İSTANBUL TEKNİK ÜNİVERSİTESİ★FEN BİLİMLERİ ENSTİTÜSÜ

SÜREKLİ SİSTEM BENZETİMİ VE DİFERANSİYEL DENKLEMLER İÇİN ETKİLEŞİMLİ İNTERNET ORTAMINDA BİR VERİTABANI YÖNETİCİSİ

YÜKSEK LİSANS TEZİ

Y. Müh. Özden ERKAN
(509960152011)

Tezin Enstitüye Verildiği Tarih : 2 Haziran 2000
Tezin Savunulduğu Tarih : 20 Haziran 2000

100911

Tez Danışmanı: Doç. Dr. Gazanfer ÜNAL
Diğer Jüri Üyeleri: Prof. Dr. Metin DEMİRALP
Doç. Dr. Şakir KOCABAŞ

HAZİRAN 2000

ÖNSÖZ

Bu tezde bana yardımcı olan, başta aileme ve bana böyle bir tez çalışmasını uygun gören değerli hocam Doç. Dr. Gazanfer Ünal'a, tez çalışmalarında eşsiz katkılarını benden esirgemeyen değerli hocam Prof. Dr. Tuncer.İ.Ören'e, TÜBİTAK Marmara Araştırma Merkezi, Bilişim Teknolojileri Araştırma Estitüsü, eski ve şimdiki müdürleri, Prof. Dr. Fuat İnce ve Dr. Hakan Kalyoncu'ya, TÜBİTAK Marmara Araştırma Merkezi, Bilişim Teknolojileri Araştırma Estitüsü, Yazılım Sistemleri Araştırma Gurubu eski ve şimdiki gurup liderleri, Selçuk Taral ve Dr. Kamil Bağde'ye, çalışma arkadaşlarıma ve emeği geçen herkese teşekkürlerimi ve şükranlarım sunarım.

HAZİRAN 2000 **Özden ERKAN**

İÇİNDEKİLER

ÖNSÖZ ii
ŞE1İL LİSTESİ vi
ÖZET viii
SUMMARY x

1. GİRİŞ 1

1.1 Giriş ve Çalışmanın Amacı 1

2. BENZETİM VE BENZETİM ORTAMLARI 5

2.1. Benzetim Kavramı 5
2.2. Bilgisayar Destekli Benzetim Ortamlarının Genel Yapısı 14
2.2.1. Model Tanımlama 17
2.2.2. Deney Tanımlama 24
2.2.3. Program Üretimi 25
2.2.4. Deneyin Yapılması ve Sonuçlar 27
2.2. Benzetim Ortamlarının Türleri 29

3. MODEL VE MODELLEME 31

3.1. Genel Sistem Teorisi 31
3.2. Model ve Modelleme Kavramı 34
3.3. Model Türleri 41
3.3.1. Sürekli Model 42
3.3.2. Ayrık Model 45
3.3.3. Belleksiz Model 47
3.4. Modeller Arası İlişki 47
3.4.1. Bağlaşım Kavramı 48
3.4.2. Giriş-Çıkış Modeli 51
3.4.3. İç Bağlaşım 51
3.4.4. Dış Bağlaşım 52
3.4.5. Geri Besleme Bağlaşım 53
3.4.6. İçi İçe Bağlaşım 55
3.4.7. Bağlaşık Model Oluşturma Kuralları 57

3.4.8. Bağlaşık Modeller İçin Benzetim Algoritması 58

4. BENZETİM DENEYİNİN TANIMLANMASI 63

4.1 Benzetim Deneyinin Genel Yapısı 63
4.2 Deney Modüllerinin Tanımlanması 64
4.2.1. Genel Deney Modülünün Tanımlanması 64
4.2.2. Bileşen Modellere Ait Deney Modüllerinin Tanımlanması 65
4.3. Çıkış Modüllerinin Tanımlanması 65
4.4. Çalışma Anı Kontrol Tanımlamaları 65

5. BENZETİM PROGRAMININ ÜRETİLMESİ 67

5.1. Genel Yaklaşım 67
5.2 . Tanımlama Progamının Üretilmesi 68
5.3 Benzetim Progamının Üretilmesi 69
5.3.1. Benzetim Programının Genel Mimarisi 70
5.3.2. Model Tanımlarının Program Koduna Dönüştürülmesi 72
5.3.2.1. Bileşen Model Tanımlarının Program Koduna Dönüştürülmesi 77
5.3.2.2 Bağlaşık Model Tanımlarının Program Koduna Dönüştürülmesi 78
5.3.3. Parametre Değerlerine Erişim Modülleri 79
5.3.4. İnternet Bağlantı Modül 80
5.3.5. Deney Koşulları Tanımlamalarına Erişim Modülleri 80
5.3.6. Çıkış Ortamı Tanımlamalarına Erişim Modülleri 80
5.3.7. Program Sonuçlarının Dosya Sistemine Aktarılması 81
5.3.8. Çalışma Anı Sınıf Kütüphaneleri 81
5.4 Benzetim Progamının Derlenmesi ve Çalıştırılması 81

6 MODEL DAVRANIŞININ ÜRETİLMESİ VE İŞLENMESİ 82

6.1. Adi Diferansiyel Denklemler ve Çözüm Yöntemleri 82
6.2. Model Davranışının İşlenmesi 84
6.2.1. Sonuçların Gösterilmesi 86
6.2.2. Sonuçların Analizi 86

7. BİR GÖRSEL MODELLEME VE BENZETİM ORTAMI : GestCell 87

7.1 GestCell'in Genel Yapısı ve Yazılım Mimarisi 87
7.1.1 Model Tanımlama 88
7.1.2 Benzetim 90
7.1.2.1 Parametre Seti Tanımlama 90
7.1.2.2 Deney Tanımlama 91
7.1.2.3 Çıkış İşleminin Tanımlanması 92
7.1.3 Benzetim Programının Üretilmesi 93

7.1.4. Grafik Ortam 94
7.1.5. İnternet Bağlantısı 94

8. İNTERNET ORTAMINDA ADİ DİFERANSİYEL DENKLEMLERİN ÇÖZÜMLERİNİ İÇEREN BİR VERİTABANI YÖNETİCİSİ:ODENET 95

8.1 Giriş 95
8.1.1. Önerilen Yaklaşım 95
8.1.2. Veritabanı 96
8.2 Sistemin İşlevsel Yapısı 97
8.2.1. Sıralama 97
8.2.2. Veritabanında Arama 104
8.2.3. Arayüz 105
8.3 Sistemin Yazılım Mimarisi 106
8.3.1. Sistemin Genel Yapısı 106
8.3.1.2 . Sıralama Modülü 110
8.3.1.3. Veritabanı Modülü 111

9. SONUÇ 112

KAYNAKLAR 114

EKLER

EK A

GestCell MODELLEME VE BENZETİM ORTAMININ KULLANICI ARAYÜZÜ ÖRNEKLERİ 117

1. Bir Modelin Giriş ve Çıkışları 117
2. Modelin İç Yapısının Tanımlanması 118
3. Model Nesnelerinin Grafik Arayüzde Temsil Edilmesi 119
4. Bağlaşık Model 120
5. Benzetim Tanımlamaları 120
6. Parametre Seti Tanımlama 121
7. Deney Tanımlama 123
8. Çıkış Ortamı Tanımlama 125

ÖZGEÇMİŞ 126

ŞEKİL LİSTESİ

		Sayfa
Şekil 2.1	: Sistem problemi çözme ya da sistem davranışlarını belirlenmesi süreci	6
Şekil 2.2	: Modelleme ve benzetim süreci	9
Şekil 2.3	: Benzetim sistemi	12
Şekil 2.4	: Benzetim dilinin blok yapısı	16
Şekil 2.5	: Yazılım sisteminin kullanıcı arayüzü ve dosya sistemi arsındaki etkileşim	17
Şekil 2.6	: Bir sistemin genel yapısı	18
Şekil 2.7	: Bağlaşık model	20
Şekil 2.8	: Parametrik model	21
Şekil 2.9	: Parametrik modelin tanımlaması	23
Şekil 2.10	: Deneyin tanımlaması	25
Şekil 2.11	: Program Üretimi	27
Şekil 2.12	: Değişken-zaman ekseni	28
Şekil 2.13	: Benzetim programı çalışma anı ve çıkış işlemi tanımlamaları	29
Şekil 3.1	: Sistemin dış ortam kavramları	32
Şekil 3.2	: Sistem hiyerarşisi	35
Şekil 3.3	: Bileşen modelin tanımlama blokları	37
Şekil 3.4	: Bağlaşık modelin tanımlama blokları	38
Şekil 3.5	: Sürekli zaman arlığı için sürekli modelde durum ve çıkış değişkenlerinin yörüngesi	44
Şekil 3.6	: Ayrık zaman arlığı için sürekli modelde durum ve çıkış değişkenlerinin yörüngesi	44
Şekil 3.7	: Sürekli zaman arlığı için sürekli modelde giriş değişkenlerinin yörüngesi	45
Şekil 3.8	: Ayrık zaman arlığı için sürekli modelde giriş değişkenlerinin yörüngesi	45
Şekil 3.9	: Sürekli zaman arlığı ayrık modelde için giriş değişkenlerinin yörüngesi	46
Şekil 3.10	: Ayrık zaman arlığı ayrık modelde için giriş değişkenlerinin yörüngesi	47
Şekil 3.11	: İki model arasında bağlaşım	48
Şekil 3.12	: Model hiyerarşisi	49
Şekil 3.13	: Üç bileşen ve bir bağlaşık model içeren model hiyerarşisi	50
Şekil 3.14	: Üç bileşen ve bir bağlaşık model içeren model hiyerarşisinin belirttiği bağlaşık modelin tanımlama şeması	50
Şekil 3.15	: İki model arasında iç bağlaşım	52
Şekil 3.16	: Dış bağlaşım	53
Şekil 3.17	: Bir modelin kendi üzerinde oluşturduğu geri besleme bağlaşım.	54
Şekil 3.18	: Bir modelin, doğrudan kendisine değer gönderen model ile yaptığı geri besleme bağlaşım	54

Şekil 3.23 : Deney öncesi..... 60
Şekil 3.24 : Deney süresi..... 61
Şekil 3.25 : Deney sonrası..... 62
Şekil 4.1 : Benzetim deneyi..... 64
Şekil 4.2 : Benzetim deneyinin çalıştırılması..... 66
Şekil 5.1 : Benzetim programının üretilmesi ve çalıştırılması..... 70
Şekil 5.2 : Bağlaşık modellerin bileşen modelleri içermesi..... 74
Şekil 5.3 : ModelA ve ModelB sınıflarına ait nesneleri içeren ModelC nesnesinin sınıf diyagramı..... 74
Şekil 5.4 : Model sınıf hiyerarşisi..... 76
Şekil 6.1 : Davranış üretiminde, sonuçların kullacıya gösterilmesi..... 86
Şekil A.1 : Bir modelin giriş ve çıkışları..... 117
Şekil A.2 : Bir modelin giriş veçıkışları..... 118
Şekil A.3 : Durum geçiş fonksiyonlarının tanımlaması..... 118
Şekil A.4 : Dokunmatik denklem ditörü..... 119
Şekil A.5 : Visual Basic form penceresi ile modelin temsil edilmesi..... 119
Şekil A.6 : Bağlaşık model..... 120
Şekil A.7 : Benzetim tanımlama diyalog penceresi..... 120
Şekil A.8 : Parametre seti tanımlama diyalog penceresi..... 121
Şekil A.9 : Parametre değerlerinin girilmesi..... 121
Şekil A.10 : Girilen parametre değerinin listede görüntülenmesi..... 122
Şekil A.11 : Parametre setinin parametre listesine eklenmesi..... 122
Şekil A.12 : Deney tanımlama diyalog penceresi..... 123
Şekil A.13 : Durum değişkenlerinin ilk değer tanımlamaları..... 123
Şekil A.14 : Girilen durum değişkenlerinin ilk değerinin listede görüntülenmesi..... 124
Şekil A.15 : Deneyin deney listesine eklenmesi..... 124
Şekil A.16 : Çıkış modülü tanımlama diyalog penceresi..... 125

SÜREKLİ SİSTEM BENZETİMİ VE DİFERANSİYEL DENKLEMLER İÇİN ETKİLEŞİMLİ İNTERNET ORTAMINDA BİR VERİTABANI YÖNETİCİSİ

ÖZET

Bu tezde, birbirleri ile etkileşim halinde olan iki farklı sistem geliştirilmiştir. Tezde geliştirilen sistemlerden ilki, sürekli sistemlerin modellendiği ve bu modeller üzerinde benzetim deneyinin tanımlanabildiği, görsel bir modelleme ve benzetim ortamıdır.

Geliştirilen ikinci sistem ise, sürekli sistemlerin modellenmesinde kullanılan adi diferansiyel denklemler ve bunların parametrik çözümlerini içeren ve internet ortamında çalışan etkileşimli bir veritabanı yönetim sistemidir.

Bir sürekli sistemin matematik modeli, model değişkenleri, model parametreleri ve bu değişkenlerin ve parametrelerin kombinasyonlarından oluşan diferansiyel denklemler içerir. Diferansiyel denklemlerin çözümünün elde edilmesi için internet ortamında yer alan veri tabanına bağlanılır. Eğer, diferansiyel denklemin çözümü varsa, bu çözüm benzetim ortamına aktarılır. Eğer çözüm yoksa, benzetim ortamında sayısal entegrasyon işlemi uygulanarak çözüm fonksiyonunu belirtilen zaman aralıkları için değerleri elde edilir.

Modelleme ve benzetim ortamında sadece tek bir sürekli sistemin modeli değil, birden fazla sürekli sistemi alt sistem olarak içeren ve hiyerarşik yapıya sahip sistemin modeli de görsel bir ortamda oluşturulur.

Prof. Dr. Tuncer İ. Ören tarafından ortaya atılan ve bir sistemin alt sistemleri arasında giriş- çıkış ilişkilerini ifade eden bağlaşım kavramı detaylı olarak incelenmiştir. Alt sistemler arasındaki bağlaşımlar, görsel modelleme ortamında tanımlanır.

Bir sistemin modeli oluşturulduğunda, model benzetim ortamında, bir benzetim programı ile işlenir. Benzetim programı modelin yapısına bağlı olarak, benzetim ortamı tarafından üretilir.

Bilim Dalı Program Kodu: 590942

CONTINIOUS SYSTEM SIMULATION AND AN INTERACTIVE DATABASE MANAGEMENT SYSTEM FOR DIFFERENTIAL EQUATOINS ON INTERNET CONTENTS

SUMMARY

In this thesis; two different systems, which interact with each other were developed. The first of the systems developed in the thesis is a visual modelling and simulation environment on which the continuous systems are modelled and the simulation experiment can be defined on these systems.

The developed second system is an interactive database management system, which contains ordinary differential equations and their parametric solutions and works on the internet environment.

The mathematical model of a continuous system contains the model variables, model parameters and differential equations which are composed of the combinations of these variables and parameters. In order to obtain the solution of the differential equation, the database on the internet environment is connected. If the solution of the differential equation exists this solution is transferred to the simulation environment. If the solution of the differential equation does not exist, in the simulation environment, by applying numerical integration methods, the values of the solution function are obtained for the time intervals determined.

In the modelling and simulation environment, not only the model of a single continuous system, but also the model of the systems which contain more than one continuous system as subsystem and have hierarchical structure is created in a visual environment.

The coupling concept which was claimed by Ph. Dr. Tuncer İ. Ören and expresses the input-output relationships between the subsytems are defined in the visual modelling environment.

When the model of a system is created, the model, in the simulation environment, is processed by the simulation program. The simulation program, depending on the structure of the model, is generated by the simulation environment.

Science Code: 590942

1.GİRİŞ

1.1 Giriş ve Çalışmanın Amacı

Sistemlerin tasarımında ya da varolan sistemler üzerinde strateji geliştirebilmek amacıyla, o sistemlerin genel yapısı hakkında, belirlenen hedefler doğrultusunda, bilgi edinme ihtiyacı doğar.

Ancak, günümüzde, teknolojinin gelişmesiyle birlikte sistemlerin karmaşıklığı gittikçe artmaktadır. Buna paralel olarak, sistemlerin davranışları hakkında bilgi edinebilmek ve bu bilgi ışığında, sistemler üzerinde strateji geliştirebilmek hem zor hem de riskli hale gelmektedir.

Belli bir sistem üzerinde strateji belirlenirken, o sistemin hangi koşullar altında, ne tür davranış göstereceği ve sistemin çalışma koşulları ya da sistemi oluşturan bileşenlerin yapılarında veya sistemdeki konumlarında meydana gelen değişikliklere ne şekilde tepki vereceği tespit edilir. Bu tespitler ışığında sistemin genel yapısı ve dinamiği hakkında bilgi edinilmiş olur. Bir sistemin, belirli koşullar altında ne tür davranış göstereceğini belirleme ve yapısal değişikliklere nasıl tepki vereceğini tespit etme işlemine benzetim (simülasyon), bu maksatla, sistem üzerinde gerçekleştirilen deneye de benzetim deneyi adı verilir.

Ancak, sistemin yapısal karmaşıklığı ya da benzetim deneyinin riskli olması, benzetim deneyinin sistem üzerinde gerçekleştirilebilmesini engeller. Bu nedenle, bir sistem üzerinde, benzetim ortamında deney gerçekleştirebilmek için, o sistemin benzetim ortamında bir model ile temsil edilir.

Bir sistemin modeli oluşturulurken, modelin, sistemin özelliklerini tamamen değil, belirlenen deney objektivitelerine ve sistem üzerinde gözlemlenecek özelliklere göre yansıttığı düşünülür.

Bu tez çalışmasında, belli bir zaman aralığında, sürekli bir hareket sergileyen sistemleri kapsayan bir görsel modelleme ve benzetim ortamı, ve bu görsel modelleme

ve benzetim ortamı ile etkileşim halinde olan, internet ortamında çalışan ve sürekli sistemlerin modellenmesinde kullanılan adi diferansiyel denklemlerin analitik çözümlerini içeren bir veritabanı yönetim sistemi geliştirilmiştir.

Bu tezde, Prof. Dr. Tuncer. İ. Ören nezaretinde, Kanada'da bulunan Ottawa Üniversitesi'nde, Kim. M. Tam ve Neil Keon tarafından yapılan tezlere konu olan sistemler baz alınmakla birlikte, bu tezde tasarlanan sistemler, her iki teze de konu olan sistemlerin genişletilmiş ve birçok eklenti içeren halidir.

Bununla birlikte, bu tez çalışmasında, genel sistem teorisi kapsamında, bir sistemin, bilgisayar ortamında matematik modelinin oluşturulması, birden fazla sistemi hiyerarşik bir yapıda alt sistem olarak içeren bağlaşık sistemlerin modellenmesi ve bu sistemler üzerinde benzetim deneyinin tanımlanması, benzetim deneyini bilgisayar ortamında icra eden benzetim programının benzetim ortamı tarafından otomatik olarak üretilmesi ve bu programın internet ortamıyla bağlantı kurma şekli, Prof. Dr. Tuncer. İ. Ören tarafından geliştirilen GEST benzetim dili ve son olarak, internet ortamında yer alan veritabanı yönetim sisteminin çalışma prensibi ve diferansiyel denklemler için geliştirilmiş terim tanıma ve sıralama algoritması ele alınmıştır.

Bu tezde tasarlanan benzetim ortamı, genel olarak görsel bir modelleme ve benzetim yazılımı olup, tamamen modüler bir yapıya sahiptir. Yazılımın modüler bir yapıda olması birbirinden bağımsız alt yazılım modüllerinin bir araya gelerek tüm yazılımı oluşturduklarını ifade eder.

Bu alt modüllerden ilki, matematik modellerin tanımlandığı modüldür. Bu modülde, sistem elementlerini temsil eden değişkenler ve parametreler, sistemin giriş-çıkışları ve bunlar arasındaki matematiksel formülasyonlar tanımlanır. Bir diğer modül olan görsel bağlaşım modülü, birden fazla modelin, görsel olarak, aralarında giriş-çıkış ilişkileri kurarak, bağlaşık bir model oluşturdukları modüldür. Diğer bir modül olan benzetim tanımlama modülünde, tanımlanan model üzerinde benzetim tanımlamalarının yapıldığı modüldür. Bu modülde, parametre setleri, deney ve çıkış işlemi tanımlanır. Program üretme modülünde ise, modele uygun olarak, benzetim programı üretilir. Son modül ise, benzetim programının derlenip çalıştırıldığı ve sonuçların grafik ortama aktarıldığı modüldür.

Mühendislik problemlerinde ve sürekli sistemlerin modellenmesinde, çözüm ya da modelin kendisi diferansiyel denklemlerin çözümüne indirgenir.

Bu aşamada, diferansiyel denklemin sayısal ya da analitik çözümü gündeme gelir. Diferansiyel denklemlerin analitik çözümlerindeki karmaşıklık veya sayısal yaklaşımlardaki belirsizlik, bazen, sistemlerin modellenmesinde sorunlara yol açar.

Bu sebeple, diferansiyel denklemlerin analitik çözümlerini en kısa zamanda ve en doğru şekilde elde etme ihtiyacı doğar.

Tasarlanan bir diğer sistem, Prof. Dr. Tuncer. İ Ören nezaretinde, Kanada'da bulunan Ottawa Üniversitesi'nde, Kim. M .Tam tarafından yapılan bir tezin internet ortamında çalışan sürümüdür. Sistem, analiz itibariyle, orijinal sistemle aynı, tasarım açısından bakıldığında ise, nesne tabanlı olarak tasarlandığından, orijinal sisteme göre, farklılık gösterir.

Kullanıcı, arayüz sayesinde, istediği formatta, diferansiyel denklemi sisteme girebilir. Daha sonra sistem, denklemi sıralayıp, derecesini. belirler ve denklemi veritabanına gönderir. Sistemin bir diğer özelliği de, lokal ortamda yer alan modelleme ve benzetim ortamıyla ilişki kurabilmesidir.

Bu tezde öngörülen amaç, iki boyutta ele alınabilir. Buna göre, belirlenen ilk hedef, genel sistem teorisi kapsamında, bir sistemin matematik olarak nasıl ifade edileceğini ve birden fazla sistemin bir araya gelerek, hiyrarşik sistemler oluşturmalarını ve bu tür sistemlerin ne şekilde modelleneceğini, ve bu sistemler üzerinde benzetim deneyinin nasıl tanımlanacağını ve bir benzetim programının üretilme prensibini genel bir çerçevede ifade etmektir.

İkinci hedef ise, diferansiyel denklemler yardımıyla modellenen sistemler için ihtiyaç duyulan analitik çözümlerin internet gibi herkesin rahatlıkla erişebileceği bir ortamda, sanal bir el kitabı oluşturmaktır. Bu sanal el kitabının yerel bir benzetim ortamıyla ilişkilendirilebilmesi, dağıtılmış mimariye sahip olan sistemlerin etkinliğini ve modülerliğini ifade eder.

