BAB II

LANDASAN TEORI

Dalam merancang dan membangun sebuah aplikasi, sangatlah penting untuk mengetahui terlebih dahulu dasar-dasar yang akan digunakan nantinya. Dasar-dasar teori tersebut akan digunakan sebagai landasan berpikir dalam pembuatan aplikasi tersebut sehingga nantinya akan terbentuk suatu aplikasi yang sesuai dengan tujuan pembuatannya.

2.1 Simulasi

Simulasi merupakan salah satu cara untuk memecahkan berbagai persoalan yang dihadapi didunia nyata. Simulasi dapat diartikan sebagai suatu system yang digunakan untuk memecahkan atau menguraikan persoalan-persoalan dalam kehidupan nyata yang penuh dengan ketidakpastian dengan atau menggunakan model atau metode tertentu dan lebih ditekankan pada pemakaian komputer untuk mendapatkan solusinya (Kakiay, 2003).

Menurut Kakiay (2003), keuntungan yang bias diperoleh dengan memanfaatkan simulasi adalah sebagai berikut:

1. *Compress Time* (Menghemat Waktu)

   Kemampuan di dalam menghemat waktu ini dapat dilihat dari pekerjaan yang bila dikerjakan akan memakan waktu tahunan tetapi kemudian dapat disimulasikan hanya dalam beberapa menit, bahkan dalam beberapa kasus hanya dalam hitungan detik.

2. *Expand Time* (Dapat Melebar-Luaskan Waktu)

   Hal ini terlihat terutama dalam dunia statistik di mana hasilnya diinginkan tersaji dengan cepat. Simulasi dapat digunakan untuk menunjukkan perubahan struktur dari suatu system nyata yang sebenarnya tidak dapat diteliti pada waktu yang seharusnya. Dengan demikian simulasi dapat membantu mengubah system nyata tersebut hanya dengan memasukkan sedikit data.

3. *Control Source of Variation* (Dapat Mengawasi Sumber-sumber yang Bervariasi)

   Kemampuan pengawasan dalam simulasi ini tampak terutama apabila analisa statistik yang digunakan untuk meninjau hubungan antara variabel bebas dengan variabel terkait faktor-faktor yang akan dibentuk dalam percobaan. Hal ini dalam kehidupan sehari-hari merupakan suatu kegiatan yang harus dipelajari dan ditanganiserta tidak dapat dipeloreh dengan cepat.

4. *Error in Meansurement Correction* (Mengoreksi Kesalahan-kesalahan Perhitungan)

   Dalam prakteknya, pada suatu kegiatan ataupun percobaan dapat saja muncul ketidak-benaran dalam mencatat hasil-hasilnya. Sebaliknya, dalam simulasi computer jarang ditemukan kesalahan perhitungan terutama bila angka-angka yang diambil dari computer secara teratur dan bebas. Komputer mempunyai kemampuan untuk melakukan perhitungan dengan akurat.

5. *Stop Simulation dan Restart* (Dapat Dihentikan dan Dijalankan Kembali)

   Simulasi computer dapat dihentikan untuk kepentingan peninjauan ataupun pencatatan semua keadaan yang relevan tanpa berakibat buruk terhadap program simulasi tersebut. Dalam dunia nyata, percobaan tidak dapat dihentikan begitu

saja. Dalam simulasi computer, setelah dilakukan penghentian maka kemudian dapat dengan cepat dijalankan kembali.

6. Easy to Replicate (Mudah Diperbanyak)

   Dengan Simulasi computer percobaan dapat dilakukan setiap saat dapat diulang-ulang. Penggunaan dilakukan terutama untuk mengubah berbagai komponen dan variabelnya, seperti dengan perubahan pada parameternya, perubahan pada kondisi operasinya, ataupun dengan memperbanyak outputnya.

Menurut Suryani (2006), model simulasi merupakan *tool* yang cukup fleksibel untuk memecahakan masalah yang sulit untuk dipecahkan dengan model matematis biasa. Model simulasi sangat efektif digunakan untuk sistem yang relatif kompleks untuk pemecahan analitis dari model tersebut. Penggunaan simulasi akan memberikan wawasan yang lebih luas pada pihak manajemen dalam menyelesaikan suatu masalah. Oleh sebab itu, manfaat yang didapat dengan menggunakan metode simulasi adalah sebagai *tool* bagi perancangan system atau pembuat keputusan, dalam hal ini manajer untuk menciptakan system dengan kinerja tertentu baik dalam tahap operational.

Simulasi dapat juga dikatakan sebagai proses perancangan model dari suatu sistem nyata dan pelaksanaannya menggunakan eksperimen-eksperimen dengan modul-modul yang bertujuan memahami tingkah laku atau untuk menyusun strategi sehubungan dengan beroperasinya sistem tersebut.

Keandalan simulasi mampu menghadapi kompleksitas permasalahan, mengukur kinerja suatu data yang bervariasi dan mampu memberikan solusi alternatif secara cepat lewat bantuan program komputer. Oleh karena itu model simulasi adalah

jawaban atas ketidakmampuan model analitis. Berikut ini adalah keterbatasan yang dimiliki model analitis:

1. Model analitis tidak mampu menyajikan karakteristik dari sistem tetapi hanya memberikan jawaban tunggal yaitu nilai optimum saja.
2. Model matematika yang digunakan pada model analitis biasanya tidak mampu menyajikan sistem nyata yang biasanya lebih komplek, kalaupun hal ini terjadi biasanya tidak mungkin diselesaikan dengan hanya menggunakan tekhnik analitis yang sudah ada.
3. Model analitis tidak mungkin digunakan untuk hal-hal yang tidak pasti dan mempunyai aspek yang dinamis (fungsi waktu).

