

**ÖRNEKLER İLE  
ARENA BENZETİM MODELLEMESİNE  
GİRİŞ**

**Ders Notları**

Müh.Alb. Engin ÇİÇEK

## İÇİNDEKİLER

GİRİŞ .....	1
1. BASİT KUYRUK MODELİ .....	3
2. ONARIM ATÖLYESİ MODELİ .....	19
3. HASTANE ACİL SERVİS MODELİ .....	25
4. ÜRETİM SİSTEMİ MODELİ .....	57
5. MOTORİN DEPOSU ENVANTER KONTROL MODELİ .....	65
6. İMALAT ATÖLYESİ MODELİ .....	83

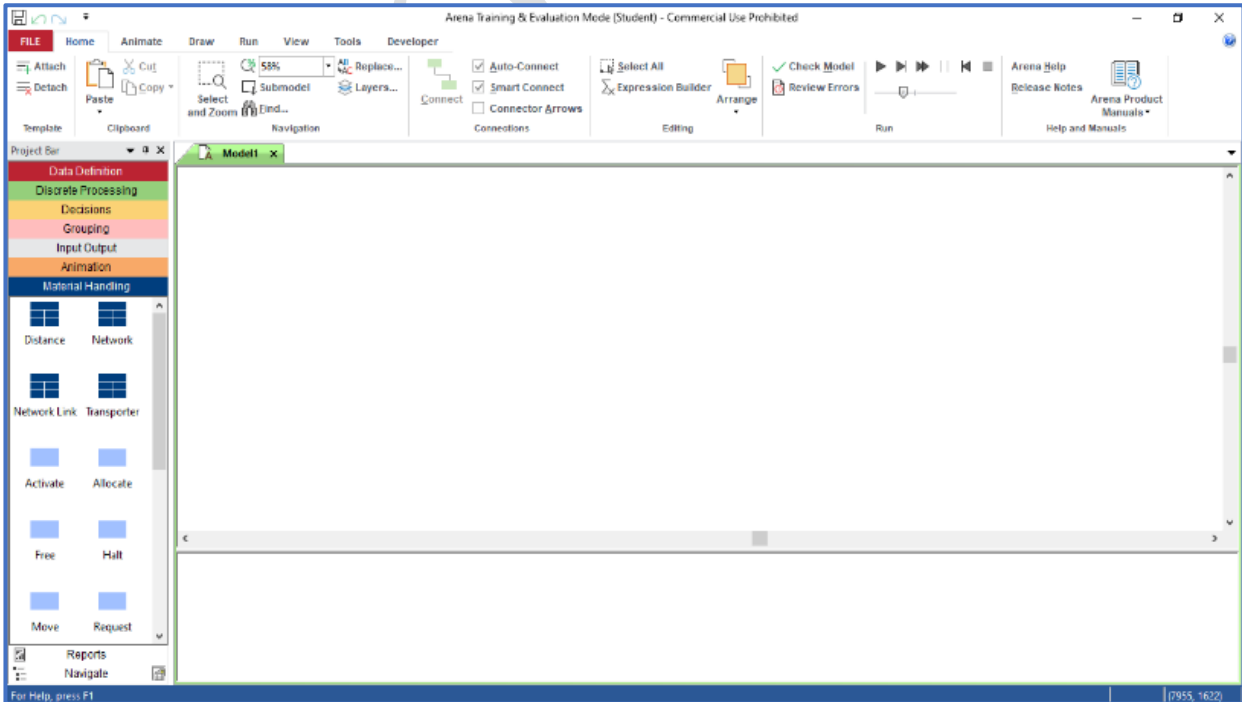
## GİRİŞ

ARENA, kesikli olay benzetimi modellemesi için kullanılan, ticari bir hazır yazılımdır. Windows işletim sisteminde çalışmakta ve kullanıcı ara yüzü diğer Windows programlarındaki menü, araç çubukları vb. araçlar ile çok benzerlik göstermektedir. Linux İşletim Sistemine yüklenememektedir 😞

ARENA programının *Profesyonel Versiyonu* ücretlidir ancak akademik kullanım maksatlı Öğrenci Versiyonu web sitesinden (<http://www.arenasimulation.com>) ücretsiz indirilebilir. Bunun için, websitesi ana sayfasında, “Academic” menüsünden “Students” bağlantısından açılan sayfada “Download Arena for Students” bağlantısına tıklanır. Yeni açılan sayfada yer alan form doldurulur. (eposta adresinin doğru olmasına dikkat edin). Belirttiğiniz adrese gelecek epostada bilgisayarınız için hangi seçenek uygun ise (64 bit veya 32 bit) o seçenek tıklanarak indirme başlatılır. İndirme tamamlanınca indirilen klasörden kurulum yapılır (setup dosyasını kullanarak). Kurulum esnasında direktifler takip edilir. (Benzetim sonuç raporlarının görüntülenmesi için, “SAP Crystal Reports” adlı bir eklentinin de kurulu olması gereklidir, kurulum esnasında bunun ile ilgili soru sorduğunda kabul edin). Programın indirilmesi ve kurulum esnasında Proxy ayarları ile ilgili bir sorun yaşarsanız Proxy ayarlarınızı düşürün.

Program indirip kurulumunu yaptıktan sonra, BAŞLAT menüsünden erişilebilir (*Rockwell Software* ana başlığı altında olacaktır). ARENA öğrenci versiyonu, programını öğrenmek ve bir çok tipte benzetim projesini yapmak için yeterlidir.

Programı açtığınızda Şekil-1’deki görünüm olacaktır. ARENA programındaki tüm komutlar, sembol isimleri İngilizcedir ve program içerisinde bir nesneye isim verilirken Türkçe karakter (ç,ş,ğ,ö,ü,İ) kullanılmaz, kullanılırsa program çalıştığında hata verir.



ARENA Yeni Sürüm (2019) Ana Ekran

2019 yılı itibarı ile internetten indirilen ARENA programı öğrenci yeni versiyondur. Bu dokümanın ilerleyen bölümlerinde de, yeni sürümden ekran görüntüleri üzerinden anlatım yapılacaktır.

Eylül 2022 itibarı ile ARENA 16.2 sürümü çıkmıştır. Bu sürümde: Crystal Reports bulunmamakta, bunun yerine Excel tabanlı Model İstatistikleri raporu yer almakta, Özet İstatistikleri CSV dosyasına aktarılabilen ve Access Report Veri tabanı yerine SQLite report veritabanı bulunmaktadır. Bunun dışında genel kullanım aynıdır.

Ekranın sol tarafındaki “Project Bar” alanında, model kurmak için kullanılan nesneler yer almaktadır. Bu nesnelerden bazıları, farklı şekillerdeki modüllerdir ve model penceresine yerleştirilirler (create, dispose,vb), bazıları da tablo formatında araçlardır ve model penceresine yerleştirilmez ancak seçildiğinde ekranın alt kısmındaki “Tablolama Alanında” görülür ve bu tabloların içinde gerekli tanımlamalar yapılır (Entity, Queue vb).

Model alanına yerleştirilen bir modül üzerine çift tıklandığında, modül ile ilgili detayların görüntülendiği, tanımlamaların yapıldığı ayrı bir pencere açılır.

Model alanına yerleştirilen modüllerin birbirine bağlanması için araç çubuğundan “Connect” (bağlama) aracı kullanılır. Connect ikonuna tıkladığınızda ekrandaki Mouse imlecinin şekli değişir ve bir noktadan diğer bir noktaya bağlanabileceği zaman yeşil ve kırmızı olarak renk değiştirir.

ARENA’daki tüm menüleri ve araçların kullanımını öğrenmenin en iyi yolu keşfederek denemektir.

Bu ders notunda, ARENA’da model kurmayı öğrenmenize yardımcı olabilecek bazı örnek benzetim modelleri, nasıl oluşturulduğuna ilişkin açıklamaları ile yer almaktadır. ARENA ile yapılabilecek benzetim modellerinin tümünü bu ders notlarında yer alması mümkün değildir. Ancak, bu Ders Notu ARENA ile modellemeye başlangıç seviyesi için yeterli detayı vermektedir.

ARENA programı ile ilgili giriş seviyesi kılavuz için “ARENA BASIC USER’S GUIDE” ve daha detaylı bilgi için ise “ARENA VARIABLES GUIDE” referanslarına bakılabilir.

ARENA programının, tüm özelliklerini birkaç uygulama ve örnek ile öğrenmek mümkün değildir. Farklı tipte modellerde ihtiyaç duydukça ve keşfederek daha detaylı özellikleri kullanmak daha uygun bir öğrenme yöntemidir. İnternetteki açık kaynaklardan erişebileceğiniz video ve dokümanlar ile de kendi gelişiminizi desteklemek faydalı olacaktır.

## 1. BASİT KUYRUK MODELİ

İlk örnek olarak, basit bir kuyruk sistemi modeli oluşturulacaktır. Sistemimiz şu şekilde işlemektedir:

Herhangi bir tipteki varlıklar (müşteri, malzeme, vb. herhangi bir kişi, nesne) sistemimize gelmekte, tek bir işlem görmekte (hizmet, montaj, paketlenme, kontrol vb. herhangi bir işlem) ve işi biten varlık sistemden ayrılmaktadır. (varlık = entity)

İşlemi yapmak üzere bir kaynağın (makine, işçi, vb. herhangi bir öge) var olması gerekmekte ve eğer bir varlık sisteme geldiğinde bu kaynak dolu (meşgul) ise sırası gelene kadar bir kuyruğa beklemektedir. (işlem = process , kaynak = resource, kuyruk = queue)

Birden fazla varlık işlem görmek için kuyruğa bekliyorsa, önce gelenin önce işlem göreceği kabul edilecektir (Kuyruk Disiplini : FIFO-First In First Out).

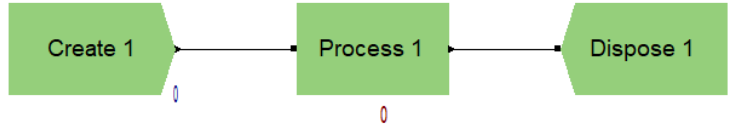
Örneğimizde;

Varlıkların sisteme gelişleri arası geçen süreler, ortalaması 8 dk./varlık (veya başka bir ifade ile 0.125 varlık/dk) olan Üstel Dağılıma uymaktadır.

Bir varlığın makinedeki işlem süresi ise, ortalaması 10 dk./varlık (yani 0.1 varlık/dk) olan Üstel Dağılıma uymaktadır.

### Modelin Oluşturulması

ARENA ekranında sol tarafta bulunan "Project Bar" panelinden; sırasıyla "Create", "Process" ve "Dispose" adlı modülleri fare ile tutup, bırakmadan model ekranına sürükleyin ve bırakın. Eğer sürükleyip bırak işlemleri arasında başka bir işlem yapmazsanız ve yukarıdaki menü alanında "Auto-Connect" ve "Smart-Connect" seçenekleri seçili ise otomatik olarak birbirlerine bağlanmaları gerekir. Ancak otomatik olarak bağlanmamışsa da bu modülleri "Connect" aracını kullanarak da bağlayabilirsiniz. Bu işlemler sonucunda model alanındaki görünüm Şekil 1-1'deki gibi olmalıdır.



Şekil 1-1 Basit Kuyruk Modeli

Şimdi, bu modüller içerisinde, kurmak istediğimiz model ile ilgili gerekli tanımlar yapılacaktır.

Bir modül üzerine çift tıklandığında o modülle ilgili yeni bir pencere açılır. Tanımlamalar bu pencere üzerinde bulunan alanlardaki seçenekler ile veya doğrudan elle uygun ifadeler yazılarak yapılır. Pencere açıkken OK düğmesi ile kapatıldığında, yapılan değişiklikler kaydedilir, CANCEL düğmesine basılarak pencere kapatılır ise en son yapılan değişiklikler kaydedilmemiş olur.

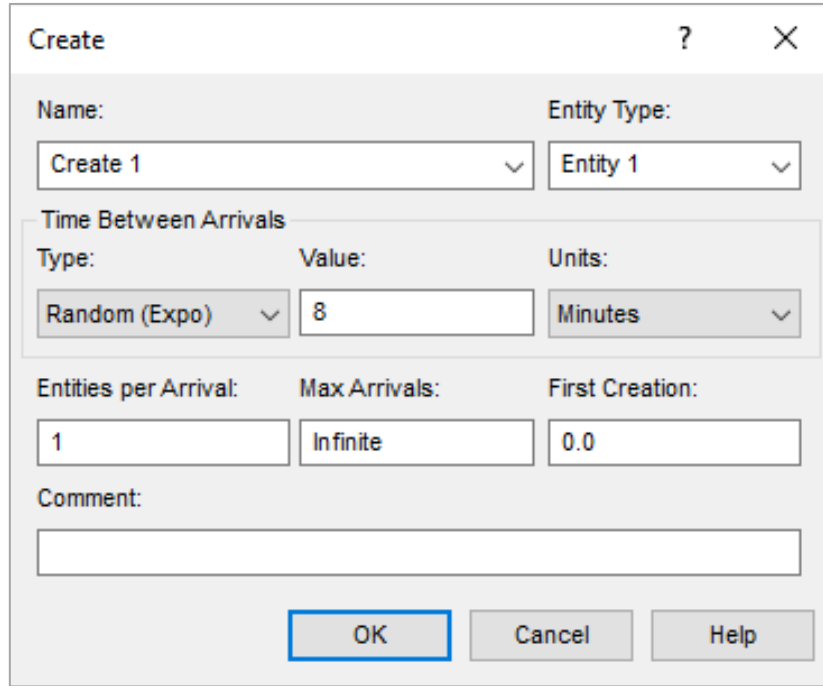
### Create Modülü

Modelimizdeki "Create 1" adlı modül sisteme varlık (entity) gönderen bir Create modülüdür. Bu modüle çift tıklandığında, sisteme girecek varlıkların tipi ve bunların sisteme nasıl girecekleri ile ilgili bazı detayların girilebileceği Şekil 1-3'te görüldüğü gibi bir pencere açılır. Bu pencerede;

"Name" alanına bu modüle bizim vereceğiniz bir isim yazılabilir (örn. bankaya giriş). Bu alana bir metin yazılması zorunlu değildir. Ancak, modelin anlaşılabilir olması için faydalı olabilir. Özellikle, daha sonraki örneklerde göreceğimiz gibi, birden fazla "Create" modülü olan sistem modelleri vardır. Bu durumda, farklı create modüllerini birbirinden ayırt edebilmek için farklı isimler kullanmak anlamlı olacaktır ancak bu modelde "Name" alanı olduğu gibi değiştirmeden bırakılabilir.

“Entity Type” alanı sistemdeki varlıkların tipinin yazılabildiği bir alandır. Buraya yazılan ifade, sistemde hareket halinde olan, işlemler arasında gezinen varlıkların ne olduğudur (örn. parça, hasta, araba, sipariş, mesaj vb.). Bu alan, şimdilik bu örnek için, “Entity 1” olarak bırakılabilir.

“Time Between Arrivals” bölümünde, sisteme girecek varlıkların hangi süre aralıklarında gireceği belirtilir. “Type” alanında “Random (Expo)” yer alırsa, gelişler arası zamanlar üstel dağılıma uygun olacak demektir. “Value” alanına ise bu dağılımın parametre değeri girilir. Örneğimizde, gelişler arası süreler ortalaması 8 dk. üssel dağılıma uygun olduğundan bu alana 8 değeri girilir. Ayrıca zaman birimi olarak “Units” alanında “minutes” seçilir (birimlere dikkat !!)



Şekil 1-2 Create Modülü Penceresi

Kesikli olay benzetim modellerinde sisteme gelişler genellikle Poisson süreci olarak kabul edilir (bu zorunlu bir şey değildir ama büyük oranda böyledir), bu durumda gelişler arası süreler için genellikle “Random (Expo)” yani Üstel Dağılım seçilir. ARENA’nın Create modülü içerisinde varsayılan (default) seçenek olarak Random (Expo) ifadesini göstermesi bu genel yaklaşıma uygundur.

“Type” alanında üç seçenek daha yer almaktadır (Schedule, Constant, Expression):

“**Constant**” seçeneği, adından da anlaşılacağı üzere, gelişler arası sürenin sabit olduğu durumlarda kullanılır ve bu da bazı modellerde karşımıza çıkabilen bir durumdur.

Siparişlerin belirli sabit zamanlarda gelmesi (örn; her gün saat 5’te yani gelişlerarası süre 24 saat) veya öğrencilerin her yıl Eylül ayında üniversiteye başlamaları (okula yeni gelen öğrenci grupları arası geçen süre 12 ay) gibi durumlar, gelişler arası sürenin sabit olmasına örnek gösterilebilir.

“**Schedule**” seçeneği, farklı zaman aralıklarında farklı gelişlerarası zaman olabildiği durumlarda kullanılabilen bir seçenektir. Örneğin, bir kahve dükkanına gelecek müşterilerin günün her saatinde veya haftanın her gününde aynı sıklıkta gelmeyecekleri gerçekçi bir yaklaşımdır. Bu farklılık modele tanımlanmak istenirse Schedule seçeneği buna olanak sağlamaktadır. Daha sonraki örneklerde buna yer verilecektir.

“**Expression**” seçeneği, Üstel dağılımdan başka bir olasılık dağılımı belirtmek veya kendi gözlemlerimize dayanarak belirli özel bir fonksiyon tanımlamak istediğimiz durumlarda kullanılır. Expression, burada matematiksel bir ifade anlamındadır. Bununla ilgili de örnek yapılacaktır.

Create penceresinin daha alt kısmındaki;

“Entities per Arrival” alanında her geliş olayında kaç varlığın birlikte geleceği,

“Max Arrivals” alanında, model çalıştığı süre boyunca toplamda en fazla kaç varlık yaratılacağı ve

“First Creation” alanında, ilk varlığın benzetim başlangıcından ne kadar zaman sonra geleceği

belirtilir. Bu alanları şimdilik bu örnekte değiştirmeyin.

Bazı modellerde sisteme gelişler gruplar halinde olabilmektedir (batch arrival), bu durumda “Entities per Arrival” alanına bu grupların her birinde bulunacak varlık miktarı yazılır. Örneğin; bir makinede işlenmek üzere parçalar, 10’lu gruplar halinde geliyor olabilir (işlemler arası taşıma yapılıyor).

Bazı modellerde, sisteme giriş yapacak toplam varlık adedini de sınırlandırmak isteyebiliriz, bu durumda “max arrival” alanına bir değer yazılır. Örneğin, eğer belirli sayıdaki bir grup insanın hizmet alması veya sisteme girerek işlem göreceği parça sayısının envanterimizde bulunan parçalar ile sınırlı olması modelleyeceğimiz sistem için ayırt edici bir özellik olabilir.

Bazı modellerde de, sisteme gelecek ilk varlığın benzetim başlangıcından belirli bir süre sonra gelmesi istenebilir, bu durumda “First Creation” alanına bu zaman yazılır (modelimizde esas aldığımız zaman biriminden). Bu alanın kullanılması da özellikle birden fazla varlık tipinin sisteme giriş yaptığı modellerde daha anlamlı olabilmektedir.

En altta bulunan “Comments” alanı, serbest metin olarak, açıklamak istenen hususların yazılması için kullanılabilir. Genelde, bir çok modülün yer aldığı karmaşık ve büyük modellerde kullanılır. Şimdilik bu örneğimiz için boş kalabilir.

Yukarıda belirtilen hususlar yapıldığında Create modülü penceresindeki alanlar Şekil 1-2’deki gibi olacaktır. Pencereyi OK düğmesine basarak kapatın ve bir sonraki modülün tanımlamasına geçin (OK düğmesine basarak kapatmazsanız yaptığınız değişiklikler kaydedilmemiş olur).

Modeldeki, “**Process 1**” adlı modül, yapılacak bir işlemi temsil etmektedir (süreç akış diyagramındaki gibi). Bizim örneğimizde 1 tane process modülü var. Modelde yer alacak işlemler için bir sınır yoktur, bir çok Process modülü aynı model içerisinde kullanılabilir. Bu modüle çift tıklandığında detayların girileceği bir pencere açılır. Bu pencerede;

“Name” alanında varsayılan isim olarak “Process 1” ifadesi görülmektedir. Bu alana istenilen bir isim girebilir veya bu şekilde bırakılabilir. Girilen isim yapılacak işlemin (faaliyetin) ne olduğunu temsil eder (örn. boyama, görüşme, kontrol yazılabilir).

“Type” alanında varsayılan ifade olarak “Standard” yazmaktadır. Şimdilik bu alanda bir değişiklik yapmaya gerek yoktur, bu şekilde kalabilir.

“Logic” bölümü işlem ile ilgili asıl detayların girildiği alanları içermektedir ve “Action” alanında varsayılan olarak “Delay” bulunmaktadır. Bir kuyruk modeli oluşturduğumuz için, bu alandaki seçeneklerden “Seize Delay Release” seçilecektir. Böylece, işleme gelecek olan her varlığın önce bir “Resource”, yani kaynak tarafından işlem için alınacağını (seize), işlem süresince belirli bir süre bekleyeceğini, süre harcayacağını (delay) ve sonunda da varlığın serbest bırakacağını (release) ifade etmiş oluruz.

“Priority” alanında varsayılan olarak “Medium(2)” vardır. Bu örnek modelimizde bu alanda da bir değişiklik yapmayacağız, olduğu şekilde bırakacağız.

Process penceresi ilk açıldığında, “Resources” bölümündeki liste boş olarak görünecektir. Bunun nedeni, henüz bu işlemi yapacak olan bir kaynak (örn. bir makine) ataması yapmamış olmamızdır. ARENA programında yapılacak bir işlem için kullanıcı tarafından bu işlemi yapacak kaynak veya kaynaklar ayrıca belirtilmelidir. “Action” alanında “Delay” seçilirse kaynak belirtilmez. “Delay” seçeneği, her hangi bir kaynak kullanmadan belirli bir süre beklemektir. Ancak, “seize delay release” seçildiğinde otomatik olarak kaynak tanımlama alanı da açılır. Her hangi bir kaynak eklemesek modelimiz çalıştırılma esnasında hata verir.

Modelimizde, bu işleme yeni bir kaynak eklemek için “Add...” düğmesine tıklanır. Böylece, bir varlığın bu işlem için hangi kaynağı kullanacağı ve bu işlem için bu kaynaktan aynı anda kaç tanesine ihtiyacı olacağını belirtmiş olacağız.

“Add...” düğmesine tıklandığında yeni bir küçük pencere açılacaktır. Bu yeni küçük pencerede;

“Type” alanında varsayılan olarak “Resource” seçilidir, bunu değiştirmeyeceğiz.

“Resource Name” alanına kaynağımızı tanımlayan bir isim verilebilir (örn. pres makinesi, doktor, gişe memuru vb.) veya olduğu gibi bırakılabilir. Bir çok farklı işlem ve kaynak tipi içeren daha büyük modellerde işlemlere ve kaynaklara isim verilmesi bunların takip edilmesi için faydalı olacaktır ancak bu örnek modelde bu alanı “Resource 1” olarak değiştirmeden bırakabiliriz.

“Units to Seize/Release” alanında bu kaynağın aynı anda kaç adet varlığa işlem yapabileceği belirtilir. Bu örnekte bunu 1 olarak bırakacağız (bazı modellerde bir makine aynı anda 3 parçayı birden işleyebilmektedir bu durumda bu alana 3 yazabiliriz).

Bu küçük yeni pencereyi “OK” düğmesine basarak kapatalım ve tekrar Process penceresi üzerinden devam edelim.

Varlıkların “Process” içerisinde ne kadar zaman geçireceğini belirtmek için öncelikle “Delay Type” alanındaki seçeneklerden birisi seçilmelidir. Bu alandaki listeden “Constant” seçilirse, varlıkların bu işlemde sabit bir süre kalacaklarını belirtmiş oluruz. Diğer seçeneklerden birini yani, Triangular (Üçgen Dağılım), Normal Dağılım ve Uniform (Düzgün Dağılım) seçeneklerinden birini seçersek altında yer alan alanlara bu dağılımların parametreleri girilir. Bu üç dağılım haricindeki bir olasılık dağılımı tanımlamak istersek de “Expression” seçeneği seçilir.

Bu örnek modelde, varlıklar işlemde üssel dağılıma göre ortalama 10 dk. kalacaktır. Bu nedenle;

“Delay Type” olarak “Expression” ve altındaki listeden de “EXPO(mean)” seçilir. “mean” ifadesinin yerine 10 yazılır (yani ifademiz EXPO(10) olur)

“Units” alanında, örneğimizde dakika biriminden değer olduğu için “Minutes” seçilmelidir.

“Allocation” alanında varsayılan olarak “Value Added” seçilidir. Bu seçenek, kaynağın kullanımının varlığa bir değer katacağı anlamına gelir. Bazen, bir kaynak tarafından yapılan işlemler kayma değeri olmayan (non-value added) işlemler olabilir (örneğin; taşıma, yerleştirme vb.). Bu modelimizde Value Added olarak bırakabiliriz.

Bu işlemler bittiğinde, Process penceresi Şekil 1-3’deki gibi görünecektir. “OK” düğmesine basılarak pencere kapatılır. “Process 1” modülü üzerinde bir çizgi görülecektir. Bu çizgi “Process 1” işlemi için bekleyecek olan varlıkların kuyruğunu temsil etmektedir.



Şekil 1-3 Process Modülü Penceresi

### Dispose Modülü

Create modülü ile yaratılarak, sisteme giriş yapan her varlık belirli işlemlerden geçtikten sonra bir "Dispose" modülüne ulaşır. Dispose modülüne üzerine çift tıklandığında açılan pencerede sadece "Name" (isim) alanı bulunur (bakınız Şekil 1-4). Bu alan bu şekilde bırakılabilir veya istenirse bir ifade yazılabilir (örn. çıkış). Pencereyi OK düğmesine basarak kapatabiliriz.

Şekil 1-4 Dispose Modülü Penceresi

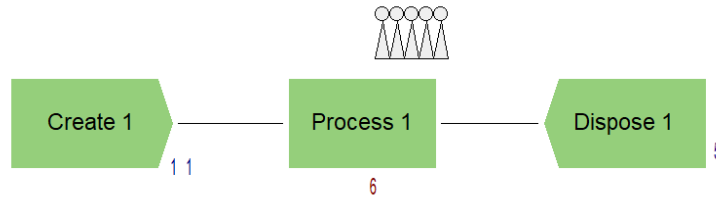
## Modelin İlk Defa Çalıştırılması

Yukarıda açıklanan hususlar tamamlandığında Basit Kuyruk Modelimizi çalıştırabiliriz. Modeli çalıştırmak için araç çubuğundan “RUN” butonuna basılır. Eğer her şey yolundaysa bazı varlıkların hareket ettiğini ve bazen “Process 1” modülü üstündeki kuyrukta bekleyen varlıklar olduğunu görmemiz gerekir. Model çalışırken Şekil 1-5’de gösterilen Run Control üzerinden, model çalışması hızlandırılabilir, yavaşlatılabilir, durdurulabilir, bitirilebilir (deneyerek gözlemleyin).



Şekil 1-5 Run Control

Çalışma esnasında model durdurularak alınan örnek bir ekran görüntüsü Şekil 1-6’da gösterilmiştir. Dikkat edilirse, modüllerin alt kısmında sayılar model çalışırken dinamik olarak değişmektedir. Create modülü altında bulunan 11 sayısı, o ana kadar sisteme giriş yapan varlık sayısını, Dispose modülü altındaki 5 sayısı o ana kadar sistemden çıkış yapmış olan varlık sayısını, Process 1 modülü altında yer alan 6 sayısı, o anda kuyrukta bekleyenler dahil Process1 modülünde bulunan varlıkları ifade etmektedir (5 kişi kuyrukta, 1 kişi işlem görüyor toplam 6 kişi) .



Şekil 1-6 Model Çalışması Esnasında Anlık Bir Ekran Görüntüsü

Modelde modüller arasında hareket eden varlığın şekli, görüntüsü değiştirilebilir. Bunun için, önce çalışan model, Run Control üzerinden ■ üzerine basılarak durdurulması gerekir (Birkaç kez basılması gerekebilir).

Daha sonra, sol kısmında yer alan Project Bar içerisinde “Entity” modülü seçilir (Tablo şeklinde bir modüldür). Pencerenin alt kısmında modelde tanımlanmış varlıklar ile ilgili bazı tanımlamaların yapılabileceği bir tablo görünür. Her satır ayrı bir varlık tipidir. Bu örnek modelde sadece 1 tip varlık tanımladığımız için, Şekil 1-7’deki gibi tek satırlık bir tablo görünür.

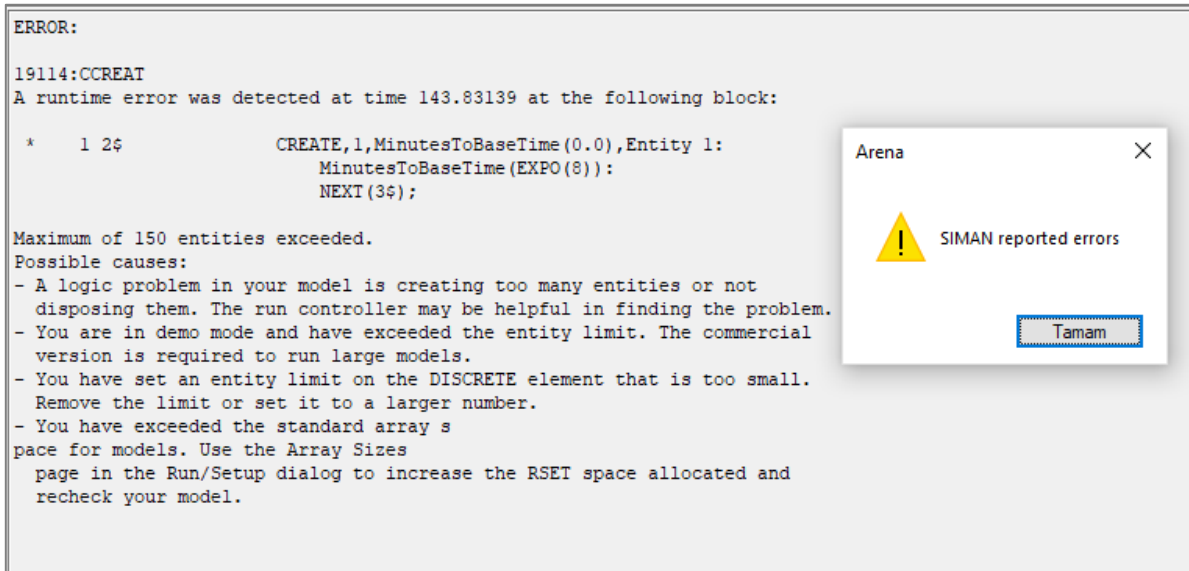
	Entity Type	Initial Picture	Holding Cost / Hour	Initial VA Cost	Initial NVA Cost	Initial Waiting Cost	Initial Tran Cost	Initial Other Cost	Report Statistics	Comment
1 ►	Entity 1	Picture.Report	0.0	0.0	0.0	0.0	0.0	0.0	<input checked="" type="checkbox"/>	

Şekil 1-7 Entity Modülü Tablosu Görünümü

Tablonun ikinci sütunundaki “Initial Picture” üzerinden farklı bir varlık görünüm alternatifi seçilebilir. (Varlığın görünümünün modelin çalışmasına bir etkisi yoktur.)

Modelimizi çalıştırılmaya devam ettirdiğimizde, bir süre sonra Şekil 1-8’deki gibi bir hata mesajı görünecektir. Bunun nedeni, ARENA öğrenci versiyonunun model içerisinde aynı anda 150 varlık olmasına izin vermesidir. Bu hatanın oluşması doğaldır, çünkü modelimizde sisteme gelişler arası ortalama süre, ortalama işlem süresinden azdır. Böyle bir tek sunumculu basit kuyruk modelinde kuyrukta biriken varlık sayısı ara ara azalıp artsa da, uzun vadede artması beklenir.

Ancak, model çalışması esnasında aynı anda sistemde bulunan varlık sayısı 150’nin üzerine çıkmadığı sürece, istediğimiz kadar uzun süre modelimizi çalıştırabiliriz.



Şekil 1-8 ARENA Öğrenci versiyonu Hata Mesajı

Şimdi, modelimizde ufak bir değişiklik yaparak, işlemde (Process 1) 1 yerine 2 kaynak kullanılacağını modele tanımlayalım (1 makine yerine birbirine paralel çalışabilen 2 makine, yani aynı anda farklı iki kaynakta 2 ayrı varlığı işleyebiliriz).

Diğer tüm hususlar aynı kaldığı sürece, yapacağımız bu değişikliğin “Process 1” adlı işlemin önündeki kuyrukta oluşacak birikmeyi azaltmasını bekleyebiliriz.

Bir işlemde kullanılan kaynak sayısını değiştirmek (artırmak veya azaltmak) için, Şekil 1-9’da görüldüğü gibi, sol taraftaki Project Bar bölümünden “Resources” modülü seçilir.

Entity modülüne benzer şekilde, model alanının altı kısmında bir tablo görünecektir. Bu tabloda, modelimizdeki tüm resource (kaynaklar) satır listelenmektedir (Bizim modelimizde 1 adet resource olduğu için 1 satır görünecektir). Bu satırdaki “Capacity” sütunundaki değer 2 yapılarak, kaynak sayımızı 1 artırabiliriz.

	Name	Type	Capacity	Busy / Hour	Idle / Hour	Per Use	StateSet Name	Failures	Report Statistics	Comment
1 ▶	Resource 1	Fixed Capacity	2	0.0	0.0	0.0		0 rows	<input checked="" type="checkbox"/>	

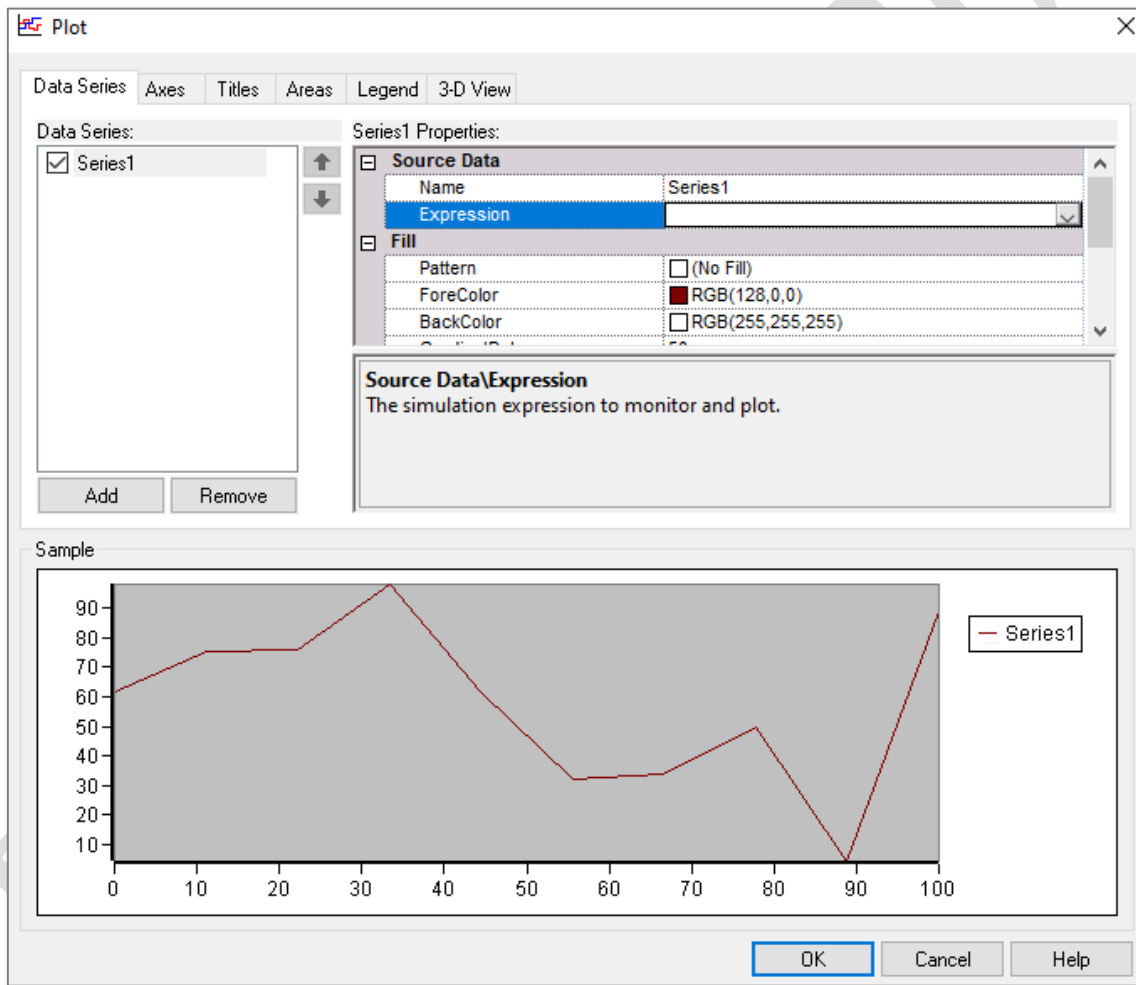
Double-click here to add a new row.

Şekil 1-9 Resource Modülü Tablo Görünümü

## Modeldeki Değişken ve Çıktılarının Grafik veya Sayaç ile Gösterimi

Model çalışırken, ekranda modele ilişkin bazı değişkenlerin aldığı değerler dinamik olarak grafik veya sayaç olarak gösterilebilir. Bu sayede, sistemde takip etmek istediğimiz bazı önemli çıktıların zaman içerisinde nasıl değiştiği, en fazla ne değere ulaştığı, genellikle hangi değerler arasında olduğu vb. hususları da gözlemlenebilir. Grafikteki değerler benzetim ile eş zamanlı olarak değişir, benzetimin yavaşlatılması veya hızlandırılması grafiğe de aynı şekilde etki edecektir.

