi pengolah kata, lembar kerja dan beberapa aplikasi lainnya. Aplikasi-aplikasi dalam suatu paket biasanya memiliki antarmuka pengguna yang memiliki kesamaan sehingga memudahkan pengguna untuk mempelajari dan menggunakan tiap aplikasi. Sering kali, mereka memiliki kemampuan untuk saling berinteraksi satu sama lain sehingga menguntungkan pengguna. Contohnya, suatu lembar kerja dapat dibenamkan dalam suatu dokumen pengolah kata walaupun dibuat pada aplikasi lembar kerja yang terpisah. (*Dahlan Abdullah ; 2013 : 152*)

Jenis - jenis *Software* Aplikasi :

1. *Software* aplikasi hiburan, contohnya yaitu *winamp* untuk mendengarkan musik, *games* dan sebagainya untuk hiburan.
2. *Software* aplikasi pendidikan yaitu *software* yang digunakan untuk mempelajari tentang pendidikan atau pengetahuan.

3. *Software* aplikasi bisnis yaitu *software* yang digunakan untuk aplikasi bisnis
4. *Software* aplikasi khusus
5. *Software* aplikasi untuk produtivitas kerja.



Gambar II.1. ***Desain Struktur Menu Aplikasi***
Sumber : Roni Setiawan ; 2010

II.2. *Video Call*

Video Call adalah salah satu media komunikasi yang memberikan kemudahan pengguna untuk dapat melihat wajah lawan bicara dalam *chatting* dengan menggunakan kamera yang terdapat di *notebook* atau perangkat komputer. *Video Call* juga membutuhkan jaringan *internet* sebagai media transmisinya. Salah satu aplikasi untuk *video call* adalah *skype*. *Skype* adalah *software* aplikasi komunikasi suara berbasis IP dengan teknologi P2P (*peer to peer*) melalui *internet* antara sesama pengguna *Skype*. Tulisan ini membahas tentang pengaruh lamanya waktu *chatting* antara *client* A dengan *client* B berdasarkan pengujian yang dilakukan. Pengujian dilakukan dengan pengambilan data menggunakan *software wireshark*. Parameter QoS yang dianalisis berupa *delay, packet loss* dan *throughput.* Hasil analisa data dari percobaan yang dilakukan menunjukkan bahwa pada saat melakukan *video call* diperoleh *delay* rata-rata sebesar 0,72 sec, *packet loss* yang bernilai 0 %, sedangkan nilai *throughput*nya akan semakin turun seiring dengan lamanya waktu *chatting.* (*Rayhan Yuvandra ; 2013 : 95*)

II.3. *Android*

Android adalah sistem operasi untuk telepon seluler yang berbasis *Linux*, yang mencakup *system* operasi, *middleware* dan aplikasi. *Android* tidak terikat ke satu merek telepon seluler. *Android* menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri hingga dapat digunakan oleh berbagai peranti *mobile*. Beberapa fitur utama dari *Android* antara lain *WiFi hotspot, Multi-touch, Multitasking, GPS, support java*, mendukung banyak jaringan (GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, *Bluetooth*, Wi-Fi, LTE,

and WiMAX) dan juga kemampuan dasar telepon seluler pada umumnya. (*Alicia Sinsuw ; 2013 : 2*)

II.3.1. Sistem Operasi *Android*

Adapun sistem operasi *android* menurut *Alicia Sinsum* pada tahun 2013 adalah sebagai berikut :

1. *Android OS*

   Android OS adalah sistem operasi yang berbasis *Linux*, sistem operasi *open source.* Selain *Android Software Development Kit (SDK)* untuk pengembangan aplikasi, *android* juga tersedia bebas dalam bentuk sistem operasi. Hal ini yang menyebabkan *vendor - vendor smartphone* begitu berminat untuk memproduksi *smartphone* dan komputer *tablet* berbasis *Android*. *Android OS* dapat diunduh dari situs resmi google, yaitu http://www.code.google.com. Saat ini *Android OS* sudah menyebar bukan hanya di *smartphone* saja, tetapi juga di komputer *tablet*.