Model simulasi dapat digunakan untuk menyelesaikan permasalahan yang komplek tersebut memiliki lima langkah pokok yang diperlukan, langkah-langkah tersebut adalah sebagai berikut:

1. Menentukan sistem atau permasalahan yang akan disimulasikan.
2. Menentukan tujuan simulasi (apa yang harus dipecahkan, dijawab, dan disimpulkan atas permasalahan yang ada) dan hal-hal lain yang mendukung terwujudnya model simulasi.
3. Mengembangkan model simulasi dan uji terhadap kebenaran proses perhitungan yang ada didalamnya.
4. Mengembangkan model simulasi dengan menentukan lamanya simulasi (dilakukan beberapa kali) dan uji.
5. Analisa hasil dari simulasi.

2.2 Antrian

Pengertian antrian menurut Ma'arif dan Tanjung (2003) adalah situasi barisan tunggu dimana sejumlah kesatuan fisik (pendatang) sedang berusaha untuk menerima pelayanan dari fasilitas terbatas (pemberi pelayanan), sehingga pendatang harus menunggu beberapa waktu dalam barisan agar dilayani.

Sedangkan menurut Heizer and Render (2006) dalam bukunya *Operation Management* yang diterjemahkan oleh Setyoningsih dan Almahdy adalah orang-orang atau barang dalam yang sedang menunggu untuk dilayani.

2.3 Pelayanan

Definisi pelayanan menurut Kotler (2002) adalah setiap tindakan atau kegiatan yang dapat ditawarkan oleh suatu pihak kepada pihak lain, yang pada dasarnya tidak berwujud dan tidak mengakibatkan kepemilikan apapun.

Menurut Assauri (1999) definisi pelayanan adalah bentuk pemberian yang diberikan oleh produsen baik terhadap pelayanan barang yang diproduksi maupun terhadap jasa yang ditawarkan guna memperoleh minat konsumen, dengan demikian pelayanan mempengaruhi minat konsumen terhadap suatu barang atau jasa dari pihak perusahaan yang menawarkan produk atau jasa.

2.4 Restoran Cepat Saji

Menurut Marsum (2000) restoran adalah suatu tempat atau bangunan yang diorganisir secara komersil, yang menyelenggarakan pelayanan dengan baik kepada semua konsumennya baik berupa makanan maupun minuman. Tujuan operasional restoran adalah untuk mencari keuntungan sebagaimana tercantum dalam definisi Prof.

Vanco Christian dari School Hotel Administration di CornellUniversity. Selain bertujuan bisnis atau mencari keuntungan, membuat puas para konsumennya pun merupakan tujuan operasional restoran yang utama.

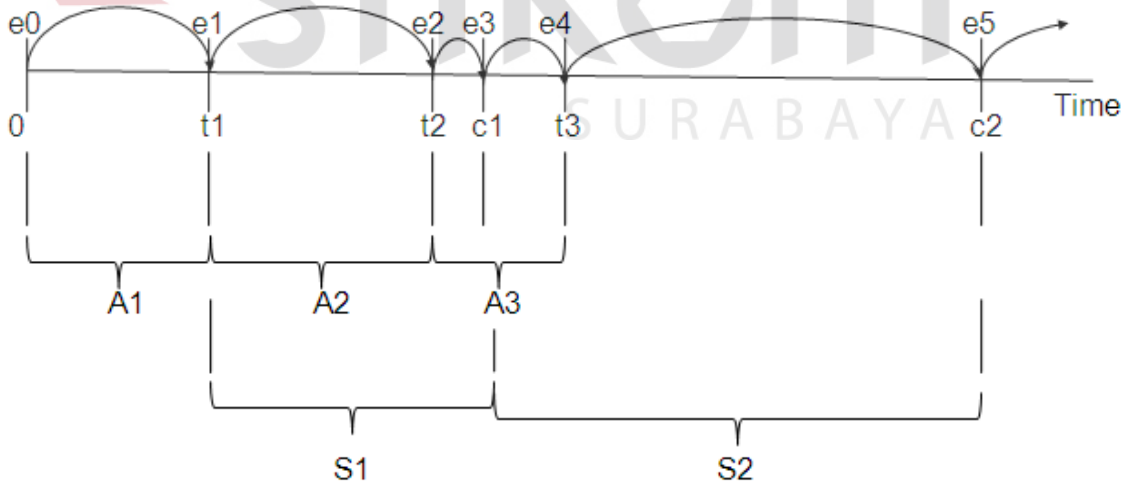
Restoran cepat saji menurut Marsum (2000) merupakan restoran dengan tempat yang tidak terlalu luas yang sifatnya tidak resmi dengan pelayanan yang cepat, dimana konsumen mengumpulkan makanan mereka diatas baki yang diambil dari atas counter (meja panjang yang membatasi dua ruangan) kemudian membawanya sendiri ke meja makan. Konsumen bebas memilih makanan yang disukai. Makanan yang tersedia umumnya hamburger , roti isi, kentang goreng, ayam goreng, nasi, dan mie.

Adapun definisi restoran cepat saji yang lain adalah salah satu industri di dunia yang berkembang dengan cepat, khususnya di area perkotaan, sebagai tanggapan terhadap gaya hidup modern dengan fleksibilitas yang semakin meningkat. Dengan adanya perubahan gaya hidup serta semakin diterimanya restoran cepat saji oleh masyarakat, maka persaingan antar gerai restoran cepat saji terhadap kualitas produk dan layanan akan semakin menonjol di masa mendatang (Goyal, Anita and Singh, N.P., 2007).