Bu örnekte, kuyrukta bekleyen varlık sayısının zamana göre değişimini gösteren bir grafiği model sayfasına ekleyeceğiz. Bunun için, programın “Animate” menüsünden “Charts > Plots” seçeneğine girilir. Açılan “Plot” başlıklı pencerede grafik tanımlanır. Önce, hangi verileri grafikte görmek istediğimizi belirtmek üzere, açılan pencerede “Add..” düğmesine basılır. Ekran görüntüsünün Şekil 1-10’daki gibi olacaktır.

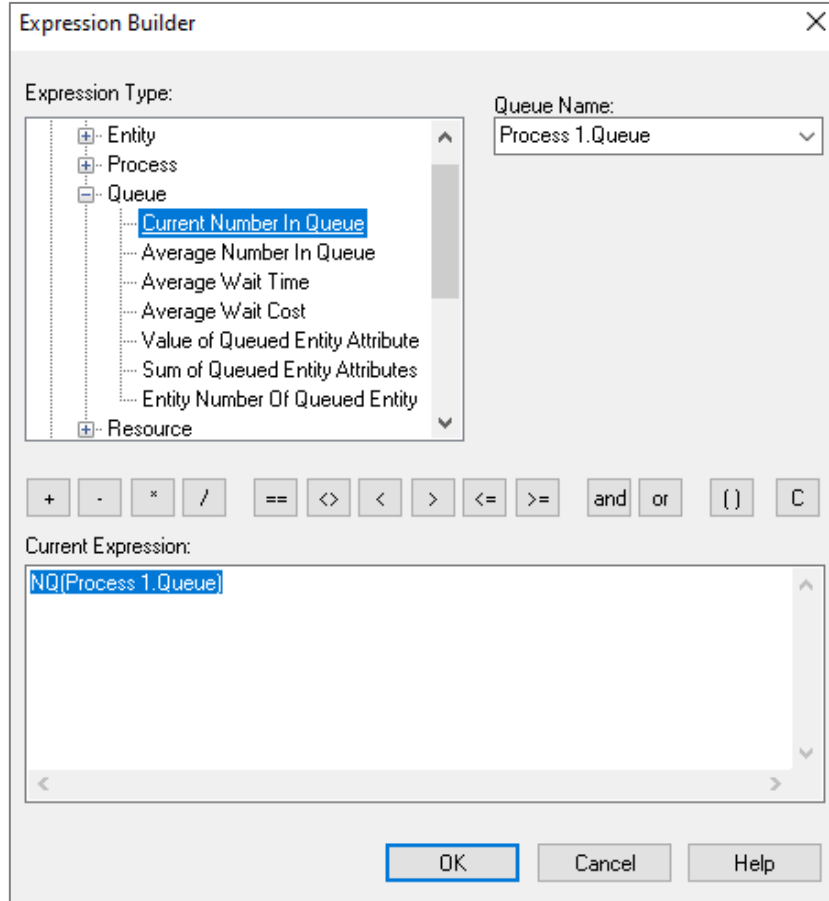


Şekil 1-10 Plot Penceresi Görünümü

Burada, “Series 1” grafiği çizilecek veri setidir ve alacağı değerler “Series 1 Properties” listesindeki “Expression” alanında belirtilir. Bu alanda, “Create1.NumberOut”, “Dispose1.NumberOut”, “Process1.NumberIn” ve “Process1.NumberOut” ifadeleri ile birlikte en üstte “Build Expression...” seçeneği yer almaktadır. Bu, Arena’nın en önemli özelliklerinden olan “Expression Builder” yani “deyim kurucu” olarak adlandırılabilir bir alt uygulamayı yeni bir pencerede açar.

“Expression Builder”, Arena’da model geliştirme esnasında farklı bir çok yerde kullanılabilir. (Hazır bir çok komut ve fonksiyonların seçebileceği veya istenirse kendimiz için ayrı bir fonksiyon oluşturabilecek bir uygulama olarak düşünülebilir).

Örneğin bu örnek için kuyruktaki varlık sayısının grafiğini çizmek istiyorsak; Expression Builder penceresinde, “Expression Type” alanında “Current Model Variables and Functions” altındaki “Queue” içerisindeki “Current Number In Queue” seçilir (Şekil 1-11’de gösterildiği şekilde).



Şekil 1-11 Expression Builder Penceresi

Bu seçim yapıldığında, penceredeki “Current Expression” alanında “NQ(Process 1.Queue)” ifadesi otomatikman gelecektir. Bu ifade, ARENA programının içerisindeki hazır fonksiyonlardan birisidir (NQ fonksiyonun ismi yani Number in the Queue, fonksiyonun parametresi de Process 1.Queue). Bu alanda istersek, kendimiz yeni ifadeler de oluşturabiliriz (İleriki modellerde bu konu ele alınacaktır).

Önce, Expression Builder penceresini sonra da Plot penceresini “OK” düğmelerine basarak kapatın ve mouse ile grafiğinizi ekranda yerleştirmek istediğiniz yeri seçerek boyutunu ayarlayın. Ekranda mouse imleci + şeklinde görünecektir. İlk tıklama ile yer belirlenir, mouse ile belirli genişlik ayarlandıktan sonra ikinci tıklama ile grafik yerleştirilir. Daha sonra istediğiniz yere taşıyabilirsiniz.

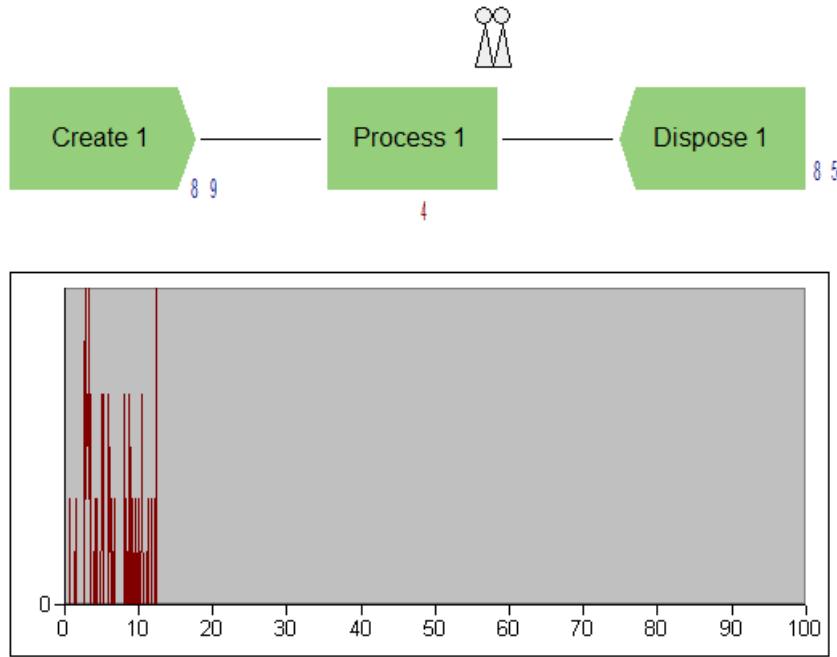
Programı tekrar çalıştırdığımızda grafiğin, Şekil 1-12’deki gibi, dinamik olarak kuyrukta bekleyen varlık sayısını göstermesi gerekmektedir (yatay eksen zaman, dikey eksen kişi sayısını ifade eder).

Ekrana birden fazla grafik eklenebilir. Her yeni eklenecek grafik için yukarıda belirtilen yol izlenir. Oluşturduğunuz bir grafikte daha sonra düzenlemeler (şekil, görünüm veya gösterilecek veri vb.) yapmak isterseniz, grafiğin üzerine çift tıklayarak, Plot penceresinden düzenleme yapabilirsiniz.

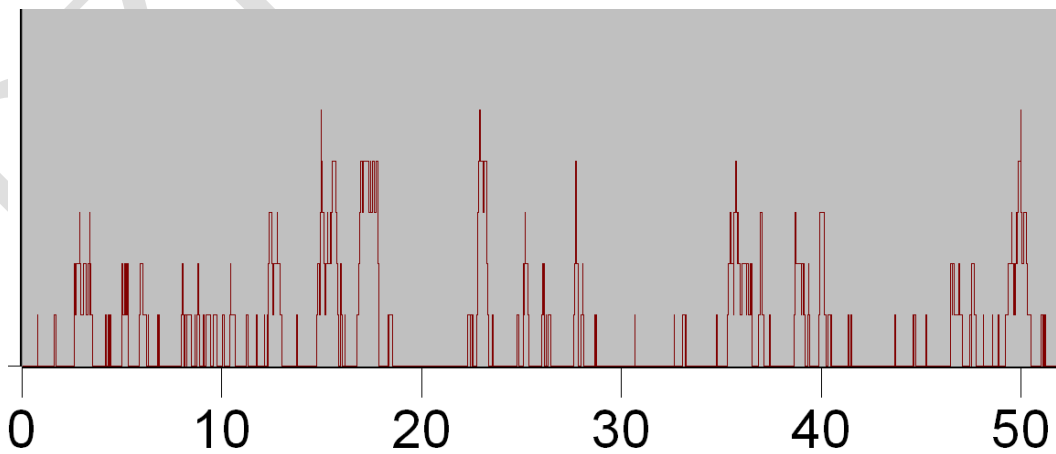
Grafiklerin eksen isimleri, çizgi ve arka plan rengi, vb. diğer hususları bu pencerede denemeler ile keşfedebilirsiniz. (Örneğin; Plot penceresinde Properties > Line > Draw Mode için “Stairs” seçeneği, kuyrukta bekleyen kişi sayısı gibi kesikli değişkenlerin grafiği için daha uygun olabilir).

Grafiklerden ayrı olarak bazı değişkenlerin değerlerini, ekranda sayısal olarak bir sayaç içerisinde görmek bazen tercih edilebilir. Bunun için, Animate menüsünden “Variable” seçeneği tıklanır. Açılan pencerede, “Expression” alanında “Build Expression” seçilerek grafik oluşturmada olduğu gibi sergilenmek istenen ifadenin tanımlaması “Expression Builder” penceresinde yapılabilir.

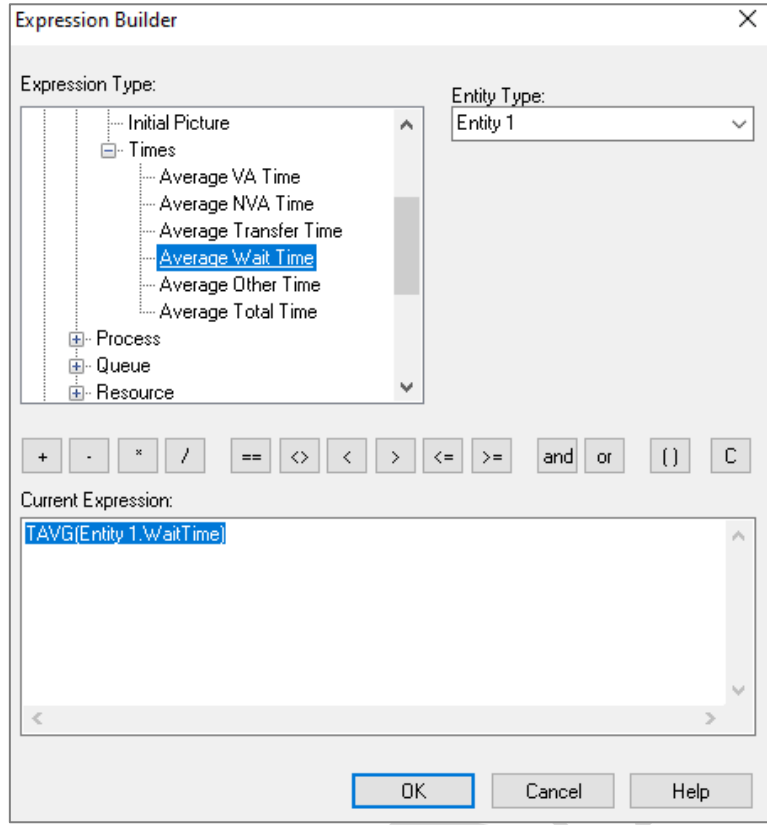
Bu örnekte, kişi başına ortalama bekleme süresi değişkenini ekranda dinamik olarak görüntülemek istersek; “Expression Builder” penceresinde “Entity > Times > Avg Wait Time” seçeneği işaretlenir. Bu pencerenin alt kısmındaki alanda “TAVG(Entity 1.WaitTime)” ifadesi görülecektir (Şekil 1-13).



Şekil 1-12.a. Modelin Kuyrukta Bekleyen Varlık Sayısı Grafiği ile Birlikte Çalışması

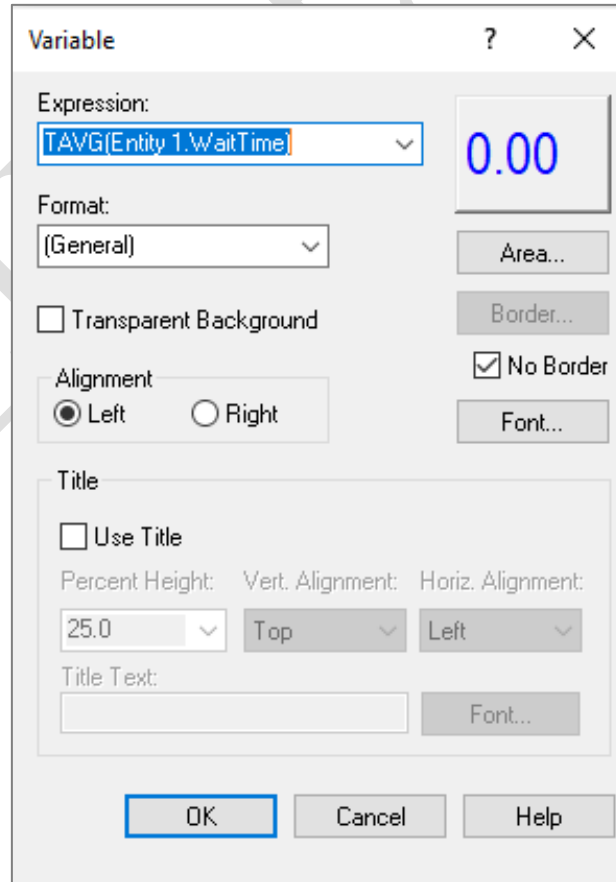


Şekil 1-12.b. Grafiğin Yakınlaştırılmış Ekran Görüntüsü



Şekil 1-13 Expression Builder Penceresi

Expression Builder penceresini kapatınca tekrar “Variable” penceresine dönülür (Şekil 1-14)



Şekil 1-14 Sayaç Eklenmiş Şekilde Model Çalıştırması

Bu pencereyi de OK düğmesi ile kapattıktan sonra, grafik ekleme işleminde olduğu gibi, sayaç ekrana yerleştirilir. Modelimizi tekrar çalıştırdığımızda Şekil 1-15'deki gibi bir görüntü olacaktır.

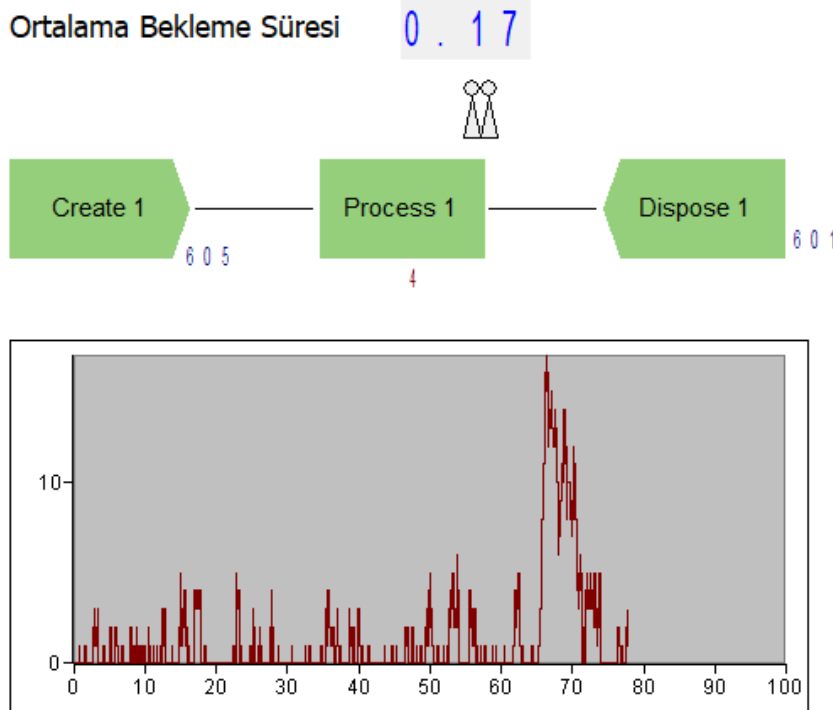
ARENA programında temel zaman birimi varsayılan olarak saattir. Şekil 1-16'da görülen 0.17 değeri de saat birimindedir. Bu değerin; dakika olarak görülmesini istersek;

- Expression Builder penceresinde yazan (Şekil 1-14'te gösterildiği gibi), TAVG(Entity 1.WaitTime) ifadesini 60 ile çarpabiliriz, yani ifadeyi  $TAVG(Entity 1.WaitTime)*60$  yaparız, veya
- RUN menüsünde "Setup" tıklanarak, burada "Replication Parameters" sekmesindeki "Base Time Unit" değerini Minutes olarak değiştirebiliriz.

ARENA programında model oluşturulurken zaman birimlerine çok dikkat edilmelidir. Gözden kaçırılan bir birim yanlışlığı, modelin istenildiği gibi çalışmamasına yol açacaktır ve çoğu zaman bu durumun sanki modelde bir mantık hatası varmış gibi algılanmasına ve zaman kaybına yol açabilecektir.

Eğer modelinizde farklı birimlerde verilmiş zaman değişkenleri var ise, tümünü aynı birime dönüştürün (Genellikle dakika seviyesinde işlemler var ise tümünü dakika olarak belirtin) ve RUN>SetUp>Replication Parameters>Base Time Units için de aynı birimi belirleyin.

Şekilde yer alan "Ortalama Bekleme Süresi" yazısı düz bir metin olarak menülerden eklenmiştir. ARENA'da model alanına bu şekilde düz metin yazılabilir.



Şekil 1-15 Sayaç Eklenmiş Şekilde Model Çalıştırması



## Model Çalışma Süresinin Tanımlanması

Benzetim modeli çalıştırıldığında daha sağlıklı çıktılar görülebilmesi için belirli bir süre çalışması arzu edilir, ancak bir yerde de bitirip sonuçları almamız gerekir.

Bu örnek modeli çalıştırdığımızda, öğrenci versiyonu için geçerli olan, aynı anda en fazla 150 varlık sınırına ulaşmadığımız sürece, hiç durmadan çalışmaya devam edecektir. Bunun nedeni modelin “Run Length”inin yani çalışma süresinin varsayılan değerinin sonsuz olmasıdır.

Modelin çalışma süresine ilişkin parametrelere “Run > Setup” üzerinden ulaşılabilir. Bu pencerede, Replication Parameters sekmesinde;

“Number of Replications” modelin birbirinden bağımsız olarak kaç kere baştan başlatılarak tekrar tekrar çalıştırılacağını belirtir. Bu alanın varsayılan değeri 1’dir (şimdilik bu değer kalabilir).

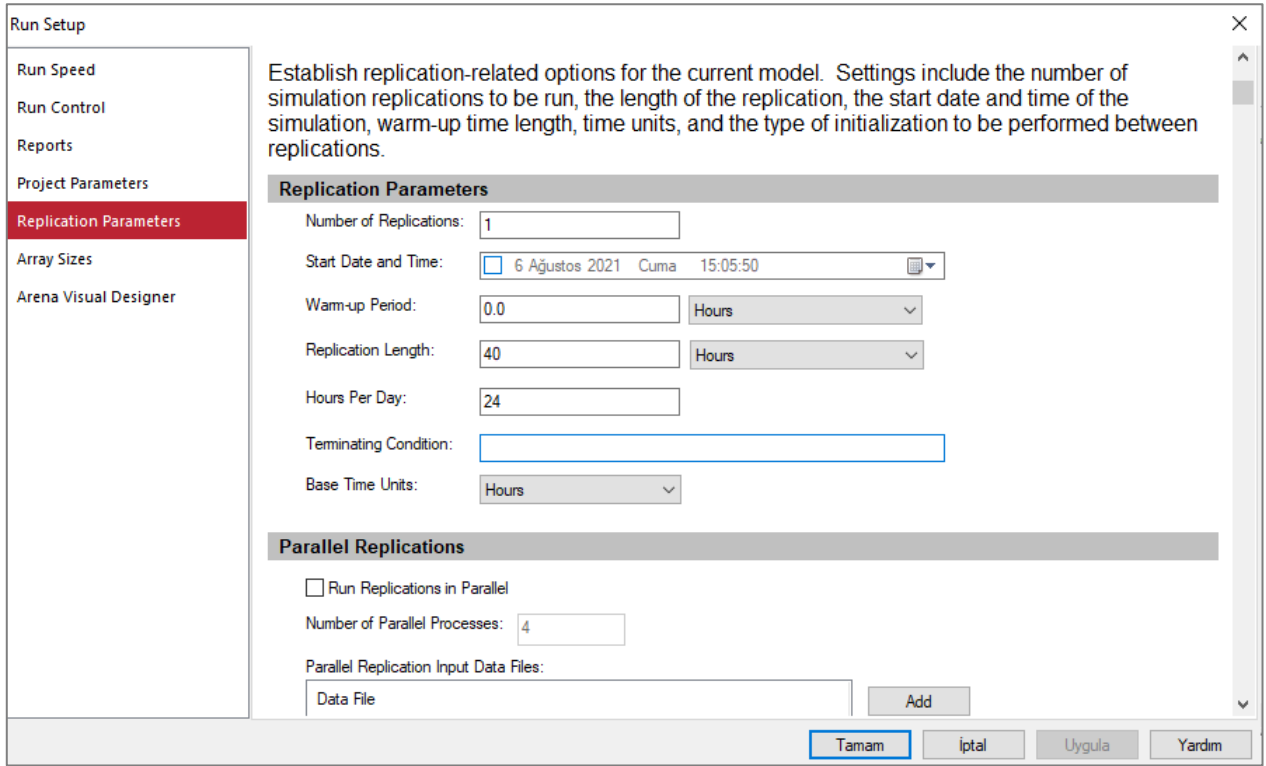
“Warm-up Period” modelin işleyişinin ve çıktılarının sağlıklı olarak gözlenmesi için durağan duruma (steady state) gelmesine gerek olduğu durumlarda kullanılır. Modele belirli bir ısınma süresi verilir ve kararlı duruma ne zaman ulaşacağı ve istediğimiz istatistiki değerleri hangi zamandan sonra hesaplamaya başlayacağımızı belirtiriz (varsayılan değeri 1’dir. şimdilik bu değer kalabilir),

“Replication Length” modelin her tekrarında ne kadar süre ile çalışacağını belirtir (belirtilecek bu süre gerçek saat değil, benzetim zamandır ve Kronometre gibi 0’dan başlayarak artar). Bu alanda, varsayılan olarak, “Infinite” ifadesi vardır, yani biz durdurmadığımız sürece çalışmaya devam edecek demektir. Buraya, istediğimiz benzetim süresini yazabiliriz. Bu model için çalıştırma süresini 40 saat olarak girelim.

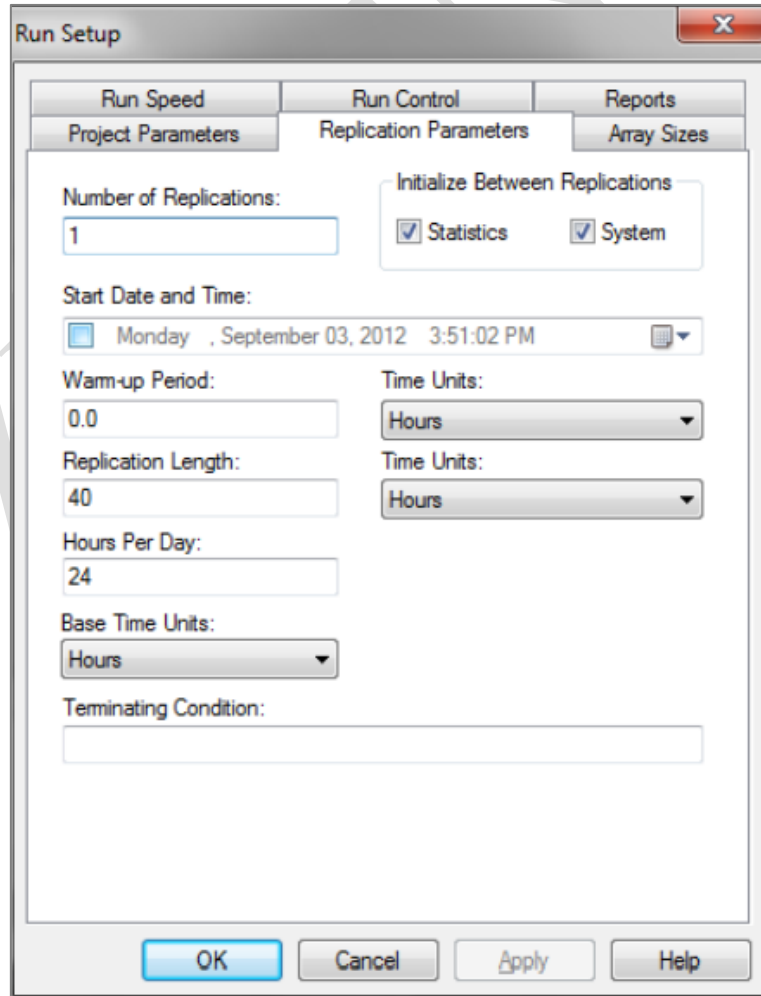
“Hours Per Day” alanı, sistemin bir günde kaç saat çalıştığını belirtir. Bu alan bazı benzetim modellerinde çok kullanışlıdır, bazen de bu alandaki değer olduğu gibi bırakılması daha uygun olabilir. Örneğin, fabrikada mesai bittiğinde sistem olduğu gibi bırakılıp ertesi gün kaldığı şekilde devreye alınıp işlere kaldığı yerden devam edilebilir ancak bu durum özellikle hizmet sektörü için mantıklı değildir. Bankalar akşam 5’de kapanır ve sabah 8’de açılır ama her sabah sistem sıfırdan başlar (kuyrukta bekleyenler tatbikî evlerine gideceklerdir). İşte, Hours Per Day alanına, 8 saat gibi belirli bir değer yazarsak, benzetim modeli çalıştırıldığında her 8 saatte bir sistemi boş olarak tekrar başlatır ama ortalama kuyruk uzunluğu, sistemde işlem gören toplam varlık sayısı gibi birikimli olarak hesaplanan istatistikler resetlenmez.

Sürekli çalışacak veya sürekli çalışmasa da kaldığı yerden devam edebilecek sistemlerde “Hours Per Day” için bir ayrıca bir değer girmeye gerek yoktur. Bizim bu örneğimiz için biz olduğu gibi 24 saat olarak bırakacağız.

Run Setup Penceresi örnek ekran görüntüsü Şekil 1-16.a. (ARENA yeni sürümü) ve Şekil 1-16.b.’de (ARENA eski sürümü) ayrı ayrı gösterilmiştir (Replication Length, yani tekrar süresi, 40 saat olarak tanımlanmış şekilde).

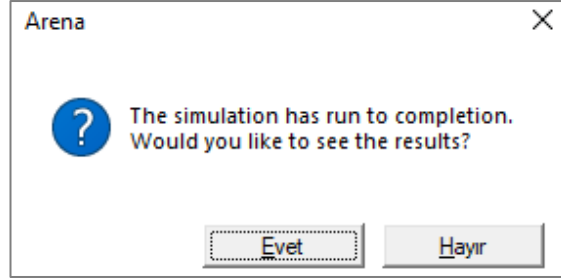


Şekil 1-16.a. ARENA yeni sürümünde Run Setup Penceresi



Şekil 1-16.b. ARENA eski sürümünde Run Setup Penceresi

Benzetimi çalıştırdığınızda, eğer hızlandırırsanız, daha kısa sürede benzetimin çalışması sona erer ve Şekil 1-17'deki mesaj görünür. Evet seçeneğine tıklandığında model sonuçları görüntülenir. Model Sonuçları birkaç sayfadan oluşmaktadır. Tüm sayfaları genel bir keşif yapıp inceleyin ve sayfalarda yer alan istatistikleri anlamlandırmaya çalışın.



Şekil 1-17 Modelin çalışması sona erdiğini gösteren mesaj penceresi

Bu raporun “Queue” seçeneğinin “Time” bölümünde, “Process 1.Queue” yani Process1’in kuyruğu satırı yer almaktadır. Bu satırın Average hanesinde varlıkların bu kuyrukta ortalama ne kadar süre bekledikleri bilgisi bulunmaktadır. Bizim örneğimizde; kuyrukta ortalama 0.069245 saat (~0.7 saat = 4.2 dk.) beklendiği görülmektedir.

Average ifadesi modelin çok defa çalıştırıldığında çıktı değişkeni değerlerinin ortalamasını için kullanılmaktadır. Bu modeli bir kez çalıştırdığımızdan, buradaki değer çıktı değerinin kendisidir (tek bir tekrarın ortalaması kendisidir). Dolayısıyla da “Half Width” değeri hesaplanmamıştır.

Modelimize tekrar dönüp, önce durduralım (model sona gelse ve rapor görüntülense de modelin tekrar baştan çalıştırılmadan önce, Run Control üzerinden ■ üzerine basılarak durdurulması gerekir). Bu sefer, Run Setup penceresinde Replication Parameters sekmesinde, Number of Replications değerini 10 yapalım (modeli birbirinden bağımsız ayrı ayrı 10 kere çalıştırmış olacağız) ve pencereyi OK düğmesi ile kapatarak modelimizi tekrar çalıştıralım. 10 kez ayrı ayrı çalıştırdıktan sonra sonuç raporuna baktığımızda Şekil 1-18'deki gibi bir sonuç göreceksiniz. Dikkat edilirse, modeli 1 defa çalıştırdığımızdan daha farklı sonuçlar görüyoruz.

## Queue

### Time

Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Process 1.Queue	0.08915286	0,02	0.04934169	0.1606	0.00	1.1536

Şekil 1-18 Sonuç Raporunda Kuyrukta Bekleme Süresi istatistikleri

Process1.Queue satırındaki;

Average: modelin her bir tekrarında elde edilen ortalama bekleme sürelerinin ortalamasını,

Half Width: bu 10 tekrarın %95 yarı güven aralığı uzunluğunu

Minimum Average: tekrarlar arasında en düşük ortalama bekleme süresi

Maximum Average: tekrarlar arasında en yüksek ortalama bekleme süresi

Minimum Value: tüm tekrarlar içerisinde tek bir varlık bazında en düşük bekleme süresini

Maximum Value: tüm tekrarlar içerisinde tek bir varlık bazında en düşük bekleme süresini ifade eder.

Replication Length (model çalıştırma süresini) 50 saat ve Number of Replications (Tekrar Sayısını) 20 yaptığımızda Şekil 1-19'da gösterilen rapor sonucu görülmektedir. Kuyrukta bekleme süresi genel ortalaması 0.1067 saat (~6 dakika) olarak görülmektedir (daha önceki sonuçtan daha yüksek). Ayrıca, bu tablodan model çalışması sırasında bir varlığın 2.5 saat kadar kuyrukta beklemiş olduğu (maksimum bekleme) görülmektedir. Aslında, daha fazla süreli ve tekrarlı model çalıştırmaları bazen uç değerler elde etmemize ve bunlarında ortalama istatistikleri belirli oranda değişmesine sebep verebilmektedir. Ne kadar süreli ve tekrarlı model çalıştırması yapılacağını belirleyen bir kural yoktur. Ancak, bilinmelidir ki rassal modeller rassal sonuçlar doğurur !!

Replications: 20      Time Units: Hours

Queue						
Time						
Waiting Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Process 1.Queue	0.1067	0,03	0.05108349	0.2843	0.00	2.5056
Other						
Number Waiting	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Process 1.Queue	0.8296	0,22	0.3445	2.0921	0.00	19.0000

Şekil 1-19 Sonuç Raporunda Kuyrukta Bekleme Süresi istatistikleri

Daha önce hiç kaydetmediyseniz, modelinizi “BasitKuyrukModeli” ismi ile kaydedin. ARENA programı, model dosyalarına .doe uzantısı vermektedir. Tekrar modelinizi açmak istediğinizde, kaydettiğiniz klasörde “BasitKuyrukModeli.doe” isimli dosyayı açmanız yeterli olacaktır.

### Bu Örnekte Ne Öğrendik ?

- Create Modülü,
- Dispose Modülü,
- Process Modülü,
- Entity Modülü,
- Resource Modülü,
- Expression Builder,
- Grafik ve Sayaç Ekleme,
- Benzetim Çalışma Süresinin Belirlenmesi,
- Benzetim Tekrar Sayısının Belirlenmesi,
- Model dosya formatı uzantısı
- Benzetim Sonu Raporunda Kuyrukta Bekleme Süresi İstatistikleri

## 2. ONARIM ATÖLYESİ MODELİ

Tersanede, gemilerin elektronik seyir cihazlarının (GPS, NAVTEX, vb.) onarımını yapan bir atölyeye günlük gelen arızalı cihaz sayısı Poisson Dağılımına (ort. 4 cihaz/ gün) uymaktadır.

Atölyeye gelen bir cihazın ilk önce bir teknisyen tarafından arıza tespiti yapılmaktadır. **Arıza Tespit** işlemi; cihazın incelenerek, arızanın neden kaynaklandığının tespit edilmesidir ve bu işlemin süresi Üçgen Dağılıma (30 dk., 60 dk., 120 dk.) uymaktadır.

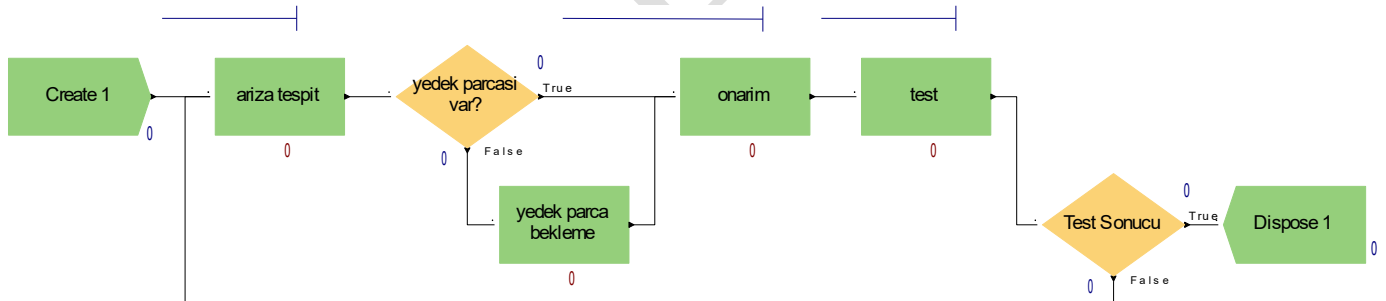
Arızası tespit edilen cihazların %20'sinin onarım için yedek parçaya ihtiyacı olmaktadır. Yedek parça, stokta tutulmamakta, üretici firmaya sipariş verilerek temin edilmektedir. Yedek parça ihtiyacı olan cihazlar sipariş edilen yedek parçalarının gelmesini beklemektedir. **Yedek Parça Bekleme** süresi rassal olarak değişmekte ve bu süre Üçgen Dağılıma (12 saat, 24 saat, 48 saat) uymaktadır.

Yedek parça ihtiyacı olmayan cihazlar Arıza Tespitinden hemen sonra, yedek parça bekleyen cihazlar da bekleme süresi sonunda onarıma girmektedir. **Onarım** işlemi, bir uzman teknisyen tarafından yapılmakta ve süresi Normal Dağılıma (Ort. 200 dk., std sapma 25 dk.) uymaktadır.

Onarım işleminden çıkan cihazlar, bir test bilgisayarında teste girerler. **Test** işlemi, cihazın tüm fonksiyonlarını test ederek son kontrolünü yapar. Test süresi sabit 30 dakikadır. Test sonucunda cihazların %90'ının testi geçtiği kabul edilecektir. Testi geçen cihazlar sistemden çıkmakta, testi geçemeyen cihazlar ise Arıza Tespiti işlemine geri gönderilmektedir (tüm işlemleri baştan tekrar etmek üzere).

Onarım atölyesi haftanın 7 günü, günde 8 saat çalışmaktadır (mesai süresi).

Yukarıda belirtilen açıklamalara göre, model alanında oluşturulan model görünümü Şekil 2-1'dedir.



Şekil 2-1

ARENA programında, benzetim modelini oluştururken, isterseniz;

Önce tüm süreçteki modülleri model alanına yerleştirip, bağlantılarını yapıp, sonra bu modüllerde gerekli tanımlamaları yapabilir, yada

Süreçteki modüllerin her birini model alanına yerleştirdikten sonra gerekli tanımlarını yapıp, adım adım ilerleyebilirsiniz,

Model alanına modüllerin yerleştirilmesi ve tanımlarının yapılması ile ilgili bir sıra sınırlaması yoktur.

Şekil 2-1'deki süreci model alanında oluşturduktan sonra aşağıda anlatıldığı gibi modüllerin detay tanımlarını yapabilirsiniz.