2. *Android SDK (Software Development Kit)*

   Android SDK adalah *tools* API *(*Aplication Programming Interface*)* yang diperlukan untuk mengembangkan aplikasi pada *platform Android* menggunakan bahasa pemrograman *Java*. Beberapa fitur-fitur *Android* yang paling penting adalah mesin *Virtual Dalvik* yang dioptimalkan untuk perangkat *mobile*, *integrated browser* berdasarkan *engine open source WebKit*, Grafis yang dioptimalkan dan didukung oleh *libraries* grafis 2D, grafis 3D berdasarkan spesifikasi *opengl* ES 1.0 (Opsional akselerasi perangkat keras), kemudian *SQLite* untuk penyimpanan data (*database*).

Fitur - fitur *android* lainnya termasuk media yang mendukung *audio, video*, dan gambar, juga ada fitur *bluetooth*, EDGE, 3G dan *WiFi*, dengan fitur kamera, GPS, dan kompas. Selanjutnya fitur yang juga turut disediakan adalah lingkungan *Development* yang lengkap dan kaya termasuk perangkat emulator, *tools* untuk debugging, profil dan kinerja memori, dan *plugin* untuk IDE *Eclipse*.

3. *AVD (Android Virtual Device)*

   Android Virtual Device merupakan emulator untuk menjalankan aplikasi *android*. Setiap AVD terdiri dari sebuah profil perangkat keras yang dapat mengatur pilihan untuk menentukan fitur *hardware* emulator. Misalnya, menentukan apakah menggunakan perangkat kamera, apakah menggunakan *keyboard* QWERTY fisik atau tidak, berapa banyak memori internal, dan lain-lain. AVD juga memiliki sebuah pemetaan versi *Android*, maksudnya kita menentukan versi dari *platform Android* akan berjalan pada emulator. Pilihan lain dari AVD, misalnya menentukan *skin* yang kita ingin gunakan pada emulator, yang memungkinkan untuk menentukan dimensi layar, tampilan, dan sebagainya. Kita juga dapat menentukan SD *Card virtual* untuk digunakan dengan di emulator.

II.4. Jaringan

Pengertian jaringan komputer merupakan sekumpulan komputer serta perangkat - perangkat lain pendukung komputer yang saling terhubung dalam suatu kesatuan. Media jaringan komputer dapat melalui kabel-kabel atau tanpa kabel sehingga memungkinkan pengguna jaringan komputer dapat saling

melakukan pertukaran informasi seperti dokumen dan data. Dapat juga melakukan pencetakan pada printer yang sama dan bersama-sama memakai perangkat keras dan perangkat lunak yang terhubung dengan jaringan. (*Choirul Muallifah ; 2013 : 2*)

II.4.1. Jenis-Jenis Jaringan

Berdasarkan jangkauan area atau lokasi jaringan menurut *Choirul Muallifah* tahun 2013 dibedakan menjadi 3 jenis, yaitu :

1. LAN (*Local Area Network*)

   Merupakan jaringan lokal yang dibuat pada area tertutup. Misalkan dalam satu gedung atau dalam satu ruangan. LAN biasa digunakan untuk jaringan kecil yang menggunakan *resource* bersama. Seperti penggunaan *printer* secara bersama. LAN dapat menggunakan media komunikasi seperti kabel dan *wireless.*

2. MAN (*Metropolitan Area Network*)

   Merupakan jaringan antara LAN satu dengan LAN lain yang dipisahkan daerah lokasi yang cukup jauh. Contoh penggunaan MAN adalah hubungan antara kantor pusat dengan kantor cabang yang ada di daerah-daerah. Dapat dikatakan MAN merupakan pengembangan dari LAN.

3. WAN (Wide Area Network)

   Merupakan jaringan yang cakupannya lebih luas dari pada MAN. Cakupan WAN meliputi satu kawasan, satu negara, satu pulau, bahkan satu benua. Metode yang digunakan WAN hampir sama dengan LAN dan MAN.

