BAB II
TINJAUAN PUSTAKA

2.1 Rancang

Menurut (Nafisah, 2003). “Rancang adalah penggambaran atau perencanaan dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah kedalam suatu kesatuan yang utuh dan fungsi perencanaan sistem dapat dirancang dalam bentukbagan alir sistem (*system flowchat*) yang merupakan alat bentuk grafik yang dapat digunakan untuk menunjukkan urutan-urutan proses dari sistem.

2.2 Bangun

Menurut (Pressman, 2002). “Bangun atau pembangunan sistem adalah kegiatan menciptakan sistem baru maupun mengganti atau memperbaiki sistem yang telah ada baik secara keseluruhan maupun sebagian. Bangun sistem adalah membangun sistem informasi dan komponen yang didasarkan pada spesifikasi desain.

2.3 Pengertian Sistem

Menurut Jogiyanto (2005). “Sistem adalah suatu jaringan kinerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran yang tertentu. Pendekatan sistem yang lebih menekankan pada elemen atau komponen. Mendefinisikan sistem sebagai kumpulan dari elemen-elemen yang berinteraksi untuk mencapai suatu tujuan tertentu”.

2.4 Karakteristik Sistem

Menurut Jogiyanto (2005). "Dalam sebuah sistem mempunyai karakteristik yang tidak terpisahkan antara satu karakteristik dengan karakteristik yang lain". Beberapa karakteristik tersebut antara lain:

1. Komponen *(Components*)

   Suatu sistem memiliki sejumlah komponen yang saling berinteraksi, dimana setiap komponen akan membentuk satu kesatuan yang saling bekerja sama. Komponen sistem dapat berupa suatu yang merupakan bagian dari sistem yang lebih besar.
2. Batas Sistem *(Boundary)*Merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lain/lingkungan luar, dengan batasan ini kita dapat mengetahui ruang lingkup sistem.
3. Lingkungan Luar Sistem *(Environment)*

   Apapun yang berada di luar batas dari sistem yang mempengaruhi operasi suatu sistem.
4. Penghubung Sistem *(Interface)*

   Merupakan media penghubung antara satu subsistem dengan subsistem yang lainnya. Dengan penghubung ini akan mengalir data-data antara subsistem dimana keluaran *(output)* dari satu subsistem akan menjadi masukan *(input)* untuk subsistem yang lain, sehingga antara satu subsistem dengan subsistem lainnya dapat berintegrasi membentuk satu kesatuan.
5. Masukan *(Input)*

   Merupakan energi yang dimasukkan ke dalam sistem, dimana masukan ini dapat berupa masukan perawatan *(maintenance input)* dan masukan sinyal *(signal input)*. Maintenance input adalah energi yang dimasukkan supaya sistem tersebut dapat beroperasi. *Signal input* adalah energi yang diproses untuk didapatkan keluaran.

6. Keluaran *(Output)*

   Merupakan hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan mampu menjadi masukan baru/informasi yang dibutuhkan.

7. Pengolah *(Process)*

   Suatu sistem pasti mempunyai pengolahan data masukan untuk diolah menjadi sebuah informasi.

8. Sasaran Sistem *(Objectives)*

   Merupakan penentu dari tujuan untuk menentukan masukan yang dibutuhkan dan keluaran yang akan dihasilkan sebuah sistem.

2.5 Metode Pengembangan Sistem

Untuk menggambarkan kegiatan-kegiatan yang dilaksanakan selama pengembangan suatu sistem digunakan metode pengembangan system yaitu *System Development Life Cycle (*SDLC) yang memiliki 4 langkah fundamental yaitu sebagai berikut :

2.5.1 Perencanaan Sistem (*System Planning*)

Pengumpulan data atau fakta yang dapat mendukung suatu sistem untuk dapat dikembangkan atau dibuat sistem baru.

2.5.2 Analisis Sistem (*System Analisis*)

Mempelajari masalah-masalah yang timbul dan menentukan kebutuhan pemakai sistem untuk mengidentifikasi pemecahan yang beralasan.

2.5.3 Perancangan Sistem (*System Design*)

Perancangan sistem merupakan proses penyiapan spesifikasi yang terperinci untuk pengembangan sistem baru. Dimulai dari spesifikasi output sistem yang diperlukan, mencakup isi, format, volume dan frekuensi laporan-laporan dan dokumen-dokumen juga input sistem dan file.

