PANDUAN PRAKTIKUM

SIMULASI KOMPUTER DENGAN SOFTWARE ARENA 14.0

Disusun oleh:

Utaminingsih Linarti, S.T., M.T.

Program Studi Teknik Industri

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS AHMAD DAHLAN

JANUARI, 2020

KATA PENGANTAR

Assalamualllaikum Wr. Wb.

Maha Suci Allah SWT yang telah menciptakan manusia dari segumpal darah dan mengajarinya berbagai ilmu pengetahuan dengan perantara kalam. Alhamdulillahirobil' alamin segala puji hanya milik Allah SWT, Dialah menuntun semua makhluk atas kehendaknya untuk melaksanakan ketentuan yang telah digariskan-Nya.

Sholawat beserta salam kita ucapkan kepada junjungan kita Nabi Muhammad SAW, yang telah menyebarkan, memperjuangkan islam hingga kami dapat memahaminya dan percaya sampai sekarang.

Praktikum Simulasi Komputer di Teknik Industri Universitas Ahmad Dahlan dimaksudkan untuk memberikan pengetahuan kepada mahasiswa tentang pengembangan logika berfikir dengan bantuan software ARENA 10.0.

Modul ini dimaksudakan untuk membantu praktikan dalam memahami konsep-konsep secara garis besar Simulasi Komputer. Untuk pengembangan dan detail dari praktikum ini, praktikan dapat menggunakan refrensi lain yang sesuai.

Kami menyadari bahwa modul Praktikum Simulasi Komputer ini jauh dari sempurna. Untuk saran, kritik yang bersifat membangun akan mendapat tempat yang pantas untuk kami jadikan refrensi dalam menyusun modul serupa dimasa yang akan datang.

Akhirnya penyusun berharap semoga modul ini dapat memberikan manfaat bagi semua pihak, Aamiin.

Wassalamualllaikum Wr. Wb.

Yogyakarta, 07 Desember 2020

Penyusun

DAFTAR ISI

MODUL I
PENGENALAN ARENA I (FITTING DATA)

A. Tujuan Praktikum

Mengenalkan Rena sehingga praktikan dapat menyusun model simulasi dengan *Software Arena 10,0* dan dapat melakukan *Fitting* Data (Menentukan distribusi data input).

B. Pengenalan Bahasa Pemrograman Simulasi

ARENA 10.0, selanjutnya disebut ARENA adalah salah satu program simulasi yang dapat dibilang merupakan suatu evolusi dari bahasa pemrograman yang lebih dahulu lahir. Dimulai dari hadirnya bahasa pemrograman FORTRAN pada tahun 1950 – 1960-an, publik menggunakan bahasa pemrograman tersebut untuk membuat program simulasi untuk sistem yang kompleks dan bahasa pemrograman FORTRAN sangat mendukung pembuatan program simulasi secara umum. Setelah *booming* simulasi, maka bermunculan bahasa pemrograman yang menawarkan keunggulan-keunggulan yang dimilikinya. GPSS-PC, SIMSCRIPT, SLAM, SIMAN, POWERSIM dan lain sebagainya adalah sebagian dari bahasa pemrograman yang ada saat ini.

Bahasa simulasi dikenal istilah *event orientation* dan *process orientation*. *Event orientation* melihat sebuah simulasi dari sisi kejadian yang menimpa sistem sedangkan *process orientation* melihat simulasi dari sisi perjalanan entiti yang terkait. Sebagai ilustrasi : Dalam sebuah sistem antrian. *event orientation* melihat kedatangan entiti , proses dan kepergian entiti sebagai hal utama yang diamati kemudian mencatatnya secara statistik sedangkan *process orientation* melihat entiti datang dan masuk kedalam sistem kemudian ia menunggu dalam antrian lalu diproses dan keluar. Atau secara gampang dapat dikatakan *Event orientation* adalah seorang profesor dalam sebuah simulasi sedangkan *Process Orientation* adalah seorang seniman.

Namun demikian bukan berarti salah satu dari orientasi tersebut adalah benar dan yang lainnya salah. Keduanya memiliki peran yang sangat penting dalam mempelajari karakteristik sebuah sistem dan keduanya juga sangat membantu seorang analyst dalam membuat / menetapkan sebuah keputusan yang berkaitan dengan sistem yang diamati tersebut. Orientasi event memberikan data–data statistik mengenai kejadian kejadian pada

sebuah sistem dan orientasi proses memberikan pemahaman yang lebih mengenai aliran / perjalanan entiti yang bersangkutan seperti membaca sebuah flowchart.

1. Bahasa Pemrograman ARENA 10.0

Arena adalah sebuah program penyusun model dan juga merupakan simulator. Arena merupakan percampuran dari 2 kategori diatas, kombinasi antara kemudahan pemakaian yang dimiliki high level program dan flesibilitas/kelenturan yang menjadi ciri *general purpose simulation language*.

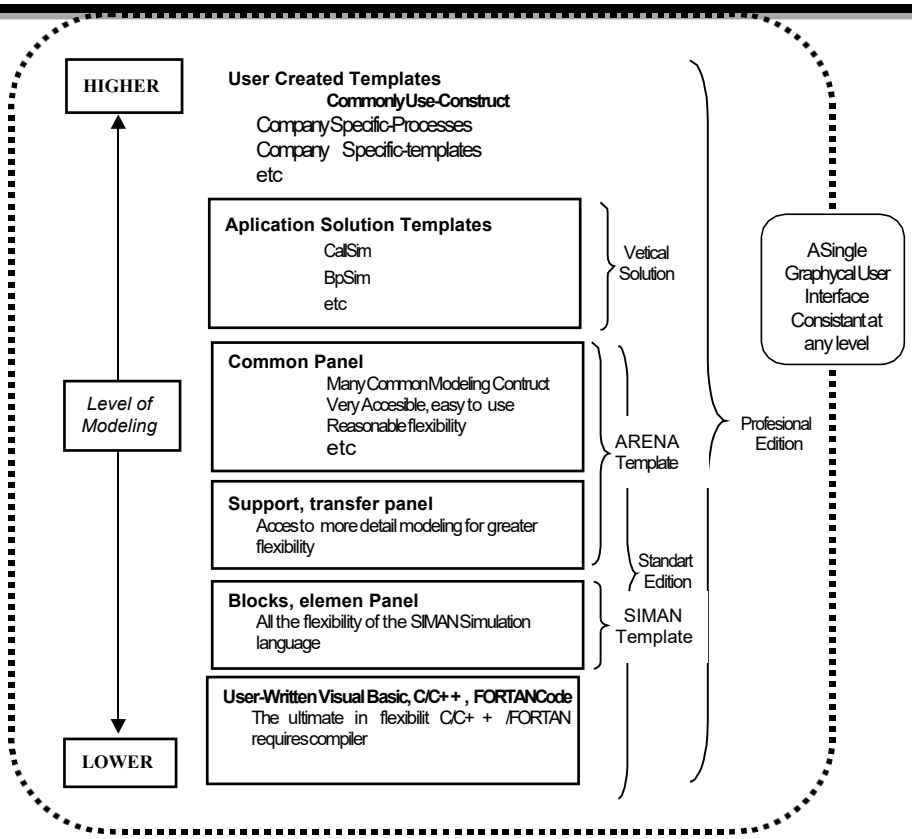
Arena masuk dalam kategori high level program karena ia bersifat sangat interaktif, pengguna dapat membangun sebuah model hampir sama mudahnya dengan membuat poster dengan menggunakan Corel Draw atau membangun flowchart dengan Visio. Hal yang membedakan hanyalah, dalam arena dibutuhkan pengetahuan mengenai sistem yang akan diamati sebelum memodelkannya.

Sedangkan predikat general purpose-pun disandangnya karena dengan Arena pengguna dapat membangun model, templetes dan bahkan pengguna dapat membuat sendiri modul jika diperlukan dengan mengunakan bantuan program seperti Visual Basic, FORTRAN dan bahkan C/C++. Dalam profesional edition, Arena memfasilitasi pengguna yang ingin membangun sendiri modul dan templetenya.

Arena secara lugas menggabungkan kedua orientasi tersebut. Disatu sisi ia memodelkan sistem dengan *process orientation* dan disisi yang lain ia memberikan informasi mengenai kejadian dalam sistem secara *event orientation*.

Kemampuan yang diperoleeh dengan menggunakan ARENA adalah:

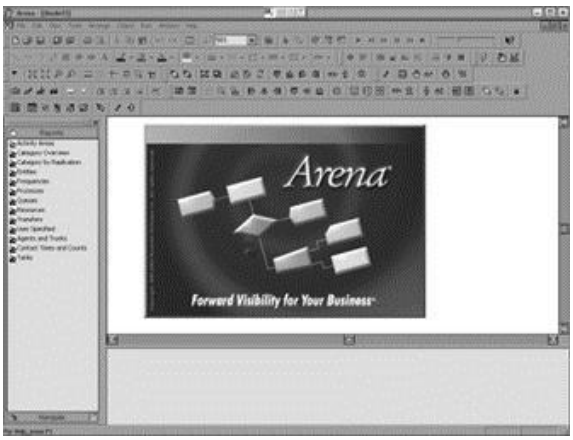
a. *Interface* yang sangat interaktif, maka *analyst* mendapatkan kemudahan dalam penggunaan Arena terutama dalam pemodelan sistem dan analisa hasil simulasi.
b. Beragamnya modul dan blok yang ada pada Arena membawa fleksibilitas yang sangat besar dalam membangun model yang sesuai dengan sistem sesungguhnya yang biasanya ada pada GPSL.
c. Selain itu, Arena sangat cocok dalam memodelkan dan mensimulasikan sistem manufaktur seperti : *material handling, inventory, quality control, bottleneck analysis*, dsb maupun industri jasa seperti : Perbankan, Rumah sakit, *Order fulfillment*.



Gambar : Struktur Hierarkis ARENA

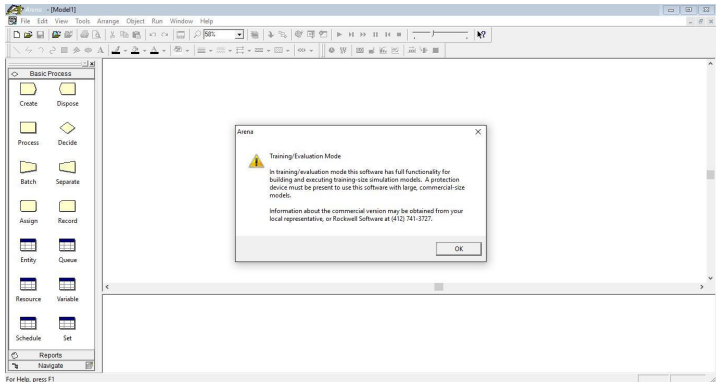
2. Bekerja Memulai ARENA 10.0

Sebelum melangkah lebih jauh, hal penting yang harus dicatat adalah : kita akan banyak menggunakan mouse, jika terdapat instruksi “click” atau “double click”, maka yang dimaksud adalah kita menggunakan tombol kiri mouse (primary button, kondisi normal). Masuklah pada sistem operasi komputer (windows/NT) dimana software arena telah terinstall, selanjutnya double click icon arena yang ada, atau membuka arena dari start menu.



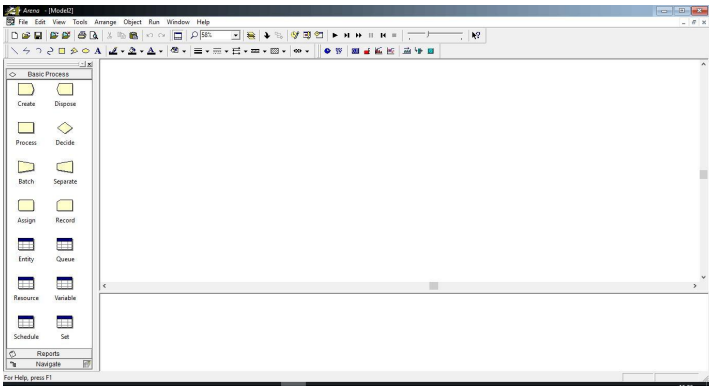
Gambar : Tampilan Awal *Software* ARENA 10.0

Setelah masuk pada Arena maka akan muncul jendela dialog yang menerangkan bahwa software ini merupakan *trainimg/evaluation mode*. Pada menu ini akan diberikan info untuk *commercial version / Rockwell Software* dapat melalui nomor yang tertera. Klik OK pada kotak bawah pada kotak Arena maka akan muncul menu utama.



Gambar : Menu/Tampilan Pertama ARENA 10.0

Secara otomatis akan setelah tulisan OK diklik akan langsung masuk lembar 1 dengan nama Model 1 dengan ektensi .doe jika disimpan. Untuk membuat sebuah lembar baru dengan mengklik icon new atau file – new ( atau CTR + N). Nama tersebut dapat diubah pada saat menyimpan model. Tampilan dapat diperbesar dengan menekan  disudut kanan atas.



Gambar : Tampilan untuk Memulai Arena 10.0

Selanjutnya dengan menggunakan perintah-perintah yang akan diuraikan lebih lanjut pada modul-modul berikutnya, dimulai memasukan elemen-elemen Arena 10.0. Setelah semua komponen dibuat maka file disimpan dengan mengklik icon save atau file – save (CTR+S)

3. *Fitting* Data

a. Pengenalan

Input Analyzer disajikan sebagai komponen standar *software* ARENA. Alat yang serbaguna dan kuat ini digunakan untuk menentukan kualitas kecocokan dari fungsi distribusi probabilitas ke *input* data. Mungkin juga digunakan untuk kecocokan fungsi distribusi yang spesifik ke *file* data yang mengizinkan kita untuk perbandingan fungsi distribusi atau menampilkan efek perubahan pada parameter untuk distribusi yang sama. Sebagai tambahan *Input Analyzer* dapat menghasilkan satuan data random yang kemudian bisa dianalisis menggunakan *software's distribution fitting features*.

Data *file* yang diproses oleh *Input Analyzer* secara khas menghadirkan interval waktu yang dihubungkan dengan suatu proses acak. Sebagai contoh *Input Analyzer* mungkin digunakan untuk menganalisa suatu set dari waktu interval, satu set dari proses *time*, atau waktu antara kegagalan sistem yang berurutan. Distribusi topik menggunakan distribusi-distribusi yang ada pada ARENA.

b. *Fitting a specific distribution* ke data

Setelah *file* data terisi dan ditampilkan sebagai histogram ke dalam data *fit window*, langkah selanjutnya adalah mencocokkan fungsi distribusi probabilitas ke data. Untuk melakukannya pertama memilih *fit menu item*. Penampilan menu *drop-down* semuanya dari fungsi distribusi *available*. Bahwa distribusi *poisson* akan menjadi tidak aktif kecuali jika deteksi *Input Analyzer* mendeteksi semua data bilangan bulat.

Pilihan berikutnya fungsi distribusi probabilitas yang diinginkan *Input Analyzer* kemudian akan menentukan parameter yang cocok dengan fungsi distribusi ke data. Secepat kalkulasi *curve fitting* menjadi komplit. Hasil fungsi *density* probabilitas diletakkan di bagian paling atas dari histogram (dalam kasus distribusi empiris, fungsi distribusi kumulatif ditunjukkan sebagai gantinya) informasi penandaan *curve fitting*, termasuk ungkapan yang bias tercakup di dalam model ARENA, ditunjukkan di bagian bawah dari *window*.

Untuk informasi yang lebih detail, termasuk tabulasi dari probabilitas *density* (histogram dan fungsi distribusi probabilitas) dan koresponden kumulatif distribution, disiapkan dalam bentuk suatu *text file* yang dituliskan ke dalam

default directory dengan nama *file* [distribution]. *Out*, di mana [distribution] adalah nama yang bersifat *eksponen* dipilih, informasi akan dituliskan ke dalam *file* dengan nama *eksponen.out*. Sebagai tambahan, suatu lintasan dari semua distribusi yang dicoba ke dalam data *file* anda (e.g. *myfile*, dst) akan tertulis untuk suatu *file* ringkasan dari nama yang sama dengan *ekstention.sum* (e.g.*myfile.sum*). *Text file* ini mungkin terlibat dalam *Input Analyzer* dengan memilih *option window* dan klik pada *curve fit summary*. Informasi lebih lanjut dari fungsi ini, mengacu pada *viewing tabular data section*.

c. *Fitting semua* distribusi ke data (*fit all command*, *fit menu*)

Sebagai tambahan terhadap semua pendukung distribusi oleh ARENA, menu *fit drop-down* meliputi : *option fit all*. Pemilihan *option* ini menyebabkan semua fungsi distribusi yang bisa diterapkan untuk dicoba ke data. Distribusi kemudian dipilih dari yang terbaik ke yang terburuk, berdasarkan atas nilai-nilai dari kesalahan kuadrat masing-masing. Hasil fungsi dari pemilihan yang terbaik kemudian akan ditampilkan ke dalam *layer* bagian paling atas histogram data. Suatu waktu tunda mungkin akan diamati, jika option ini dipilih terutama jika data *file* relatif besar.

d. File data fit dan windows

Topik ini meliputi di dalam data *fit window*, informasi mengenai *file* data ditampilkan di bawah histogram yang menghadirkan nilai-nilai dalam *file* data. Ini meliputi informasi seperti tipe data, banyaknya *point data*, banyak interval, *histogram range*, contoh *mean*, contoh *standard deviation*, dan nilai data maksimum dan minimum.

Jika nilai data ditampilkan sebagai *real world* (nilai yang sebenarnya) batas histogram awalnya ditentukan sebagai nilai integer, hanya di luar poin-poin data maksimum dan minimum. Contoh batas histogram paling bawah akan menjadi integer terbesar yang tidak melebihi poin data + data terkecil, dan batas teratas akan menjadi data terkecil ≥ point data terbesar. Jika data point minimal dan maksimal ditafsirkan sebagai *integer valued*, batas histogram yang disesuaikan sedemikian rupa sebagai sedikit meluas di luar nilai minimal dan maksimal. Jika hasil pemilihan ini menyebabkan banyak ruang kosong (i.e. *set histogram* memiliki nilai tidak teramati), batas disesuaikan sedemikian rupa sehingga mendekati poin data minimal dan maksimal. Kondisi ini bisa terjadi pada *file* data

yang mempunyai variasi kecil. Batas histogram pada *file* data bisa diubah secara manual (mengacu pada *changing parameter's* perubahan parameter). Selama data tidak pergi di luar nilai-nilai yang mula-mula ditentukan. Biasanya kamu perlu berhati-hati untuk menghindari batas yang relatif jauh dari poin data terkecil atau terbesar.

Sekali batas histogram telah ditentukan, maka interval akan dihasilkan secara otomatis. Untuk data *real valued* → nilai sebenarnya, banyaknya data histogram ditentukan dengan akar dua dari jumlah poin data, dengan batasan jumlah interval tidak kurang dari 5 dan tidak lebih dari 40.

Untuk data integer – *valued*, banyaknya interval sama dengan beda antara nilai data maksimal dan minimal, tambah satu. Dengan kata lain, akan ada suatu interval terpisah untuk masing-masing integer berurutan, berkisar antara nilai data terkecil ke yang terbesar. Jika, bagaimanapun jarak/perbedaan dari poin data integer adalah lebih besar dari 99 (mencakup atau berisi labih dari 100 interval), maka data akan dianggap *real valued* dan kemudian diproses secara *real valued*.

e. *Data fit option* (*option*, *menu data fit*)

Option, *menu* data *fit* membuka dialog yang berisi 3 *cheek box* : *include KS text*, *output summary files*, dan *auto data translation*. Fungsi dari tiap bagian diuraikan di bawah :

1) Include KS text

Option (pilihan ini menentukan ada atau tidaknya suatu *text kolmogorrov-smirnov* (k)) pada bagian akhir. Dari tiap *curve fit* atau kurva kecocokan. Dengan *default*/tidak dihadirkan, *feature* ini akan aktif saat *file* data pertama kali terisi. Untuk menonaktifkannya, dengan cara sederhana yaitu dengan klik pada *option* ini. Karena *feature* ini dapat memperlambat perhitungan pada saat bekerja dengan data *file* yang besar, kemu bisa menghilangkannya jika kamu berencana mengadakan percobaan secara ekstensif dengan beberapa distribusi kemungkinan yang berbeda.