II.4.2. Manfaat dan Keuntungan Jaringan

Adapun manfaat dan keuntungan membangun jaringan menurut *Choirul Muallifah* tahun 2013 adalah sebagai berikut :

1. Manfaat Jaringan Komputer:
    a. *Sharing resources*
    b. Media komunikasi
    c. Integrasi data
    d. Pengembangan dan pemeliharaan
    e. Keamanan data
    f. Sumber daya lebih efisien dan informasi terkini
2. Keuntungan Membangun Jaringan Komputer Yaitu:
    a. Dapat berbagi peralatan (*peripheral*) dan penggunaanya, seperti printer, *harddisk*, modem.
    b. Memudahkan bertukar data diantara pengguna komputer tanpa harus menggunakan *disket atau flashdisk*.
    c. Kita dapat menggunakan program - program yang ada di komputer pusat.
    d. Bisa mengirim dan menerima *email* dari *internet* dan mencari informasi lain melalui fasilitas *internet*.

II.5. *Wifi*

Awalnya *wifi* ditujukan untuk pengunaan perangkat nirkabel dan Jaringan Area Lokal (LAN), namun saat ini lebih banyak digunakan untuk mengakses *internet*. Hal ini memungkinan seseorang dengan komputer dengan kartu nirkabel (*wireless card*) atau *personal digital assistant* (PDA) untuk terhubung dengan

internet dengan menggunakan titik akses (*hotspot*) terdekat. *Wifi* dirancang berdasarkan spesifikasi IEEE 802.11. Sekarang ini ada empat variasi dari 802.11, yaitu: 802.11a, 802.11b, 802.11g, and 802.11n. Spesifikasi b merupakan produk pertama *wifi*. Variasi g dan n merupakan salah satu produk yang memiliki penjualan terbanyak pada 2005. (*Sony Bahagia Sinaga ; 2012 : 47*) Spesifikasi *wifi* yaitu :

Tabel II.1. Spesifikasi *Wifi*

Spesifikasi	Kecepatan	Frekuensi Band	Cocok dengan
802.11b	11 Mb/s	2.4 GHz	b
802.11a	54 Mb/s	5 GHz	A
802.11g	54 Mb/s	2.4 GHz	b,g
802.11n	100 Mb/s	2.4 GHz	b,g,n

Versi *wifi* yang paling luas dalam pasaran AS sekarang ini (berdasarkan dalam IEEE 802.11b/g) beroperasi pada 2.400 MHz sampai 2.483,50 MHz. Dengan begitu mengijinkan operasi dalam 11 channel (masing-masing 5 MHz), berpusat di frekuensi berikut :

1. Channel 1 - 2,412 MHz
2. Channel 2 - 2,417 MHz
3. Channel 3 - 2,422 MHz
4. Channel 4 - 2,427 MHz
5. Channel 5 - 2,432 MHz
6. Channel 6 - 2,437 MHz
7. Channel 7 - 2,442 MHz
8. Channel 8 - 2,447 MHz

9. Channel 9 - 2,452 MHz
10. Channel 10 - 2,457 MHz
11. Channel 11 - 2,462 MHz

Secara teknis operasional, *wifi* merupakan salah satu varian teknologi komunikasi dan informasi yang bekerja pada jaringan dan perangkat WLAN (*wireless local area network*). Dengan kata lain, *wifi* adalah sertifikasi merek dagang yang diberikan pabrikan kepada perangkat telekomunikasi (*internet*) yang bekerja di jaringan WLAN dan sudah memenuhi kualitas kapasitas interoperasi yang dipersyaratkan. (*Sony Bahagia Sinaga ; 2012 : 47*)