2.5.4 Implementasi Sistem (*System Implementation*)

Pelaksanaan mencakup pelaksanaan alternatif yang dipilih agar sistem siap untuk dioperasikan.

2.6 Pengertian Aplikasi

Menurut (Hendrayudi 2009) menyatakan bahwa "Aplikasi adalah kumpulan perintah program yang dibuat untuk melakukan pekerjaan - pekerjaan tertentu atau khusus".

2.7 Pengertian Pelumas

Menurut A.R Lansdown (2003) "Pelumas adalah salah satu penopang utama dari kerja sebuah mesin. Pulumas juga menentukan *performance* dan endurance dari mesin, maka semakin baik kualitas yang digunakan, semakin baik pula performance dan endurance mesin". Fungsi utama pelumas ada dua,yaitu untuk mencegah kontak langsungantara komponen yang terbuat dari logam dan sebagai pendingin. Pelumas harus mampu mengurangi panas yang ditimbulkan oleh gesekan antar komponen yang bergeak pada mesin.

2.8 Kendaraan

2.8.1. Definisi

Mobil adalah kendaraan darat yang digerakkan oleh tenaga mesin, beroda empat atau lebih(selalu genap)biasanya berbahan bakarnya minyak (bensin atau solar)untuk menghidupkan mesinnya.

2.8.2. Sejarah

Mobil (kependekan dari otomobil yang berasal dari bahasa Yunani 'autos' (sendiri) dan Latin 'movére' (bergerak)) adalah kendaraan beroda empat atau lebih yang membawa mesin sendiri. Jenis mobil termasuk bus dan truk. Pengoperasian mobil disebut menyetir. Bila yang dimaksud mobil seperti sekarang ini, cukup sulit untuk

menentukan siapa penemunya. Mobil sebenarnya terdiri dari ribuan komponen yang ditemukan dan dikembangkan secara bertahap. Meski demikian, bisa disebutkan bahwa Nicolaus J. Cugnot adalah orang pertama kali yang berhasil menluncurkan kendaraan berbadan besar, beroda tiga dan bermesin uap. Kendaraan ini digunakan untuk menarik Meriam pada tahun 1769. Kendaraan hasil ciptaan Nicolaus J. Cugnot sedang menabrak tembok. Pada kecelakaan ini tercatat sebagai kecelakaan mobil pertama di dunia. Paten mobil pertama di Amerika Serikat diberikan kepada Oliver Evans pada 1789; pada 1804 Evans mendemonstrasikan mobil pertamanya, yang bukan hanya mobil pertama di AS tapi juga merupakan kendaraan amfibi pertama, yang kendaraan tenaga-uapnya sanggup jalan di darat menggunakan roda dan di air menggunakan roda padel. Umumnya mobil pertama mesin pembakaran dalam yang menggunakan bensin dibuat hampir bersamaan pada 1886 oleh penemu Jerman yang bekerja secara terpisah. Carl Benz pada 3 Juli 1886 di Mannheim, dan Gottlieb Daimler dan Wilhelm Maybach di Stuttgart Pada 5 November 1895, George B. Selden diberikan paten AS untuk mesin mobil dua tak. Paten ini memberi dampak negatif pada perkembangan industri mobil di AS. Penerobosan spektakuler dilakukan oleh Berta Benz pada 1888. Mesin-uap, listrik, dan bensin bersaing untuk beberapa dekade, dengan mesin bensin pembakaran dalam meraih dominasi pada 1910an. Dalam periode dari 1900 ke pertengahan 1920-an perkembangan teknologi otomotif sangat cepat, disebabkan oleh jumlah besar (ratusan) pembuat mobil kecil yang semuanya bersaing untuk meraih perhatian dunia. Pengembangan utama termasuk penyalaan elektronik dan self-starter elektronik (keduanya oleh Charles Kettering, untuk Perusahaan mobil Cadillac di tahun (1910-1911), suspense independen, dan rem empat ban.

2.9 Metode yang *K-Means Clustering*

Fajar Astuti Hermawati (2009). "Merupakan pendekatan partitional *clustering*. Tiap *cluster* dihubungkan dengan sebuah *centroid* (titik pusat). Tiap titik ditempatkan kedalam *cluster* dengan *centroid* terdekat. Jumlah cluster, K, harus ditentukan. Algoritma dasarnya sangat sederhana, yaitu:

1. Pilih K *cluster* awal.
2. Ulangi.
3. Bentuk K *cluster* dengan menempatkan semua titik yang terdekat.
4. Ulangi perhitungan *centroid* dari tiap *cluster*.
5. Sampai *centroid* yang tidak berubah.