## Create Modülü

Gelen arızalı cihaz sayısının günde 4 adet ortalama ile poisson dağılımına uygun olduğu ve günlük çalışma süresi 8 saat olduğu belirtilmiştir.

Possion sürecine uygun olarak, ortalama 2 saatte bir arızalı cihaz geliyor ise gelişler arası süre, ortalaması 2 saat olan Üstel Dağılıma uygundur.

Type: Random (EXPO) Value: 2 Units : Hours

Entity Type alanına “arizali cihaz” yazılabilir.

Diğer alanlar olduğu gibi kalabilir.

Şekil 2-2

## Arıza Tespit İşlemi (Process Modülü)

Name : “ariza tespit” yazılır.

Action : “Seize Delay Release” seçilir.

Resources: Add.. düğmesine basılarak, açılan küçük pencerede yeni bir resource tanımlanır.

Yeni resource için açılan pencerede;

Name : “teknisyen” yazılır.

Units to Seize/Release : 1 değeri yazılı kalır.

Delay Type : Triangular seçilir

Minimum : 30

Value (Most Likely) : 60

Maximum : 120

Units : Minutes seçilir

Diğer alanlar olduğu gibi kalabilir.

Şekil 2-3

## Yedek Parça İhtiyacı Kontrolü (Decide Modülü)

Name : “yedek parçasi var?” yazılır.

Type : “2-way by Chance” seçilir.

Percent True (0-100) : 80 yazılır.

2-way by Chance, doğru/yanlış olarak iki sonucu olabilen bir karar, seçim için kullanılır. Bu karar, modelde yer alan belirli bir olasılık değerine göre verilir. “Percent True” alanına yazılan değer, bu karar modülünün “doğru” çıkışının olasılığıdır.

Şekil 2-4

### Yedek Parça Bekleme (Process Modülü)

Name : “yedek parca bekleme” yazılır.

Action : “Delay” seçilir.

Delay seçeneği, her hangi bir resource kullanımı olmadığı işlemler için kullanılır. Belirli bir süre alan ve işlemin tamamlanması için gerekli adımlardan birisi olan bekleme vb. durumlarda kullanılır. Her hangi bir resource tanımlaması yapılmaz.

Model açıklamasında belirtildiği şekilde;

Delay Type : Triangular seçilir

Minimum : 12

Value (Most Likely) : 24

Maximum : 48

Units : Hours seçilir

Diğer alanlar olduğu gibi kalabilir

The screenshot shows the 'Process' window for 'yedek parca bekleme'. The 'Name' field is 'yedek parca bekleme' and the 'Type' is 'Standard'. The 'Logic' section shows 'Action: Delay'. The 'Delay Type' is 'Triangular', 'Units' is 'Hours', and 'Allocation' is 'Non-Value Added'. The 'Minimum' is 12, 'Value:(Most Likely)' is 24, and 'Maximum' is 48. The 'Report Statistics' checkbox is checked. The 'Comment' field is empty. The 'OK', 'Cancel', and 'Help' buttons are at the bottom.

Şekil 2-5

### Onarım İşlemi (Process Modülü)

Name : “onarım” yazılır.

Action : “Seize Delay Release” seçilir.

Resources: Add.. düğmesine basılarak, açılan küçük pencerede yeni bir resource tanımlanır.

Yeni resource için açılan pencerede;

Name : “uzman teknisyen” yazılır.

Units to Seize/Release : 1 değeri yazılı kalır.

Delay Type : Normal seçilir

Value (Mean) : 200

Std Dev : 25

Units : Minutes seçilir

Diğer alanlar olduğu gibi kalabilir

The screenshot shows the 'Process' window for 'onarım'. The 'Name' field is 'onarım' and the 'Type' is 'Standard'. The 'Logic' section shows 'Action: Seize Delay Release' and 'Priority: Medium(2)'. The 'Resources' section shows a list with 'Resource, uzman teknisyen, 1' and '<End of list>'. The 'Add...', 'Edit...', and 'Delete' buttons are to the right. The 'Delay Type' is 'Normal', 'Units' is 'Minutes', and 'Allocation' is 'Value Added'. The 'Value:(Mean)' is 200 and 'Std Dev' is 25. The 'Report Statistics' checkbox is checked. The 'Comment' field is empty. The 'OK', 'Cancel', and 'Help' buttons are at the bottom.

Şekil 2-6

### Test İşlemi (Process Modülü)

Name : "test" yazılır.

Action : "Seize Delay Release" seçilir.

Resources: Add.. düğmesine basılarak, açılan küçük pencerede yeni bir resource tanımlanır.

Yeni resource için açılan pencerede;

Name : "test bilgisayarı" yazılır.

Units to Seize/Release : 1 değeri yazılı kalır.

Delay Type : Constant (yani sabit) seçilir

Value : 30 yazılır

Units : Minutes seçilir

Diğer alanlar olduğu gibi kalabilir

Şekil 2-7

### Test Sonucu Kontrol (Decide Modülü)

Name : "test sonucu" yazılır.

Type : "2-way by Chance" seçilir.

Percent True (0-100) : 90 yazılır.

Bir önceki decide modülünde yapılabilecek benzer şekilde yapılır. Modelde, test sonucunun %90 olumlu çıktığı kabul edilmektedir. Bu sebepten, modüle gelecek varlıkların %90 oranında true çıkışına yönlendirilmesi için bu değer yazılmaktadır.

Şekil 2-8

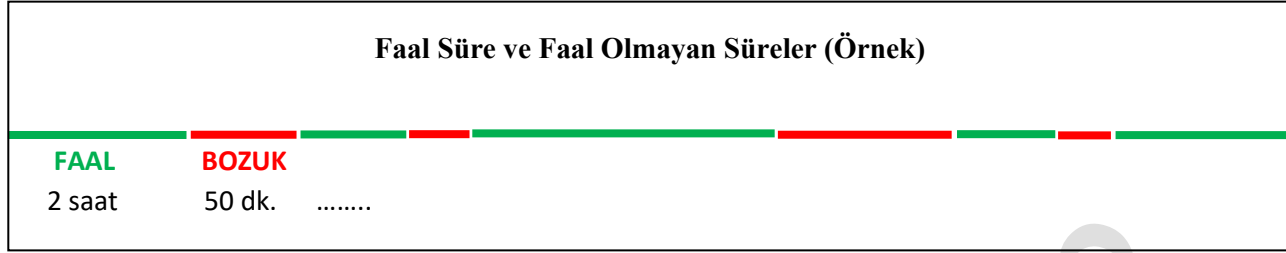
Dispose Modülü içerisinde ayrıca bir tanımlama yapılmasına gerek yoktur. Model, yukarıda belirtildiği şekilde kurulduğunda, hatasız çalışması gerekmektedir. Bir önceki modelde anlatıldığı şekilde, Run Setup üzerinden model tekrar uzunluğu, tekrar sayısı tanımlayabilirsiniz.

### Resource Bozulma (Failure) Durumlarının Modelde Tanımlanması

Şimdi, modelimize, test için kullanılan bilgisayarın bazen bozularak kullanım dışı kalması durumunu da dahil etmek istiyoruz. Buna göre, test bilgisayarı bir süre çalıştıktan sonra bozulabilmekte ve tekrar çalışır hale gelmesi için bir süre geçmesi gerekmektedir. Bu durumun modele dahil edilmesi için zaman biriminden iki değişkene ihtiyaç vardır ve bunlar benzetim modellerinde genellikle rassal değişkenlerdir : (1) Bozulmaya Kadar Geçen Süre (Faal Süre), (2) Tamir Süresi (Faal Olmayan Süre).



işlemlerde kullanılan kaynakların (Resource) bozulmalarına kadar geçen faal süreler için “Time-to-failure” veya “Up-Time” terimi ve tekrar çalışır hale gelmeleri için geçen süre için de “Time-to-repair” veya “Down-Time” terimi kullanılır (bakınız Şekil 2-9).



Şekil 2-9

Bu iki rassal değişkeni, birer olasılık fonksiyonu ile modelimize tanımlayacağız (tıpkı varlıkların sisteme gelişleri arasında geçen süreler için uyguladığımız şekilde). Böylece, bozulacağı öngörülen kaynak rastgele sürelerde bozulacak ve rastgele süreler boyunca faal olamayacak (yani işlem yapamayacak). Tatbikî, bu rastgele süreler belirteceğimiz olasılık dağılımına uygun olacaktır.

Bunun için, ARENA ekranının sol tarafında yer alan Project Bar alanından, tablo formatındaki modüllerden Resources adlı modülü seçin. Sayfanın alt kısmında, modelimizde yer alan 3 farklı kaynak ile ilgili bir tablo görünecektir (Şekil 2-10’da görüldüğü gibi).

	Name	Type	Capacity	Busy / Hour	Idle / Hour	Per Use	StateSet Name	Failures	Report Statistics	Comment
1	teknisyen	Fixed Capacity	1	0.0	0.0	0.0		0 rows	<input checked="" type="checkbox"/>	
2	uzman teknisyen	Fixed Capacity	1	0.0	0.0	0.0		0 rows	<input checked="" type="checkbox"/>	
3	test bilgisayarı	Fixed Capacity	1	0.0	0.0	0.0		1 rows	<input checked="" type="checkbox"/>	

Double-click here to add a new row.

Şekil 2-10

Bu tabloda, “test bilgisayarı” satırında “Failures” başlıklı sütundaki hücreye tıklayın. Böylece, yeni bir küçük pencere açılır. Yeni bir bozulma durumu tanımlamak için “Double-click here to Add a New Row” yazan yere çift tıklayın, böylece yeni bir satır açılır. Bu yeni satırdaki, “Failure Name” sütunundaki “Failure 1” ifadesi bu şekilde kalabilir, “Failure Type” sütununda 3 seçenek mevcuttur. Bunlar arasından, “Preempt” seçeneği seçilir (Resource bozulduğunda işlem durur). (Şekil 2-11)

Failures									
	Name	Type	Capacity	Busy / Hour	Idle / Hour	Per Use	StateSet Name	Failure Name	Failure Rule
1	teknisyen	Fixed Capacity	1	0.0	0.0	0.0		Failure 1	Wait
2	uzman teknisyen	Fixed Capacity	1	0.0	0.0	0.0			
3	test bilgisayarı	Fixed Capacity	1	0.0	0.0	0.0			

Double-click here to add a new row.

Double-click here to add a new row.

Şekil 2-11

Bundan sonra, “Failure 1” adlı bozulma durumunun detaylarını tanımlamamız gerekmektedir.

Modelde, test bilgisayarı için;

“Bozulmaya Kadar Geçen Sürelerinin” ortalaması 3 saat olan Üstel Dağılıma uyduğu,

“Tamir Sürelerinin” de Üçgen Dağılıma (60 dk., 120 dk., 180 dk.) uygun olduğunu farz edelim.

Bunun için, yine Project Bar alanında yer alan tablo formatındaki modüllerden “Failure” modülünü seçin. Biraz önce, “test bilgisayarı” satırında tanımladığımız için “Failure 1” adlı bir satır tabloda görünmektedir. Bu satırda;

“Type” sütunundaki hücrede, “Time” seçeneğini seçin (bozulmalar zamana bağlı ortaya çıkacaktır)

“Up Time” sütunundaki hücrede, “EXPO (mean)” seçeneğini seçin ve mean ifadesi yerine 3 yazın (yani EXPO (3) olacaktır)

“Up Time Units” sütunundaki hücrede, “hours” seçeneği seçilir

“Down Time” sütunundaki hücrede, “TRIA (mean, mode, max)” seçeneğini seçin ve mean ifadesi yerine 60, mode ifadesi yerine 120, max ifadesi yerine de 180 yazın (yani TRIA (60,120,180) olacaktır)

“Down Time Units” sütunundaki hücrede, “minutes” seçeneği seçilir.

Bu işlemlerden sonra, modeli tekrar çalıştırdığımızda, “test bilgisayarı” adlı resource, yukarıda belirtilen olasılık dağılımlarına uygun rassal sürelerde bozulacak ve bir süre bozuk kalacak, bu esnada test işlemi yapılamayacak, test işlemi duracaktır. Bu durum, test işleminin kuyrukta bekleme süreleri ortalamasını artıracaktır. (Bu artışı sonuç raporlarından görmeye çalışın)

### **Bu Örnekte Yeni Ne Öğrendik ?**

- Decide Modülünün “Two-Way by Chance” olarak kullanımı
- Process Modülünün, “Delay” mod kullanımı
- Kullanılan Kaynakların (Resource) bozulma durumlarının modele tanımlanması,

### 3. HASTANE ACİL SERVİSİ MODELİ

Bir Acil Servise iki şekilde hasta gelişi olmaktadır; yürüyerek veya ambulansla. Hastaların **gelişler arası geçen süreleri** geliş tipine göre farklı olacak şekilde rassal dağılıma uymaktadır:

Ambulansla gelen hastalar: Üstel Dağılım (ortalaması 30 dk.),

Yürüyerek gelen hastalar: Üstel Dağılım (ortalaması 5 dk.)

Her hasta için; gelişinden hemen sonra “Triyaj ve Kayıt” işlemi yapılmaktadır. Triyaj, hastaları bir ön incelemeye alıp rahatsızlıklarına göre aciliyet kategorisi belirlemektir ve uzman hemşire tarafından yapılır. Kayıt işlemi de triyaj sırasında yine aynı hemşire tarafından yapılır. Triyaj sonucunda hastalara, rahatsızlık durumlarına göre, Kırmızı (1), Sarı (2), Yeşil (3) olarak birer kategori verilir. Acil Serviste, “Triyaj ve Kayıt” işlemi yapan 2 adet uzman hemşire bulunmaktadır. “Triyaj ve Kayıt” işlem süresi Üçgen Dağılıma (5 dk., 8 dk., 12 dk.) uymaktadır.

Acil Servise gelen hastalardan;

- Ambulansla gelenlerin %70’inin Kırmızı (en acil), %30’unun Sarı (2.derece acil) kategori,
- Yürüyerek gelenlerin %1 Kırmızı, %19 Sarı, %80 Yeşil (en düşük) kategori olduğu kabul edilecek

Triyaj ve Kayıt işleminden sonra, hastalar boş yatak yok ise beklemektedir (bekleme kuyruğunda triyaj kodu önceliği dikkate alınır). Yatak kapasitesi 6’dır. Boş bir yatağa geçen hasta, doktor tarafından muayene edilir (muayene sırası için hastaların Triyaj kodu önceliği vardır). Doktor muayenesi sırasında, bir hemşire doktora yardımcı olmak üzere yanında bulunur. Bu işlem için vardiyada 3 doktor, 3 muayene hemşiresi vardır. Doktor muayene süresi rassal ve triyaj koduna göre farklı dağılımlara göre olmaktadır:

Kırmızı: Düzgün Dağ (10, 20) dk.,

Sarı: Üçgen Dağılım (4, 7, 12) dk.,

Yeşil: Üçgen Dağılım (3, 5, 8) dk.

Doktor muayenesi sonucunda; hastaların bazıları ilaç yazılarak hemen taburcu edilmekte (Acil Servisten çıkarılmakta), bazı hastalardan ise kan testi veya röntgen tetkikleri alınmaktadır.

Yeşil (3) kodlu hastaların %90’ının, Sarı (2) kodlu hastaların %70’inin test veya X-ray istenmeden doktor muayenesi sonunda hemen taburcu edilecekleri, Kırmızı (1) kodlu hastaların ise hepsinden mutlaka kan testi veya X-ray isteneceği kabul edilecektir. Hemen taburcu edilmeyen hastaların %85’inden kan testi alınacağı, %15’inin röntgeninin çekileceği kabul edilecektir (Bir hastadan ya kan testi yada X-ray istenir, ikisi birlikte yapılmamaktadır ve bu oranlar tüm triyaj kodları için ortaktır).

Kan testi yapmak için 1 hemşire bulunmakta, işlem sabit 3 dk. sürmekte ve test sonucu 15-20 dk. Düzgün Dağ. uygun bir sürede çıkmaktadır. Röntgen çekmek için, 1 Röntgen Teknisyeni bulunmakta, süresi 5-15 dk. arası değer alan Beta Dağılımına ( $\alpha=5$ ,  $\beta=2$ ) uymakta, sonuç hemen çıkmaktadır.

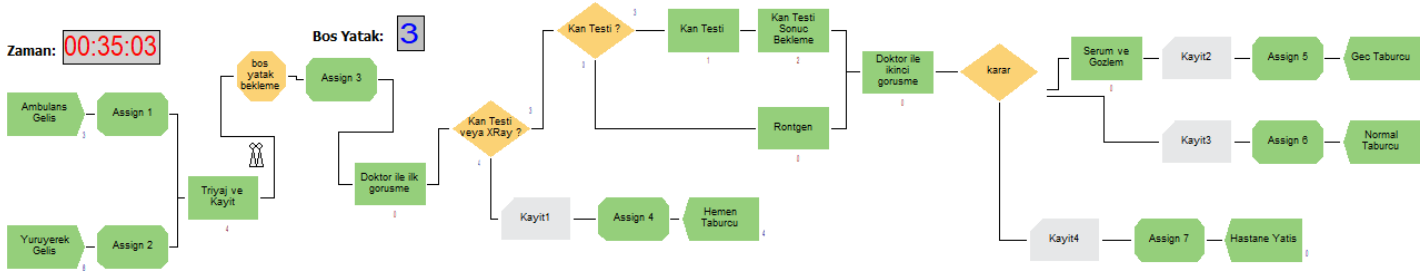
Kan Testi ve Röntgen çekimi sonrasında, hastalar doktor tarafından ikinci defa görülmekte (hemşire ile birlikte), bu görüşme süresi Üçgen Dağılıma (5,7,10) dk. uymaktadır. İkinci görüşmeden sonra;

Hastaların %60’ı hemen taburcu edilmekte,

%35’si bir süre daha acil serviste kaldıktan sonra (serum, gözlem vb.) geç taburcu edilmekte, bu süre Üçgen Dağılıma (30,60,120) dk. uymakta, her hangi bir kaynak kullanmamakta,

%5’i de hastaneye yatmaktadır.

Benzetim modelinin tamamlanmış hali aşağıda görülmektedir. Bağlantılar doğru olduğu sürece, modüllerin konumları, yerleşimi birebir aynı olmak zorunda değildir. Aşağıda, bu modelin nasıl oluşturulacağı adım adım tarif edilmektedir. Takip ederek oluşturmaya çalışın. Modelinizi AcilServis.doe olarak kaydedin ve çalışmalarınızı ara ara kaydetmeyi unutmayın !!!



Şekil 3-0 Benetim Modelinin Tamamlanmış Hali

### Hastaların Acil Servise Gelişleri (Create Modülleri)

Modeldeki varlıklar acil servise gelen hastalar olacaktır. Hastalar iki farklı kanaldan giriş yapmaktadır. Bu yüzden, modele iki ayrı create modülü ekleyeceğiz.

İlk Create modülünü sayfaya ekledikten sonra, çift tıklayarak açılan pencere üzerinden, "Name" alanına "Ambulans Geliş", "Entity Type" alanına "hasta" yazılır. "Time Between Arrivals" başlığı altındaki "Type" alanında görülen "Random(Expo)" olduğu gibi bırakılır, "Value" alanına 30 yazılır ve "Units" alanında "Minutes" seçilir. Diğer alanlar olduğu gibi bırakılır. (bakınız Şekil 3-1).



Şekil 3-1 Create Modülü Tanımlama Penceresi

İkinci Create modülü de sayfaya eklendikten sonra, benzer şekilde, "Name" alanına "Yürüyerek Geliş" yazılır. Biraz önceki Create modülünde "hasta" adlı varlık modele tanımlanmış olduğu için, "Entity Type" alanına tekrar elle "hasta" yazmaya gerek yoktur, seçebiliriz (istersek elle yazabiliriz). "Time Between Arrivals" başlığı altındaki "Type" alanı "Random (EXPO)" kalır, "Value" alanına 5 yazılır, "Units" alanında "Minutes" seçilir ve başka değişiklik yapmadan "OK" ile pencere kapatılır.

Şu anda modelimizde 2 adet create modülü var. Eğer bu modüllerden birisini mouse ile seçerseniz alt kısımdaki tablo alanında bu modüllere ait tanımlanmış olan bilgileri göreceksiniz (Şekil 3-2). İstenirse, ilk tanımlamadan sonra bu tablo üzerinden de değişiklik yapmak mümkündür.

	Name	Entity Type	Type	Value	Units	Entities per Arrival	Max Arrivals	First Creation	Comment
1	Ambulans Geliş	hasta	Random (Expo)	30	Minutes	1	Infinite	0.0	
2	Yürüyerek Geliş	hasta	Random (Expo)	5	Minutes	1	Infinite	0.0	

Şekil 3-2 Create Modüllerinin Eklenmesinden Sonra Tablo Görünümü

Modelimize iki farklı create modülü ekleyerek acil servise gelişleri iki ayrı kanaldan yapmış olduk, ancak gelen varlıkların geliş tipinin ne olduğunu (ambulansla mı, yürüyerek mi) varlıklara attribute olarak kaydetmemiz gerekir. Bunu aşağıda anlatılan “Assign” modülünde yapacağız.

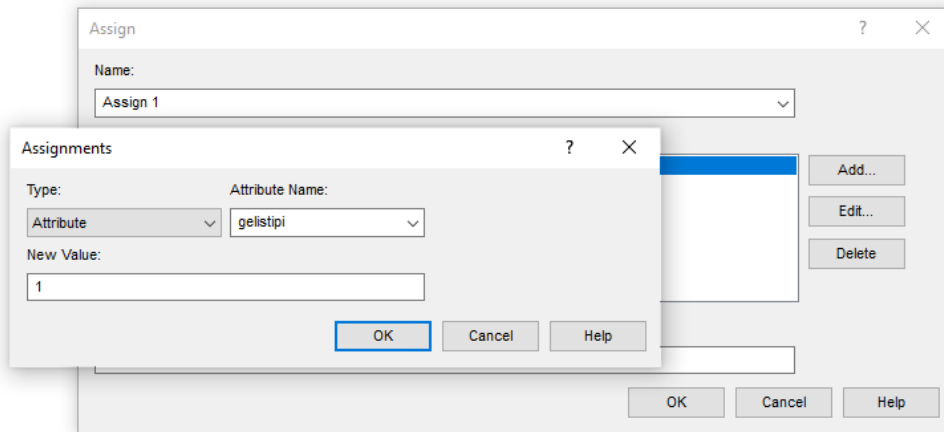
### Geliş Tipi, Geliş Zamanı ve Aciliyet Derecesinin Attribute Olarak Tanımlanması (Assign Modülleri)

Sisteme giren varlıkların özellikleri (attribute), “Assign Attribute” adı verilen modülde kaydedilir veya değiştirilir. ARENA programının eski sürümünde sadece tek tip Assign modülü bulunmaktaydı. Bu modül, hem varlıkların attribute’ları hem de modelde kullanılan diğer değişkenlerin değerlerinin kaydedilmesi için kullanılmaktaydı (Bazı değişkenler varlığa ait özellik olmayabilir, örn. ortamdaki sıcaklık). ARENA yeni sürümünde varlık attribute için “Assign Attribute” adlı ayrı bir modül bulunmakta, ancak “Assign” adlı modül de mevcut olup hem attribute hem de diğer değişkenler için kullanılabilir. Modelimizde, attribute tanımlamak için “Assign” modülünü kullanacağız.

Bunun için, Project Bar altındaki modüllerden iki adet “Assign” modülü model ekranına eklenir ve Create modüllerine ayrı ayrı bağlanır. Modeldeki tüm bağlantılar için; Şekil 3-0’da gösterilen şekilden faydalanabilirsiniz.

Modelde, Acil Servise giriş yapan her hastaya 3 ayrı attribute (özellik) tanımlayacağız: “gelistipi”, “geliszamani” ve “triyajkodu” (hem Assign 1 hem de Assign 2 modülünde).

Assign 1 modülüne çift tıklayarak tanımlama penceresi açılır. “Name” alanı bu şekilde kalabilir. “Assignments” alanı henüz boştur. “Add..” düğmesine basılarak yeni bir attribute tanımlaması yapılabilir. “Add..” düğmesine basıldığında açılan yeni penceredeki “Type” alanında “Attribute” seçeneği seçilir, “Attribute Name” alanına “gelistipi” yazılır, “New Value” alanına da Ambulansla Gelişleri ifade edecek 1 değeri yazılır (bu bizim belirlediğimiz bir değer, başka bir değer de verebilirdik, önemli olan varlığın bu özelliğini artık bu değer ile takip edecek olmamızdır). Yani “Ambulansla Geliş” adlı Create modülünden sisteme giriş yapmış olan bir varlığın (yani hastanın) “gelistipi” adlı bir özelliği var ve bu özelliğin değeri 1’dir. (bakınız Şekil 3-3)

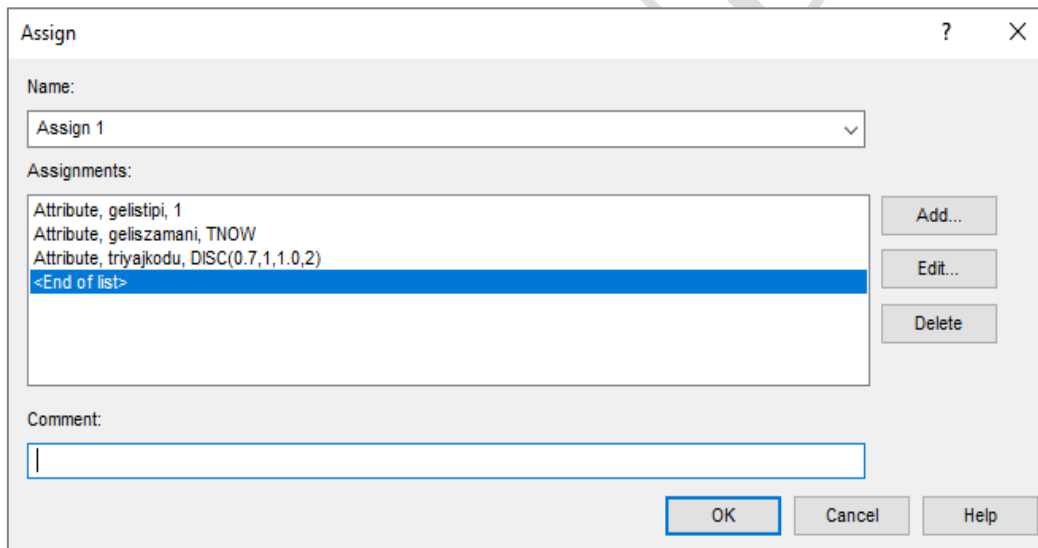


Şekil 3-3 Assign Modülünde Attribute Tanımlama

Assign penceresinde, “Add..” düğmesine tekrar basıldığında açılan yeni penceredeki “Type” alanında yine “Attribute” seçeneği seçilir, “Attribute Name” alanına “geliszamani” yazılır, “New Value” alanına da **TNOW** yazılır (bu ifade, ARENA programındaki hazır bir fonksiyondur, kullanıldığı andaki zaman değerini verir). Yani, her hastanın “geliszamani” diye ayrı bir attribute (özelligi) olacak ve bu attribute değeri her hastanın acil servise geldiği zamana eşit olacaktır (benzetim zamanı).

Assign penceresinde, “Add..” düğmesine tekrar basıldığında açılan penceredeki “Type” alanında yine “Attribute” seçeneği seçilir, “Attribute Name” alanına “triyajkodu” yazılır, “New Value” alanına **DISC(0.7, 1, 1.0, 2)** ifadesi yazılır. DISC ifadesi, kesikli bir olasılık dağılımını tanımlamak üzere kullanılan bir ARENA hazır fonksiyonudur. ARENA, bu olasılıklara göre rassal değişken üretir. Bu fonksiyondaki ilk parametre olasılık değeridir (0.7 değeri %70 olasılığı temsil eder). Sonrasında gelen 1 değeri ise triyaj kodu değeridir. Modelimizde triyaj kodu 1, “Kırmızı” rengi temsil edecektir. DISC fonksiyonu olasılık değerini birikimli olarak kabul eder. Dolayısıyla sonraki 1.0 değeri bir önceki %70 olasılıktan %100’e kadar olan olasılıkları, yani %30’luk bir dilimi temsil etmektedir. Bu olasılığa karşılık gelen triyaj kodu değeri 2 yani “Sarı”dır. Ambulans ile gelişlerde yeşil triyaj kodlu hasta olmadığından burada ona karşılık gelen bir olasılık tanımlanmamıştır.

Assign 1 modülü penceresi, yapılan tanımlamalardan sonra Şekil 3-4’de olduğu gibi görünecektir.



Şekil 3-4 Assign Modülünde Tanımlanan Attribute’lar

Özetle; Assign 1 modülü şunu yapacaktır: Acil Servise ambulansla gelmiş olan hastaların her birine 3 ayrı attribute tanımlar ve bunlara şu değerleri verir: gelistipi=1 (yani ambulans), geliszamani= Şimdiki Zaman, Triyajkodu : %70 olasılıkla 1 (kırmızı), %30 olasılıkla 2 (Sarı), %0 olasılıkla 3 (Yeşil).

Aynı işlemler, benzer şekilde, Assign 2 modülü için de yapılmalıdır. Assign 1’de olduğu gibi “Add..” düğmesine basarak, aynı ayrı ayrı attribute tanımları yapılır. Assign 2 modülünde;

“gelistipi” attribute için “Value” alanına 2 yazılmalı (yürüyerek gelişleri temsil etmekte),

“geliszamani” attribute için “Value” alanına aynı şekilde “TNOW” ifadesi yazılmalı,

“triyajkodu” attribute için “Value” alanına **DISC(0.01, 1, 0.2, 2, 1.0, 3)** ifadesi yazılmalıdır.

DISC(0.01, 1, 0.2, 2, 1.0, 3) ifadesi → %1 Kırmızı , %20 - %1 = %19 Sarı , %100 - %20 = %80 Yeşil  
(1: Kırmızı, 2: Sarı, 3:Yeşil)

## Triyaj ve Kayıt İşlemi (Process Modülü)

Acil Servise gelen hastaların Triyaj ve Kayıt işlemi için soldaki Project Bar alanından bir “Process” modülü seçilir ve model alanına eklenir, Assign 1 ve Assign 2 modülleri de buna bağlanır.

Eklenen process modülüne çift tıklanarak penceresi açılır. Bu pencerede;

“Name” alanına “Triyaj ve Kayıt” yazılır (process modülü, işlem yapan kişi değil, yapılan işlemidir).

“Action” alanında “Seize Delay Release” seçilir. Seize Delay Release seçildiğinde process modülü üzerinde kuyruğu temsilen bir hat çizgi görünür. Process modülünün “Seize Delay Release” olması şu demektir: Bu işleme gelen bir varlık, bu işlem için gerekli kaynak müsait değilse (meşgul) kuyruğa girer, müsait olduğunda yakalar (seize) ve belirli bir süre işlem için burada kalır (Delay), işlemi bittiğinde de otomatikman kaynağını serbest bırakır (release) ve gider. Benzetim modellerinde yaygın olarak kullanılan ve çoğu işleme uygun olan seçenek “Seize Delay Release” seçeneğidir.

Process Modülü penceresindeki “Action” alanındaki diğer seçenekleri de özetle açıklamak gerekirse;

“Delay” seçeneği, varlığın bu işlemde her hangi bir kaynak kullanımı olmayacağı, sadece belirli bir süre zaman geçirmesi gereken bir işlem olduğunu belirtir (örn. boyanın kurummasını bekleme),

“Seize Delay” seçeneği, bu işlem için belirli bir süre kaynak kullanılacağı ancak bu kaynağın kullanım süresi bittikten sonraki bir zaman ayrıca serbest bırakılacağını belirtir (örn. işlemi biten bir ürünün kaynağı serbest bırakıp bir sonraki işleme geçmek için başka bir işlemi bekliyorsa)

“Delay Release” seçeneği ise, varlığa daha önce bir kaynak atandığı ve bu işlemde sadece belirli bir süre işlem göreceği ve bu süre bitince de kaynağı serbest bırakacağı belirtilir.

Modelimizde, bu işlemde gerekli olan kaynakları ve bu kaynaklardan elimizde ne kadar olduğunu tanımlamamız gerekmektedir. Bunun için, Resources alanının yanında bulunan “Add..” düğmesine tıklanır. Açılan yeni pencerede, “Resource Name” alanına “uzman hemşire” yazılır, “Units to Seize/Release” alanındaki değer 1 olarak bırakılır. Bu, bir varlığın bu işlem için belirtilen kaynak tipinden kaç tanesine ihtiyaç duyduğudur. Bu alana 2 yazarsak, tek bir hastanın triyaj ve kayıt işlemi için aynı ana iki uzman hemşireye ihtiyaç duyulduğu anlamına gelir, ki bu da modelimize göre doğru değildir (Bu işlemde görevli olan uzman hemşire sayısını sonra ayrıca “Resources” modülünde tanımlayacağız). Şimdi, bu pencere “OK” düğmesi ile kapatılır.

Process penceresinde alt kısımda bulunan “Delay Type” alanında “Triangular”, “Units” alanında “Minutes” seçilir. Modelin açıklamasında belirtildiği gibi “Minimum” alanına 5, “Value (Most Likely)” alanına 8, “Maximum” alanına 12 yazılır. (bakınız Şekil 3-5)

Şimdi, Triyaj ve Kayıt işlemi için kullanılacak kaynak olan “uzman hemşire”den Acil Serviste kaç adet olduğu Resource modülünde belirtilmelidir. Bunun için, sol taraftaki Project Bar alanından, tablo görünümüne bir modül olan “Resources” modülüne tıklanır. Model alanının altında görünen “Resources” tablosunda tek bir satır olarak “uzman hemşire” kaynağı görülür.

Bu tabloda, “Capacity” sütununa, model açıklamasında belirtildiği gibi, 2 yazılır (yani iki uzman hemşire bu işlemde görevli, ayrı ayrı farklı hastalara işlem yapabiliyorlar). (bakınız Şekil 3-6)

Şekil 3-5 Triyaj ve Kayıt İşlemi Process Modülü Tanımlama Penceresi

	Name	Type	Capacity	Busy / Hour	Idle / Hour	Per Use	StateSet Name	Failures	Report Statistics	Comment
1 ▶	uzman hemsire	Fixed Capacity	2	0.0	0.0	0.0		0 rows	<input checked="" type="checkbox"/>	

Şekil 3-6 Resources Modülü Tablo Görünümü

### Hastaların Boş Yatak için Beklemesi ve Boş Yatak Sayısının Takibi (Hold ve Assign Modülü)

Trijaj ve kayıt işleminden sonra, hastalar Acil Servisteki yataklara alınır, ancak yatak kapasitesi 6'dır. Boş yatak yok ise beklemek zorunda kalırlar ve doktor muayenesine giremezler.

Gerçek durumda, özellikle aciliyeti yüksek hastalar için süreç tam olarak bu şekilde olmayabilir ancak bu modelde, basitlik açısından, tüm hasta tiplerinin boş yatak bekleyeceklerini farz edeceğiz ama modelde tarif edildiği gibi, boş yatak bekleme kuyruğunda triyaj koduna göre öncelik tanımlayacağız.

Boş yatak için beklemeyi, "Hold" modülü kullanarak yapacağız. Hold modülü de diğer modüller gibi soldaki Project Bar alanından bulunur (sekizgen bir şekil), seçilerek model alanına yerleştirilir ve "Trijaj ve Kayıt" modülüne bağlanır. Hold modülüne çift tıklayarak penceresini açtığımızda; "Name" alanına "Bos Yatak Bekleme" ifadesini yazalım. "Type" alanında 3 seçenek mevcuttur;

"Wait for Signal" : bu modüle gelen bir varlığın buradan çıkabilmesi için bir sinyal alması gerektiğini belirtir (sinyal, başka bir modül ile modele tanımlanan ve belirli uygun bir durumun sağlandığını haber veren bir mesaj gibi düşünülebilir (Bununla ilgili başka bir örnek yapılacaktır).

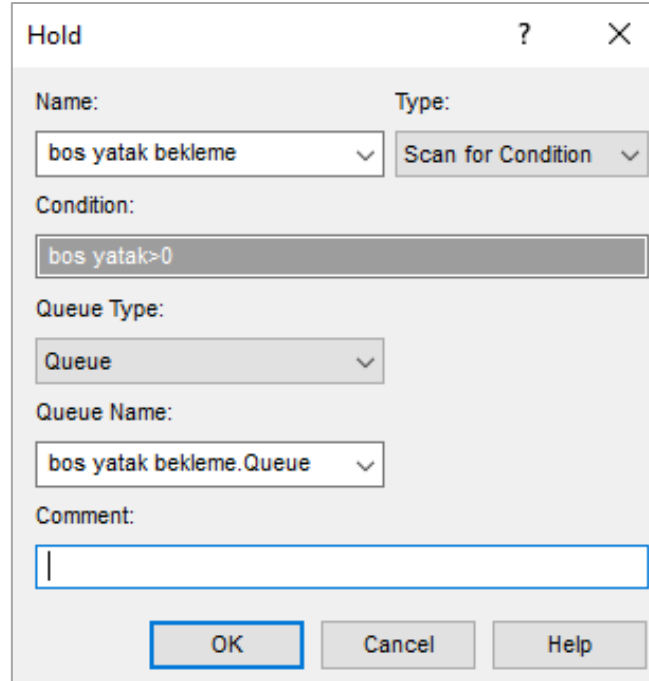
"Scan for Condition" : bu modüle gelen bir varlığın buradan çıkabilmesinin bizim belirteceğimiz bir koşula bağlı olduğu anlamına gelir.

"Infinite Hold" : bu modüle gelen varlığın buradan çıkmayacağı anlamına gelir (bizim modelimizde kullanılabilecek bir seçenek değildir ancak buna uygun kullanımı olan modeller vardır).