2.5 Next-Event Time Advance

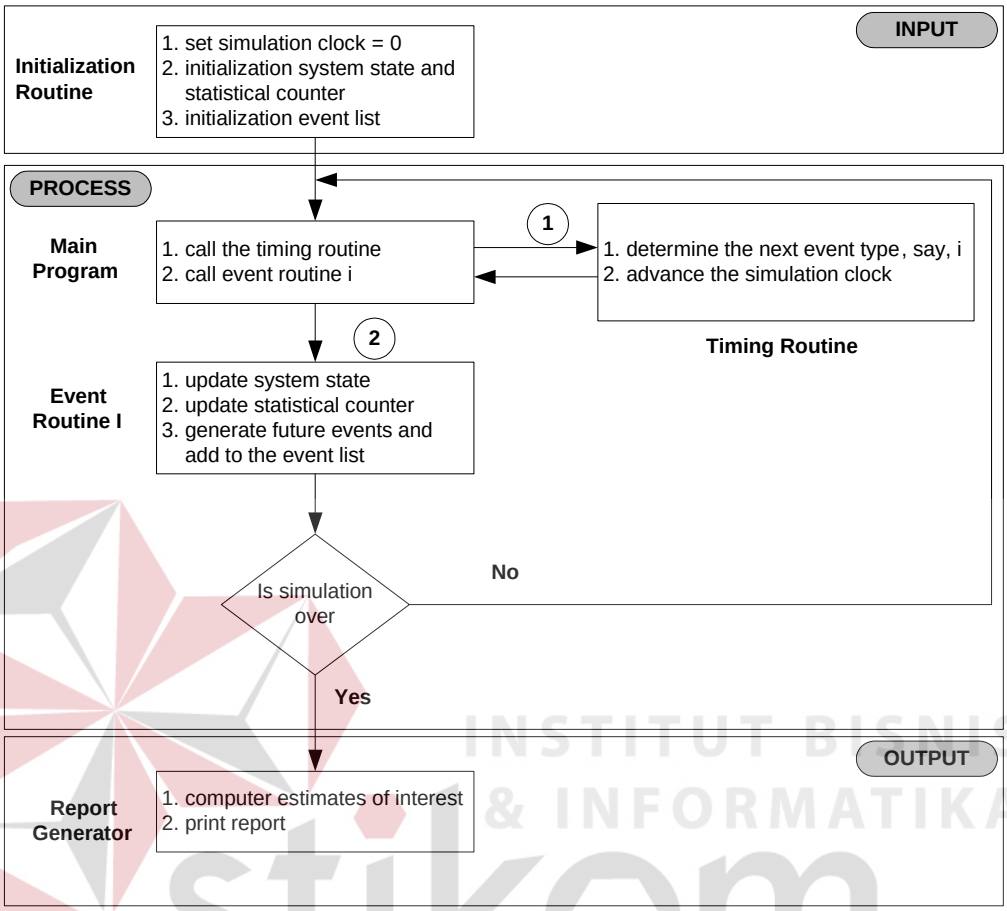
Dengan menggunakan pendekatan *Next-Event Time Advance,* waktu simulasi diinisialisasikan dengan 0 (nol) dan waktu kejadian pada event selanjutnya dideklarasikan pada saat server melayani sejumlah permintaan dan pesanan. Waktu simulasi kemudian mendekat pada waktu kejadian berikutnya yang paling dekat

(diutamakan) sebagai *event* selanjutnya, yang menunjuk keadaan sistem yang diperbaharui untuk menjelaskan fakta bahwa suatu kejadian telah berlangsung secara kontinyu. Waktu simulasi dideklarasikan menuju pada waktu yang paling dekat berikutnya berikutnya (kejadian baru), batasan dari sistem diperbaharui, dan waktu simulasi pada suatu kejadian-kejadian yang lain dilanjutkan sampai akhir. Proses pendekatan dari waktu simulasi pada satu kejadian ke kejadian yang lain dilanjutkan sampai akhir. Kondisi pelayanan terhadap konsumen tersebut dianggap terlayani dalam keadaan aman. Dengan melihat batas-batas waktu pelayanan yang selalu berubah kejadiannya, maka peristiwa ini termasuk dalam model *Discrete-Event Simulation*, karena dalam periode aktivitas dalam sistem selalu diputar dengan melompati waktu dari satu waktu kejadian tiap periode ke waktu kejadian yang lainnya.



Gambar 2.1. Ilustrasi Model *Next-Event Time Advance*
(Nurlita H.;2006)

2.6 Komponen dan Organisasi Model Next-Event Time Advance



Gambar 2.2. Alur Kontrol Menggunakan Pendekatan *Next-Event Time Advance* (Nurlita H.;2006)

Walaupun simulasi telah digunakan pada berbagai sistem yang sesungguhnya, namun hampir semua model *discrete-event simulation* memiliki sejumlah komponen-komponen yang sama serta menyediakan logika dari komponen-komponen tersebut, juga menawarkan *coding debugging* begitu juga perubahan-perubahan lebih lanjut pada program komputer yang berkaitan dengan model-model simulasi. Lebih rinci tentang komponen-komponen yang pasti ada pada *discrete event simulation models* yang menggunakan *Fixed-Increment Time Advance Approach*, adalah sebagai berikut:

1. *System state* (kondisi sistem).

   Sekumpulan kondisi (variabel) yang perlu untuk menjelaskan sistem pada waktu tertentu.

2. *Simulation clock* (jam simulasi)

   Variabel yang memberikan nilai waktu simulasi yang sedang berlangsung.

3. *Event-list* (daftar kejadian).

   Suatu daftar yang berisi waktu untuk berikutnya pada saat masing-masing jenis *event* (kejadian) yang akan terjadi.

4. *Statistical counter* (counter statistik).

   Variabel-variabel yang digunakan untuk melakukan proses statistik dan penyimpanan sebagai informasi statistik mengenai kemampuan dari sistem.

5. *Initialization routine* (rutin awal).

   Sub rutin yang diperlukan untuk mengawali model simulasi pada saat "0".

6. *Timing routine.*

   Sub rutin yang akan menetapkan kejadian berikutnya dari *event list* (daftar kejadian) dan kemudian mempercepat jam simulasi sampai waktu pada saat *event* tersebut harus terjadi.