Cluster1 (C1) = {2,3}

Cluster2 (C2) = {4,10,11,12,20,25,35}

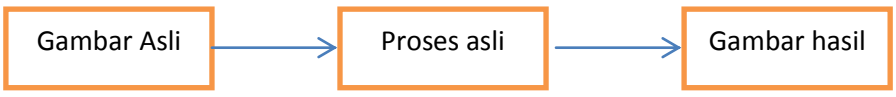
Perhitungan centroid baru

$\mu'_1 = \frac{2+3}{2} = \frac{5}{2} = 2,5$ $\mu'_1 = \frac{2+3}{2} = \frac{5}{2} = 2.5$

$\mu'_2 = \frac{4+10+11+12+20+25+35}{7} = \frac{112}{7} = 16$

2.10 Pengolahan Citra

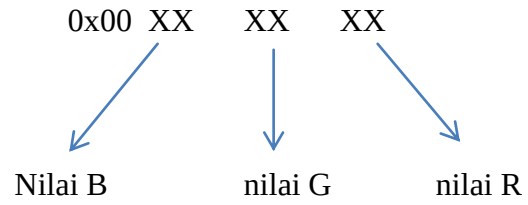
Image processing atau sering disebut pengolahan citra digital merupakan suatu proses filter gambar asli menjadi gambar lain sesui dengan keinginan kita, misalnya kita mendapatkan suatu gambar yang terlalu gelap. Dengan image processing, kita dapat memproses agar mendapatkan gambar yang jelas. Secara garis besar , kita bisa menggambarkan seperti blok diagram pada gambar



- Membuka file
- Menambah healder file
- Menjalankan program

A. Pengolahan warna RGB

Dasar pengolahan citra adalah pengolahan warna RGB pada posisi tertentu. Dalam pengolahan citra, warna dipresentasikan dengan nilai hexadesimal dari 0x0000000 sampai 0x00ffffff. Warna hitam adalah 0x00000000 dan warna putih adalah 0x00ffffffff. Menunjukkan definisi nilai warna dan variabel 0x00 menyatakan angka di belakangnya adalah hexadesimal.



2.11 MATLAB

Menurut Sianipar (2013). "MATLAB adalah suatu paket prangkat lunak yang memampkan anda untuk melakukan komputasi matematik menganalisis data, megembangkan algoritma, melakukan simulasi dan pemodelan, dan menghasilkan tampilan grafik dan antarmuka grafikal.

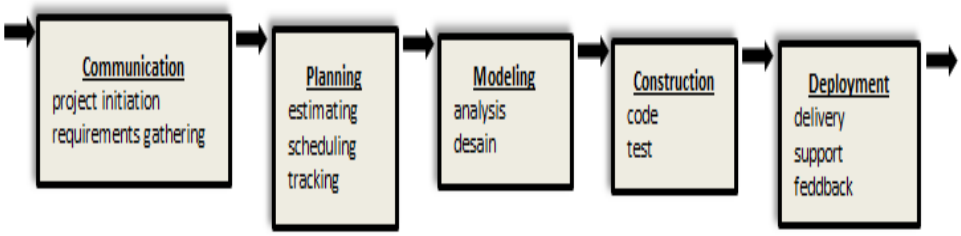
Untuk menjalanka MATLAB pada suatu komputer pribadi, anda hanya perlu mengeklik dua kali ikon MATLAB.

2.12 Aplikasi Dekstop

Aplikasi adalah program yang digunakan orang untuk melakukan suatu pada sistem komputer. Destop Komputer adalah komputer pribadi yang ditunjukan untuk penggunaan secara umum di situ lokasi yang berlawanan dengan komputer jinjing atau komputer dekstop. Periferal-periferal komputer meja seperti tampilan komputer, CPU (Central Processing Unit) dan papan ketik terpisah satu sam lain relatif berukuran besar (juga berlawanan dengan periferal besar komputer jinjing yang terintergritasi dan berukuran kecil).