Teknologi *internet* berbasis *wifi* dibuat dan dikembangkan sekelompok insinyur Amerika Serikat yang bekerja pada *Institute of Electrical and Electronis Engineers* (IEEE) berdasarkan standar teknis perangkat bernomor 802.11b, 802.11a dan 802.16. Perangkat *wifi* sebenarnya tidak hanya mampu bekerja di jaringan WLAN, tetapi juga di jaringan *Wireless Metropolitan Area Network* (WMAN). Karena perangkat dengan standar teknis 802.11b diperuntukkan bagi perangkat WLAN yang digunakan di frekuensi 2,4 GHz atau yang lazim disebut frekuensi ISM (*Industrial, Scientific* dan *Medical*). Sedang untuk perangkat yang berstandar teknis 802.11a dan 802.16 diperuntukkan bagi perangkat WMAN atau juga disebut Wi-Max, yang bekerja di sekitar pita frekuensi 5 GHz. (*Sony Bahagia Sinaga ; 2012 : 47*)

II.6. *Java*

Java memiliki cara kerja yang unik dibandingkan dengan bahasa perograman lainya yaitu bahasa perograman *java* bekerja mengunakan *interpreter* dan juga *compiler* dalam proses pembuatan program, *Interpreter java* dikenal sebagai perograman *bytecode* yaitu dengan cara kerja mengubah paket *class* pada *java* dengan *extensi*. *Java* menjadi *.class*, hal ini dikenal sebagai *class bytecode*, yaitunya *class* yang dihasilkan agar program dapat dijalankan pada semua jenis perangkat dan juga *platform*, sehingga program *java* cukup ditulis sekali namun mampu bekerja pada jenis lingkungan yang berbeda. (*Defni, Indri Rahmayun* ; 2014 : 64)

II.7. Data *Flow* Diagram (DFD)

Data *Flow* Diagram (DFD) adalah alat pembuatan model yang memungkinkan profesional sistem untuk menggambarkan sistem sebagai suatu jaringan proses fungsional yang dihubungkan satu sama lain dengan alur data, baik secara manual maupun komputerisasi. DFD ini sering disebut juga dengan nama Bubble chart, Bubble diagram, model proses, diagram alur kerja, atau model fungsi.

DFD ini adalah salah satu alat pembuatan model yang sering digunakan, khususnya bila fungsi-fungsi sistem merupakan bagian yang lebih penting dan kompleks dari pada data yang dimanipulasi oleh sistem. Dengan kata lain, DFD adalah alat pembuatan model yang memberikan penekanan hanya pada fungsi sistem.

DFD ini merupakan alat perancangan system yang berorientasi pada alur data dengan konsep dekomposisi dapat digunakan untuk penggambaran analisa maupun rancangan ,sistem yang mudah dikomunikasikan oleh profesional sistem kepada pemakai maupun pembuat program. (*Dahlan Abdullah ; 2013 : 154*)

II.8. UML (*Unified Modelling Language*)

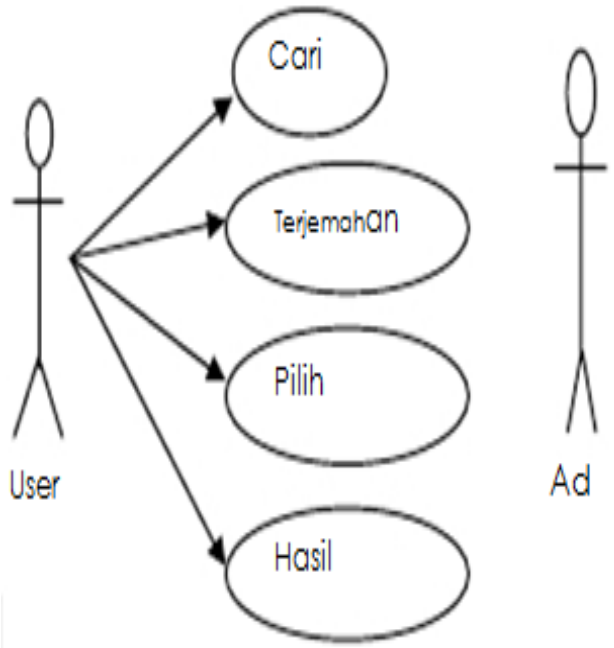
Unified Modelling Language (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan peranti lunak dalam bahasa-bahasa berorientasi objek seperti C++, *Java*, C# atau *VB.NET*. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C. Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan *syntax*/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML *syntax* mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: *Grady Booch OOD* (*Object-Oriented Design*), Jim Rumbaugh OMT (*Object Modeling Technique*), dan Ivar Jacobson OOSE (*Object-Oriented Software Engineering*). Sejarah UML sendiri cukup