7. *Event routine.*

   Sub rutin yang digunakan untuk meng-update (memperbarui) kondisi sistem pada saat suatu jenis *event* khusus terjadi (ada suatu kejadian untuk masing-masing jenis kejadian).

8. *Report generator.*

   Sub rutin yang digunakan untuk memperhitungkan estimasi-estimasi (dari counter statistical) pada pengukuran-pengukuran kemampuan yang diinginkan dan mencetak report pada saat akhir simulasi.

9. *Main program* (program utama).

   Program yang digunakan untuk memanggil *timing routine* untuk menetapkan kejadian berikutnya dan kemudian memindahkan (mentransfer) kontrol ke event routine yang telah ditentukan guna meng-update kondisi sistem yang tepat.

Hubungan-hubungan logika (alur kontrol) diantara komponen-komponen tersebut menunjukkan saat simulasi dimulai dengan zero ("0") yang kemudian dipanggil oleh program utama untuk pada awal rutin dan pada saat ini *simulation clock* (jam simulasi) dikondisikan pada zero, keadaan system dan *statistical counter* diawali dan *counter list* dimulai. Setelah kontrol dikembalikan pada program utama, maka kontrol memanggil *timing routine* untuk menentukan jenis *event* mana yang paling mendekati akan terjadi. Apabila suatu *event* jenis I adalah *event* yang akan terjadi, maka jam simulasi dipercepat sampai pasa waktu dimana *event* jenis I akan terjadi dan kontrol dikembalikan pada program utama. Kemudian, *main program* memanggil *event routine* I, dimana terdapat 3 (tiga) jenis aktivitas yang terjadi, yaitu: (1) *updating* kondisi system sampai kepada fakta bahwa *event* jenis I telah terjadi; (2) mengumpulkan informasi mengenai *system performance* dengan cara meng-*update statistical counter;* (3) menghasilkan waktu-waktu *event* untuk kejadian-kejadian berikutnya dan menambahkan informasi ini pada *event list* (daftar kejadian).

Setelah semua proses dilalui, maka baik *event routine* I atau program, sebuah *check* (pemeriksaan) dibuat untuk menentukan (berhubungan dengan beberapa kondisi berhenti) apakah simulasi tersebut harus dibatalkan sekarang atau tidak.

2.7 Pengujian Data

Pada beberapa eksperimen, dibutuhkan suatu proses pengambilan data secara langsung di lapangan atau diperlukan suatu pembangkitan data pada proses eksperimen yang memerlukan simulasi. Pada proses ini tentunya diinginkan adanya kesamaan antara distribusi data yang diperoleh, dengan distribusi data yang tepat secara teori. Oleh karena itu diperlukan suatu proses pengujian kecocokan distribusi.

Distribusi data ada dua macam, distribusi data yang bersifat diskrit dan distribusi data yang bersifat kontinu. Tentunya kedua macam distribusi ini akan berbeda proses pencocokan distribusinya. Untuk distribusi data yang bersifat diskrit, akan tepat jika digunakan pengujian distribusi dengan metode *Pearson's Test Goodness of Fit.* Sedangkan untuk distribusi data yang bersifat kontinu, akan tepat jika digunakan pengujian distribusi dengan metode *Kolmogorov-Smirnov*.

Karena data sampel yang didapat oleh penulis merupakan data yang bersifat kontinu, maka untuk pengujian distribusi menggunakan metode *Kolmogorov-Smirnov Normal.*

2.7.1 Pengujian *Kolmogorov-Smirnov Normal*

Pengujian bertujuan melihat tingkat kesesuaian antara fungsi distribusi hasil pengamatan dengan fungsi distribusi teoritik tertentu, dengan menetapkan suatu titik yang menggambarkan perbedaan maksimum keduanya.

1. Statistik Uji

   Menggunakan rumus seperti pada rumus 1.

2. Kriteria Penolakan

   Jika nilai $T \geq W_{1-\alpha}$, maka H_0 ditolak (tabel yang digunakan adalah tabel Kolmogorov-Smirnov).

   Langkah-langkah Pengujian :

   1) Menetapkan hipotesis awal dan hipotesis tandingan

      Hipotesis :

      H_0 : data mengikuti distribusi eksponensial

      H_1 : data tidak mengikuti distribusi eksponensial

   2) Menghitung statistik uji

      Banyaknya parameter pada distribusi eksponensial adalah β yang menyatakan nilai rata-rata. Untuk menentukan harga *F(x)* maka nilai β harus ditentukan dengan menggunakan rumus 2.

   3) Menetapkan α (taraf signifikan)

      $\alpha = 0{,}05$

   4) Menentukan daerah penolakan

      $W_{1-\alpha}$ didapatkan dari tabel *Kolmogorov-Smirnov* sesuai dengan *n* yang ada, atau dengan rumus $W_{1-\alpha} = (1{,}36/\sqrt{n}) - 0{,}002$, untuk $\alpha = 0{,}05$

5) Membuat kesimpulan

Membandingkan antara T dengan $W_{1-\alpha}$, jika $T < W_{1-\alpha}$ maka H_0 gagal tolak dan bila nilai $T \geq W_{1-\alpha}$, maka H_0 ditolak.

6) Membuat interpensi dari kesimpulan

Jika H_0 gagal tolak maka data yang diuji adalah berdistribusi normal.

