BAB II

LANDASAN TEORI

2.1 Tinjauan Teori Umum

2.1.1 Sistem Informasi

Sistem informasi adalah suatu sistem yang ada di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi yang bersifat manajerial dan kegiatan strategi dari suatu organisasi dan pihak luar tertentu dengan laporan yang diperlukan (Zakiyudin, 2013: 13).

Menurut Alter dalam Kadir (2014: 9) Sistem informasi adalah kombinasi atau prosedur kerja, informasi, orang, dan teknologi informasi yang diorganisasikan untuk mencapai tujuan. Sedangkan menurut Gelinas *et al.*, dalam Kadir (2014: 9) Sistem informasi adalah suatu sistem buatan manusia yang secara umum terdiri atas sekumpulan komponen berbasis komputer dan manual yang dibuat untuk menghimpun, menyimpan, dan mengelola data serta menyediakan informasi keluaran kepada para pemakai.

Sedangkan menurut Wilkinson dalam Kadir (2014: 9).Sistem informasi adalah kerangka kerja yang mengoordinasikan sumber daya (manusia, computer) untuk mengubah masukan *(input)* menjadi keluaran atau informasi, guna mencapai sasaran - sasaran perusahaan.

Dari pendapat beberapa ahli diatas, maka penulis menyimpulkan bahwa sistem informasi mencakup sejumlah komponen (manusia, teknologi informasi, dan prosedur kerja).Ada sesuatu yang diproses (data menjadi informasi), dan dimaksudkan untuk mencapai sesuatu sasaran atau tujuan.

2.1.1.1 Komponen-komponen Sistem Informasi

Zakiyudin (2013: 13) menyatakan bahwa dalam suatu sistem informasi, terdapat komponen-komponen seperti :

1. Perangkat keras (*hardware*), mencakup peranti-peranti fisik seperti computer dan printer.
2. Perangkat lunak (*software*) atau program, yaitu sekumpulan instruksi yang memungkinkan perangkat keras untuk dapat memproses data.
3. Basis data (*database*), adalah sekumpulan tabel, hubungan dan lain-lain yang berkaitan dengan penyimpanan data.
4. Prosedur, adalah semua pihak yang bertanggung jawab dalam pengembangan sistem informasi, pemrosesan data dan pembangkitan keluaran yang dikehendaki.
5. Personal atau orang, adalah semua pihak yang bertanggung jawab dalam pengembangan sistem informasi, pemrosesan dan penggunaan sistem informasi.
6. Jaringan komputer dan komunikasi data, merupakan sistem penghubung yang memungkinkan sumber (*resource*) dipakai secara bersama atau diakses oleh sejumlah pemakai.

Namun demikian, tidak semua sisteminformasi mencakup keseluruhan komponen-komponen tersebut.Sebagai contohnya adalah, sistem informasi pribadi yang hanya melibatkan seseorang pemakai dan sebuah komputer, tidak melibatkan fasilitas jaringan dan komunikasi.Akan tetapi sistem informasi

kelompok kerja (*workgroup information system*) yang melibatkan sejumlah orang dan sejumlah komputer memerlukan sarana jaringan komunikasi.

2.1.2 SDLC (Software Development Life Cycle)

SDLC atau *Software Development Life Cycle* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakanmodel-model dan metodologi yang digunakan untuk mengembangkan sistem-sistem perangkat lunak sebelumnya. Tahapan-tahapan yang ada pada SDLC secara global menurut Rosa (2011: 24-26) adalah sebagai berikut:

1. Inisiasi (*initiation*)

   Tahap ini biasanya ditandai dengan pembuatan proposal proyek perangkat lunakPengembangan konsep sistem (*system concept development*) Mendefinisikan lingkup konsep termasuk dokumen lingkup sistem, analisis manfaat biaya, manajemen rencana, dan pembelajaran kemudahan sistem.