Tez metninin ikinci bölümünde, benzetim kavramı, detaylı olarak ele alınmıştır. Bu bölümde, ayrıca, genel sistem problemlerinin deneysel yöntemlerle çözümünde, buna bağlı olarak, benzetim işleminde takip edilen metot ve bilgisayar destekli benzetim ortamlarının yapısı ve türleri hakkında bilgi verilmiştir.

Üçüncü bölümde, genel sistem teorisi kapsamında, model ve modelleme kavramı ve kullanılan matematiksel formülasyona göre model türleri üzerinde durulmuştur. Bu bölümde, ayrıca, matematik modelin içirildiği bileşen model ve bileşen modellerin bir araya gelerek oluşturdukları bağlaşık model ve bağlaşım kavramları ve bağlaşık modeller için benzetim algoritması anlatılmıştır.

Dördüncü bölümde benzetim deneyinin tanımlanması ele alınmıştır.

Beşinci bölümde, benzetin deneyinin bilgisayar ortamında icra edilmesini sağlayan benzetim programının nesne tabanlı yaklaşımlar çerçevesinde nasıl üretildiği anlatılmıştır.

Altıncı bölümde, adi diferansiyel denklemlerden kısaca bahsedildikten sonra, model davranışı ve davranış türleri hakkında bilgi verilmiştir.

Yedinci bölümde, bu tezde tasarlanan modelleme ve benzetim ortamı olan GestCell hakkında kısaca bilgi verilmiştir.

Sekizinci bölümde, internet ortamındaki ODENET veritabanı yönetim sistemi ele alınmış olup, diferansiyel denklemlerin veritabanına uygun hale getirilmesinde kullanılan terim tanıma ve sıralama algoritması ayrıntılı olarak anlatılmıştır.

Bu tezin hazırlanmasında, TÜBİTAK Marmara Araştırma Merkezi, Bilişim Teknolojileri Araştırma Enstitüsü'nü bilgisayar olanaklarından ve Prof. Dr. Tuncer. İ. Ören'in teorik katkılarıdan yararlanılmıştır.

İnternet ortamında yer alan veritabanı yönetim sisteminin internet adresi:

http://undp.mam.gov.tr/ozden/ode/entry.htm

2 BENZETİM VE BENZETİM ORTAMLARI

2.1 Benzetim Kavramı

Günümüzde, teknolojinin gelişmesine paralel olarak, sistemlerin karmaşıklığı gittikçe artmaktadır. Buna paralel olarak, gerek sistem tasarımında, gerekse, varolan bir sistemin davranışı hakkında strateji belirlemede etkin bir metodolojinin belirlenip, uygulanması zorunludur.

Buna bağlı olarak, sistemin ne tür davranış göstereceğini belirlemenin yolu, o sistemin "belirli koşullar altında test edilmesi" olarak düşünülebilir. Ancak, bu her zaman mümkün olmayabilir. Şöyle ki, yapılan test işleminin riskli olması, sistemin geleceğini tehlikeye sokabileceği gibi, maddi bakımdan da sorun yaratabilir.

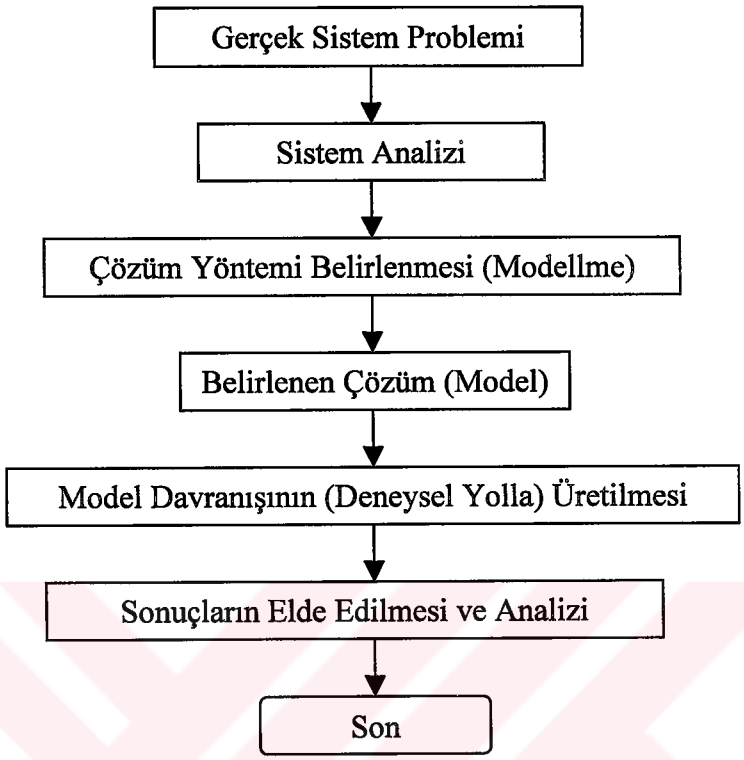
Bu noktadan hareketle, test işlemi, "kontrollü deney" işlemine dönüşür ki, bu da, hem deneyin daha metodolojik bir yöntemle yapılmasını, hem de üzerinde deney yapılacak olan sistemin, deneye uygun hale dönüştürülmesiyle mümkündür.

Burada söz konusu deneyin, "kontrollü", ve buna bağlı olarak, "metodolojik" bir yöntemle yapılması, deney objektivitelerinin önceden saptanmasıyla mümkündür. Bu dönüşüm işlemi, sistemin yapısal olarak değişimi ile değil, sistemin, "deney objektivitelerine bağlı uygun olarak, temsil edilmesi" ile gerçekleştirilir. Sistemin, belirlenen görüş açılarına bağlı olarak, bir nesne ile temsil edilmesi işlemine, "modelleme" , sistemi temsil eden nesneye de "model" adı verilir.

Buna göre, bir sistem üzerinde strateji belirlemenin bir yolu, o sistemi temsil eden model üzerinde kontrollü bir deney işleminin gerçekleştirmektir. Bu işleme, "benzetim" adı verilir.

Bir sistem probleminin çözümünün elde edilebilmesi ve sistem hakkında genel stratejinin belirlenebilmesi için, önce "sistemin analizi" yapılır. Daha sonra, problemin çözümü için uygulanacak çözüm yöntemi belirlenir. Eğer uygulanacak çözüm yöntemi, deneysel bir yöntem ise, bu deneysel yöntemle modelin davranışı üretilir.

Daha sonra, elde edilen sonuçlar analiz edilerek, sistem hakkında genel strateji belirlenir. Bu süreç aşağıdaki şemada belirtilmiştir.



Şekil 2.1 Sistem problemi çözme ya da sistem davranışlarını belirlenmesi süreci

Yukarıda en geniş anlamda açıklanan sistem problemi çözme ya da sistem davranışlarını belirlenmesi sürecinde belirtilen, "model davranışının üretilmesi" aşaması, modelin temsil ettiği sistemin karmaşıklığına ve çözümün uygulanmasında belirlenen objektivitelere bağlı olarak , farklılıklar gösterir.

Genel anlamda, oluşturulan model üzerinde uygulanan çözüm yöntemi iki başlık altında toplanabilir. Bunlar,

- Analitik çözüm
- Benzetim

Bir sistem modeli, analitik çözüm yöntemi kullanılarak, uygulamaya tabi tutulurken, çözümün bir kural tabanından, bir çıkarım (inference) mekanizması yardımı ile elde edilebileceği düşünülür. Burada ifade edilen "kural tabanı" , sadece klasik anlamda

bir "uzman sistemi" değil, belirli bir guruba ait problem takımının çözüm kuramını temsil eden kurallar bütününü ifade eder.

Bu yapıda, çözümü aranan problemin hangi guruba ait olduğu belirlendikten sonra, o guruba ait kurallar, çıkarım mekanizması ile işleme tabi tutularak, sonuca gidilir.

Örneğin; sonraki bölümlerde açıklanacağı gibi, sürekli sistemlerin modellenmesinde, sistem değişkenleri ve parametreleri arasında oluşturulan matematiksel formülasyonlarda, adi diferansiyel denklemler kullanılır. Bir adi diferansiyel denklemin "analitik" çözümü aranırken, önce, denklemin hangi guruba ait olduğu belirlendikten sonra, ilgili kurallar devreye sokularak, çözüme ulaşılır.

Ancak, problemin karmaşıklığından dolayı, kural tabanı çözümü elde etmede yetersiz kalabilir. Böyle bir durumda, modelin basitleştirilebileceği düşünülebilir. Ancak, modelin basitleştirilmesi, sistemin tam olarak temsil edilebilmesini engeller.

Bu esnada, problemin çözümünde, belirli bir guruba ait kuralların uygulanarak sonuca varılması yerine, yaklaşım metotları uygulanarak, belli koşullar altında sistem üzerinde "benzetim deneyi" uygulanır.

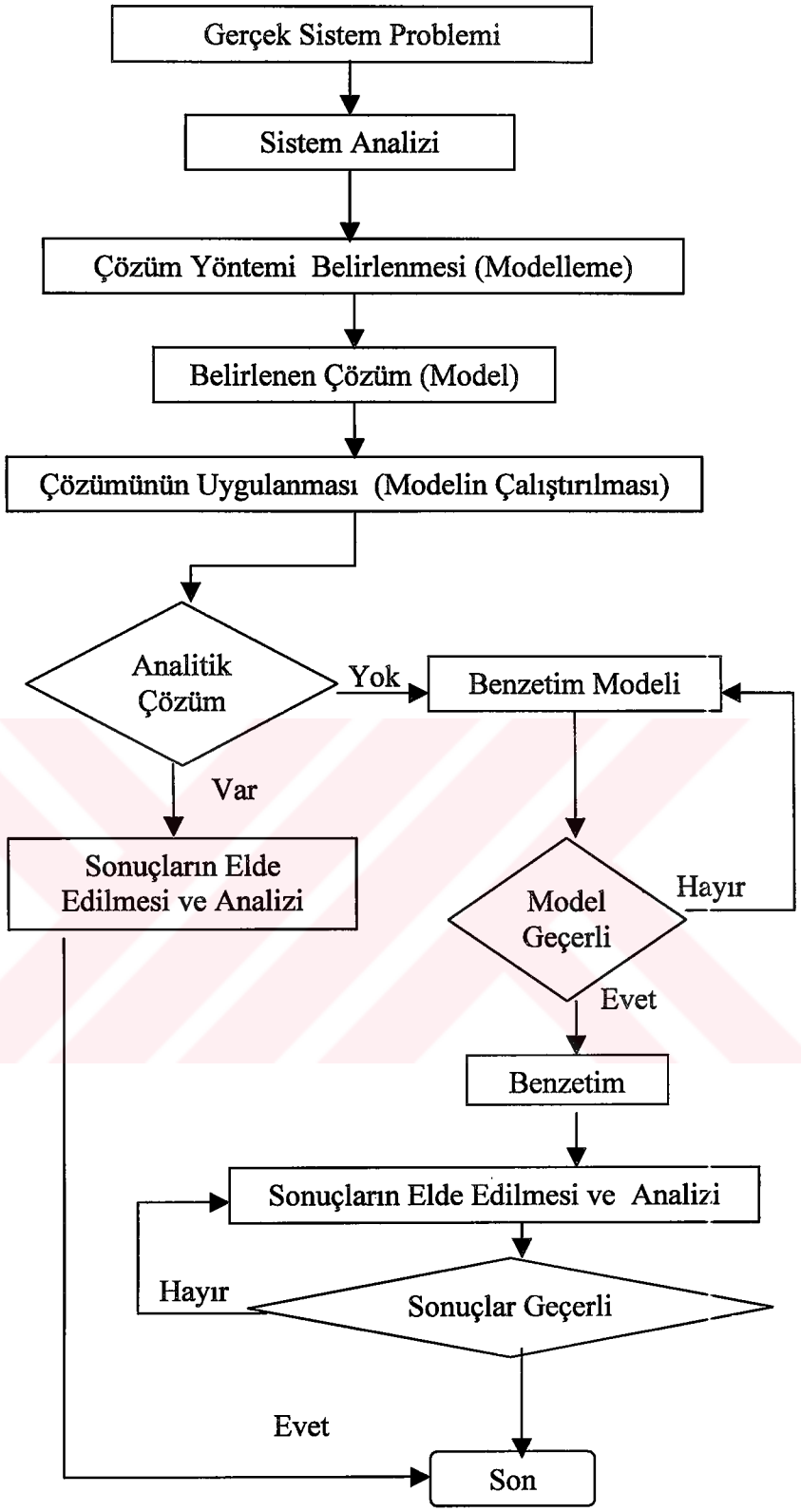
Bir problemin çözümü "benzetim deneyi" yardımıyla belirlenirken, belirlenen çözüm modeli, ilk etapta, sistem unsurlarını, "kavramsal" olarak temsil eder. Bu aşamada model, "kavramsal model"adını alır. Çözümü karakterize eden modelin, daha sonra, deney koşullarına uygun hale getirilmesi gerekir. Bir kavramsal modelin benzetim deneyine uygun bir şekilde, yeniden yapılandırılmasından bir "üst düzey" model meydana gelir. Bu modele, "benzetim modeli" ya da "parametrik model" adı verilir. Benzetim deneyi ve parametrik model, bir araya gelerek, bir "benzetim sistemi"nin tanımlayıcı unsurlarını oluştururlar. Bir sistem probleminin çözümünde, analitik çözüm yerine, benzetim sisteminden faydalanılıyorsa, çözümün gerçeğe en yakın sonucu verebilmesi için, gerçek sistem ve benzetim sistemi arasında "bir uyum"un olması gerekir. Bu uyum iki farklı unsuru kapsar. Bunlar:

- Modelin, gerçek sistemi, sistem probleminde belirlenen objektiviteler çerçevesinde temsil edebilmesi
- Benzetim deneyinin doğru sonuçları üretebilmesi.

Modelin, gerçek sistemi, sistem probleminde belirlenen objektiviteler çerçevesinde temsil edebilme yeteneği, kavramsal modelden elde edilen benzetim modelinin,

"geçerliliği" ile, benzetim deneyinin doğru sonuçları üretebilmesi de, kullanılan algoritma ve deney koşullarının "doğruluğu" ile ölçülebilir.

Buna göre, Şekil 2.1 ile ifade edilen, bir sistem probleminin çözüm süreci, aşağıdaki şemada olduğu gibi yeniden yapılandırılabilir.



Şekil 2.2 Modelleme ve benzetim süreci

Bir benzetim sisteminin tanımlanmasında, benzetim deneyinin uygulanacağı modelin, "parametrik model" şeklinde ifade edilmesinin nedeni, bir sistemin davranışında, "önceden belirlenebilen" koşullara bağlı olarak, farklılıklar gösterebileceğidir. Bu da, sistemin ve sistemi karakterize eden modelin, belirli "parametrelere" bağlı olmasıyla açıklanabilir. Bir benzetim deneyinde, modelin sahip olduğu parametre değerlerinin kümesi, "parametre seti" olarak adlandırılır.

Bir model üzerinde, bir benzetim deneyi, farklı parametre setlerine göre tekrarlanabileceği gibi, belirli bir parametre setine sahip model üzerinde, birden fazla benzetim deneyi yapılabilir. Parametre setleri ve benzetim deneylerindeki esneklik ve çeşitlilik, modeller için de geçerlidir.

Şöyle ki, bir gerçek sistem probleminin çözümü ya da modellenmesi çoğu kez, sistemin hiyerarşisine bağlı olarak, birden fazla "alt çözüm" ya da "alt sistem" bulunabilir. Her bir alt sistem farklı bir model ile temsil edilebilir. Bu modeller, daha sonra farklı sistemlerin modellenmesinde kullanılabilir. Bu yapı, "model tabanı" kavramını gündeme getirir. Bir model üzerinde benzetim deneyi gerçekleştirilirken, alt modellere bağlı olarak, benzetim deneyi üzerinde modüler bir yapı oluşturulur. Oluşturulan bu modüler yapıda, deneyin her bir elementine "deney modülü" denir

Bir model, bir benzetim deneyine tabi tutulurken, elde edilen deney sonuçları,"modelin davranışını" yansıtır. Modelin davranış şekli, benzetim sisteminde bulunan bir "davranış üretici" tarafından, deney objektivitelerine göre belirlenir ve gerçek sistemin davranışı hakkında bilgi sunar.

Bu bilginin sunuluş şekli, belirlenen "çıkış ortamına" göre farklı türde olabilir. Örneğin, bilgisayar destekli benzetim ortamlarında, deney sonuçları, bir grafik ortamına aktarılabileceği gibi, bir veritabanına ya da "bir başka benzetim sistemine" de aktarılabilir. Deney sonuçları, çıkış ortamına aktarıldıktan sonra, sonuçlar analiz edilir.

Benzetim sistemlerindeki, "model", "benzetim deneyi", "parametre setleri" ve "çıkış ortamı" çeşitliliği aşağıdaki matematiksel formülasyonla ifade edilebilir.

S=< M, E , P, O>

S:Benzetim sistemi (2.1)

M: Model

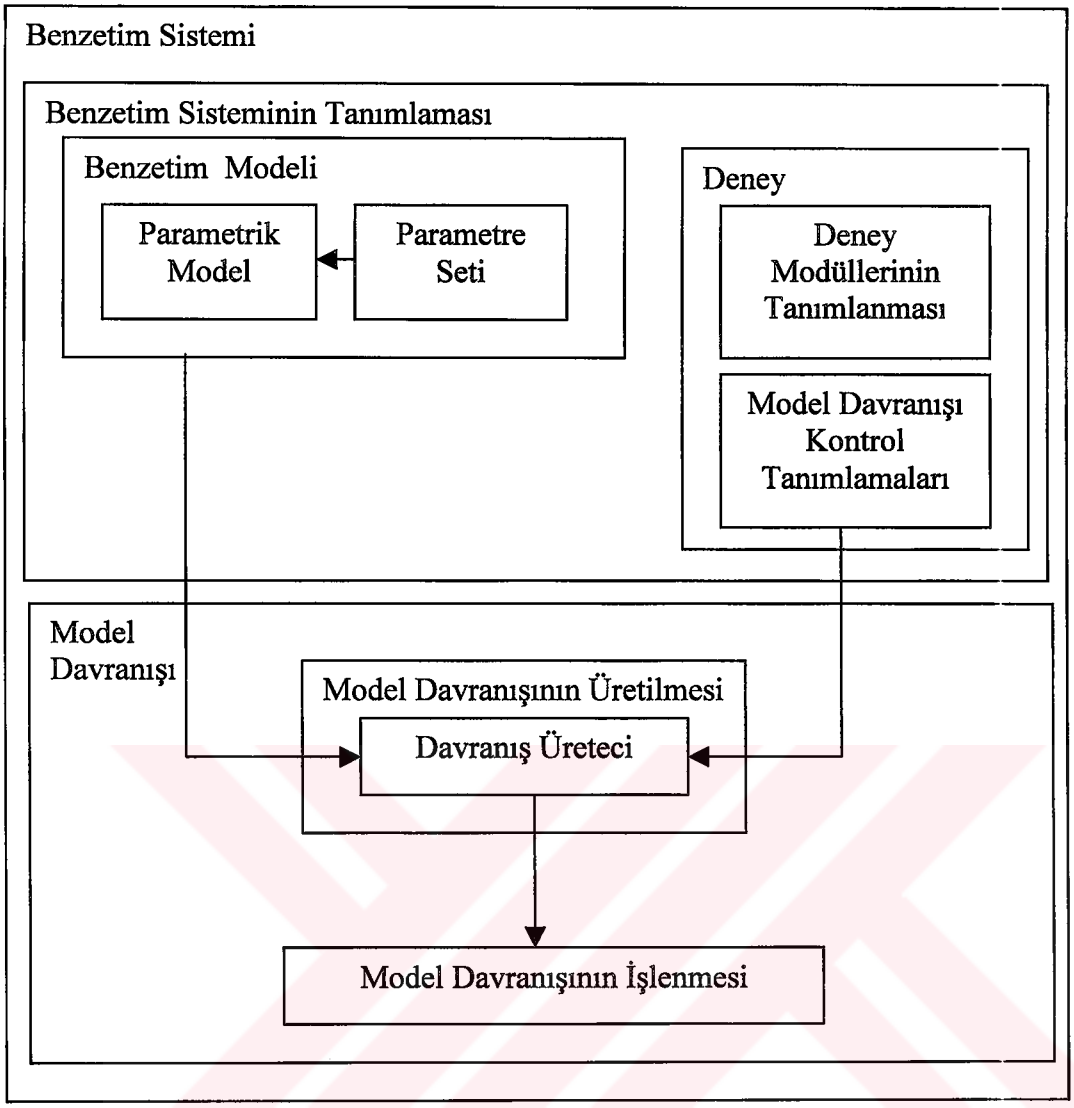
E: Deney koşulları ve davranış üretimi

P: Parametre setleri tabanı

O: Model davranışı ve çıkış

Buna göre, bir benzetim sisteminde, bir model üzerinde, farklı parametre değerleriyle, birden fazla deney gerçekleştirilebilir ve deney sonuçları, birden fazla ortama aktarılabilir.

Yukarıda anlatılanlar ışığında, bir benzetim sisteminin elemanları, Ören tarafından belirlenen yaklaşım çerçevesinde aşağıdaki şema ile gösterilebilir.



Şekil 2.3 Benzetim sistemi

Benzetim sistemlerinde kullanılan benzetim modelleri kullanım amaçlarına ve temsil ettikleri sistemin yapısına göre, genel olarak üç türde olabilir (Praehofer 1992). Bunlar,

- Fiziksel modeller,
- Grafik modeller,
- Bilgisayar ortamında çalıştırılabilen modeller

Fiziksel modeller, gerçek sistemin istenen ölçülere göre boyutlandırılmış şekli olup, sistem davranışını, "belirlenen objektivitelere göre", görsel olarak, gerçek sistem gibi, aynen yansıtırlar. Örneğin, maketler.

Grafik modeller, gerçek sistemin davranışının, "belirlenen objektivitelere göre", "resmedilmiş" şeklidir. Örneğin, bir ekonomik faaliyetin verilerinin, kağıt üzerinde grafiksel olarak yansıtılması.

Bilgisayar ortamında çalıştırılabilen modeller ya da bilgisayar modelleri. gerçek sistemin davranışının, tamamen, bilgisayar olanakları kullanılarak belirlendiği modellerdir. Bu modeller bilgisayar olanaklarını iki şekilde sunarlar. Bunlar,

- Donanım ve,
- Yazılım

Gerçek sistemlerin davranışlarını bilgisayar ortamından kontrol ederken, modelin ya da sistemin gözlenen her bir harareti bilgisayar devreleri kanalıyla, bilgisayar ekranına yansır. Anolog bilgisayar olarak adlandırılan bu tür modellere örnek olarak, radarlar ve uydu kontrol merkezlerinde kullanılan modeller verilebilir.