panjang. Sampai era tahun 1990 seperti kita ketahui puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. Diantaranya adalah: *metodologi booch, metodologi coad, metodologi OOSE, metodologi OMT, metodologi shlaer-mellor, metodologi wirfs-brock*, dsb. Masa itu terkenal dengan masa perang metodologi (*method war*) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasi sendiri-sendiri, yang mengakibatkan timbul masalah baru apabila kita bekerjasama dengan group/perusahaan lain yang menggunakan metodologi yang berlainan. Dimulai pada bulan Oktober 1994 *Booch, Rumbaugh dan Jacobson*, yang merupakan tiga tokoh yang boleh dikata metodologinya banyak digunakan mempelopori usaha untuk penyatuan metodologi pendesainan berorientasi objek. Pada tahun 1995 dirilis *draft* pertama dari UML (versi 0.8). Sejak tahun 1996 pengembangan tersebut dikoordinasikan oleh *Object Management Group* (OMG – http://www.omg.org). Tahun 1997 UML versi 1.1 muncul, dan saat ini versi terbaru adalah versi 1.5 yang dirilis bulan Maret 2003. *Booch, Rumbaugh* dan *Jacobson* menyusun tiga buku serial tentang UML pada tahun 1999. Sejak saat itulah UML telah menjelma menjadi standar bahasa pemodelan untuk aplikasi berorientasi objek. *(Yuni Sugiarti ; 2013 : 33)*

Dalam pembuatan skripsi ini penulis menggunakan diagram *Use Case* yang terdapat di dalam UML. Adapun maksud dari *Use Case* Diagram diterangkan dibawah ini.

1. *Use Case Diagram*

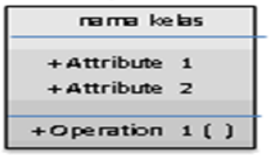
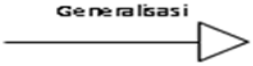
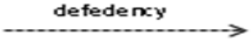
Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case diagram* dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem. Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain. *(Yuni Sugiarti ; 2013 : 41)*



Gambar II.2. *Use Case* Diagram
Sumber : (*Junaedi Siregar ; 2013 : 76*)

2. *Class* Diagram

Diagram kelas atau *class* diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Berikut adalah simbol-simbol pada diagram kelas :

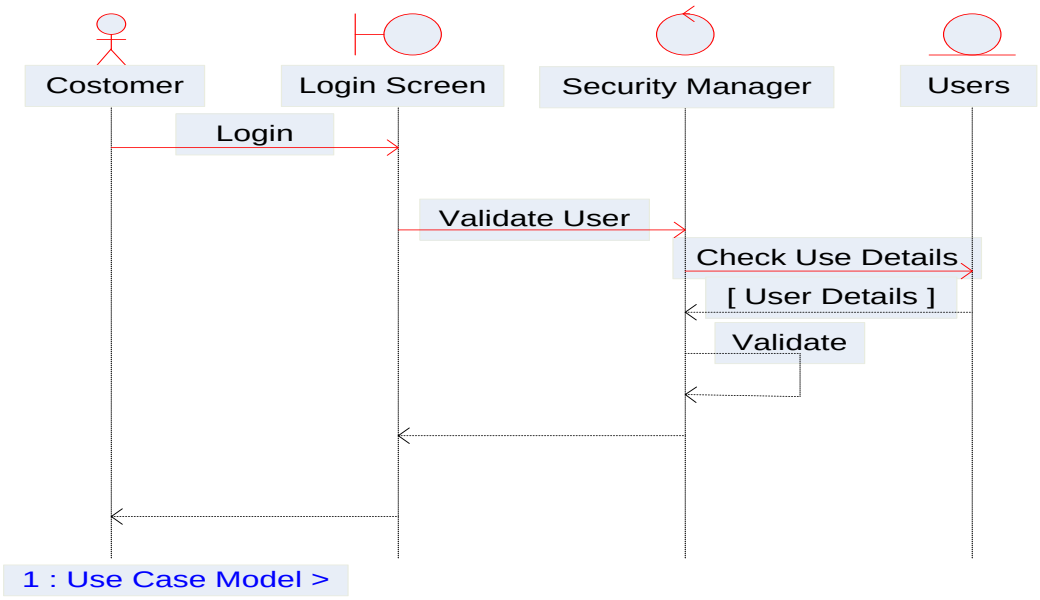
Simbol	Deskripsi
Package 	Package merupakan sebuah bungkusan dari satu atau lebih kelas
Operasi 	Kelas pada struktur sistem
Antarmuka / interface 	sama dengan konsep interface dalam pemrograman berorientasi objek
Asosiasi 	relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan multiplicity
Asosiasi berarah/directed asosiasi	relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan multiplicity
Generalisasi 	relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus)
Kebergantungan / defedency 	relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi 	relasi antar kelas dengan makna semua-bagian (whole-part)