2) Output summary file

Option/pilihan ini menentukan ada atau tidaknya suatu *file text* yang dihasilkan ketika suatu fungsi distribusi dicoba ke data dengan *default*/menghilangkannya. *Feature* ini akan aktif saat *file* data pertama kali terisi. Penghilangan kreasi atau ciptaan dari *file* ringkasan mungkin diperlukan

jika kamu berencana untuk mengadakan eksperimen secara ekstensi dengan beberapa distribusi probabilitas yang berbeda.

3) *Auto data translation*

Pengaruh *option* ini hanya pada *erlang*, eksponensial, *gamma*, *lognormal,* dan distribusi *weibull*. Saat *feature* ini aktif, data yang asli dikenal sebagai Xi ≥ X0, secara otomatis diterjemahkan sebagai suatu *file* data baru, Ui di mana Ui = Xi – X0 ≥ 0. Dengan begitu, fungsi distribusi dimasukkan dengan nilai-nilai terjemahan bukannya data yang asli. *Feature* ini menyakinkan kalau/bahwa lima fungsi distribusi ini hanya akan dimasukkan dengan data *non* negatif, seperti yang diperlukan. Dengan *dedault*, *feature auto data translation* akan katif saat data *file* pertama kali terisi. Sekali option ini *default vated*, erlang, eksponensial, gamma, lognormal, dan distribusi weibull akan menjadi tidak aktif jika ada nilai-nilai di dalam *file* data yang dimasukkan ditentukan berbentuk negatif.

f. Modifying parameter

1) Histogram, *command*/ histogram perintah (*option*, *menu parameter*)

Di dalam *option* menu parameter, pemilihan *option* histogram akan membuat dialog *appear* yang mengijinkan kamu untuk mengubah banyaknya interval, batas terbawah (mengabaikan semua data yang ada di bawah batas), dan batas teratas (mengabaikan semua data yang ada di atas batas). Banyaknya interval harus tidak kurang dari 5 dan tidak lebih dari 40. Sebagai tambahan, batas paling bawah dari histogram harus ≥ dari integer terbesar yang tidak melebihi nilai data minimum di dalam *file*. Batas paling atas histogram kemudian harus ≤ integer terkecil yang sama dengan atau lebih besar dari nilai data maksimal di dalam *file*.

Jika parameter histogram diubah setelah fungsi distribusi tidak dipilih, *curve-fit* baru akan secara otomatis dilaksanakan menggunakan parameter histogram yang baru.

2) Distribusi *command*/distribusi perintah (*option*, *menu parameter*)

Option distribusi hanya akan menjadi aktif sekali fungsi distribusi (selain empiris) telah dimasukkan ke dalam data. Jika memilih *option* ini, dialog akan nampak dan membiarkan merubah parameter fungsi distribusi.

Sekali parameter fungsi distribusi telah diubah, evaluasi baru dari perbaikan dari distribusi *fit* (pencocokan distribusi-distribusi) ke data segera dilakukan.

g. *Viewing tabular data*

1) *Input data command*/masukan data perintah (*menu window*)

Input data dalam *menu window* menyebabkan *file* data yang dimasukkan akan ditampilkan di dalam *text scroll window*. *Feature* ini mengizinkan untuk melihat data *input* actual didasarkan berbagai histogram dan *curve-fit*. Lihat juga pencetakan data atau grafis.

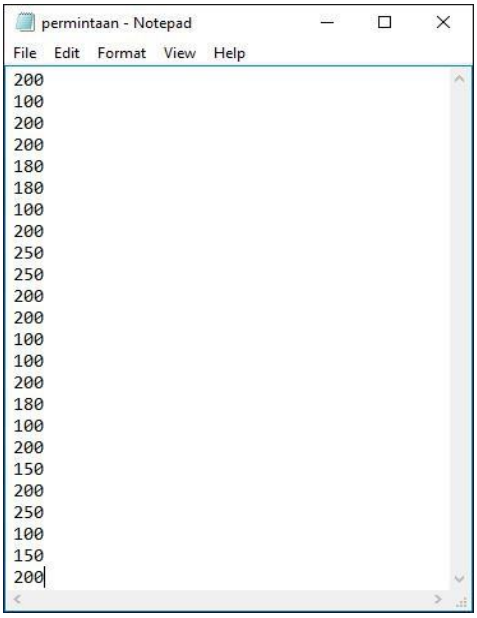
2) *Curve fit summary command*/ringkasan perintah atau kurva kecocokan (*menu window*)

Item curve-fit summary menyebabkan suatu dialog untuk memunculkan, mendaftarkan semua fungsi distribusi yang tersedia. Itu hanya fungsi distribusi yang telah dicoba ke dalam file data akan dimungkinkan. Jika fungsi distribusi memungkinkan telah dipilih dari menu ini, muatan atau isi dari *file* ringkasan yang berkesesuaian telah ditampilkan di dalam *text scroll window*. *File* ringkasan ini menyediakan kumpulan paling lengkap dari gambaran informasi *curve-fit*. Lihat juga pembuatan tabel atau data grafis.

C. Penggunaan *Tools Input Analyzer*

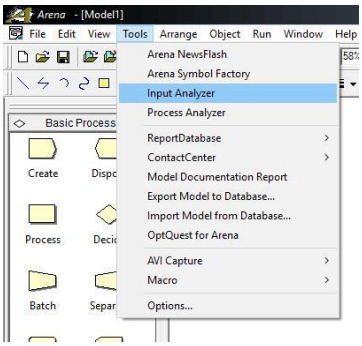
Berikut adalah langkah-langkah melakukan fitting data:

1. Masukan data berikut ke *Notepad* kemudian *save file* di data komputer.



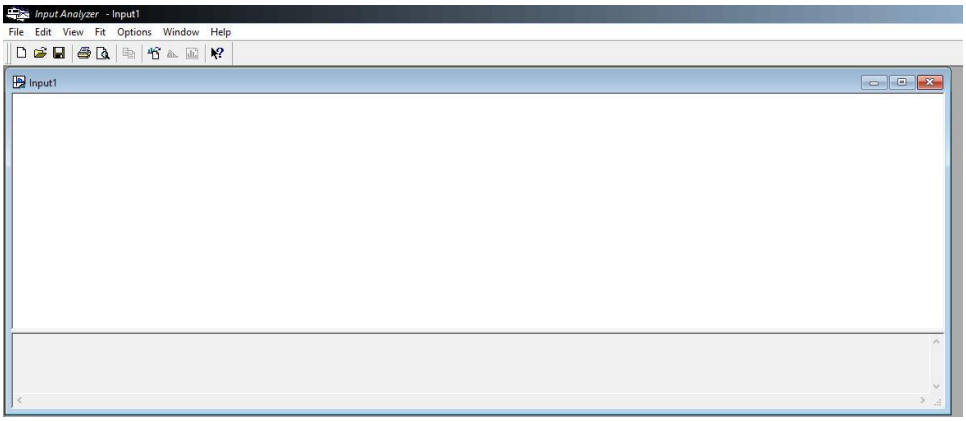
Gambar : Data Permintaan pada *Notepad*

2. Buka *software* ARENA 10.0 dengan cara klik *Program Files > Rockwell Software > Arena 10.0 > Arena*.
3. Pilih *Tools > Input Analyzer* untuk membuka *Input Analyzer.*



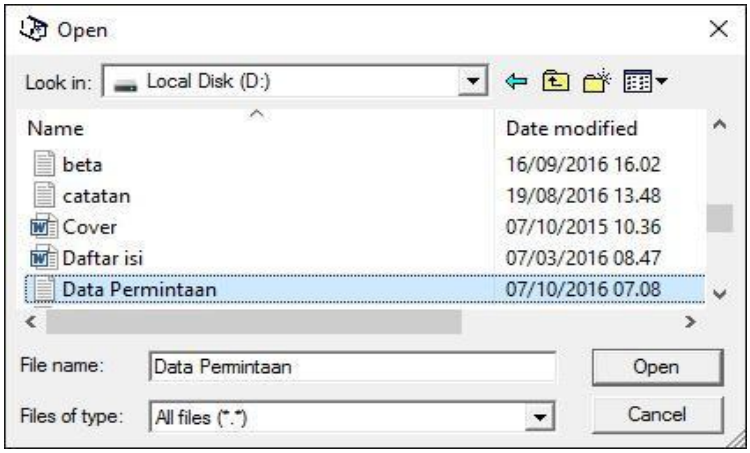
Gambar : Cara membuka *Input Analyzer*

4. Klik *New* untuk membuka jendela baru *Input Analyzer.*



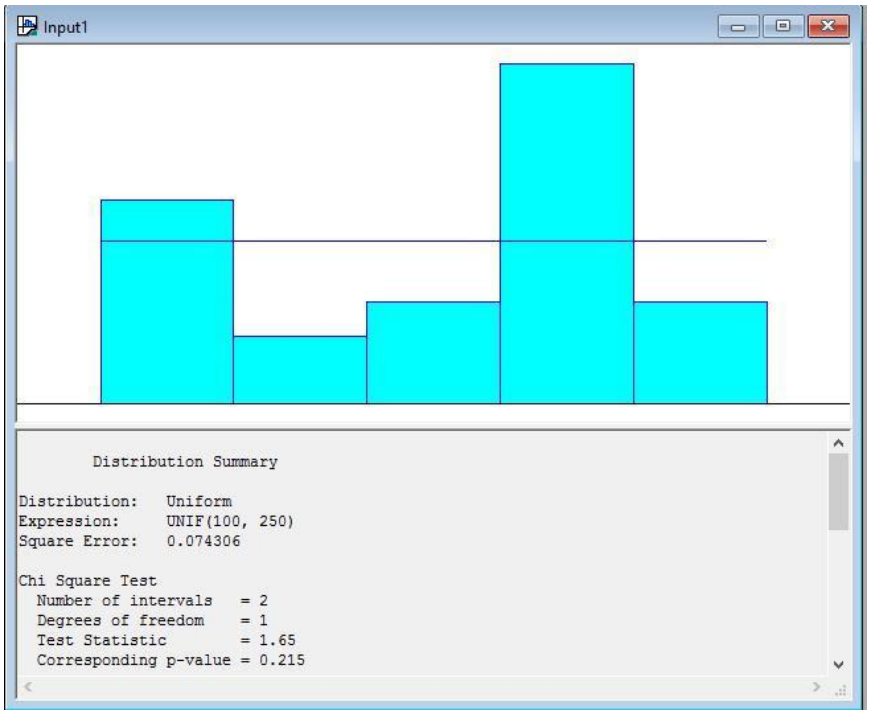
Gambar : Tampilan jendela baru *Input Analyzer*

5. Buka *file* Data Permintaan yang sudah di *save* pada *notepad* tadi degan cara klik (*use existing data file*). Kemudian ubah *file of type* ke *All Files > open.*



Gambar : *Open File* pada *Notepad*

6. Klik *Fit All* untuk mengetahui distribusi dari data.



Gambar : *Distribution Summary*

D. Tugas Praktikum

Data yang akan diolah untuk *fitting* data akan diberikan oleh Asisten Praktikum.

MODUL II
PEMBANGKITAN VARIABEL RANDOM DAN REPLIKASI

A. Tujuan Praktikum

Mahasiswa mampu membangkan variabel random kontinyu dan diskrit, serta melakukan replikasi data.

B. Pembangkit Bilangan Acak (Random)

Prosedur yang dipakai untuk mensimulasikan sebuah sample berdasarkan kepada penggunaan bilangan acak. Bilangan acak adalah bilangan yang dipilih dengan cara sedemikian rupa, sehingga setiap bilangan memiliki kesempatan atau probabilitas pemilihan yang sama. Sekali bilangan acak telah dipilih, kemudian kita mengubahnya ke dalam suatu pengamatan yang digambarkan dari distribusi probabilitas yang dinyatakan oleh model yang tengah dipelajari.

Dalam pembangkitan bilangan random sangat dibutuhkan fungsi komputer untuk lebih mempermudah dalam pembangkitan bilangan random tersebut. Salah satu program yang dapat digunakan untuk membangkitkan bilangan random yaitu dengan *excel*.

Untuk membangkitkan bilangan random dengan menggunakan program *excel* yaitu dengan mengetikkan atau menuliskan [=rand()] pada sel yang dikehendaki untuk pembangkitan bilangan random secara bebas. Sedangkan untuk pembangkitan bilangan random secara terikat atau terkunci dengan menuliskan [=rand()] ditambah dengan menekan [F9].

C. Pembangkitan Variabel Random

Sebuah simulasi yang mempunyai berbagai aspek-aspek random haruslah melibatkan sampling atau membangkitkan variabel random dari distribusi probabilitas.

Unsur-unsur dasar dibutuhkan untuk setiap metode pembangkitan variable random dari tiap-tiap distribusi atau pelayanan random adalah sumber variabel random IID U(0,1). Sebagai alasan, hal itu penting menurut statistik yang dapat dipercaya U(0,1), pembangkit bilangan random harus tersedia. Tanpa pembangkitan bilangan random yang dapat diterima, maka mustahil untuk membangkitkan variabel random secara tepat dari distribusi lainnya.

Beberapa algoritma alternatif dapat digunakan untuk pembangkitan variabel random dari suatu distribusi dan beberapa faktor-faktor yang dapat menjadi pertimbangan pemilihan algoritma pembangkitan variabel random adalah sebagai berikut :

1. *Exactness*

   Suatu algoritma yang menghasilkan variabel random dengan tepat sesuai dengan distribusi yang diinginkan.

2. *Efficient*

   Dalam kaitannya dengan "ruang penyimpanan" dan waktu pelaksanaan/eksekusi. Beberapa algoritma memerlukan gudang/penyimpanan dalam jumlah besar yang tetap. Waktu pelaksanaan/eksekusi terdiri dari dua faktor :

   a. Waktu pelaksanaan marginal

      Memenuhi pembuatan masing-masing variabel random di dalam suatu waktu.

   b. Waktu *set-up*

      Beberapa algoritma harus melakukan komputasi awal untuk menetapkan konstanta atau tabel yang tergantung pada parameter dan distribusi yang tertentu; waktu yang diperlukan untuk melakukan ini disebut waktu *set-up*. Kebanyakan simulasi, kita akan membangkitkan sejumlah besar variabel random dari distribusi yang ditentukan, maka waktu pelaksanaan marginal akan lebih penting dibanding waktu *set-up*. Jika parameter suatu distribusi sering berubah atau secara acak sepanjang keadaan simulasi, bagaimanapun, waktu *set-up* bisa menjadi suatu pertimbangan penting.

3. Tingkat kompleksitas

   Keseluruhan kompleksitas, termasuk yang konseptual seperti halnya faktor implementasi. Dalam menerapkan suatu metode untuk pembangkitan variabel random yang lebih efisien tetapi penggunaan suatu algoritma yang lebih rumit untuk mudah dipahami dan ditetapkan.

4. *Robustness*

   Beberapa algoritma bersandar pada suatu sumber dari variabel random dari distribusi selain U(0,1), yang mana tidak diinginkan, lain berbagai hal tetap sama. Algoritma yang efisien untuk semua nilai-nilai parameter atau yang disebut algoritma *robustness*/ketahanan.

D. Algoritma-algoritma Pembangkit Variabel Random

1. Pembangkit Variabel Random Kontinyu

a. Distribusi *uniform*

Algoritma :

1) Bangkitkan U ~ U[0,1]
2) X = a + (b-a) U

Di mana : a = nilai minimum

b = nilai maksimum

b. Eksponensial

Algoritma :

1) Bangkitkan U ~ U[0,1]
2) $X = -\beta \ln U$

c. m-Erlang

1) Bangkitkan $U_1, U_2, \ldots, U_n$ U[0,1]
2) $X = -\frac{\beta}{m} \ln\left(\prod_{i=1}^{m} U_i\right)$

d. Gamma

Algoritma :

1) Bangkitkan $U_1 \sim U(0,1)$ dan $P = bU_1$, jika P>1, kemudian ke langkah 3 jika tidak maka ke langkah 2
2) $Y = P^{1/\alpha}$ dan bangkitkan $U_2 \sim U[0,1]$, jika $U_2 \leq e^{-y}$, X=Y, jika tidak kembali ke langkah 1
3) $Y = -\ln\left[\left(\frac{b-p}{\alpha}\right)\right]$ dan bangkitkan $U_2 \sim U[0,1]$, jika $U_2 \leq Y^{\alpha-1}$, X=Y, jika tidak maka kembali ke langkah 1

Jika kita dapatkan X ~ gamma (α, 1), maka kita cukup menggunakan algoritma di atas, tetapi untuk gamma (α, β) kita masih harus membangkitkan X' = βX.

e. Weibull

Algoritmanya :

1) Bangkitkan U ~ U[0,1]

2) $X = \beta(-\ln U)^{1/\alpha}$

f. Normal

Ada banyak cara untuk membangkitkan variable random berdistribusi normal. Metode yang pertama muncul adalah metode yang diperkenalkan oleh Box and Muller (1958). Metode ini sangat sederhana.

Algoritmanya :

1) Bangkitkan U_1 dan U_2 sebagai IID U(0,1)

2) $X_1 = \sqrt{-2\ln U_1}\cos 2\pi U_2$ dan $X_2 = \sqrt{-2\ln U_1}\sin 2\pi U_2$

3) X_1 dan X_2 adalah variable random IID N(0,1)

Metode Box and Muller ini mengalami perkembangan dengan mengeliminasi penghitungan trigonometri. Hal ini dikenalkan oleh Marsaglia and Bray (1964) dan lebih dikenal dengan sebutan “metode polar”.