Bir modellin, bilgisayar ortamında, yazılım olanakları kullanılarak işlenmesinde, sistemin "dinamik" unsurlarının "algoritmik" yapılar içerdiği göz önüne alınır. Daha sonra, model, bir "matematik modele" dönüşür. Bu dönüşümde amaç, bilgisayar algoritmasının, bir matematik model yardımıyla karakterize edebilmektir. Örneğin, daha önce belirtildiği gibi, sürekli sistemlerin modellenmesinde, sistem, adi diferansiyel denklemlerin kullanıldığı bir matematik model ile karaktrize edilir. Adi diferansiyel denklemlerin analitik çözümü bulunamadığı taktirde, bilgisayar ortamında, "sayısal entegrasyon" yöntemi kullanılarak sonuca gidilir.

Bilgisayar ortamında, benzetim modellerinin işlenebilmesi iki kavramı gündeme getirir. Bunlar,

- Bilgisayar destekli benzetim ortamları
- Bilgisayar destekli modelleme, ve sistem teorisi

Bu iki kavram birbirine çok yakın, ancak farklıdır. Bundan sonra ele alınacak tüm modelleme ve benzetim kavramları ele alınırken, bilgisayarların sadece "yazılım olanakları" göz önüne alınacaktır.

2.2 Bilgisayar Destekli Benzetim Ortamlarının Genel Yapısı

Bir benzetim sisteminde, modelin ve benzetim deneyinin tanımlanması, ve daha sonra "benzetim deneyinin çalıştırılması" işlemlerinin bilgisayar ortamında gerçekleştirilmesi, "bilgisayar destekli benzetim ortamları" kavramını gündeme getirir. Bilgisayar destekli benzetim ortamlarında gerçek bir sistem üzerinde benzetim işlemi yapılırken, üç temel öğe göz önüne alınır. (Zeigler 1976). Bunlar:

- Gerçek sistem
- Model
- Bilgisayar

Gerçek sistem, daha önceden belirlenen perspektifler ışığında, gerçek verilere dayanarak, üzerinde deney yapılacak nesneler topluluğunu, model ise, bu verilerin "bilgisayar ortamında", işlenmesini ve sonuçların elde edilmesini sağlayan "deney düzeneğindeki" sistemin "temsilcisini" ifade eder.

Bilgisayar destekli benzetim sistemi, bilgisayar olanakları ile, bir benzetim modelinin ve deyinin tanımlanması, ve model üzerinde belirlenen koşullara uygun olarak, deney gerçekleştirilmesini sağlar. Bilgisayar ortamlarının sağladığı olanaklar, birçok işlevselliği beraberinde getirir. Bunlar,

- Tanımlamalar
- Benzetim programının üretilmesi
- Dosya işlemleri
- Grafik ortam olanakları
- Dokümantasyon

Benzetimin temel amacı, bir sistem modeli üzerinde, belli parametre değerlerine göre, deney gerçekleştirmek ve deney sonuçlarını dış ortama aktarıp, analiz işlemine olanak tanımaktır.

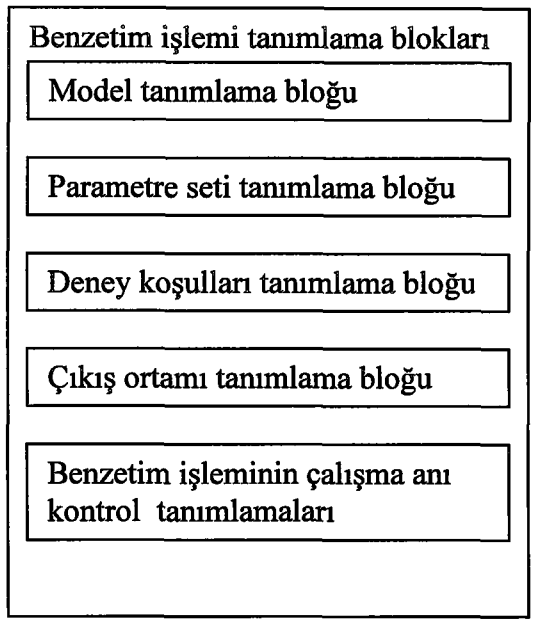
Bu bağlamda, benzetim işlemlerinde, modelin yapısının, modelin parametre değerlerinin, model davranışının genel özelliklerinin, deney koşullarının ve deney sonuçlarının dış ortama ne şekilde aktarılacağının benzetim sisteminde, "tanımlanması" gerekir.

Bilgisayar destekli benzetim ortamlarında, model, parametre değerleri, model davranışı, deney koşulları ve çıkış ortamları için yapılan bu tanımlamalar, "görsel bir kullanıcı arayüzü" ile "belli kurallara göre" yapılır. Bu kurallar bütününe, "benzetim dili" denir.

Benzetim sistemlerinde, benzetim dillerinin kullanılmasındaki amaç, benzetim işlemine ait model, parametre değerleri, deney koşulları ve diğer kontrol işlemlerinin, kullanılan dilin gramer yapısına göre, tanımlamasını yapmaktır. Bu tanımlama işlemi, yazılım sisteminin "kullanıcı arayüzü" yardımıyla, görsel olarak gerçekleştirilir. Yapılan bu tanımlamalar, yazılım sisteminin "dosya işleme" olanaklarına bağlı olarak, bir "dosya sisteminde" saklanır. Tüm tanımlama işlemleri "kullanıcı arayüzü" yardımıyla yapıldığından, kullanıcının benzetim dilinin gramer yapısını bilmesine gerek yoktur.

Bu avantaj kullanıcının yanlışlıkla, yazılım hataları yapmasını engellemekle kalmaz, aynı zamanda, deneyin model üzerinde uygulanıp uygulanamayacağını kontrol eder. Yukarıda bahsedilen, modelleme ve benzetim ortamlarında önemli bir yer tutan, "doğruluk" ve "geçerlilik" kontrolleri de otomatik olarak yapılır.

Benzetim dillerinin gramer yapıları, "modüler bir tanımlama ortamı" oluşturmaları bakımından çeşitli "bloklar" içerirler. Bu blok yapısı, genel olarak, aşağıdaki şemada gösterildiği gibidir.



Şekil 2.4 Benzetim dilinin blok yapısı

Bir benzetim dili ile yapılan tanımlama, "belirli" bir benzetim işlemi için, bir model, bir parametre seti, benzetim deneyi ve çıkış ortamının özelliklerini içerir.

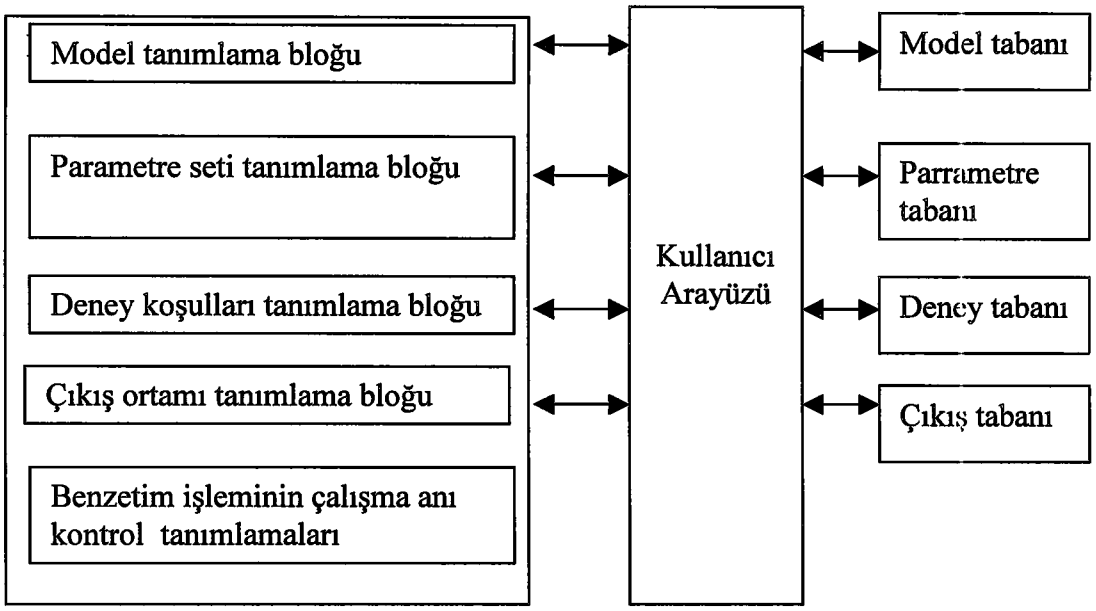
Bu tanımlamalar, arayüz vasıtasıyla, benzetim deneyi çalıştırılmadan önce, ilk anda yapılabileceği gibi, daha önce bahsedilen ve S=< M, E , P, O> şeklinde ifade edilen bir benzetim sisteminin, model, deney, parametre ve çıkış ortamları tabanları yardımıyla da yapılabilir.

Bilgisayar destekli benzetim ortamlarında, model, deney, parametre ve çıkış ortamları tabanları, yazılım sisteminin "dosya işleme" olanakları yardımıyla oluşturulur ve kullanılır. Her bir model, deney, parametre seti ve çıkış ortamı tanımlaması, benzetim dilinin gramer yapısına göre yapılır ve yazılım sisteminde, bir "dosya" ile temsil edilir.

Bu yapı, "bir model üzerinde, farklı parametre değerleriyle, birden fazla deneyin yapılabilmesi ve sonuçların farklı çıkış ortamlarına aktarılabilmesi" kavramını netleştirir.

Şöyle ki, modüler bir tanımlama ortamında her bir blok için ayrı bir tanımlama gerekeceğinden, gerekli tanımlamalar, "geçerlilik" ve "doğruluk" testleri olumlu

olmak kaydıyla, hem deneyi çalıştırılmadan önce, ilk anda, hem de dosya sistemi yardımıyla yapılır.



Şekil 2.5 Yazılım sisteminin kullanıcı arayüzü ve dosya sistemi arsındaki etkileşim

Şekil 2.5'de gösterildiği gibi, her iki durumda da kullanıcı arayüzün sağladığı esneklikten yararlanılır.

Benzetim dilleri kullanılarak yapılan bu tanımlamalar, daha sonra bir "program üretici" yardımıyla bir "üst düzey" programlama dili kullanılarak yazılmış program koduna dönüştürülür. Daha sonra, üretilen bu kod derlenip çalıştırılır.

"Benzetim programı" adı verilen bu program yardımıyla, yazılım sisteminin grafik ortamı sayesinde, modelin davranışı görsel olarak yansıtılır.

Bilgisayar destekli benzetim ortamları, sistem benzetimi ve bilgisayar destekli sistem mühendisliği projelerinde dokümantasyon imkanı da sağlar.

2.2.1 Model Tanımlama

Bilgisayar destekli benzetim ortamlarında, üzerinde benzetim deneyi gerçekleştirilecek olan modelin tanımlanması, bilgisayar ortamının sağladığı olanaklar yardımıyla yapılır. Bu olanaklar iki tür işlevi içerir. Bunlar,

- Kullanıcı arayüzü yardımıyla, modelin, benzetim sisteminin ilk tanımlandığı anda ve benzetim deneyinden önce tanımlanması,

- Modelin bir model tabanından elde edilmesi

Bir model üzerinde deney yapılabilmesi ve dolayısıyla, modelin temsil ettiği sistemin davranışının, bilgisayar ortamında belirlenebilmesi için, o modelin, sistem elementleri ve bu elementler arasındaki ilişiklileri ifade eden kurallar ve tanımlamalar bütününü içermesi gerekir. Sistem elementleri arasındaki ilişki bir "matematik model" yardımıyla belirtilir.

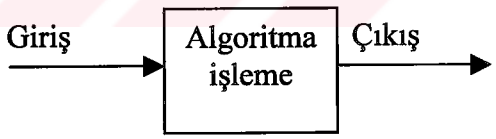
Bu matematik modelin içerdiği kurallar, bir "davranış üreteci" tarafından, bir bilgisayar algoritması ile işlenir.

Bir sistemin, birçok elementi olabilir. Bu elementler arasındaki ilişkiler, sistemin belirli seviyelerdeki, "işlevsel özelliklerini" yansıtırlar. Sistemin en küçük seviyede, yani en basit düzeydeki işlevini belirten model türüne "bileşen model" adı verilir. Bir bileşen model, sistemin en küçük seviyedeki "alt sistemini" temsil eder.

Bir sistem ve onu temsil eden model, en genel anlamda, sırasıyla,

- Giriş modülü,
- Durum geçiş ya da algoritma işleme modülü,
- Çıkış modülü,

elemanlarını içerir.



Şekil 2.6 Bir sistemin genel yapısı

Şekilde de gösterilen bu yapı, en basit düzeydeki, bileşen modeller için geçerlidir. Şekil 2.6'da şematik yapısı gösterilen bir modele ait "Giriş", sistem ya da alt sistem bazında düşünüldüğünde, bir sistemin, çalışma süresinde, "belli bir anda" dış ortamdan aldığı etkiyi temsil eder. Burada, "dış ortam", sistemin bünyesinde bulunan başka alt sistemleri ifade edebileceği gibi, sistemden tamamen farklı, başka sistemleri de ifade edebilir.

Matematik model açısından bakıldığında, bir bileşen modele bir t anında yapılan giriş, "giriş değişkeninin" aldığı değere göre belirlenir. Giriş değişkenleri, en genel anlamda, "belli bir t anında", bilgisayar algoritmasının veya bir giriş-çıkış işleminin başlangıcını karakterize eder ve t anındaki değeri "giriş fonksiyonları" tarafından belirlenir.

Giriş modülünün takip eden, "durum geçiş" modülü, bir bileşen modelin temsil ettiği sistem ya da alt sistemin, giriş anından, çıkış anına kadar geçen süre içinde, "belirlenen zaman aralıklarında", hangi durumda olduğunun belirlendiği modüldür.

Bu yapı, matematik model olarak, "durum değişkenlerinin", "t zaman parametresine" göre aldığı değerlerin belirlenmesinde kullanılan algoritmaları içerir. Örneğin bir sonraki bölümde açıklanacağı gibi, sürekli bir sisteme ait, herhangi bir elementin durumu, bir "y" değişkeni ile verildiğinde,

$$y=y(t);$$

$$y'=f(y,t) \qquad (2.1)$$

ifadeleri, bu durum değişkeninin, belli bir zaman aralığında, her bir t anında aldığı değeri elde etmek için kullanılan matematik model elementleridir. Bu matematik model üzerinde bir algoritma çalıştırılarak, diferansiyel denklemi karakterize eden y(t) fonksiyonunun her bir t anı için değeri belirlenir. "y" durum değişkeninin belli bir t anındaki değeri, sistemin, "y" değişkeni tarafından temsil edilen, o andaki durumunu sayısal olarak gösterir. y(t) fonksiyonunun belirli bir andaki değeri, diferansiyel denklemin "analitik çözümü" bulunarak ya da verilen zaman aralığında "sayısal entegrasyon" yöntemi uygulanarak bulunur.

"Çıkış", bir alt sistemin, belli bir t anında, sistemdeki diğer alt sistemlere, ya da tüm sistemin, diğer sistemlere yaptığı etkiyi ifade eder.

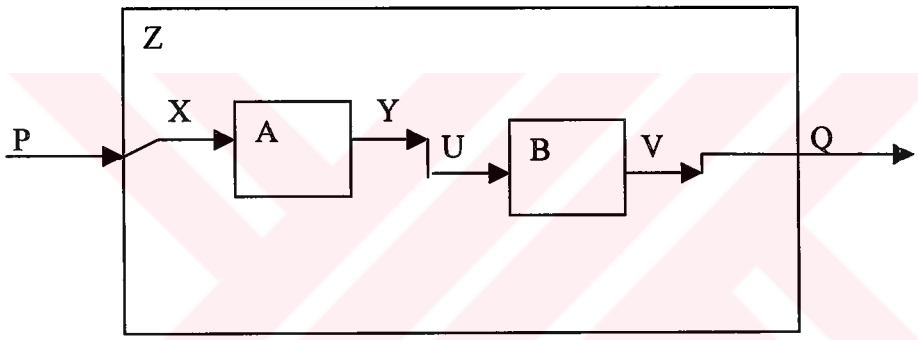
Matematik model olarak, bir t anındaki çıkış, o çıkışı temsil eden. "çıkış değişkeninin" değeri ile belirlenir. Bir çıkış değişkeninin bir t anındaki değeri, matematik modelin yapısına göre o andaki durum değişkenlerinin değerlerini, o değişkene ait "çıkış fonksiyonuna" parametre olarak verilmesi ile elde edilir.

Bir matematik modeldeki, bir takım matematiksel ifadeler "parametrik yapıda" olabilir. Bir ifadenin parametrik yapıda olması, matematik modelin uygulanacağı

algoritmanın, farklı parametre değerlerine göre, farklı sonuçlar üretmesini sağlar. Bir bileşen model üzerinde deney yapılırken(algoritma çalıştırılırken) deneyde kullanılacak parametre değerleri de, "parametre setleri" yardımıyla belirlenir.

Bir sistemi oluşturan alt sistemler arasındaki ilişki birbirleri, arasında giriş-çıkış bağlantıları ile sağlanır. Burada, bir alt sistemin çıkışı, diğer alt sisteme giriş olarak gelir. Bu şekilde, alt sistemleri arasında, giriş-çıkış bağlantıları oluşturulan sistemlere "bağlaşık sistem", bu tür sistemler için oluşturulan modeller de, "bağlaşık model" adı verilir. Bir bağlaşık sistemin alt sistemleri, "bileşen modeller " ile temsil edilir.

Her bir sistem, başka sistemlerin bir alt sistemi olabilir. Dolayısıyla, her bir bağlaşık model, başka bağlaşık modellerde, bileşen model olarak yer alabilir. Bu yapı, sistem hiyerarşisini gündeme getirir.

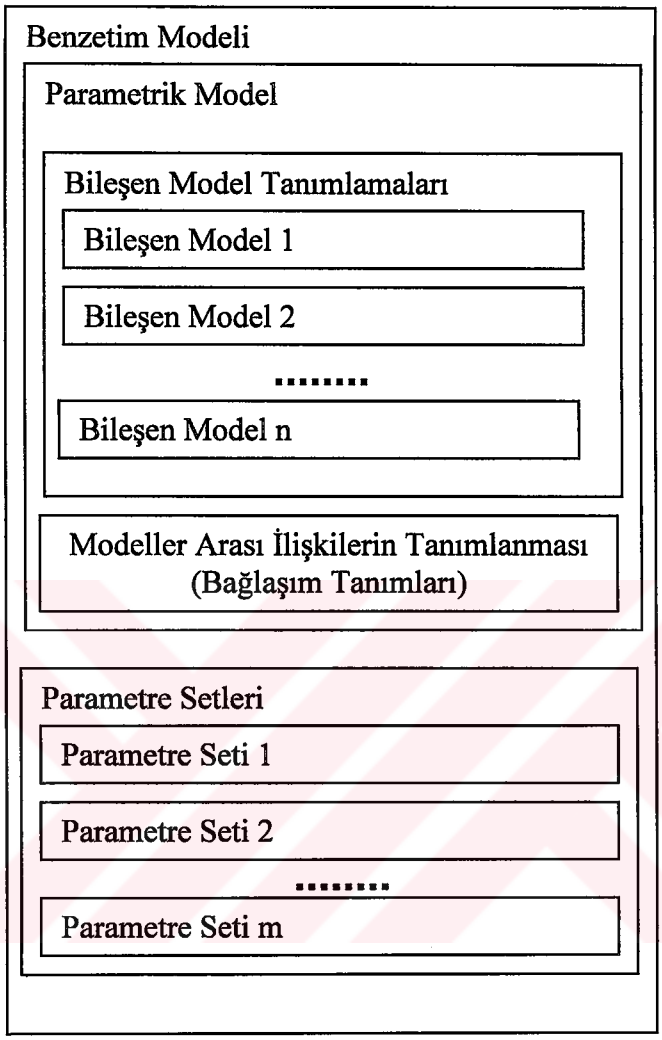


Sekil 2.7 Bağlaşık model

Şekilde, Z bir bağlaşık sistem olup, sırasıyla A ve B sistemleri, Z sistemine ait sistemdir ve en basit düzeyde bileşen modeller ile temsil edilirler. Burada P ve Q, Z sisteminin (modelinin) giriş ve çıkışlarıdır. Z sistemine ait bir bağlaşık model oluşturulurken, Z sisteminin P girişi, A sisteminin X girişine, A sisteminin Y çıkışı, B sisteminin U girişine ve B sisteminin V çıkışı da Z sisteminin Q çıkışına bağlanır Bir benzetim modeli, en genel anlamda, bir ya da birden fazla bileşen modeli olan, dolaysıyla, bağlaşık model özelliği gösteren bir "parametrik model" ve her bir bileşen model için tanımlanan "parametre setlerini" içerir.

Bir parametrik modelin tanımlanmasında, sırasıyla her bir bileşen modelin ayrı ayrı tanımını ile bu bileşen modellerin aralarındaki giriş-çıkış ilişkilerinin tanımları

yapılır .Aşağıdaki şemada bir parametrik model ve parametre setlerinin tanımlama yapıları gösterilmiştir.



Sekil 2.8 Parametrik model

Bir benzetim modelinde bulunan parametre setlerinin içerdiği bileşen modellere ait parametre değerleri, deney öncesinde, tanımlanabileceği gibi, bir parametre tabanından da elde edilebilir. Deney öncesinde yapılan tanımlamalarda, verilen değerlerin modele uygun olup olmadığı benzetim sistemi tarafından kontrol edilir.