2. Perencanaan (*planning*)

   Mengembangkan rencana manajemen proyek dan dokumen perencanaan lainya.Menyediakan dasar untuk mendapatkan sumber daya (*resources*) yang dibutuhkan untuk memperoleh solusi.

3. Analisa kebutuhan (*requirments analysis*)

   Menganalisa kebutuhan pemakai sistem perangkat lunak (*user*) dan mengembangkan kebutuhan *user*.Membuat dokumen kebutuhan fungsional.

4. Desain (*design*)

   Mentransformasikan kebutuhan detail menjadi kebutuhan yang sudah lengkap, dokumen *design* sistem fokus pada bagaimana dapat memenuhi fungsi-fungsi yang dibutuhkan.

5. Pengembangan (*development*)

   Mengkonversi *design* ke sistem informasi yang lengkap termasuk bagaimana memperoleh dan melakukan instalasi lingkungan sistem yang dibutuhkan. Membuat basis data dan mempersiapkan prosedur kasus pengujian; mempersiapkan berkas atau *file* pengujian, pengodean, pengompilasian, memperbaiki dan membersihkan program, peninjauan pengujian.

6. Integrasi dan pengujian (*intergration and test*)

   Mendemonstrasikan sistem perangkat lunak bahwa telah memenuhi kebutuhan yang spesifikasi pada dokumen kebutuhan fungsional. Dengan diarahkan oleh staf penjamin kualitas (*quality assurance*) dan *user*.Menghasilkan laporan analisis pengujian.

7. Implementasi (*implementation*)

   Termasuk pada persiapan implementasi, implementasi perangkat lunak pada lingkungan produksi (lingkungan pada *user*) dan menjalankan resolusi dari permasalahan yangterindentifikasi dari fase integrasi dan pengujian.

8. Operation dan pemeliharaan (*Operations and maintenance*)

Mendeskripsikan pekerjaan untuk mengoperasikan dan memelihara sistem informasi pada lingkungan produksi, termasuk implementasi akhir dan masuk pada proses peninjauan.

9. Disposisi (*disposition*)

   Mendeskripsikan aktifitas akhir dari pengembangan sistem dan membangun data yang sebenarnya sesuai dengan aktifitas *user*.

Hal terpenting adalah bagaimana mengenali tipe pelanggan (*customer*) dan memilih menggunakan model SDLC yang sesuai dengan karakter pelanggan dan sesuai dengan karakter pengembang.

SDLC memiliki beberapa model dalam penerapan tahapan prosesnya, berikut adalah model-model SDLC menurut (Rosa 2011: 26-37):

1. Model *Waterfall*

   Model SDLC air terjun (*waterfall*) sering juga disebut model sekuensal linier (*sequential linier*) atau alur hidup klasik (*classic life cycle*). Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, *design*, pengodean, pengujian, dan tahap pendukung (*support*).

2. Model *Prototype*

   Model *prototype* dimulai dari pengumpulkan kebutuhan pelanggan terhadap perangkat lunak yang akan dibuat. Lalu dibuatlah program *Prototype* agar pelanggan lebih terbayang dengan apa sebenarnya dinginkan. Program

Prototype biasanya merupakan program *Prototype* yang belum jadi. Program ini biasanya merupakan program yang belum jadi.

3. Model *Rapid Application Development* (RAD)

   Rapid Application Development (RAD) adalah model proses pengembangan perangkat lunak yang bersifat inkremental terutama untuk waktu pengerjaan yang pendek. Model RAD adalah adaptasi dari model air terjun untuk mengembangkan setiap komponen perangkat lunak.

4. Model *Iteratif*

   Model iteratif mengkombinasikan proses-proses pada model air terjun dan iteratif pada *prototype*. Model inkremental akan menghasilkan versi-versi perangkat lunak yang sudah mengalami penambahan fungsi untuk setiap pertambahannya (inkremen/*increment*).

5. Model *Spiral*

   Model spiral memasangkan iteratif dan model *prototype* dengan *control* dan aspek sistematik yang diambil dari model air terjun.