2.13 Model Waterfall

Menurut (Pressman 2010) "model *waterfall* adalah model klasik yang bersifat sistematis, berurutan dalam membangun *software*. Berikut ini ada dua gambaran dari *waterfall* model". Fase-fase dalam model *waterfall* menurut referensi Pressman:



Gambar 2.1 *Waterfall Pressman*

1. ***Communication***

   Langkah ini merupakan analisis terhadap kebutuhan *software*, dan tahap untuk mengadakan pengumpulan data dengan melakukan pertemuan dengan *customer* maupun mengumpulkan data-data tambahan baik yang ada di jurnal, artikel, maupun dari *internet*.

2. ***Planning***

   Proses *planning* merupakan lanjutan dari proses *communication (analysis requirement)*. Tahapan ini akan menghasilkan dokumen *user requirement* atau bisa dikatakan sebagai data yang berhubungan dengan keinginan *user* dalam pembuatan *software*, termasuk rencana yang akan dilakukan.

3. ***Modeling***

   Proses *modeling* ini akan menerjemahkan syarat kebutuhan ke sebuah perancangan *software* yang dapat diperkirakan sebelum dibuat coding. Proses ini berfokus pada rancangan struktur data, arsitektur *software*, *representasi interface*, dan detail (algoritma) prosedural. Tahapan ini akan menghasilkan dokumen yang disebut *software requirement*.

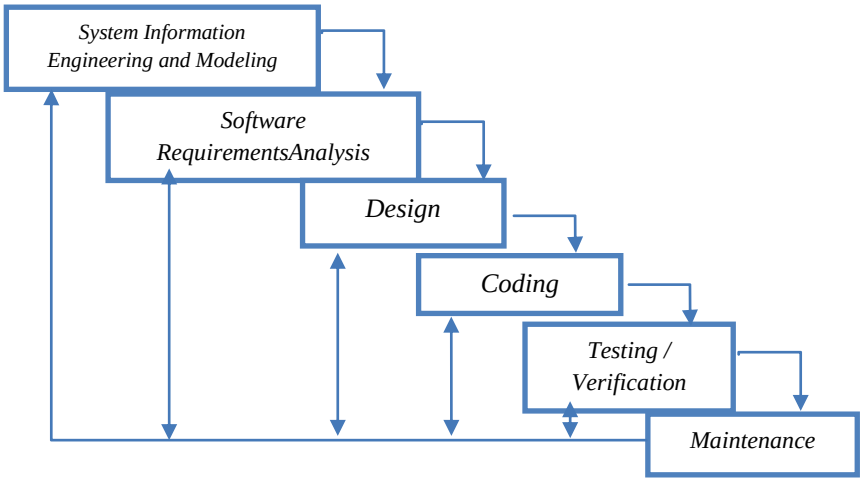
4. *Construction*

Construction merupakan proses membuat kode. Coding atau pengkodean merupakan penerjemahan desain dalam bahasa yang bisa dikenali oleh komputer. *Programmer* akan menerjemahkan transaksi yang diminta oleh user. Tahapan inilah yang merupakan tahapan secara nyata dalam mengerjakan suatu *software*, artinya penggunaan komputer akan dimaksimalkan dalam tahapan ini. Setelah pengkodean selesai maka akan dilakukan testing terhadap sistem yang telah dibuat tadi. Tujuan testing adalah menemukan kesalahan-kesalahan terhadap sistem tersebut untuk kemudian bisa diperbaiki.

5. *Deployment*

Tahapan ini bisa dikatakan final dalam pembuatan sebuah *software* atau sistem. Setelah melakukan analisis, desain dan pengkodean maka sistem yang sudah jadi akan digunakan oleh user. Kemudian *software* yang telah dibuat harus dilakukan pemeliharaan secara berkala.

Sedangkan fase-fase model *waterfall* menurut referensi Sommerfille terlihat pada Gambar 2.2:



Gambar 2.2 *Waterfall Sommerfille*

1. ***Requirements Analysis and Definition***

   Mengumpulkan kebutuhan secara lengkap kemudian dianalisis dan didefinisikan kebutuhan yang harus dipenuhi oleh *software* yang akan dibangun. Hal ini sangat penting, mengingat *software* harus dapat berinteraksi dengan elemen-elemen yang lain seperti *hardware*, *database*, dsb. Tahap ini sering disebut dengan *Project Definition*.