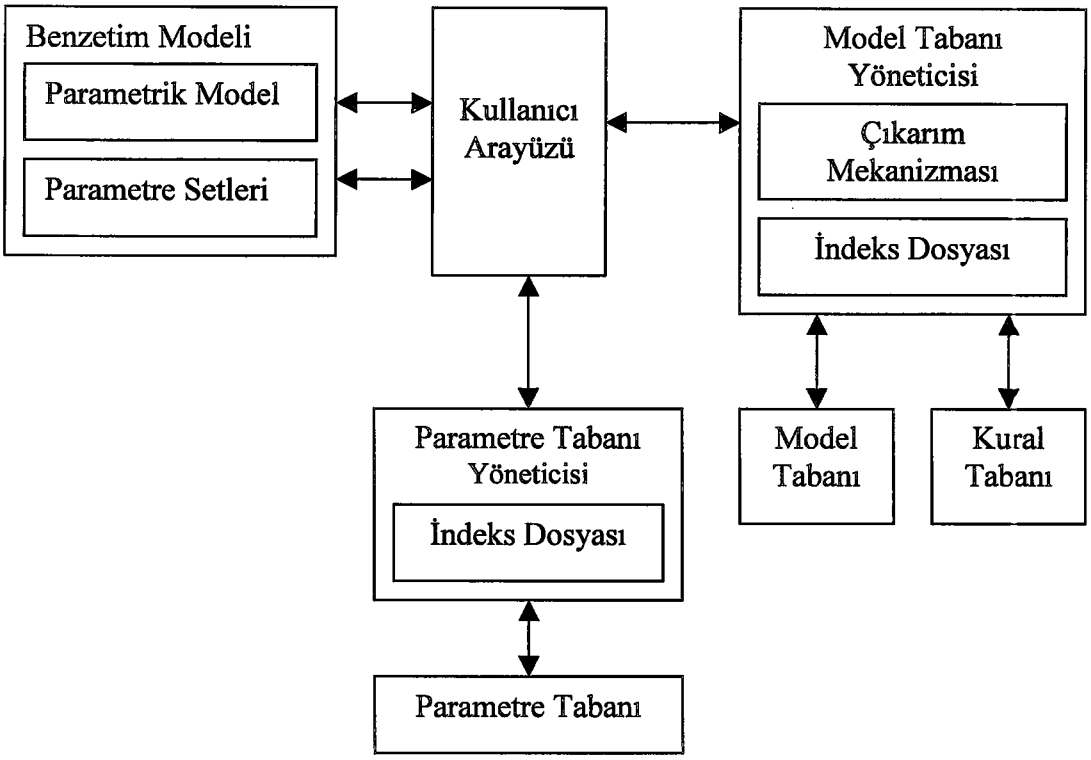
Bir sistemin davranışı belirlenirken ya da sistem problemi üzerinde çözüm geliştirilirken, probleme ait çözüm, yeni bir model oluşturmak yerine, çözüm için gereken model, bir "model tabanından" elde edilebilir.

Benzetim yoluyla, bir sistem hakkında bir strateji belirlenirken, önce benzetimde kullanılacak modele ilişkin objektiviteler belirlenir. Daha sonra, belirlenen objektivitelere göre, model tabanından uygun model seçilerek, benzetim sistemine konumlandırılır. Model tabanından alınan model tam olarak sistemin ilgili bileşenini temsil edemiyorsa, bu model üzerinde birtakım değişikliler yapılarak, benzetim ortamına uygun hale getirilir. Eğer, model tabanı, istenilen modeli içermiyorsa, yukarıda anlatılanlar ışığında, yeni bir model oluşturulur.

Bilgisayar ortamında, bir model tabanına ait her bir model, ve parametre tabanında yer alan her bir parametre seti, sırasıyla bir "model dosyası" ve bir "parametre seti tanımlama dosyası" ile belirtilir.

Bir model dosyası, eğer temsil ettiği model, bir bileşen model ise, o model ile belirtilen "matematik model ait" tanımlamaları, eğer temsil edilen model, bağlaşık model ise, o bağlaşık modele ait bileşen modellerin, "model isimlerini, model dosya isimlerini, girş-çıkş değişkenlerinin isimlerini ve birbirleri ile olan giriş-çıkş ilişkilerini ifade eden bağlaşım tanımlamalarını" içerir.

Bir parametre seti tanımlama dosyası da, bir benzetim modelindeki parametrik modele ait bileşen modellerin parametre değerlerini içerir. Dosyalara erişim, "model tabanı yöneticisi yöneticisi" ve "parametre tabanı yöneticisi" adı verilen yazılım modülleri ile sağlanır. Bu modüller, dosya sisteminde (model ve parametre tanımlamalarını içeren veritabanında), ekleme, silme ve güncelleme işlemlerini gerçekleştirirler.



Şekil 2.9 Parametrik modelin tanımlaması

Şekil 2.9'da gösterilen şemada model tabanı yöneticisinin, aranan bir modelin seçimini, bir "kural tabanı" yardımıyla gerçekleştirdiği ifade edilmiştir. Ancak, bu tezde tasarlanan sistemde, model tabanı yöneticisinin işlevi, sadece dosya işleme olanaklarıyla sınırlandırılmıştır. "İndeks dosyası", model ve parametre seti tanımlama dosyalarını isimlerini ve dosyada belirtilen model veya parametre setinin ismini içerir. Model tabanı kavramı "genel sistem teorisi" çerçevesinde bir sonraki bölümde ayrıntılı olarak ele alınacaktır.

2.2.2 Deney Tanımlama

Bilgisayar destekli benzetim sistemlerinde, bir benzetim modelini benzetim deneyine tabi tutmak için, bir parametrik model ve ilişkili parametre seti tanımlandıktan sonra deneyin sistematik olarak tanımlaması gerekir.

Daha önce de belirtildiği gibi bir parametrik model, bir ya da birden çok bileşen model ve bunların giriş-çıkış ilişkilerinin tanımlamalarını içerir. Bir bileşen model, en alt seviyede, bir sistemin, elementleri ve bunlar arasındaki ilişkileri bir matematiksel model ile yansıtır. Bu matematiksel model, bir davranış üreteci tarafından bir algoritma ile işlenir ve çıkan sonuçlar, analiz edilerek sistem hakkında bir sonuca varılır.

Deney tanımlama işlemi, modüler bir yapıda olup, her bir deney modülünde,bileşen modellerin içerdiği matematik modelin işlenmesinde kullanılan algoritmanın tanımlama ve başlangıç koşullarını belirtmek için yapılır. "Her bir bileşen model için" yapılan bu tanımlamalar, "bileşen modellere ait deney modülü" adı verilen tanımlama bloklarında belirtilir. Dolayısıyla, bir parametrik model için yapılan deney tanımlaması, parametrik modelde bulunan, tüm bileşen modeller için yapılan deney modülü tanımlamalarını ve "genel deney modülünü" içerir. Genel deney modülü, tüm deney modüllerinin ortak tanımlamalarını kapsar. Bunlar;

- Algoritmada kullanılan yöntem. Örneğin; sayısal entegrasyon işlemi için, Runge-Kutta yöntemi gibi...,
- Benzetim deneyi için belirlenen zaman aralığı,

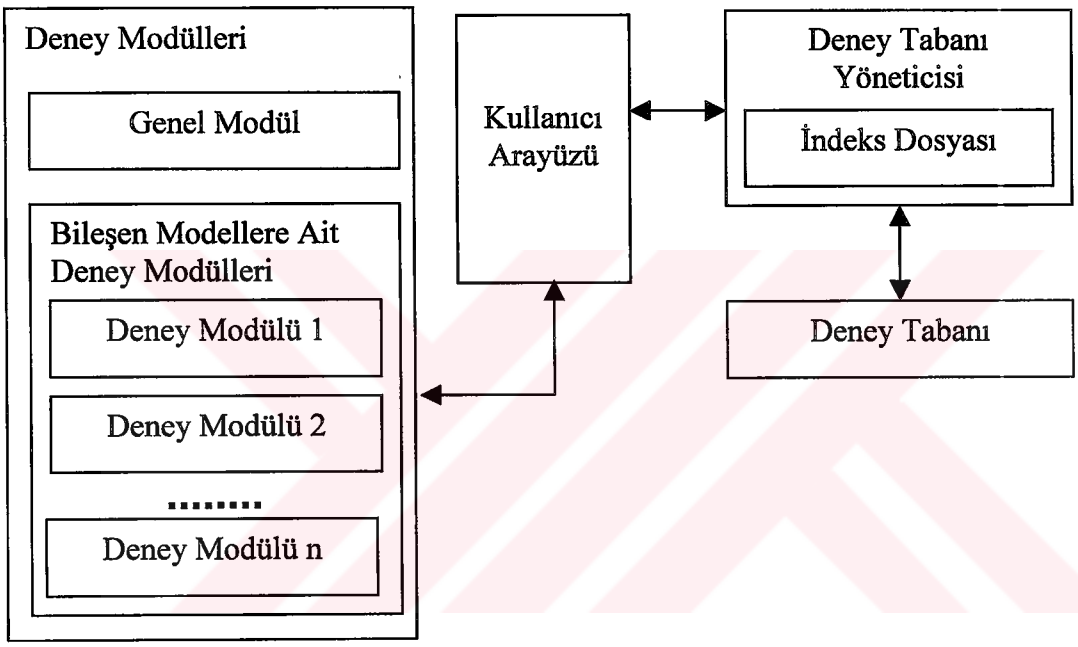
olarak sıralanabilir.

Ören ve Zeigler'e göre, bir algoritmanın tanımlama ve başlangıç koşullarını belirtmek için tanımlanan bir bileşen modele ait deney modülü şu tanımlamaları içerir:

- Durum değişkenlerinin ilk değerlerini
- Modelin verilen zaman aralığında, alabileceği giriş değişkenlerinin en küçük ve en büyük değerleri (Bu değerler, değişkenlerin tanımlanması sırasında da verilebilir. Ancak bu tezde, giriş değişkenlerinin en büyük ve en küçük değerlerine ilişkin kontroller yapılmaz.)
- Deneyin hangi koşullarda sona ereceği

- Deneyin sonunda grafik ortamda görülme istenen değişkenler. Bunlar, sadece çıkış değişkenleri değil, aynı zamanda, giriş ya da durum değişkenleri de olabilir.

Bir önceki bölümde de belirtildiği üzere, bir model ya da parametre seti, deney öncesinde, tanımlanabileceği gibi, bir model ya da parametre tabanından da elde edilebilir. Aynı durum deney modülü tanımlamaları içinde geçerlidir. Benzetim deneyini her defasında yeniden tanımlamak yerine, deney tabanından gerekli deney modülleri, "deney tabanı yöneticisi" adı verilen modül yardımıyla belirlenir ve deney yapılır. (Şekil 2.10)



Şekil 2.10 Deneyin tanımlaması

2.2.3 Program Üretimi

Bilgisayar destekli benzetim sistemlerinde, model tanımlama, parametrik model tanımlama, parametre seti tanımlama, deney tanımlama ve deney sonuçlarına ait çıkış tanımlamaları, bilgisayarın sağladığı, dosyalama (veritabanı) ve etkin kullanıcı arayüzü sayesinde gerçekleştirilir. Bu tanımlamaların bilgisayar ortamında gerçekleştirilmesi, temelde iki amaca yöneliktir. Bunlar; tanımlama esnasında, yapılan tanımlamanın, "doğru (yazım hataları bakımından)" ve "geçerli" olup olmadığının kontrolü ve varolan herhangi bir tanımlamanın yeniden kullanılabilmesi.

Bu tanımlamalar, daha önce ifade edilen ve benzetim dili adı verilen kurallar bütününe göre, benzetim diline ait ilgili tanımlama bloklarında yer alır.

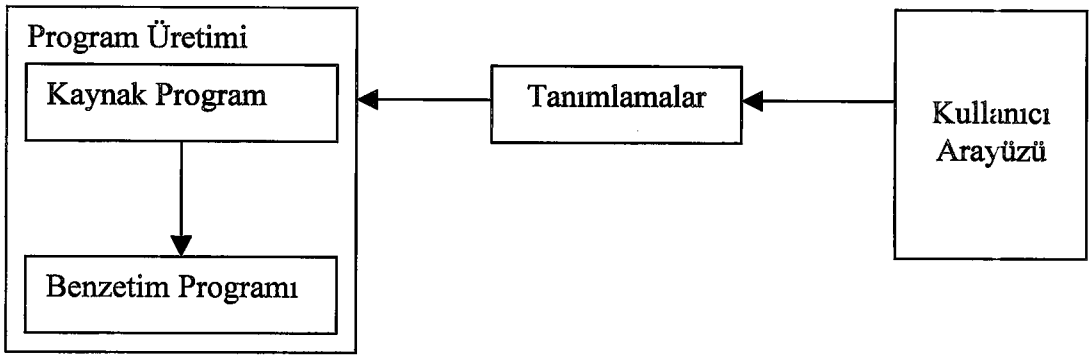
Bu tanımlara göre, bir benzetim modeli, parametrik model ve parametre setleri ile belirtilir. Parametrik modelde tanımı yapılan her bir bileşen model, bir matematik model içerir. Bu matematik modelin işlenmesi ve bileşen modellerin aralarındaki ilişkiye göre, giriş-çıkış akışının gerçekleştirilmesi, bir "benzetim algoritmasının", bir "benzetim programı" yoluyla çalıştırılmasını gerektirir.

Buna göre, bir "benzetim işlemi", "tanımlamaların yapılması" ve "benzetim programını çalıştırılması" işlemlerini kapsar.

Bilgisayar ortamında , kullanılan benzetim diline uygun olarak, yapılan tanımlamaları, bir dosya oluşturarak, bu dosyaya yazar. Bu dosyada yapılan tanımlamalar, "kaynak program" adını alır Daha sonra bu kaynak program, bileşen model tanımlamalarında yer alan matematik modeli işlemek için geliştirilen algoritmayı çalıştıran ve üst düzey bir programlama dili (C++ veya Java gibi) ile yazılmış program koduna dönüşür. Bu koda, "hedef program" ya da "benzetim programı" adı verilir. (Şekil 2.11)

Benzetim programı üretilirken "çalışma anı kütüphaneleri" adı verilen ve "nesne tabanlı" programlamada, "yeniden kullanılabilen sınıflar" olarak adlandırılan yazılım elementleri de kullanılır. Bunun nedeni, benzer algoritma yapılarını çalıştıran kodların tekrar üretilmesini önlemektir. Benzetim programı üretildikten sonra, bu programın, "derleme" ve "link" işlemleri gerçekleştirilir. Derleme ve link anında, "link editörü" adı verilen arayüz elmentleri, üretilen progrm dosyaları hakkında bilgi verir

Benztim programının üretilmesi, modelin yapısına bağlıdır. Her yeni model için, yeni bir benzetim programı üretilir. Yani, model değiştikçe, benzetim programı değişir. Kaynak programın içerdiği, parametre seti, deney koşulları, çıkış ortamlarının yapısı ve çalışma anı kontrol tanımlamaları, verilen modele göre üretilen modelin çalışma şartlarını belirler.

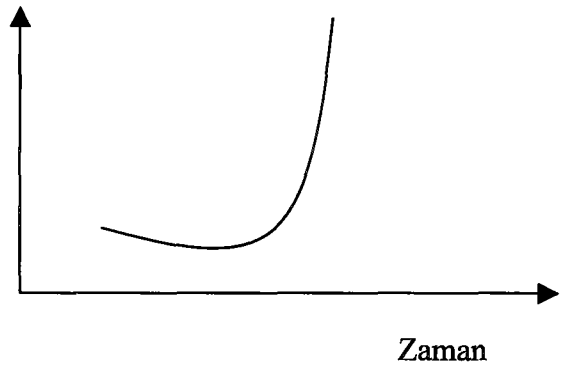


Şekil 2.11 Program Üretimi

2.2.4 Deneyin Yapılması ve Sonuçlar

Yapılan model tanımlamasına uygun olarak üretilen benzetim programı, yukarıda belirtildiği gibi, bir parametrik modelde yer alan bileşen modellere ait matematik modelleri işleyen algoritmanın, bilgisayar ortamında çalıştırılmasını sağlar. Buna göre, verilen parametre seti ve deney koşullarına uygun olarak, "model davranışının üretilmesi" sağlanır.

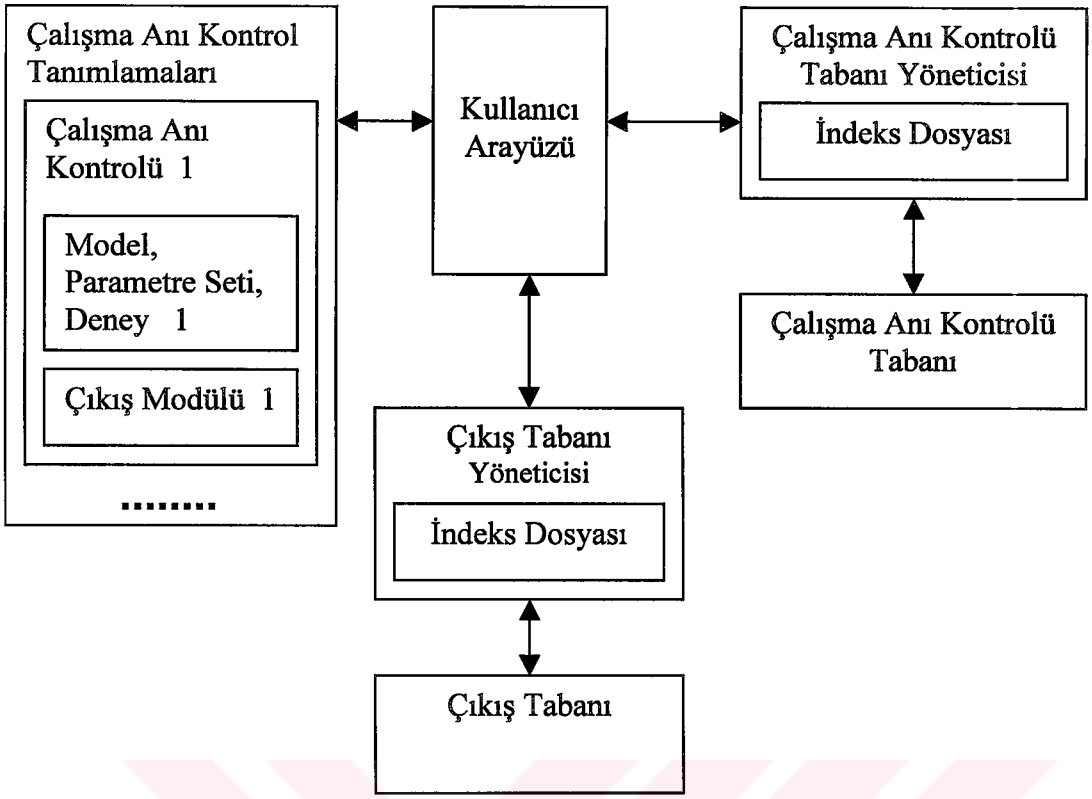
Benzetim programının çalıştırdığı algoritmanın temel işlevi olan model davranışının üretilmesinde, modelin yapısına bağlı olarak, giriş, durum ve sonuç değişkenlerinin, ya da sadece durum ve sonuç değişkenlerinin, verilen zaman aralığında, her t anı için almış oldukları değerler hesaplanır. Hesaplanan bu değerler, iki boyutlu değişken-zaman ekseninde grafik olarak gösterilir. (Şekil 2.12)



Şekil 2.12 Değişken-zaman ekseni

Benzetim programı çalıştırılma şekli, "çalışma zamanı kontrol tanımları" ve bu tanımların içerdiği, "çıkış modülü tanımlamalarına" göre belirlenir. Çalışma zamanı kontrol tanımları, bir parametrik model ile, bir parametre seti ve bir deney modülleri ailesini (her biri parametrik modelde yer alan bileşen modeller için tanımlanmış) eşleştirir. Çıkış modülleri tanımlamaları da hangi değişkenlerin grafik çıkış ortamına aktarılacağını belirler.

Bu tanımlamalar, model, parametre seti ve deney tanımlamalarında olduğu gibi, arayüz vasıtasıyla, benzetim programı çalışmadan önce, ya da bir "çalışma anı kontrol yöneticisi" ve "çıkış yöneticisi" yardımıyla, veritabanından yapılır. (Şekil 2.13)



Şekil 2.13 Benzetim programı çalışma anı ve çıkış işlemi tanımlamaları

2.3 Benzetim Ortamlarının Türleri

Bilgisayar destekli benzetim ortamları, farklı görüş açılarına göre sınıflandırılabilir. Bu bölümde, Ören tarafından yapılan sınıflandırma kısaca ele alınmıştır.

Bu sınıflandırmada göz önüne alınan görüş açıları şunlardır:

- Kullanılan benzetim dili,
- Bilgisayar ortamının yapısı,
- Benzetimin uygulama alanı,
- Modelin türü,
- Modelin davranışı

Kullanılan benzetim dili, bir benzetim sisteminde var olan, model, parametre seti, deney koşulları ve çıkış ortamlarına ilişkin tanımlamaların yapıldığı kurallar bütünüdür. Bilgisayar ortamında, yapılan bu tanımlamalara göre, bir model üzerinde benzetim deneyinin gerçekleştiren programın, oluşturulması ve çalıştırılması benzetim dilinin "işlemci" yapısına bağlıdır. Benzetim dilinin kullanıldığı benzetim

ortamı, bir "yorumlayıcı" içerebilir. Bu durumda, benzetim dilinin gramer yapısına göre yapılan tanımlamalar, yorumlayıcı tarafından yorumlanarak, benzetim deneyinin gerçekleştiren program çalıştırılır. Benzetim ortamı, yapılan tanımlamaları, icra edilebilir (*.exe) bir programa dönüştüren bir "derleyici" içerebilir. Bu durumda yapılan tanımlamalar, üst düzey bir programlama dili gibi derlenip çalıştırılır. Son olarak, benzetim ortamı, yapılan benzetim tanımlamalarından, modele ilişkin tanımlamaları, üst düzey bir programlama dili ile yazılmış bilgisayar programına dönüştürür. (Bölüm 5)

Bilgisayar destekli benzetim ortamlarına ilişkin sınıflandırma, kullanılan bilgisayar sisteminin kullanıcıya sunduğu donanım ve yazılım olanaklarına göre de yapılabilir. Benzetimde kullanılan bilgisayarlar, benzetim deneyinde dışarıdan veri alış şekillerine göre sınıflandırılır Buna göre, bazı bilgisayar sistemleri,ses, kamera ya da daha farklı türde veri alabilir. Yazılım olarak ele alındığında, benzetim ortamında benzetim ortamının, görsel modelleme ve benzetim işlemlerin birbirinden ayrı iki işlem olarak ele alınması ve yazım hatalarnın bulunup düzeltilmesine göre bir sınıflandırma yapılabilir. Benzetimin uygulama amaçlarına göre, genel ya da özel amaçlı olarak kullanılabilir

Benzetim modelleri ve dolayısıyla temsil ettikleri sistemler, verilen zaman aralığında belleksiz, ayrık ya da sürekli davranış gösterirler. (Bölüm 3.3)

Sistemlerin bu şekilde farklı davranış göstermesi ve zaman aralığının da ayrık ya da sürekli olması, kullanılan model formülasyonunu dolayısıyla modelleme ve benzetim ortamının farklı türde olmasını sağlar.