Bu modelde, Hold modülünden çıkış için "Scan for Condition" seçeneğini kullanacağız. Çıkış koşulu olarak boş yatak bulunma durumunu matematiksel olarak ifade edeceğiz.



Bunun için, benzetim modeli çalıştığı sürece, boş yatak sayısını dinamik olarak tutacak bir değişkene ihtiyacımız vardır. Bu değişken, her hangi bir varlığa ait bir attribute değil, bir sistem değişkenidir. Bu değişkenimizin adı “bos yatak” olsun. Şimdi, Hold penceresindeki “Condition” alanına **bos yatak > 0** yazalım (yani koşulumuz “bos yatak” değişkeni değerinin 0’dan büyük olmasıdır). (bakınız Şekil 3-7)



Şekil 3-7 Hold Modülü Penceresi

Bundan sonra, “OK” düğmesi ile pencereyi kapatalım.

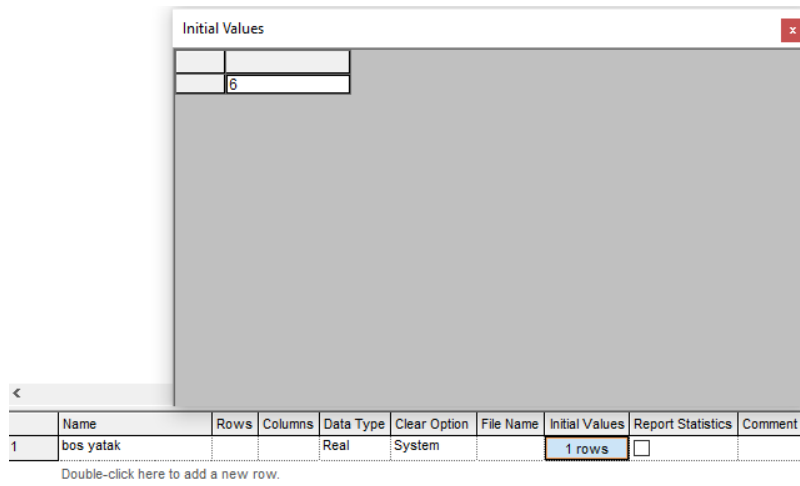
Böylece, Hold modülünden çıkış koşulunu belirtmiş olduk, ancak “bos yatak” adlı değişkeni modelimize tanıtmamız ve başlangıç değerini belirtmemiz gerekir (başlangıç değerimiz Acil Servis yatak kapasitesi, yani 6 olacaktır). Bunun için, sol taraftaki Project Bar alanında, tablo görünümlü modüller arasından “Variable” modülünü tıklayın. Model alanının altındaki Tablo alanında benzetim modelinde tanımlı sistem değişkenleri yer alır (hiç tanımlama yapılmadığı için şu anda boş).

Yeni bir değişken tanımlamak için “Variable” tablo görünümü alanında “Double-click here to Add a New Row” yazan yere çift tıklayın. Böylece, yeni bir satır açılır.

Bu satırın, “Name” sütununa tanımlamak istediğimiz değişkenin adı olan “bos yatak” yazılır (Hold penceresinde yazan değişkenin adı ile buradaki birebir aynı olmalıdır, buna dikkat edin).

Yeni açılan satırın, “Initial Values” sütununda “0 rows” yazmaktadır. Bu hücreye tıkladığınızda, yeni bir pencerede tek hücresi olan bir tablo ortaya çıkar. Bu tabloda “0.0” değeri yerine modelimizde belirtildiği gibi 6 yazarız ve pencereyi kapatırız. Böylece, boş yatak sayısını tutacak olan sistem değişkenimizi sisteme tanımlamış olduk. (bakınız Şekil 3-8)

Model alanına yerleştirdiğimiz Hold modülü üzerinde, Process modülünde olduğu gibi, bir kuyruk görülmektedir. Bunun anlamı; varlıkların Hold modülünden çıkmak için, belirtilen koşul sağlanmasını beklemek için de bir sıraya girecekleridir (şart sağlandıkça da, yani boş yatak bulunduğu zaman kuyruktan teker teker çıkacaklar).



Şekil 3-8 Variable Modülü Tablo Görünümü

Modelimizde, boş yatak bekleyen hastalar arasında triyaj koduna göre öncelik olduğu belirtilmiştir. Bu nedenle, bu öncelik durumunu da modelimizde tanımlamamız gerekir. Bunun için, sol taraftaki Project Bar alanında, tablo görünümlü modüller arasından, “Queue” (kuyruk) adlı tablo modülü seçilir. Model alanının altındaki Tablo alanında benzetim modelindeki tüm kuyruklar, her biri ayrı satır olarak, yer alır. Hold modülünü oluşturduğumuzda, otomatik olarak, bu modüle ait kuyruk da modele tanıtılmış oldu ve bu tabloya bir satır olarak eklendi. Name sütununda “Bos Yatak Bekleme.Queue” yazan kuyruk biraz önce eklediğimiz Hold modülünün kuyruğudur.

Bu satırın “Type” alanında varsayılan olarak “First In First Out” yazmaktadır. Bu hücreyi seçerseniz, diğer kuyruk tiplerini de görebilirsiniz. Bunlara arasından “Lowest Attribute Value” seçeneği seçilir ve “Attribute Name” sütununa tıklayarak, çıkan listenin en alt kısmındaki “triyajkodu” seçilir. Yani, bu kuyrukta önceliğini triyajkodu attribute değeri en düşük olana vermiş olduk (bakınız Şekil 3-9).

Triyaj ve Kayıt.Queue	First In First Out	Attribute 1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
bos yatak bekleme.Queue	Lowest Attribute Value	triyajkodu	<input type="checkbox"/>	<input checked="" type="checkbox"/>	

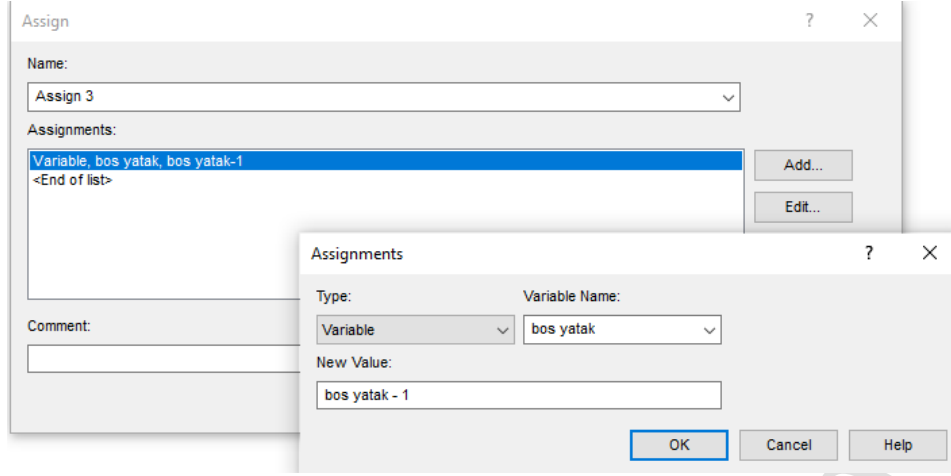
Double-click here to add a new row.

Şekil 3-9 Queue Modülü Tablo Görünümü

Bu tanımlarla, Hold modülü ile ilgili işlemleri bitirdik. Hold modülünden çıkan her varlık, boş yatak sayısını (yani “bos yatak” değişkenini) 1 azaltması gerekmektedir. Benzer şekilde, ama tam tersi olarak, Acil Servisten çıkan her bir hasta da boş yatak sayısını 1 artıracaktır. Bu değişiklik de örneğin ileriki bölümlerinde, sistemden Dispose modüllerinde çıkışlar öncesi ile ilgili kısımlarda, yapılacaktır.

Modelimizde, “bos yatak” değişkenin dinamik olarak değişmesi için, yeni bir Assign modülünü model alanımıza ekleyerek Hold modülüne bağlayalım. Yeni eklediğimiz Assign modülüne çift tıklayarak penceresini açtığımızda, “Add..” düğmesine basarak açılacak yeni penceredeki “Type” alanında zaten “Variable” seçilidir. “Variable Name” alanında “bos yatak” seçilir (değişkenimizi modele daha önceden tanımladığımız için otomatik olarak buradaki seçenekler arasında görülmektedir). “New Value” alanına **bos yatak -1** ifadesi yazılır (yani yeni değer bos yatak değişkeninin o anki değerinin 1 eksiği olacaktır). (bakınız Şekil 3-10)

Pencereleri “OK” düğmelerine basarak kapatalım.



Şekil 3-10 Assign Modülünde Değişken Değeri Güncelleme

### Doktor ile İlk Görüşme (Process Modülü)

Model ekranına yeni bir process modülü ekleyin ve son eklenen Assign modülüne bağlayın. Yeni eklediğiniz process modülüne çift tıklayın ve açılan Process penceresinde (Triyaj ve Kayıt adlı process modülünde yaptığımız işlemlere benzer şekilde);

“Name” alanına “Doktor ile ilk görüşme” yazın, “Action” alanında da “Seize Delay Release” seçin, Resources alanında “doktor” ve “muayene hemşiresi” adları ile iki yeni resource ekleyin (modelde, bir hasta muayenesinde 1 doktor ve 1 hemşirenin olduğu belirtilmişti, 2 farklı resource gerektiriyor).

Bu işlemin süresi hastanın triyaj koduna göre farklı rassal dağılımlara uymaktadır, bu yüzden doğrudan “Delay Type” alanında yer alan bir olasılık dağılımını seçmek yeterli olmayacaktır, koşullu bir ifade yazmamız gerekir. Bunun için; “Delay Type” alanında “Expression” seçeneğini seçin ve altındaki “Expressions” alanında boşluğa mouse ile sağ tuşa bastığınızda çıkan seçeneklerden “Build Expression” seçeneğini tıklayın. Böylece, yeni bir pencerede “Expression Builder” açılır (Şekil 3-11).

Expression Builder penceresinde, “Current Expression” alanına:

$(\text{triyaj kodu}==1) * \text{UNIF}(10,20) + (\text{triyaj kodu}==2) * \text{TRIA}(4,7,12) + (\text{triyaj kodu}==3) * \text{TRIA}(3,5,8)$  yazın.

Bu ifade, işleme gelene hastaya hastaya muayene süresi değeri üretir

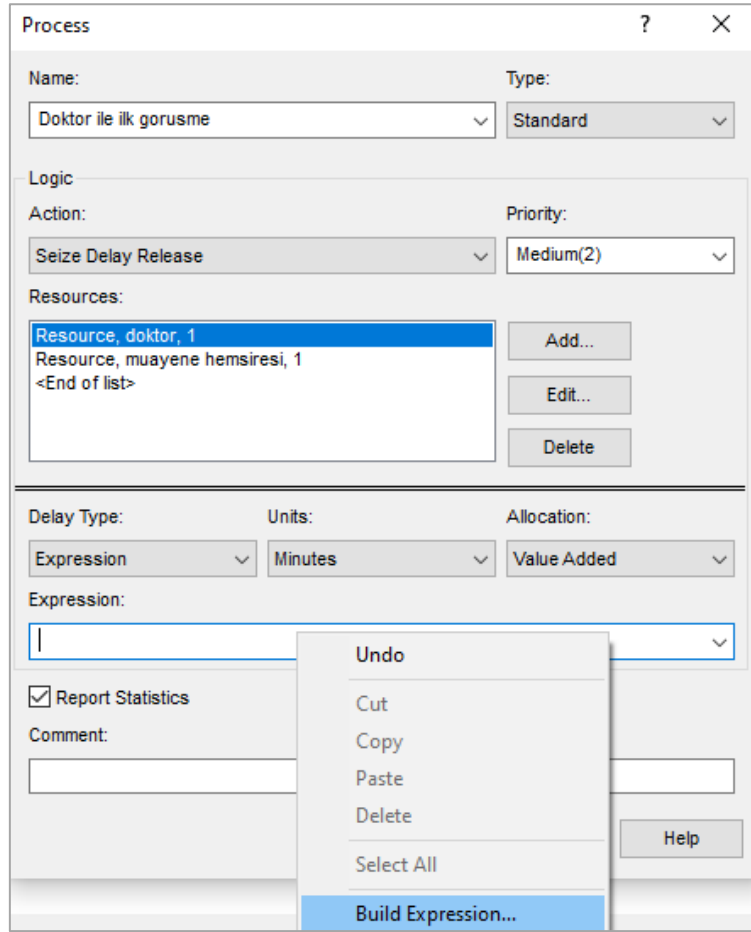
Bu ifadeyi yazarken, Expression Builder penceresinde yer alan öğeleri de kullanabilirsiniz, bu daha kolay ve doğru bir şekilde ifade yazmaya yardımcı olur. (bakınız Şekil 3-12)

Örneğin;  $(\text{triyaj kodu}==1)$  ifadesi, sonucu 0 veya 1 olan bir mantıksal sınamadır (hastanın triyaj kodu 1 ise doğru yani 1 değerini, triyaj kodu 1 değil ise yanlış yani 0 değeri sonucunu verir. Yani eğer hastanın triyaj kodu 1 ise tüm ifademiz şu şekilde değerler alacaktır;

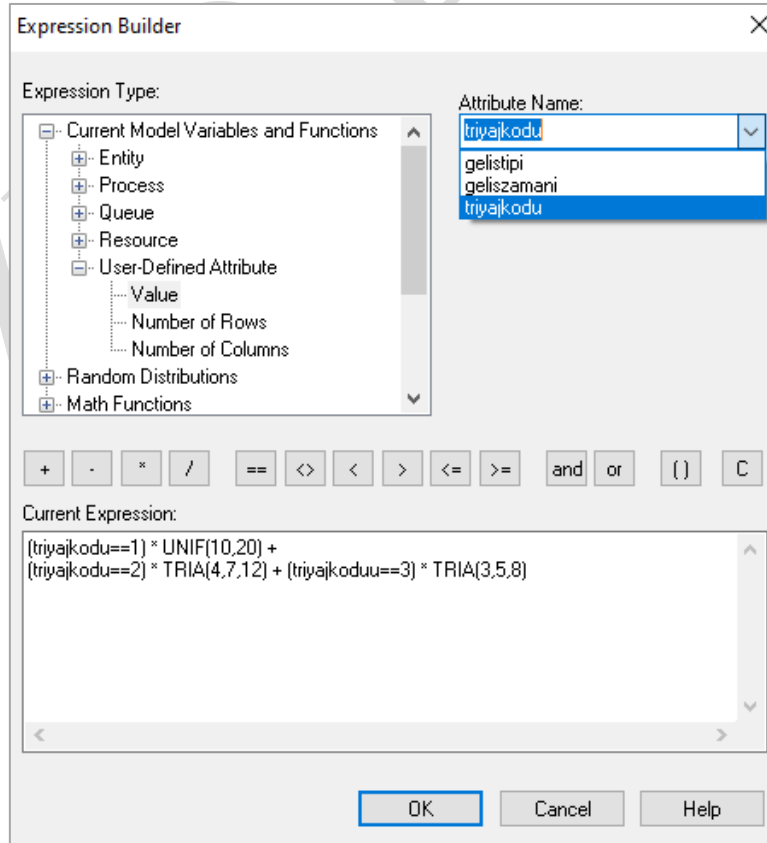
$$1 * \text{UNIF}(10,20) + 0 * \text{TRIA}(4,7,12) + 0 * \text{TRIA}(3,5,8) = \text{UNIF}(10,20)$$

ve istediğimiz gibi 10 ile 20 arasında düzgün dağılıma uyan bir rassal değişken üretecektir.

“Expression Builder”da bir ifade yazarken kullanacağınız olasılık dağılımlarını veya attribute’ları pencerenin üst kısmında bulunan listeden seçebilirsiniz. Bu şekilde, ifadelerinizi daha hızlı ve doğru bir şekilde yazabilirsiniz ve büyü kolaylık sağlar. Örneğin, yukarıdaki uzun ifadeyi yazarken, “triyaj kodu” adlı attribute bu listeden kademeli olarak açarak ve seçerek bulunmuş ve eklenmiştir (bakınız Şekil 3-12).



Şekil 3-11 Doktor ile Görüşme Process Penceresi



Şekil 3-12 Expression Builder Penceresi

ARENA programında hemen hemen her modelde Expression Builder kullanılmasını gerektirecek durumlar çıkmaktadır. Bu sebeple, Expression (yani ifade) yazmayı bilmek gerekir. Bu ifadeler, de mantıksal sınamalar, aritmetik operatörler içeren matematiksel ifadelerdir.

Örneğin; aşağıdaki ifadeler birer mantıksal sınamadır ve sonuçları ya 1 yada 0'dır.

$(\text{TriyajRenk} == 1 \mid \mid \text{TriyajRenk} == 2)$  : triyaj rengi 1 veya 2 mi ?

$(\text{TriyajRenk} == 1 \ \&\& \text{geliszamani} \leq 10)$  : triyaj rengi 1 ve geliş zamanı 10'dan az mı ?

$(\text{TriyajRenk} \neq 3 \ \&\& \text{UNIF}(0,1) \leq 0.2)$  : triyaj rengi 3'den farklı ve %20 olasılıkla

İfadeyi yazdıktan sonra, "Expression Builder" penceresi "OK" düğmesi ile kapatılır. Process penceresinde "Expressions" alanında oluşturduğunuz ifade görünecektir. Yani bu modül işlem sürelerini her hasta için sizin tanımladığınız bu ifadeye göre belirleyecektir.

Process penceresini de "OK" ile kapatın (öncesinde "Units" alanında "Minutes" seçmeyi unutmayın)

Ayrıca, Resources tablo modülünde (Trijaj ve Kayıt modülünde yaptığımıza benzer şekilde), "doktor" ve "muayene hemsiresi" satırlarındaki Capacity sütunu değerleri 3 yapılır, çünkü model açıklamasında bu işlemde 3 doktor ve 3 hemşirenin görevli bulunduğu belirtilmektedir.

### **Hastaların Kan Testi/ Röntgene Gönderilmesi veya Hemen Taburcu Edilmesi (Decide Modülü)**

Doktor ile ilk görüşmeden sonra, hastaların bir kısmı hemen taburcu edilmekte veya ileri tetkik için kan testi alınmakta veya röntgen çekilmektedir. ARENA programında, "Decide" modülü, varlıkların belirli olasılıklara veya koşullara göre farklı yollara ayrılarak yönlendirilmesi için kullanılmaktadır.

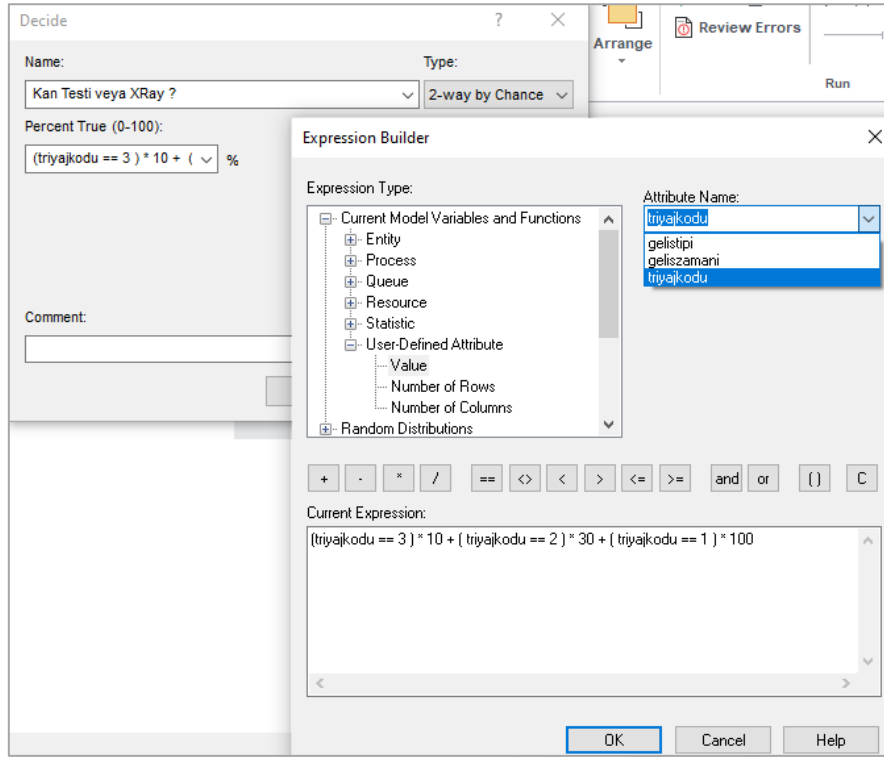
Project Bar alanından, Decide modülünü seçerek, model ekranına yerleştirin ve sol uç köşesinden "Doktor ile ilk Görüşme" process modülüne bağlayın. Decide modülünün içine girerek; "Name" alanına "Kan Testi veya Xray?" yazın ve "Type" alanında "'2-way by Chance" seçeneğini seçin.

Decide modülünde "2-way by Chance" seçildiğinde, modül şeklinin bir köşesinden "True", diğer bir köşesinden "False" çıkışı olur. "Percentage True" alanına yazılacak olan değer 100 üzerinden TRUE olasılığını ifade etmektedir (yani buraya 50 yazarsak, bu modüle gelen varlıkların %50'si modülün TRUE çıkışına giderler).

Modelimizde, Yeşil (3) kodlu hastaların %90'ının, Sarı (2) kodlu hastaların %70'inin kan testi veya X-ray istenmeden doktor muayenesi sonunda taburcu edilecekleri, Kırmızı (1) kodlu hastaların ise hepsinden mutlaka kan testi veya X-ray isteneceği belirtilmektedir. Yani, olasılık hastanın triyaj kodu attribute değerine göre değişmektedir, bu yüzden "Percentage True" alanına doğrudan sayısal tek bir olasılık değeri yazamıyoruz. Bunun için, bir önceki process modülünde yaptığımız gibi "Expression Builder" kullanarak, "Percentage True (0-100)" alanına:

$(\text{triyaj kodu} == 3) * 10 + (\text{triyaj kodu} == 2) * 30 + (\text{triyaj kodu} == 1) * 100$  ifadesini eklenir. (Şekil 3-13)

Bu ifade; triyaj kodu=3 ise 10, triyaj kodu=2 ise 30 ve triyaj kodu=1 ise 100 değeri verecektir. Yani, örneğin triyaj kodu 3 olan hastaların %10'u hemen taburcu edilmeyip teste (kan veya Xray) girecektir. (hangi teste gidecekleri daha sonra ekleyeceğimiz ayrı bir decide modülü üzerinden belirlenecektir, şimdi bu decide modülünde modelimiz sadece "hemen taburcu edilecek mi yoksa edilmeyip teste alınacak mı, diye karar veriyor).

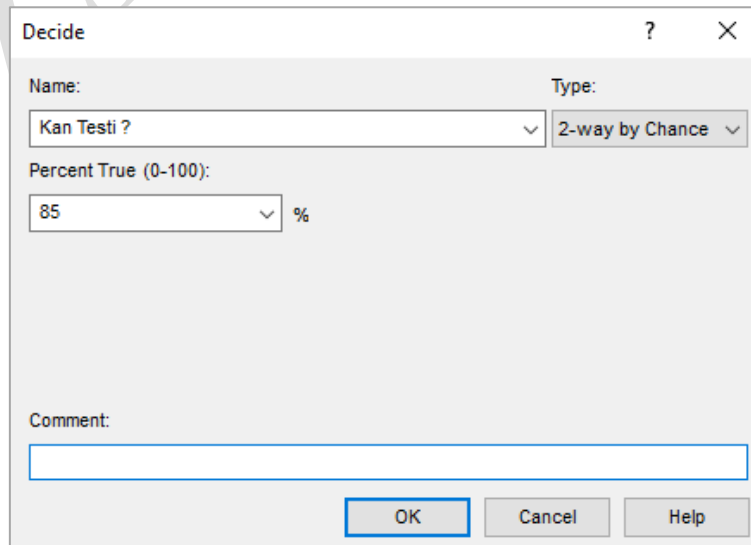


Şekil 3-13 Decide Modülü ve Expression Builder Penceresi

Hemen taburcu edilmeyen hastaları, Kan Testi veya Röntgene ayırmak için de ikinci bir Decide modülünü model ekranına yerleştirerek ilk Decide modülünün TRUE çıkışına bağlarız. Bu Decide modülünün adına da “Kan Testi?” yazalım (çünkü bu modülün True çıkışını kan testi işlemine yönlendireceğiz). Modülün “Type” alanında, önceki Decide modülü gibi, “2-way by Chance” seçelim.

Model açıklamasında, hemen taburcu olmayanların %85’inin kan testine, %15’inin de Röntgene gideceği belirtiliyor. Triyaj kodunun bu kararda her hangi etkisi yok. Bu sebeple, koşullu bir ifade yazmaya gerek olmadan, “Percentage True (0-100)” alanına 85 yazabiliriz (bakınız Şekil 3-14).

“Kan Testi veya XRay” adlı ilk decide modülünün FALSE çıkışını şimdilik boş bırakalım ve bir yere bağlamayalım. Buraya, sistemden çıkış ile ilgili olan Dispose modülünü bağlayacağız (ama sonra). Tüm dispose modüllerini, en sonda birlikte açıklayarak modele ekleyeceğiz.



Şekil 3-14 İkinci Eklenen Decide Modülünün Penceresi

## Kan Testi İşlemi ve Röntgen İşlemi (Process Modülleri)

“Kan Testi?” adlı ikinci Decide modülünün TRUE çıkışına, “Kan Testi” adını vereceğimiz bir process modülü, FALSE çıkışına da “Röntgen” adını vereceğimiz başka bir process modülü bağlayalım (yeni bir process modülünü model alanına yerleştirme, adını verme ve diğer bir modüle bağlamayı daha önceden biliyorsunuz).

Röntgen sonuçlarının çıkması için her hangi bir süre gerekmemekte (hemen çıkıyor ve bilgisayar ekranından bakılabiliyor) ancak kan testi sonuçlarının çıkması belirli bir süre almaktadır. Bunun için, “Kan Testi Sonuc Bekleme” adı vereceğimiz yeni bir process modülünü model alanına yerleştirerek “Kan Testi” modülü çıkışına bağlayalım. Yeni process modüllerini model alanına yerleştirip, bağlantılarını yapıp, adlarını verdikten sonra, önce “Kan Testi” modülünün tanımlarını yapacağız.

“Kan Testi” modülüne çift tıkla girerek, “Seize Delay Release” seçeneğini seçelim, “Add..” düğmesine basarak “test hemsiresi” adında yeni bir Resource tanımlayalım, “Delay Type” alanında “Constant” seçerek “Value” alanına 3 yazalım (“Units” alanı da “Minutes” olmalı) (bakınız Şekil 3-15).

The screenshot shows the 'Process' dialog box for a process named 'Kan Testi'. The 'Type' is set to 'Standard'. Under the 'Logic' section, the 'Action' is 'Seize Delay Release' and the 'Priority' is 'Medium(2)'. In the 'Resources' section, 'Resource, test hemsiresi, 1' is selected. Below this, there are buttons for 'Add...', 'Edit...', and 'Delete'. The 'Delay Type' is set to 'Constant', 'Units' is 'Minutes', and 'Allocation' is 'Value Added'. The 'Value' field contains the number '3'. The 'Report Statistics' checkbox is checked. There is a 'Comment' field at the bottom. At the very bottom are 'OK', 'Cancel', and 'Help' buttons.

Şekil 3-15 Kan Testi Process Modülü Penceresi

“OK” düğmesine basarak process penceresini kapattıktan sonra, daha önceki process modüllerinde yaptığımız gibi, sol taraftaki Project Bar alanında yer alan “Resources” tablo modülüne tıklamayarak, sayfanın alt kısmında görülen tabloda “test hemsiresi” olarak görünen satırdaki “Capacity” sütunu değerini 1 yapalım (kan testi için 1 hemşirenin görevli olduğu modelde belirtilmişti).

Sonra, “Kan Testi Sonuç Bekleme” adlı işleme girelim. Bu işlem, belirli bir süre almaktadır ancak her hangi bir görevli personele ihtiyaç duymamaktadır. Modelde, sonuçlar için gereken bir cihazdan da bahsedilmemiştir. Aslında, bir test cihazı kaynak olarak tanımlanabilir ve bu cihazın da bir kapasitesi olabilirdi ancak bu modelde test cihazı ve kapasitesi göz önüne alınmayacaktır.

Kan Testi Sonuç Bekleme, her hangi bir görevliye veya cihaza bağlı olmayacağından, bu modülün “Action” alanını “Delay” olarak seçebiliriz (kaynak olmadığından kuyruk da olmayacak), “Delay Type” alanında “Uniform” seçilir, değerleri 15 ve 20 olarak girilir (Units alanı Minutes!). (bakınız Şekil 3-16)

The screenshot shows the 'Process' dialog box for 'Kan Testi Sonuç Bekleme'. The 'Name' field is 'Kan Testi Sonuç Bekleme' and the 'Type' is 'Standard'. The 'Logic' section shows 'Action' as 'Delay'. The 'Delay Type' is 'Uniform', 'Units' is 'Minutes', and 'Allocation' is 'Non-Value Added'. The 'Minimum' is 15 and the 'Maximum' is 20. The 'Report Statistics' checkbox is checked. The 'Comment' field is empty. The 'OK', 'Cancel', and 'Help' buttons are at the bottom.

Şekil 3-16 Kan Testi Sonuç Bekleme Process Modülü Penceresi

Şimdi de, “Röntgen” adlı modüle girelim. Şekil 3-17’de görüldüğü gibi “Seize Delay Release” seçilir, “Röntgen Teknisyeni” adında yeni bir kaynak tanımlanır.

Röntgen süresinin 5-10 dk. arası Beta Dağılımına ( $\alpha=5$ ,  $\beta=2$ ) uyduğunu tanımlamak için Expression yazmamız gerekir (Delay Type seçenekleri arasında Beta Dağılımı vardır, ancak Beta Dağılımı 0-1 arası değer alır ve istenen aralığa, yani 5-15 aralığına, uydurmamız gerekmektedir).

Bunun için; Expressions alanına:  $BETA(5, 2) * (15 - 5) + 5$  ifadesi yazılır. (Units alanı: Minutes !)

The screenshot shows the 'Process' dialog box for 'Röntgen'. The 'Name' field is 'Röntgen' and the 'Type' is 'Standard'. The 'Logic' section shows 'Action' as 'Seize Delay Release' and 'Priority' as 'Medium(2)'. The 'Resources' section shows 'Resource, Röntgen Teknisyeni, 1' and '<End of list>'. The 'Delay Type' is 'Expression', 'Units' is 'Minutes', and 'Allocation' is 'Value Added'. The 'Expression' field contains the formula  $BETA(5, 2) * (15 - 5) + 5$ . The 'Report Statistics' checkbox is checked. The 'Comment' field is empty. The 'OK', 'Cancel', and 'Help' buttons are at the bottom.

Şekil 3-17 Röntgen Process Modülü Penceresi



### Doktor ile ikinci Görüşme (Process Modülü)

Test veya röntgen işlemini bitiren hastalar doktor tarafından ikinci bir defa görüşürler (Test sonuçları ile birlikte doktor tekrar muayene eder). Bu işlemde, ilk muayenede olduğu gibi, doktor ve hemşire birlikte bulunur. Resources alanına, “Add..” düğmesine basılarak açılarak yeni pencere üzerinden, daha önce tanımlanmış “doktor” ve “muayene hemşiresi” kaynakları seçilerek eklenir.

Görüldüğü gibi, bu işlem için yeni bir kaynak tanımlaması yapmadık, daha önceden (ilk muayene process modülünde), “doktor” ve “muayene hemşiresini” tanıtmıştık ve 3'er adet bulunduklarını Resources tablosunda belirtmiştik. Yani, “doktor ile birinci görüşme işlemi” ile “doktor ile ikinci görüşme işlemi” aynı kişileri kullanıyor. “Delay Type”, “Value” ve “Units” alanlarını modelde belirtildiği şekilde tanımlayalım ve kapatalım. (bakınız Şekli 2-18).

Şekil 3-18 Doktor ile İkinci Görüşme Process Modülü Penceresi

### Doktorun Teşhisi ve Kararı (Decide Modülü)

Test veya röntgen sonuçları ile birlikte hastayı tekrar gören doktor bir karar verir. Model bize burada 3 farklı karar olabileceğini söylüyor (1-Taburcu Etmek, 2-Serum ve Gözlem için bir süre daha kaldıktan sonra taburcu etmek, 3-Hastaneye Yatış).

Yeni bir Decide modülünü model alanına yerleştirerek “Doktor ile ikinci görüşme” process modülüne bağlayın, bu yeni modüle “Karar” adını verin, “Type” alanında “N-way by Chance” seçin (olasılıklara göre 3 farklı çıkış olacağı için).

“Percentages” alanında, “Add..” düğmesine basarak olasılık değerleri belirtilir. İlk olarak, 35 değerini ekleyelim (serum veya gözlem maksatlı olarak bir süre daha acil serviste kalacak olan hastalar için), daha sonra da 60 değerini ekleyelim (taburcu olacak hastalar için). Geriye kalan ve %5’lik olasılığa denk gelen “Hastaneye Yatış” çıkışı için ayrıca bir olasılık tanımlamaya gerek yoktur, diğer olasılıklar dışında kalan hastalar Decide modülünün FALSE çıkışına %5 olasılıkla gidecektir (bakınız Şekil 3-19).

Şekil 3-19 Karar adlı Decide modülü Penceresi

### Serum ve Gözlem için Acil Serviste Kalma (Process Modülü)

Serum veya gözlem için Acil Serviste kalmasına karar verilen hastaların gireceği işlem için yeni bir process modülünü model alanına yerleştirin. Bu modülü, “Karar” adını verdiğimiz Decide modülünün “35” yazan çıkışına bağlayalım (Decide modülünün çıkışlarında olasılık değerleri yazmaktadır, görüntüyü yakınlaştırmak daha net görebilirsiniz).

Yeni eklenen process modülüne çift tıklayarak penceresini açtığınızda, “Name” alanına “Serum ve Gözlem” yazılır. Bu işlem herhangi bir kaynak istememektedir, hasta bulunduğu yatakta serum almakta, beklemektedir. Bu nedenle, daha önce “Kan Testi Sonuc Bekleme” adlı modülde olduğu gibi “Action” alanından “Delay” seçilir.

Pencerenin alt kısmında yer alan “Delay Type” alanından “Triangular” (yani üçgen dağılım) seçilir, Minimum, Value (Most Likely) ve Maximum alanları da sırasıyla 30, 60, 120 olarak doldurulur. “Units” alanında “Minutes” seçmeyi unutmayın ! (bakınız Şekil 3-20)

Şekil 3-20 Karar adlı Decide modülü Penceresi

Process pencerelerine yer alan “Allocation” alanında řu zamana kadar bir deęiřiklik yapmadık. Varsayılan deęer olarak görünen “Value Added” ifadesi, varlıęa bu iřlemde sistemde bulunma amacına uygun olarak bir deęer katıldıęı anlamına gelir. Bazı iřlemler, özellikle de sadece “Delay” olarak belirlenen iřlemler, bazı modellerde deęer katan iřlem olmayabilir. Bu durumlarda, dięer seeneklerden birisi olan “Non-Value Added” seilebilir veya bu iřlemin bekleme istatistikleri arasında görölmesi isteniyorsa “Wait” seeneęi seilebilir veya sistem ierisinde özellikle bir yerden bařka bir yere aktarma, tařıma olması gibi durum varsa ve bunu istatistiklerde takip etmek istedięimizde de “Transfer” seeneęi seilebilir (bu husus modelin alıřmasına engel teřkil etmez).

### **Hastaların Acil Servisten ıkıřları (Record, Assign ve Dispose Modölleri)**

Hastalar, Acil Servisten 4 farklı ařamada ıkıř yapabilmektedir:

- **Hemen Taburcu** : Doktor ile ilk Görüřme sonrasında, Kan Testi veya Röntgen testine girmesine gerek duyulmayan ve hemen taburcu edilen hastaların Acil Servisten ıkıřı,
- **Normal Taburcu** : Doktor ile ikinci Görüřme sonrasında bekletilmeden taburcu edilen hastalar,
- **Ge Taburcu** : Doktor ile ikinci Görüřme sonrasında serum verilmesi ve gözlem altında tutulması iin bir süre daha acil serviste kaldıktan sonra taburcu edilen hastalar,
- **Hastane Yatıř** : Doktor ile ikinci Görüřme sonrasında, hastaneye yatıřına karar verilen hastalar (Acil Servis sistemini modelledięimiz iin bu da Acil Servisten bir ıkıř olarak kabul edilmektedir)

Modeldeki tüm olası ıkıřlar eninde sonunda bir Dispose modölüne baęlanmalıdır. ıkıřı aıkta kalmıř bir modöl, aık baęlantı, modelin alıřtırılmasını engeller, zaten program HATA verir.

řimdi, yukarıda belirtilen 4 farklı ıkıřı modelimize Dispose modölleri ile tanımlayacaęız. Ayrıca, her bir ıkıř öncesinde;

- Hastaların Acil Serviste kalıř süresi ve ıkan hasta sayısını kaydedeceęimiz bir Record modölü ve
- Boř yatak sayısını 1 artıracadıımız bir Assign modölü yer alacaktır.