Algoritmanya :

1) Bangkitkan U_1 dan U_2, sebagai IID U(0,1) kemudian

$V_i = 2U_1 - 1$ untuk i = 1,2 dan $W = V_1^2 + V_2^2$

2) Jika W > 1, kembali ke langkah 1, jika tidak maka

$Y = \sqrt{\frac{(-2\ln W)}{W}}$, $X_1 = V_1 Y$ dan $X_2 = V_2 Y$,

X_1, X_2 adalah variabel random N(0,1).

g. Log Normal

Algoritmanya :

1) Bangkitkan $Y \sim N(\mu, \sigma^2)$

2) $X = e^Y$

h. Beta

Algoritmanya :

1) Bangkitkan $Y_1 \sim$ gamma $(\alpha_1, 1)$ dan $Y_2 \sim$ gamma $(\alpha_2, 1)$

2) $X = \dfrac{Y_1}{(Y_1 + Y_2)}$

i. Pearson type-V

Algoritmanya :

1) Bangkitkan $Y \sim$ gamma $\left(\alpha, \dfrac{1}{\beta}\right)$

2) $X = \dfrac{1}{Y}$

j. Perason type-VI

Algoritmanya :

1) Bangkitkan $Y_1 \sim$ gamma (α_1, β) dan $Y_2 \sim$ gamma (α_2, β), di mana Y_1 = variabel random bebas

2) $X = \dfrac{Y_1}{Y_2}$

k. Triangular

Algoritmanya :

1) Bangkitkan $U \sim U[0,1]$

2) Jika $U \leq c$, kembali ke $X = \sqrt{cU}$ atau $X = 1 - \sqrt{(1-c)(1-U)}$

l. Distribusi empirik

Algoritmanya : untuk satu data

1) Bangkitkan $U \sim U(0,1)$, $P = (n-1)\,U$, dan $I = [P] + 1$

2) $X = X_{(I)} + (P-I+1)\,(X_{(i+1)} - X_{(I)})$

Algoritmanya : untuk data kelompok

1) Bangkitkan $U \sim U(0,1)$

2) Temukan integer non negatif J ($0 \leq J \leq k-1$), seperti

$G(\alpha_j) \leq U < G(\alpha_{j+1})$, dan $X = \alpha_j + \frac{[U - G(\alpha_j)][\alpha_{j+1} - \alpha_j]}{[G(\alpha_{J+1}) - G(\alpha_j)]}$

2. Pembangkit Variabel Random Diskrit

a. *Bernoulli*

Algoritmanya :

1) Bangkitkan U ~ U(0,1)
2) Jika U ≤ P, kembali ke X = 1, jika tidak X = 0

b. Diskrit *uniform*

Algoritmanya :

1) Bangkitkan U ~ U(0,1)
2) X = i + [(j-i+1) U]

c. Distribusi *diskrit arbitrary*

Algoritmanya :

1) Bangkitkan U ~ U(0,1)
2) X = 1, tetap

d. Binomial

Algoritmanya :

1) Bangkitkan $Y_1, Y_2, \ldots, Y_i$ sebagai IID variabel random Bernoulli (P)
2) Kembali ke $X = Y_1 + Y_2 + \ldots + Y_i$

e. Geometrik

Algoritmanya :

1) Bangkitkan U ~ U(0,1)
2) X = [ln U / ln (1-P)]

f. Binomial (-)

Algoritmanya :

1) Bangkitkan $Y_1, Y_2, \ldots, Y_i$ sebagai IID variabel random geometrik (P)
2) $X = Y_1 + Y_2 + \ldots + Y_i$

g. *Poisson*

Algoritmanya :

1) Bangkitkan a = $e^{-\lambda}$, b = 1, dan i = 0
2) Bangkitkan $U_{i+1} \sim U(0,1)$ dan ulangi b dengan b U_{i+1}, jika b<a, kembali ke X = i. Jika tidak ke langkah 3.
3) Ulangi i = i + 1 dan kembali ke langkah 2

E. Replikasi

Pendekatan yang digunakan dalam simulasi adalah dengan membangkitkan data dalam jumlah yang kemudian melakukan pengulangan (replikasi) untuk dapat menganalisis hasil simulasi yang dijalankan. Metode yang umum untuk menentukan panjang simulasi (jumlah replikasi) adalah dengan beberapa percobaan menggunakan bilangan acak berbeda untuk mendapatkan *mean* dan standar deviasi dari variabel yang diukur. Dengan mengasumsikan bahwa variabel yang diukur berdistribusi normal, panjang simulasi dapat ditentukan untuk akurasi dan tingkat kepercayaan yang diberikan.

Rumus berikut ini dapat digunakan untuk menentukan panjang simulasi yang diinginkan :

$$n_a(\beta) = i \geq n; t_{i-1,i-a}\sqrt{\frac{\sigma_{(n)}}{i}} \leq \beta \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (4)$$

Di mana : μ = Rata-rata populasi

$\bar{\mu}$ = Rata-rata sampel

na = jumlah replikasi kondisi steady state

t = data ke-i

σ = standar deviasi

Sehingga dapat ditentukan $\beta = \left|\mu - \bar{\mu}\right|$, di mana β adalah *absolute error* dan selisih rata-rata sampel. Apabila nilai nilai *interval convident* dari n replikasi yang dicari, maka sistem sudah dikatakan *steady state*.

F. Tugas Praktikum

Data untuk pembangkitan variabel random menggunakan data minggu kemarin dan dilakukan replikasi.

MODUL III
MODEL SIMULASI PERSEDIAAN

E. Tujuan Praktikum

1. Praktikan dapat membuat model simulasi sistem persediaan.
2. Praktikan dapat menganalisis *output* berdasarkan *report* hasil simulasi persediaan.

F. Landasan Teori

1. Definisi Persediaan

Persediaan adalah sumber daya menganggur yang menunggu proses lebih lanjut. Yang dimaksud dari proses lebih lanjut tersebut adalah berupa kegiatan produksi pada sistem manufaktur, kegiatan pemasaran pada sistem distribusi ataupun kegiatan konsumsi pada sistem rumah tangga.

Persediaan juga dapat diartikan sebagai aktifitas penyediaan sejumlah barang milik perusahaan (bahan baku atau komponen-komponen) yang disediakan untuk kelancaran proses produksi pembuatan barang jadi atau setengah jadi untuk memenuhi permintaan di setiap waktu. Persediaan bahan baku merupakan unsur yang penting dalam perusahaan, karena perusahaan akan dihadapkan pada suatu waktu akan mengalami kekurangan bahan yang diproduksi sehingga tidak dapat memenuhi kebutuhan konsumen yang akan menyebabkan kerugian. Hal ini dapat diperkecil dengan mengadakan perencanaan yang teliti, persediaan barang dimaksudkan untuk mempermudah atau memperlancar jalannya proses produksi.

Pengendalian adalah aktivitas manajerial dalam memonitor pelaksaan rencana dan melakukan tindakan perbaikan yang dibutuhkan, sedangkan pengertian pengendalian persediaan adalah aktivitas persediaan sejumlah barang milik perusahaan yang dilaksanakan sesuai rencana untuk kelancaran proses produksi dan melakukan tindakan perbaikan yang dibutuhkan untuk memenuhi permintaan dari pelanggan setiap waktu.

Dalam sistem manufaktur, persediaan terdiri dari tiga bentuk, yaitu:

a. Bahan baku, yaitu yang merupakan input awal dari proses transformasi menjadi produk jadi.

b. Barang setengah jadi, yaitu yang merupakan bentuk perakitan antara bahan baku dengan produk setengah jadi.
c. Barang jadi, yaitu merupakan hasil akhir dari proses transformasi yang siap dipasarkan kepada konsumen.

2. Motode Monte Carlo

Salah satu metode teknik simulasi adalah metode Monte Carlo. Metode Monte Carlo memakai bilangan acak yang digunakan untuk menyelesaikan masalah-masalah yang mencakup keadaan ketidakpastian atau stokastik dimana evaluasi matematis tidaklah mungkin. Simulasi Monte Carlo merupakan alat khusus yang berguna untuk mensimulasikan situasi yang mengandung resiko, sehingga didapatkan jawaban-jawaban yang tidak dapat diperoleh dari penelitian-penelitian secara fisik maupun matematis.

Simulasi ini bertitik tolak pada generalisasi fakta-fakta yang terjadi dengan cara mempresentasikannya ke dalam bilangan random dan distribusi probabilitas yang digunakan sebagai sampel untuk menggambarkan generalisasi dari obyek yang sedang diamati.

Bilangan-bilangan *random* tersebut dibangkitkan dari generator bilangan *random* sehingga dapat dimunculkan sesuai dengan digit data. Untuk digit tunggal dari bilangan 0,0 sampai 0,9. Sedangkan untuk digit ganda bilangan random diawali dari bilangan 0,0 sampai 99.

Penggunaan komputer sangat membantu dalam pembuatan Simulasi Monte Carlo. Hal ini dikarenakan dalam Simulasi Monte Carlo diperlukan banyak replikasi untuk mendapatkan hasil simulasi yang mempunyai tingkat ketelitian tinggi dan variansi sampel dengan tingkat akurasi paling rendah.

Simulasi Monte Carlo mengandung suatu solusi yang sangat mendekati optimal, tetapi tidak perlu solusi yang eksak. Metode simulasi Monte Carlo berdasarkan kepada pengamatan sistem yang berkesinambungan selama pengalaman suatu periode waktu yang panjang.

Beberapa algoritma bersandar pada suatu sumber dari variabel random dari distribusi selain U(0,1), yang mana tidak diinginkan, lain berbagai hal tetap sama. Algoritma yang efisien untuk semua nilai-nilai parameter atau yang disebut algoritma *robustness* atau ketahanan.

3. Fungsi Persediaan

Persediaan yang ada dalam perusahaan berguna untuk menjaga stabilitas proses operasi pabrik dan menjamin kelancaran produksi. Kegunaan yang lain adalah menghilangkan resiko keterlambatan datangnya barang yang dibutuhkan oleh industri, memperkecil resiko dari bahan baku yang dihasilkan secara musiman sehingga dapat digunakan pada waktu bahan tidak ada di pasar dan perusahaan dapat memberikan pelayanan yang sebaik-baiknya kepada konsumen atau pelanggan.

a. Pernyataan permasalahan

- Sebuah perusahaan yang menjual produk tunggal, berkeinginan untuk memutuskan berapa banyak produk yang harus disimpan (persediaan) untuk masing-masing n bulan ke depan, di mana :

  a) Waktu antar permintaan merupakan variabel Random berdistribusi eksponensial IID, dengan rata-rata 0,1 bulan.

  b) Ukuran permintaan (D), juga merupakan variabel random IID, yaitu :

$$0 = \begin{cases} 1 \text{ dengan } & \text{Prop } 1/6 \\ 2 \text{ dengan } & \text{Prob } 1/3 \\ 3 \text{ dengan } & \text{Prob } 1/3 \\ 4 \text{ dengan } & \text{Prob } 1/6 \end{cases}$$

- Setiap awal bulan, perusahaan mereview level inventory dan membuat keputusan "berapa banyak produk yang harus dipesan dari pemasok ".

  a) Jika perusahaan memesan Z item, maka biaya yang dikeluarkan perusahaan adalah sebesar : K + i.Z

  Di mana : K : biaya set-up : $ 32

  i : biaya pembelian/unit : $ 3

  Jika Z = 0, perusahaan tidak mengeluarkan biaya.

  b) Ketika pemesanan dilakukan, waktu yang dibutuhkan untuk kedatangan pesanan dinamakan *delivery lag* atau *lead time* merupakan variabel random yang berdistribusi *uniform* antara 0,5 dan 1 bulan.

- Perusahaan menggunakan kebijaksanaan (s,S) untuk memutuskan berapa yang harus dipesan, sehingga :

$$Z = \begin{cases} S - I, \text{ jika } I < S \\ 0, \text{ jika } I \geq S \end{cases}$$

Di mana : I : level inventory pada awal bulan

- Ketika ada permintaan, maka langsung dipenuhi, jika tingkat persediaan ≥ demand.
- Jika permintaan > tingkat persediaan, maka kelebihan permintaan merupakan backlog (ditunda pemenuhannya) dan dipenuhi dari kedatangan pesanan berikutnya.
  a) Pada kasus ini, tingkat inventory yang baru = tingkat inventory lama – permintaan = tingkat inventory.
  b) Ketika pesanan datang, pertama digunakan untuk menghilangkan *backlog*, sisa pesanan jika ada, ditambahkan pada inventory.
- Tambahan notasi yang digunakan :

  I (t) : level inventory pada saat t

  (I(t) dapat +, -, atau 0)

  $I^{+}(t) = \max\{I(+),0\}$

  jumlah produk yang masih disimpan pada saat $t \rightarrow I^{+}(t) \geq 0$

  $I^{-}(t) = \max\{-I(t),0\}$ backlog pada saat $t \rightarrow I^{-}(t) \geq 0$
- Biaya lain yang harus dikeluarkan perusahaan adalah biaya simpan.

  (h) = \$1/produk/bulan.
- Karena $I^{+}(t)$ adalah jumlah produk yang disimpan pada saat t, waktu rata-rata (per bulan) jumlah produk yang tersimpan untuk periode n bulan adalah :

$$\bar{I}^{+} = \frac{\int_{0}^{n} I^{+}(t)\, dt}{n} \approx \text{rata-rata jumlah konsumen dalam antrian}$$

h. $\bar{I}^{+}$ = rata-rata biaya simpan per bulan

- Biaya backlog : μ= \$ 5 /produk/bulan.
- Rata-rata jumlah produk yang terjadi dalam backlog.

$$\bar{I}^{-} = \frac{\int_{0}^{n} I^{-}(t)\, dt}{n}$$

$\mu\bar{I}$ = rata-rata backlog/bulan

- Diasumsikan tingkat inventory awal I(0) = 60, disimulasikan untuk n = 120 bulan. Dan digunakan total biaya/bulan (jumlah dari rata-rata biaya *set-up*/pesan.rata-rata biaya simpan dan rata-rata biaya *shortage/backlog*), untuk membandingkan sembilan kebijakan inventory berikut :

s	20	20	20	20	40	40	40	60	60
S	40	60	80	100	60	80	100	80	100

Variabel status untuk model simulasi sistem inventory ini adalah:

a) Tingkat inventory I(t)
b) Jumlah barang/produk yang dipesan ke pemasok
c) Waktu terjadinya event terakhir

(untuk menghitung luas kurva di bawah I $I^{+}(t)$ & $I^{-}(t)$)

b. Organisasi dan logika program

Model sistem inventory menggunakan tipe-tipe event sebagai berikut :

Tipe event	Penjelasan
1	Kedatangan pesanan dari pemasok
2	Permintaan produk dari konsumen
3	Akhir simulasi setelah n bulan
4	Evaluasi invent (kemungkinan order) pada awal bulan

Pada model ini terdapat tiga tipe variabel random yang digunakan untuk mensimulasikan sistem inventory tersebut yaitu :

1) Waktu antar permintaan, yang berdistribusi eksponensial (algoritma pada sistem antrian dapat digunakan).
2) Ukuran permintaan merupakan variabel random diskrit dan dapat dibangkitkan dengan cara sebagai berikut :
   a) Bagi unit interval dalam sub interval :
      - C1 = $(0, \frac{1}{6})$
      - C2 = $(\frac{1}{6}, \frac{1}{2})$
      - C3 = $(\frac{1}{2}, \frac{5}{6})$

- C4 = ($\frac{5}{6}$,1)

b) Dapatkan sebuah var. Random U(0,1)
(Bilangan Random) dari pembangkit bilangan Random

c) Jika U jatuh pada C1, maka D = 1
Jika U jatuh pada C2, maka D = 2, dst.
Di mana :

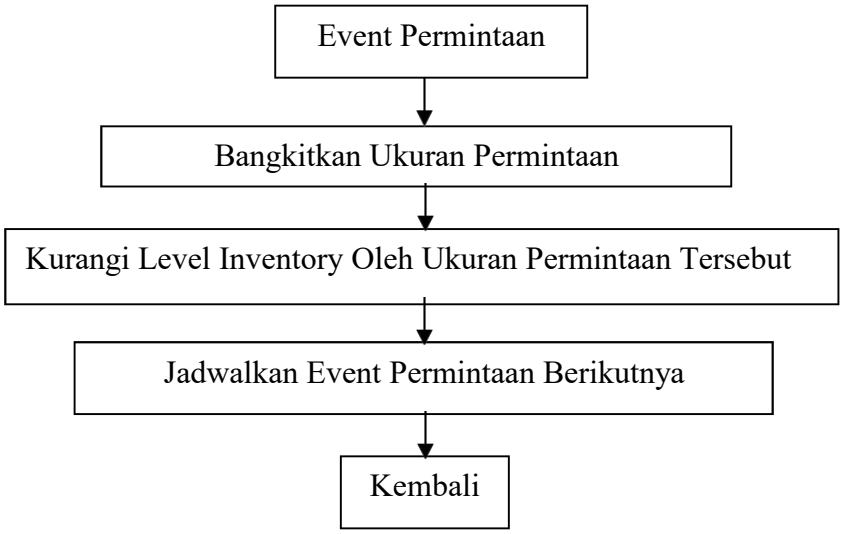
- Karena lebar C1 = $\frac{1}{6}$ -0 = $\frac{1}{6}$
- Dan karena U berdistribusi uniform pada [0,1], maka probabilitas U jatuh pada C1 (sehingga D = 1) adalah $\frac{1}{6} \approx$ probabilitas yang diinginkan bahwa D = 1.
- Dengan cara yang sama, D=2 jika U jatuh pada C2 dengan probabilitas yang sesuai dengan lebar dari C2 = $\frac{1}{2}$ - $\frac{1}{6}$ = $\frac{1}{3}$

3) *Delivery Lag* berdistribusi *uniform*, tetapi tidak pada interval [0,1]. Secara umum, dapat dibangkitkan variabel random berdistribusi *uniform* pada interval [a,b], dengan membangkitkan bilangan random U(0,1) dan mengembalikan ke a + U (b-a).

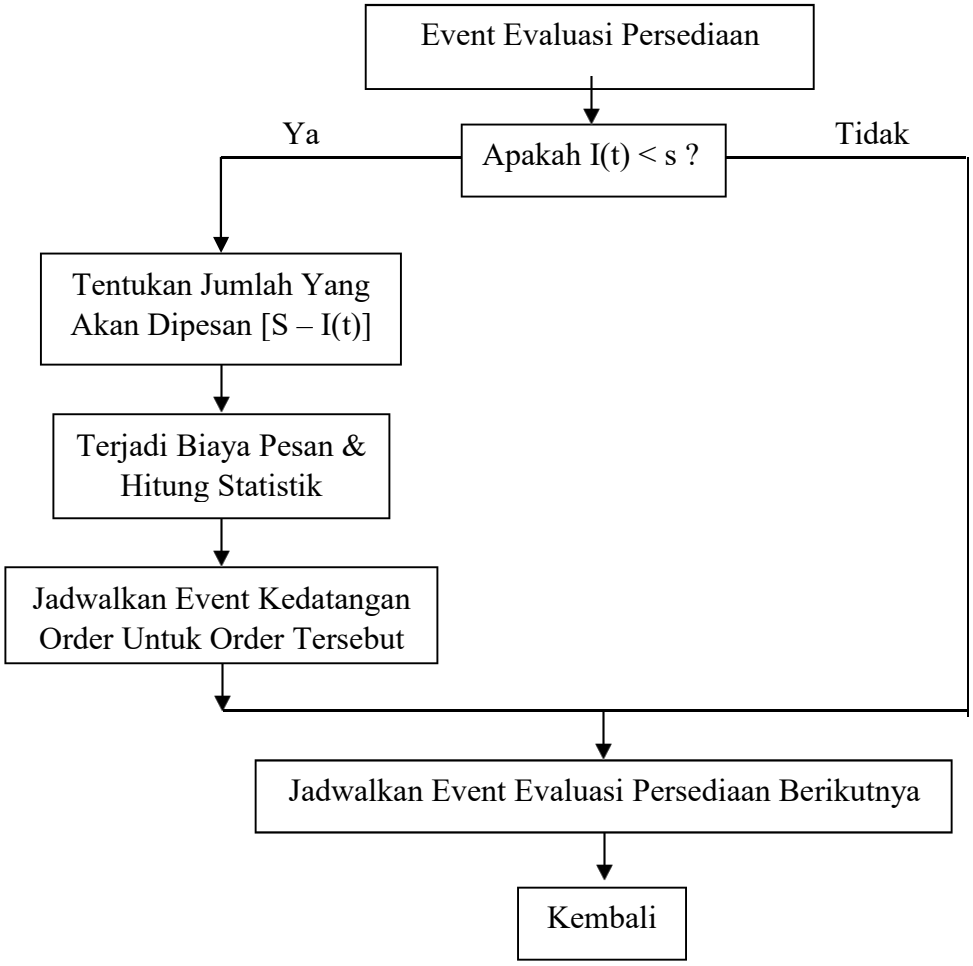
c. Flow chart untuk model sistem inventory

1) Flow chart untuk routine kedatangan order :

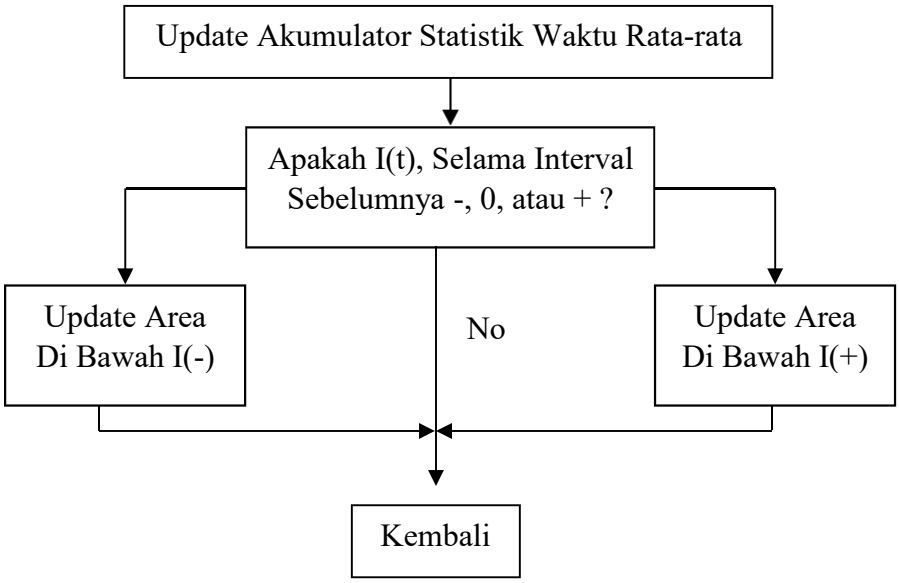
2) Flow chart untuk routine permintaan :



3) Flow chart untuk routine evaluasi inventory :



4) Diagram alir untuk routine update akumulator statistik waktu kontinyu



4. Model EOQ untuk menentukan ROQ dan ROP

Model EOQ dengan lead time ≠ 0 untuk penentuan ROQ dan ROP adalah sebagai berikut :

a. Lebih realistik.

b. Jika LT diketahui dengan pasti, maka perhitungan dapat dilakukan seperti model

EOQ klasik untuk Q* = ROQ = $\sqrt{\frac{2KD}{h}}$.