Benzetim deneyinde, modelin davranışı verilen deney koşullarına göre üretilir ve işlenir. Model davranışının üretilmesiyle sistem hakkında bilgi edinilir. Modelin benzetim deneyinde gösterdiği davranış, verilen zaman aralığında, sırasıyla, yörüngesel, yapısal ve noktasal düzeyde olabilir. Bu da benzetim ortamının yapısını etkiler. (Bölüm 6)

3 MODEL VE MODELLEME

3.1 Genel Sistem Teorisi

Genel sistem teorisi, tabiattaki tüm sistemlerin, genel olarak, "aynı kurallar bütününe" uyduğunu kabul eder. Bu yaklaşım, fiziksel özellikleri, davranış şekilleri ve kullanım amaçlar birbirinden farklı olan sistemlerin, benzer kurallar ve aksiyomlara göre formüle edilebileceğini belirtir. Sistem teorisinin ortaya koyduğu bu yaklaşım, "sistem analiz ve tasarımı" kavramlarını, "model oluşturma" ve "problem çözme" kavramlarına dönüştürür.

Sistem kavramı, birçok disiplin tarafından içerilir. Sistem teorisinin genel amacı, disiplinler arası farklılık gösteren sistem kavramını, ortak bir noktada toplamaktır. Sistem teorisi çerçevesinde, bir sistem, göz önüne alınan perspektife göre iki farklı şekilde yorumlanabilir:

- Bir sistem, birçok elementinin birbiriyle ilişkili olduğu, belli bir andaki durumunun, tüm elementlerinin o andaki durumlarına göre belirlendiği ve dışarıdan aldığı etkilere yanıt veren ve başka sistemleri de etkileyen bir yapı teşkil eder.
- Bir sistem, genel modelleme kavramına uygun olarak, modellenebilirlik kurallarına uyar.

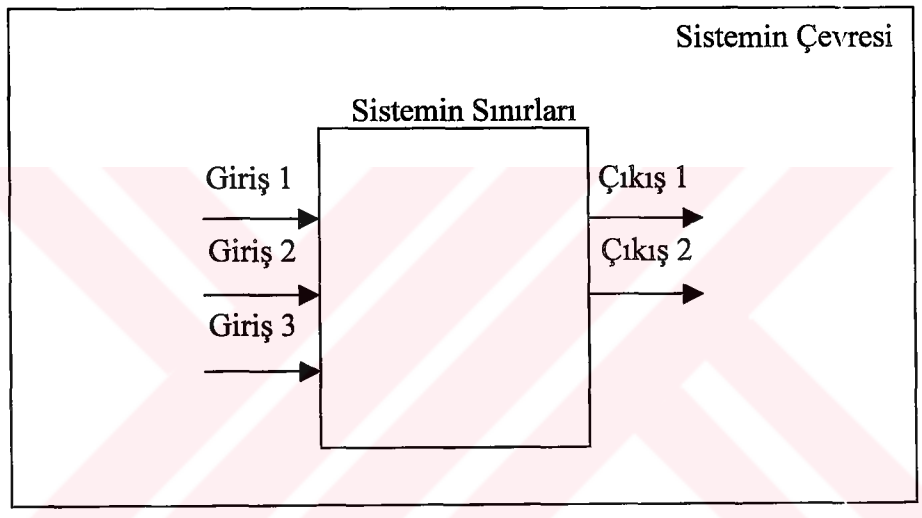
Yukarıda yapılan ilk yoruma göre, bir sistemde, birbiriyle ilişkili birden fazla element, "hiyerarşik" bir yapıda, belli bir amaca yönelik olarak bir arada olabilir. İkinci tanım ise, bir sistemin, üzerinde benzetim deneyi gerçekleştirilebileceği, bu nedenle, belli kurallar göre, modelinin oluşturulabileceğini ifade eder.

Bir sistemin elementleri, o sisteme ait alt sistemler olup, kendileri de, başlı başına bir sistem olarak ele alınabilir. Buna göre, bir sistemin elementleri aralarındaki ilişki, birbirinden bağımsız, farklı yapıdaki sistemlerin aralarındaki ilişkiye dönüşür. Bir sistem, göz önüne alınırken, sistemin, sadece belirli bir amaca yönelik olduğu ve

sistem hakkında yapılan incelemenin, belirli bir ölçekte olması gerektiği kabul edilir. Bu ölçeklendirme, "sistemin sınırları" ile belirlenir.

Sistem teorisine göre, "çevresiyle" etkileşim halinde olan bir sistem, bu etkileşimi, giriş-çıkış ilişkileriyle sağlar. Bir sistemin, belli bir anda, dış ortamdan aldığı etki, o anda, sisteme yapılan "girişi", sistemin dış ortama verdiği etki ise o andaki "çıkış" ifade eder. Bir sisteme yapılan giriş ve çıkış sayısı birden fazla olabileceği gibi, giriş ve çıkış sayıları eşit olmak zorunda değildir.

Bir sistemin çevresi, sınırları ve belli bir andaki giriş ve çıkışları, o sistemin dış ortam kavramlarını ifade eder.



Şekil 3.1 Sistemin dış ortam kavramları

Yukarıda yapılan ikinci tanımlamaya göre, modeli oluşturulan bir sistem üzerinde, benzetim ortamında deney gerçekleştirilirken, sistemin "en basit düzeydeki", alt sistemlerinin, dış ortam kavramlarının yanında, iç ortamlarının yapısı da göz önüne alınır.

En basit düzeydeki bir sistem ya da alt sistem göz önüne alındığında, sisteme yapılan giriş ve çıkış arasında kalan süreçte, sistemin hangi durumda olduğu "durum değişkenleri" ile belirlenir. Bir sisteme belli bir anda giriş yapıldığında. durum değişkenin alacağı değer, "durum geçiş fonksiyonu" tarafından hesaplanır. Durum

geçiş fonksiyonuyla hesaplanan bu değer, zaman, sistem parametre değerleri ve o andaki "giriş değişkeninin" değerine bağlıdır. Belli bir andaki "çıkış değişkeninin" değeri, "çıkış fonksiyonu" ile belirlenir. Durum geçiş fonksiyonu ve çıkış fonksiyonu bir sistemin "dinamik yapısını" oluştururlar.

Bir sistemin tanımlanması, kullanılan "sistem tanımlama yaklaşımına" göre değişir. Zeigler tarafından önerilen yaklaşımda, sistem teorisine göre, sistemin tanımlaması "farklı seviyelerde yapılır. Bu seviyelendirme, "en soyut tanımlamadan, en somut tanımlamaya kadar", gittikçe, tanımlama işlemi, daha ayrıntılı olarak ele alınır. Aşağıda, bu tanımlama seviyeleri açıklanmıştır.

Seviye 0 <T, X, Y>

Bu seviyede, sistemin üzerinde yapılan inceleme ve deneyin zaman aralığı (T) verilir ve sisteme girişler yapıldığı (X) ve sistemden çıkış alındığı (Y) kabul edilir.

Seviye 1 <T, X, Y, R>

Giriş ve çıkışlar arasında bir R bağıntısı kurulur.

Seviye 2 <T, X, Y, F>

Seviye 1'deki R bağıntısı, modüler bir şekilde parçalanarak, $F=(f_1, f_2,.....f_n)$ fonksiyon vektörüne dönüşür. Buradaki her bir f_i fonksiyonu, belli giriş değişkenleri için, "bir çıkış değeri" hesaplar.

Seviye 3 <T, X, Y, F, Q, G>

Sistemin iç yapısının ele alındığı bu seviyede, G durum geçiş fonksiyonu ile belli bir andaki, durum değişkeninin (Q) değeri hesaplanır.

Seviye 4

X, Y ve Q ile belirtilen, giriş, çıkış ve durum değişkenlerinin, birden fazla değişkeni ifade eden birer vektör oldukları kabul edilir.

Seviye 5 <T, X, Y, Bileşen (alt) sistemler>

Bileşen sistemler arasındaki giriş-cıkış tanımlamaları yapılır.

3.1 Model ve Modelleme Kavramı

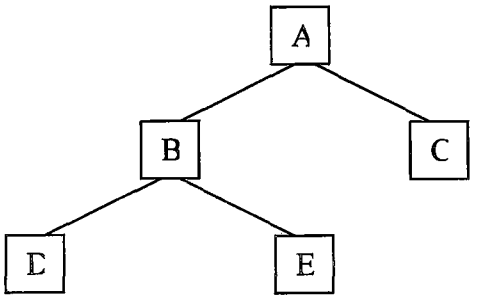
Daha önce de belirtildiği gibi, bir sistem üzerinde benzetim deneyi gerçekleştirebilmek için, sistemin, deney ortamında temsil edilmesi gerekir. Bunun yolu da o sistemin, "belirlenen deney objectivitelerine" göre modelini oluşturmaktır. Buradaki benzetim deneyi kavramı, bir sistem probleminin çözümünde, "analitik" çözüm bulunamadığı taktirde uygulanan yöntemdir. Eğer, analitik çözüm varsa, bu çözüm, belirlenen ilkeler doğrultusunda, sistemin gözlenmek istenilen kısmının, "sembolik" anlamda ifadesini, dolayısıyla modelini teşkil eder Her iki durumda da, oluşturulan model, bir sistemin elementleri arasındaki ilişkileri, belirlenen objectivitelere göre yansıtır.

Bilgisabenzetim ortamlarında, oluşturulan modeller, "işlevsellik seviyeleri" açısınyar destekli dan ikiye ayrılır. Bunlar;

- Bileşen ya da atomik modeller,
- Bağlaşık modelle,

Burada bahsedilen "işlevselsellik seviyesi" kavramı, modelin temsil ettiği, sistem elementleri arsındaki ilişkinin, hangi düzeyde olduğunu ifade eder.

Şöyle ki, sistemin belirli bir işlevi, hiçbir alt sistem içermeyen dolayısıyla, en "atomik düzeyde" alt sistemler tarafından yerine getiriliyorsa, bu tür sistemler, bileşen ya da atomik modeller ile temsil edilir. Eğer, sistemin belirli bir seviyedeki işlevini yerine getiren sistem ya da alt sistem, birden fazla alt sistem içeriyorsa, yani, yerine getirilen işlev, alt sistemlerin işlevlerimin toplamı şeklinde düşünülüyorsa, bu tür sistemler, bağlaşık modeller ile ifade edilir.
Bu yapı, sistemin hiyerarşik yapısını ifade eder.



Şekil 3.2 Sistem hiyerarşisi

Yukarıdaki sistem hiyerarşisinde, A ana sistem olup, B ve C sistemleri, A sisteminin alt sistemlerini teşkil eder. B ve C sistemlerinin yerine getireceği işlevlerin toplamı, A sisteminin belirli bir andaki işlevine denktir. Benzer şekilde, D ve E sistemleri de B sisteminin alt sistemleridir ve yerine getirdikleri işlevlerin toplamı, B alt sisteminin işlevine denktir.

Bir atomik sistemin, en alt seviyede, belirli bir işlevi yerine getiriş şeklini, bilgisayar ortamına yansıtabilmek için, bu sistemin davranışını karakterize eden bileşen modelin, bir matematik model olduğu kabul edilir. Bu matematik model, bir davranış üreteci tarafından, bir algoritma ile işlenir.

Bu algoritma, bir benzetim deneyinde, belirlenen zaman aralığında, her t anı için tekrarlanan bir döngü yapısı içerir. Bu döngüde, her t anı için sırasıyla, şu adımlar izlenir :

- Girişlerin dış ortamdan alınması,
- Durum geçiş fonksiyonu ile durum değişkenlerinin o andaki değerinin hesaplanması
- Çıkış geçiş fonksiyonu ile çıkış değişkenlerinin o andaki değerinin hesaplanması

Bir bileşen model ile temsil edilen sisteme, belli bir t anında yapılan girişler, o anda, dış ortamdan aldığı etkileri ifade eder. Bu girişlerin sayısal değeri, giriş değişkenlerinde tutulur. Bu değerler, belli bir t anında, giriş fonksiyonunun aldığı değere göre ya da başka sistemler ile olan giriş-çıkış ilişkilerini ifade eden bağlaşım yoluyla giriş değişkenlerine atanır.

Bu tür sistemlerde, sistemin belli bir t anındaki durumu, durum değişkenlerinin aldığı değere göre belirlenir. Durum değişkenlerinin belli bir t anı için değeri, başlangıç

değerleri verilmek koşuluyla, bir iterasyon algoritmasyla hesaplanır. Bu algoritmada kullanılan iterasyon fonksiyonu, durum geçiş fonksiyonudur. Durum geçiş fonksiyonu, durum değişkenin herhangi bir t anı için değerini, giriş ve durum değişkenlerinin bir önceki değerlerine ve sistemin parametre değerlerine göre hesaplar.

Durum geçiş fonksiyonun matematiksel formülasyonu, modelin türüne göre değişir. Örneğin, bir sonraki bölümde de açıklanacağı gibi, sürekli bir modelde, durum geçiş fonksiyonu, "y'=f(y, t)" diferansiyel denklemindeki "f " fonksiyonu olup, iterasyon algoritması, bir sayısal entegrasyon algoritmasıdır. Her entegrasyon adımında, o andaki giriş değerleri, diferansiyel denkleme sabit değer olarak girer.

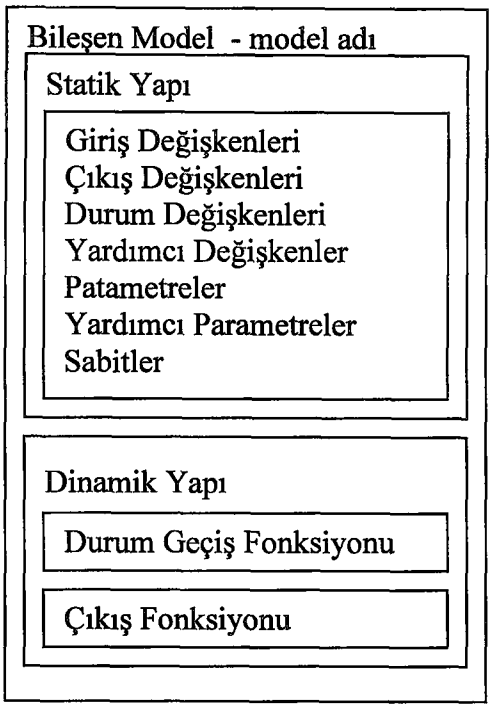
Sistemin belli bir t anındaki çıkışı, çıkış değişkenlerinin aldığı değere göre belirlenir. Çıkış değişkenlerinin belli bir t anındaki değeri, durum ve/veya giriş değişkenlerinin o andaki değerine bağlı olup, çıkış fonksiyonu ile hesaplanır.

Bir benzetim ortamında, bir bileşen modelin tanımlması, Ören tarafından geliştirilen Gest benzetim diline göre, iki aşamada yapılır. Bunlar,

- Statik yapının tanımlaması
- Dinamik yapının tanımlaması

Buradaki her bir tanımlama "ayrı tanımlama bloklarında" yapılır.

Statik yapının tanımlandığı blokta, sistemin tanımlayıcı değişkenlerin (giriş, çıkış ve durum değişkenleri), yardımcı değişkenlerin, sistem parametrelerinin, yardımcı parametrelerin ve sabitlerin tanımlamaları yer alır. Dinamik yapının tanımlandığı blokta ise, durum geçiş ve çıkış fonksiyonlarının tanımlama blokları yer alır.
(Şekil 3.3)



Şekil 3.3 Bileşen modelin tanımlama blokları

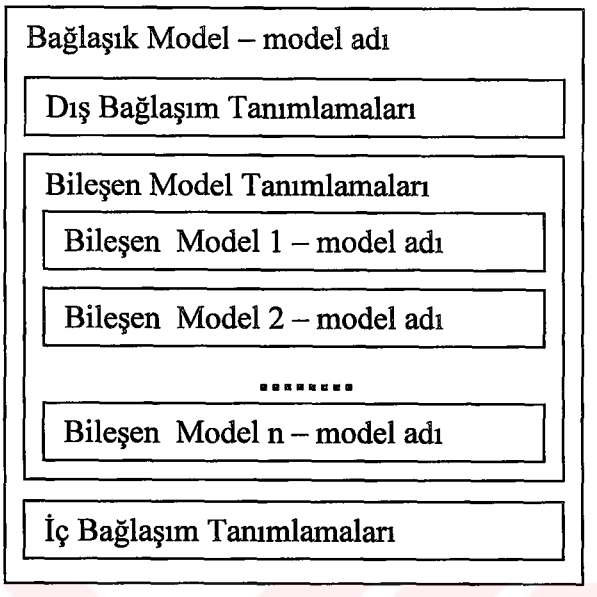
Şekil 3.3'de görülen model tanımlama bloğunda, "model adı" ifadesi modelin adını, sabitler ise, matematik modelde kullanılan sabit isimlerini gösterir. Bir sabitin tanımlaması, sabitin adı ve değeri verilerek yapılır. Örneğin pi = 3.14 gibi. Yardımcı değişkenler, durum geçiş ve çıkış fonksiyonlarında hesaplama kolaylığı için, geçici değişken olarak kullanılırlar. Yardımcı parametreler de yardımcı değişkenler gibi, hesaplama kolaylığı için kullanılır ve değerleri, parametre ve sabitlere göre hesaplanır.

Bir bağlaşık model, işlevsel özellikleri birbirinden bağımsız, alt sistemleri içeren sistem ya da alt sistemleri temsil eder. Bağlaşık model ile temsil edilen bir sistemin, yerine getirdiği işlev, içerdiği alt sistemlerin işlevleri toplamına denk kabul edilir.

Bağlaşık bir sistemin alt sistemlerinin bir araya gelerek, tüm sistemin toplam işlevini oluşturabilmeleri için aralarında bir "veri akışının" olması gerekir. Bu veri akışı, alt sistemler arasında, her t anı için, giriş ve çıkış değerlerinin aktarımıyla geçekleşir. Veri akışının yönü, "bağlaşım tanımlamaları" ile belirlenir.

Bir bağlaşık sistemin belli bir t anındaki durumu, içerdiği alt sistemleri temsil eden bileşen modellerin o andaki durum değişkenlerinin değerine göre belirlenir.

Bir bağlaşık modelin oluşturulması, bileşen modeller arasındaki veri akışı ve bağlaşım türleri ilerdeki bölümlerde açıklanmıştır.



Şekil 3.4 Bağlaşık modelin tanımlama blokları

Şekil 3.4'de gösterilen bağlaşık model tanımlama bloklarında, "model adı" ifadeleri ilgili bağlaşık ve bileşen modelin adını belirtir.

"Dış Bağlaşım Tanımlamaları" bloğu, bir bağlaşık sistemin dış ortamdan aldığı giriş değerlerini, iç bileşen sistemlerin giriş değerlerine aktarılış şeklini ve iç bileşen sistemlerin çıkış değerlerini, dış ortama aktarılış şeklini tanımlar.

"Bileşen Model Tanımlamaları" bloğu, her bir bileşen model için yapılan tanımlamaları içerir. Bu tanımlamaların blok yapısı, şekil 3.3'de gösterilmiştir.

"İç Bağlaşım Tanımlamaları" bloğu, bileşen modeller arasındaki giriş-çıkış ilişkilerini tanımlar.

Yukarıda belirttiği gibi, bir sistemin, belirli bir seviyedeki işlevsel özellikleri, sistem hiyerarşisinin alt seviyelerinde yer alan alt sistemlerin işlevsel özelliklerine göre belirlenir. Buna göre, bir sistemin alt sistemleri, dolayısıyla, bu alt sistemleri temsil eden modeller arasında yapılan "bağlaşım tanımlamaları", bir sistem hiyerarşisi "formülasyonu" olarak düşünülebilir.

Bir sistemin hiyerarşik yapısının analizi, sitemin işleyişi hakkında genel veya detaylı bir bilgi vermekle kalmaz, aynı zamanda, o sistem üzerinde ortaya atılabilecek, farklı türdeki sistem problemlerinin çözümü için, genel bir iskelet yapı oluşturur

Bu yapıda, göz önüne alınan en önemli nokta, problemin çözümünü, birde fazla "objektiviteyi" içermesidir. Bir sistem probleminin birden fazla objektiviteyi içermesi, problemin çözümünün, birden fazla alt çözüm içeren bir "çözüm ailesi" teşkil etmesi ile ifade edilebilir. Çözüm ailesinin üyeleri, sisteme ait genel bir problemde, her biri, belli bir objektiviteye göre belirlenen alt problemlerin çözümleridir. Bu çözümler arasındaki toplam "sebep-sonuç" ilişkisi, tüm problemin çözümünü meydana getirir.

Daha öncede ifade edildiği gibi, bir sistem problemine ait çözüm, sisteme ait bir model ile ifade edilebilir. Oluşturulan model, sistemin tüm unsurlarını yansıtamaz yani bir model, sistemi ancak ve ancak problem objektiviteklerine göre, "kısmi" olarak temsil edebilir.

Objektiviteye dayalı problem çözümünde, Zeigler'in önerdiği "çok yüzlü modelime" kavramı önemli bir yer tutar. Bu kavram sistem teorisi ve yapay zekanın birlikteliğinde, modüler bir problem çözme platformu oluşturur.

Sistem teorisi, sistemin, belli bir görüş açısı ve problem objektivitelerne göre, parçalara (en küçük atomik düzeydeki alt sistemlere) ayrışmasını, ve daha sonra bağlaşım tanımlamalarına göre bu parçaların bir araya gelerek istenen nitelikteki sistemi oluşturduklarını varsayar. Sistem teorisi, sistemin parçaları arasındaki ilişkinin sadece giriş-çıkış ilişkilerini ifade eder. Sistem teorisinin bu yaklaşımında, sistem parçacıkları arasında her hangi bir bilgi hiyerarşisi bulunmaz. Yani, belli bir sınıfa ait sistemden, nesneye dayalı programlama ve yapay zekadaki çatı (frame) yapılarında olduğu gibi, miras alma yöntemi ile, o sistem sınıfının özel bir halini teşkil eden yeni sınıflar oluşturulamaz.

Sistem parçalarının oluşturdukları bu hiyerarşik yapı, "sistem parçacık yapısı" olarak adlandırır. Bu yapı, çok yüzlü modellme kavramının temelini teşkil eder.

Bir sisteme ait "sistem parçacık yapısı", o sistemin statik yapısını oluşturur. Bu statik yapıda, bir sistemin alt sistemleri arasındaki ilişki giriş-çıkış ilişkisi düzeyinde ele alınır. Alt sistemler arasında oluşturulan ve verilen objektivitelere dayanan bu ilişkilendirme, yukarıda ifade edilen ağaç benzeri bir sistem hiyerarşisi meydana

getirir. (Şekil 3.2) Bir sistem problemine ait objectiviteler değiştikçe, hiyeraşininin yapısı da buna bağlı olarak değişir. Böylece, bir sisteme ait alt sistemlerin, farklı kombinasyonlar oluşturarak, farklı sistemleri meydana getirebilirler.