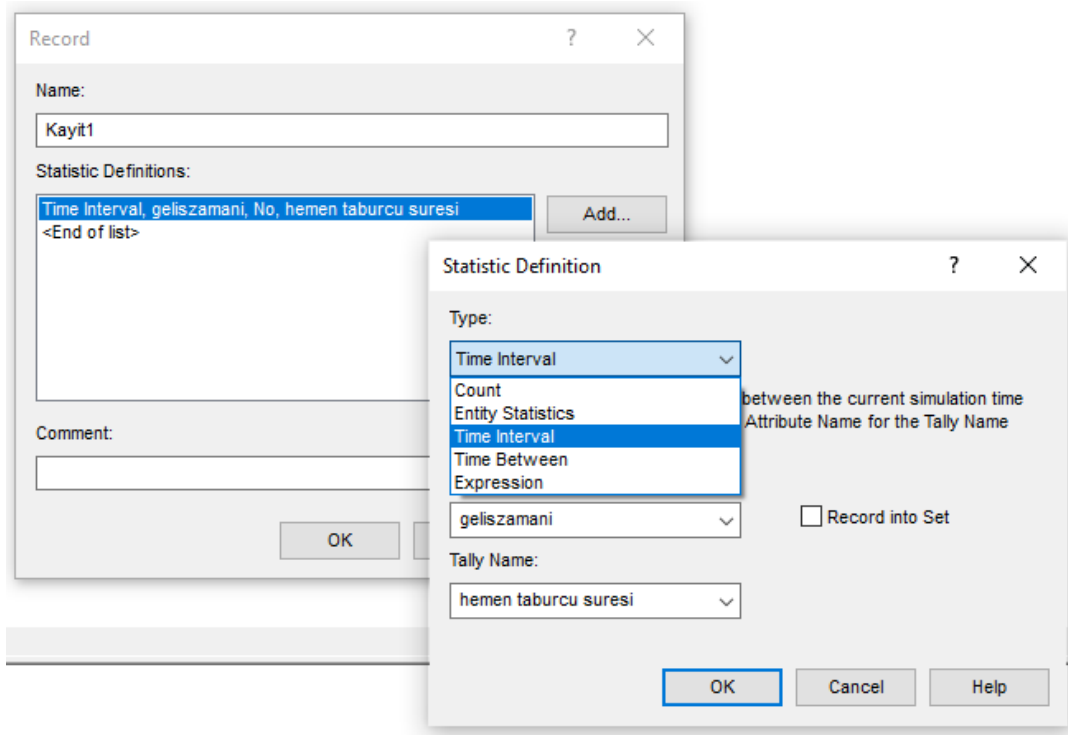
#### **Kayıt Modölü (Sistemde Kalıř Süresini ve ıkan Hasta Sayısını Kayıt Altına Almak iin) :**

Record modölü, benzetimin ıktı raporunda görmek isteyeceęimiz bazı deęiřkenlerin deęerlerini veya zaman aralıklarını hesaplayarak sisteme kaydettięimiz bir modöldür.

Sol taraftaki Project Bar alanından bir “Record” modölünü ekrana ekleyelim. Bu modüle ift tıklayarak penceresini atıęımızda, “Name” alanına “Kayıt1” yazalım, “Statistic Definitions” alanına yeni bir kayıt eklemek iin “Add..” düęmesine basarak yeni bir pencere aarız.

Bu pencerede, “Type” alanındaki seeneklerden “Time Interval” (yani zaman aralıęı) seeriz. Hatırlarsanız, hastalar modeli ilk giriř yaptıktan sonra, bir Assign modölü ierisinde, her birinin sisteme geliř zamanlarını birer attribute olarak tanımlamıřtık. řimdi hastanın bu attribute deęeri (yani geliř zamanı) ile ıkıř zamanı arasında geen süreyi kaydetmek istiyoruz. Bunun iin “Attribute Name” alanından “geliszamanı” seilir. “Tally Name” alanına da “hemen taburcu suresi” yazalım (raporda takip edebilmek iin).

Pencereleri “OK” düęmeleri ile kapattıktan sonra modölü “Kan Testi veya XRay” Decide modölünün FALSE ıkıřına baęlayalım (daha önce bu Decide modölünün TRUE ıkıřını baęlamıřtık ama FALSE ıkıřı bořta kalmıřtı). Record modölünde yapılan iřlemler řekil 3-21’de görölmektedir.

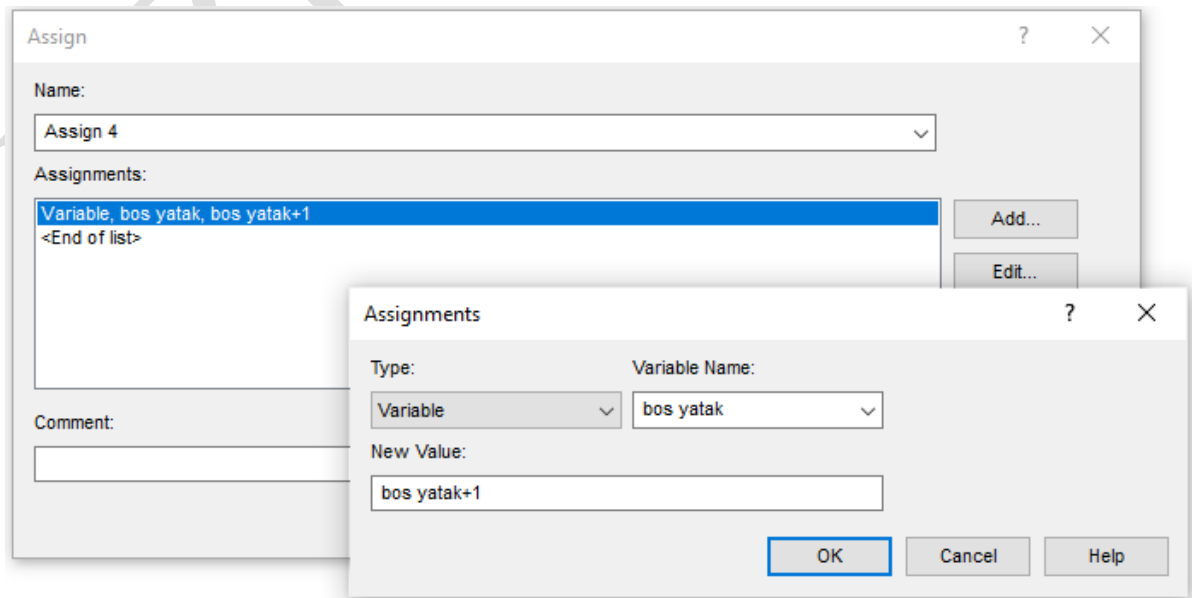


Şekil 3-21 Record modülü Penceresi

#### Assign Modülü (Boş Yatak Sayısını Artırmak İçin) :

Sol taraftaki Project Bar alanından bir “Assign” modülünü ekrana ekleyelim. Bu modüle çift tıklayarak penceresini açtığımızda, “Add..” düğmesine basarak yeni bir pencere açarız. Burada, Hold modülü sonrasındaki Assign modülünde yaptığımızı tersini yapacağız. “Variable” alanından “bos yatak” seçeneğini seçeriz ve “New Value” alanına da **bos yatak + 1** ifadesini yazarız (hasta çıkış yaptığı için yatağı boşalır, boş yatak sayısı 1 artar). (bakınız Şekil 3-22)

Pencereleri “OK” düğmeleri ile kapattıktan sonra modülü, “Kayıt1” adını verdiğimiz Record modülüne bağlarız (bağlamayı tanımlama öncesinde de yapabilirsiniz).

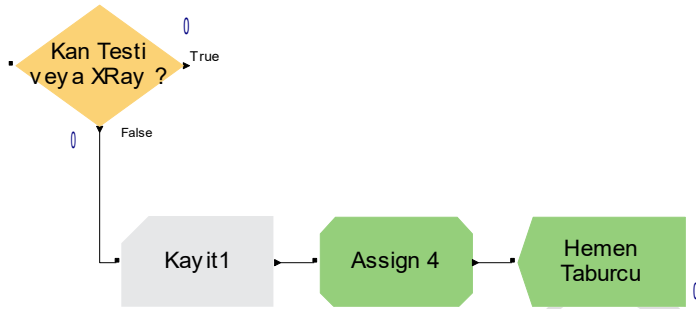


Şekil 3-22 Record modülü Penceresi

### Dispose Modülü (Acil Servisten Çıkış) :

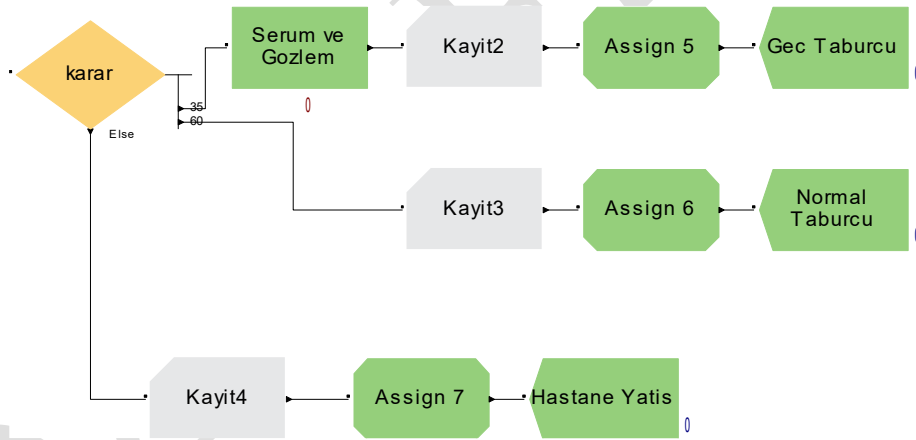
Sol taraftaki Project Bar alanından bir “Dispose” modülünü ekrana ekleyelim. Bu modüle çift tıklayarak penceresini açtığımızda, “Name” alanına “Hemen Taburcu” yazıp, “OK” düğmesi ile kapatalım. Dispose modülünü biraz önce eklediğimiz Assign modülüne bağlayalım.

Yukarıda belirtilen ekleme, tanımlama ve bağlama işlemlerinden sonra Acil Servisten ilk çıkıştaki bağlantı görünümü Şekil 3-23’de görüldüğü gibi olacaktır.



Şekil 3-23 Hemen Taburcu Çıkışında modüllerin bağlantısı

Diğer çıkışlar için de benzer şekilde, her çıkışa ayrı ayrı Record, Assign ve Dispose bağlantıları yapılmalıdır (Şekil 3-24’de görüldüğü gibi). Record modüllerine; Kayıt2, Kayıt3, Kayıt4 isimleri, Dispose modüllerine “Gec Taburcu” , “Normal Taburcu”, “Hastane Yatis” isimleri verin.



Şekil 3-24 Geç Taburcu, Normal Taburcu ve Hastane Yatış çıkışlarının bağlantıları

Record ve Assign modüllerinin isimleri hariç pencerelerinde yapılan tanımlamaların hepsi aynıdır.

Not : Kısa yol olarak; bir modül seçili iken Ctrl ve D tuşlarına birlikte aynı anda basarsanız, seçili modülün tüm içeriği ile kopyasını elde ederseniz (İsmi değiştirilmeyi unutmayın. Aynı isimli modül olursa ARENA çalıştırıldığında hata verir, çalışmaz)

Bu işlemlerden sonra model tamamlanır. Modelinizi kaydedip, RUN control üzerinden modeli çalıştırmayı deneyin. Hata veriyor ise hata açıklamasını okuyarak hatanın nedenini öğrenip düzeltin. Eğer, modelinize varlıklar giriyor ve sistemde dolaşıyor ise modeliniz teknik olarak çalışıyor demektir.

Not : Zaman ile ilgili birimleri “minutes” olarak tanımladığınızdan emin olun (modelde tüm birimler dakika olarak verilmiştir).

## Boş Yatak Sayısı Sayacı ve Benzetim Saatinin Canlı Olarak Ekranda Görüntülenmesi

Daha önceki basit kuyruk modeli örneğinde, sayaç eklemeyi görmüştük. Bu modelde de, boş yatak sayısını canlı olarak görüntüleyebiliriz. Bunun için, “Animate” menüsünden “Variable” üzerine tıklanır ve bir pencere açılır. Bu pencerede; “Expression” alanına, değişkenimizin adı olan “bos yatak” yazılır. “Format” alanında “ \* ” seçeneği seçilir (yatak sayısını ondalıklı olarak görmeye ihtiyacımız olmadığı için ondalıksız gösterme seçeneği olan \* seçilir). Pencereyi “OK” düğmesi ile kapattığımızda, sayacımızı yerleştireceğimiz yeri seçeriz ve boyutlarını ayarlayarak model ekranına ekleriz. İstenirse; bu sayacın hemen yanına, metin olarak “Boş Yatak Sayısı” yazısı eklenebilir.

Model ekranına benzetim saati eklemesi de benzer şekilde, yine “Animate” menüsü altında yapılmaktadır. Burada, “Clock” seçeneğine tıklanarak, açılan pencereden, istenirse düzenlemeler yapılarak, aynı şekilde ekrana yerleştirilir.

Sayaç, saat gibi nesnelerin boyutları (eni, boyu) ve konumları daha sonra da değiştirilebilmektedir.

## Modelin Çalışma Kurulum (Run Setup) Parametrelerinin Girilmesi ve Çalıştırılması

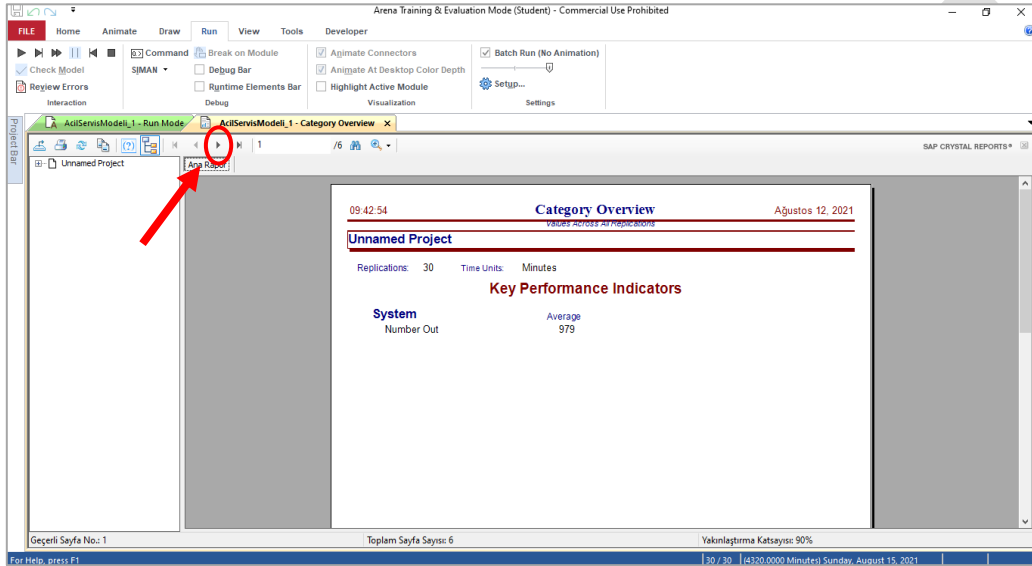
Run Setup Penceresi, Replication Parameters sekmesinde, Şekil 3-25’deki gibi, düzenleme yapılır.

- Number of Replications (Tekrar Sayısı) : 30
- Warm-up Period (Isınma Periyodu) : 1 saat
- Replication Length (Tek bir Benzetim Tekrarının Uzunluğu) : 72 saat
- Hours Per Day (Günlük Süre) : 24 saat
- Base Unit Time : Minutes

Şekil 3-25 Run Setup Penceresi, Replication Parameters Sekmesi Görünümü

Benzetimi çalıştırdığınızda, eğer hızlandırırsanız, daha kısa sürede benzetimin çalışması sona erer ve sonuçları görmek isteyip istemediğimizi soran bir mesaj ekranda görünür. Evet seçeneğine tıkladığında model sonuçları ayrı bir sekmede görüntülenir ve ayrıca modelin yer aldığı aynı klasörde sonuçları içeren “.out” uzantılı bir dosya oluşur (Sonuç Raporu Görüntülemek için “SAP Crystal Reports” adlı bir uygulamanın indirilerek bilgisayara kurulması gerekmektedir).

Sonuç Raporu, her sayfada farklı bir öğeye ait istatistikleri içerecek şekilde, birkaç sayfadan oluşur. Sonuçlar sekmesinde, bir sonraki sayfaya geçmek için Şekil 3-27’de gösterildiği gibi, ► düğmesine basılması gerekir (mouse ile sayfayı aşağı kaydırarak bir sonraki sayfaya geçilmemektedir).



Şekil 3-27 Sonuç Raporu Görünümü

- 1. sayfada;** sistemden çıkış yapan toplam varlık sayısı görülür. Bu sayı, 30 tekrarın ortalamasıdır. Tek bir tekrar yapmış olsaydık, sadece bu tekrarda çıkış yapan toplam varlık sayısı olurdu.
- 2. sayfada;** Entity (varlık) istatistikleri
- 3. sayfada;** Queue (kuyruk) istatistikleri
- 4. ve 5. sayfada;** Resource (kaynak) istatistikleri
- 6. sayfada;** User Specified (Kullanıcı Tarafından Tanımlanmış) istatistikler görüntülenmektedir.

Not : Yukarıda belirtilen sayfa numaraları, her modelde sabit değildir, bizim modelde elde edilen sonuç raporu için verilmiştir. Örneğin; bir çok farklı tipte varlık ve çok daha fazla miktarda kaynak tipi veya kullanıcı tarafından tanımlanmış sistem değişkenleri sayısı çok daha fazla olan başka bir modelin sonuç raporunda, sayfalar daha uzun olacağından rapor toplam sayfa sayısı da artacaktır.

Her sayfadaki istatistikler; “Average”, “HalfWidth”, “MinimumAverage”, “Maximum Average”, “Minimum Value”, “Maximum Value” başlıkları altında (sütunlarda), tablo olarak sergilenmektedir.

Average: tüm tekrarlar arasındaki ortalama

HalfWidth: %95 güven aralığı sınırlarının ortalama (Average) değerinden uzaklığı

Minimum Average: Tekrarlar arasındaki en düşük ortalamaya sahip tekrarın değeri

Maximum Average: Tekrarlar arasındaki en yüksek ortalamaya sahip tekrarın değeri

Minimum Value: Benzetim süresince karşılaşılan en düşük değer (tüm tekrarlar içerisinde)

Maximum Value: Benzetim süresince karşılaşılan en yüksek değer (tüm tekrarlar içerisinde)

## Sonuç Raporu – Entity (Varlık)

### Entity

#### Time

VA Time	Average	HalfWidth	Minimum Average	Maximum Average	Minimum Value	Maximum Value
hasta	20.9075	0,32	18.9150	22.5636	5.4067	161.28
NVA Time	Average	HalfWidth	Minimum Average	Maximum Average	Minimum Value	Maximum Value
hasta	3.5020	0,09	3.0777	4.1409	0.00	19.9992
Wait Time	Average	HalfWidth	Minimum Average	Maximum Average	Minimum Value	Maximum Value
hasta	16.4033	4,57	5.2413	69.2604	0.00	439.60
Transfer Time	Average	HalfWidth	Minimum Average	Maximum Average	Minimum Value	Maximum Value
hasta	0.00	0,00	0.00	0.00	0.00	0.00
Other Time	Average	HalfWidth	Minimum Average	Maximum Average	Minimum Value	Maximum Value
hasta	0.00	0,00	0.00	0.00	0.00	0.00
Total Time	Average	HalfWidth	Minimum Average	Maximum Average	Minimum Value	Maximum Value
hasta	40.8128	4,78	27.3655	94.6899	5.6918	511.16

#### Other

NumberIn	Average	HalfWidth	Minimum Average	Maximum Average		
hasta	982.67	11,82	921.00	1038.00		
Number Out	Average	HalfWidth	Minimum Average	Maximum Average		
hasta	978.67	12,03	926.00	1031.00		
WIP	Average	HalfWidth	Minimum Average	Maximum Average	Minimum Value	Maximum Value
hasta	9.4814	1,18	6.1900	22.7229	0.00	88.0000

Sonuç raporunun, Entity (Varlık) İstatistiklerinde, varlıkların sistemde geçirdikleri süre, bekleme süresi ortalamaları ile sisteme giriş çıkış yapan ve sistemde bulunan varlık miktarları ile ilgili istatistikler yer almaktadır.

- Total Time:**

Hastalar, Acil Serviste ortalamada 40.8 dakika zaman geçirmekte ve %95 güven aralığı [36 , 45.6] Yapılan 30 tekrarın sonuçlarında, sistemde bulunma süresi ortalaması en az 29.5 dk., en fazla ~ 70 dk. olarak çıkmış,  
30 tekrar içerisindeki tüm varlıklar arasında sistemde en az zaman geçiren 5.7 dk., en fazla zamanı geçiren varlık da 511 dk. sistemde kalmıştır.

- Number In:**

Acil Servise her 72 saatte ortalama 983 hasta gelmiş (tekrar uzunluğunu 72 saat olarak tanımlamıştık) ve %95 güven aralığı [971, 995] kişi.



- **WIP:**

Aynı anda sistemde bulunan hasta sayısı (WIP–Work In Progress), ortalama 9.5 kişi (yani her hangi bir anda Acil Servise birisi gelse ve baksa, içeride ortalamada 9-10 kişi görecektir), %95 güven aralığı [8.3 , 10.7] kişi.

Yapılan 30 tekrarda, sistemde aynı anda yer alan hasta sayısı ortalaması en az 6.2, en fazla 22.7 kişi olarak görülmüştür.

Tüm tekrarlar ve bu tekrarlardaki tüm zamanlarda WIP en az 0 kişi olarak görülmektedir (sistem bazı anlarda tamamen boş kalmış ki bu teknik olarak imkansız değil)

En yüksek WIP ise 88 kişi olarak görünmektedir (bu istatistik bize bir keresine aynı anda sistemde toplam 88 kişi bulunduğunu söylüyor).

Modelimizde, tek bir tip varlık vardır (hasta). Bazı modellerde, değişik tipte varlıklar bulunabilir (örn. montajı yapılan parçalar için; parca1, parca2, parca3 vb. farklı varlık tipleri). Aslında, modelimizde hastalar aciliyet durumuna göre 3 farklı tip varlık olarak tanımlanabilirdi (Kırmızı\_hasta, Sarı\_hasta, Yeşil\_hasta gibi) ve böylece varlık istatistikleri sayfasında her varlık tipi için ayrı satır gösterilirdi.

İlave bir çalışma konusu olarak, varlıkların önce sisteme tek tip hasta varlığı olarak girmesi, daha sonrasında Assign modüller ile varlık tiplerinin değiştirilerek Kırmızı\_hasta, Sarı\_hasta, Yeşil\_hasta tipinden birisi olarak değişmesi, hatta varlık görüntüsünün de buna göre değişmesi detaylarını içeren model değişikliğini çalışabilirsiniz. Modelde böyle bir değişiklik yaparsak, modeldeki diğer tanımlar, değişkenler vb. diğer her hangi bir hususta daha değişiklik yapmamıza gerek duyulup duyulmayacağını da düşünüp gerekirse bu değişiklikleri de yaparak yeni modeli oluşturabilirsiniz.

## Sonuç Raporu - Queue (Kuyruk) İstatistikleri

### Queue

#### Time

Waiting Time	Average	HalfWidth	Minimum Average	Maximum Average	Minimum Value	Maximum Value
boş yatak bekleme.Queue	13.2653	4,50	2.7524	65.9446	0.00	436.31
Doktor ile ikinci görüşme.Queue	0.6644	0,04	0.4712	0.8740	0.00	10.0649
Doktor ile ilk görüşme.Queue	0.6467	0,05	0.3903	1.0061	0.00	18.9017
Kan Testi.Queue	0.1626	0,01	0.08338327	0.2307	0.00	5.8418
Röntgen.Queue	0.3957	0,14	0.00	1.3623	0.00	14.6710
Triyaj ve Kayıt.Queue	2.3116	0,14	1.6847	3.3227	0.00	28.6889

#### Other

Number Waiting	Average	HalfWidth	Minimum Average	Maximum Average	Minimum Value	Maximum Value
boş yatak bekleme.Queue	3.1126	1,08	0.6288	15.7741	0.00	80.0000
Doktor ile ikinci görüşme.Queue	0.03598461	0,00	0.02566207	0.05022528	0.00	3.0000
Doktor ile ilk görüşme.Queue	0.1492	0,01	0.08730698	0.2435	0.00	3.0000
Kan Testi.Queue	0.00755613	0,00	0.00346452	0.01209748	0.00	2.0000
Röntgen.Queue	0.00339718	0,00	0.00	0.01151217	0.00	2.0000
Triyaj ve Kayıt.Queue	0.5349	0,04	0.3685	0.8100	0.00	11.0000

Modelimizde, 6 adet kuyruk bulunmaktadır. Sonuç Raporunda, her bir kuyruk için, ayrı satırlarda, Bekleme Süresi ve Bekleyen kişi (kuyruk uzunluğu) ile ilgili istatistikler yer almaktadır.

Modeli yavaş hızda çalışırken gözlemlerseniz, kuyruklardaki yoğunluğu görerek genel bir fikir edinebilirsiniz. Ancak rapordaki istatistikler, modelin tüm çalışma süresini dikkate aldığı ve bir çok tekrarı kullandığı için, daha net sonuçlar sunmaktadır.

- **Bekleme Süresi (Waiting Time)** istatistikleri, kişi başına ortalama bekleme süreleridir (yani o kuyruğa girmiş olan tüm varlıkların tek tek kuyrukta bekleme süreleri toplamının, o kuyruğu kullanan toplam varlık sayısına bölünmesi ile hesaplanmaktadır, birimi “dk./kişi”dir).

Örneğin; Boş Yatak Bekleme kuyruğunda hasta başına ortalama 13.3 dk. beklendiği görülmektedir. Tabiki bu kuyruğa gelen bazı hastalar hiç beklemeden geçmişlerdir ve bu hastaların bu kuyruktaki bekleme süresi 0’dır. Bekleme süresi yüksek olan diğer bazı hastalar nedeniyle ortalama 13.3 dk. çıkmaktadır. Ayrıca, bu değeri ortalama bekleme süresinin 30 tekrar ortalaması olarak görmeliyiz, çünkü zaten tek bir benzetim modeli tekrarının sonucunda o tekrar için bir bekleme süresi ortalaması değeri hesaplanıyor, yani 30 farklı tekrardan gelen birbirinden bağımsız 30 farklı bekleme süresi ortalaması değeri var ve sonuç raporunda sunulan Average değeri bunların da ortalamasıdır.

Not : Tek bir benzetim tekrarı içerisinde, bir hastanın kuyrukta bekleme süreleri, o kuyruğa geldiği anda zaten beklemekte olan önündeki hastaların sayısından ve bunların bekleme sürelerinden etkilenir. Bu yüzden, bekleme süreleri kişi bazında birbirinden bağımsızdır diyemeyiz. Ancak, benzetimi birbirinden bağımsız olarak bir çok kere çalıştırdığımızda, bu tekrarlardan elde edilen ortalamalar birbirinden bağımsız diyebiliriz. (Bağımsız değerlerin ortalaması daha güvenilirdir !)

Boş Yatak Bekleme kuyruğunda ortalama bekleme süresinin %95 güven aralığı hesabı için; ilk sütundaki “Average” değerinden “HalfWidth” sütundaki değer çıkarılarak alt sınır, toplanarak üst sınır bulunur. Bu şekilde, %95 güven aralığının [8.8 , 17.8] dakika olduğu hesaplanabilir. Bu yine,

30 farklı tekrardan gelen 30 farklı ortalama bekleme süresinin güven aralığıdır (yani bekleme süresi ortalamalarının %95'i bu aralık içerisinde).

Yapılan 30 tekrar arasında, ortalama bekleme süresi sonucu en düşük 2.75 dk./kişi, en yüksek 95.9 dk./kişi olarak çıkmıştır.

Yapılan 30 tekrar dikkate alınarak sisteme girmiş olan tüm hastalar arasında bu kuyrukta en fazla bekleyen kişi 436 dk. beklemiş, en az bekleyen kişi de 0 dk. beklemiş olduğu görülmektedir (hiç beklemeden devam etmiş bir çok hastalardan dolayı minimum 0 çıkmaktadır).

Boş yatak bekleme süresinin, ortalamada 13 dk./kişi olması iyi midir, kötü müdür şeklinde bir hükme varmak bu sonuç tablosu kullanılarak verilecek bir karar değildir. Bunun için; başka bir referans acil servis veya sektör ortalaması veya bu acil servisin daha önceki durumu veya başka bir acil servis sistem tasarımı benzetim sonuçları ile karşılaştırma yapılmalıdır (bu karşılaştırma ve sonunda verilecek karar da ayrı bir istatistiki hipotez testini gerektirir). Ya da, başka bir Acil Servis ile karşılaştırma yapılmayacak ise, acil servisler ile ilgili ulusal veya uluslararası standartlar, kurallar arasından yatak bekleme süresi ile ilgili bir kural, yeterlik standardı varsa onunla karşılaştırma yapılabilir (Örneğin; uluslararası standardı 5 dk. ama bizim sistem 13 dk., buna göre bizim modellediğimiz sistem yetersiz, uygun değil görünüyor diyebiliriz). Ayrıca, modele dahil edilmeyen bazı yönetsel faktörler de son karar da önemlidir.

- **Bekleyen Sayısı (Number Waiting) istatistikleri;** kuyrukta bekleyen kişi sayısı ortalamalarıdır (yani her hangi bir anda kuyruğa bakıldığında, kaç kişinin kuyrukta olduğunu görmeyi bekleriz sorusunun cevabıdır) ve birimi "kişi"dir.

Örneğin; Boş Yatak Bekleme kuyruğundaki kişi sayısı ortalaması 3.1 olarak görülmektedir. Boş Yatak Bekleme kuyruğunda bekleyen kişi sayısı ortalamasının %95 güven aralığı, ilk sütundaki "Average" değerinden "HalfWidth" sütundaki değer çıkarılarak ve toplanarak [2.0 , 4.2] kişi olarak bulunur. Yapılan 30 tekrar arasında, ortalama kuyruk uzunluğu en düşük 0.6, en yüksek 15,8 kişi olarak çıkmıştır. Yapılan 30 tekrardaki tüm zamanlar birlikte dikkate alınarak, bu kuyrukta aynı anda en fazla 80 kişi bulunduğu görülmektedir (böyle bir kuyruk çok uç bir değer, outlier).

Boş yatak bekleme kuyruk uzunluğu ile ilgili istatistikler için de benzer şekilde başka sistem/ benzetim sonuçları veya belirli standart, kurallar ile karşılaştırma yapılması gerekir. Aksi takdirde, sadece bu değerlere bakarak bir hükümde bulunmak anlamlı olmaz.

Bir sistemdeki kuyruklara ait istatistikler, ayrıca, bize sistemdeki "Darboğazlar (Bottlenecks)" hakkında da önemli bilgi sunar. Belirgin bir şekilde, diğer kuyruklardan daha fazla bekleme süresine sahip bir kuyruk, o işlemden kullanılan kaynakların yeterliliğinin veya genel süreç akışı tasarımının sorgulanması için sağlam bir gerektirir.

Modelimizdeki kuyruklara ait istatistikler birlikte incelendiğinde, özellikle Boş Yatak Bekleme kuyruğunun diğer kuyruklara göre daha uzun bekleme süresi ortalamasına ve daha fazla kuyruk uzunluğu ortalamasına sahip olduğu görülmektedir. Ayrıca, güven aralığı hesabında kullandığımız HalfWidth değeri, bize varyans (dağılım) ile ilgili fikir vermektedir. Varyansın ortalamaya oranı yüksek olduğu durumlar elde edilecek sonuçların geniş bir aralığa dağılacağını, belirsizliğin yüksek olacağını sinyali verir. Triyaj ve Kayıt işleminin kuyruğu da, kalan diğer 4 kuyruktan daha yüksek istatistiklere sahip görülmektedir.

Bu iki istasyondaki kuyruklara (özellikle de Yatak Bekleme kuyruğına) neden olan faktörlere yönelik iyileştirmelerin, sistemdeki genel yükün dengelenmesi bakımından uygun olabileceğı görünüyor. En azından iyileştirme alternatiflerinin de benzetimleri yapılarak gerçekleştirilecek farklar da tahmin edilebilir (Benzetim modelleri bu maksatlar için çok kullanışlı araçlardır).

Yatak Bekleme kuyruğına etki eden temel faktör, Acil Servis yatak kapasitesidir. Modelimiz, 6 yatak kapasitesi üzerine kurulmuştu. Yatak kapasitesini 7, 8, 9 vb. farklı alternatiflerle modelimizi yeniden çalıştırabilir ve diğer tüm faktörler aynı kaldığında yatak kapasitesi artışının istatistiklere nasıl yansyacağı incelenebilir. Bu incelemeler sonucunda yatak kapasitesinin artırılmasını teklif edebiliriz. Tabiki, tasarladığımız sistemin maliyet ve yer kısıtları ile birlikte değerlendirme yapmak ve bunları da dikkate almak gerekir.

Triyaj ve kayıt işlemi ile diğer işlemler arasında da benzer veya ortak kaynak (hemşire) kullanımı olduğu dikkate alınarak, kaynakların işlemlere dağılımı ile ilgili farklı alternatifler düşünülebilir ve aynı şekilde benzetim modeli ile denemesi yapılarak, mevcut modeldeki istatistikler ile karşılaştırılabilir. Özellikle, kan testinin kuyruğı diğer kuyruklara göre belirgin bir şekilde düşük görünüyor. Aslında, bu iyi bir şey gibi görünse de, sistemdeki süreçte dengeyi sağlama adına, bu işlemde sürekli burada duracak ayrı bir test hemşiresine gerek olmayacağı, muayenede görevli diğer hemşirelerden uygun olanın test işlemini yapabileceğı gibi bir alternatif düşünülebilir.

Ancak, bu konuda tasarlanacak alternatifler ve denemeler için önce Resource (kaynak) istatistiklerinin de incelenmesi daha uygun olacaktır.

## Sonuç Raporu - Resource (Kaynak) İstatistikleri

### Resource

#### Usage

Instantaneous Utilization	Average	HalfWidth	Minimum Average	Maximum Average	Minimum Value	Maximum Value
doktor	0.6579	0,01	0.6072	0.7116	0.00	1.0000
muayene hemşiresi	0.6579	0,01	0.6072	0.7116	0.00	1.0000
Röntgen Teknisyeni	0.0959	0,01	0.05745305	0.1377	0.00	1.0000
test hemşiresi	0.1389	0,00	0.1183	0.1641	0.00	1.0000
uzman hemşire	0.6548	0,01	0.6129	0.6961	0.00	1.0000
Number Busy	Average	HalfWidth	Minimum Average	Maximum Average	Minimum Value	Maximum Value
doktor	1.9738	0,03	1.8217	2.1349	0.00	3.0000
muayene hemşiresi	1.9738	0,03	1.8217	2.1349	0.00	3.0000
Röntgen Teknisyeni	0.0959	0,01	0.05745305	0.1377	0.00	1.0000
test hemşiresi	0.1389	0,00	0.1183	0.1641	0.00	1.0000
uzman hemşire	1.3096	0,02	1.2259	1.3922	0.00	2.0000

Sonuç raporunun, Resource (Kaynak) istatistiklerinde, modelimizde tanımladığımız kaynakların (hemşire, doktor vb.) kullanım durumlarına ilişkin bilgiler, her bir kaynak tipi için ayrı satır olacak şekilde sunulmaktadır. Modelimizde, tümü sağlık çalışanı olan, 5 farklı tip kaynak bulunmaktadır (doktor, uzman hemşire muayene hemşiresi, test hemşiresi, röntgen teknisyeni). Bunlardan doktor ve muayene hemşiresinden 3'er kişi, uzman hemşireden 2 kişi ve diğerlerinden 1'er kişi bulunmakta ve tanımlandıkları işlemlerde kullanılmaktadırlar (görev almaktadırlar).

Benzetim modellerinde, tanımlanan kaynaklar cansız varlıklar da olabilir (çoğu modelde makine, cihaz vb. kaynaklar bulunur), ayrıca bir model içerisinde kaynakların bazıları canlı (örn. görevli personel), bazıları da cansız olabilir, buna bir engel yoktur. Bizim modelde, kaynak olarak sadece sağlık çalışanları var.

Sonuç Raporu Resources bölümünde yer alan istatistikler başlıklar halinde aşağıda açıklanmıştır:

- **Instantaneous Utilization (Anlık Kullanım):**

Her satırda ayrı kaynak tipi olacak şekilde, o tipteki bir kaynağın benzetim süresince ortalama ne oranda dolu (meşgul) olduğunu gösterir (doluluk oranı da denir).

Tanımı itibarı ile, Anlık Kullanım (kullanım oranı) değeri 1'in üzerinde olamaz ve birimi de yoktur (doluluk oranı dk./dk. olduğundan pay ve paydadaki birimler birbirini götürür).

Average sütununda yer alan değer, modelin 30 tekrarında ayrı ayrı bulunan ortalama kullanım oranı değerlerinin ortalamasıdır. Modelde, bir kaynak tipinden 1 adet var ise, bu kaynağın meşgul olduğu zamanların toplamı benzetim süresine bölünerek doluluk oranı hesaplanır. Modelde o kaynak tipinden birden fazla sayıda var ise, bunların ortalaması alınarak o kaynak tipine ait doluluk oranı bulunur.

Örneğin, doktor ve muayene hemşiresi kaynak tiplerinin ikisinin de ortalama %66 oranında kullanımda olduğu görülmektedir (oranların eşit olması doğaldır, çünkü muayenede bir doktor her zaman yanında bir hemşire ile birlikte hasta bakıyor). Yani, sistemdeki bir doktor toplam zamanın ortalama %66'sında sorumlu olduğu işlemde fiilen iş yapmış (meşgul olmuş).