2.8 Bilangan Acak Uniform

Sekali dan rutin adalah cara untuk membangkitkan bilangan acak *uniform* yang mempunyai jarak antar bilangannya adalah (0,1). Untuk mendapatkan bilangan acak *uniform* dapat menggunakan rumus :

$$Ui = ni/m \quad \text{.....(2.1)}$$

Keterangan:

Ui = bilangan acak *uniform*

Ni = data ke i

m = konstanta modulus

2.9 Distribusi Probabilitas

Dalam ketidakpastian permintaan pelanggan yang ada menimbulkan banyak kemungkinan-kemungkinan. Salah satu cara untuk memperkecil beberapa kemungkinan tersebut adalah dengan mempelajari pola dari distribusi probabilitasnya. Distribusi probabilitas teoritis yang sering digunakan dalam fungsi permintaan adalah distribusi Normal, distribusi Poisson, dan distribusi Eksponensial, sebagaimana dijelaskan oleh Tersine (1994) bahwa *"The normal, poisson, and exponential distributions have been found to be considerable value in describing*

demand functions. The normal distribution has been found describe many demand functions at the factory level; the poisson, at the retail level; and the exponential, at the wholesale and retail levels".

2.9.1 Distribusi frekuensi

Untuk dapat memahami data dengan mudah, maka baik data Kualitatif maupun data Kuantitatif harus disajikan dalam bentuk yang ringkas dan jelas.

Salah satu cara untuk meringkas data adalah dengan distribusi frekuensi, yaitu pengelompokan data ke dalam beberapa kelompok atau kelas dan kemudian dihitung banyaknya data yang masuk ke dalam tiap kelas. Distribusi frekuensi menunjukkan jumlah atau banyaknya item dalam setiap kategori atau kelas.

Dalam menentukan kelas yang digunakan pada distribusi frekuensi sebaiknya harus hati-hati. Ada tiga hal yang perlu diperhatikan dalam menentukan kelas bagi distribusi frekuensi untuk data kuantitatih, yaitu jumlah kelas, lebar kelas, dan batas kelas. H.A. Sturges pada tahun 1926 menulis artikel dengan judul: *"The Choice of a Class Interval"* dalam *Journal of the American Statistical Association,* yang mengemukakan suatu rumus untuk menentukan banyaknya kelas sebagai berikut:

$$k = 1 + 3{,}33 \log n \quad \text{.....(2.2)}$$

(Supranto; 2000)

Di mana k = banyaknya kelas

n = banyaknya nilai observasi

rumus tersebut diberi nama *Kriterium Sturges* dan merupakan suatu ancar-ancar tentang banyaknya kelas. Kemudian disarankan interval atau lebar kelas adalah sama

untuk setiap kelas, dan untuk menentukan besarnya kelas (panjang interval) digunakan rumus:

$$C = \frac{X_n - X_i}{k} \quad \text{.....(2.3)}$$

(Supranto; 2000)

Di mana c = perkiraan besarnya *(class width, class size, class length)*

k = banyaknya kelas

X_n = nilai observasi terbesar

X_i = nilai observasi terkecil

2.9.2 Distribusi normal

Distribusi normal memegang peranan yang sangat penting dalam statistic inferensial, yaitu sebagai model distribusi probabilitas. Ada 3 (tiga) alasan yang melandasi pentingnya distribusi normal, yaitu :

1. Distribusi normal merupakan model yang sangat baik untuk mendekati frekuensi dari fenomena alam dan sosial jika sampel besar. Populasi berbagai perilaku dan karakteristik alam dan social yang bersekala interval dan rasio umumnya diasumsikan berdistribusi normal.
2. Ada hubungan yang kuat antara besarnya sampel dengan distribusi rata-rata yang diperoleh dari sampel-sampel acak yang diambil dari suatu populasi yang sama. Semakin besar sampel, distribusi rata-rata sampel semakin mendekati normal.
3. Distribusi normal mendekati penghampiran (aproksimasi) yang baik terhadap distribusi teoritis lainnya yang pada umumnya lebih sulit digunakan untuk memodelkan distribusi peluang.

Model matematik yang digunakan pada distribusi normal adalah :

$$Y = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \quad \text{.....(2.4)}$$

(Element Of Stochastic Process Simulation Gottfried; Byron. S; 1984)

Di mana :

Y = ordinat pada grafik π = 3,1416 (pembulatan)

X = skor yang diperoleh e = 2,7183 (pembulatan)

μ = rata-rata populasi σ = simpangan baku populasi

Karena distribusinya kontinyu, maka cara menghitung probabilitasnya dilakukan dengan jalan menentukan luas di bawah kurvanya, sayangnya fungsi frekuensi normal tidak memiliki integral yang sederhana sehingga probabilitasnya dihitung dengan menggunakan distribusi normal standar di mana variabel normal standar di berikan rumus sebagai berikut :

$$Z = \frac{X-\mu}{\sigma} \quad \text{.....(2.5)}$$

(Element Of Stochastic Process Simulation Gottfried; Byron. S; 1984)

Di mana :

Z = *number of standard deviation from* X *to the mean of this distribution*

X = nilai tengah

μ = rata-rata (dalam pola distribusi ini didekati dengan $\bar{X}$, karena menggunakan data sampel dari populasi)

σ = *standard deviation of this distribution* (didekati dengan S karena menggunakan data sampel yang mewakili populasi)

Kenyataan hidup yang dihadapi setiap hari sering memiliki perilaku berdistribusi normal, baik dalam perhitungan nilai maupun kejadian-kejadian yang lainnya. Distribusi normal memiliki bentuk simetri dengan densitas peluang menyerupai bell (melengkung seperti gambar gunung).

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left\{-\frac{1}{2}\left[\frac{x-\mu}{\sigma}\right]\right\} \quad \text{.....(2.6)}$$

Di mana :

μ = merupakan rata-rata

σ = simpangan baku (standard deviation)

Seperti halnya dengan fungsi gamma dan fungsi poisson maka distribusi normal juga tidak dapat dianalisa dengan integral secara langsung; sehingga dapat dicoba dengan membangkitkan *generate random* menggunakan simulasi langsung. Untuk sekedar mempermudah dalam pemecahan masalah terhadap data yang berdistribusi normal di mana nilai σ = 1. Sehingga akan didapat nilai *standard normal* Z, rumusnya seperti pada rumus 9.