Sistemin statik yapısını temsil eden ve sistem probleminin objectivitelerine bağlı olarak oluşturulan sistem parçacık yapısı ya da sistem hiyerarşisi sadece statik yapıyı belirttiği için, sistemin "dinamik davranışı" hakkında net bir bilgi vermez.

Alt sistemlerden oluşan bir sistemin belli bir andaki durumu, dolayısıyla dinamik davranışı, sistem hiyerarşisinde yer alan tüm atomik seviyedeki alt sistemlerin durumuna göre belirlenir.

Atomik seviyedeki bir alt sistemin dinamik davranışı, yukarıda belirtilen bileşen model tanımlama yöntemiyle tanımlanır. Bu tanımla, belli bir bilgi formatında, "model tabanına" aktarılır.

Bu anlatılanlar ışığında, bir sistemin tüm davranışı, alt sistemlerin aralarındaki giriş-çıkş ilişkisini belirten statik yapıya ve sisteme ait her bir atomik sistemin dinamik yapısına bağlıdır. Sistem hiyerarşisine ait bilgiler ve atomik sistemlerin dinamik davranış bilgileri, modelleme ortamında kullanılan "billgi tabanını" oluştururlar.

Bu bilgi tabanındaki sistem hiyerarşisine ait bilgiler, objektivitelere bağlı olarak, alt sistemlerin, meydana getirebileceği, sistem kombinasyonlarına ilişkin bilgileri (bağlaşım tanımlamalarını) içerir. Model tabanındaki dinamik sistem tanımlamaları ise, sistem değişkenleri (giriş, çıkış, durum, yardımcı), parametreler, durum geçiş ve çıkış fonksiyonlarına ilişkin tanımlamaları içerir.

Bir sistem problemi verildiğinde, önce sistem hiyerarşisi oluşturulur. Daha sonra oluşturulan sistem hiyerarşisinde yer alan atomik sistemlerin dinamik davranış tanımlamaları, model tabanından elde edilir. Eğer bilgi tabanında, sitem hiyerarşisine ilişkin bilgi yoksa, hiyerarşi yeniden oluşturulur Yani verilen atomik sistemleri temsil eden bileşen modeller ile, bir bağlaşık model oluşturulur.

Benzer şekilde, model tabanı alt sistemlere ilişkin, dinamik davranış tanımlamalarını içermiyorsa, bu tanımlamaları içeren bileşen model oluşturulup, model tabanına eklenir. Model tabanından alınan model, tamamen olmasa da, kısmen verilen objectiviteleri içerebilir Bu durumda, model üzerinde gerekli değişikler yapılarak, model, objectivitelere uygun hale getirilir. Bu durum, modellemenin, belirli bir

süreyle sınırlı kalmayıp, sistem isterleri (objectiviteleri) değiştikçe, yinelenen bir süreç olduğunu gösterir.

Model tabanının kullanımının bir diğer yararı, modelin geçerlilik ve doğruluk testlerinin, kısmen yapılabilmesidir. Şöyle ki, bir sistemin herhangi bir seviyesindeki herhangi bir alt sistemin modeli oluşturulduğunda ve geçerlilik testleri yapıldığında, model başka kombinasyonlarda kullanıldığı zaman, aynı testleri tekrarlamaya gerek yoktur.

3.2 Model Türleri

Sistem hiyerarşisinin en alt seviyelerindeki atomik sistemlerin, belli bir anda aldıkları durumun sayısal değer (durum değişkenlerinin aldığı değer), o anda almış olduklar giriş değişkenlerinin değerine, zamana ve durum geçiş fonksiyonunun yapısına bağlıdır.

Atomik sistemleri temsil eden bileşen modeller arasındaki farklılık genel olarak, Tanımlayıcı değişkenlerin (giriş, çıkış ve durum değişkenlerinin) kimi modellere göre, eksik olup olmamasından veya durum geçiş fonksiyonlarında kullanılan "formülasyonun" farklı olmasından kaynaklanır.

Bazı sistemler ve bu sistemleri temsil eden modeller, giriş ve çıkış değişkenleri içermezler. Bu tür sistemlere, "kapalı sistem" denir. Bu tezde ele alınan sistemler , "açık sistem" olup, dış ortamlarıyla etkileşim halindedir.

Bazı modeller ise, bölüm 3.3.3'de açıklanacağı gibi durum değişkenleri içermezler. Bu tür modellere "belleksiz model" adı verilir. Durum değişkeni içeren modeller de, "bellekli model" olarak adlandırılır. Bölün 3.3.1 ve 3.3.2'de anlatılan sürekli ve ayrık modeller, bellekli modellerdir.

Daha önce de belirtildiği gibi, sistemin belli bir t anındaki durumu, bir davranış üreteci ile sağlanan algoritmanın, o anda, durum değişkeni için ürettiği (hesapladığı) değere göre belirlenir.

Bir modele ait tanımlayıcı değişkenin, $[t_0, t_n]$ aralığında almış olduğu "değerler kümesine", o değişkenin "yörüngesi" adı verilir. Şöyle ki, $t_0, t_1, \ldots\ldots, t_n$ değerleri için, bir X(t) değişkeninin almış olduğu $X(t_i)$ değerler kümesi $\{X_1, X_2\ldots\ldots\ldots, X_n\}$, X değişkeninin bir yörüngesidir.

Bir davranış üretecinin, durum değişkenleri için değer üretirken kullanacağı algoritmanın yapısı, durum geçiş fonksiyonlarının formülasyonuna bağlıdır. Durum geçiş fonksiyonlarının ve çıkış fonksiyonlarının formülasyonları sistemin genel formülasyonunu oluşturur. Buradaki formülasyon kavramı model değişkenleri arasındaki ilişkinin matematiksel ifadesidir. Bu matematiksel yapıda kullanılan matematiksel kavramların (diferansiyel denklemler, fark denklemleri...) türü, modelin türüne göre seçilir. Aşağıda temsil ettikleri sistemlerin formülasyonlarında kullanılan matematiksel kavramlara ve benzetim zaman aralığının sürekli olup olmamasına göre modeller sınıflandırılmıştır

Burada, zaman aralığının sürekli olması, değişkenlerin, verilen zaman aralığının "her noktasında" değer aldıklarını ifade eder. Bu durumda zaman parametresinin değeri reel sayıdır. Zaman aralığı sürekli değilse, bu zaman aralığı, "ayrık zaman aralığı" olarak adlandırılır. Ayrık zaman aralığında, değişkenler, zaman parametresinin sadece tamsayı değerlerinde değer alırlar.

3.2.1 Sürekli Model

Sürekli modeller, verilen bir $[t_0, t_n]$ aralığında (sürekli ya da ayrık), zamanın belirtilen noktalarında, hareketinde ani bir sıçrama olmayan yani sürekli bir hareket sergileyen sistemleri temsil ederler.

Bu tür modellerde, sistem formülasyonu için "diferansiyel denklemler" kullanılır. Kullanılan diferansiyel denklemlerin bağımlı değişkenleri durum değişkenleri, bağımsız değişken ise t zaman parametresidir. Çıkış fonksiyonları durum değişkenlerine ya da hem durum, hem de çıkış değişkenlerine bağlıdır. Bu tür modellerde kullanılan formülasyonunda

X giriş değişkenleri kümesi
Y durum değişkenleri kümesi
Z çıkış değişkenleri kümesi (3.1)
F durum geçiş fonksiyonu
G çıkış fonksiyonu

olmak üzere, belli bir $t_i \in [t_0, t_n]$ anı için, $x_1, x_2,.....,x_m \in X$, $y \in Y$ ve $z \in Z$ iken, sistemin çıkış değeri,

$z=G(y)$ veya $z=G(x_1, x_2,.....,x_m, y)$ (3.2)

ve sistemin durum değişkeni ile ifade edilen diferansiyel denklem,

$y'=F(t, y, x_1, x_2,.....,x_m)$ (3.3)

şeklinde ifade edilir.

(3.3)'de ifade edilen diferansiyel denklem, bir "parametrik diferansiyel denklemdir." Diferansiyel denklemin parametreleri, sistem parametrelerinin ve giriş değişkenleridir.

Bir sürekli modelin içerdiği matematik modelin işlenmesinde, davranış üretecinin kullandığı algoritma bir sayısal entegrasyon algoritmasıdır. Bu algoritmada, önce, durum değişkeninin başlangıç değeri ve entegrasyon adım uzunluğu verilir ve daha sonra, her adım için, seçilen entegrasyon yöntemine göre, durum değişkeninin değeri, bir önceki değere göre hesaplanır.

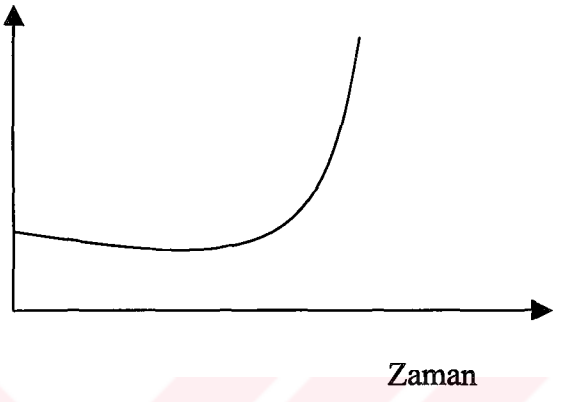
Diferansiyel denklem, sistem parametrelerine ve giriş değişkenlerine göre parametrik yapıda olduğundan, parametre değerleri sabit kalmak koşuluyla, giriş değişkenlerinin o an için değeri diferansiyel denkleme, dolayısıyla, entegrasyon işlemine "sabit değer" olarak girer. Dolayısıyla, durum değişkenin t_i anındaki değeri, t_{i-1} anındaki "durum ve giriş değişkenlerinin" değerine bağlıdır.

Diferansiyel denklemin analitik çözümü varsa, durum değişkenleri için hesaplanan değerler, parametrik bir fonksiyon olan çözüm fonksiyonunun, o andaki t zaman, giriş değişkenleri ve parametre değerlerine göre hesaplanan değerine eşittir.

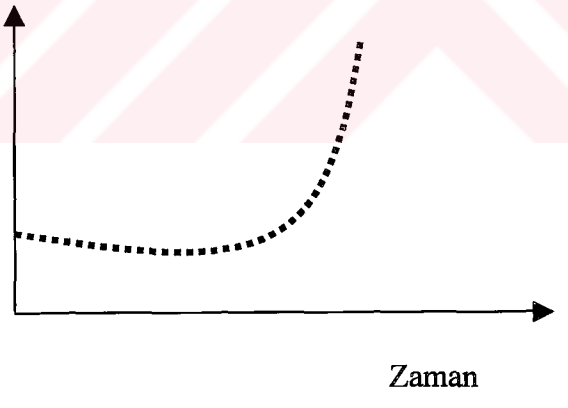
Sürekli bir modelde, durum ve çıkış değişkenlerinin yörüngeleri, verilen zaman aralığında sürekli olup, herhangi bir sıçrama noktası içermezler. Giriş değişkenlerinin yörüngeleri ise, parçalı sürekli olup, beli noktalarda sıçrama özelliği gösterir. Sürekli ve ayrık zaman aralıkları için, bir sürekli modele ait durum, çıkış ve giriş değişkenlerinin yörüngeleri, Şekil 3.5, 3.6, 3.7 ve 3.8'de gösterilmiştir.

Giriş değişkenleri, belli zaman değerleri için sıçrama özelliği gösterdiğinde, sayısal entegrasyon algoritmasının yerini o an için,"süreksizliği giderme algoritması" alır. Süreksizlik giderildikten sonra, entegrasyon algoritması kaldığı yerden devam eder. Aynı durum, durum değişkenleri için de geçerlidir. Durum değişkenlerinin süreksizliği, değişken değerlerinde ya da diferansiyel denklemin türev fonksiyonu

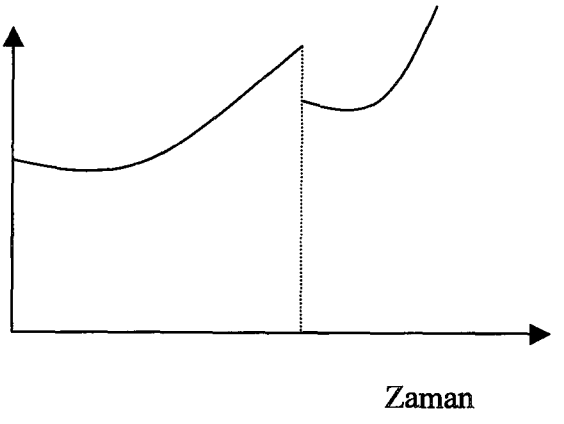
değerlerinde ortaya çıkar. Süreksizliğin değişken değerlerinde oluşması durumunda, başlangıç değer koşulları yeniden verilir. Türev fonksiyonu değerlerinde ortaya çıkması durumunda ise, model yenilenir ve benzetim işlemi yeniden başlar. Süreksizlik algoritması, bu tez çalışmasının dışında olup, ele alınan tüm değişken yörüngelerinin sürekli olduğu kabul edilmiştir.



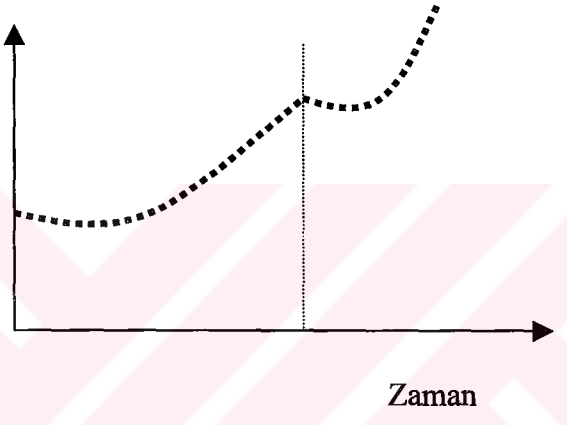
Şekil 3.5 Sürekli zaman arlığı için sürekli modelde durum ve çıkış değişkenlerinin yörüngesi



Şekil 3.6 Ayrık zaman arlığı için sürekli modelde durum ve çıkış değişkenlerinin yörüngesi



Şekil 3.7 Sürekli zaman arlığı için sürekli modelde giriş değişkenlerinin yörüngesi



Şekil 3.8.Ayrık zaman arlığı için sürekli modelde giriş değişkenlerinin yörüngesi

3.2.2 Ayrık Model

Ayrık modeller, model değişkenlerinin, verilen bir $[t_0, t_n]$ aralığında, t zaman parametresinin sadece tamsayı değerleri için değeri değişen ve diğer değerlerde yani iki tamsayı arasında değeri değişmeyen model türleridir.

Bu tür modellerin, durum geçiş fonksiyonu, "adım fonksiyondur." Durum değişkenlerinin ilk değerleri verildikten sonra algoritma başlar. Yukarıda belirtilen entegrasyon algoritmasına benzer şekilde, durum değişkenlerinin belli bir t_i anındaki değeri, durum geçiş fonksiyonunun, durum ve giriş değişkenlerinin t_{i-1} anındaki değerleri için aldığı değere eşittir. Çıkış değişkenin değeri aynı t anı çıkış fonksiyonun aldığı değere eşittir.

X giriş değişkenleri kümesi
Y durum değişkenleri kümesi
Z çıkış değişkenleri kümesi (3.4)
F durum geçiş fonksiyonu
G çıkış fonksiyonu

olmak üzere, belli bir $t_i \in [t_0, t_n]$ için $x \in X$, $y \in Y$ ve $z \in Z$ ve, $x_i=x(t_i)$, $x_{i-1}=x(t_{i-1})$; $y_i=y(t_i)$, $y_{i-1}=y(t_{i-1})$; $z_i=z(t_i)$, $z_{i-1}=z(t_{i-1})$; iken sistemin çıkış değeri,

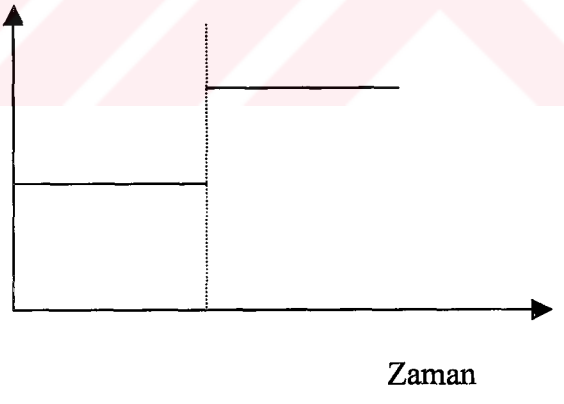
$$z_i=G(y_i) \text{ veya } z_i=G(x_i, y_i) \quad (3.5)$$

ve bir adım fonksiyonu olan durum geçiş fonksiyonu,

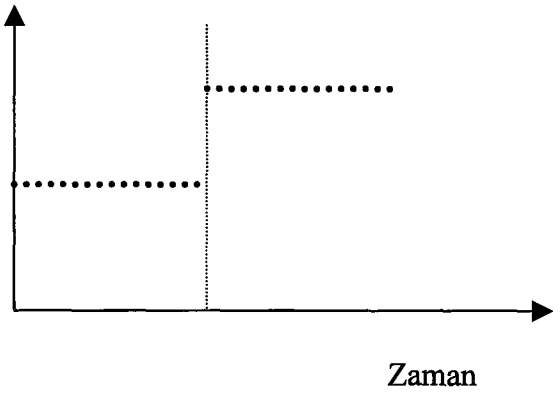
$$y_i=F(y_{i.-1}, x_{i-1}) \quad (3.6)$$

şeklinde ifade edilir.

Bir ayrık modelin, giriş, çıkış ve durum değişkenlerinin yörüngeleri, sürekli ve ayrık zaman aralıklarında adım fonksiyonu özelliği gösterir. (Şekil 3.9 ve 3.10) Bu tez çalışmasında, ele alınan model türleri, ayrık modelleri içermez.



Şekil 3.9.Sürekli zaman arlığı ayrık modelde için giriş değişkenlerinin yörüngesi



Şekil 3.10 Ayrık zaman arlığı ayrık modelde için giriş değişkenlerinin yörüngesi

3.3.3 Belleksiz Model

Durum değişkeni, dolayısıyla, durum geçiş fonksiyonu içermeyen modellere belleksiz model adı verilir. Bu tür modellerde, belli bir t anındaki çıkış değeri, çıkış fonksiyonunun o andaki giriş değerine göre hesapladığı değere eşittir.

X giriş değişkenleri kümesi
Z çıkış değişkenleri kümesi (3.7)
G çıkış fonksiyonu

olmak üzere, belli bir $t_i \in [t_0, t_n]$ için $x \in X$ ve $z \in Z$ ve, $x_i=x(t_i)$; $z_i=z(t_i)$; iken sistemin çıkış değeri,

$z_i=G(y_i)$ veya $z_i=G(x_i, y_i)$ (3.8)

şeklinde hesaplanır.

3.4 Modeller Arası İlişki

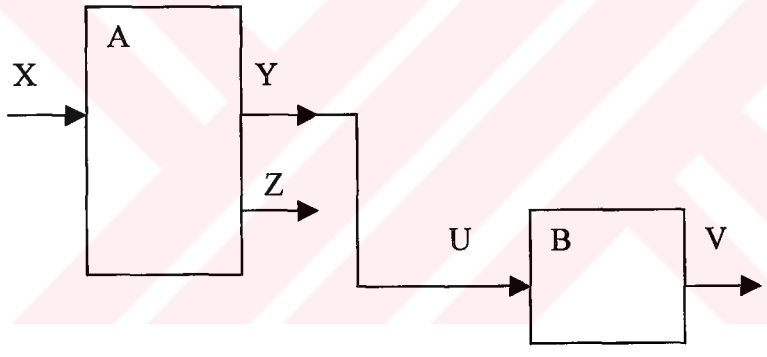
Bundan önceki bölümlerde de bahsedildiği gibi, belli bir sistemin işlevsel özelliği, objektivitelere bağlı olarak oluşturulan sistem hiyerarşisinde yer alan alt sistemlerin işlevsel özelliklerinin toplamı gibi düşünülebilir.

Hiyerarşiyi meydana getiren sistemlerin en alt düzeydeki atomik sistemlerden başlayarak, hiyerarşinin en tepe noktasına kadar, tüm sistemin dinamiğini meydana getirebilmeleri aralarındaki giriş-çıkış ilişkileri, dolayısıyla bağlaşım oluşturabilmeleri, ile mümkündür.

Aşağıdaki bölümlerde, farklı türdeki alt sistemlerden oluşan, hiyerarşik yapıdaki sistemlerin, benzetim ortamında, bağlaşık modellerle nasıl temsil edilebileceği, ağlaşım türleri ve bağlaşık modeller için oluşturulan benzetim algoritması üzerinde durulmuştur.

3.4.1 Bağlaşım Kavramı

Hiyerarşik yapıdaki sistemlerin modellenmesi, sistem hiyerarşisini meydana getiren sistemlerin giriş-çıkış ilişkilerini ifade eden "bağlaşım tanımlamaları" ile yapılır.
İki sistem, dolayısıyla, bunları temsil eden modeller arasındaki bağlaşım tanımlaması, "görsel olarak", modellerden birinin bir ya da birden çok çıkışının, diğer modelin bir ya da birden çok girişine, bir çizgi ya da bir ok işareti ile yapılır. Bağlaşım tanımlamasını yönü, "çıkış gönderen" modelden (sistemden), "giriş alan" modele (sisteme) doğrudur. (Şekil 3.11)



Şekil 3.11 İki model arasında bağlaşım

Şekilde, A ve B modelleri arasında oluşturulan bağlaşımda, bağlaşımın yönü, A modelinden, B modeline doğrudur. Bu bağlaşım,

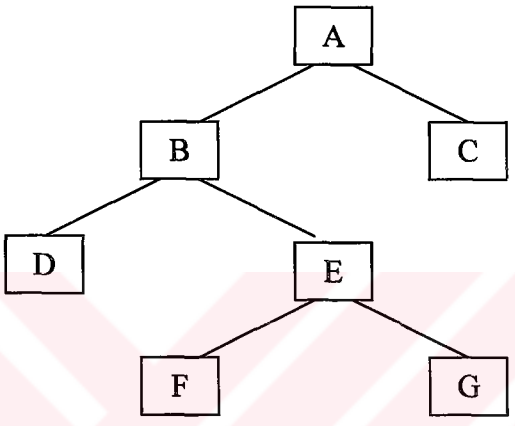
$$A.Y \rightarrow B.U \tag{3.9}$$

şeklinde ifade edilir.