Bir uzman hemşirenin ortalama doluluk oranı, bir doktorun ortalama doluluk oranına hemen hemen eşit görünmekte, test hemşiresi ve röntgen teknisyeni doluluk oranı ortalamaları ise nispeten daha düşük (en düşük röntgen teknisyeni) görünmektedir.

Anlık kullanım ile ilgili %95 güven aralığı, minimum ve maksimum değerler sütunları, bir önceki kısımda (Kuyruk İstatistiklerinde) açıklandığı gibi incelenebilir, burada tekrar açıklanmayacaktır.

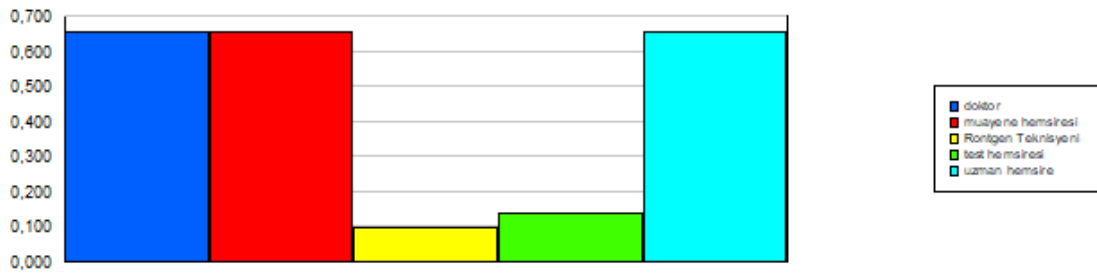
- **Number Busy (Dolu Kaynak Sayısı):**

Anlık kullanım (doluluk oranı) ile doğrudan ilişkilidir ve “Anlık Kullanım” değerlerinin kaynak sayısı katı kadardır. Dikkat edilirse, modelde sadece 1 adet olan kaynak tipleri için “Anlık Kullanım” ve “Dolu Kaynak Sayısı” değerleri eşittir. Diğerleri için kaynak sayısı ile çarpımı kadardır.

Number Scheduled	Average	HalfWidth	Minimum Average	Maximum Average	Minimum Value	Maximum Value
doktor	3.0000	0,00	3.0000	3.0000	3.0000	3.0000
muayene hemşiresi	3.0000	0,00	3.0000	3.0000	3.0000	3.0000
Röntgen Teknisyeni	1.0000	0,00	1.0000	1.0000	1.0000	1.0000
test hemşiresi	1.0000	0,00	1.0000	1.0000	1.0000	1.0000
uzman hemşire	2.0000	0,00	2.0000	2.0000	2.0000	2.0000

Scheduled Utilization	Average	HalfWidth	Minimum Average	Maximum Average
doktor	0.6579	0,01	0.6072	0.7116
muayene hemşiresi	0.6579	0,01	0.6072	0.7116
Röntgen Teknisyeni	0.0959	0,01	0.05745305	0.1377
test hemşiresi	0.1389	0,00	0.1183	0.1641
uzman hemşire	0.6548	0,01	0.6129	0.6961



- **Number Scheduled (Planlanan Miktar) ve Scheduled Utilization (Planlanan Kullanım) :**

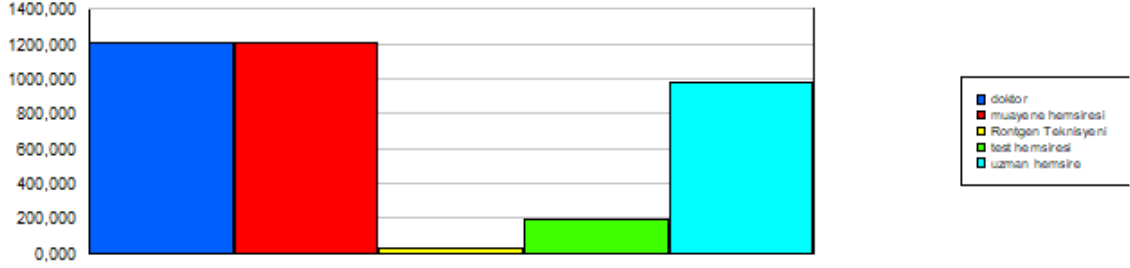
Number Scheduled (Planlanan Miktar) ve Scheduled Utilization (Planlanan Kullanım) başlıklı istatistikler modelimizde “Schedule” özelliği kullandığımızda anlamlı sonuçlar verir. (Schedule özelliğinin kısa açıklaması için Basit Kuyruk Modeli örneğine tekrar bakabilirsiniz).

Bizim modelimizde “Schedule” özelliğini hiç kullanmadık. Bu sebepten dolayı, dikkat edilirse, “Number Scheduled” ile ilgili tüm değerler kaynak sayılarına eşittir ve “Scheduled Utilization” altındaki tüm değerler daha önceki “Anlık Kullanım” değerleri ile bire bir aynıdır.

Bunların altındaki sütun grafiği, “Anlık Kullanım” (Doluluk Oranı) istatistiklerinde yer alan ortalama doluluk oranlarının grafiğidir.

## Usage

Total Number Seized	Average	HalfWidth	Minimum Average	Maximum Average
doktor	1210.47	16,16	1130.00	1297.00
muayene hemşiresi	1210.47	16,16	1130.00	1297.00
Röntgen Teknisyeni	33.5667	2,53	21.0000	47.0000
test hemşiresi	197.13	6,11	168.00	233.00
uzman hemşire	982.77	11,81	921.00	1035.00



- **Total Number Seized (Yakalanan Miktar):**

Her satırda yer alan kaynak tipi için, benzetim süresinde o tip kaynaktan farklı seferde kaç kere kullanıldığı görülebilmektedir (30 tekrarın ortalaması şeklinde).

Örneğin, doktor kullanımı ortalamada 1210 farklı kerelerde gerçekleşmiştir. Doktorlar, hem ilk muayene hem de ikinci muayene işlemlerinde kullanıldığından (aynı hasta 1 veya 2 defa doktora muayeneye girmekte) değerleri yüksek görülmektedir.

Dikkat edilirse, uzman hemşire için kullanım sayısı (982.77), daha önce Entity İstatistik sayfasında gösterilen sisteme giriş yapan toplam varlık sayısı ortalamasına eşittir (Sisteme giren tüm varlıklar mutlaka ve sadece 1 kere uzman hemşireyi görüyor).

## Sonuç Raporu – User Specified (Kullanıcı Tarafından Tanımlanmış) İstatistikler

### User Specified

#### Tally

Interval	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
gec taburcu suresi	122.98	4,23	116.76	137.11	62.2664	346.03
hastane yatis suresi	52.4721	5,15	45.6769	70.0759	30.5220	132.52
hemen taburcu suresi	29.8435	8,88	18.1609	62.6875	5.6918	295.71
normal taburcu suresi	52.5624	3,57	47.3195	65.4001	24.5729	253.49

#### Time Persistent

Variable	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
bos yatak	1.4860	0,22	0.8037	1.9870	0.00	6.0000

Sonuç Raporunun son bölümünde, benzetim modelini oluşturan kullanıcı tarafından, sisteme tanımlanan “Record (Kayıt)” modülleri ile ilişkili istatistikler ile yine kullanıcı tarafından tanımlanarak model içerisinde kullanılan sistem değişkenleri istatistikleri yer almaktadır.

**Interval** başlıklı kısımda; Record modülleri içerisinde hesaplanarak kaydedilmesini istediğimiz, sistemden çıkış aşamalarına göre çıkış süreleri ortalamaları görülmektedir. Buna göre, 30 tekrarın ortalaması olarak;

Sisteme giren bir hasta, ortalama 30 dakika sonra hemen taburcu olarak Acil Servisten çıkmakta, Doktor ile ikinci görüşmeye giren hastalar eğer serum ve gözleme alınmaz ise ortalama 52.5 dakikada Acil Servisten çıkmakta, Serum ve Gözlem için tutulanlar da Acil Servise giriş yaptıktan ortalama 123 dakika (~ 2 saat) sonrasında Acil Servisi terk etmiş olmaktadır.

**Variable** başlıklı kısımda ise; modelde ayrıca tanımladığımız ve model içerisinde kullandığımız sistem değişkenleri ile ilgili istatistikler yer almaktadır. Her satır ayrı bir değişken içindir. Bizim modelimizde, tanımladığımız tek bir sistem değişkeni olduğundan tek bir satır görülmektedir.

Tablodaki istatistikleri incelendiğimizde, “bos yatak” adlı değişkeninin ortalama değerinin 1.486 (yaklaşık 1.5) olduğu görülmektedir. Yani, her hangi bir anda acil servise giren bir hastanın ortalamada 1-2 boş yatak ile karşılaşması beklenmektedir. Toplam 6 yatak kapasitesinin ortalamada 1.5’i boş. Buradan da yatak doluluk oranını:  $4.5/6 = \%75$  olarak hesaplayabiliriz.

### Bu Örnekte Yeni Ne Öğrendik ?

- Decide Modülünün “N-Way by Chance” olarak kullanımı
- Assign Modülü kullanımı, Varlıklara Özellik (Attribute) tanımlama
- Farklı özelliğe sahip varlıkların farklı işlem süresi, olasılık alabilmesi (Expression Builder)
- Hold Modülü kullanımı ve bir Sistem Değişkeni (Variable) tanımlama, güncelleme
- Record Modülünün kullanımı
- Bir İşlem için birden fazla kaynak tipi tanımlama



## Hastane Acil Servis Modelini Kullarak Bazı Durumların Değerlendirilmesi:

Aşağıda verilen durumlara göre modelinizde değişiklikler yapın.

1. Acil Servise 1 uzman hemşire tayin oldu. Bu arada, Test Hemşiresi de başka bir hastaneye tayin oldu ve ayrıldı. Yani, sistemdeki toplam “Uzman Hemşire” sayısı 3 olacak. Bu durum sonrası; hastane yönetimi şu şekilde yeni bir düzenleme yapılmasını düşünüyor:

*Kan Testi işlemi için ayrıca bir “test hemşiresi” olması yerine bu testin bir “uzman hemşire” tarafından yapılması (her hangi bir uzman hemşireden birisi yapılabilir).*

Karar öncesi, benzetim modeli kullanarak sisteme olası etkileri görülmek isteniyor.

Modelde Yapılacak Değişiklikler:

Artık sistemde Test Hemşiresi olarak ayrı bir kaynak (resource) olmayacak. Kan Testi işlemi için tek bir resource olacak o da Uzman Hemşire. Uzman Hemşireler artık hem “Trijay ve Kayıt” işleminde hem de “Kan Testi” işleminde kaynak olarak görülecek. Ancak, her iki işlemde de bekleyen hastalar olması durumunda, bir uzman hemşirenin hangi işleme öncelik vereceğini modelde tanımlamak istiyoruz. Bunun için; “Trijay ve Kayıt” işleminin Priority alanındaki değer High(1) olarak seçilecek (Öncelik Triyaj ve Kayıt).

Trijay ve Kayıt işlemi ve Kan Testi işlemi kuyurğu yeni düzenleme sonrası nasıl değişiyor?

2. Mevcut yatak kapasitesinin yeterli olmadığı, özellikle aciliyeti yüksek olan hastaların boş yatak için bekleme süreleri konusunda şikayetler artmıştır. Bu durum ile ilgili olarak, hastane yönetimi aşağıdaki alternatifi düşünüyor:

*aciliyeti düşük (triyaj kodu=3) olan hastaların yatak kullanmaması, bunların “Trijay ve Kayıt” işlemi sonrasında ayrı bir yerde muayene edilmeleri, bu işlem için hastaneye yeni gelen stajyer doktorların kullanılması (2 stajyer doktor), test ve röntgen işlemleri aynen olduğu gibi devam edecek*

Karar öncesi, benzetim modeli kullanarak sisteme olası etkileri görülmek isteniyor.

Modelde Yapılacak Değişiklikler:

Trijay ve Kayıt işlemi sonrasında, Hold modülüne sadece triyaj kodu=1 veya 2 olan hastalar gidecek, triyaj kodu=3 olan hastalar “Muayene” adı verilecek ayrı yeni bir process modülüne yönlendirilecek (Decide Modülü kullanarak). Bu modülde, kaynak olarak “Stajyer Doktor” tanımlanacak ve bunların sayısının 2 olduğu modelde belirtilecek. Bu işlemde de kuyruk olacak. Bu işlemde çıkan hastalar, modelde belirtilen oranlara göre aynı şekilde test veya röntgene yönlendirilecek veya taburcu edilecek.

Bu düzenlemeden sonra boş yatak bekleme kuyruğunda nasıl bir değişiklik oluyor ?

Ayrıca, doktor ve muayene hemşiresi sayısı 1’er azalırsa (tayin, izin vb.) sistem nasıl etkilenir ?

3. Madde-2’de belirtilen durum ile ilgili modelde yapılan deęişiklięin üzerine devam edin.

65 yaşı üstü hastalara öncelik verilmedięi konusunda da şikayet gelmiştir. Geçmiş kayıtlardan, tüm gelen hastaların yaşlarının; ortalaması 40, standart sapması 12 olan Normal Dağılıma uyduęu görölmüştür. Bu durum ile de ilgili olarak, hastane yönetimi aşığıdaki alternatifi düşünüyor:

*aciliyeti ne olursa olsun 65 yaşı üstü hastaların yataęa alınması,  
bunların test veya röntgen işlemi yapılmadan hemen taburcu edilmemesi ve  
bunlara kan testi ve röntgen kuyruęunda öncelik verilmesi,  
doktor ile ikinci görüşmeden sonra da mutlaka serum, gözlem için bir süre kalmaları*

Karar öncesi, benzetim modeli kullanarak sisteme olası etkileri görölmek isteniyor.

Modelde Yapılacak Dięer Deęişiklikler:

Ayrı bir Assign modöünde gelen tüm hastalara, “yas” adlı yeni bir özellik (attribute) tanımlanacak (belirtilen olasılık dağılımına uygun olarak).

Triyaj ve Kayıt işlemi sonrası, eęer hasta 65 yaşı üstü ise triyajkod=1 olsa bile stajyer doktor muayenesine girmeyecek (yeni eklenen Decide modöü içerisindeki kriterler güncellenecek).

Kan Testi ve XRay adlı Decide modöündeki mantıksal kriter güncellenecek.

Kan Testi ve Röntgen kuyruklarında ayrı ayrı öncelik tanımlaması yapılacak.

Son Decide modöü sonrasında yeni bir Decide modöü ilave edilerek 65 yaşı üstü hastaların Normal Taburcu çıkışından çıkmaları engellenecek. 1\*2?E\_h.

4. Hastaların, “Doktor ile İlk Görüşme” işleminde geçirecekleri süreleri bir Özellik (Attribute) tanımlayın (görüşme yani muayene süresinin olasılık dağılımı, hastanın triyaj koduna göre farklılık gösterdięinden, “Triyaj ve Kayıt” işlemi sonrasında, “Doktor ile İlk Görüşme” işlemi öncesi ayrı bir assign modöü içerisinde)

#### ÖRNEK-4 ÜRETİM SİSTEMİ MODELİ

Bir üretim firması 2 müşterisinden aldığı siparişlere göre A, B, C ürünlerini üretmektedir.

Müşteri X'in siparişleri arası geçen süre, ort. 10 dk üstel dağılıma uymakta ve sadece A ürünü sipariş etmekte,

Müşteri Y'nin siparişleri arası geçen süre, ort. 7 dk üstel dağılıma uymakta ve verdiği siparişlerin %70'i B ürünü, %30'u ise C ürünü olmaktadır (Müşteri Y, A ürünü sipariş etmiyor).

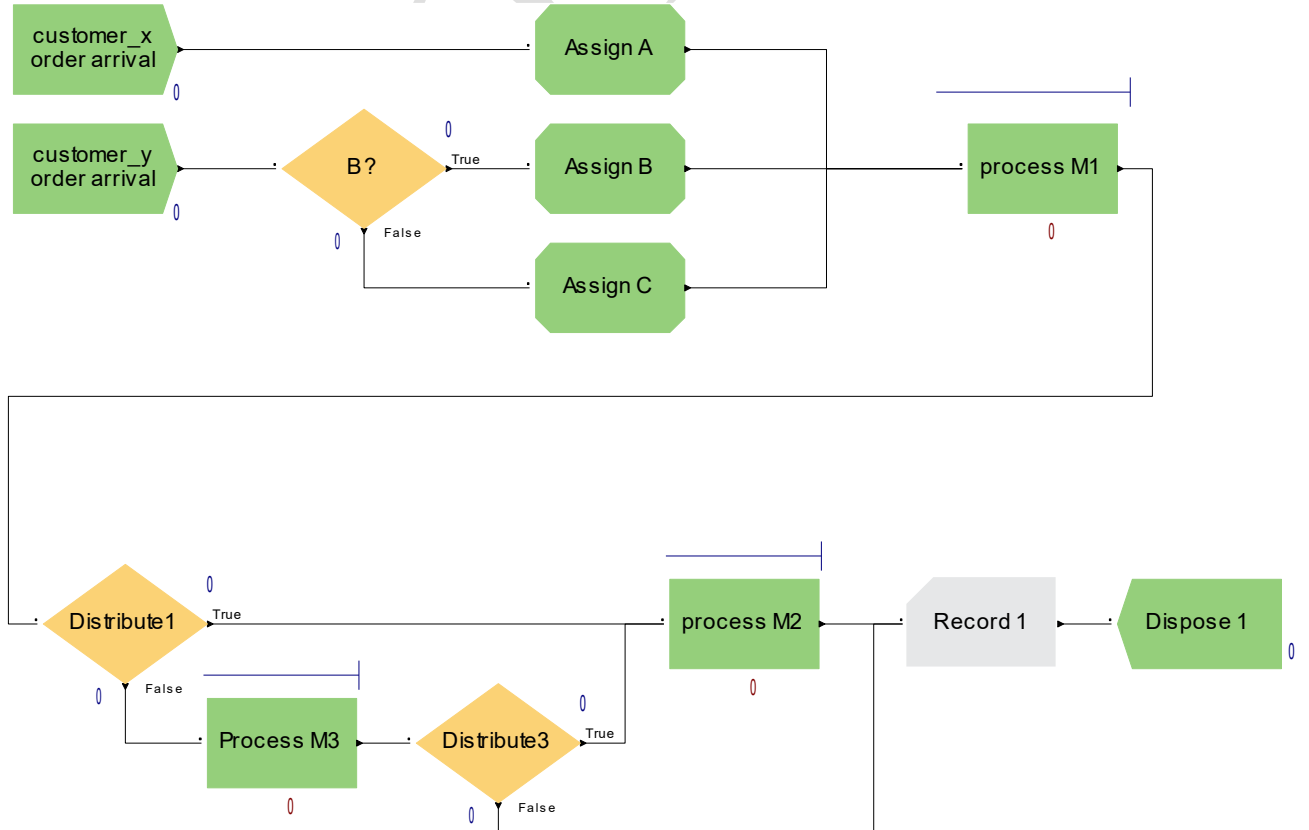
Ürün tiplerinin takip ettiği işlemler (M1, M2, M3) ve bu işlemlerde geçirecekleri sürelerin olasılık dağılımı aşağıdaki tabloda verilmiştir.

	M1 işlemi	M2 işlemi	M3 işlemi
A Ürünü	NORMAL ( 3 , 1 ) dk.	SABİT ( 3 ) dk.	-
B Ürünü	NORMAL ( 3 , 0.5 ) dk.	NORMAL ( 5 , 1 ) dk.	NORMAL ( 6 , 2 ) dk.
C Ürünü	NORMAL ( 2 , 0.5 ) dk.	-	NORMAL ( 5 , 2 )

M1, M2 ve M3 işlemlerinin her birinde 1'er adet bu işlemi yapacak makine vardır (Resource).

İşlemleri biten ürün sistemden çıkış yapmaktadır. Ürünler müşteriler tarafından teslim alınmaktadır.

Bu üretim sistemi için oluşturulan ARENA benzetim modeli aşağıda gösterilmiştir.



## Create Modülleri

Create ? X

Name: customer\_x order arrival Entity Type: Entity 1

Time Between Arrivals

Type: Random (Expo) Value: 10 Units: Minutes

Entities per Arrival: 1 Max Arrivals: Infinite First Creation: 0.0

Comment:

OK Cancel Help

Create ? X

Name: customer\_y order arrival Entity Type: Entity 1

Time Between Arrivals

Type: Random (Expo) Value: 7 Units: Minutes

Entities per Arrival: 1 Max Arrivals: Infinite First Creation: 0.0

Comment:

OK Cancel Help

## B? Adlı Decide Modülü

Decide ? X

Name: B ? Type: 2-way by Chance

Percent True (0-100): 70 %

Comment:

OK Cancel Help

## Assign Modülleri

Assign ? X

Name:  
Assign A

Assignments:

- Attribute, t\_arrival, TNOW
- Attribute, t\_m2, 3
- Attribute, t\_m1, NORM(3,1)
- Entity Picture, Picture.Blue Ball
- Entity Type, prodA
- <End of list>

Add... Edit... Delete

Comment:

OK Cancel Help

Assign ? X

Name:  
Assign B

Assignments:

- Attribute, t\_arrival, TNOW
- Attribute, t\_m3, NORM(6,2)
- Attribute, t\_m2, NORM(5,1)
- Attribute, t\_m1, NORM(3,0.5)
- Entity Picture, Picture.Green Ball
- Entity Type, prodB
- <End of list>

Add... Edit... Delete

Comment:

OK Cancel Help

Assign ? X

Name:  
Assign C

Assignments:

- Attribute, t\_arrival, TNOW
- Attribute, t\_m3, NORM(5,2)
- Attribute, t\_m1, NORM(2,0.5)
- Entity Picture, Picture.Red Ball
- Entity Type, prodC
- <End of list>

Add... Edit... Delete

Comment:

OK Cancel Help

## Process M1 Modülü

Process

Name: process M1 Type: Standard

Logic

Action: Seize Delay Release Priority: Medium(2)

Resources:

Resource, M1, 1  
<End of list>

Add...  
Edit...  
Delete

Delay Type: Expression Units: Minutes Allocation: Value Added

Expression: t\_m1

☒ Report Statistics

Comment:

OK Cancel Help

## Distribute1 Decide Modülü

Decide

Name: Distribute1 Type: 2-way by Condition

If: Entity Type Named: prodC

Comment:

OK Cancel Help

## Process M2 İşlemi

**Process** ? X

Name: Process M2 Type: Standard

Logic

Action: Seize Delay Release Priority: Medium(2)

Resources:

Resource, M2, 1  
<End of list>

Add...  
Edit...  
Delete

Delay Type: Expression Units: Minutes Allocation: Value Added

Expression: t\_m2

☒ Report Statistics

Comment:

OK Cancel Help

## Distribute3 Decide Modülü

**Decide** ? X

Name: Distribute3 Type: 2-way by Condition

If: Entity Type Named: prodB

Comment:

OK Cancel Help

## Process M3 Modülü

Process

Name: process M3 Type: Standard

Logic

Action: Seize Delay Release Priority: Medium(2)

Resources:

Resource, M3, 1  
<End of list>

Add...  
Edit...  
Delete

Delay Type: Expression Units: Minutes Allocation: Value Added

Expression: t\_m3

☒ Report Statistics

Comment:

OK Cancel Help

## Record Modülü

Record

Name: Record 1

Statistic Definitions:

Count, 1, No, Counter 1  
Time Interval, t\_arrival, No, cycletime  
<End of list>

Add...  
Edit...  
Delete

Comment:

OK Cancel Help



### Replication Parameters

Number of Replications:

Start Date and Time:

Warm-up Period:  Hours

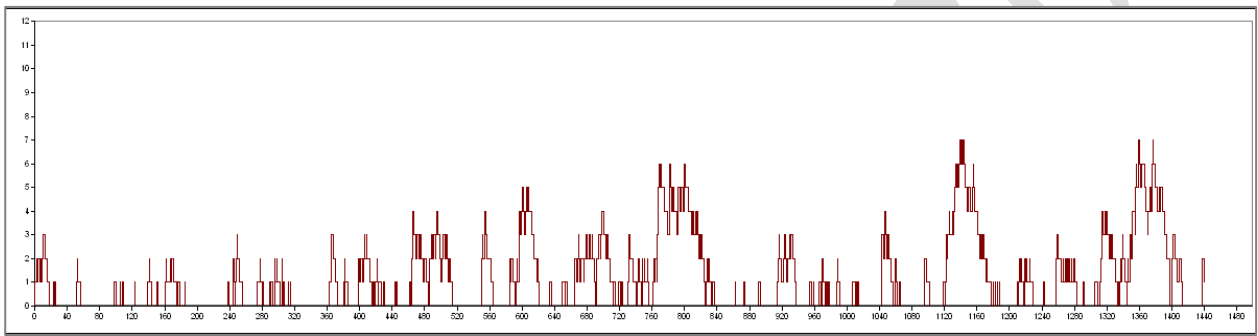
Replication Length:  Days

Hours Per Day:

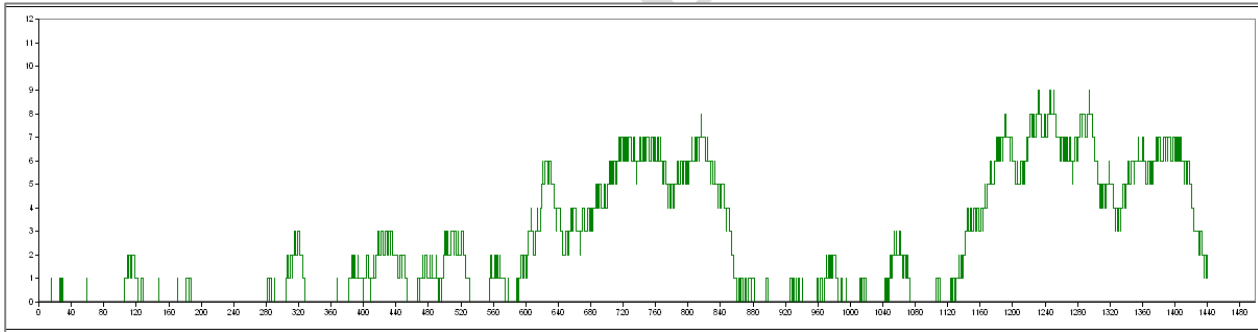
Terminating Condition:

Base Time Units:

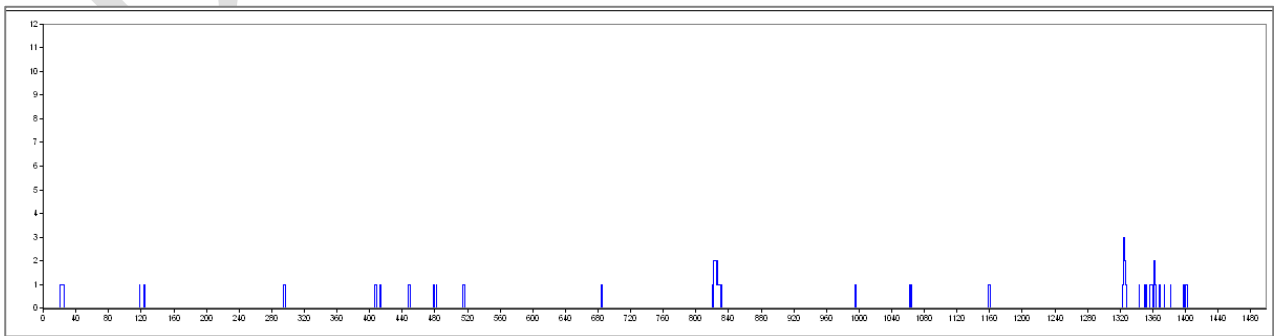
*Benzetim Tekrar Parametreleri (24 saat çalışacak)*



M1 İşlemi Kuyruğundaki Zamana Göre Kuyruk Uzunluğu Grafiği



M2 İşlemi Kuyruğundaki Zamana Göre Kuyruk Uzunluğu Grafiği



M3 İşlemi Kuyruğundaki Zamana Göre Kuyruk Uzunluğu Grafiği

## Kuyruklardaki Ortalama Bekleme Süreleri ve Kuyrukta Bekleyen Ürün Ortalamaları

### Queue

#### Time

Waiting Time	Average
process M1.Queue	4.2283
process M2.Queue	11.8551
Process M3.Queue	0.2768

#### Other

Number Waiting	Average
process M1.Queue	1.0680
process M2.Queue	2.4074
Process M3.Queue	0.04132808

## Üretim Sistemi Benzetim Modelinde Yapılacak Değişiklikler (Çalışma Soruları)

Durum-1 :

Y müşterisi; A ürünü de sipariş edecektir. %25 A ürünü, %25 B ürünü ve %25 C ürünü

M3 işlemi için ilave bir makine dahi olacaktır (Resource aynı, sayısı 2 olacak)

Durum-2 (Durum-1'e ilave olarak) :

X müşterisinin siparişleri 5'erli olarak gelecek, Gelişlerarası süre ~ EXPO(30 dk)

Y müşterisinin siparişleri 10'arlı olarak gelecek (her siparişte tümü ya A, ya B yada C ürünü olarak), Gelişlerarası süre ~ EXPO(50 dk)

Durum-3 (Durum 1 ve 2'ye ilave olarak):

Müşteriler tamamlanan ürünleri kendisi teslim almayacak, bunun yerine firma teslim edecek.

Her 20 ürün bittiğinde bir kamyon dolacak ve yola çıkacak. Teslimat için yol süresi: X müşterisi için 30-40 dk Düzgün Dağılım, Y müşterisi için 60-70 dk Düzgün Dağılım. İki araç var.

Yukarıdaki tüm durumlar için; kuyruk istatistiklerindeki değişimi gözlemleyin ve yorumlayın.

## ÖRNEK-5 MOTORİN DEPOSU ENVANTER KONTROL MODELİ

Aksaz Deniz Üs K.İğında konuşlu bulunan veya liman ziyaretine gelen gemilerin motorin taleplerini karşılayacak bir motorin deposu kurulması planlanmaktadır. Geçmiş dönem verilerine göre; motorin talepleri, ortalaması 0.125 talep/saat olan poisson sürecine uygun olarak gelmekte, gelen taleplerin miktarı (ton biriminden) da Üçgen Dağılıma uymaktadır (min:5 ton, mod:10 ton, maks:30 ton).

Bir motorin talebi alındığında, 1 astsubay tarafından talep bilgisayara kaydedilme ve çıktı alınarak dosyalanmakta (bu işlem süresi 10-20 dk. arası Düzgün dağılıma uymaktadır):

eğer depoda mevcut stok miktarı yeterli ise, 1 tanker kamyon, 1 uzman erbaş ve 1 sözleşmeli er ile gemiye motorin transferi yapılacak, her motorin transferinde gerekli bağlantıların takılması için sabit 1 saate ihtiyaç duyulacak, ayrıca her bir ton motorin için 15 dk. süre gerekecektir (örneğin, 20 ton motorin transferi için 60 dk. + 20 ton x 15 dk./ ton = 360 dk. = 6 saat gereklidir)

eğer depodaki mevcut motorin stoku yeterli değil ise, talep karşılanamayacaktır (kısmi talep karşılama olmayacak, ya talebin tamamı verilecek yada hiç verilmeyecek),

Her 7 günde bir, anlaşma yapılan akaryakıt firması tarafından depoya motorin tedariki yapılarak, motorin miktarı 250 ton (toplam kapasite) olan **referans stok** seviyesine tamamlanacaktır (örneğin depoda 180 ton motorin kalmış ise, 70 ton motorin tedariki yapılarak 250 tona tamamlanacak). Depoya motorin tedariki, belirli bir süre alan faaliyettir ve bunun için 1 astsubay ve 2 sözleşmeli er gerekmekte, depoya motorin transferi süresi alınan her bir ton motorin için 10 dk. sürmektedir.

Depoda; 24 saat esasına göre her daim 1 astsubay, 2 uzman erbaş ve 3 sözleşmeli er görevli olacak ve gemilere motorin transfer etmesi için 2 adet tanker kamyon tahsis edilecektir.

Görev yapan personelin her zaman tam mevcutla görev başında olacakları, ancak tanker kamyonların, arıza veya bakım nedeni ile, bazen hizmet dışı kalabilecekleri göz önüne alınmaktadır. Tanker kamyonların arıza/bakımları arasında geçen süreler (faal süreler), ortalaması 10 gün olan Üstel dağılıma uymakta, arıza onarımı/ bakımında geçen süreleri (faal olmayan süreler) ise 12-24 saat arası Düzgün Dağılıma uymaktadır.

Kurulması planlanan motorin deposu için, yukarıda açıklanan sistemin benzetim modeli kurularak;

- referans stok seviyesi, kurulumu yapılacak motorin tankı sayısı (planlanan: 5x50 ton = 250 ton),
- dışarıdan depoya motorin tedariki periyodu (planlanan : her 7 günde bir),
- depoda görevli personel sayısı (planlanan: 1 astsubay, 2 uzman erbaş, 3 sözleşmeli er) ve
- araç sayısının (planlanan: 2 tanker kamyonu) uygun olup olmayacağı analiz edilecektir.

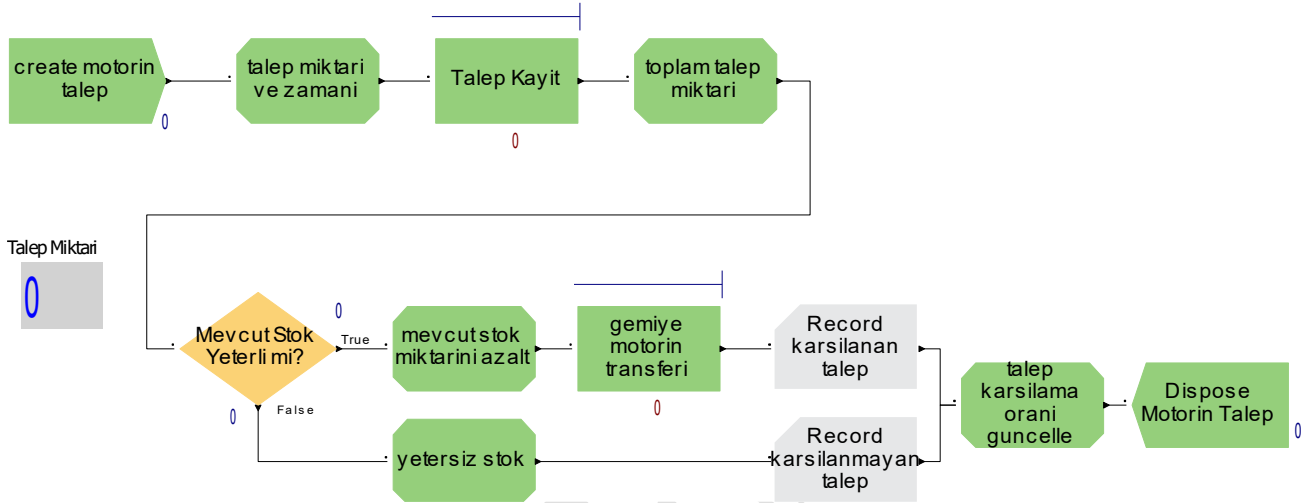
Gemilerin motorin taleplerinin ne oranda karşılandığı, ne kadar motorin talebinin karşılanamadığı ve talep karşılama süreleri gibi performans göstergeleri analizde dikkate alınacaktır (Taleplerin %90 oranında karşılanması, taleplerin en fazla 8 saat içinde karşılanması hedeflenmektedir).

Motorin Deposuna her biri 50 ton kapasiteli 5 adet motorin tankı kurulması planlanmıştır. Her bir tankın satın alma+kurulum maliyeti 30.000 TL'dir (Tankların bakımı ve iklimlendirme sistemleri için kullanılacak elektrik giderleri gibi değişken maliyetler dikkate alınmayacak). Ayrıca, depoya gelecek her bir tedarik için 1000 TL masraf vardır (alınan motorin miktarından bağımsız olarak sabit maliyet).

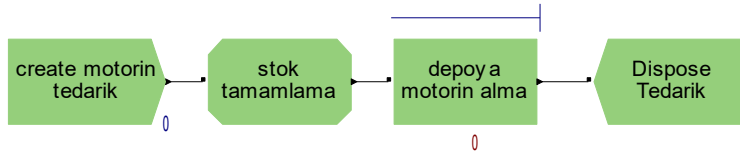
Tanker kamyonlarının; kapasitesi, depoda doldurulmaları ve depo ile gemiler arası gidiş gelişleri modelde göz önüne alınmayacaktır (Motorin bitince anlık, hemen dolacakları kabul edilmiştir).