Dari kelima model SDLC yang sudah dijelaskan diatas maka penulis akan menggunakan model *waterfall* karena pengaplikasian menggunakan model ini mudah dan tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan dari analisis, desain, *coding*, dan *testing*.

Kelebihan dari model ini adalah ketika semua kebutuhan sistem dapat didefinisikan secara utuh, di awal *project*, maka aplikasi dapat berjalan dengan baik dan tanpa masalah.Meskipun seringkali kebutuhan sistem tidak dapat didefinisikan sesuai yang diinginkan, tetapi paling tidak, *problem* pada kebutuhan

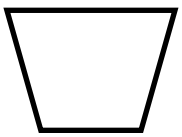
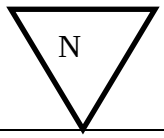
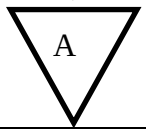
sistem di awal *project* lebih ekonomis dalam hal uang (lebih murah), usaha, dan waktu yang terbuang lebih sedikit jika dibandingkan problem yang muncul pada tahap-tahap selanjutnya.

2.1.2 Bagan alir (*Flowchart*)

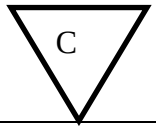
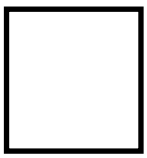
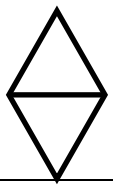
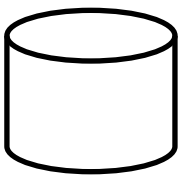
Bagan alir (*flowhcart*) adalah bagian bagan (*chart*) yang menunjukkan alir (*flow*) di dalam program atau prosedur sistem secara logika. Bagan alir digunakan terutama untuk alat bantu komunikasi dan untuk dokumentasi (Jogiyanto,2009: 795).

Bagan alir sistem digambarkan dengan menggunakan simbol-simbol sebagai berikut ini.

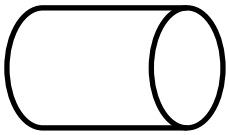
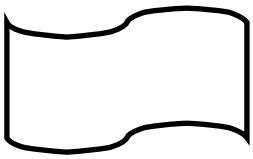
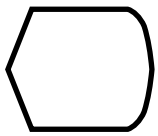
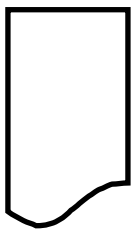
Tabel 2.1. Bagan alir *system* (*flowchart*)

Simbol	Keterangan
Simbol dokumen 	Menunjukkan dokumen *input* dan *output* baik untuk proses manual, mekanik atau computer
Simbol kegiatan manual 	Menunjukkan pekerjaan manual
Simbol simpanan *offline* 	*File non* komputer yang di arsip urut angka (*numerical*)
	*File non* komputer yang di arsip urut huruf (*alfabetical*)

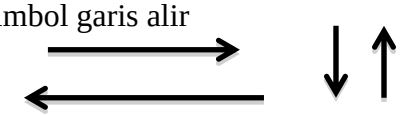
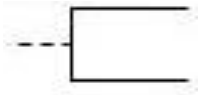
Tabel 2.1 lanjutan Bagan alir *system(flowchart)*

	*File non* komputer yang di arsip urut tanggal (*cronological*)
Simbol kartu plong 	Menunjukkan *input/output* yang menggunakan kartu plong (*purchase card*)
Simbol proses 	Menunjukkan kegiatan proses dari operasi program computer
Simbol operasi luar 	Menunjukkan operasi yang dilakukan diluar proses computer
Simbol pengurutan *offline* 	Menunjukkan proses pengurutan data diluar proses computer
Simbol pita *magnetic* 	Menunjukkan *input/output*menggunakan pita *magnetik*
Simbol hard disk 	Menunjukkan *input/output*menggunakan*hard disk*