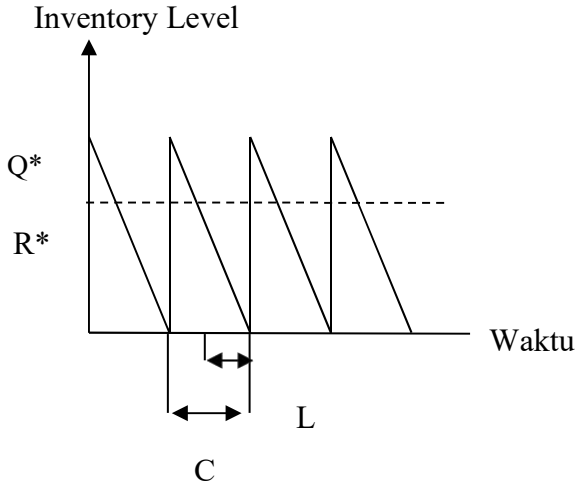
Di mana: K = Biaya pesan

D = Rata-rata demand

H = Biaya simpan

Ada 2 macam model EOQ dengan LT ≠ 0, yakni sebagai berikut:

a. LT < siklus inventory (c)



C : Siklus inventory

L : Panjang LT rata-rata

R* (ROP) : Reader point (level inventory ketika harus dilakukan pemesanan)

Keterangan :

Ketika level inventory mencapai R*, harus dilakukan pemesanan, sehingga order datang, ketika level inventory = 0 (habis).

$C = Q^*/D$

$$R^* = \left(\frac{L}{C}\right).Q^* = \left((L)./\frac{Q^*}{D}\right).Q^* = \left(L.\frac{D}{Q^*}\right).Q^* = L.D$$

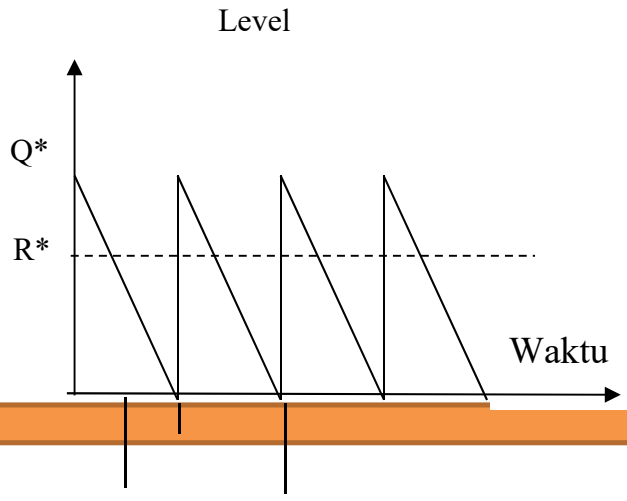
Contoh : dari sebelumnya diketahui Q* : 5477 yard dan C : 0,183 tahun = 67 hari, jika L = 30 hari, berapa R*?

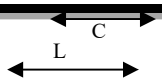
R* (ROP) = L.D

$$= \frac{30}{(365)}.30000 = 2466 \text{ yard}$$

Sehingga ketika level inventory tinggal 2466 yard, order sebanyak 5477 yard dipesan.

b. LT > siklus inventory (c)





R* : Level *inventory* yang digunakan untuk menandai ketika harus dilakukan pesanan.

- Order dipesankan pada siklus *inventory* sebelumnya.
- Setidaknya, terdapat 1 pesanan datang selama LT.
- Pesanan yang digunakan untuk mengantikan *inventory* pada akhir siklus *inventory* ke-2, harus dipesan sebelum pesanan untuk menggantikan *inventory* pada akhir siklus *inventory* datang.
- R* : F(L/C).Q* di mana F(L/C) : fraksi dari hasil bagi L/C.

Contoh : seperti sebelumnya diketahui L : 75 hari , sehingga :

R*= F (L/C) . Q*

= F (75/67). Q* = F(1,12).Q* = 0,12.5477 = 657 yard

Artinya : apabila level *inventory* tinggal 657 yard, order sejumlah 5477 yard dipesan tetap order tersebut dapat digunakan untuk memenuhi permintaan pada siklus *inventory* berikutnya.

G. Langkah-langkah Simulasi Persediaan

1. Fitting Data

Tahapan fitting data sesuai dengan materi pada modul 1

2. Replikasi

Tahapan replikasi sesuai dengan materi pada modul 2

3. Model

Model simulasi persediaan yaitu meliputi :

a. Menghitung Riil
b. Simulasi Monte Carlo Riil
c. Optimasi Riil
d. Simulasi Optimasi

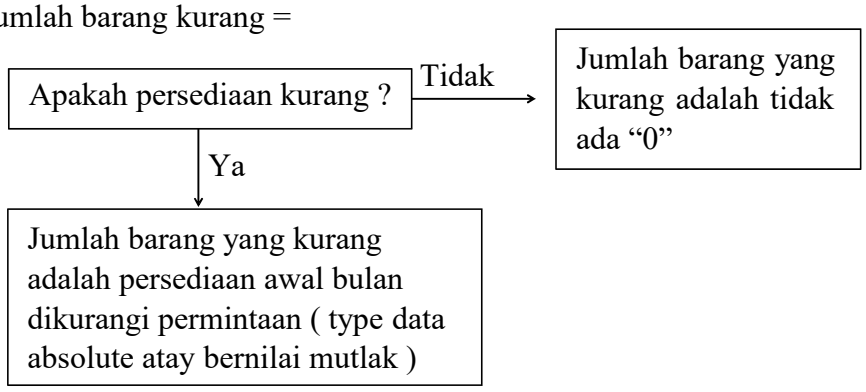
Algoritma yang digunakan dalam simulasi persediaan monte carlo adalah

a. Persediaan awal = jumlah pesan + persediaan akhir

b. Persediaan akhir bulan = persediaan awal bulan dikurangi dengan permintaan jika permintaan lebih besar dibandingkan persediaan awal, maka persediaan akhir bulan tidak ada (0).

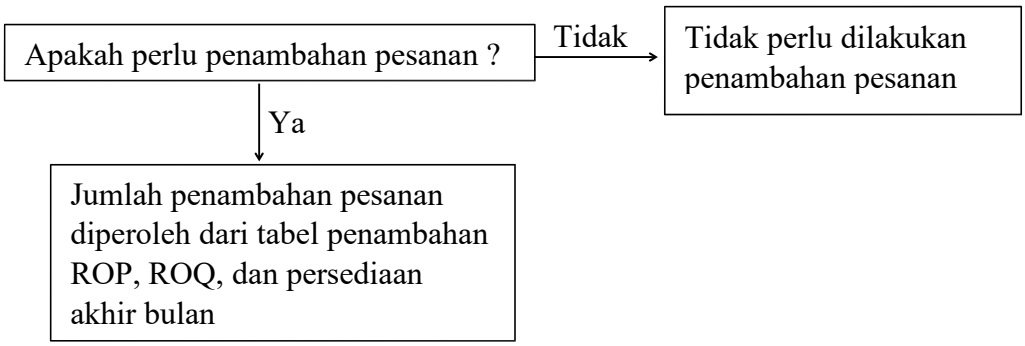
c. Apakah persediaan kurang = persediaan kurang terjadi akibat jumlah permintaan lebih besar di banding persediaan awal bulan "Ya" dan jika sebaliknya "Tidak".

d. Jumlah barang kurang =



e. Apakah perlu penambahan pesanan = persediaan awal bulan lebih besar dari ROP maka tidak perlu penambahan pesanan "Tidak" dan jika sebaliknya maka perlu penambahan pesanan "Ya"

f. Jumlah penambahan pesanan =



g. Waktu tunggu penambahan pesanan = jumlah penambahan pesanan lebih dari nol "0" atau dilakukan penambahan pesanan maka waktu tunggu penambahan pesanan adalah lead time.

h. Biaya simpan = jumlah pesanan * biaya pesan

i. Biaya simpan = persediaan akhir * biaya simpan

j. Biaya backlog = jika jumlah barang kurang "tidak ada" maka biaya backlog tidak ada dan sebaliknya jika jumlah barang kurang "ada" maka biaya biaya backlog * backlog

4. Validasi dan Verifikasi

Memilih kriteria performansi:

a. Persediaan akhir

b. Persediaan Awal

Validasi model merupakan langkah untuk mengetahui apakah hasil simulasi model sesuai atau sudah mendekati sistem nyatanya.

a. Statistik Non Parametrik

Syarat :

- Bentuk atau type data nominal atau ordinal
- Sampel berdistribusi selain normal
- Jumalh sampel < 30

Jika data berdistribusi non-normal, maka pengujiannya menggunakan uji Kolmogorov Smirnov atau uji Mann-Whitney U.

b. Statistik Parametrik

Syarat :

- Bentuk atau type data interval atau ratio
- Sampel berdistribusi normal
- Jumlah sampel ≥ 30

Jika data berdistribusi normal, menggunakan uji kesamaan dua rata-rata atau uji kesamaan dua varians. Yaitu dengan cara membandingkan antara *output* kondisi nyata dari sistem dengan *output* hasil simulasi.

1) Pengujian kesamaan dua rata-rata yaitu sebagai berikut :

a) Pasangan hipotesisnya adalah :

$H_0 : \mu_1 = \mu_2$

$H_1 : \mu_1 \neq \mu_2$

b) Diasumsikan tingkat kesalahan (α) = 0,05

c) H_0 diterima jika :

t/z hit < t/z tabel

H_0 ditolak jika :

t/z hit > t/z tabel

2) Pengujian kesamaan dua varians adalah sebagai berikut :

a) Pasangan hipotesisnya adalah :

$H_0 : \sigma_1^2 = \sigma^2_2$

$H_1 : \sigma^2_1 \neq \sigma^2_2$

b) Diasumsikan tingkat kesalahan (α) = 0,05

c) Kriteria pengujian :

- Apabila $S_1^2 > S_2^2$

  H_0 diterima apabila : $F \leq F_{\alpha/2;n_1-1;n_2-1}$

  H_0 ditolak apabila : $F < F_{\alpha/2;n_1-1;n_2-1}$

- Apabila $S_1^2 < S_2^2$

  H_0 diterima apabila : $F \leq F_{\alpha/2;n_2-1;n_1-1}$

  H_0 ditolak apabila : $F > F_{\alpha/2;n_2-1;n_1-1}$

d) Perhitungan nilai F :

$$F = \frac{\text{Varians terbesar}}{\text{Varians terkecil}}$$

5. Analisis Hasil

Pada analisis hasil yang dilakukan adalah menjelaskan hasil dari validasi dan verifikasi.

H. Tugas Praktikum

1. Pembuatan simulasi persediaan usulan pada kasus real adalah sebagai berikut :

a. Menentukan ROP dan ROQ baru, sesuai dengan data

$$D = \frac{\text{jumlah permintaan * frekuensi permintaan}}{\text{jumlah frekuensi permintaan}}$$

$$L = \frac{\text{jumlah lead time * frekuensi lead time}}{\text{jumlah frekuensi lead time}}$$

Dimana : D = demand rata – rata

L = lead time rata – rata

b. Mengusulkan beberapa alternatif model (s, S) merupakan nilai yang mendekati ROP dan ROQ, T (lead time).

c. Membuat model simulasi bagi masing – masing alternatif usulan dan hitung TIC-nya.
d. Membandingkan TIC dari model – model yang diusulkan terhadap simulasi kondisi riil pada modul III dan dipilih model TIC terkecil sebagai usulan.

2. Buatlah simulasi model Monte Carlo untuk setiap kasus pada masing – masing kelompok!

3. Lakukan validasi dan replikasi model, serta menghitung TIC!

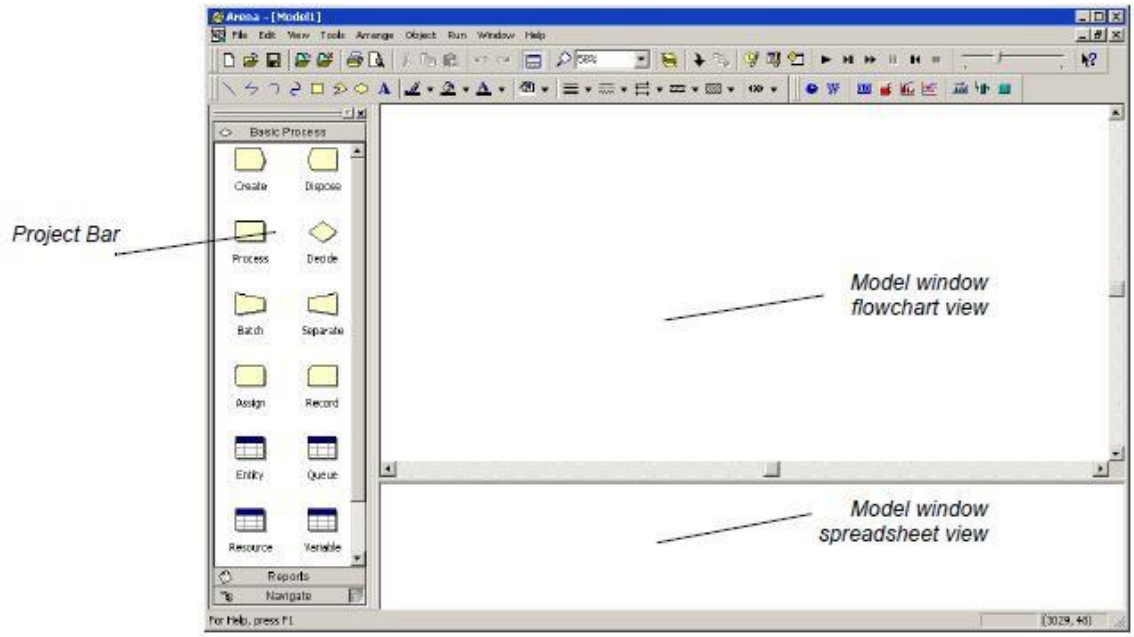
MODUL IV
PENGENALAN ARENA II

I. Tujuan Praktikum

Mengenalkan beberapa fungsi yang sering digunakan dalam pembuatan model simulasi dan membuat model simulasi antrian.

J. Model Utama

Jika Arena belum *running*, untuk membuka mulai dengan menu *Program Files* > *Rockwell Software* > *Arena*. Arena. Area pemodelan Arena akan membuka dengan jendela model baru, seperti yang ditunjukkan di bawah ini.



Gambar : Area Pemodelan Arena

Proses pembuatan model di Arena, akan bekerja di tiga wilayah utama jendela aplikasi. *Project Bar*, panel dengan jenis utama dari objek yang akan bekerja dengan:

1. ***Basic Process, Advanced Process***, dan ***Advanced Transfer panels***: Merupakan bentuk pemodelan, disebut modul, yang akan digunakan untuk menentukan proses pemodelan.
2. ***Reports panel***: Berisi laporan yang tersedia untuk menampilkan hasil *running* simulasi.
3. ***Navigate panel***: Digunakan untuk menampilkan pandangan yang berbeda dari model, termasuk menavigasi melalui submodel.

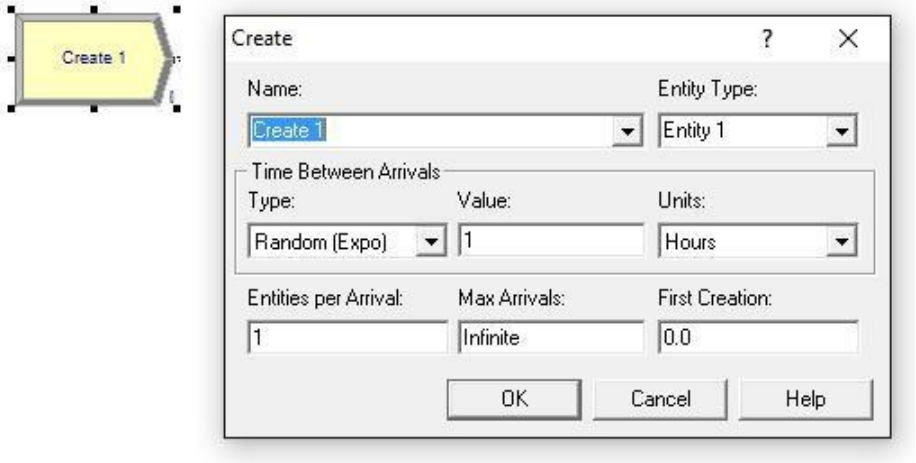
Pada jendela Model, ada dua daerah utama. *Flowchart View* akan berisi semua Model grafis yang dibuat, termasuk flowchart proses, animasi, dan elemen gambar lainnya. Sedangkan, *Spreadsheet View* menampilkan model data, seperti waktu, biaya, dan parameter lainnya.

1. Basic Process Panel

Basic process panel berisikan modul-modul yang digunakan untuk memodelkan simulasi sebuah sistem. Berikut adalah module yang terdapat dalam *Basic Process Panel*.

a. Create Module

Modul ini merupakan sebagai titik awal untuk entitas dalam model simulasi. Entitas dibuat menggunakan jadwal atau berdasarkan waktu antara kedatangan. Contoh penggunaan: Kedatangan pelanggan dalam proses pelayanan, Kedatangan bahan baku dalam proses produksi dan lain-lain.