2. ***System and Software Design***

   Proses pencarian kebutuhan diintensifkan dan difokuskan pada *software*. Untuk mengetahui sifat dari program yang akan dibuat, maka para *software engineer* harus mengerti tentang domain informasi dari *software*, misalnya fungsi yang dibutuhkan, *user interface*, dsb. Dari dua aktivitas tersebut (pencarian kebutuhan sistem dan *software*) harus didokumentasikan dan ditunjukkan kepada user. Proses *software* design untuk mengubah kebutuhan-kebutuhan di atas menjadi representasi ke dalam bentuk "*blueprint*" *software* sebelum *coding* dimulai. Desain harus dapat mengimplementasikan kebutuhan yang telah disebutkan pada tahap sebelumnya. Seperti dua aktivitas sebelumnya, maka proses ini juga harus didokumentasikan sebagai konfigurasi dari *software*.

3. ***Implementation and Unit Testing***

   Desain program diterjemahkan ke dalam kode-kode dengan menggunakan bahasa pemrograman yang sudah ditentukan. Program yang dibangun langsung diuji baik secara unit.

4. ***Integration and System Testing***

   Untuk dapat dimengerti oleh mesin, dalam hal ini adalah komputer, maka desain tadi harus diubah bentuknya menjadi bentuk yang dapat dimengerti oleh mesin, yaitu ke dalam bahasa pemrograman melalui proses *coding*. Tahap ini merupakan implementasi dari tahap *design*

yang secara teknis nantinya dikerjakan oleh *programmer*. Penyatuan unit-unit program kemudian diuji secara keseluruhan *(system testing)*.

5. *Operation and Maintenance*

Sesuatu yang dibuat haruslah diujicobakan. Demikian juga dengan *software*. Semua fungsi-fungsi *software* harus diujicobakan, agar *software* bebas dari *error*, dan hasilnya harus benar-benar sesuai dengan kebutuhan yang sudah didefinisikan sebelumnya. Pemeliharaan suatu software diperlukan, termasuk di dalamnya adalah pengembangan, karena software yang dibuat tidak selamanya hanya seperti itu. Ketika dijalankan mungkin saja masih ada error kecil yang tidak ditemukan sebelumnya, atau ada penambahan fitur-fitur yang belum ada pada software tersebut. Pengembangan diperlukan ketika adanya perubahan dari eksternal perusahaan seperti ketika ada pergantian sistem operasi, atau perangkat lainnya.

Kelebihan dari model ini adalah selain karena pengaplikasian menggunakan model ini mudah, kelebihan dari model ini adalah ketika semua kebutuhan sistem dapat didefinisikan secara utuh, eksplisit, dan benar di awal proyek, maka *Software Engineering (SE)* dapat berjalan dengan baik dan tanpa masalah. Meskipun seringkali kebutuhan sistem tidak dapat didefinisikan se-eksplisit yang diinginkan, tetapi paling tidak, problem pada kebutuhan sistem di awal proyek lebih ekonomis dalam hal uang (lebih murah), usaha, dan waktu yang terbuang lebih sedikit jika dibandingkan problem yang muncul pada tahap-tahap selanjutnya. Kekurangan yang utama dari model ini adalah kesulitan dalam mengakomodasi perubahan setelah proses dijalani. Fase sebelumnya harus lengkap dan selesai sebelum mengerjakan fase berikutnya. Masalah dengan *waterfall* :

1. Perubahan sulit dilakukan karena sifatnya yang kaku.

2. Dengan sifat kakunya, model ini cocok ketika kebutuhan dikumpulkan secara lengkap sehingga perubahan bisa ditekan sekecil mungkin. Tapi pada kenyataannya jarang sekali konsumen/pengguna yang bisa memberikan kebutuhan secara lengkap, perubahan kebutuhan adalah sesuatu yang wajar terjadi.
3. *Waterfall* pada umumnya digunakan untuk rekayasa sistem yang besar yaitu dengan proyek yang dikerjakan di beberapa tempat berbeda, dan dibagi menjadi beberapa bagian sub-proyek.