Tabel 2.1 lanjutan. Bagan alir *system(flowchart)*

Simbol disket 	Menunjukkan*input/output* menggunakan *diskette*
Simbol drum magnetic 	Menunjukkan *input/ouput* menggunakan *drum magnetik*
Simbol pita kertas berlubang 	Menunjukkan *input/output* menggunakan pita kertas berlubang
Simbol *keyboard* 	Menggunakan*input* yang menggunakan *online keyboard*
Simbol *diplay* 	Menunjukkan *output* yang ditampilkan monitor
Simbol pita control 	Menunjukkan penggunaan pita *control* dalam *batch control total* untuk pencocokan diproses *batch processing*
Simbol hubungan komunikasi 	Menunjukkanproses transmisi data melalui *channel* komunikasi

Tabel 2.1 lanjutan Bagan alir *system(flowchart)*

Simbol garis alir 	Menunjukkan arus dari proses
Simbol penjelasan 	Menunjukkan penjelasan dari suatu proses
Simbol penghubung 	Menunjukkan penghubung kehalaman yang masih sama atau kehalaman lain

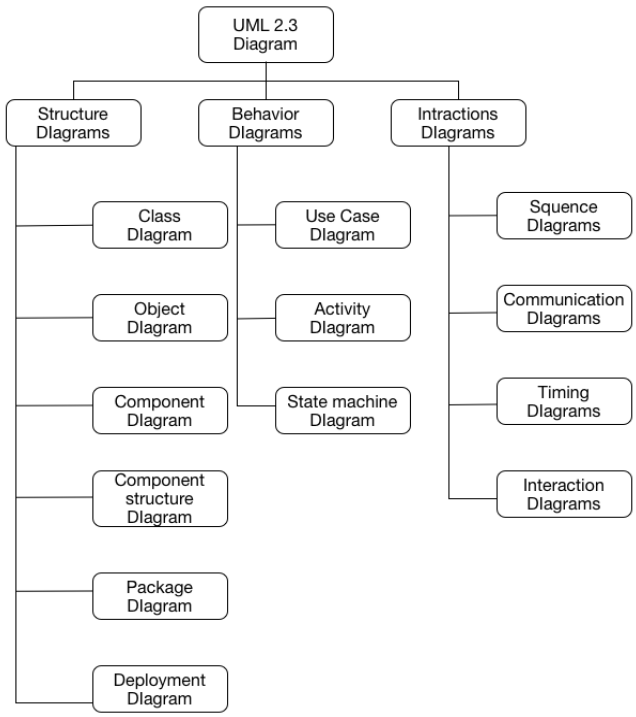
Sumber: Jogiyanto (2005 : 797)

2.1.3 UML (Unifed Modeling Languange)

UML (*Unifed Modeling Langunge*) adalah salah satu alat bantu yang sangat handal didunia pengembangan sistem berorientasi obyek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan *visual* yang memungkinkan bagi pengembang membuat cetak biru atas visi mereka dalam bentuk yang baku. Menurut (Rosa, 2011:120) UML adalah sekumpulan spesifikasi yang dikeluarkan oleh *Object Management Group* (OMG) yang terdiri dari UML *Superstructure*, dan *Object Constraint Language* (OCL).

2.1.4.1. Diagram UML

Diagram UML 2.3 terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar 2.1 dibawah ini:



Gambar 2.1. Diagram UML

Berikut ini penjelasan singkat dari pembagian kategori tersebut:

Structure diagram yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.

Behavior diagram yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakukan sistem atau rangkain perubahan yang terjadi pada sebuah sistem.

Interaction diagram yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antara subsistem pada suatu sistem.