Gambar : *Create Module*

Prompt	***Description***
Name	Digunakan untuk memberi nama pada modul yang dibuat
Entity Type	Mendeskripsikan jenis entitas yang akan dibangkitkan
Type	Menentukan jenis aliran kedatangan yang akan dihasilkan
Value	Menentukan *mean* dari distribusi eksponensial (jika *Random* digunakan) atau nilai konstan (*Constant* jika digunakan) untuk waktu antara kedatangan.
Unit	Menjelaskan satuan waktu yang digunakan untuk pembuatan antar kedatangan.
Entity per arrival	Jumlah entitas yang akan masuk sistem pada waktu tertentu dalam masing-masing kedatangan.

Max Arrival	Maksimum jumlah entitas dapat dihasilkan oleh modul ini. Ketika nilai ini tercapai, penciptaan entitas baru dengan modul ini berhenti.
First Creation	Mulai waktu untuk entitas pertama tiba ke dalam sistem. Tidak berlaku ketika *Type* adalah *Schedule*.

b. Dispose Module

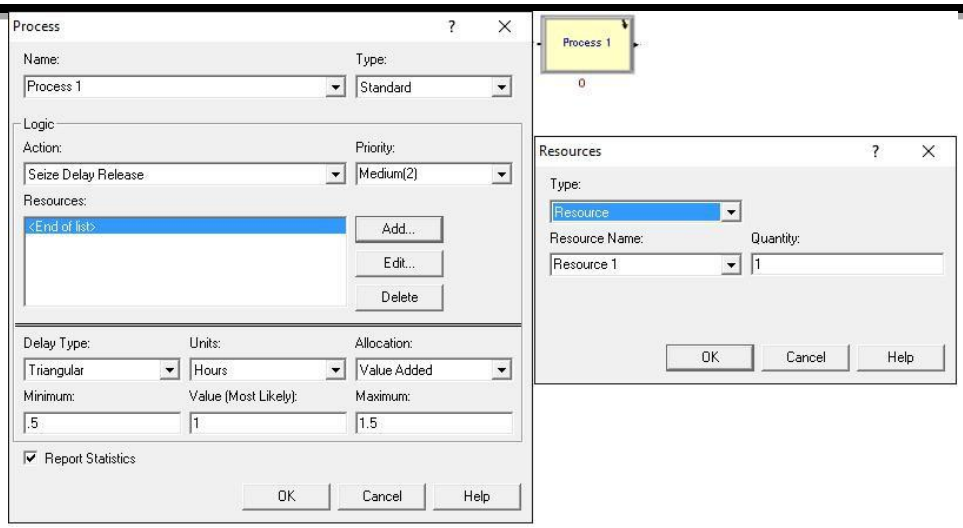
Modul ini adalah titik akhir untuk entitas dalam model simulasi yang dimana entitas statistik dapat direkam sebelum entitas tersebut dijual. Contoh penggunaan: Part-part meninggalkan model fasilitas, Customer keluar dari sebuah toko dan lain-lain.

Gambar : *Dispose Module*

Prompt	***Description***
Name	*Identifier* modul yang unik ditampilkan pada bentuk modul.
Record entity statistic	Menentukan penyimpanan statistik entitas yang masuk.

c. Process Module

Modul ini sebagai metode pengolahan utama dalam simulasi. Pilihan untuk mengambil dan melepaskan keterbatasan sumber daya yang tersedia. Selain itu, ada pilihan untuk menggunakan "submodel" dan menentukan *user-defined* logika hirarki. Waktu proses dialokasikan ke entitas dan dapat dianggap sebagai nilai tambah, non-nilai tambah, transfer, menunggu atau lainnya. Biaya yang terkait akan ditambahkan ke kategori yang sesuai.



Gambar : *Process Module*

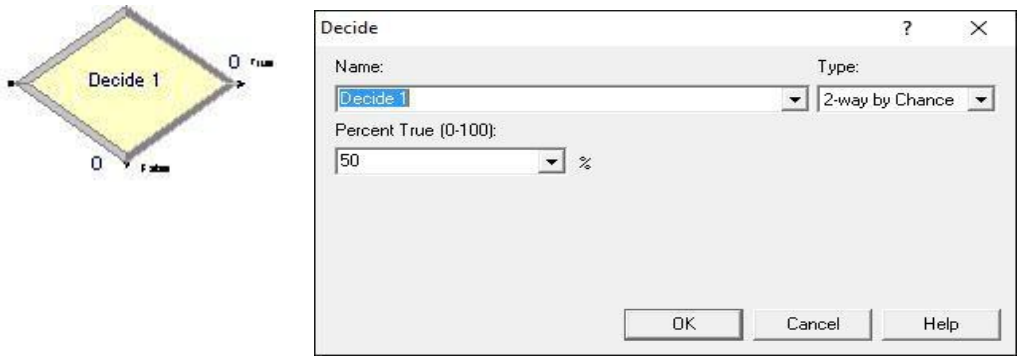
Prompt	***Description***
Name	*Identifier* modul yang unik ditampilkan pada bentuk modul.
Type	Metode menentukan logika dalam modul. pengolahan standar menandakan bahwa semua logika akan disimpan dalam modul Proses dan didefinisikan oleh *Action* tertentu. Submodel menunjukkan bahwa logika akan hierarkis didefinisikan dalam "submodel" yang dapat mencakup sejumlah modul logika.
Action	Jenis pengolahan yang akan terjadi dalam modul. Penundaan hanya menunjukkan bahwa penundaan proses akan dikeluarkan dengan tidak ada keterbatasan sumber daya. *Seize delay* menunjukkan bahwa sumber daya (s) akan dialokasikan dalam modul ini dan penundaan akan terjadi, tapi itu rilis sumber daya akan terjadi di kemudian hari. *Seize delay Release* menunjukkan bahwa sumber daya (s) akan dialokasikan diikuti dengan penundaan proses dan kemudian sumber daya dialokasikan (s) akan dirilis. *Delay Release* menunjukkan bahwa sumber daya (s) sebelumnya telah dialokasikan dan bahwa entitas hanya akan menunda dan melepaskan sumber daya yang ditentukan (s). Hanya berlaku ketika *Type* adalah *Standard*.
Priority	Nilai prioritas dari entitas menunggu di modul ini untuk sumber daya yang ditentukan (s). Digunakan ketika satu atau lebih entitas dari modul lain sedang menunggu sumber daya yang sama (s). Tidak berlaku ketika *Action* adalah *Delay* atau *Delay Release*, atau ketika Jenis adalah Submodel

Resources	Daftar sumber daya atau set sumber daya yang digunakan untuk pengolahan entitas. Tidak berlaku ketika Action adalah *Delay*, atau ketika *Type* adalah Submodel.
Resource Name	Nama sumber daya yang akan disita da /atau dilepaskan. Hanya berlaku ketika Type adalah *Resource*.
Set Name	Nama set sumber daya yang anggota akan diambil dan/atau dilepaskan. Hanya berlaku ketika Type Set.
Quantity	Jumlah sumber daya dari nama yang diberikan atau dari himpunan yang akan diambil/diirilis. Untuk set, nilai ini hanya menentukan jumlah sumber daya yang dipilih yang akan diambil/dirilis (berdasarkan kapasitas sumber daya ini), bukan jumlah anggota set yang akan diambil/dirilis.
Selection Rule	Metode pemilihan antara *resource* yang tersedia dalam satu set. akan siklus siklus melalui anggota tersedia (1 anggota anggota-3 anggota-2 anggota-3 anggota-1 anggota-2). Acak akan memilih secara acak anggota. *Preferred Order* akan selalu memilih anggota pertama yang tersedia (anggota 1 jika tersedia, anggota kemudian 2 jika tersedia, anggota kemudian 3). Spesifik Anggota membutuhkan nilai atribut masukan untuk menentukan anggota himpunan (sebelumnya disimpan dalam *Save Attribute*). *Largest Remaining Capacity* dan *Smallest Number Busy* digunakan untuk sumber daya dengan kapasitas beberapa. Hanya berlaku ketika *Type Set*.
Save Attribute	Atribut nama yang digunakan untuk menyimpan nomor indeks ke set dari anggota yang dipilih. Atribut ini nantinya bisa dirujuk dengan aturan seleksi *Specific Member*.. Berlaku hanya jika Seleksi *Rule* adalah selain *Specific Member*. Tidak berlaku bila Seleksi *Rule* adalah *Specific Member*. Jika Action ditetapkan sebagai *Delay Release*, nilai yang ditentukan mendefinisikan yang anggota (nomor indeks) dari himpunan akan dirilis. Jika tidak ada atribut yang ditentukan, entitas akan merilis anggota himpunan yang diambil.
Set Index	Nomor indeks ke set dari anggota yang diminta. Berlaku hanya jika Seleksi Rule adalah *Specific Member*. Jika *Action* ditetapkan sebagai *Delay Release*, nilai yang ditentukan mendefinisikan yang anggota (nomor indeks) dari himpunan akan dirilis.
Delay Type	Jenis distribusi atau metode menentukan parameter *delay*. *Constant* dan *Expression* memerlukan nilai tunggal, sementara Normal, Uniform dan Triangular memerlukan

	beberapa parameter.
Units	Unit waktu untuk parameter *delay*.
Allocation	Menentukan bagaimana waktu dan proses biaya pengolahan akan dialokasikan untuk kesatuan. Proses ini dapat dianggap sebagai nilai tambah, non-nilai tambah, transfer, menunggu atau lain dan biaya yang terkait akan ditambahkan ke kategori yang sesuai untuk entitas dan proses.
Minimum	Parameter untuk menentukan nilai minimum baik untuk distribusi Uniform atau Triangular.
Value	Parameter untuk menentukan *mean* untuk distribusi normal, nilai Konstan waktu tunda, atau *modus* untuk distribusi Triangular.
Maximum	Parameter untuk menentukan nilai maksimum untuk baik Uniform atau distribusi Triangular.
Std Dev	Parameter untuk menentukan standar deviasi untuk distribusi normal
Expression	Parameter untuk menentukan ekspresi yang nilainya dievaluasi dan digunakan untuk waktu pemrosesan *delay*.
Report Statistics	Menentukan apakah atau tidak statistik secara otomatis akan dikumpulkan dan disimpan dalam database laporan untuk proses ini.

d. Decide Module

Modul ini memungkinkan untuk proses pengambilan keputusan dalam sistem. Ini termasuk pilihan untuk membuat keputusan berdasarkan satu atau lebih kondisi (misalnya, jika jenis entitas Kartu Gold) atau berdasarkan pada satu atau lebih probabilitas (misalnya, 75% benar; 25% palsu). Kondisi dapat didasarkan pada nilai atribut (misalnya, Prioritas), nilai-nilai variabel (misalnya, Nomor Ditolak), jenis entitas, atau ekspresi.



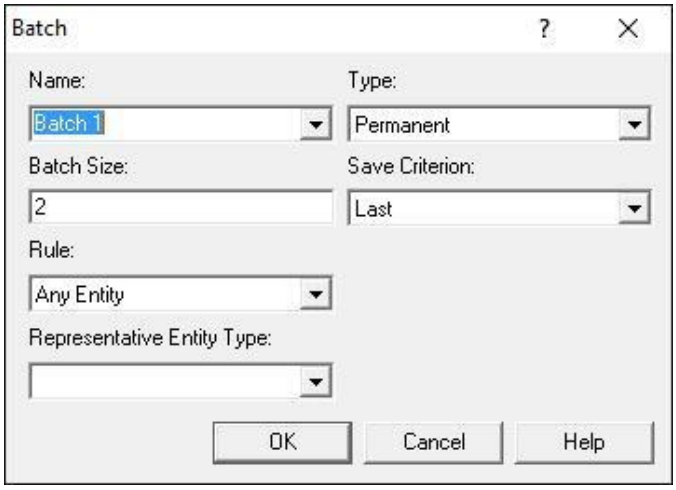
Gambar : *Decide Module*

Prompt	*Description*
Name	*Identifier* modul yang unik ditampilkan pada bentuk modul.
Type	Menunjukkan apakah keputusan didasarkan pada kondisi (jika X> Y) atau secara kebetulan / persentase (60% ya, 40% tidak ada). Jenis yang dapat ditetapkan sebagai baik 2-way atau N-way. 2-way memungkinkan untuk satu kondisi atau probabilitas (ditambah "false" keluar). N-way memungkinkan untuk sejumlah kondisi atau probabilitas yang akan ditentukan serta "else" keluar..
Condition	Mendefinisikan satu atau lebih kondisi digunakan untuk entitas langsung ke modul yang berbeda. Hanya berlaku ketika *Type N-way by Condition*.
Percentages	Mendefinisikan satu atau lebih persentase digunakan untuk entitas langsung ke modul yang berbeda. Berlaku hanya jika *Type N-way by Condition*.
Percent True	Nilai yang akan diperiksa untuk menentukan persentase entitas dikirim keluar yang benar diberikan.
If	Jenis kondisi yang tersedia untuk evaluasi.
Named	Menentukan tidak nama variabel, atribut, atau jenis entitas yang akan dievaluasi kapan entitas memasuki modul. Tidak berlaku ketika *Type* adalah *Expression*.
Row	Menentukan indeks baris untuk array variabel.
Column	Menentukan indeks kolom untuk array variabel.
Is	*Evaluator* untuk kondisi tersebut. Hanya berlaku untuk *Attribute* dan *Variable conditions*.
Value	*Expression* yang akan baik dibandingkan dengan atribut atau variabel, atau yang akan dievaluasi sebagai ekspresi tunggal untuk menentukan apakah itu benar atau salah. Tidak berlaku untuk *Entity Type* kondisi. Jika *Type* adalah *Expression*, nilai ini juga harus mencakup evaluator (misalnya, Warna < > Red).

e. Batch Module

Modul ini sebagai mekanisme pengelompokan dalam model simulasi. *Batch* entitas dapat secara permanen atau sementara dikelompokkan. *batch* sementara harus kemudian dibagi dengan menggunakan modul terpisah. *Batch* dapat dilakukan dengan jumlah tertentu dari entitas memasuki atau mungkin dicocokkan

bersama-sama berdasarkan atribut. Entitas tiba di modul *Batch* ditempatkan dalam antrian sampai jumlah yang diperlukan entitas telah mengumpulkan. Setelah akumulasi, badan perwakilan baru dibuat.



Gambar : *Batch Module*

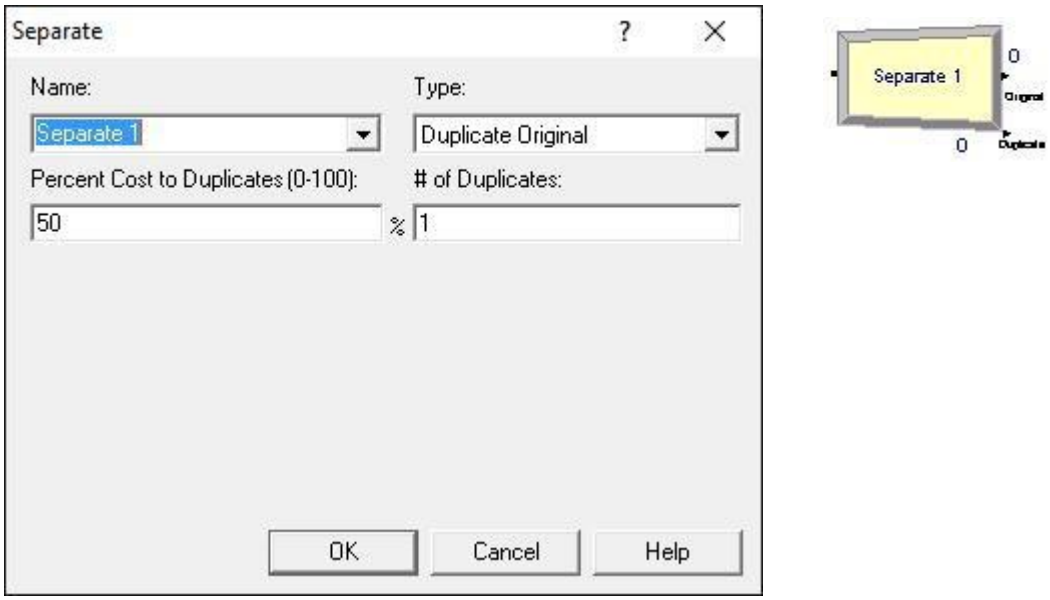
Prompt	***Description***
Name	*Identifier* modul yang unik ditampilkan pada bentuk modul.
Type	*M*etode *batching* entitas bersama-sama.
Batch Size	Jumlah entitas yang akan *batched*.
Save Criterion	Metode untuk nilai yang ditetapkan pengguna untuk menugaskan perwakilan entitas atribut.
Rule	Menentukan bagaimana entitas masuk akan ditumpuk. Setiap *Entity* akan mengambil pertama "*Batch Size*" jumlah entitas dan menempatkan mereka bersama-sama. *By Attribute* menandakan bahwa nilai-nilai dari atribut yang ditentukan harus sesuai untuk entitas untuk dikelompokkan. Misalnya, jika *Attribute Name* adalah warna, semua entitas harus memiliki nilai Color yang sama untuk dikelompokkan. Jika tidak, mereka akan menunggu di modul untuk entitas yang masuk tambahan.
Atributte Name	Nama dari atribut yang nilainya harus sesuai dengan nilai entitas lain yang masuk agar kelompok yang akan dibuat. Nilai atribut dipotong ke integer. Hanya berlaku ketika *Rule* adalah *By Attribute.*

f. Separate Module

Modul ini dapat digunakan untuk meng-copy suatu entitas yang masuk ke dalam beberapa entitas atau untuk membagi sebuah entitas yang sebelumnya ditumpuk. Aturan untuk mengalokasikan biaya dan waktu untuk duplikat

ditentukan. Aturan untuk atribut tugas untuk entitas anggota ditentukan juga. Ketika membelah *batch* yang ada, entitas perwakilan sementara yang dibentuk adalah bekas dan entitas asli yang menjadi thegroup pulih. Entitas melanjutkan berurutan dari modul dalam urutan yang sama di mana mereka awalnya ditambahkan ke *batch*.

Ketika duplikasi entitas, jumlah yang ditentukan salinan dibuat dan dikirim dari modul. Entitas yang masuk asli juga meninggalkan modul.



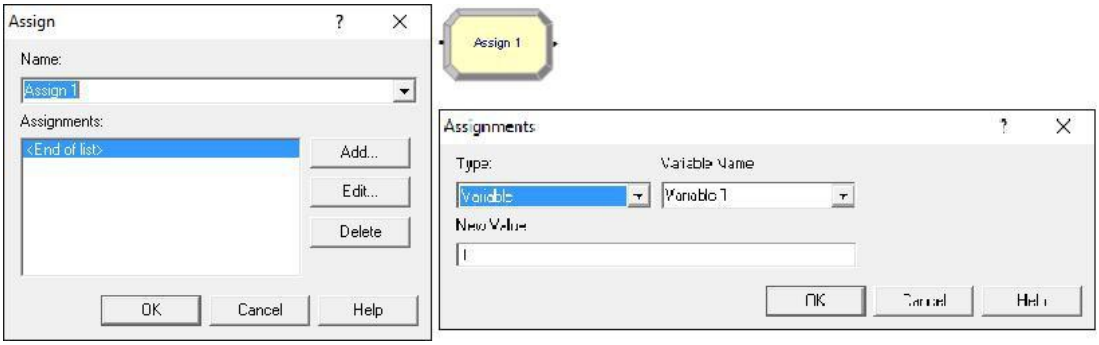
Gambar : *Separate Module*

Prompt	***Description***
Name	*Identifier* modul yang unik ditampilkan pada bentuk modul.
Type	Metode memisahkan entitas yang masuk. *Duplicate Original* hanya akan mengambil entitas asli dan membuat beberapa jumlah duplikat. *Split Existing Batch* yang ada mensyaratkan bahwa entitas yang masuk menjadi entitas sementara ditumpuk menggunakan ***Batch Module***. Entitas asli dari batch yang akan dibagi.
Percent Cost to Duplicates	Alokasi biaya dan waktu dari entitas yang masuk ke duplikat keluar. Nilai ini ditentukan sebagai persentase dari biaya dan waktu (antara 0-100) entitas asli. Persentase tertentu akan dibagi secara merata antara duplikat, sedangkan entitas asli akan mempertahankan tersisa persentase biaya/waktu. Hanya berlaku ketika*Type* adalah *Duplicate Original*.
# of Duplicates	Jumlah entitas keluar yang akan meninggalkan modul, selain entitas yang masuk asli. Hanya berlaku ketika Jenis

	adalah *Duplicate Original*.
Member Attributes	Metode untuk menentukan bagaimana untuk menetapkan entitas perwakilan nilai atribut untuk entitas aslinya. Pilihan ini berhubungan dengan enam atribut tujuan khusus (*Entity.Type, Entity.Picture, Entity.Station, Entity.Sequence, Entity.Jobstep, and Entity.HoldCostRate*) dan semua pengguna didefinisikan atribut. Hanya berlaku ketika Type adalah
Attribute Name	Nama dari perwakilan entitas atribut (s) yang ditugaskan untuk entitas asli dari kelompok. Hanya berlaku ketika Anggota Atribut ini *Take Specific Representative Values*.

g. Assign Module

Modul ini digunakan untuk menetapkan nilai-nilai baru untuk variabel, atribut entitas, jenis entitas, gambar badan, atau variabel sistem lainnya. Beberapa tugas dapat dibuat dengan *Assign Module*.