Sehingga persamaan diatas akan menjadi :

$$f(z) = \frac{1}{\sqrt{2\pi}} e^{-z^{0,5}} \quad \text{.....(2.7)}$$

(Element Of Stochastic Process Simulation Gottfried; Byron. S; 1984)

Sekarang dapat diketahui bahwa hasil dari kumpulan (rangkaian) sampel yang diambil menunjukkan satu atau beberapa fungsi distribusi statistic, dengan ukuran sampel yang besar (disebut *consequence of the central limit theorem)* akan menjadi distribusi normal atau bias dianggap distribusa normal.

Dalam hal khusus, bila sampel dapat menunjukkan perilaku dari sejumlah N bialngan acak U(0,1), maka isi dari *random number* tersebut :

Dari rumus ini untuk mencari Z maka jumlahkan saja sebanyak 12(0,1) dan hasilnya dikurangi dengan 6.

Selanjutnya bila dikehendaki membangkitkan bilangan acak berdistribusi normal dengan rata-rata = μ dan standard deviasi = σ maka dengan mudah bisa dicari dengan persamaan berikut:

$$X = \mu + \sigma Z \quad \text{.....(2.8)}$$

(Element Of Stochastic Process Simulation Gottfried; Byron. S; 1984)

Dalam distribusi normal standar tersebut diatas, yang pertama harus dilakukan dengan menentukan μ (jika populasi yang digunakan untuk penelitian) atau menggunakan $\bar{X}$ (penulis mengambil data berupa data sampel dari populasi yang digunakan dalam penelitian). Rumus yang digunakan dalam menemukan $\bar{X}$ tersebut dapat dihitung dengan menggunakan rumus 2.

Kemudian dalam mencari simpangan baku (σ) dapat didekati dengan S, hal tersebut dikarenakan data yang diambil merupakan data sampel bukan populasi.

$$S = \sqrt{\frac{\sum_{i=0}^{n} fi\,(Xi-\bar{X})^2}{n}} \quad \text{.....(2.9)}$$

(Supranto; 2000)

Di mana :

n = jumlah sampel dari populasi

Xi = nilai tengah

fi = frekuensi

Jadi dalam menggunakan distribusi normal nantinya, penulis dalam mengaplikasikannya akan menggunakan distribusi normal standar tidak menggunakan rumus kumulatif distribusi normal yang ada integralnya.

Untuk membangkitkan bilangan acak berdistribusi normal masih bias dengan menggunakan cara lain yaitu dengan rumus :

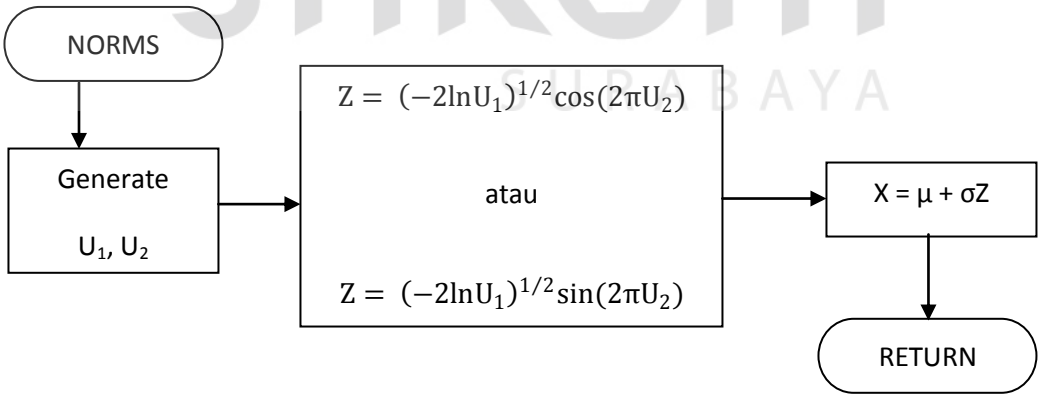
$$Z = (-2\ln U_1)^{\frac{1}{2}} \sin(2\pi U_2) \quad \text{.....(2.10)}$$

atau

$$Z = (-2\ln U_1)^{\frac{1}{2}} \cos(2\pi U_2) \quad \text{.....(2.11)}$$

(Supranto; 2000)

Kedua rumus diatas ini memberikan hasil bilangan acak yang berdistribusi standard normal. Sehingga untuk membangkitkan bilangan acak berdistribusi normal dengan rata-rata μ (didekati dengan $\bar{X}$) dan standard deviasi σ (didekati dengan S), maka alur atau flowchart penyelesaian dengan rumus tersebut adalah :



Gambar 2.3. Flowchart Perhitungan Bilangan Random Distribusi Normal

2.10 Pengukuran Penggunaan (Utilization)

Kebutuhan umum akan simulasi adalah untuk mengukur beban pada suatu entitas. Pengukuran paling sederhana adalah penentuan fraksi waktu penggunaan suatu item selama berlangsungnya simulasi. Istilah penggunaan atau pemanfaatan (utilization) dipakai untuk menggambarkan statistik ini. Untuk mengukur tingkat penggunaan, dibutuhkan rekaman sampai di saat itu adalah saat terakhir penggunaan peralatan. Nilai *utilization* (u) dapat diperoleh dengan cara membagi waktu total penggunaan dengan waktu total pengaktifan suatu item. Untuk menemukan utilitas sistem pada suatu item, jika item tersebut berjumlah lebih dari satu, maka utilitas sistem didapat dengan menghitung rata-rata dari jumlah utilitas masing-masing item.