Dari 13 diagram diatas disini peniliti hanya menggunakan*Class diagram*, *Use Case diagram*, *Activity diagram* dan *Squance diagram*, berikut penjelasan dari masing-masing diagram:

1. *Class Diagram*

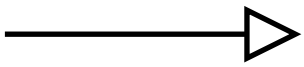
Diagran kelas atau *class diagram* menggambarkan struktur sistem dari pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas Memiliki atribut dan metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas, sedangkan operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas. Dalam mendefinisikan metode yang ada di dalam kelas perlu memperhatikan *cohesion* dan *coupling*. *Cohesion* adalah ukuran seberapa dekat keterkaitan instruksi di dalam sebuah metode terkait satu sama lain sedangkan *coupling* adalah ukuran seberapa dekat keterkaitan instruksi antara metode yang satu dengan metode yang lain dalam suatu sebuah kelas.

Berikut adalah simbol-simbol yang ada pada diagram kelas:

Tabel 2.2. *Class diagram* simbol

Simbol	Deskripsi
Kelas **nama_kelas** +atribut +operasi()	Kelas pada struktur sistem
Antarmuka / *Interface* **nama_interface**	Sama dengan konsep *interface* dalam pemograman berorientasi objek
Asosiasi / *association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*

Tabel 2.3. Lanjutan *Class diagram* simbol

Asosiasi berarah / *ditected association* 	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 / *dependency* 	Relasi antara kelas dengan makna kebergantungan antar kelas
Agregasi / *aggregation* 	Relasi antar kelas dengan makna semua bagian (*whol –part*)

Sumber: Rosa (2011 : 123)

2. *Use Case Diagram*

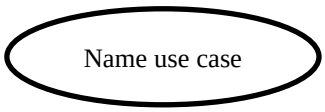
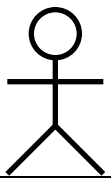
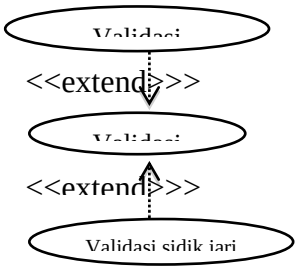
Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*.

Aktor: merupakan orang, proses, atau sistem lain berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.

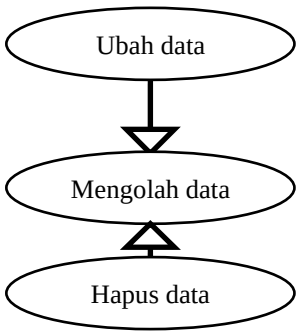
Use case: merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

Berikut adalah simbol-simbol yang ada pada diagram *use case* :

Tabel 2.3.*Use case* simbol

Simbol	Deskripsi
Use case 	Fungsional yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antara unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja diawal di awal frase nama *use case.*
Aktor / *actor* 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun *symbol* dari aktor adalah gambar orang, biasanya dinyatakan menggunakan kata benda diawal frase nama aktor.
Asosiasi / *association* 	Komunikasi antara aktor dan *use case*yang berpartisipasi pada *use case*atau*use case* memiliki interaksi dengan aktor.
Ektensi / *extend* 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case*tambahan yaitu , mirip dengan prinsip *inheritance*pada pemrograman berorientasi objek, biasanya *use case* tambahan memiliki nama depan sama dengan *use case*yang ditambahkan misal  Arah panah mengarah pada *use case* yang ditambahkan

Tabel 2.3 Lanjutan.*Use case* simbol

Generalisasi / *generalization* 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainya, misalnya : 
	Arah panah mengarah pada *use case*yang menjadi generalisasi (umum)
Menggunakan / include / *uses* <<include>>  <<uses>> 	Relasi *use case*tambah ke sebuah *use case*dimana *use case*yang ditambahkan memerlukan *use case*ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini Ada dua sudut pandang yang cukup besar mengenai *include* di *usecase*: Include berarti use case yang ditambahkan akan selalu dipanggil saat *use case*yang ditambahkan dijalankan Include berarti *use case*yang tambahan akan selalu melakukan pengecekan apakah *use case*yang ditambahkan telah dijalankan sebelum *use case*tambahan dijalankan Kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan

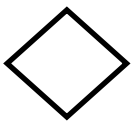
Sumber: Rosa (2011 : 135)