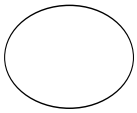
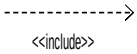
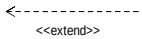
2.14 UML (*Unified Markup Languange*)

Menurut (Rosa & Salahudin 2014) menyatakan "UML adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak. UML merupakan metodelogi dalam mengembangkan sistem berorientasi object dan juga merupakan alat untuk mendukung pengembangan sistem. Alat bantu yang dipergunakan dalam perancangan berorientasi objek berbasis UML adalah sebagai berikut :

2.14.1 *Use Case Diagram*

Use case diagram merupakan pemodelan untuk kelakuan (*behavior*) sistem yang dibuat. *Usecase* mendeskripsikan sebuah interaksi antara satu atau lebih actor dengan sistem yang akan dibuat. Dapat dikatakan use case digunakan untuk mengetahui fungsi apa saja yang dibuat dan siapa saja yang berhak mengunakan fungsi-fungsi tersebut. Berikut simbol yang digunakan pada *use case diagram* :

Tabel 2.1 Simbol *Use Case Diagram*

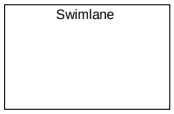
Gambar	Keterangan
	Use case menggambarkan fungsionalitas yang disediakan system sebagai unit-unit yang bertukar pesan antar unit dengan actor, biasanya dinyatakan dengan mengunakan kata kerja diawal nama use case.
	Aktor adalah orang lain atau system lain yang mengaktifkan fungsi dari target system. Untuk megidentifikasikan actor, harus ditentukan pembagian kerja dan tugas-tugas yang berkaitan dengan peran pada target system. Orang atau system bias muncul dalam beberapa peran.
	Ososiasi antara actor dan use case yang mengunakan panah terbuka untuk mengindikasikan bila actor berinteraksi secara pasif dengan system.
	Ososiasi antara actor dan use case yang mengunakan garis untuk mengindikasikan siapa atau apa yang meminta interaksi langsung bukan mengindikasikan aliran data.
	Include merupakan pemangilan use case oleh use case lainya , contohnya pemanggilan sebuah fungsi program.
	Perluasan dari use case lain jika kondisi atau syarat terpenuhi.

2.14.2 *Activity Diagram*

Secara grafis digunakan untuk menggambarkan rangkaian aliran aktivitas baik proses bisnis maupun *use case*. *Activity diagram* dapat juga digunakan untuk memodelkan action yang akan dilakukan saat sebuah operasi dieksekusi, dan memodelkan hasil dari action tersebut**.** Simbol-simbol yang digunakan dalam *activity diagram* yaitu :

Tabel 2.2 Simbol *Activity Diagram*

Gambar	**Keterangan**
	Mengambarkan suatu proses/kegiatan sistem

	Menunjukan eksekusi dari suatu aksi
	Start Point merupakan awak dari suatu aktivitas
	End point, akhir aktivitas
	Fork/Rake Node Satu aliran atau beberapa aliran yang pada tahap tertentu berubah menjadi satu atau beberapa aliran.
	Swinlane, pembagian activity diagram untuk menunjukan siapa melakukan apa

2.14.3 *Class Diagram*

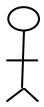
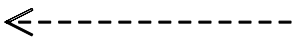
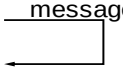
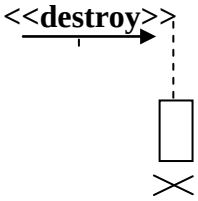
Class diagram merupakan hubungan antara kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem. *Class diagram* juga menunjukan atribut-atribut dan operasi-operasi dari sebuah kelas dan constraint yang berhubungan dengan objek yang dikoneksikan.

Class diagram secara khas meliputi: kelas (*class*), relasi, *Associations*, *Generalizion* dan *Agregation*, Atribut, operasi, dan *visibility*, tingkat akses objek eksternal kepada suatu operasi atau object.

2.14.4 *Sequence Diagram*

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

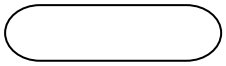
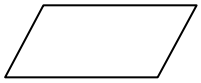
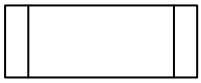
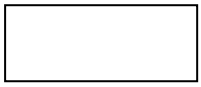
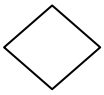
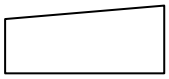
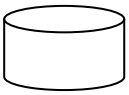
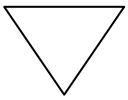
Tabel 2.3 Simbol *Sequence Diagram*