Oluşturulan benzetim modeli Şekil 4-1’de sunulmuştur. Bu modelde, başlangıç ve bitişleri arasında birbirine bağlantı yapılmayan iki süreç (**Gemilerden Motorin Talebi** ve **Depoya Motorin Tedarik**) tek bir model alanında yer almaktadır. İlk süreçte, varlık olarak **gemilerden gelen talepler**, ikincisinde ise **depoya motorin tedarikler** kullanılmıştır. Aralarında bağlantı olmasa da, bir süreç içerisinde güncellenen sistem değişkenleri diğer süreç içinde de kullanıldığı için ve ortak **resource** kullanımları olduğu için birbirini etkilemektedir. Model alanına, grafikler hariç, diğer modülleri aşağıdaki gibi yerleştirerek ve bağlantıları yaparak örneğimizi takip etmeye başlayabilirsiniz (modülleri isterseniz farklı yerleştirebilirsiniz. Bağlantılar doğru yapıldığı sürece, modüllerin yerleşimi modele etki etmez)

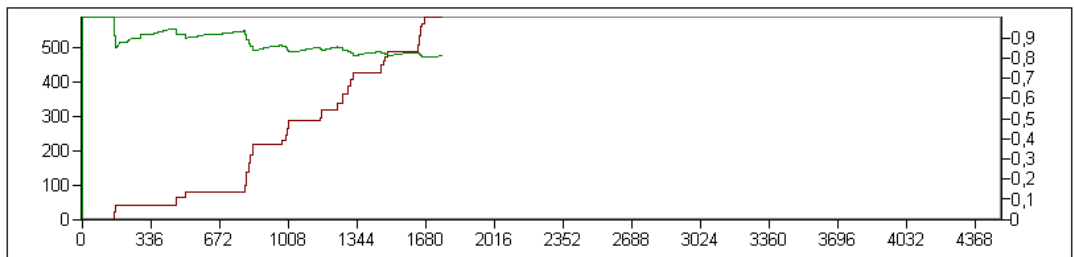
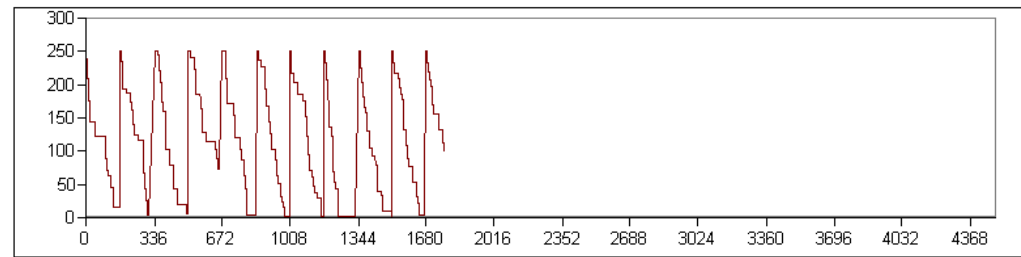
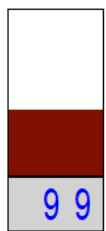
### Gemilerden Motorin Talebi



### Depoya Motorin Tedarik



Mevcut Stok (ton)



Toplam Karsılanamayan Talep (ton)

5 8 5 . 2

Talep Karsılama Oranı (%)

8 1 . 2

Şekil 4-1 Model Alanı Ekran Görüntüsü

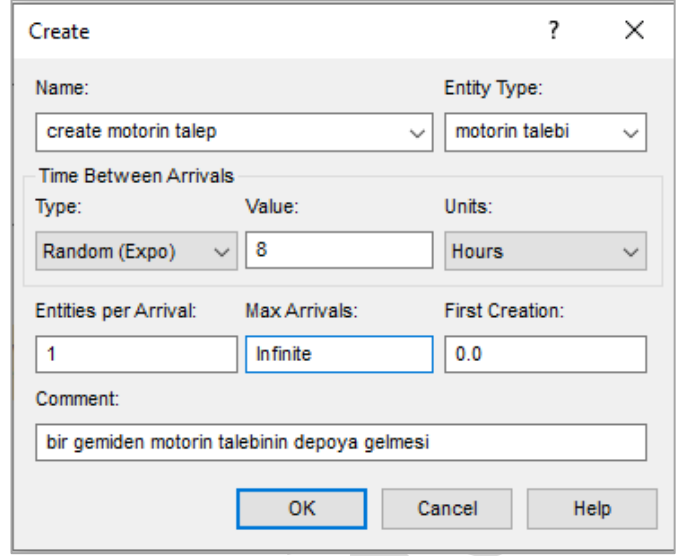
## Gemilerden Motorin Talebi Süreci

### Motorin Talebi Create Modülü

İlk süreç için yerleştirilen create modülü ile ilgili tanımlamalar Şekil 4-2’de görülmektedir.

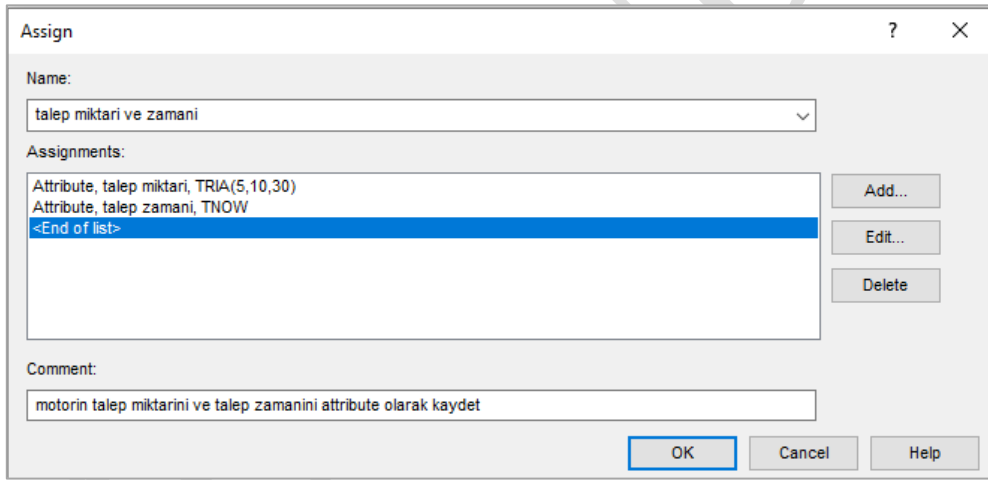
Bu modül, modelde belirtilen dağılıma uygun zaman aralıklarında, gemilerden motorin talebi üreterek modele göndermektedir.

Modelde, poisson sürecine uygun olarak saate 0.125 talep geldiği belirtilmektedir. Bu demektir ki, gelişler arası süreler, ortalaması 8 saat olan üstel (Random EXPO) dağılıma uyar (Birbirini takip eden her hangi iki talebin geliş zamanları arasında ort.8 saat olması beklenir).



Şekil 4-2

### Talep Miktarı ve Zamanı Assign Modülü



Şekil 4-3

Motorin talebi sisteme geldikten hemen sonra, ilk olarak bu talebinin ton biriminden miktarı belirlenir ve bir attribute değeri olarak kaydedilir. Ayrıca, talebin geliş zamanı da ayrı bir attribute değeri olarak kaydedilir. Bunun için; “Talep Miktarı ve Zamanı” adlı assign modülü içerisinde “Add..” düğmesi kullanılarak “talep miktarı” ve “talep zamanı” adlı iki ayrı attribute tanımlanır. Talep Miktarı için “New Value” alanına TRIA (5,10,30) ifadesi, Talep Zamanı için de “New Value” alanına TNOW ifadesi yazılır (Daha önceki örneklerde aynı şekilde varlıkların geliş zamanlarını kaydetmiştik).

Yapılan bu tanımlardan sonraki Assign modülü penceresinin görünümü Şekil 4-3’de gösterilmiştir.

### Talep Kayıt İşlemi (Process Modülü)

Gelen motorin talebinin kayıt ve dosyalanma işlemi için modele eklenen Process modülünde yapılan tanımlamalar Şekil 4-4’de gösterilmektedir. Resource ve işlem süresi tanımları, model açıklamasında belirtildiği gibi yapılmıştır (Process modülü penceresindeki alanların doldurulması ve tanımlamalar önceki örneklerde açıklandığından dolayı burada ayrıca detaya girilmeyecektir).

Daha önceki örneklerden farklı olarak, ilk defa “Priority” alanında farklı seçenek seçilmiştir.

Priority alanında; “High”, “Medium”, “Low” olmak üzere 3 seçenek mevcuttur ve değişiklik yapılmaz ise varsayılan olarak “Medium(2)” seçilidir.

Bu alanda yapılan seçim şöyle açıklanabilir:

Bu işlemde kullanılan bir kaynak (resource) eğer başka bir işlemde daha kullanılıyor ise bu kaynağın hangi işleme öncelik vereceği Priority seçeneğine bağlıdır. “Talep Kayıt” işlemi 1 astsubay tarafından yapılmaktadır. Bu astsubay ayrıca, depoya motorin transferi işleminde de görevlidir (ve o işlemin Priority seçeneği Medium olarak kalacaktır). Eğer iki işlem aynı zamana denk gelirse, astsubay önceliği Depoya Motorin Transferi işlemine verecek, Talep Kayıt işlemi bekleyecektir.

Şekil 4-4

Eğer bir kaynak eşit Priority seviyesinde iki işlemde kullanıyor ise, hangi işlem önce başladıysa onda işlemini bitirir ve diğer işleme geçebilir. İki işlemin (faaliyetin) oluş zamanları çakışmadığında zaten sorun yoktur. (Örneğin, Acil Servis modelinde, doktor iki ayrı işlemde Resource olarak tanımlıydı ve her iki işlemin Priority seçeneği Medium olarak, değiştirilmeden bırakılmıştı).

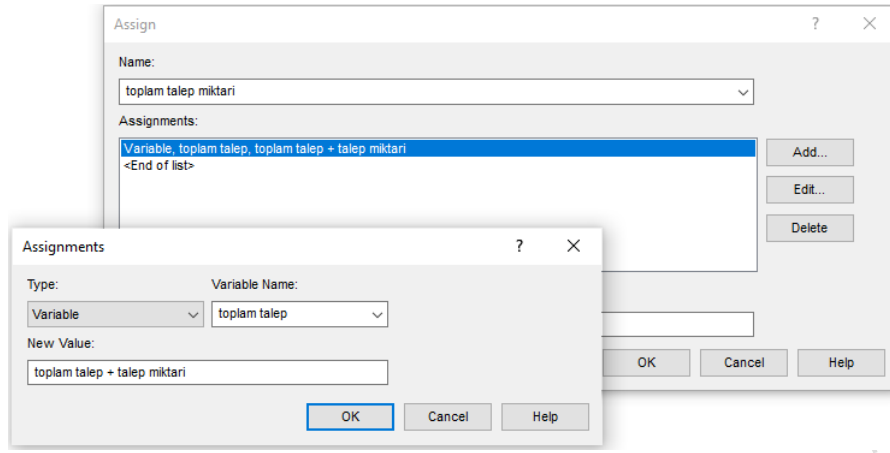
Ayrıca, yine önceki örneklerden farklı olarak, “Allocation” alanında “Non-Value Added” seçeneği seçilmiştir. Bunun anlamı, yapılacak olan “Talep Kayıt” işlemi Motorin Taleplerini karşılama sürecinde doğrudan katma değeri olan bir işlem olmadığına belirtilmesidir. Bu seçim, modelin çalışmasında farklı sonuçlara yol açabilecek bir seçim değildir. Sadece, Sonuç Raporunda işlem sürelerinin farklı satırlarda ayrı ayrı görüntülenmesi içindir.

### Toplam Talep Miktarı Assign Modülü

Model çalıştığı süre boyunca, depoya gelen toplam talep miktarını bir sistem değişkeni (variable) olarak tutmak için bir Assign modülü kullanılmıştır. Bir önceki assign modülünde attribute olarak kaydedilmiş olan talep miktarı, o zamana dek gerçekleşen toplam talep miktarına ilave edilir. Böylece, toplam talep miktarı her yeni talep ile birlikte güncellenir ve birikimli (kümülatif) bir şekilde bir değişken olarak tutulmuş olur.

Bunun için Şekil 4-5’de gösterildiği gibi Assign modülü içerisinde, “Add..” düğmesine basılır ve açılan yeni pencerede yeni bir değişken tanımlanır. Bu değişkenin alacağı değer için de, “New Value” alanına **toplam talep + talep miktarı** ifadesi yazılır. Bu ifadede yer alan, “talep miktarı” bir önceki assign modülünde attribute olarak tanımladığımız terimdir. Yani, yeni gelen talep ile birlikte, toplam talebimiz, o ana kadar gelen taleplerin toplamına yeni talep miktarını ekleyerek güncellenmiş olur.

Programlama dillerinde, bir değişkenin değeri artırılırken, o değişkenin mevcut değeri de formülde kullanılır. Örneğin;  $x = x + 5$  ifadesi, ilk bakıldığında matematiksel olarak yanlış görünebilir. Ancak, burada yapılan, eşitliğin sol tarafında bulunan değişkene yeni değer atamaktır. Eşitliğin sağ tarafında bulunan  $x$  terimin değeri, güncelleme yapılmadan önceki değerdir.



Şekil 4-5

### Mevcut Stok Yeterli mi? Karar Modülü

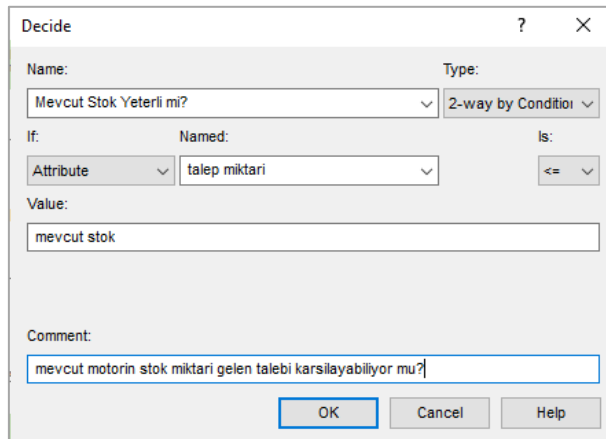
Yeni bir talep alındığında, bu talebi karşılayıp karşılayamayacağımızı kontrol etmemiz gerekir. Bunun için bir Karar Modülü kullanılır. Bu modül içerisinde; Şekil 4-6’da gösterildiği gibi, “Type” alanında “2-way By Condition” seçilir (yani karar bir koşula göre verilecek ve kararın sonucu Evet veya Hayır şeklinde olacaktır).

Bir motorin talebinin karşılanabilmesi için, depodaki mevcut motorin miktarının (mevcut stok) gelen talepten büyük olması gerekir. Gelen talep miktarını ilk Assign modülünde “talep miktarı” adıyla attribute olarak kaydetmiştik. Karşılaştırma yapılması için, model içerisinde dinamik olarak değişecek ve depodaki motorin miktarını tutacak yeni bir sistem değişkenine ihtiyaç vardır. Bu değişkenin adı “mevcut stok” olsun. Yani koşulumuz mantıksal ifade ile şöyledir:

eğer (talep miktarı  $\leq$  mevcut stok) ise Doğru (True), değil ise Yanlış (False)

Şimdi, bu koşulu Şekil 4-6’da gösterildiği gibi modele tanımlayabiliriz. Decide modülü penceresinde, “If” alanında “Attribute” seçilir ve “Named” alanındaki seçeneklerden daha önceden tanımlamış olduğumuz “talep miktarı” seçilir. “Is” alanında da küçük eşittir sembolü seçilir ve “Value” alanına “mevcut stok” ifadesi yazılır.

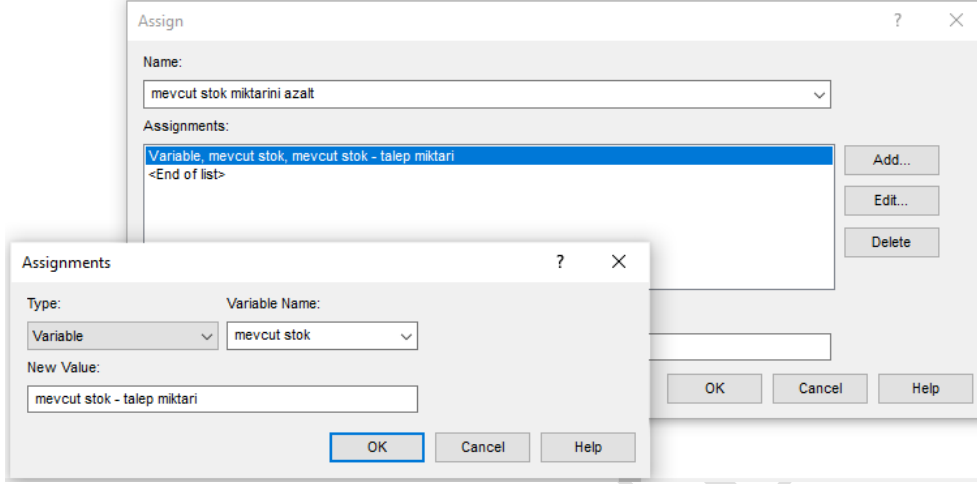
Böylece, bu modüle gelen talep eğer mantıksal koşul Doğru çıkarsa, modülün TRUE çıkışına gidecek, aksi takdirde modülün FALSE çıkışına gidecektir (Decide modülü şekli üzerinde TRUE çıkışı sağ köşede, FALSE çıkışı alt köşededir. Görüntü yakınlaştırılırsa daha kolay görünür).



Şekil 4-6

### Stok Miktarını Azaltma (Assign Modülü)

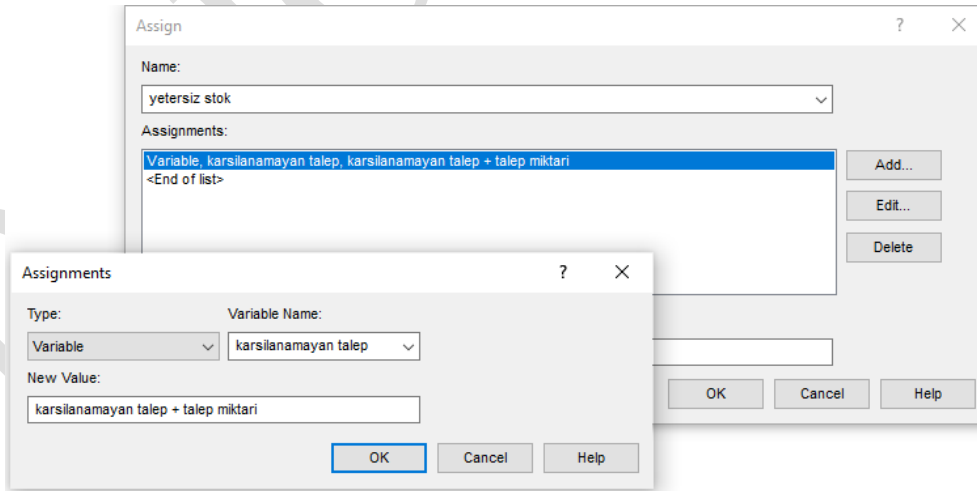
Eğer mevcut stok miktarı gelen talebi karşılamaya yetiyorsa (yani bir önceki Decide modülünün True çıkışı), stoktaki miktarı talep miktarı kadar azaltmak için bir Assign modülü kullanılarak, “mevcut stok” adlı sistem değişkeni güncellenir (bakınız Şekil 4-7). Assign modülü ile bir sistem değişkeninin değerinin değiştirilmesi ile ilgili daha önceki örneklerde açıklamalar yapılmıştı. Burada, ekran görüntüsünden, yapılmak istenen değişiklik anlaşılabilir.



Şekil 4-7

### Karşılanamayan Talep Miktarını Artırma (Assign Modülü)

Eğer mevcut stok miktarı yeterli değilse (Decide modülü False çıkışı), stok miktarı değişmez ancak, “karşılanamayan talep” adlı sistem değişkeni, talep miktarı kadar artırılır (bakınız Şekil 4-8). Bu değişken, talep karşılama durumu istatistiğini takip etmek için kullanılacaktır. Sonuç Raporunda, model tekrar süresi boyunca, gemilerden gelen taleplerin kaç tonluk miktarının karşılanamadığı görülebilecek, bu da kuracağımız sistemin bir performans göstergesi olarak, karara destek olacaktır.



Şekil 4-8

Assign modülleri içerisinde yukarıda gösterilen tanımlamalar yapıldıktan sonra, bu değişkenlerin Variable modülü içerisinde de tanımlanması gerekmektedir. “mevcut stok” adlı değişkenin ilk değeri (initial value sütunu) 250 olacak, “toplam talep” ve “karşılanamayan talep” değişkenlerin ilk değerleri ise 0 olacaktır (Tablo formatında olan Variable modülüne nasıl erişileceği daha önceki örneklerde açıklanmıştır).



## Gemiye Motorin Transferi (Process Modülü)

Decide modülünün True çıkışında, mevcut stok miktarını Assign modülü ile azalttıktan sonra, Gemiye Motorin Transferi işlemini için process modülü kullanılır. Bu modül için, model açıklamasına uygun olarak, Şekil 4-9'da gösterildiği gibi tanımlamalar ve seçimler yapılır.

Priority alanında "High(1)" seçilmiştir, böylece kaynakların kullanımı bakımından diğer işlemlere göre önceliği vardır. Aslında, uzman erbaş ve tanker kamyonlar sadece bu işlem için kullanılan kaynaklar olduğundan bu kaynaklara etki eden bir durum yoktur ancak sözleşmeli er "depoya motorin alma" işleminde de kullanılmaktadır ve sözleşmeli erler önceliği bu işleme verecektir (çakışma olması durumunda).

Şekil 4-9

## Record Modülleri

Modelde, 2 adet Record modülü kullanılmıştır. Biri, karşılanan talebin karşılanma süresini (ilk geldiği andan beri geçen zamanı) kaydetmek, diğeri ise karşılanamayan talep sayısını (talep adedi olarak) kaydetmek için kullanılmaktadır. Şekil 4-10'da, karşılanan bir talebin "talebi karşılama süresi"nin kaydedilmesi için yapılan tanımlamalar görülmektedir.

Not : Sonuç raporunda istenen bir istatistik için iki yöntem kullanılabilir: 1. değişken (variable) tanımlamak, 2. record modülü kullanmak.

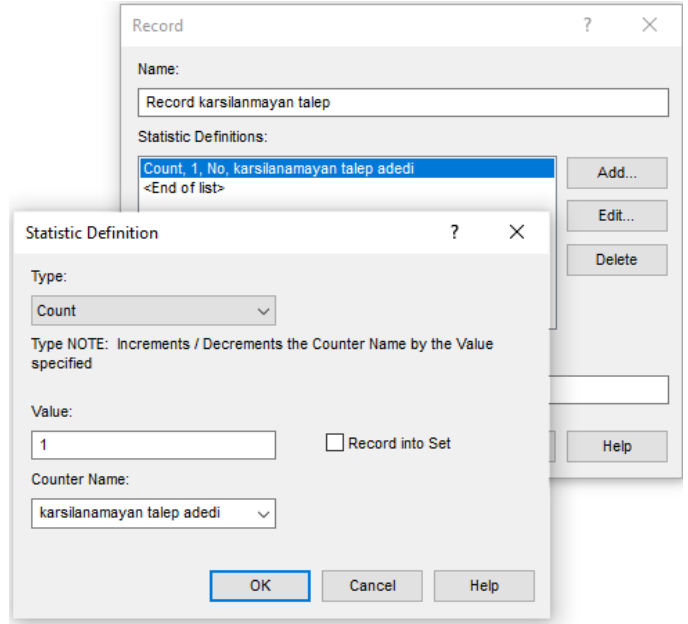
Şekil 4-10

Record modülünde tanımlanan bir kayıt için "Tally Name" alanına yazılan ifade Sonuç Raporunda bir satır olarak görülür (Raporun son bölümünde). Tanımlanan bir sistem değişkeni (variable) var ise, ona ait istatistiklerde de Sonuç Raporunun bu bölümünde yer alır. Ancak, her değişkeni sonuç raporunda görülmesi istenmeyebilir. Bunun için, görülmekörmek istenen değişken için, değişken tablo görünümüne "Report Statistics" sütununun işaretlenmesi gerekmektedir (bakınız Şekil 4-11).

	Name	Rows	Columns	Data Type	Clear Option	File Name	Initial Values	Report Statistics	Comment
1	ref stok			Real	System		1 rows	<input type="checkbox"/>	
2	mevcut stok			Real	System		1 rows	<input checked="" type="checkbox"/>	
3	karşılanamayan talep			Real	System		0 rows	<input checked="" type="checkbox"/>	
4	toplam talep			Real	System		0 rows	<input checked="" type="checkbox"/>	

Şekil 4-11

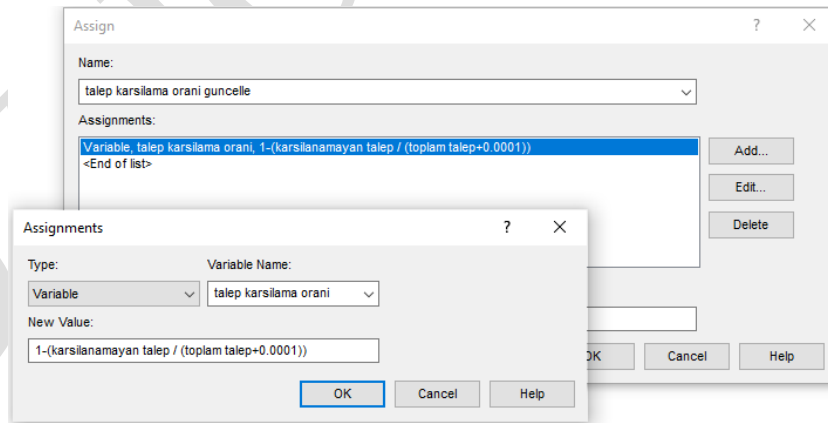
Modeldeki diğer Record modülü olan ve karşılanamayan taleplerin sayısını kaydeden modüle ilişkin tanımlar da Şekil 4-12’de görülmektedir. Bu modülde, diğerinden farklı olarak, belirli bir zaman aralığının değil bu modüle gelen varlıkların (yani stok yetersizliği nedeniyle karşılanamamış motorin talepleri) sayısının kaydı tutulmaktadır. Bunun için, *Statistics Definitions* başlıklı ikinci pencerede, *Type* alanında “Count” (yani sayı) seçilir, *Value* alanında da 1 değeri olur (her varlık için 1 artır). Counter Name alanına da Sonuç Raporunda görmek istediğimiz ifade yazılır.



Şekil 4-12

### Talep Karşılanma Oranı (Assign Modülü)

Gemilerden motorin talebi ile ilgili süreç için son olarak, “talep karşılama oranı” adı verdiğimiz ve gelen taleplerin % olarak ne kadarını karşılayabildiğimizi gösteren bir değişkenin değerinin güncellendiği bir Assign modülü yer almaktadır. Bu değişken Şekil 4-13’de gösterildiği gibi tanımlanmıştır. “talep karşılama oranı” değişkeni bir yüzdelik değeri ifade etmektedir ve şu formül ile bulunur:  $1 - (\text{karsilanamayan talep} / \text{toplam talep})$ . Ancak, bu hali ile ARENA programında “sıfıra bölüm” hatası vermektedir. Bunun için, paydaya sonucu neredeyse hiç etkilemeyecek çok küçük bir değer eklenerek hata vermesi engellenmiştir:  $1 - [\text{karsilanamayan talep} / (\text{toplam talep} + 0.0001)]$ .



Şekil 4-13

Tanımlanan bu değişkene ait istatistikleri de eğer Variable modülündeki tabloda “Report Statistics” sütunundaki hücresi işaretlenmiş ise Sonuç Raporunda görülebilecektir.

Not : Modelde assign modülleri içerisinde kullandığınız değişkenlerin, Variable modülünde bir satır olarak, adı doğru yazılmış olarak yer aldığından emin olunuz.

“Dispose Motorin Talep” adı verilen Dispose modülü de gelen taleplerin sistemden çıkışı için eklenir.

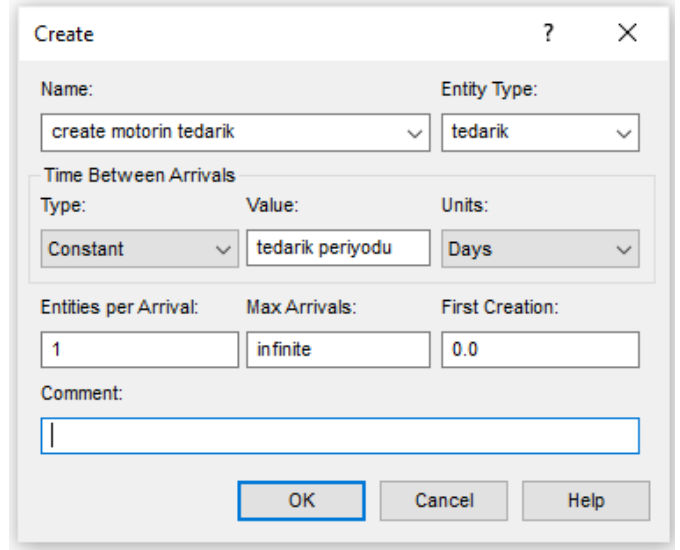
## Gemilerden Motorin Talebi Süreci

### Motorin Tedarik Create Modülü

İkinci süreç için yerleştirilen create modülü tanımlamaları Şekil 4-14’de görülmektedir.

Bu modül, model açıklamasında belirtildiği gibi, sabit aralıklar ile sisteme “tedarik” adı verilen bir varlığın girmesini sağlamaktadır (her tedarik varlığı depoyu doldurmak için dışarıdan tedarik gelişi demektir).

Sabit zaman aralıklarında geliş için Type alanında “Constant” seçilir. Bu modülde, daha öncekilerden farklı olarak, Value alanında bir sayısal değer bulunmamaktadır.



Şekil 4-14

Şekil 4-14’de görüldüğü gibi, Value alanında yazan “tedarik periyodu” ifadesi bir değişken adıdır. Bu değişkenin Variable modülünde tanımlanması ve “Initial Value” değerinin modelde belirtildiği gibi “7” yapılması gerekir (Units alanında “Days” seçilir). Bu değişken, model çalışması boyunca değişmeyecektir. Eğer modelimizi farklı tedarik periyotları ile çalıştırmak istersek (örneğin 5 günde bir tedarik), Variable modülündeki tabloda “initial value” değeri de ona göre değiştirilir.

Model açıklamasında, depoya 50’şer ton kapasiteli 5 motorin tankı kurulacağı belirtilmektedir (her tedarikte toplam 250 tonluk kapasite tamamlanacak). Ancak, modelde farklı kapasite alternatiflerini de denemek isteyebiliriz (örneğin 4 tank x 50 = 200 ton). Bunun için stok kapasitesini modelde bir sayı olarak kullanmak yerine, “ref stok” adlı yeni bir değişken tanımlanmıştır (ref stok: motorinin tamamlanacağı referans stok seviyesi). Bu değişkeni de Variable modülünde bir satır olarak tanımlayarak, “initial value” sütununa 250 değeri yazarız. Modelimizi farklı bir depo kapasitesi ile çalıştırmak istediğimizde de buradan değişiklik yapabiliriz.

Her bir tedarik için sabit 1000 TL masraf çıkmaktadır. Yani, daha sık periyotlar ile tedarik yaparsak (örneğin 2 günde 1), depoda stok bitiş riskini azaltabiliriz ancak bunun maliyeti de yüksek olur. Daha uzun periyotlarda tedarik yaparsak (örneğin 10 günde 1), maliyet düşer ancak talepleri karşılamada sorun yaşayabiliriz. Bir performans göstergesi olarak tedarik maliyetini de takip edebilmek için “toplam tedarik maliyeti” adında yeni bir değişkeni Variable modülünde tanımlayın (initial value=0).

Daha önceki değişkenler ile birlikte, Variable modülü tablosu Şekil 4-15’deki gibi olacaktır. Dikkat edilirse, bazı değişkenlerin “initial value” hücresi boştur. (Boş olması otomatikman sıfır demektir)

	Name	Rows	Columns	Data Type	Clear Option	File Name	Initial Values	Report Statistics	Comment
1	ref stok			Real	System		1 rows	<input type="checkbox"/>	
2	mevcut stok			Real	System		1 rows	<input checked="" type="checkbox"/>	
3	karsılanamayan talep			Real	System		0 rows	<input checked="" type="checkbox"/>	
4	toplam talep			Real	System		0 rows	<input checked="" type="checkbox"/>	
5	talep karşılama oranı			Real	System		0 rows	<input checked="" type="checkbox"/>	
6	tedarik periyodu			Real	System		1 rows	<input type="checkbox"/>	
7	toplam tedarik maliyeti			Real	System		0 rows	<input checked="" type="checkbox"/>	

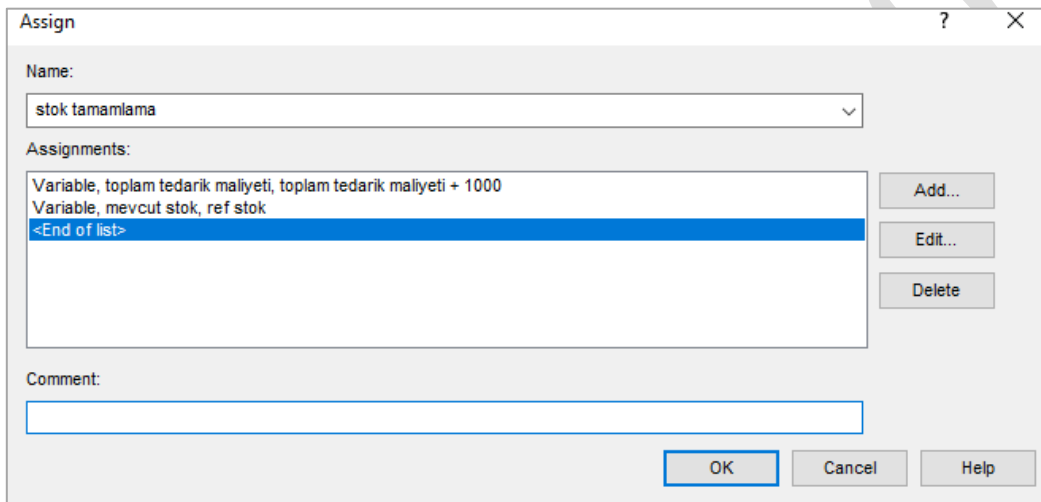
Double-click here to add a new row.

Şekil 4-15

Aslında, yeni bir değişken tanımlamadan ve Create modülünün Value alanına doğrudan 7 yazarak da modeli çalıştırabilirdik, farklı bir tedarik periyodu denemek istersek de yine buradan yeni bir değer yazarak değiştirebilirdik. Ancak, model ile ilgili bazı değişkenlere ait değerlerin tek bir yerden belirlenmesi takip kolaylığı açısından daha uygundur. Bu husus, özellikle programlama dillerinde kod içerisinde, sayısal değer olmaması ve denklemlerde parametre kullanılması ilkesi ile de uyumludur.

### Stok Tamamlama (Assign Modülü)

Motorin Tedarik create modülünden sonra, ilk olarak bir Assign modülü içerisinde, “mevcut stok” ve “toplam tedarik maliyeti” değişkenleri güncellenir. Her yeni motorin tedarikinde depo tam kapasitesine kadar doldurulmaktadır. Eğer depo boş ise 250 ton alınmakta, eğer depoda 120 ton motorin kalmış ise 130 ton alınmaktadır (her durumda motorin stoku 250 tona tamamlanmaktadır).



Şekil 4-16

Bunun için; Şekil 4-16’da gösterildiği gibi iki değişkene ait tanımlamalar yapılır. Toplam tedarik maliyeti değişkeni için New Value alanına **toplam tedarik maliyeti + 1000** ifadesi yazılır (her yeni tedarik sabit 1000 TL masraf), mevcut stok değişkeni için New Value alanına **ref stok** ifadesi yazılır (her yeni tedarik mevcut stok miktarını referans stok değerine getiriyor).

Bu noktada, aklımıza “Neden alınan motorin miktarına göre bir maliyeti dikkate alınmadığı” sorusu gelebilir. Tabiki, gerçek durumda böyle bir masraf çıkacaktır. (örneğin, 100 ton motorin, 20 ton motorinden 5 kat daha fazla maliyetli olacaktır). Ancak, tedarik edilecek motorine ödenecek ücret bizim kurguladığımız depo sisteminden etkilenmeyecektir.

Gemilerden gelecek toplam motorin talebi uzun vadede kurgulanacak her sistem için aynı kabul edilebilir ve böylece motorine ödenecek ücret de farklı depo modelleri için sabit kabul edilebilir. Bu sebepten, motorin maliyetini modelde bir performans göstergesi olarak bu modelde dikkate almıyor ve hesaplamıyoruz.

Eğer modelimizde, motorinin birim fiyatının bir seferde alınan motorin miktarına göre değiştiği, farklı firmalardan farklı zamanlarda alınan motorinin fiyatının da değişiklik gösterdiği gibi durumlar söz konusu olsaydı modelimizi ona göre tasarlayabilir ve motorin masraflarını da dikkate alabilirdik.

## Depoya Motorin Alma (Process Modülü)

Depoya motorin alma işlemi için modele eklenen process modülü içerisinde yapılan tanımlamalar ve seçimler Şekil 4-17’de görülmektedir.

Model açıklamasında belirtildiği gibi, resource olarak 1 astsubay ve 1 sözleşmeli er tanımlanmış, işlem süresi için Expression seçilerek:

$10 * (\text{ref stok} - \text{mevcut stok})$  ifadesi yazılmıştır.

(Alınan motorin miktarı, referans stok olan 250 tondan ne kadar eksik olduğuna bağlıdır ve 10 dakikada 1 ton alınabilmektedir)

Priority alanında ise, “Medium(2)” seçilmiştir. Daha önce tanımladığımız işlemlerden birisi “Low”, diğeri de “High” öncelik dereceliydi.

Şekil 4-17

“Dispose Motorin Tedarik” adlı Dispose modülünde ilave bir tanımlama yapmaya gerek yoktur.

Ancak, modelimizde tanımlamamız gereken iki husus daha vardır:

- kaynakların miktarlarının modele girilmesi ve
- tanker kamyonlarının arızalanma durumlarının tanımlanması.

Bunun için önce, Resource modülü açılır (sayfanın alt kısmında tüm kaynakların satır satır listelendiği bir tablo). Bu tabloda, kaynakların miktarları “Capacity” sütunundaki hücrelere yazılır.

Daha sonra, tanker kamyon satırında “Failures” sütununa tıklayarak “kamyon arıza” adında ve “Preempt” tipinde bir arıza tanımlanır (Önceki bir örnekte, resource (kaynak) arıza durumu ile ilgili nasıl tanımlama yapılacağı detaylı bir şekilde açıklanmıştı).