3. Activity Diagram

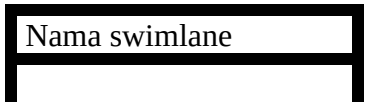
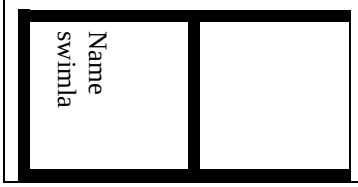
Diagram aktivitas menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut :

1. Rancangan proses bisnis di mana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari *sistem/userinterface* dimana setiap aktivitas di anggap memiliki sebuah rencanngan antarmuka tampilan.
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan khusus ujinya.

Tabel 2.4.*Activity* diagram simbol

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan / *decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan / *join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Status Akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir

Tabel 2.4. Lanjutan. *Activity* diagram simbol

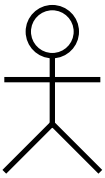
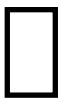
Swimlane  Atau  	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Sumber: Rosa (2011 : 135)

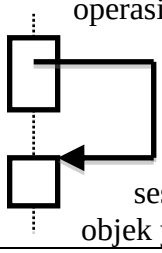
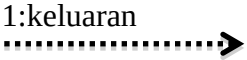
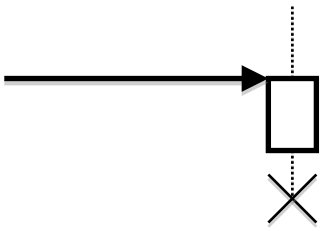
4. Sequence Diagram

Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Banyaknya diagram sekuen 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 sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak.

Tabel 2.5.*Sequence* diagram simbol

Simbol	Deskripsi
Aktor  Nama_aktor Atau Nama_aktor Tanpa waktu aktif	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama actor
Garis hidup / *lifeline*	Menyatakan kehidupan suatu objek
Objek Nama_objek :nama_kelas	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan
Pesan tipe create «create» 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat

Tabel 2.5 Lanjutan.*Sequnce diagram* simbol

Pesan tipe *call* 1:nama_metode 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri, Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka  operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi
Pesan tipe *send* 1: masukan 	Menyatakan bahwa suatu objek mengirim data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi
Pesan tipe *return* 1:keluaran 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian
Pesan tipe *destroy* «destory» 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada *create* maka ada *destroy*

Sumber: Rosa (2011: 138-139)

2.2 Tinjauan Teori Khusus

2.2.1 Aplikasi

Jogiyanto (2005) dalam Afrian (2014:45) dalam Pelita Informatika Budi Darma, volume: VI, Nomor: 1, Maret 2014 Perangkat lunak aplikasi adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk untuk melakukan suatu tugas yang diinginkan pengguna. Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer, tapi tidak secara langsung menerapkan kemampuan tersebut untuk mengerjakan suatu tugas yang menguntungkan pengguna. Contoh utama perangkat lunak aplikasi adalah pengolah kata, lembar kerja, dan pemutar media. Menurut Indrajani (2014:4) aplikasi adalah program yang menentukan aktivitas pemrosesan informasi yang dibutuhkan untuk penyelesaian tugas-tugas khusus dari pemakai komputer.

2.3 Transportasi

2.3.1 Pengertian Transportasi

Pengertian transportasi yang dikemukakan oleh Nasution dalam Muthalib dan Oleo dalam Progres Ekonomi Pembangunan Volume 1, Nomor 1, 2016 diartikan sebagai pemindahan barang dan manusia dari tempat asal ke tempat tujuan. Sehingga dengan kegiatan tersebut maka terdapat tiga hal yaitu adanya muatan yang diangkut, tersedianya kendaraan sebagai alat angkut, dan terdapatnya jalan yang dapat dilalui. Proses pemindahan dari gerakan tempat asal,

diamana kegiatan pengankutan dimulai dan ketempat tujuan dimana kegiatan diakhiri.