Buna göre, model (sistem) hiyerarşisinin herhangi bir seviyesinde bu tür bir bağlaşım içeren bağlaşık model, davranış üretecinde yer alan bir benzetim algoritmasıyla işlendiğinde, A modelinde, belli bir t anı için, Y çıkış değişkeninin değeri, bu

değişkene ait çıkış fonksiyonu ile hesaplanır. Daha sora, hesaplanan değer, "aynı t anı için", B modelinin, U giriş değişkenine atanır.

Bağlaşık modellerde, tüm sistemin dinamiği, bileşen modellerin içerdiği durum değişkenlerinin (model bellekli ise) sayısal değerlerine göre belirlenir. Buna göre bir bağlaşık modelin belli bir t anındaki durumu, "sayısal olarak ifade edildiğinde", model hiyerarşisinde, hiyerarşinin en üst seviyesinde modelin kendisi olmak üzere, tüm atomik modellere ait durum değişkenlerinin, o t anı için aldıkları değerler, modelin (sistemin) durumunu yansıtırlar.



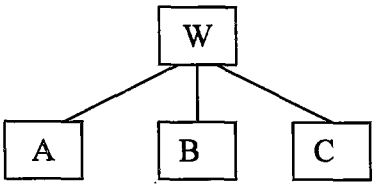
Şekil 3.12 Model hiyerarşisi

Şekil 3.12'de gösterilen model hiyerarşisi, ağaç benzeri bir yapıda olup, ağacın en uç yaprakları olan C, D, F ve G modelleri atomik modeldir. Buna göre, belli bir t anında, A modelinin temsil ettiği sistemin durumu, B ve C modellerinin temsil ettikleri sistemlerin durumuna, B modelinin temsil ettiği sistemin durumu, D ve E modellerinin temsil ettikleri sistemlerin durumuna ve benzer şekilde, E modelinin temsil ettiği sistemin durumu, F ve G modellerinin temsil ettikleri sistemlerin durumuna ağlıdır.

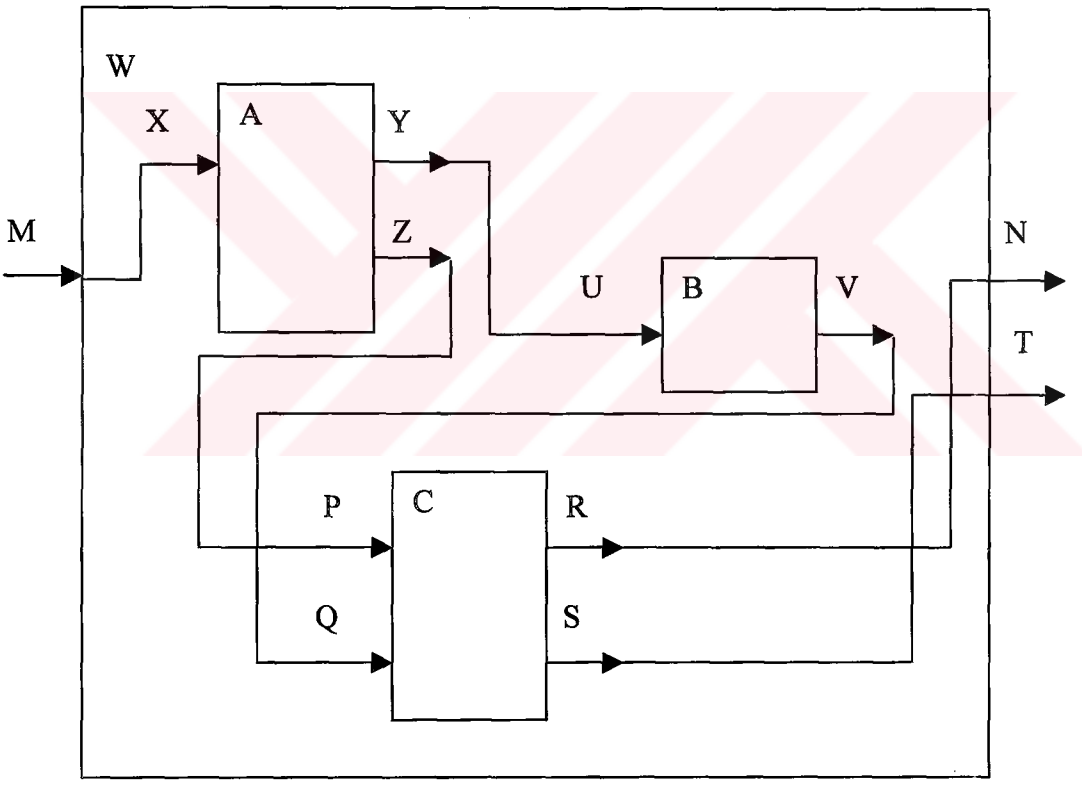
Model hiyerarşisinde en uç yapraklardaki modeller, bileşen model olup, eğer model bellekli model ise, durum geçiş fonksiyonlarını ve çıkış fonksiyonlarını, belleksiz model ise sadece çıkış fonksiyonlarını içerir. Buna göre, bir model hiyerarşisindeki bileşen modeller, sürekli, ayrık veya belleksiz model olabilir.

Model hiyerarşisinin düğüm noktalarında bulunan modeller ise, bağlaşık model olup, durum geçiş ve çıkış fonksiyonu içermezler. Bu modeller, kendilerine bağlı olan

düğüm ve yapraklardaki, bileşen ya da bağlaşık modellerin, giriş-çıkış ilişkilerini ifade eden bağlaşım tanımlamalarını içerir. Aşağıdaki şekillerde, üç bileşen ve bir bağlaşık model içeren model hiyerarşisi ve bunun tanımlama şeması gösterilmiştir.



Şekil 3.13 Üç bileşen ve bir bağlaşık model içeren model hiyerarşisi



Şekil 3.14 Üç bileşen ve bir bağlaşık model içeren model hiyerarşisinin belirttiği bağlaşık modelin tanımlama şeması

Şekilde tanımlama şeması gösterilen W modelinin alt bileşen modelleri sırasıyla A, B ve C modelleri olup, "en az biri bellekli model" olmak koşuluyla, W modelinin

durumunu belirtirler. W modeli ise sadece bağlaşım tanımlamalarını içerir. Buradaki M, W modelinin girişi, N ve T ise çıkışlarıdır.

3.4.2 Giriş-Çıkış Modeli

Bağlaşık model oluşturulurken, bileşen modellerin, modelin "bellekli model" olması durumunda, başlangıçta durum değişkenleri, durum geçiş fonksiyonu, çıkış fonksiyonu verileyip, sadece giriş ve çıkışları verilebilir. Bu tür modellere "giriş-çıkış modeli" adı verilir.

Bağlaşık model tanımlamasında, tüm modeller arasında, önce bağlaşım tanımlamaları ve daha sonra modelin iç yapısını oluşturan durum değişkenleri, durum geçiş fonksiyonları ve çıkış fonksiyonlarının tanımlamaları yapılabilir.

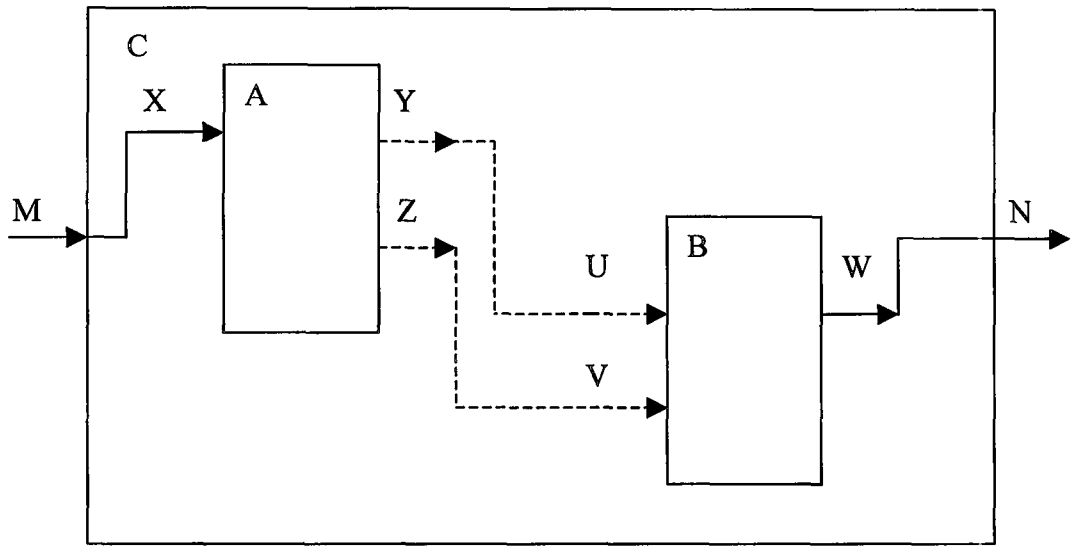
3.4.3 İç Bağlaşım

Bir bağlaşık modelin tanımlanmasında, bağlaşık modele ait bileşen modellerin kendi aralarında oluşturdukları bağlaşım türüne iç bağlaşım denir. Bu bağlaşımlarda, bileşen modellerin çıkış değerleri.,sadece, diğer bileşen modellerin giriş değerlerine ya da kendi giriş değerlerine (geri besleme bağlaşım yoluyla, bölüm 3.4.5) aktarılır.

Şekil 3.15'de tanımlama şeması gösterilen C bağlaşık sisteminde, A ve B bileşen modelleri arasında oluşturulan bağlaşımda, ifadeleri,

$$A.Y \rightarrow B.U$$
$$A.Z \rightarrow B.V \qquad (3.10)$$

olan ve kesikli çizgilerle gösterilen bağlaşım tanımlamaları iç bağlaşımdır.



Şekil 3.15 İki model arasında iç bağlaşım

3.4.4 Dış Bağlaşım

Bir bağlaşık modelin tanımlanmasında, bağlaşık modelin, bileşen modeller ile oluşturdukları bağlaşım türüne dış bağlaşım denir. Bu bağlaşımlarda, bağlaşık modelin dış ortamdan aldıkları giriş değerleri, bileşen modellerin giriş değerlerine ve bileşen

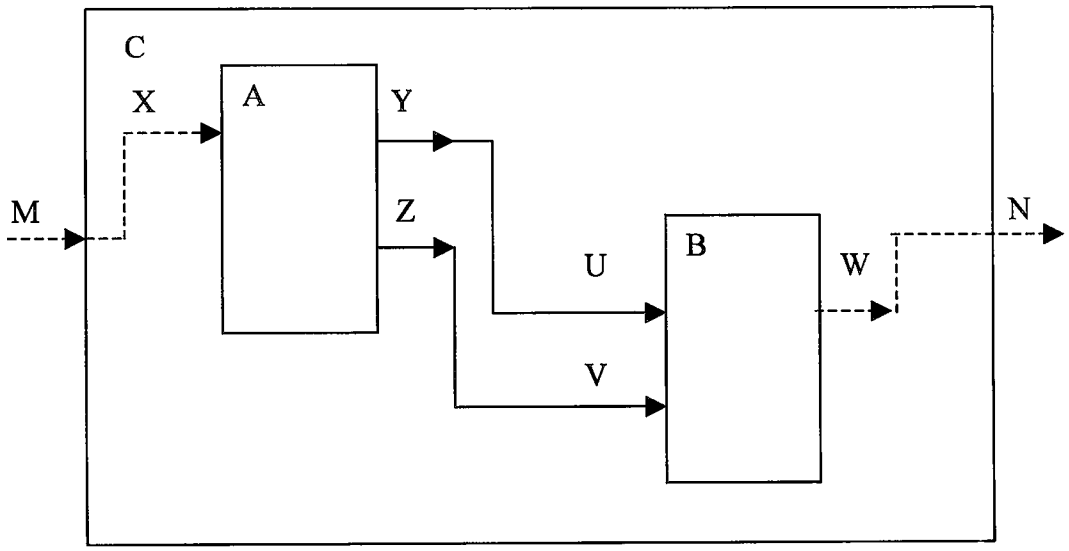
modellerin çıkış değerleri de, bağlaşık modelin çıkış değerlerine aktarılır.

Şekil 3.16'da tanımlama şeması gösterilen C bağlaşık sisteminde, A ve B bileşen modelleri arasında oluşturulan bağlaşımda, ifadeleri,

C.M → A.X

B.W → C.N (3.11)

olan ve kesikli çizgilerle gösterilen bağlaşım tanımlamaları dış bağlaşımdır.

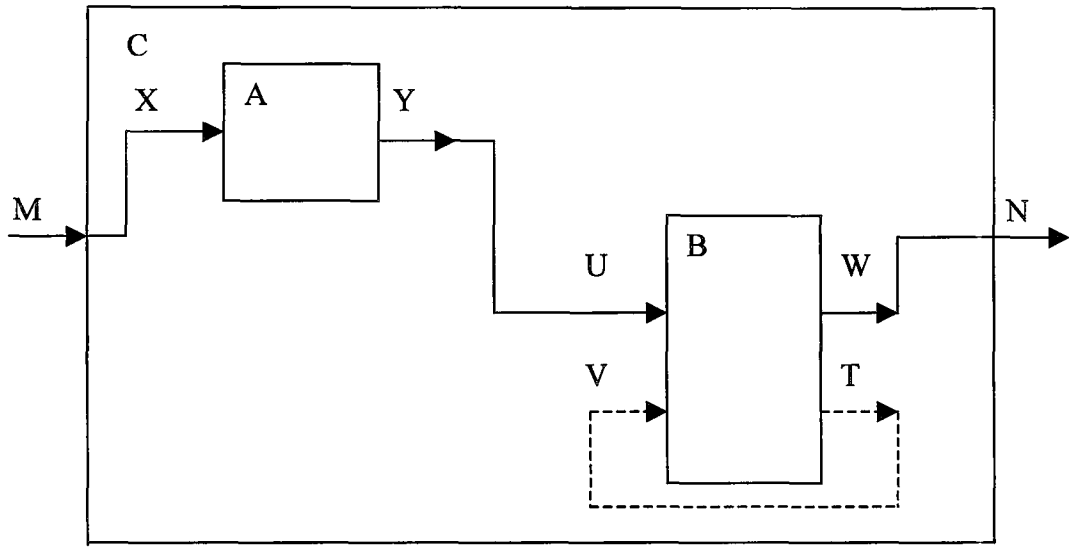


Şekil 3.16 Dış bağlaşım

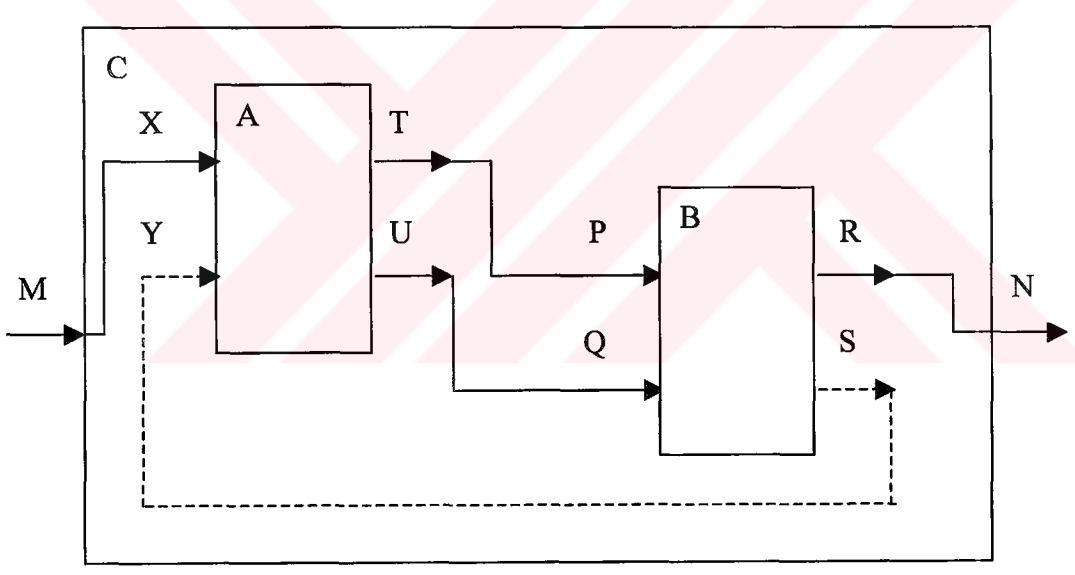
3.4.5 Geri Besleme Bağlaşım

Bir bağlaşık modelin tanımlanmasında, her herhangi bir bileşen modele ait çıkış değerleri, modelin kendi giriş değerlerine, ya da "doğrudan veya dolaylı" olarak, modele değer gönderen diğer modellerin giriş değerlerine aktarıldığı bağlaşım türüne "geri bağlaşım" denir.

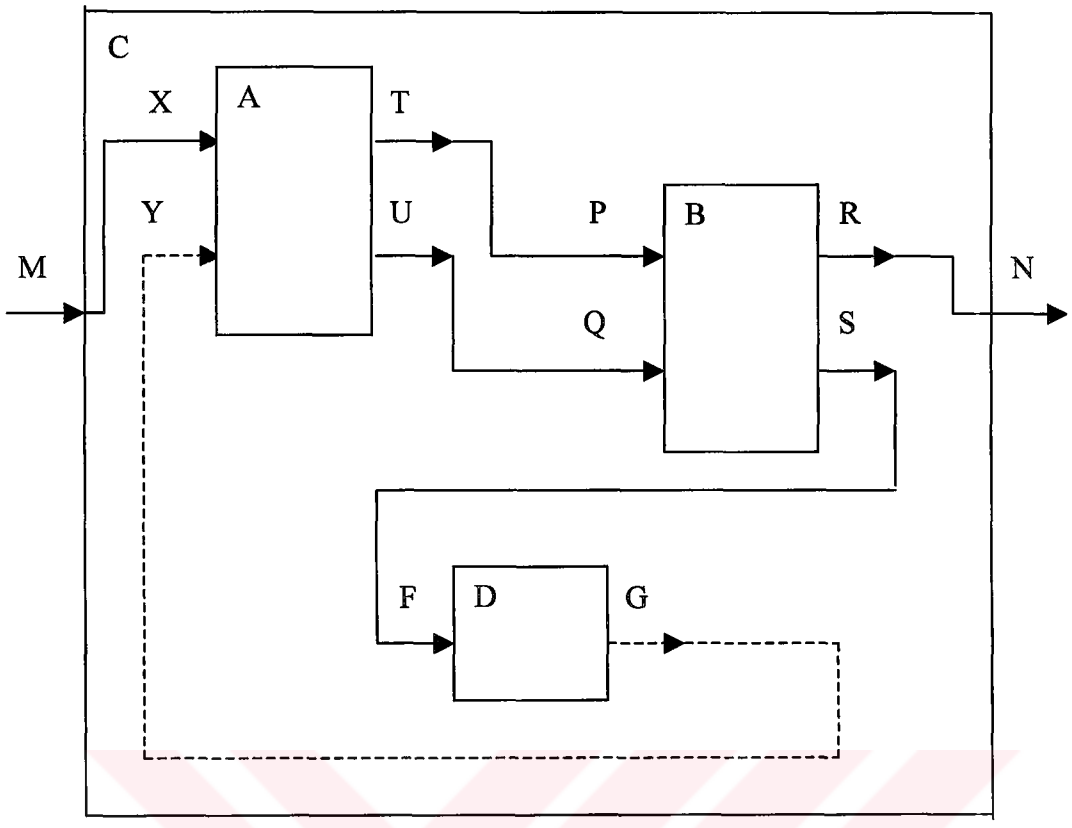
Şekil 3.17, 3.18 ve 3.19'da, sırasıyla, "bir modelin kendi üzerinde", "bir modelin doğrudan kendisine değer gönderen model ile" ve "bir modelin dolaylı olarak kendisine değer gönderen model ile" oluşturduğu geri besleme bağlaşım kesikli çizgilerle gösterilmiştir.



Şekil 3.17 Bir modelin kendi üzerinde oluşturduğu geri besleme bağlaşım



Şekil 3.18 Bir modelin, doğrudan kendisine değer gönderen model ile yaptığı geri besleme bağlaşım



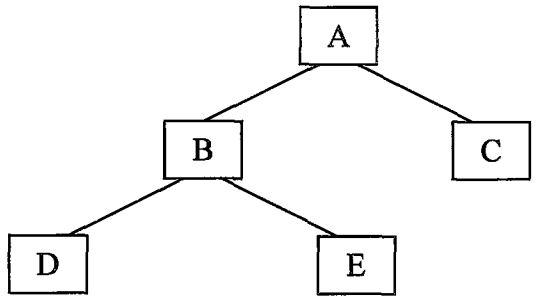
Şekil 3.19 Bir modelin, dolaylı olarak kendisine değer gönderen model ile yaptığı geri besleme bağlaşım

Belleksiz modeller şekil 17'de gösterildiği gibi kendisiyle geri besleme bağlaşım oluşturamaz. Şekil 18 ve 19'oluşturulan bağlaşık modelde, bileşen modellerden en az biri bellekli model olmalıdır.

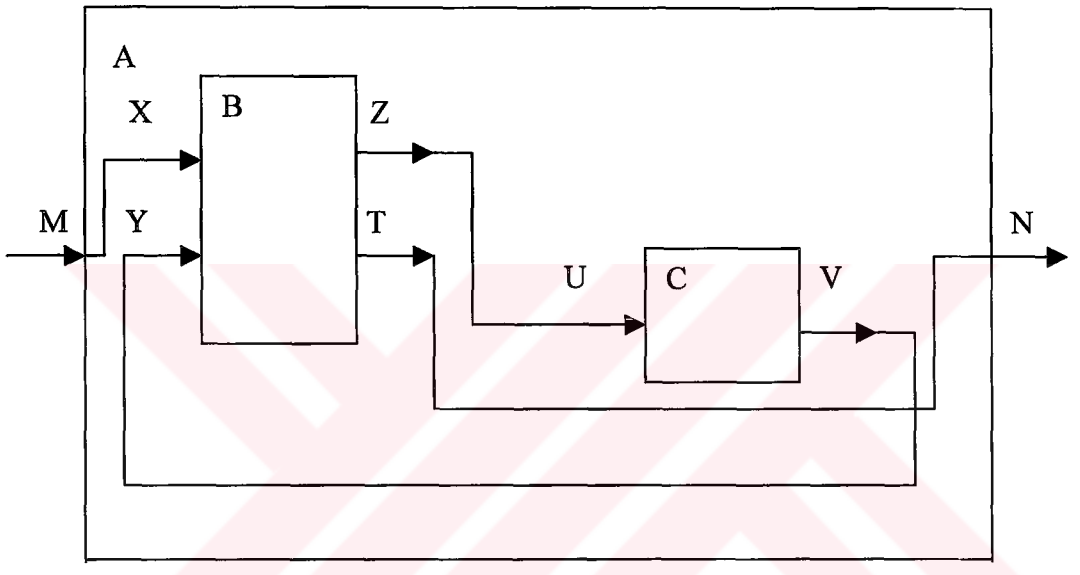
3.4.6 İç İçe Bağlaşım

İç içe ya da hiyerarşik bağlaşımda, bir bağlaşık model, bir başka bağlaşık modelin bileşen modeli olabilir. Bu tür modellere, "bağlaşık bileşen model" adı verilir.