2.11 Database Management System (DBMS)

Database adalah sekumpulan data/informasi yang tersimpan secara teratur berdasarkan criteria/kelompok tertentu, dimana criteria tersebut saling berhubungan satu sama lain. Sedangkan *Database Management Sistem (DBMS)* adalah kumpulan program perangkat lunak (software) yang memperbolehkan user untuk membuat dan memelihara database. Semua *DBMS* tersebut disatukan oleh *Relational Database Management System (RDBMS).*

2.12 *Random Seed* (Bibit Bilangan Acak)

Metode ini menggunakan hubungan kongruensial rekursif, yang mana menggunakan metode ini peruntukkan dalam mengendalikan *random number* yang akan dibangkitkan pada saat proses simulasi setelah uji distribusi. Berlakunnya *random seed* untuk semua distribusi yang hasilnya terima H_0, baik itu distribusi normal, eksponensial, poisson, dan empiris. Berikut rumus metode kongruensial rekursif.

$$n_i = a * n_{i-1} (\text{mod } m) \quad \text{.....(2.12)}$$

Dimana :

n_i & n_{i-1} : integer acak

a (pengali) : nilai yang telah ditetapkan

m (modulus) : nilai yang telah ditetapkan

perolehan rumus diatas memerlukan beberapa penurunan yang tidak perlu penulis tuangkan dalam buuku ini.

Untuk mendapatkan n_i yang bersifat acak tersebut maka nilai-nilai m, n_0 dan a ditetapkan sedemikian hingga dengan menggunakan aturan yang telah ditentukan, berikut langkah dalam menentukan perolehan nilai m, n_0, dan a :

1. Pertama m ditentukan nilainya sebesar mungkin sesuai dengan kemampuan computer bila menggunakan computer. Dengan w bits setiap word (kemampuan daya tampung memory untuk menerima proses penyimpanan data). Persamaan dalam penentuan nilai m adalah sebagai berikut :

$$m \cong 2^{w-1} \quad \text{.....(2.13)}$$

(Watson; 1981)

2. Kedua a ditentukan, dimana proses penentuan niilai a ini harus disesuaikan sedemikian hingga yang mana agar ada hubungan (ketertarikan) antara *random number* yang dibangkitkan mulai dari nilai *random* yang paling kecil, rumus berikut memungkinkan untuk dapat digunakan :

   $a \cong 2w - 1$ .....(2.14)

   $a \equiv \pm 3 \pmod 8$ .....(2.15)

   (Watson; 1981)

3. n_0 dapat juga disebut dengan *seed* (bibit) diman n_0 adalah bilangan integer positif yang ganjil (dengan m dan a yang sama), jika n_0 (bibit) yang diberikan nilainya berbeda, maka tetap didapat jumlah bilangan acak yang sama. Dalam arti bahwa dengan bibit yang dimasukkan sebagai inputan sama jika yang dibangkitan bilang acaknya sama hasil yang diperoleh juga sama, dengan bibit yang sama jumlah bilangan acak yang dibangkitkan berbeda hasil yang diperoleh dalam himpunan bilangan acak akan berbeda.

   Bila diikuti aturan-aturan ini, maka secara tidak langsung akan diperoleh bilangan acak dengan periode m/4 dengan n_i yang didapat berharga mulai dari 1 sampai dengan m-1. Sedangkan untuk mendapatkan bilangan acak yang terdistribusi secara uniform (0,1) akan ditampung dalam memori dengan nilai pointer seperti berikut : u1, u2,…,un; maka bagi saja masing-masing n_i dengan m. Rumus yang digunakan untuk memproses langkah tersebut adalah :

   $ui = \frac{n_i}{m}$ .....(2.16)

   (Watson, 1981)

Dan perlu diingat lagi bahwa keberadaan *random seed* sangat menguntungkan bagi user pada saat melakukan proses ulang terhadap data sampel yang sama setelah komputer dimatikan, dimana hasil yang di peroleh ketika *random seed* sebelumnya akan tersimpan dalam memori computer tanpa harus membangkitkan ulang untuk melihat hasilnya. Atau nilai random yang dihasilkan tidak berubah-ubah pada saat sistem atau aplikasi yang berhubungan dengan simulasi sedang dijalankan. Dari *random seed* ini proses simulasi dapat terkendali dengan baik tanpa selalu melakukan ulang terhadap pembangkitan bilangan acak setiap computer dijalankan.

Random seed pada tugas akhir ini sesuai dengan perangkat keras yang digunakan yaitu dengan rumus umum :

Rs = - 2^{w} sampai dengan 2^{w-1} .....(2.17)

Dimana :

Rs = random seed

w = word size

2.13 *Entity Relationship Diagram*

Entity Relationship Diagram, atau yang lebih dikenal dengan nama ERD, digunakan untuk mengimplementasikan, menentukan, dan mendokumentasikan kebutuhan-kebutuhan untuk sistem pemrosesan *database*.

2.14 *Data Flow Diagram*

Menurut Kristanto (2004), *Data Flow Diagram* (DFD) adalah suatu model logika data atau proses yang dibuat untuk menggambarkan darimana asal data dan kemana tujuan data yang keluar dari sisem, di mana data tersebut disimpan, proses

apa yang menghasilkan data tersebut dan interaksi antara data yang tersimpan, dan proses yang dikenakan pada data tersebut.

Data Flow Diagram merupakan suatu metode pengembangan sistem yang terstruktur (*structure analysis and design*). Penggunaan notasi dalam data flow diagram sangat membantu untuk memahami suatu sistem pada semua tingkat kompleksitas. Pada tahap analisi, penggunaan notasi ini dapat membantu dalam berkimunikasi dengan pemakai sistem untuk memahami sistem secara logika.

Di dalam *data flow diagram*, terdapat empat simbol yang digunakan yaitu *process, external entity*, data *store*, dan *data flow*. Simbol *process* digunakan untuk melakukan suatu perubahan berdasarkan data yang dimasukkan dan menghasilkan data dari perubahan tersebut.