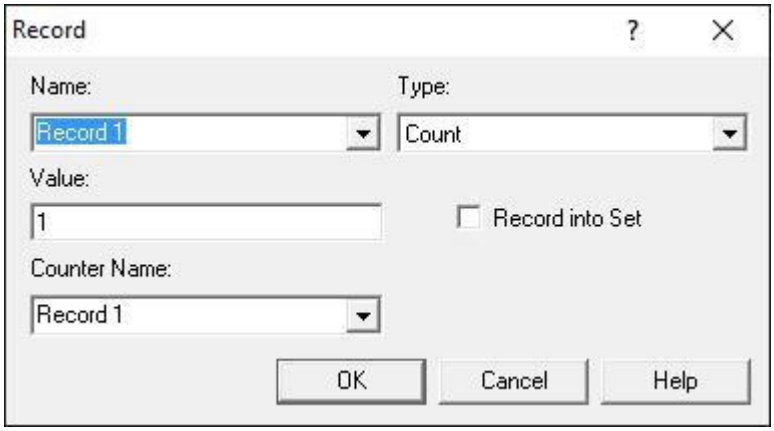
Gambar : *Assign Module*

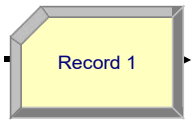
Prompt	***Description***
Name	*Identifier* modul yang unik ditampilkan pada bentuk modul.
Assigments	Menentukan satu atau lebih tugas yang akan dilakukan ketika entitas mengeksekusi modul.
Type	Jenis tugas yang akan dibuat. Lainnya dapat mencakup variabel sistem, seperti kapasitas sumber daya atau waktu simulasi akhir.
Variable Name	Nama variabel yang akan diberi nilai baru ketika entitas memasuki modul. Hanya berlaku ketika *Type* adalah *Variable, Variable Array* (1D), *or Variable Array* (2D).
Attribute Name	Nama atribut entitas yang akan diberi nilai baru ketika entitas memasuki modul. Hanya berlaku ketika *Type* adalah *Attribute.*

New Value	Penugasan nilai atribut, variabel, atau variabel sistem lainnya. Tidak berlaku ketika Type adalah *Entity Type* atau *Entity Picture*.

h. Record Module

Modul ini digunakan untuk mengumpulkan statistik dalam model simulasi. Berbagai jenis statistik observasional yang tersedia, termasuk waktu antara keluar melalui modul, statistik entitas (waktu, biaya, dll), pengamatan umum, dan statistik Interval (dari beberapa cap waktu ke waktu simulasi saat ini). Jenis hitungan statistik juga tersedia. *Tally* dan *Counter* set juga dapat ditentukan.





Gambar : *Record Module*

Prompt	***Description***
Name	*Identifier* modul yang unik ditampilkan pada bentuk modul.
Type	Jenis observasional (tally) atau menghitung statistik yang akan dihasilkan. Menghitung akan menambah atau mengurangi nilai statistik yang ditunjuk oleh nilai yang ditentukan. *Entity Statistics* akan menghasilkan statistik entitas umum, seperti waktu dan biaya/informasi durasi. *Time Interval* akan menghitung dan mencatat perbedaan antara nilai atribut tertentu dan waktu simulasi saat ini. *Time Between* akan melacak dan mencatat waktu antara entitas memasuki modul. *Expression* akan mencatat nilai ekspresi yang ditentukan.
Attribute Name	Nama dari atribut yang nilainya akan digunakan untuk statistik interval. Hanya berlaku ketika Jenis *Time Interval*.
Value	Nilai yang akan disimpan untuk statistik pengamatan ketika *Type Expression* atau ditambahkan ke meja ketika *Type* adalah *Count*.
Tally Name	Mendefinisikan nama simbol penghitungan mana

	pengamatan harus dicatat. Hanya berlaku ketika Jenis adalah *Time Interval, Time Between,* atau *Expression*.
Counter Name	Field ini mendefinisikan nama simbol *counter* untuk kenaikan/penurunan. Hanya berlaku ketika *Type* adalah *counter*.
Record into Set	Kotak centang untuk menentukan apakah suatu penghitungan atau *counter set* akan digunakan.
Tally Set Name	Nama dari penghitungan set yang akan digunakan untuk mencatat jenis pengamatan statistik. Hanya berlaku ketika *Type adalah Time Interval, Time Between, atau Expression.*
Counter Set Name	Nama set meja yang akan digunakan untuk mencatat jenis hitungan statistik. Hanya berlaku ketika *Type* adalah *Count*.
Set Index	Indeks ke dalam penghitungan atau *counter set.*

i. Entity Module

Modul Data ini mendefinisikan berbagai jenis entitas dan nilai-nilai gambar awal mereka dalam simulasi. Informasi awal biaya dan biaya simpan juga didefinisikan untuk entitas.

Prompt	***Description***
Entity Type	*Identifier* modul yang unik ditampilkan pada bentuk modul.
Initial Picture	Representasi grafis dari entitas pada awal simulasi. Nilai ini dapat diubah selama simulasi menggunakan Assign Module.
Holding Cost/Hour	Biaya per jam dari pengolahan entitas melalui sistem. Biaya ini timbul ketika entitas di mana saja dalam sistem.
Initial VA Cost	Nilai biaya awal yang akan ditugaskan untuk nilai tambah biaya atribut entitas. Atribut ini timbul biaya yang dikeluarkan ketika suatu entitas menghabiskan waktu dalam kegiatan nilai tambah.
Initial NVA Cost	Nilai biaya awal yang akan ditugaskan ke non-nilai tambah biaya atribut entitas. Atribut ini timbul biaya yang dikeluarkan ketika suatu entitas menghabiskan waktu di non-nilai tambah kegiatan.
Initial Waiting Cost	Nilai biaya awal yang akan ditugaskan untuk biaya tunggu atribut entitas. Atribut ini timbul biaya yang dikeluarkan ketika entitas adalah menghabiskan waktu dalam kegiatan menunggu, misalnya menunggu untuk batched atau menunggu sumber daya (s) pada modul Proses.
Initial Transfer Cost	Nilai biaya awal yang akan ditugaskan untuk biaya atribut lainnya dari entitas. Atribut ini timbul biaya yang

	dikeluarkan ketika suatu entitas menghabiskan waktu dalam kegiatan transfer.
Initial Other Cost	Nilai biaya awal yang akan ditugaskan untuk biaya atribut lain dari kesatuan. Atribut ini timbul biaya yang dikeluarkan ketika suatu entitas adalah menghabiskan waktu di kegiatan lain.
Report Statistics	Menentukan apakah atau tidak statistik akan dikumpulkan secara otomatis dan disimpan dalam database laporan untuk jenis entitas ini

j. Queue Module

Modul data ini dapat digunakan untuk mengubah aturan peringkat untuk antrian ditentukan. Peringkat aturan default untuk semua antrian adalah First In, First Out kecuali ditentukan lain dalam modul ini. Ada lapangan tambahan yang memungkinkan antrian untuk didefinisikan sebagai bersama.

Prompt	***Description***
Name	Nama antrian yang karakteristik yang didefinisikan.
Type	Peringkat aturan untuk antrian, yang dapat didasarkan pada atribut. Jenis termasuk *First In First Out, Last In First Out*, *Lowest Attribute Value* (pertama) dan *Highest Attribute Value* (pertama). Sebuah nilai atribut yang rendah akan menjadi 0 atau 1, sedangkan nilai tinggi mungkin 200 atau 300
Attribute Name	Atribut yang akan dievaluasi untuk *Lowest Attribute Value* atau *Highest Attribute Value*. Entitas dengan nilai terendah atau tertinggi dari atribut akan peringkat pertama dalam antrian, dengan ikatan yang rusak dengan menggunakan aturan *First In First Out*.
Shared	Centang kotak yang menentukan apakah antrian khusus digunakan di banyak tempat dalam model simulasi.Antrian bersama hanya dapat digunakan untuk menangkap *resource* (misalnya, dengan Seize modul dari panel Proses Lanjutan).
Report Statistics	Menentukan apakah statistik akan dikumpulkan secara otomatis dan disimpan dalam database laporan untuk antrian ini.

k. Resource Module

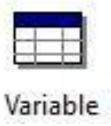
Modul Data ini mendefinisikan *resource* dalam sistem simulasi,

termasuk biaya informasi dan ketersediaan sumber daya. *Resource* memiliki kapasitas tetap yang tidak berbeda selama menjalankan simulasi atau mungkin beroperasi berdasarkan jadwal. Kegagalan *resource* dan *State* juga dapat ditentukan dalam modul ini untuk digunakan dengan *Advanced Pocess* dan *Advanced Transfer Panel* (tidak tersedia di Arena *Basic Edition*).

Prompt	***Description***
Name	Nama *resource* yang karakteristik yang didefinisikan.
Type	Metode untuk menentukan kapasitas untuk *resource*. *Fixed Capacity* tidak akan berubah selama menjalankan simulasi. *Based on Schedule* menandakan bahwa *Schedule Module* yang digunakan untuk menentukan informasi kapasitas dan durasi untuk sumber daya.
Capacity	Jumlah unit *resource* dari nama yang diberikan yang tersedia untuk sistem untuk diproses. Hanya berlaku ketika Jenis adalah *Fixed Capacity* .
Schedule Name	Mengidentifikasi nama jadwal yang akan digunakan oleh *Resource*. Jadwal mendefinisikan kapasitas sumber daya untuk jangka waktu tertentu. Hanya berlaku ketika Type adalah *Schedule*.
Schedule Rule	Menentukan ketika perubahan kapasitas sebenarnya adalah terjadi ketika penurunan kapasitas diperlukan untuk unit sumber daya yang sibuk. Hanya berlaku ketika Type adalah *Schedule*.
Busy / Hour	Biaya per jam dari *Resource* yang memproses suatu entitas. Sumber daya menjadi sibuk ketika itu awalnya dialokasikan untuk suatu entitas dan menjadi siaga ketika dilepaskan. Selama waktu ketika sibuk, biaya akan menumpuk berdasarkan sibuk / biaya jam. Biaya sibuk per jam secara otomatis dikonversi ke unit basis waktu sesuai yang ditentukan dalam halaman Parameter Replikasi item menu Run/Setup.
Idle / Hour	Biaya per jam dari sumber daya yang idle. sumber daya idle ketika sedang tidak memproses suatu entitas. Selama waktu ketika idle, biaya akan menumpuk berdasarkan idle/biaya jam. Biaya menganggur per jam secara otomatis dikonversi ke unit basis waktu sesuai yang ditentukan dalam halaman Parameter Replika item menu Run/Setup
Per Use	Biaya sumber daya secara penggunaan, terlepas dari waktu yang digunakan. Setiap kali sumber daya yang dialokasikan untuk suatu entitas, maka akan dikenakan biaya per

	penggunaan.
State Set Name	Nama dari set menyatakan bahwa sumber daya dapat diberikan selama menjalankan simulasi. Properti ini tidak tersedia di Arena *Basic Edition*.
Initial State	Keadaan awal dari sumber daya. Jika ditentukan, nama harus didefinisikan dalam kelompok pengulangan nama *state*. Bidang ini hanya ditampilkan bila Nama *state* didefinisikan.
Failures	Daftar semua kegagalan yang akan dikaitkan dengan sumber daya. Properti ini tidak tersedia di Arena *Basic Edition*.
Report Statistic	Menentukan apakah atau tidak statistik secara otomatis akan dikumpulkan dan disimpan dalam database laporan untuk sumber daya ini.

l. Variable Module



Modul Data ini digunakan untuk menentukan dimensi dan nilai awal variabel (s). Variabel dapat dirujuk dalam modul-modul lain (misalnya, *Decide Module*), dapat dipindahkan nilai baru dengan modul Pilih, dan dapat digunakan dalam ekspresi apapun.

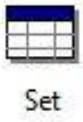
Prompt	***Description***
Name	Nama ariabel yang karakteristik yang didefinisikan.
Rows	Jumlah baris dalam variabel dimensi.
Columns	Jumlah kolom dalam variabel dimensi.
Statistics	Centang untuk menentukan apakah atau tidak statistik akan dikumpulkan dan disimpan dalam database laporan untuk variabel ini. Bidang ini terlihat ketika baris dan kolom yang tidak ditentukan (single variabel).
Clear Option	Mendefinisikan waktu (jika sama sekali) ketika nilai (s) dari variabel yang ulang ke nilai awal (s) ditentukan. Menentukan Statistik menunjukkan untuk me-reset variabel ini untuk nilai awalnya (s) setiap kali statistik dihapus. Menentukan Sistem menunjukkan untuk me-reset variabel ini untuk nilai awalnya (s) setiap kali sistem tersebut akan terhapus. Tak satu pun menunjukkan untuk tidak pernah mengatur ulang variabel ini untuk nilai awalnya (s), kecuali sebelum replikasi pertama. Properti ini tidak tersedia di Arena *Basic Edition*.
Initial Value	Daftar nilai awal (s) dari variabel. Nilai ini (s) dapat berubah dengan *Assign Module*.

m. Schedule Module

Modul data ini dapat digunakan bersama dengan modul *resource* untuk menentukan jadwal operasi untuk sumber daya atau dengan *create* modul untuk menentukan jadwal kedatangan. Selain itu, jadwal dapat digunakan dan dirujuk penundaan waktu berdasarkan faktor waktu simulasi. Durasi jadwal juga didefinisikan dalam modul ini. Kalender jadwal didefinisikan dengan memilih Jadwal kalender, Pola Waktu perintah dari menu Edit.

Prompt	***Description***
Format Type	Format dari jadwal yang ditetapkan. Jika format durasi, jadwal didefinisikan dengan koleksi pasangan nilai-durasi. Jika format kalender, jadwal didefinisikan menggunakan editor pola waktu.
Type	Jenis jadwal yang ditetapkan. Ini mungkin terkait Kapasitas (untuk jadwal *Resource*), Terkait Kedatangan (untuk modul *Create*), atau lainnya.
Time Units	Unit waktu yang digunakan untuk informasi durasi waktu.
Scale Factor	Metode skala jadwal kenaikan atau penurunan Kedatangan / nilai-nilai lainnya. Nilai yang ditentukan akan dikalikan dengan faktor skala untuk menentukan nilai-nilai baru. Tidak tersedia untuk jenis Kapasitas jadwal.
Duration	Daftar nilai dan durasi pasang untuk jadwal. Nilai dapat kapasitas, kedatangan atau nilai-nilai jenis lain, sementara durasi ditentukan dalam satuan waktu. Jadwal pasangan akan mengulangi setelah semua jangka waktu telah selesai, kecuali durasi terakhir dibiarkan kosong (tak terbatas). Jadwal data dapat dimasukkan grafis menggunakan jadwal editor grafis atau manual menggunakan bidang Nilai / Durasi.
Value (Capacity)	Merupakan salah kapasitas sumber daya (jika Type *Capacity*), tingkat kedatangan (jika Type adalah *Arrival*) atau nilai lainnya (jika Jenis lainnya). Contoh lain mungkin menjadi faktor yang digunakan dalam ekspresi penundaan untuk skala waktu tunda selama berbagai bagian dari hari.

n. Set Module



Modul Data ini mendefinisikan berbagai jenis set, termasuk *resource*, *counter*, penghitungan, jenis entitas dan entitas gambar.

Sumber daya set dapat digunakan dalam Proses (*Seize*, *Release, Enter* dan *Leave* untuk *Advanced Process* dan *Advanced Transfer panels*) modul. *Counter* dan *Tally set* dapat digunakan dalam *Record* modul. Queue set dapat digunakan dengan *Seize, Hold, Access,* Request, *Leave* dan *Allocate modules* untuk *Advanced Process* dan *Advanced Transfer panels*.

Prompt	***Description***
Type	Jenis set yang didefinisikan.
Members	Daftar *resource*, *counter tally*, jenis entitas atau gambar anggota dalam set.
Resource Name	Nama dari *resource* dalam set sumber daya. Hanya berlaku ketika *Type* adalah *Resource.*
Tally Name	Nama dari penghitungan dalam penghitungan set. Hanya berlaku ketika *Type* adalah *Tally*.
Counter Name	Nama dari *counter* dalam set meja. Hanya berlaku ketika *Type* adalah *counter*.
Entity Type	Nama dari jenis entitas dalam entitas set tipe. Hanya berlaku ketika Jenis adalah *Entity Type*.
Picture Name	Nama dari gambar dalam gambar ditetapkan. Hanya berlaku ketika Jenis adalah Entity Picture.

2. *Advance Process*

a. *Pickup Module*

Modul *Pickup* menghilangkan sejumlah entitas berturut-turut dari antrian yang diberikan, mulai dari pangkat yang ditentukan dalam antrian. Entitas yang dijemput ditambahkan ke akhir kelompok entitas yang masuk.

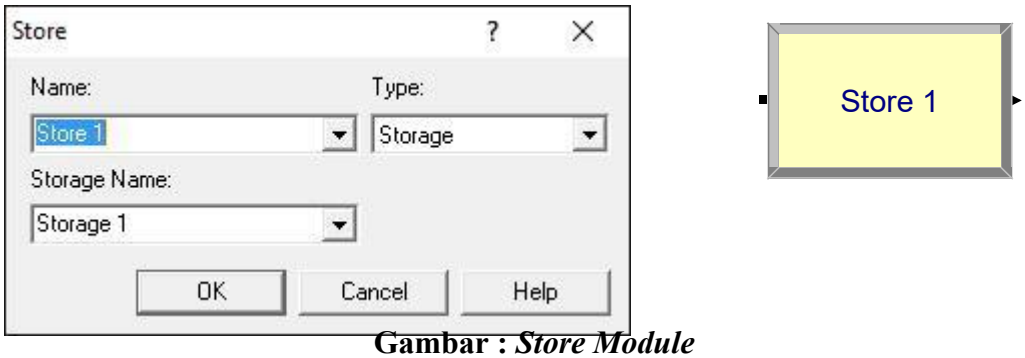
Gambar : *Pickup Module*

Prompt	***Description***
Quantity	Jumlah entitas untuk *Pick Up*.
Queue Name	Nama dari antrian dari mana entitas akan dijemput, mulai

	dari pangkat yang ditentukan.
Starting Rank	Mulai pangkat entitas untuk mengambil dari antrian, Nama Antrian.

b. Store Module

Modul ini menambahkan entitas untuk penyimpanan. Modul *Unstore* kemudian dapat digunakan untuk menghapus entitas dari penyimpanan. Ketika entitas tiba di modul Store, penyimpanan ditentukan bertambah, dan entitas segera bergerak ke modul berikutnya dalam model. Penyimpanan berguna untuk menampilkan entitas animasi sementara entitas mengalami pengolahan dalam modul lainnya. Selain itu, statistik dapat disimpan pada jumlah entitas dalam penyimpanan.