2.4 Android

Menurut Lee (2012:2), Android merupakan sistem operasi ponsel yang didasari oleh Linux. Android awalnya dibuat dari suatu proyek yang bernama sama yaitu Android, Inc. Pada tahun 2005, sebagai strategi untuk merambah dunia ponsel, Google membeli proyek Android juga tim yang membuatnya. Android disenangi oleh masyarakat karena fleksibilitasnya, dan disenangi oleh para pembuat aplikasi karena ketika membuat buat aplikasi untuk Android, maka aplikasi itu dapat dijalankan pada semua ponsel bermerek apapun, selama ponsel itu mempunyai sistem operasi Android.

2.5 Fitur-fitur Android

Karena Android bersifat *open-source* dan dapat digunakan oleh pembuataplikasi secara bebas, tidak ada pengaturan resmi untuk perangkat keras dan perangkat lunak untuk Android. Walaupun begitu, Android mempunyai *fitur-fitur* tersendiri seperti yang diutarakan oleh Lee (2012:3), antara lain :

1. Penyimpanan Data, menggunakan SQLite, sebuah basis data yang ringan untuk penyimpanan data.
2. Konektifitas, mendukung fitur GSM/EDGE, IDEN, CDMA, EVDO, UMTS, Bluetooth (termasuk A2DP dan AVRCP), Wi-Fi, LTE, dan WiMAX.
3. Pengiriman Pesan, mendukung SMS dan MMS.

4. *Web Browser*, berbasis pada aplikasi *open-source* WebKit dengan menggunakan mesin Chrome's V8 JavaScript.
5. Media, mendukung media H.263, H.264 (dalam wujud 3GP atau MP4), MPEG-4 SP, AMR, AMR-WB (dalam wujud 3GP), AAC, HE-AAC (dalam wujud MP4 atau 3GP), MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF, dan BMP. Perangkat Keras, dilengkapi sensor Accelerometer, Kamera, Kompas Digital, Sensor Jarak, dan GPS.
6. *Multi-Touch*, mendukung fitur layar *multi-touch*.
7. *Multi-Tasking*, mendukung aplikasi yang mempunyai fitur *multitask.*
8. Flash, Android versi 2.3 mendukung Flash versi 10.1.
9. Tethering, mendukung fitur berbagi koneksi internet sebagai *hotspot* tanpa kabel.

Fitur akan terus bertambah seiring dengan terus berkembangnya teknologi.

2.5.1 Javascript

Javascript merupakan bahasa pemrograman untuk membuat *website* di internet dan dapat bekerja di sebagian besar *browse*r.Javascript berkerja bersama dengan HTML5 untuk membuat sebuah logika pemrograman serta membuat website menjadi lebih beragam.Javascript memungkinkan objek pada web untuk berinteraksi dengan pengguna, mengontrol *browserweb*, dan mengubah isi dokumen yang muncul dalam layar *webbrowser*.Javascript merupakan bahasa pemrograman pada web.Mayoritas *website* saat ini menggunakan Javascript.Javascript merupakan bahasa pemrograman tingkat tinggi, dinamis

yang sangat cocok untuk pemrograman berbasis objek (*object-oriented*) dan fungsi (Flanagan, 2012).

Penggunaan Javascript pada perancangan ini dikarenakan aplikasi yang dirancang ini berbasis *web* aplikasi.

2.5.2 AngularJS

Merujuk (*AngularJS*, 2015) *AngularJS* yaitu merupakan *framework* javascript berbasis *open-source* yang dirilis oleh Google pada tahun 2009 dengan tagline berikut ini***"HTML Enhanced for Web apps!"***yang di Maksud dari *tagline* AngularJS ini adalah HTML yang ditingkatkan fungsinya untuk membangun *web app*. Melihat sejarah kemunculan HTML, awalnya HTML hanya digunakan untuk membuat dokumen statis (website) bukan untuk membuat *web app.* Nah, sekarang bayangkan kalau sejak awal HTML memang dikembangkan untuk membuat *web app*, seperti itulah konsep AngularJS.AngularJS bukan merupakan pustaka (*library*) *javascript* melainkan sebuah *framework* yang solid untuk membangun *web app*, seperti *framework javascript* pada umumnya AngularJS mengadopsi konsep MVC (*Model*, *View*, *Controller*), meskipun menggunakan implementasi yang berbeda dengan konsep asli MVC.