Gambar II.3. *Class* Diagram
Sumber : (Yuni Sugiarti ; 2013 : 59)

3. *Sequence Diagram*

Diagram *Sequence* menggambarkan kelakuan/prilaku objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram *sequence* maka harus diketahui objek - objek yang terlibat dalam sebuah *use case* beserta metode - metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

Banyaknya diagram *sequence* yang harus digambar adalah sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram *sequence* sehingga semakin banyak *use case* yang didefinisikan maka diagram *sequence* yang harus dibuat juga semakin banyak.



Gambar II.4. *Contoh Sequence Diagram*
Sumber : (Yuni Sugiarti ; 2013 : 63)

4. *Activity Diagram*

Activity diagram menggambarkan berbagai alur aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alur berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

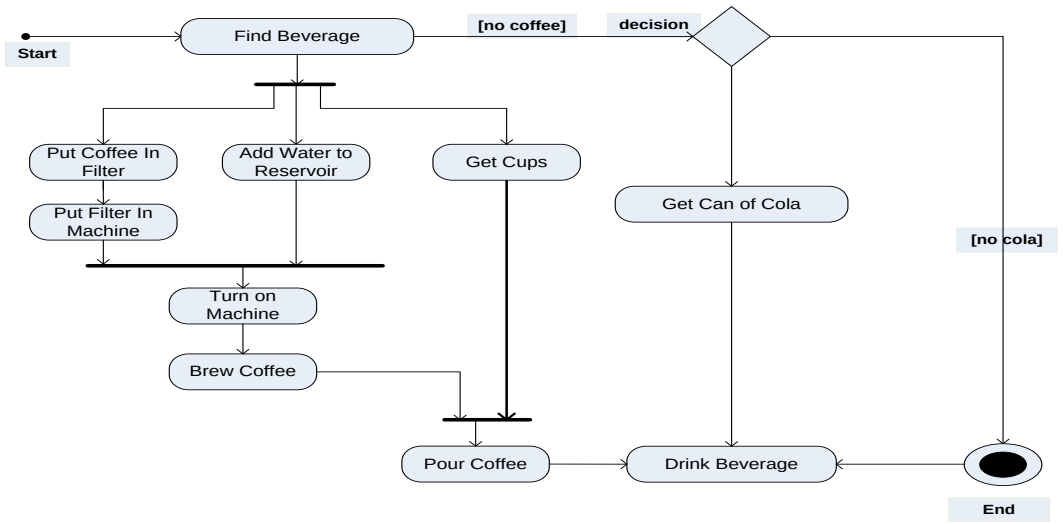
Activity diagram merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak

menggambarkan *behaviour* internal sebuah sistem (dan interaksi antar *subsistem*) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur - jalur aktivitas dari *level* atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas.

Sama seperti *state*, standar UML menggunakan segi empat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal.

Activity diagram dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.



Gambar II.5. *Activify* Diagram
Sumber : (Yuni Sugiarti ; 2013 : 76)