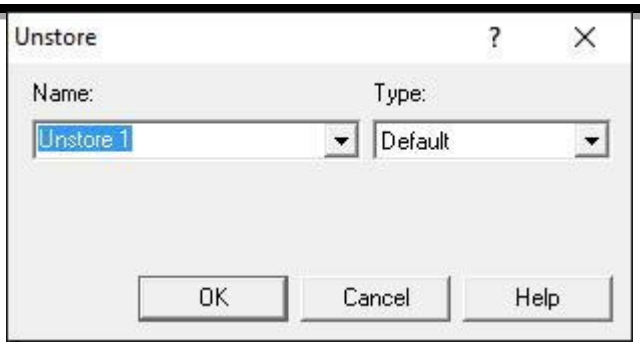


Gambar : *Store Module*

Prompt	***Description***
Type	Menentukan nama penyimpanan, yang mungkin dengan menggunakan satu set, ekspresi, atribut atau hanya nama penyimpanan.
Storage Name	Nama dari penyimpanan ke mana entitas akan ditempatkan. Hanya berlaku ketika Jenis ini *Storage.*

c. Unstore Module

Modul *Unstore* menghilangkan entitas dari penyimpanan. Ketika entitas tiba di modul *Unstore*, penyimpanan ditentukan menurun dan entitas segera bergerak ke modul berikutnya dalam model.



Gambar : *Unstore Module*

Prompt	***Description***
Type	Menentukan nama storage sebagai tempat *storage, set, attribute* atau *expression*. Default akan menghapus entitas dari penyimpanan lalu bahwa ia masuk.
Storage Name	Nama penyimpanan yang mana entitas akan ditambahkan. Hanya berlaku ketika Jenis ini *Storage.*

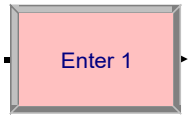
d. Storage Module

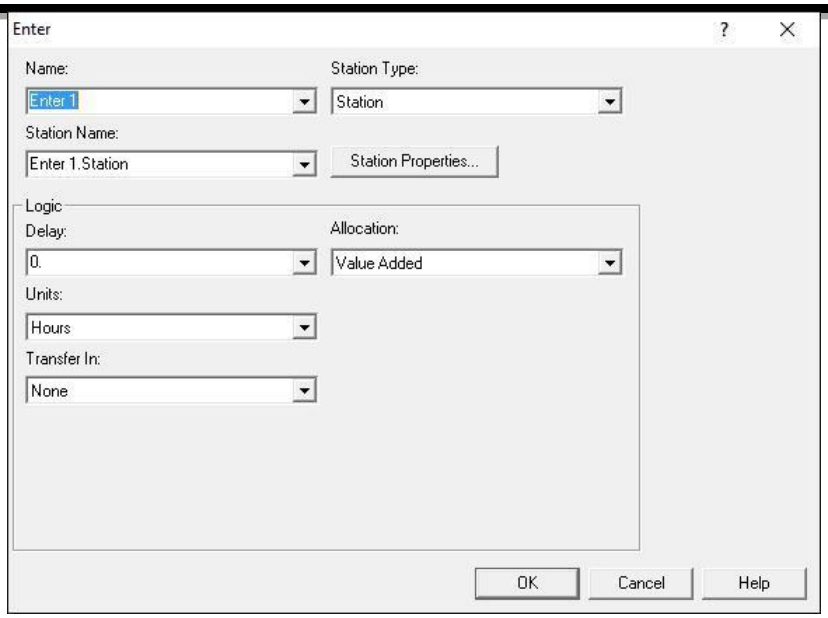
Modul *Storage* mendefinisikan nama penyimpanan. Penyimpanan secara otomatis dibuat oleh modul yang merujuk penyimpanan sehingga modul ini jarang diperlukan. Satu-satunya saat modul ini yang dibutuhkan adalah ketika penyimpanan ditentukan menggunakan atribut atau ekspresi.

3. Advance Transfer

a. Enter Module

Enter modul mendefinisikan sebuah stasiun (atau satu set stasiun) sesuai dengan lokasi fisik atau logis mana pengolahan terjadi. Jika Masukkan modul mendefinisikan satu set stasiun, itu secara efektif mendefinisikan beberapa lokasi pengolahan.



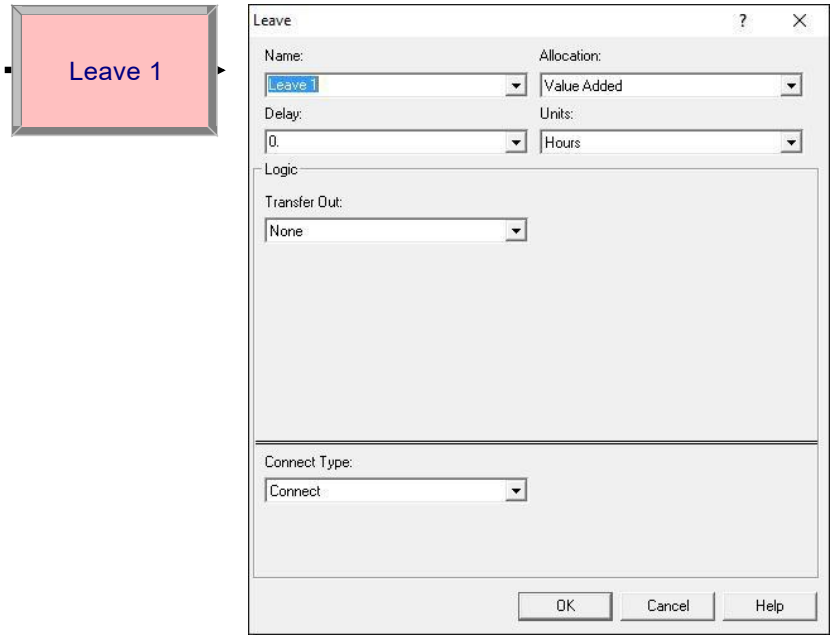


Gambar : *Enter Module*

Prompt	***Description***
Station Type	Menentukan apakah sebuah stasiun individu atau sekumpulan stasiun digunakan untuk mengidentifikasi titik masukan untuk modul ini. Jika Set dipilih, ini menunjukkan bahwa modul ini mendefinisikan pintu masuk ke dalam submodel stasiun generik.
Station Name	Hanya akan terlihat jika Station Type Station, dan ia mendefinisikan nama simbol stasiun yang berhubungan dengan modul ini.
Delay	Mendefinisikan delay yang akan dialami oleh Entitas setelah tiba di stasiun. Penundaan ini biasanya digunakan untuk mewakili waktu yang terkait dengan bongkar entitas dari perangkat pengalihan tiba. Jika suatu entitas tiba di stasiun tanpa perangkat transfer, delay ditentukan masih terjadi.
Allocation	Jenis kategori yang waktu tunda yang terjadi entitas dan biaya akan ditambahkan.
Units	Unit waktu yang digunakan untuk waktu tunda.
Transfer In	Jika *resource*, *transporter* atau *conveyor* digunakan untuk mentransfer entitas untuk stasiun ini, ini dapat digunakan untuk melepaskan, gratis, atau keluar perangkat. Jika *resource* Rilis dipilih, sumber daya tertentu dilepaskan.

b. Leave Module

Modul *Leave* digunakan untuk mentransfer suatu entitas ke stasiun atau modul. Entitas dapat ditransfer dalam salah satu dari dua cara: dapat ditransfer ke modul yang mendefinisikan stasiun dengan referensi stasiun dan routing, menyampaikan, atau transportasi ke stasiun atau koneksi grafis dapat digunakan untuk mentransfer suatu entitas ke modul lain.

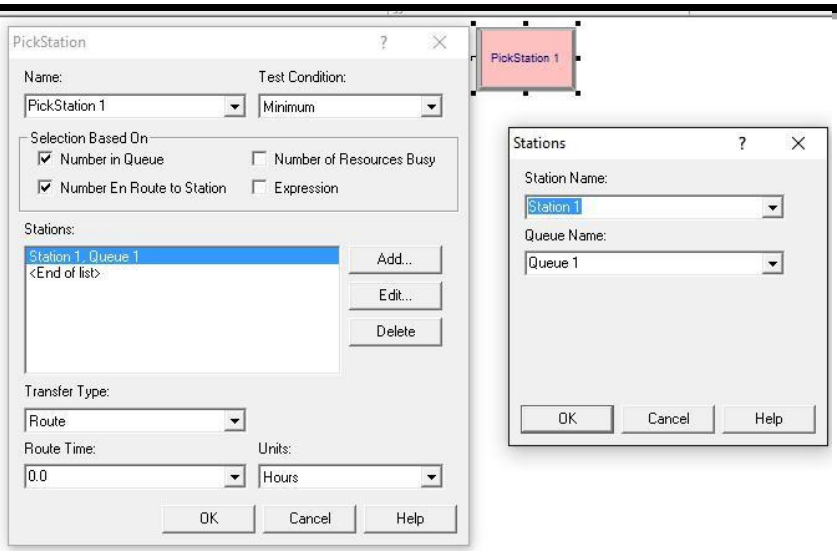


Gambar : *Leave Module*

Prompt	***Description***
Allocation	Jenis kategori yang waktu tunda yang terjadi entitas dan biaya akan ditambahkan.
Delay	Menentukan waktu beban yang timbul setelah mendapatkan perangkat transfer.
Units	Waktu yang digunakan untuk waktu tunda.
Transfer Out	Menentukan apakah *resource*, *transporter*, atau *conveyor* diperlukan sebelum mentransfer entitas dari modul ini.
Connect Type	Menentukan bagaimana suatu entitas harus dipindahkan dari modul ini ke tujuan berikutnya.

c. PickStation Module

Modul *PickStation* memungkinkan suatu entitas untuk memilih stasiun tertentu dari beberapa stasiun tertentu. Modul ini mengambil antara kelompok stasiun berdasarkan logika pilihan didefinisikan dengan modul. entitas mungkin kemudian rute, transportasi, menyampaikan atau terhubung ke stasiun tertentu. Jika metode yang dipilih adalah connect, stasiun yang dipilih ditugaskan untuk atribut entitas.

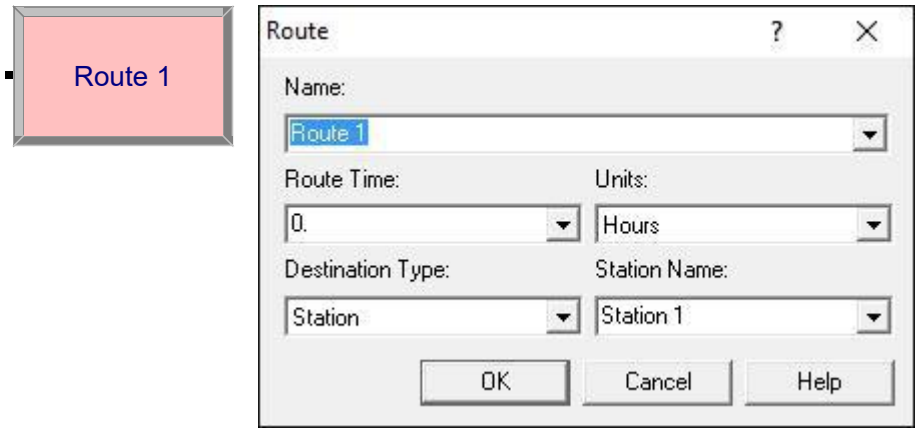


Gambar : *PickStation Module*

Prompt	***Description***
Test Condition	Menentukan kondisi tes yang akan digunakan untuk proses seleksi stasiun. Jika Minimum yang ditentukan, maka minimal semua Pilih Station Berdasarkan kondisi akan dipilih. Jika Maksimum dipilih, maka maksimum semua *Station* Pilihlah Berdasarkan kondisi akan dipilih.
Number in Queue	Menentukan apakah jumlah entitas dalam antrian di stasiun dianggap dalam proses seleksi stasiun.
Number En Route to Station	Menentukan apakah jumlah entitas mentransfer ke stasiun dianggap dalam proses seleksi stasiun.
Number of Resource Busy	Menentukan apakah jumlah sumber daya sibuk di stasiun dianggap dalam proses seleksi stasiun.
Expression	Menentukan apakah ekspresi yang ditetapkan pengguna tambahan dipertimbangkan dalam proses seleksi stasiun.
Transfer Type	Menentukan bagaimana suatu entitas harus dipindahkan dari modul ini untuk stasiun tujuan berikutnya (ditentukan dengan Atribut Simpan).
Route Time	Menentukan jumlah waktu untuk entitas untuk bergerak dari stasiun saat ini ke stasiun ditentukan melalui modul ini. Bidang ini terlihat ketika Jenis Transfer adalah Route.
Units	Waktu yang digunakan untuk waktu rute.

d. Route Module

Modul *Route* transfer entitas ke stasiun tertentu, atau stasiun berikutnya dalam urutan stasiun kunjungan yang ditetapkan untuk entitas. Sebuah penundaan waktu untuk mentransfer ke stasiun berikutnya dapat didefinisikan. Ketika entitas memasuki modul *Route*, atribut *Station* nya (*Entity.Station*) diatur ke stasiun tujuan. Entitas tersebut kemudian dikirim ke stasiun tujuan, menggunakan waktu rute yang ditentukan.

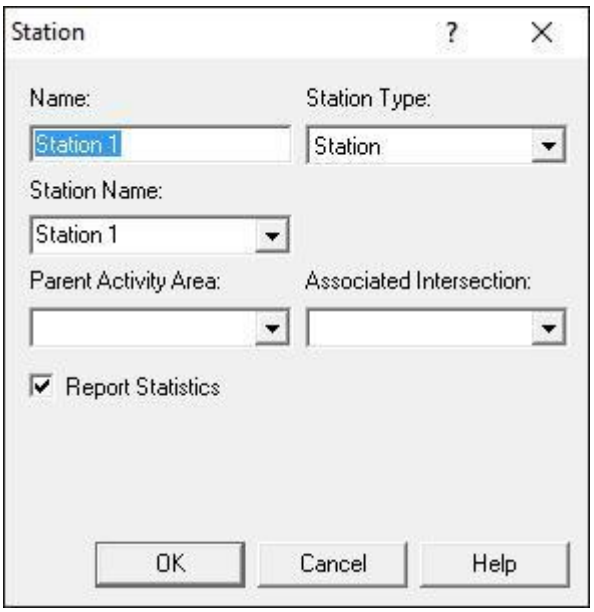


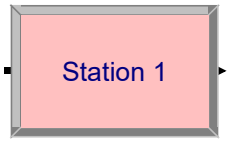
Gambar : *Route Module*

Prompt	***Description***
Route Time	Perjalanan waktu dari entitas lokasi saat ini ke stasiun tujuan.
Units	Waktu yang digunakan untuk waktu rute.
Destination Type	Menentukan lokasi entitas tujuan. Pemilihan dengan urutan mengharuskan entitas telah ditetapkan nama urutan dan urutan sendiri telah didefinisikan.
Station Name	Mendefinisikan nama stasiun tujuan entitas.

e. *Station Module*

Modul *Station* mendefinisikan sebuah stasiun (atau satu set stasiun) sesuai dengan lokasi fisik atau logis mana pengolahan terjadi. Jika modul *Station* mendefinisikan satu set stasiun, itu secara efektif mendefinisikan beberapa lokasi pengolahan.





Gambar : *Station Module*

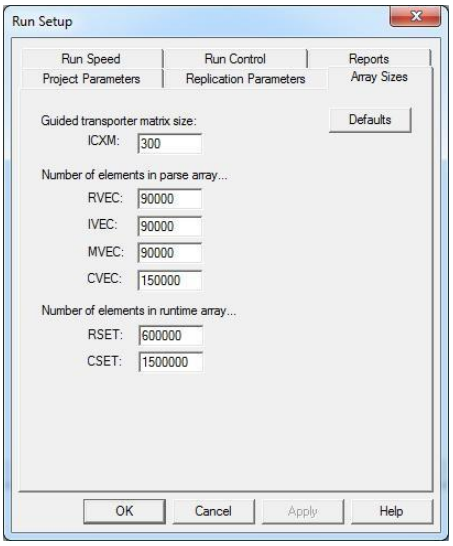
Prompt	***Description***
Station Type	Menentukan jenis stasiun, baik stasiun individu atau satu set stasiun.
Station Name	Mendefinisikan nama simbol stasiun yang berhubungan dengan titik pintu masuk ini.
Parent Activity Area	Nama dari induk Kegiatan Area.
Associated Intersection	Nama persimpangan terkait dengan stasiun ini di jaringan transporter.

K. Aturan Simulasi (*Run Set Up*)

Run set up memberikan akses ke banyak pilihan untuk menjalankan model simulasi. *Setting run set up* dengan memperhatikan kondisi yang ada pada sistem untuk menyelesaikan model simulasi. Menetapkan kondisi *run* dan mengontrol berbagai aspek pada dialog *run set up* model simulasi. Berikut merupakan dialog *run set up* pada *software* ARENA:

1. *Array sizes*

Jika menerima pesan kesalahan yang menunjukkan bahwa dimensi *array* internal telah terlampaui, maka perlu untuk meningkatkan ukuran array untuk model yang dibuat. Jika didalam kotak dialog dilakukan perubahan, maka perubahan itu berlaku untuk semua model yang aktif saja. Setelah dilakukan perubahan, dapat langsung disimpan. Untuk menentukan ukuran semua model baru ada pada tab *array Sizes*

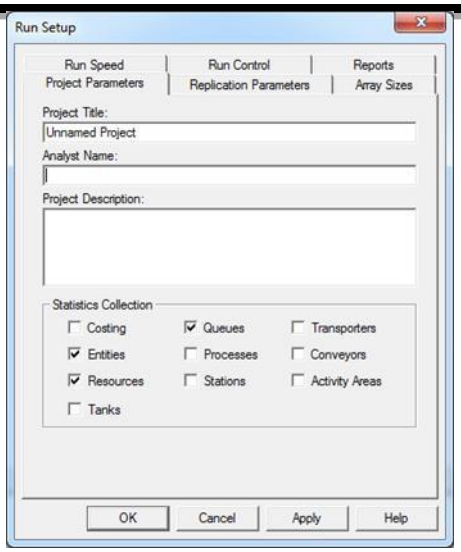


2. *Project Parameter*

Project parameter menyediakan informasi dasar tentang proyek simulasi seperti judul proyek, nama pembuat, dan deskripsi proyek yang dimasukkan kedalam laporan. Berikut merupakan tab yang ada pada *project parameter*:

- *Project title* : judul pada laporan untuk model simulasi
- *Analyst name* : digunakan untuk mengidentifikasi yang melakukan analis laporan studi simulasi
- *Project Description* : digunakan untuk memasukkan catatan dokumentasi model (misalnya, desain skenario, evaluasi).

Pada dialog *project parameter* kita juga bisa memilih koleksi statistik laporan yang kita inginkan seperti entitas, sumber, antrian, proses, dan lain-lain.