Angular digunakan dalam perancangan ini karena merupakan pendukung dari Javascript. Dengan penggunaan Angular proses pembuatan aplikasi menjadi lebih cepat.

2.5.3 PHP

Menurut (Peranginangin, 2008: 2–3) PHP singkatan dari PHP *Hypertext Preprocessor* yang digunakan sebagai bahasa *script server-side* dalam

pengembangan web yang disisipkan pada dokumen HTML.PHP diciptakan pertama kali oleh Rasmus Lerdorf pada tahun 1994.Awalnya, PHP digunakan untuk mencatat jumlah serta untuk mengetahui siapa saja pengunjung pada *homepagenya*.

Pada tahun 1996, PHP telah banyak digunakan dalam *website* didunia. Sebuah kelompok pengembang *software* yang terdiri dari Rasmus, Zeew Suraski, Andi Gutman, Stig Bakken, Shane Caraveo, dan Jim Winstead bekerja sama untuk menyempurnakan PHP 2. Akhirnya, pada tahun 1998 PHP 3.0 diluncurkan.Penyempurnaan terus dilakukan sehingga pada tahun 2000 dikeluarkan PHP 4.Tidak berhenti sampai disitu, kemampuan PHP terus ditambah, dan saat buku ini disusun versi terbaru yang telah dikeluarkan adalah PHP 5.x.

Salah satu fitur yang dapat diandalkan oleh PHP adalah dukungannya terhadap banyak *database*.*Database* yang dapat didukung oleh PHP adalah Adabas D, dBase, Direct MS-SQL, Empress, FilePro (read only), FrontBase, Hyperwave, IBM DB2, dan Informix.(Peranginangin, 2008: 2-3)

Penggunaan PHP pada perancangan ini dikarenakan penyesuaian dengan bahasa pemrograman untuk digunakan pada aplikasi sistem pemesanan tiket Bus Trans Batam agar *intergasi* data menjadi lebih mudah.

2.5.4 Laravel

McCool (2012: 15) menyatakan bahwa Laravel adalah *framework* MVC (*Model View Controller*) yang dikembangkan dengan bahasa pemograman PHP.Laravel sendiri dirancang untuk meningkatkan kualitas perangkat lunak yang

dibangun untuk menyederhanakan tugas-tugas yang digunakan sehingga menjadi lebih praktis seperti *otentikasi, routing*, *sessions* dan *caching*, sehingga dengan mengurangi biaya pengembangan dan pemeliharaan.

Laravel menjadi pilihan bagi penulis dalam merancang aplikasi ini, dikarenakan Laravel merupakan sebuah *framework* yang mendukung PHP.

2.5.5 MySQL

MySQL (My Structured Query Language) Menurut Hirin dan Virgi (2011), *MySQL* adalah salah satu perangkat lunak sistem manajemen basis data *(database) SQL* atau sering disebut dengan *DBMS (Database Management System).*Berbeda dengan basis data konvensional seperti.*Dat, .dbf, .mdb,MySQL* memiliki kelebihan yaitu bersifat multithread, dan multi-user serta mendukung sistem jaringan.*MySQL* didistribuasikan secara gratis dibawah lisensi *GNU General Public License (GPL),* namun ada juga versi komersial bagi kalangan tertentu yang menginginkannya.MySQL sebenarnya merupakan turunan salah satu konsep utama dalam database sejak lama, yaitu *SQL (Structured Query Language).SQL* adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.