Resource modülünde yer alan satırlar ve tanker kamyon için 1 arıza tanımlanmış olduğunu gösteren ekran görüntüsü: Şekil 4-18.a ve Failure modülünde tanker kamyon için tanımlanan arızaya ait detaylar 4-18.b üzerinde görülebilir.

	Name	Type	Capacity	Busy / Hour	Idle / Hour	Per Use	StateSet Name	Failures	Report Statistics	Comment
1	soz er	Fixed Capacity	3	0.0	0.0	0.0		0 rows	<input checked="" type="checkbox"/>	
2	astsubay	Fixed Capacity	1	0.0	0.0	0.0		0 rows	<input checked="" type="checkbox"/>	
3	uzman erbas	Fixed Capacity	2	0.0	0.0	0.0		0 rows	<input checked="" type="checkbox"/>	
4	tanker kamyon	Fixed Capacity	2	0.0	0.0	0.0		1 rows	<input checked="" type="checkbox"/>	

Şekil 4-18.a

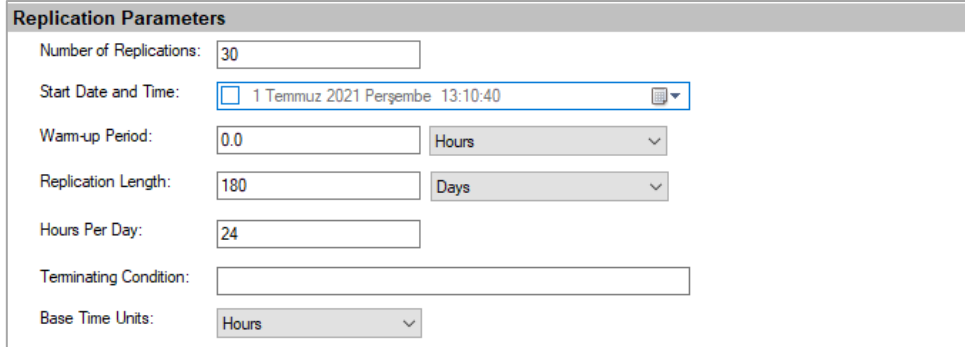
	Name	Type	Up Time	Up Time Units	Down Time	Down Time Units	Uptime in this State only	Comment
1	kamyon arıza	Time	EXPO( 10 )	Days	UNIF(12 , 24 )	Hours		

Double-click here to add a new row.

Şekil 4-18.b

## Model Çalıştırma Parametrelerinin Belirlenmesi ve Modelin Çalıştırılması

Run menüsünde Set up düğmesine basarak, Run Setup penceresini açın ve “Replication Parameters” sekmesinden Şekil 4-19’da görüldüğü gibi parametreleri girin. (her biri 180 günlük 30 tekrar olacak ve Base Time Units: Hours)



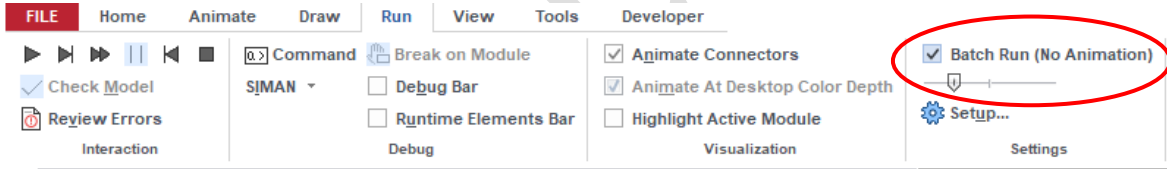
The image shows the 'Replication Parameters' dialog box in Arena software. It contains the following fields and settings:

- Number of Replications: 30
- Start Date and Time: 1 Temmuz 2021 Perşembe 13:10:40
- Warm-up Period: 0.0 Hours
- Replication Length: 180 Days
- Hours Per Day: 24
- Terminating Condition: (empty field)
- Base Time Units: Hours

Şekil 4-19

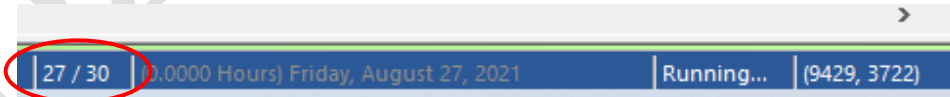
Yukarıda belirtildiği şekilde model alanı yerleşimi, modüllerin içerisinde gerekli tanımları yaptıktan ve Run Setup parametrelerini girdikten sonra, modeli çalıştırabilirsiniz.

Benzetimin tamamlanması biraz uzun zaman alabilir (bunun başlıca nedeni model ekranında varlıkların modüller arasında hareket etmesi yani bir animasyon olmasıdır). Modelin çalışmasını, Run Menüsünden hızlandırıp, yavaşlatıp takip edebilirsiniz. Ancak, ilk tekrar tamamlandıktan sonra, sonuçları daha hızlı görebilmek için “Batch Run (No Animation)” (yani toplu çalıştırma) seçeneğine tıklayabilirsiniz. (bakınız Şekil 4-20).



Şekil 4-20

“Batch Run” seçeneğini seçmeniz durumunda da, benzetim hemen sonuçlanmaz ama çok daha hızlı ilerler (sadece ekranda hareket eden varlıklar görmezsiniz). Biz modelimizde 30 tekrar olacağını belirtmiştik. Programın hangi tekrarı çalıştırdığını Şekil 4-21’deki gibi ARENA programı penceresinin sağ alt kısmından takip edebilirsiniz. (Aşağıdaki şekilde 30 tekrarın 27’ncisi çalışırken görülmektedir)



Şekil 4-21

Modeli çalıştırdığınızda ekranda hata görürseniz, açıklanan hususlarda yanlış tanımlama yapmışınız demektir. Bu durumda, ekranda görülen hata mesajında ne yazıyorsa ona göre modele geri dönüp düzeltme yapabilirsiniz.

Uyarı : Daha önceki örneklerde de belirtildiği gibi, işlem sürelerinde zaman birimlerinin doğru seçildiğinden emin olunuz.

## Grafik ve Diğer Göstergelerin Model Alanına Eklenmesi

### a. Depodaki Motorin Miktarı ile İlgili Grafik ve Göstergeler

Modelimizde, depodaki anlık motorin miktarını gösteren bir değişkenimiz vardır : “mevcut stok”

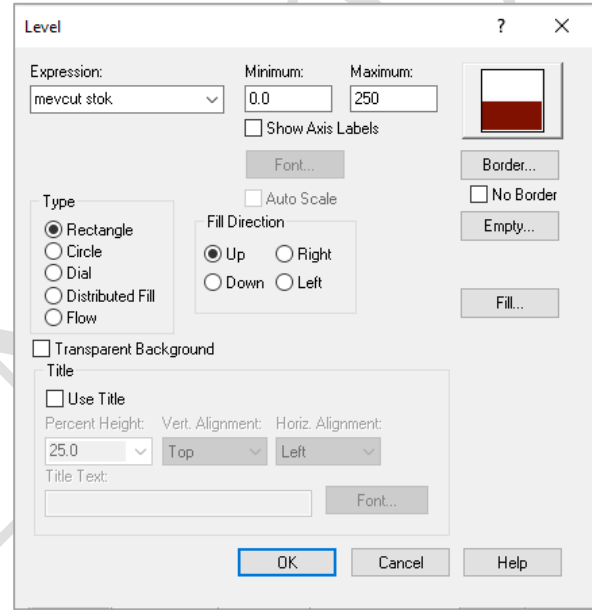
Bu değişkeni kullanarak 3 farklı biçimde, depodaki motorin miktarını model alanında gösterebilir ve model çalışırken dinamik olarak değişimleri gözlemleyebiliriz. Bunun için, Animate menüsünde yer alan 3 araç kullanılacaktır.

Variable ve Level araçları; mevcut stok değişkeninin anlık değerini gösterecek, Plot ise zamana göre grafiği sergileyecektir. Variable aracı ile sayısal (dijital bir ekran gibi), Level aracı ise sütun şeklinde bir seviye göstergesidir (farklı şekiller de seçeneklerde mevcuttur).

Variable aracı ile daha önce bir uygulama yapıldığından ayrıca detaylı açıklanmayacaktır. Level gösterge aracı için yapılan tanımlamalar Şekil 4-22’de gösterilmiştir (Farklı renk seçenekleri deneyebilirsiniz).

Grafik aracı ile ilgili de daha önceki örneklerde bir uygulama yapıldığından burada ayrıca detaylı açıklanmayacaktır.

Burada önemli olan nokta, her üç araç için yapılacak tanımlamalarda, depodaki motorin miktarı için tanımlamış olduğumuz “mevcut stok” değişkeninin araçlarda veri kaynağı olarak doğru bir şekilde belirlenmesidir.

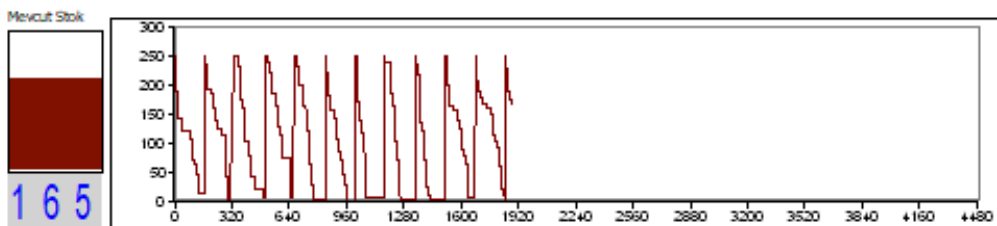


Şekil 4-22

Grafikte ayrıca, eksenlerin limitlerini uygun şekilde belirlemek görsellik açısından önemlidir. Bunun için, x (yatay) ekseninde üst limit olarak, tekrar süremiz olan 180 gün x 24 saat = 4320 belirlenmesi, y (dikey) eksen için de üst limit olarak da toplam kapasitemiz olan 250 belirlenmesi uygun olacaktır.

Model ekranına yukarıda belirtilen grafik ve göstergeleri ekledikten ve gerekli tanımları yaptıktan sonra, modelinizi tekrar çalıştırın (“batch run” işaretli olmadan). Grafikteki çizgilerin ve göstergelerdeki değerlerin dinamik olarak değiştiğini görmeniz gerekmektedir. Model çalışırken alınan bir örnek ekran görüntüsü Şekil 4-23’de verilmiştir.

Grafikte, depodaki motorin miktarının (mevcut stok), kademe kademe düştüğü ve belirli aralıktan gelen tedarik ile tekrar 250 ton olan referans seviyeye çıktığı, soldaki göstergelerde de motorin seviyesi ve değeri anlık olarak görülmektedir.



Şekil 4-23

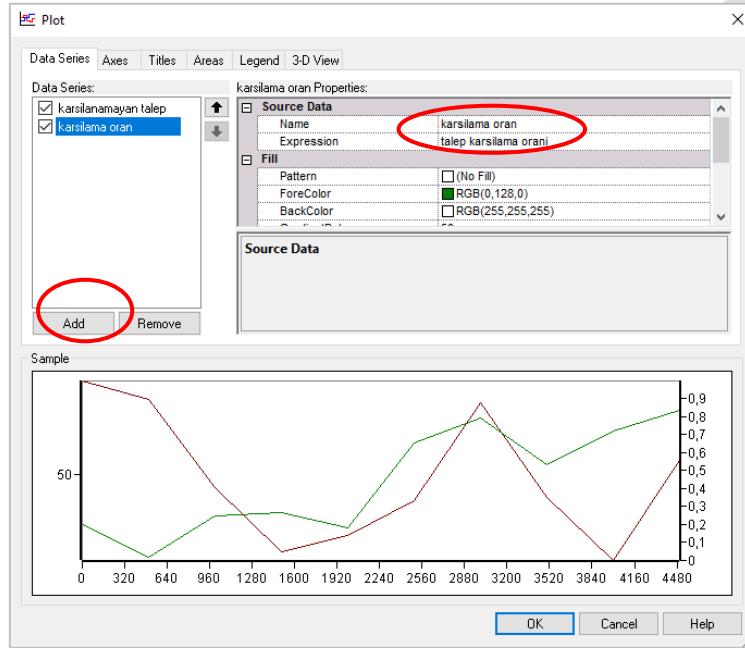


## b. Taleplerin Karşılama Durumu ile İlgili Grafik ve Göstergeler

Modelimizde, karşılanamayan talep miktarı ve talep karşılama oranı ayrı ayrı iki değişken ile dinamik olarak değer almaktadır. Bu değişkenleri kullanarak ikinci grup gösterge ve grafikleri oluşturabiliriz.

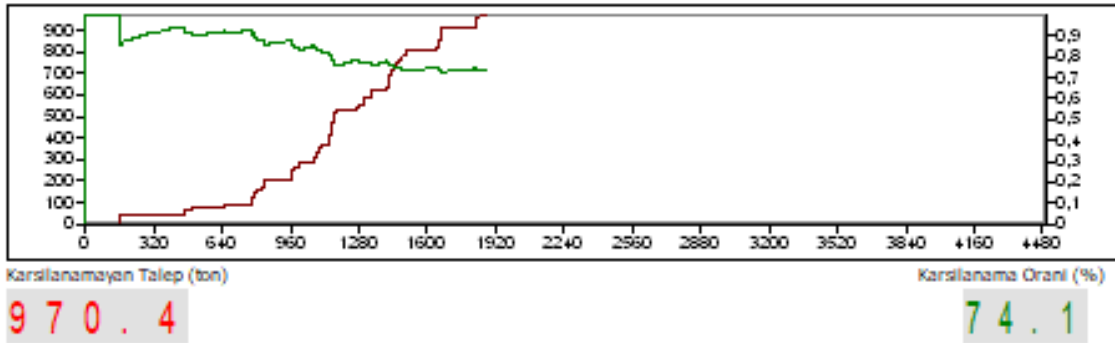
Daha önceki grafik örneklerinden farklı olarak, bu grafikte, iki değişkeni aynı grafikte göstereceğiz.

Karşılanamayan talep değişkeninin alacağı değerler grafiğin sol dikey ekseninde, talep karşılama oranı ise sağ dikey ekseninde değerler alacaktır. Bunun için, öncelikle, “karsilanamayan talep” adlı değişkene göre, plot aracı ile grafiğimizi oluşturabiliriz. Daha sonra, aynı grafik tanımlama penceresinde, “Data Series” sekmesinde “Add..” düğmesine basarak, ikinci veri setini (“karsilanamayan talep oranı” değişkeninin alacağı değerleri) grafiğe tanımlarız (bakınız Şekil 4-24)



Şekil 4-24

Grafiğin alt kısmında da, bu iki değişkenin sayısal olarak anlık değerlerini gösteren bir “Variable” gösterge aracı kullanabiliriz. Bu işlemleri yaptıktan sonra, modeli tekrar çalıştırıp grafikte değerlerin dinamik olarak değişip değişmediği kontrol edilebilir. İkinci grup grafik ve göstergelerin modelin bir çalışma anındaki görünümü örnek olarak Şekil 4-25’de gösterilmiştir.



Şekil 4-25

Grafikte, karşılanamayan talep miktarının kademeli olarak arttığı (sol taraftaki dikey eksene göre değer almakta) ve talep karşılama oranının da yaklaşık %74 seviyelerinde bir değere oturmaya başladığı görülmektedir.



## Sonuçlar

Modelin çalışması tamamlandığında Sonuç raporu görüntülenebilir. Daha önceki örnekte, sonuç raporunun içeriği, kısımları ve istatistiklerin nasıl anlaşılması gerektiği konusunda detaylı açıklama yapılmıştır. Bu örnekte, sonuç raporunun her kısmı hakkında detaylı açıklama yapılmayacak, sadece belirli kısımlar hakkında inceleme yapılacaktır.

### Sonuç Raporu Sayfa 2 (Entity)

Total Time	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
motorin talebi	4.9191	0,12	4.2550	5.4589	0.1667	51.0094
tedarik	0.2183	0,05	0.06977873	0.7254	0.00	11.2470
Number In	Average	Half Width	Minimum Average	Maximum Average		
motorin talebi	537.10	10,22	485.00	609.00		
tedarik	26.0000	0,00	26.0000	26.0000		

Gelen bir motorin talebi ortalama 4.9 saat içerisinde sonuçlanmış ve sistemden çıkış yapmış (Dikkat: bu süre taleplerin tümü içindir. Karşılana taleplerin bitiş süresini bu istatistikte ayırt edemeyiz). 180 günlük süre içerisinde ortalama 537 adet motorin talebi gelmiş, 26 kere de depoya motorin tedariki yapılmış (her 7 günde bir sabit zaman aralıklarında tedarik olduğu için).

### Sonuç Raporu Sayfa 4 (Resource)

Instantaneous Utilization	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
astsubay	0.03215819	0,00	0.02886708	0.03566469	0.00	1.0000
soz er	0.1600	0,00	0.1525	0.1668	0.00	1.0000
tanker kamyon	0.2270	0,00	0.2194	0.2331	0.00	1.0000
uzman erbas	0.2389	0,00	0.2282	0.2482	0.00	1.0000

En meşgul olan kaynak gemilere transferde görevli tanker kamyonlar ve uzman erbaşlar olarak ortaya çıkmış. Diğer kaynakların doluluk oranı hayli düşük. Bu durum, bize depoda görevlendirilmek istenen personel ve araç sayısının yeterli olduğunu hatta belki de daha az miktarda personel ve araç ile faaliyetlerin yapılabileceği konusunda fikir veriyor.

### Sonuç Raporu Sayfa 6 (User Specified)

#### Tally

Interval	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
talebi karşılama süresi	6.2402	0,11	5.7254	6.7529	2.4818	51.0094

Record modülü ile kayıt altına aldığımız bir istatistik olan talep karşılama süresi, bize karşılanabilen bir talebin ortalama karşılanma süresini vermektedir. Bu istatistik, Entity sayfasında görülen istatistikten daha anlamlıdır. Burada, ortalamada 6.24 saatte bir talebin karşılanabildiğini görüyoruz. Hedeflenen sürenin altında. (Maximum Average değeri de 6.75 saat)

## Time Persistent

Variable	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
karsilanamayan talep	958.36	76.63	520.39	1462.18	0.00	2498.49
mevcut stok	106.83	2.48	95.1927	119.74	0.00773979	250.00
talep karsilama orani	0.7798	0.01	0.6948	0.8614	0.00	1.0000
tedarik maliyeti	13361.11	0.00	13361.11	13361.11	0.00	26000.00
toplam talep	4067.75	89.13	3509.86	4644.79	0.00	8863.61

180 gün süresi sonunda toplam karşılanamayan talep, ortalama 958 ton olarak gerçekleşmiş,

Depodaki stok miktarı ortalama 106.83 ton olarak gerçekleşmiş olup toplam kapasite olan 250 tonun yarısından çok daha azdır. Yani genelde stok, tedarik gelemenden önce bitmiş gibi görünüyor,

Talep karşılama oranı, ortalama yaklaşık %78 olarak gerçekleşmiş ve güven aralığı da oldukça dardır (kararlı bir sonuç göstergesi). Ancak, bu oran modelin açıklamasında belirtilen %90'lık oranın hayli altında, yükseltilmesi için planlanan sistemde bir değişiklik yapılması gerekebilir.

Yukarıda görünen istatistiklere bakıldığında, hedeflediğimiz azami 8 saatlik talep karşılama süresi hedefimizi karşıladığımızı ancak talep karşılama oranının hedeflenenden düşük olduğu, personel ve araçların kullanım oranlarının da düşük olduğu görülmektedir. Bu aşamada, yapılabilecek şey, modelin farklı alternatifler ile tekrar çalıştırılması ve elde edilen sonuçların ne kadar iyileştirme sağlayıp sağlamadığının görülmesidir.

Bunun için ilk önce, tedarik periyodunu 5 gün ve 6 gün olarak iki farklı alternatifle çalıştırılalım (tedarik periyodu adlı değişkenin değerini modül içerisinde değiştirerek).

## Sonuçlar (tedarik periyodu = 6 gün alternatifi)

### Tally

Interval	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
talebi karsilama suresi	6.2935	0,13	5.7532	7.2859	2.4818	56.5684

## Time Persistent

Variable	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
karsilanamayan talep	593.47	71.44	170.55	1032.62	0.00	1822.28
mevcut stok	122.50	2.09	110.03	132.21	0.00606496	250.00
talep karsilama orani	0.8634	0.02	0.7667	0.9545	0.00	1.0000
tedarik maliyeti	15500.00	0.00	15500.00	15500.00	0.00	31000.00
toplam talep	4070.35	89.83	3509.86	4644.79	0.00	8863.61

Bu sonuçlara göre, depoya motorin tedarikini 6 günde bir yaparsak, talep karşılama oranı ort %86'ya yükselmekte ancak tedarik maliyetimiz de artmaktadır. Talep karşılama süresinde önemli bir değişiklik yok, çünkü gemiye motorin tedarikini doğrudan etkileyen bir değişiklik yapmadık.

## Sonuçlar (tedarik periyodu = 5 gün alternatifi)

### Time Persistent

Variable	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
karsılanamayan talep	300.63	48,20	67.2013	588.80	0.00	940.81
mevcut stok	140.41	1,94	127.79	147.74	0.01838813	250.00
talep karşılama oranı	0.9295	0,01	0.8629	0.9796	0.00	1.0000
tedarik maliyeti	18500.00	0,00	18500.00	18500.00	0.00	37000.00
toplam talep	4064.47	87,73	3509.86	4644.79	0.00	8863.61

Bu sonuçlara göre, depoya motorin tedarikini 5 günde bir yaparsak, talep karşılama oranı ort %92'ye yükselmekte ancak tedarik maliyetimiz daha da artmaktadır. (Talep karşılama süresi istatistikleri gösterilmemiştir ancak önemli bir değişiklik yoktur).

Bunun yanında, farklı depo kapasitesi alternatifleri de denenebilir. Her bir tank 50 ton olduğu ve bir tankın satın alma + kurulum maliyetinin 30.000 TL olduğu bilgisi verilmiştir. Modelde planlanan tasarıma göre, 250 tonluk kapasite için 5 tank gerekir yani 150.000 TL kurulum maliyeti vardır. Eğer, örneğin 400 tonluk bir kapasite (yani referans stok) istersek, 8 tanka ihtiyaç duyulur ve bunların kurulum maliyeti de 240.000 TL olur.

Fazla kapasitenin kurulum maliyeti yüksektir ancak daha fazla depo kapasitemiz olursa tedarik periyotlarını artırabilir bu sayede toplam tedarik maliyetlerimiz de düşer. Örnek olarak; depo kapasitesini 400 ton (yani ref stok değişkeni değeri 400) ve her 9 günde bir tedarik planı (yani tedarik periyodu değişkeni değeri 9) olan başka bir senaryo ile modeli çalıştırdığımızda aşağıdaki sonuçlar elde edilmiştir.

### Time Persistent

Variable	Average	Half Width	Minimum Average	Maximum Average	Minimum Value	Maximum Value
karsılanamayan talep	367.68	58,20	50.3220	733.54	0.00	1175.21
mevcut stok	203.11	3,48	188.63	228.66	0.00460032	400.00
talep karşılama oranı	0.9179	0,01	0.8362	0.9905	0.00	1.0000
tedarik maliyeti	10500.00	0,00	10500.00	10500.00	0.00	21000.00
toplam talep	4061.63	87,95	3509.86	4644.77	0.00	8863.61

Buna göre, talep karşılama oranı ort. yaklaşık %92 ve tedarik maliyeti 180 gün için 21.000 TL olmaktadır. Tedarik periyodu 5 gün ve depo kapasitesi 250 tonluk alternatifimizdeki tedarik maliyeti olan 37.000 TL'ye göre 16.000 TL tasarruf sağlayabileceğimiz görünüyor (tatbikî 180 günlük dönem için). Ancak, ilk kurulum maliyeti 150.000 TL'den 240.000 TL'ye çıkmaktadır (90.000 TL ilave maliyet).

Yani, kurulum maliyetindeki artışı, yaklaşık 3 yıl sonunda, tedarik maliyetlerindeki düşüş ile karşılayabilir gibi görünüyor :  $(180 \text{ gün} / 365 \text{ gün}) \times (90.000 \text{ TL} / 16.000 \text{ TL}) = 2.77$

Bu noktada, benzetim modelinin bize sağlayacağı karar desteği aslında elde etmiş bulunuyoruz. Benzetim modelleri ile elde edilen sonuçlar, doğrudan karar olarak kullanılmaz. Karar vermeye destek olur. (Örn. bütçemiz, 3 yılda amorte etme durumuna uygunsa, en son alternatif uygulanabilir)

Modelde denenebilecek diğer alternatifler personel ve araçlar ile ilgili olabilir. Yukarıda yapılan analize benzer şekilde, örneğin 1 tanker kamyonu, 1 uzman erbaş ve 2 sözleşmeli er alternatifi de denenebilir ve sonuçları talep karşılama süreleri için belirlenen hedef ile karşılaştırılabilir.

Görüldüğü üzere, farklı alternatifleri farklı sonuçlar üretebilmektedir. Benzetim modelleri, hatırlanacağı üzere, kurulması planlanan sistem alternatiflerinin denenmesi için de uygun araçlardır. Ancak, farklı alternatifleri rastgele ve deneme yanılma yolu ile denemek yerine, kapsamlı ve sistematik bir yaklaşım kullanılması daha uygundur. Benzetim modelinin her çalıştırılması, esasen bir deneydir ve deneyin alternatifler ile yapılacağının belirlenmesi “Deney Tasarımı (Experimental Design)” adı verilen bir yaklaşım kullanılabilir (Şu aşamada bu konuya detaylı girilmeyecektir).

Özetle; Deney Tasarımı, hangi değişkenlerin hangi aralıklarda değişebileceği, hangi değişkenin hangi değişken ile birlikte değişme durumunun belirlenmesi vb. hususlarda sistematik bir yol gösterir. Örneğin; modelimizde tedarik periyodu için 3, 4, 5, 6, 7, 8, 9, 10 gün olarak 8 adet, depo kapasitesi için de 150, 200, 250, 300, 350, 400, 450, 500 ton olarak 7 adet alternatif olabileceğini, bunun için de toplam  $8 \times 7 = 56$  farklı kombinasyonda modelin çalıştırılması düşünülebilir.

## ÖRNEK-5 TERSANE SAC LEVHA İMALAT ATÖLYESİ MODELİ

Tersanede, gemilerin çeşitli bölümlerinde kullanılmak üzere sac levha imalatı yapılan bir atölye, günde 8 saat haftada 5 gün esasına göre çalışmaktadır.

Her çalışma günü, atölyenin çalışmaya başladığı 08:00'da 25 adet "ham sac levha" (raw sheet metal) kamyon ile atölyeye getirilmektedir. Ham sac levhalar, çeşitli işlemlerden geçerek son ürün haline getirilmektedir. Atölyede, ham sac levhalardan 2 farklı tip sac levha (son ürün) imal edilmektedir.

Gelen ham sac levhalar, önce tek tek incelenerek yüzey ölçüleri vb. özelliklerine göre, Tip-1 veya Tip-2 levha imalatı için kullanımları belirlenmektedir (işlemlere başlamadan hemen öncesinde tip-1 veya tip-2 olup olmadığı belli oluyor). Geçmiş dönem verilerine göre, atölyeye gelen bir ham sac levhanın tip-1 olarak belirlenme olasılığı %60'dır. Tip-1, Tip-2 ayrımı için yapılan inceleme çok kısa bir zaman almaktadır (levha başına 2-3 saniye), bu sebepten bu süre benzetimde göz ardı edilebilir.

Atölyede; levhaları işlemek üzere 4 iş istasyonunda 4 ayrı işlem yapılmaktadır (Kesme, Delme, Fırçalama, Temizleme). Takip edilmesi gereken işlem sıralaması şu şekildedir:

	1.İşlem	2.İşlem	3.İşlem	4.İşlem
Tip-1	Kesme	Fırçalama	Temizleme	-
Tip-2	Kesme	Delme	Fırçalama	Temizleme

### İşlem Süreleri

	Kesme	Delme	Fırçalama	Temizleme
Tip-1	Üçgen Dağılım (15, 18, 25) dk	-	Üçgen Dağılım (20, 24, 25) dk	Düzgün Dağ. (20,30) dk.
Tip-2	Düzgün Dağılım (19, 21) dk	Beta Dağılımı (alfa=5, beta=2) [15, 30] dk	Düzgün Dağılım (15, 20) dk	

Atölyede, her bir işlem için 1'er adet o işleme uygun makine (resource) vardır. Bir işleme gelen sac levha, makine dolu ise makinenin hemen arkasında bekletilmekte (birden çok bekleyen levha var ise kuyruğa girmekte), işlemi biten levhalar da makinenin çıkışındaki alanda bir sonraki işleme taşınmak için bekletilmektedir.

### Taşıma İşlemleri

Atölyedeki tüm taşıma işleri forklift ile yapılmakta ve forklift her transferde sadece 1 adet sac levha taşıyabilmektedir. Tüm işlemleri biten sac levha da yine forklift ile depoya taşınmaktadır. Atölyede 1 adet forklift (resource) mevcuttur.

Forkliftin bir levhayı taşıma süresi, iki makine tipi arası mesafeye göre değişmektedir:

Kesme → Delme	10 dk
Kesme → Fırçalama	5 dk
Delme → Fırçalama	15 dk
Fırçalama → Temizleme	3 dk
Tüm makinelerden → Hurda Deposu	20 dk

Forkliftin taşınmayı bekleyen levhanın yanına boş olarak gidişi için bir süre gerekmediği, forkliftin taşıyacağı levhaya anında ulaşabildiği kabul edilecek (Gerçek durumda, forkliftin katettiği mesafe; yükleme noktasına gidiş + indirme noktasına gidiş olmalıdır. Ancak forklift boş iken çok hızlı olduğundan dolayı yüklemeye gidiş süreleri göz ardı edilecektir).

Ayrıca, her sabah atölyeye getirilen ham sac levhaların kamyonundan indirildiği yer ile ilk işlem yeri olan kesim makinesine taşınması arasında da bir süre geçmemekte, tüm levhlar kesim makinesinin hemen yanındaki alana indirilmektedir.

### İşçilerin Mola Süreleri

Her makinede, sadece o makineyi çalıştırmayı bilen birer işçi görevlidir (resource). İşçiler, rastgele zamanlarda mola vermektedir. Mola süresince o işçinin makinesi durmaktadır (işçi makinenin içerisinde devam eden bir iş varken molaya çıkmıyor, en azından o işin bitmesini bekliyor). Bir işçi için molalar arası geçen süre ve mola süresine ait olasılık dağılımı bilgileri aşağıda verilmiştir:

Molalar Arası Geçen Süre	Mola süresi
Düzgün Dağılım (2,3) saat	Üçgen Dağılım (5, 10, 20) dk

### Kalite Kontrol, Yeniden İşleme, Hurdaya Ayırma :

Geçmişten elde edilen verilere göre aşağıdaki oranlarda kusurlu işlem ve buna bağlı olarak hurda çıktığı veya yeniden işlem yapılmak durumunda kalındığı görülmektedir:

- Kesim işleminden çıkan Tip-1 levhaların %10'u , Tip-2 levhaların ise %20'si hatalı kesim nedeniyle bir sonraki işleme geçmiyor, "hurda levha" oluyor ve Hurda Deposuna taşınmak üzere ayrılıyor.
- Delme işlemi sonrasında da levhaların %10'u "hurda levha" oluyor ve Hurda Deposuna taşınmayı beklemek üzere ayrılıyor.
- Temizleme işleminden sonra yapılan kontrolde levhaların %25'ine yeterince iyi fırça ve temizlik yapılamadığı görüldüğünden, Fırçalama işlemine geri döndürülüyor (yeniden fırçalama işlemine döndürülen levhalar sonrasında temizlik işlemine de tekrar giriyor).

### Depolama ve Envanter :

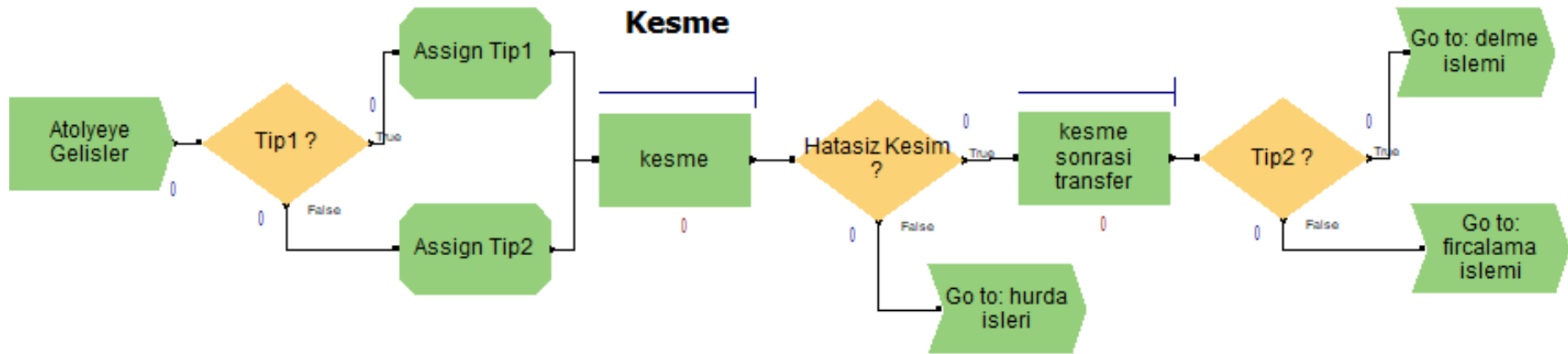
Depoya gelen levhalar tip-1 ve tip-2 olarak ayrı ayrı depo içerisinde stoklanıyor ve bunların envanter sayıları ayrı ayrı tutuluyor. Depo kapasitesi çok fazla olduğundan, bir kısıt olarak düşünülmesine gerek yok.

Hurda Deposuna ise, her iki günde (48 saat) bir kere bir kamyon gelerek, hurda deposunda birikmiş olan tüm hurda levhaları alıyor ve hurda deposunu tamamen boşaltıyor.

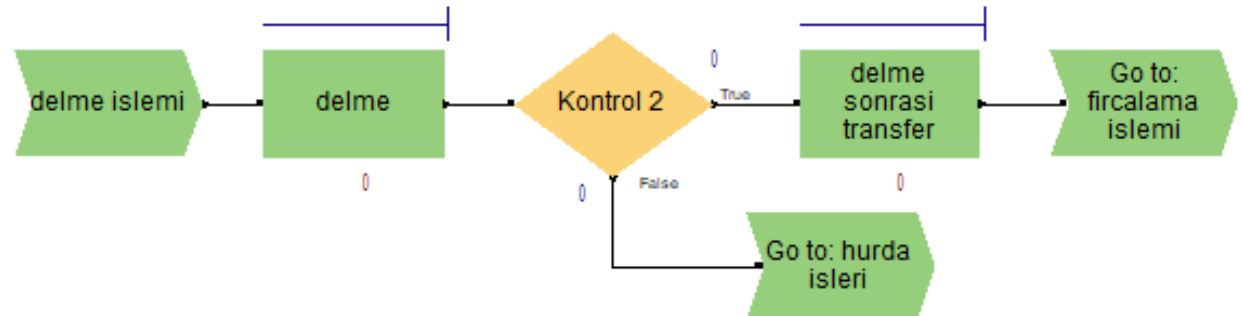
### İsteneler :

**1-** Yukarıda anlatılan sistemin benzetim modelini oluşturun. Varlıklar farklı görünümde olacak (farklı renkli yuvarlak şekil olabilir). Run Setup ayarlarından; 30 tekrar, 8 saat ısınma süresi, 55 saat tekrar süresi, günde 8 saat çalışma, temel zaman birimi "saat" olarak belirtin. Sonuçları yorumlayın.

**2-** Daha sonra, modelinizde şu değişiklikleri yaparak tekrar çalıştırın ve sonuçları bir önceki durum ile karşılaştırın: **2 forklift, günde 30 ham sac levha teslimatı, Temizleme İşlem süresi sabit 20 dk.**



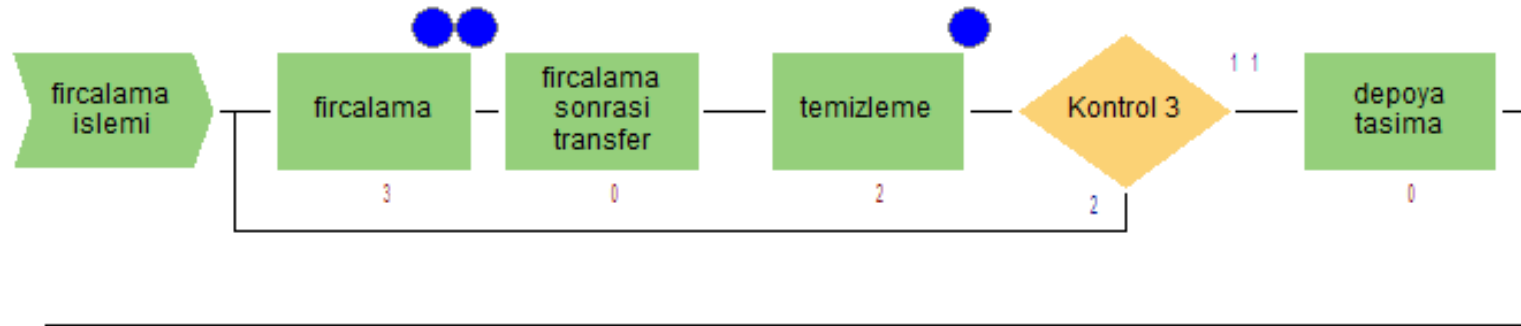
### Delme



### Hurda



## Fircalama Temizleme



## Envanter