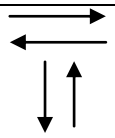
Gambar	Keterangan
	Orang, proses atau system lain yang berinteraksi dengan sistem
1: Masukan	Menyatakan bahwa suatu objek mengirimkan data atau masukan atau informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi.
	Activation, activation mewakili sebuah eksekusi operasi dari objek.
1: Keluaran 	Message return, simbol membalas pesan
1: [condition] message name 	Recursive, mengambarkan pengiriman pesan untuk dirinya sendiri.
<<**create**>>	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah objek yang di buat.
<<**destroy**>> 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy.
Garis Hidup / Lifeline	Menyatakan kehidupan suatu objek

2.15 Bagan Alir Program (*Flowchart*)

Menurut (Pahlevy 2010) Flowchart adalah gambaran dalam bentuk diagram alir algoritma-algoritma dalam suatu program yang dinyatakan arah alur program tersebut. Dapat dilihat pada tabel 2.4.

Tabel 2.4 Simbol-simbol bagan alir program *(Flowchart)*

Simbol	Keterangan
	*Terminator* Menunjukkan awal dan akhir suatu proses.
	*Input / Output* Menunjukkan data masukan (*input*) atau data keluaran (*output*).
	*Prosedur* Menunjukkan proses data.
	*Process* Menunjukkan kegiatan proses dari operasi program computer
	*Decision* Menunjukkan penjelasan suatu proses
	*Keybord* Memasukkan atau input data melalui keyboard
	*Harddisk* Tempat penyimpanan data pada komputer
	Simpan *Offine* Penyimpanan arsip manual
	Connector Menunjukkan penghubung ke halaman yang sama
	Off-page Connentor Menunjukkan penghubung ke halaman yang berbeda

	Garis Alir Menunjukkan simbol dari alur system atau proses

2.16 Penelitian Terdahulu

Tabel 2.5 berikut ini merupakan penelitian terdahulu yang menerapkan Rancang Bangun Alat Uji Kelayakan Pelumas Kendaraan Bermotor Berbasis Mikrokontroler.

Tabel 2.5 berikut penelitian terdahulu

No.	Nama	Judul	Terbit/ Tahun	Abstrak
1.	Teguh Febrianto	Rancang Bangun Alat Uji Kelayakan Pelumas Kendaraan Bermotor Berbasis Mikrokontroler	UNNES Semarang/ 2012	Viskositas merupakan salah satu indikator penting untuk mengetahui pelumas dalam kondisi layak atau tidak. Viskositas dapat diukur dengan metode bola jatuh, namun metode ini mempunyai beberapa kekurangan. Dari kekurangan itu dan dibutuhkannya sebuah alat praktis dan mudah digunakan untuk mengetahui viskositas maka dibuatlah viskometer rotasi berbasis mikrokontroler. Rancang bangun alat ini menggunakan motor DC dengan *rotary encoder*, mikrokontroler ATMega16 dan LCD sebagai tampilannya. Data yang nantinya didapat adalah dari banyaknya putaran motor DC yang diukur menggunakan sensor kecepatan berupa *rotary encoder* dan arus yang terjadi di motor DC yang diukur dengan sensor arus berupa *shunt resistor*. Pengambilan data untuk penelitian ini menggunakan oli baru dan oli bekas dengan kode kekentalan

				yang sama yaitu SAE 20W-50. Hasil pengukuran dari oli tersebut adalah oli baru viskositasnya lebih tinggi dibandingkan dengan oli bekas. Semakin encer sebuah oli maka hambatan yang terjadi pada putaran motor DC semakin berkurang. Jika hambatan berkurang maka arus yang terjadi juga kecil. Namun hasil ini belum menunjukkan nilai pasti viskositas dari oli tersebut. Alat yang digunakan belum dikalibrasi dengan alat standard, karena untuk sampai dapat dikalibrasi alat ini masih perlu banyak perbaikan. Hasil pengambilan data masih berupa asumsi bahwa jika oli dalam keadaan standard berada dalam *range* pada tabel refrensi. Dari penelitian alat uji kelayakan pelumas kendaraan bermotor berbasis mikrokontroler didapatkan hasil berupa viskometer rotasi dengan tampilan LCD. Kemampuan alat ini dapat membedakan viskositas dari oli baru dan oli bekas dalam tampilan angka. Namun demikian, pada alat ini masih perlu dilakukan kalibrasi lebih lanjut untuk dapat menunjukkan hasil yang lebih akurat sesuai dengan standard yang ada. Kata Kunci: viskometer rotasi, pelumas, viskositas.