3. *Replication Parameter*

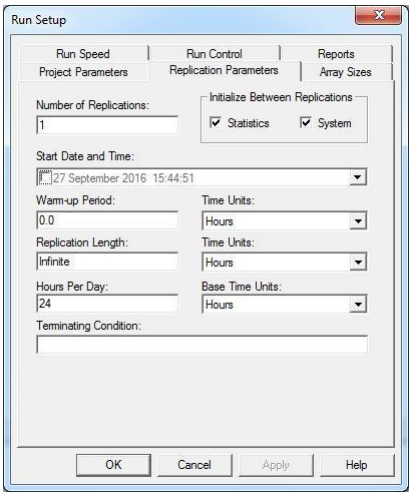
Parameter replikasi menyediakan informasi tentang pengulangan model simulasi. *Replication parameter* digunakan untuk mengatur *properties* model simulasi yang dijalankan/*running*. Ada beberapa tab dalam *Replication parameter* antara lain :

- *Number of replication* : Jumlah replikasi atau pengulangan model simulasi yang akan di *running.*
- *Start date and time* : Tanggal dan waktu mulai model simulasi dijalankan
- *Warm-up period* : Sejumlah waktu yang digunakan untuk menjadikan model kita *steady-state*. Fungsi dari *warm-up period* adalah untuk mengurangi efek random pada simulasi.
- *Replication length* : Jumlah waktu yang ada untuk tiap replikasi model simulasi yang digunakan.
- *Time unit* : Satuan waktu yang digunakan dalam menjalankan model simulai.
- *Hours per day* : Banyaknya waktu per hari saat dilakukan *running* model.
- *Base time unit* : Satuan waktu untuk laporan, status bar, waktu simulasi dan animasi plot. Semua penundaan waktu, panjang replikasi akan di konversi ke satuan waktu dasar ini.
- *Initialize between replications*

Initialize system : Bila ada (di centang), setiap replikasi akan dimulai dengan sistem kosong di saat waktu 0. Bila tidak ada (tidak di centang), hanya replikasi

pertama yang di mulai dengan sistem kosong di saat waktu 0, dan akan terus berlanjut untuk seterusnya. Pekerjaan yang tidak terselesaikan di hari pertama akan di lanjutkan di hari berikutnya.

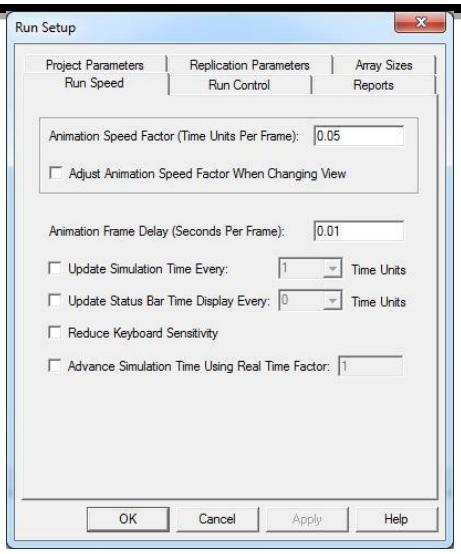
Initialize statistics : Bila ada, report dengan format yang sama akan di *generate* untuk setiap replikasi dan data untuk replikasi tunggal bila *initialize system* dipilih. Bila *initialize system* tidak dipilih, dataseharusnya untuk replikasi pertama dalam report 1, untuk 3 replikasi yang pertama di report 2, etc. Bila tidak ada, report yang di *generate* adalah kumulatif, jadi, report 2 seharusnya terdiri dari statistik dua replikasi yang pertama; report 3 seharusnya terdiri dari statistik tiga replikasi yang pertama, etc.



4. *Run Speed*

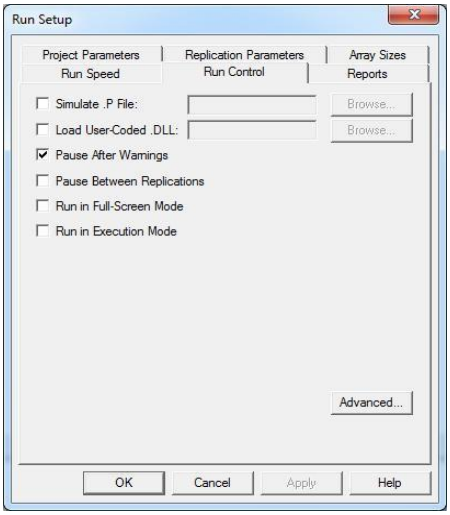
Run speed digunakan untuk mengatur kecepatan *running* model simulasi. Model simulasi yang dijalankan akan bergerak sesuai dengan jumlah dan waktu kecepatan animasi yang telah ditentukan. Beberapa tab yang ada pada *run speed*:

- *Animation Speed Factor (Time Units Per Frame) :* Tempat mengatur jumlah kecepatan animasi yang di inginkan saat *running* model simulasi.
- *Animation Frame Delay (Seconds Per Frame) :* Mengatur waktu untuk animasi setiap *frame* (dalam desimal detik, seperti 0,01 detik).



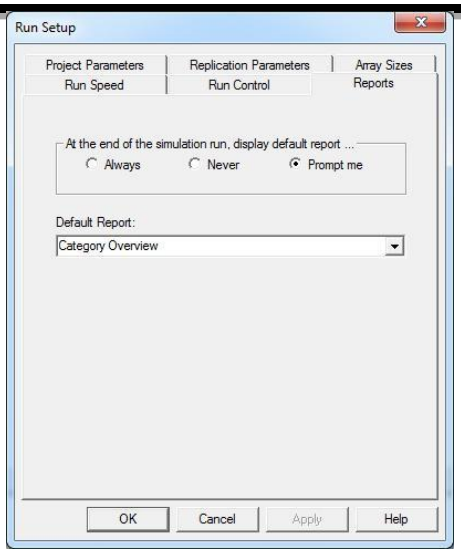
5. *Run control*

Run control merupakan tempat memberikan kontrol apa saja yang berhubungan dengan *running* Arena yang akan kita jalankan. Ada beberapa tab yang ada pada *run control* yang memberikan kita akses pengaturan *running* model simulasi, salah satunya adalah *pause after warning* artinya *running* akan berhenti ketika ada pemberitahuan kesalahan pada model simulasi yang kita buat sebelumnya.



6. *Reports*

Reports merupakan laporan yang akan kita dapatkan setelah akhir dari *running* model simulasi. Ada beberapa *default report* yang ada di Arena, kita dapat memilih jenis *reports* hasil *running* simulasi model yang kita inginkan pada *default report*, misalnya *category overview, category by replication, entities, queues, processes,* dan *frequencies*.



L. Animasi (*Toolbar*)

Toolbar merupakan suatu *window* yang berisi daftar perintah yang sering digunakan dan di presentasikan dalam bentuk tombol. *Toolbar* memudahkan untuk mengatur perintah dan fungsi dalam menggunakan Arena sehingga kita dapat menggunakan *toolbar* yang kita inginkan dengan cepat . Kita dapat dengan mudah menemukan dan menggunakan *toolbar*, misalnya menambahkan, memindahkan, menghapus, membuat dan memberi nama *toolbar* sendiri, menyembunyikan atau menampilkan *toolbar*, memindahkan dan mengubah ukuran *toolbar*. Berikut adalah *toolbar* dan kegunaannya:

1. Standar toolbar

- *New* : Memulai dan membuat model baru
- *Open* : Membuka file model
- *Save* : Menyimpan model
- *Template attach* : Menemukan nama dan lokasi file yang akan ditampilkan
- *Template detach* : Menghilangkan panel yang aktif
- *Print :* Mencetak *project* model
- *Print preview :* Untuk melihat hasil berkas dokumen sebelum dicetak
- *Cut : Untuk memotong / menghilangkan objek*
- *Copy : Untuk menyalin objek ke dalam lembar kerja*
- *Paste : Untuk menampilkan objek yang terekam di layar kerja*

- *Undo/redo :* Mengulangi dan membatalkan perintah
- *Toogle split screen :* Mengaktifkan jendela Arena antara tampilan penuh dan tampilan split-screen
- *View region :* Memperbesar wilayah tertentu dari jendela
- *Zoom :* Memperbesar/memperkecil ukuran jendela *window*
- *Layer :* Beralih di layar benda Arena
- *Submodel :* Menempatkan Submodel kosong ke dalam tampilan model aktif
- *Connect :* Menghubungkan dua modul
- *Edit time pattern :* Membuat dan mengatur waktu *project*
- *Edit exception :* Membuat dan mengelola semua pengecualian, yang ditampilkan dalam daftar ringkasan
- *Display composite view :* Mengelola kapasitas data terkait elemen sistem tertentu
- *Go :* Menjalankan model simulasi
- *Step :* Mempercepat waktu *running* pada satu waktu
- *Fast-forward :* Mempercepat waktu *running*
- *Pause :* Memberhentikan *running* sementara
- *Start over :* Kembali pada kondisi awal sebelum *running*
- *End :* Mengakhiri *running* model
- *Speed :* Mengatur kecepatan *running*
- *Help :* Bantuan *toolbar*

2. *Draw toolbar*

Draw toolbar menyediakan 8 objek menggambar sesuai warna dan jenis/bentuk objek yang di inginkan dalam Arena.

- *Line :* Membuat garis lurus
- *Polyline :* Membuat garis bebas
- *Arc :* Lingkaran berbentuk busur
- *Bezier curve :* Garis melengkung
- *Polygon :* Membuat poligon
- *Ellipse :* Membuat lingkaran elips
- *Text :* Membuat teks pada modul

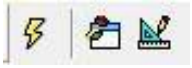
- *Line color :* Membuat warna pada garis
- *Fill color :* Membuat warna pada objek
- *Text color :* Membuat Warna teks
- *Window background color :* Merubah Warna latar jendela
- *Line width :* Merubah Bentuk ketebalan garis
- *Line style :* Memilih jenis garis
- *Arrow style :* Membuat warna garis anak panah
- *Line pattern :* Memilih bentuk garis
- *Fill pattern :* Merubah bentuk objek *toolbar*
- *Show dimension :* Menampilkan dimensi ukuran window

3. *Animate toolbar*

Animate toolbar menyediakan tool objek animasi dasar pada Arena. *Animate toolbar* memberikan bentuk grafis dinamis dalam Arena, tool ini juga sering digunakan untuk proses verifikasi dan validasi model simulasi. Berikut merupakan tool animasi pada *Animate toolbar*

- *Clock :* Animasi bentuk Jam/waktu
- *Date :* Animasi Tanggal
- *Variable :* Animasi nama variabel *tool*/obyek
- *Level :* Animasi level modul
- *Histogram :* Animasi bentuk histogram
- *Plot :* Animasi plot
- *Queue :* Animasi antrian
- *Resource :* Animasi sumber
- *Global :* Animasi gambar global

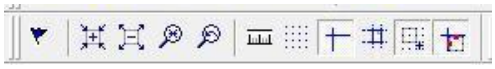
4. *Integration toolbar*



- *Module data transfer :* Mentransfer data modul ke sumber data alternatif lain seperti MS. excel, Acces, dan text
- *Visual basic editor :* Membuat, melihat, debug, dan menjalankan program VBA yang tersimpan dalam model Arena.

- *Design mode :* Menonaktifkan VBA

5. *View toolbar*



- *Managed named view :* Melampirkan nama tampilan untuk setiap model atau jendela gambar
- *Zoom in* : Memperbesar tampilan jendela aktif
- *Zoom out* : Mengembalikan perbesaran jendela
- *View all* : Memperbesar atau keluar dari jendela saat ini untuk menampilkan semua objek dalam satu tampilan
- *View previous* : Kembali ke tampilan terakhir
- *Rules :* Menampilkan atau menonaktifkan tampilan jendela
- *Grid :* Menampilkan atau menutup displai
- *Guides :* Vertikal /horisontal jendela gambar
- *Page breaks :* Menampilkan model yang akan dicetak
- *Snapto grid :* mengaktifkan titik di mana kursor mouse dapat memulai dan kegiatan lengkap
- *Glue to guides :* Menempelkan objek aktif atau tidak aktif

6. *Arrange toolbar*

- *Bring to front :* Membuat posisi suatu objek atau kelompok objek di atas semua objek lain.
- *Send to back :* Posisi suatu objek atau kelompok objek di bawah semua objek lain
- *Group :* Membuat kelompok sejumlah objek individu yang dipilh ke dalam satu unit
- *Ungroup :* Memisahkan kumpulan objek
- *Vertical vlip* : Membalik suatu objek atau kelompok objek 180 derajat terhadap sumbu vertikal.
- *Horizontal vlip :* Membalik suatu objek atau kelompok objek 180 derajat terhadap sumbu horizontal
- *Rotate :* Memutar obyek atau kelompok obyek 90 derajat searah jarum jam
- *Align top :* Membuat dua atau lebih objek dengan posisi tepi atas objek

- *Align bottom :* Membuat dua atau lebih objek dengan posisi tepi bawah objek
- *Align left :* Membuat dua atau lebih objek dengan posisi tepi kiri objek
- *Align right :* Membuat dua atau lebih objek dengan posisi tepi kanan objek
- *Space across :* Mendistribusikan objek yang dipilih secara horizontal, relatif terhadap satu sama lain
- *Space down :* Mendistribusikan objek yang dipilih secara vertikal, relatif terhadap satu sama lain

Snap object to grid : Menempatkan objek atau benda yang dipilih di area gambar

7. *Run interaction toolbar*

- *Check :* Melakukan proses pengecekan pada model
- *Command :* Mengatur dan memonitor jendela aktif saat model dijalankan
- *Breakpoint :* Membuka Debug Bar jika ditutup dan menampilkan jendela Breakpoint
- *Watch :* Membuka Debug Bar jika tertutup dan menampilkan jendela pertama
- *Module break :* Menghentikan modul aktif untuk diatur dan dibersihkan
- *Connection :* Mengaktifkan dan mematikan konektor animasi

8. ***Record macro toolbar***

Catatan makro menyimpan informasi tentang setiap langkah saat melakukan serangkaian perintah. Arena menyimpan laporan Visual Basic yang dibutuhkan untuk mengulang atau "pemutaran ulang" langkah-langkah dalam catatan makro.

- *Resume/pause recording :* Memutar kembali atau memberhentikan rekaman makro
- *Stop recording :* Mengakhiri rekaman makro

9. ***Animate transfer toolbar***

Animated transfer adalah kumpulan modul-modul yang ada di *toolbar* ARENA yang digunakan untuk animasi dari *Advance Transfer*.

- *Storage* : Modul animasi untuk menentukan parameter dari tempat penyimpanan serta animasi dan statistik
- *Seize* : Modul animasi yang mengalokasikan unit dari satu atau lebih sumber daya untuk suatu entitas
- *Parking* : Modul animasi yang menyimpan satu atau lebih transporter (atau sumber daya) yang tidak transit antar stasiun
- *Transporter* : Modul animasi yang memungkinkan definisi perangkat transporter *free path* atau di pandu untuk gerakan entitas dari satu lokasi ke lokasi lain
- *Station* : Modul animasi stasiun pada *advanced transfer*
- *Intersection :* Modul animasi yang digunakan untuk menentukan lokasi dan jalur transporter
- *Route :* Modul animasi rute stasiun model pada *advanced transfer*
- *Segment :* Modul animasi yang menggambarkan secara animasi pergerakan dari konveyor yang ada pada model arena
- *Distance :* Modul animasi yang digunakan untuk menggambarkan secara animasi pergerakan antar stasiun dengan menggunakan transporter
- *Network link :* Modul animasi yang mendefinisikan karakteristik jalur guided-transporter antara persimpangan *beginning intersection name* dan *ending intersection name*
- *Promote path :* Modul animasi untuk mengubah garis statis, polyline, atau kurva Bezier untuk jalur animasi.

SURAT PERNYATAAN

Yang bertanda tangan di bawah ini:

N a m a	:	Utaminingsih Linarti
Kewarganegaraan	:	Indonesia
Alamat	:	Jl. Kol. Sugiyono 74, Desa Brontokusuman RT/RW 068/019, Mergangsan, Yogyakarta

Dengan ini menyatakan bahwa:

1. Karya Cipta yang saya mohonkan:

   Berupa : Panduan Praktikum

   Berjudul : ***Simulasi Komputer dengan Software ARENA 14.0***

   - Tidak meniru dan tidak sama secara esensial dengan Karya Cipta milik pihak lain atau obyek kekayaan intelektual lainnya sebagaimana dimaksud dalam Pasal 68 ayat (2);
   - Bukan merupakan Ekspresi Budaya Tradisional sebagaimana dimaksud dalam Pasal 38;
   - Bukan merupakan Ciptaan yang tidak diketahui penciptanya sebagaimana dimaksud dalam Pasal 39;
   - Bukan merupakan hasil karya yang tidak dilindungi Hak Cipta sebagaimana dimaksud dalam Pasal 41 dan 42;
   - Bukan merupakan Ciptaan seni lukis yang berupa logo atau tanda pembeda yang digunakan sebagai merek dalam perdagangan barang/jasa atau digunakan sebagai lambang organisasi, badan usaha, atau badan hukum sebagaimana dimaksud dalam Pasal 65 dan;
   - Bukan merupakan Ciptaan yang melanggar norma agama, norma susila, ketertiban umum, pertahanan dan keamanan negara atau melanggar peraturan perundang-undangan sebagaimana dimaksud dalam Pasal 74 ayat (1) huruf d Undang-Undang Nomor 28 Tahun 2014 tentang Hak Cipta.

2. Sebagai pemohon mempunyai kewajiban untuk menyimpan asli contoh ciptaan yang dimohonkan dan harus memberikan apabila dibutuhkan untuk kepentingan penyelesaian sengketa perdata maupun pidana sesuai dengan ketentuan perundang-undangan.

3. Karya Cipta yang saya mohonkan pada Angka 1 tersebut di atas tidak pernah dan tidak sedang dalam sengketa pidana dan/atau perdata di Pengadilan.

4. Dalam hal ketentuan sebagaimana dimaksud dalam Angka 1 dan Angka 3 tersebut di atas saya / kami langgar, maka saya / kami bersedia secara sukarela bahwa:
   a. permohonan karya cipta yang saya ajukan dianggap ditarik kembali; atau
   b. Karya Cipta yang telah terdaftar dalam Daftar Umum Ciptaan Direktorat Hak Cipta, Direktorat Jenderal Hak Kekayaan Intelektual, Kementerian Hukum Dan Hak Asasi Manusia R.I dihapuskan sesuai dengan ketentuan perundang-undangan yang berlaku.
   c. Dalam hal kepemilikan Hak Cipta yang dimohonkan secara elektronik sedang dalam berperkara dan/atau sedang dalam gugatan di Pengadilan maka status kepemilikan surat pencatatan elektronik tersebut ditangguhkan menunggu putusan Pengadilan yang berkekuatan hukum tetap.

Demikian Surat pernyataan ini saya/kami buat dengan sebenarnya dan untuk dipergunakan sebagimana mestinya.

ɜ , 11 Februari 2020

(Utaminingsih Linarti)

PROVINSI DAERAH ISTIMEWA YOGYAKARTA
KOTA YOGYAKARTA

NIK : 3471124510820002

Nama : UTAMININGSIH LINARTI
Tempat/Tgl Lahir : JAK-BAR, 05-10-1982
Jenis Kelamin : PEREMPUAN Gol Darah : O
Alamat : JL KOL SUGIYONO 74 YK
RT/RW : 068 / 019
Kel/Desa : BRONTOKUSUMAN
Kecamatan : MERGANGSAN
Agama : ISLAM
Status Perkawinan : BELUM KAWIN
Pekerjaan : KARYAWAN SWASTA
Kewarganegaraan : WNI
Berlaku Hingga : 05-10-2017

KOTA YOGYAKARTA
09-03-2013