2.15 *System Flow*

Menurut Jogiyanto (2005) *system flow* adalah bagan yang menunjukkan arus pekerjaan secara menyeluruh dari suatu sistem di mana bagan ini menjelaskan urutan prosedur-prosedur yang ada dalam sistem dan biasanya dalam membuat *system flow* sebaiknya ditentukan pula fungsi-fungsi yang melaksanakan atau bertanggung jawab terhadap sub-sistem yang ada.

Terdapat berbagai macam bentuk simbol yang digunakan untuk merancang sebuah desain dari sistem, diantaranya adalah *terminator, manual operation, document, process, database, manual input, decision, off-line storage, on-page reference,* dan *off-page reference.*

2.16 *Testing*

Menurut Romeo (2003), *testing* adalah proses pemantapan kepercayaan akan kinerja program atau sistem sebagaimana yang diharapkan. Testing *software* adalah proses mengoperasikan *software* dalam suatu kondisi yang dikendalikan untuk verifikasi, mendeteksi error dan validasi. Verifikasi adalah pengecekan atau pengetesan entitas-entitas, termasuk *software*, untuk pemenuhan dan konsistensi dengan melakukan evaluasi hasil terhadap kebutuhan yang telah ditetapkan. Validasi adalah melihat kebenaran sistem apakah proses yang telah dituliskan sudah sesuai dengan apa yang dibutuhkan oleh pengguna. Deteksi *error* adalah testing yang berorientasi untuk membuat kesalahan secara intensif, untuk menentukan apakah suatu hal tersebut terjadi bilamana tidak seharusnya terjadi atas suatu hal tersebut tidak terjadi. *Tase case* merupakan suatu tes yang dilakukan berdasarkan pada suatu inisialisasi, masukan, kondisi ataupun hasil yang telah ditentukan sebelumnya. Adapun kegunaan dari *tase case* ini, adalah sebagai berikut:

1. Untuk melakukan testing kesesuaian suatu komponen terhadap desain *White Box Testing.*
2. Untuk melakukan testing kesesuaian suatu komponen terhadap spesifikasi *Black Box Testing.*

2.16.1 *White Box Testing*

Menurut Romeo (2003), *white box testing* adalah suatu metode desain *tase case* yang menggunakan struktur kendali dari desain prosedural. Seringkali *white box testing* diasosiasikan dengan pengukuran cakupan tes, yang mengukur persentase

jalur-jalur dari tipe yang dipilih untuk dieksekusi oleh *test cases. White box testing* dapat menjamin semua struktur *internal* data dapat dites untuk memastikan validasinya.

Cakupan pernyataan, cabang dan jalur adalah suatu teknik *white box testing* yang menggunakan alur logika dari program untuk membuat *test cases.* Alur logika adalah cara dimana suatu bagian dari program tertentu dieksekusi saat menjalankan program. Alur logika suatu program dapat dipresentasikan dengan *flow graph.*

2.16.2 *Black Box Testing*

Menurut Romeo (2003), *black box testing* dilakukan tanpa adanya suatu pengetahuan tentang detail struktur *internal* dari sistem atau komponen yang dites, juga disebut sebagai *functional testing. Black box testing* berfokus pada kebutuhan fungsional pada *software,* berdasarkan pada spesifikasi kebutuhan dari *software.*

Dengan adanya *block box testing,* perekayasa *software* dapat menggunakan kebutuhan fungsional pada suatu program. *Black box testing* dilakukan untuk melakukan pengecekan apakah sebuah *software* telah bebas dari *error* dan fungsi-fungsi yang diperlukan telah berjalan sesuai dengan yang diharapkan.

2.17 Skala *Likert*

Angket atau kuisioner adalah daftar pertanyaan yang diberikan kepada orang lain yang bersedia memberikan respon, sesuai dengan permintaan pengguna. Tujuan dari menyebarkan angket adalah mencari informasi dari responden tanpa khawatir bila responden memberikan jawaban yang tidak sesuai dengan kenyataan (Riduwan, 2005).

Menurut Husein (2003), skala *likert* berhubungan dengan pernyataan seseorang terhadap sesuatu. Skor pada skala *likert* berarah *positif* dan *negatif*. Skala *likert* digunakan untuk mengukur sikap, pendapat, dan persepsi seseorang atau kelompok tentang kejadian atau gejala sosial.

Perhitungan skor penilaian untuk setiap pertanyaan (QS) didapatkan dari jumlah pengguna (PM) dikalikan dengan skala nilai (N). Jumlah skor tertinggi (STtot) didapatkan dari skala tertinggi (NT) dikalikan jumlah pertanyaan (Qtot) dikalikan total pengguna (Ptot). Sedangkan nilai persentase akhir (Pre) diperoleh dari jumlah skor hasil pengumpulan data (JSA) dibagi jumlah skor tertinggi (STtot) dikalikan 100%.

Rumus skala *likert*:

$$QS(n) = PM \times N \quad \text{.....(2.18)}$$

$$STtot = NT \times Qtot \times Ptot \quad \text{.....(2.19)}$$

$$Pre = JSA / STtot \times 100\% \quad \text{.....(2.20)}$$

Keterangan:

QS(n) = Skor pertanyaan ke-n

PM = Jumlah pengguna yang menjawab

N = Skala nilai

STtot = Total skor tertinggi

NT = Skala nilai tertinggi

Qtot = Total pertanyaan

Ptot = Total pengguna

Pre = Persentase akhir (%)

JSA = Jumlah skor akhir

Analisis dilakukan dengan melihat persentase akhir dari proses perhitungan skor. Nilai persentase kemudian dicocokkan dengan criteria interpretasi skor yang dapat dilihat pada table 2.1.

Tabel 2.1 Keterangan Nilai

Nilai	**Keterangan**
0% - 20%	Sangat Kurang
21% - 40%	Kurang
41% - 60%	Cukup
61% - 80%	Baik
81% - 100%	Sangat Baik