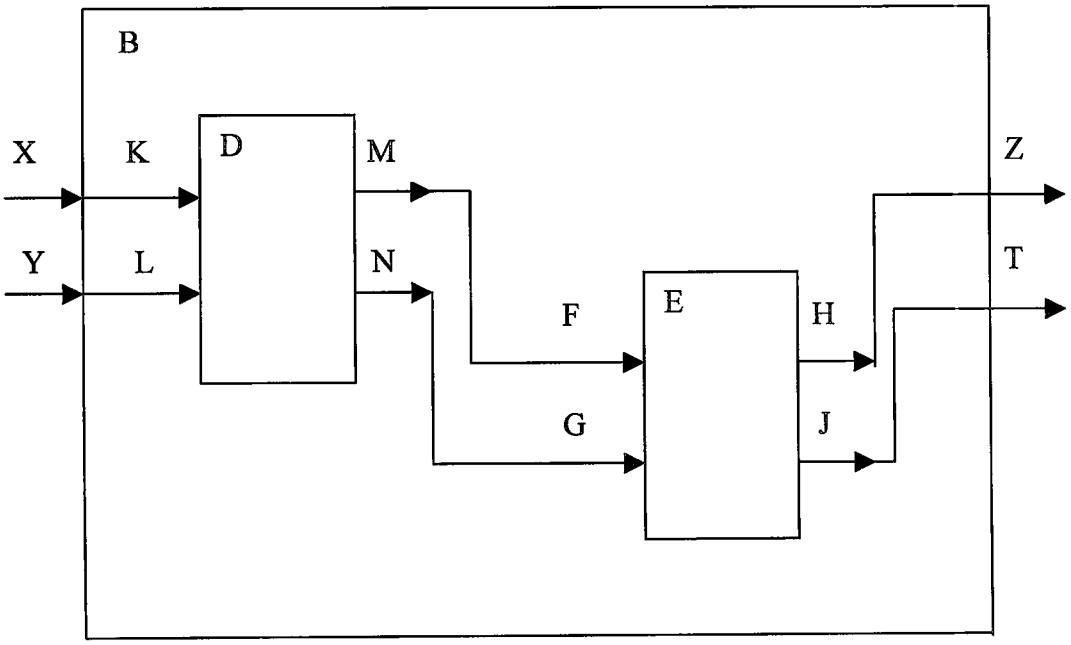
Şekil 3.20'deki model hiyerarşisinde, B modeli bir bağlaşık model olup, şekil 3.21'de tüm hiyerarşinin, şekil 3.22'de ise, B modelinin tanımlama şeması gösterilmiştir.



Şekil 3.20 İç içe bağlaşımda model hiyerarşisi



Şekil 3.21 İki model arasında bağlaşım



Şekil 3.22 B modeli

3.4.7 Bağlaşık Model Oluşturma Kuralları

Bu bölümde kısaca, bir bağlaşık modelin oluşturulmasında uyulması gereken kurallar sıralanmıştır. Buna göre,

- Bir dış bağlaşımda, bağlaşık modelin giriş değerlerinden herhangi biri, bileşen modellerin giriş değerlerinden, bir ya da birden fazlasına aktarılabilir.
- Bir iç bağlaşımda, bileşen modellere ait çıkış değerleri, kendisinin ya da diğer modellerin bir ya da birden çok giriş değerine aktarılabilir.
- Bir iç bağlaşımda, bileşen modellere ait giriş değerleri, kendisinden ya da diğer modellerden, en çok tane çıkış değeri alabilir.
- Bir dış bağlaşımda, bağlaşık modelin çıkış değerlerine, tüm bileşen modellerin çıkış değerlerinden en çok bir tanesi aktarılabilir.

3.4.8 Bağlaşık Modeller İçin Benzetim Algoritması

Bu bölümde, benzetim ortamında, bir bağlaşık modelin işlenmesinde kullanılan ve benzetim deneyini ifade eden, benzetim algoritması anlatılmıştır. Bu algoritma, verilen bir zaman aralığında, her t anı için, model hiyerarşisinde yer alan bileşen modeller arasındaki giriş-çıkış değer aktarım algoritmalarını ve bellekli bileşen modellere ait durum değişkenlerinin hesaplanmasında kullanılan sayısal entegrasyon algoritmasını kapsar.

Bu anlatım sadece metinsel ifadeleri değil, aynı zamanda, algoritmanın, bir dokümantasyon niteliği taşıyan şematik gösterimini de içerir.

Buna göre algoritma, üç ana safhadan oluşur. Bunlar sırasıyla,

1 - Deney öncesi

2 - Deney süresi

3 - Deney sonrası

Deney öncesinde, algoritmanın başlangıç koşulları belirlenir. Bu adımda yapılan tanımlamalar şunlardır :(Şekil 3.23)

- Bir modelin, ve bu modele ait parametre seti, deney koşulları tanımlamaları ve çıkış ortamı tanımlamalarından birer tane seçilir.
- Sayısal entegrasyon koşulları için ilk değerler belirlenir. Bunlar, zaman aralığının başlangıç değeri, adım sayısı ve adım uzunluğu değerleridir.
- Bellekli bileşen modellere ait durum değişkenlerinin ilk değerleri belirlenir.

Deney süresince, zaman aralığının her t anı için, model hiyerarşisindeki tüm modeller arasındaki giriş-çıkış değerlerinin hesaplanması ve aktarılması, ve bellekli bileşen modellerin durum değişkenlerinin hesaplanması işlemleri yapılır. Bu safha "sırasıyla" şu adımları içerir :(Şekil 3.24)

- Her bileşen model için çıkış değişkenlerinin değerlerinin hesaplanması. Çıkış değişkenlerinin değeri, çıkış fonksiyonu yardımıyla, durum değişkenlerine göre hesaplanır. Bileşen modellere ait durum değişkenlerinin ilk değerleri verildiğinden, t_0 anı için, tüm bileşen modellerin çıkış değerleri, "giriş-çıkış değer aktarımı" olmaksızın çıkış fonksiyonuyla hesaplanır.
- Hesaplanan çıkış değerlerinin, iç bağlaşım yoluyla, model hiyerarşisindeki her bileşen model için, "iç giriş değişkenlerine" atanması
- Bağlaşık modele ait "dış giriş değişkenlerinin" değerlerinin hesaplanması.

- Dış giriş değişkenleri ile dış bağlaşım yapan bileşen modellere ait iç giriş değerlerinin hesaplanması.
- Dış bağlaşım yapan bileşen modellerin, çıkış değerlerinin, bağlaşık modele ait "dış çıkış değerlerine" atanması.
- Bir sonraki t anı için durum değişkeninin değerinin hesaplanması.

Deney sonrasında, çıkış modülü tanımlamalarına göre, sonuçlar, çıkış ortamına aktarılır. (Şekil 3.25)

.

DENEY ÖNCESİ

(m , p_i, dk_j, $çt_k$) gurubunu seç

İlk değerler : Entegrasyon koşulları

$t = t_0$, h , n

İlk değerler : Durum değişkenleri

Her bileşen model için

Her durum değişkeni için

İlk değeri ata

Bir sonraki durum değişkeni

Bir sonraki bileşen model

DENEY SÜRESI

DENEY SONRASI

Şekil 3.23 Deney öncesi

DENEY ÖNCESI

DENEY SÜRESI

Her t için ($t_0 <= t <= t_n$)

Çıkış değişkenlerinin değerini hesapla

Her bileşen model için

Her iç çıkış değişkeni için

Değeri hesapla

Bir sonraki iç çıkış değişkeni

Bir sonraki bileşen model

İç bağlaşım yoluyla, iç giriş değişkenlerinin değerini ata

Her bileşen model için

Her iç giriş değişkeni için

İç bağlaşım yoluyla, değeri ata

Bir sonraki iç giriş değişkeni

Bir sonraki bileşen model

Dış giriş değişkenlerinin değerini hesapla

Her dış giriş değişkeni için

Değeri hesapla

Bir sonraki dış giriş değişkeni

Kalan iç giriş değişkenlerinin değerlerine dış girişlerin değerini ata

Her dış giriş değişkeni için

Değeri ata (Tablodan oku, Kuvvet fonksiyonu ile hesapla, Dış giriş değişkeni yöntemi)

Bir sonraki dış giriş değişkeni

Dış bağlaşımdaki dış çıkış değerlerini hesapla

Bir sonraki t değeri için durum değişkenlerinin değerini hesapla

Bir sonraki t, $t = t + h$

DENEY SONRASI

Şekil 3.24 Deney süresi

DENEY ÖNCESI
DENEY SÜRESI
DENEY SONRASI
Sonuçları, çıkış ortamına aktar

Şekil 3.25 Deney sonrası

4. BENZETİM DENEYİNİN TANIMLANMASI

4.1 Benzetim Deneyinin Genel Yapısı

Bölüm 2.2.2'de bilgisayar destekli benzetim ortamlarında, deney tanımlamaları yapılırken, benzetim modelinin, parametrik model yapısında olduğu ve buna göre, parametrik modelde yer alan her bir bileşen model için, değerleri, parametre setlerinde tanımlanan parametre değerleri ve o bileşen modele ait deney modülünün ve deney modüllerini kapsayan, bir de genel deney modülünün tanımlandığı belirilmişti. Benzetim deneyi tanımlamalarında yer alan deney modülleri, davranış üretecinin sağladığı benzetim algoritmasının işleyiş şeklini tanımlar.

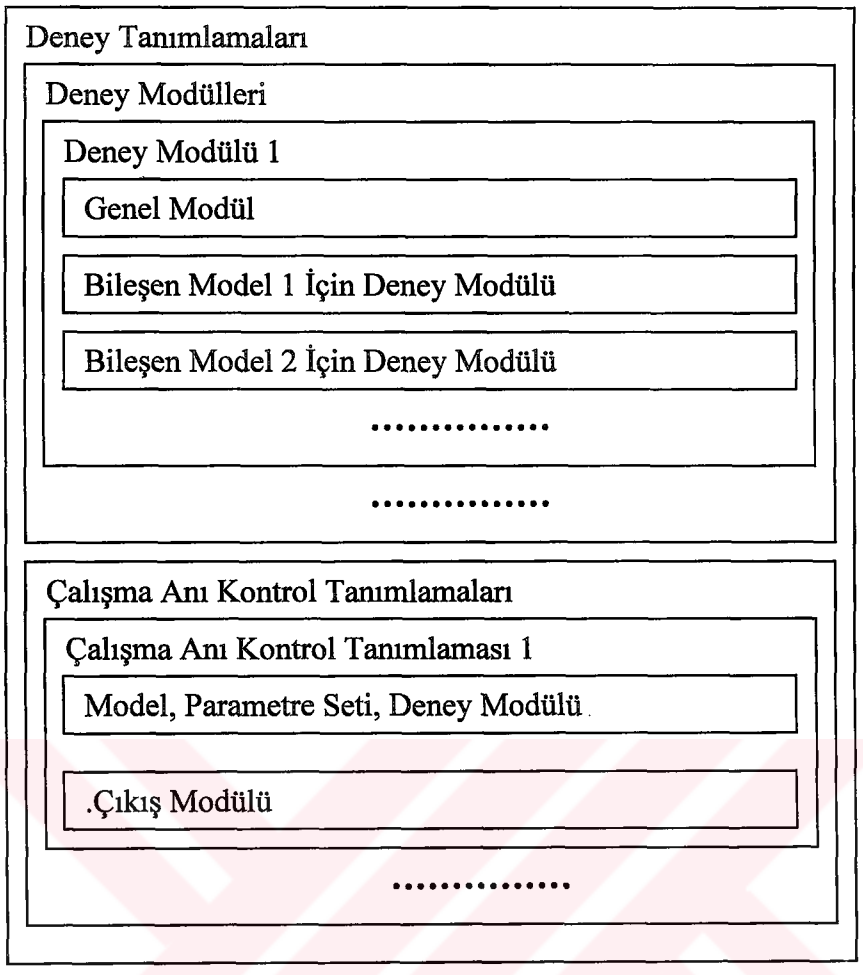
Benzetim deneyinde elde edilen sonuçların, çıkış ortamına ne şekilde aktarılacağı, çıkış modülü tanımlamaları ile belirtilir.

Bir benzetim deneyinin, parametrik model ile eşleşen hangi parametre setine ve hangi deney koşullarında yapılacağı ve sonuçların ne şekilde çıkış ortamına aktarılacağı, çalışma anı kontrol tanımlamaları ile belirlenir.

Bir benzetim deneyinde, birden fazla çalışma anı kontrol tanımlaması yapılabilir. Buna göre, benzetim deneyini bilgisayar ortamında icra eden ve bir modele göre üretilen benzetim programı, farklı parametre değerlerine, farklı deney koşullarına ve farklı çıkış ortamı tanımlamalarına göre birden fazla çalıştırılabilir.

Sonuç olarak, bir benzetim deneyinin tanımlanması aşağıdaki tanımlamaları kapsar : (Şekil 4.1)

- Deney modüllerinin tanımlanması
- Çıkış modüllerinin tanımlanması
- Çalışma anı kontrol tanımlamaları



Şekil 4.1 Benzetim deneyi

4.2 Deney Modüllerinin Tanımlanması

Benzetim deneyi tanımlamalarında yer alan deney modüllerine ilişkin tanımlamalar, yukarıda ifade edildiği gibi, davranış üreticinin sağladığı algoritmanın işleyiş şeklini belirtir. Bu tanımlamalar, genel modül ve parametrik modelde yer alan her bir bileşen modele ait deney modülü tanımlamalarını içerir.
Bölüm 2.2.2'de açıklamaları yapılan bu kavramlara, bu bölümde tekrar değinilecektir.

4.2.1 Genel Deney Modülünün Tanımlanması

Bir deney tanımlamasında yer alan genel deney modülü,

- Algoritmada kullanılan yöntem.
- Benzetim deneyi için belirlenen zaman aralığı,

tanımlamalarını kapsar.

Burada belirtilen, algoritmada kullanılan yöntem kavramı, sürekli modellerin durum geçiş işleminin modellenmesinde kullanılan diferansiyel denklemlerin, analitik çözümleri bulunmadığı taktirde uygulanacak sayısal entegrasyon yöntemini ifade eder. Benzetim deneyi için belirlenen zaman aralığı, sayısal entegrasyon adımı olup, sabit ya da değişken olabilir.

4.2.2 Bileşen Modellere Ait Deney Modüllerinin Tanımlanması

Bir parametrik modelde yer alan bileşen modellere ait deney modülleri tanımlamaları,

- Durum değişkenlerinin ilk değerlerini
- Modelin verilen zaman aralığında, alabileceği giriş değişkenlerinin en küçük ve en büyük değerleri
- Deneyin hangi koşullarda sona ereceği

Buna göre, sayısal entegrasyon işlemlerinde, diferansiyel denklemlerin bağımlı değişkenleri olan durum değişkenlerinin ilk değerleri, bileşen modellere ait deney modüllerinde tanımlanır. Giriş değişkenlerinin değerleri, belirli koşullara göre sınırlandırılabilir. Entegrasyon işleminde, giriş, çıkış ve durum değişkenlerinin aldığı değerler göre, entegrasyon aralığının sonuna gelmeden, deney sonlandırılabilir.

4.3 Çıkış Modüllerinin Tanımlanması

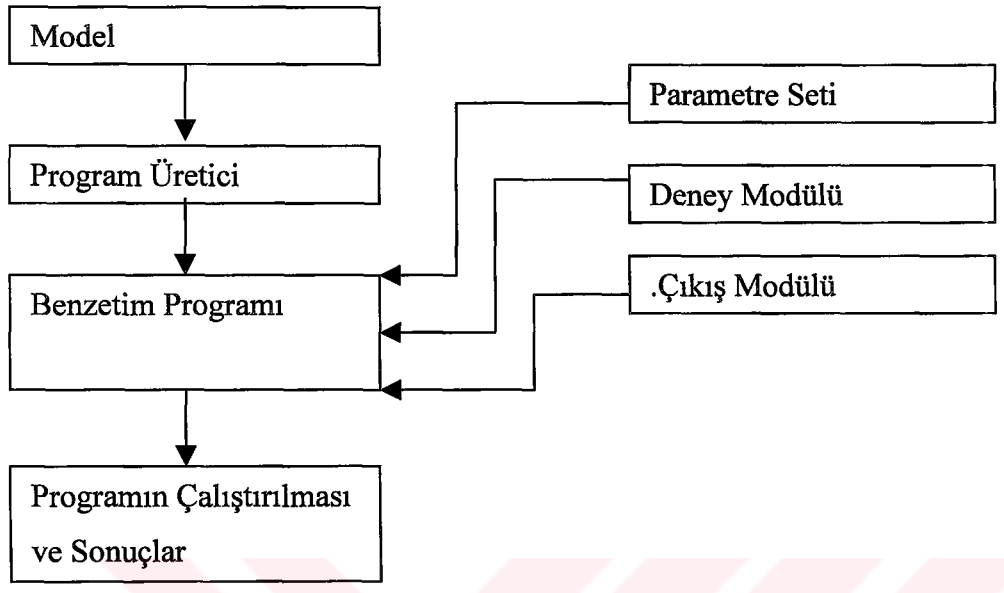
Çıkış modüllerinde, deney sonunda, iki boyutlu değişken-zaman ekseninde hangi model değişkenlerinin görüleceği belirtilir.

Bu değişkenler, bellekli modeller için giriş, çıkış ve durum değişkenleri, belleksiz ve bağlaşık modeller için ise, sadece giriş ve çıkış değişkenleridir. Bellekli modeller için yardımcı değişkenler de çıkış ortamına aktarılabilir. Ancak, bu tez çalışmasında bu durum ihmal edilmiştir.

4.4 Çalışma Anı Kontrol Tanımlamaları

Bir çalışma anı kontrol tanımlaması, bir parametrik modeli, bir parametre seti, bir deney modülü tanımlaması ve bir çıkış modülü tanımlaması ile eşleştirilir.

Buna göre, bir modele göre üretilen benzetim programı (bölüm 5.3), belli bir parametre seti, belli bir deney modülü ve belli bir çıkış modülüne (bir vertabanından) göre çalıştırılır. (Şekil 4.2)



Şekil 4.2 Benzetim deneyinin çalıştırılması

5. BENZETİM PROGRAMININ ÜRETİLMESİ

5.1 Genel Yaklaşım

Bilgisayar destekli benzetim sistemlerinde, daha önce de belirtildiği gibi, benzetim modelinin (parametrik model), parametre setlerinin, deney koşullarının ve çıkış

ortamlarının tanımlanması, bilgisayar ortamının kullanıcıya sağladığı, etkin arayüz ve dosya işleme (veritabanı) olanaklarıyla, otomatik olarak yapılır.

Buradaki "otomatik" kelimesi, kullanıcının sadece, sistem probleminin çözümüne dolayısıyla modelin tanımlamasına ve diğer tanımların (parametre setleri, deney koşulları ve çıkış ortamları) yapılmasına odaklanması geretiğini ifade eder.

Bilgisayar ortamında, bir sistem, bir model ile temsil edildiğinde, modelin üzerinde benzetim deneyinin gerçekleştirilmesi ve sonuçların elde edilmesi, bir "benzetim programının" yardımıyla olur. Bu benzetim programının işlevi, daha önce anlatılan benzetim algoritmasını, farklı parametre değerlerine, deney koşullarına ve çıkış ortamına ilişkin tanımlamaya göre çalıştırıp, sonucu kullanıcıya sunmaktır.

Bir benzetim sisteminde, bir model, birden fazla parametre seti ve deney koşullarına göre benzetim deneyine tabi tutulabilir ve sonuç, yapılan birden fazla çıkış modülü tanımlamasından herhangi birine göre elde edilir.

Daha önce belirtildiği gibi, bir benzetim algoritması bellekli bileşen modellerin durum geçiş algoritmalarını ve bağlaşık modelde yer alan bileşen modeller arasındaki giriş-çıkış tanımlamalarına (bağlaşım) göre değer aktarım yapısını içerir. Bileşen modellerin statik ve dinamik yapısının, ve bağlaşık modellerde yer alan bağlaşım tanımlamalarının değişmesiyle benzetim algoritması da değişir. Bu durumda, benzetim algoritmasını çalıştıran benzetim programının yapısı, modelin yapısına bağlıdır. Dolayısıyla, model değiştikçe, benzetim programı da değişir.

Bilgisayar destekli benzetim sistemlerinde, otomatik olarak yapılan tanımlamalarla birlikte, benzetim algoritmasını çalıştıran benzetim programı da, otomatik olarak üretilir. Burada kullanıcının bilgisayar programlamayı ve programı derlemeyi bilmesi

gerekmez. Kullanıcı yeni bir model tanımladığında ya da model üzerinde bir değişiklik yaptığında, buna uygun program üretilir ve daha sonra derlenip çalıştırılır. Eğer model değişmiyorsa, varolan program, verilen parametre seti, deney koşulları ve çıkış modülüne göre çalıştırılır.

Benzetim programının üretilmesi iki aşamada gerçekleştirilir. İlk aşamada, bir benzetim diline göre, tanımlama bloklarının yer aldığı, "tanımlama programı (kaynak program)", ikinci aşamada ise, model tanımlama bloğunda yer alan ifadelere göre, "benzetim programı (kaynak program)" üretilir.

5.2 Tanımlama Programının Üretilmesi

Bu aşamada, benzetim sistemine ilişkin tanımlamaların yer aldığı ve bir benzetim diline göre oluşturulan, benzetim tanımlama dosyası (programı) üretilir. Burada ifade edilen benzetim dili, her biri, benzetim sistemine ait belli bir kavramın tanımını içeren tanımlama bloklarından oluşan bir kurallar bütünüdür. Bölüm 2.2'de ifade edildiği gibi, bu tanımlama blokları şunlardır: (Şekil 2.4)

- Model tanımlama bloğu
- Parametre seti tanımlama bloğu
- Deney koşulları tanımlama bloğu
- Çıkış ortamı tanımlama bloğu
- Benzetim işleminin çalışma anı kontrol tanımlamaları

Model tanımlama bloğu, benzetim sistemine ait model tanımlarını içerir. Eğer model, bileşen model ise, tanımlama bloğunda, modelin statik ve dinamik yapısına ilişkin tanımlamalar yer alır.(Şekil 3.3) Benzetim modeli, bağlaşık model ise, tanımlama bloğu, iç ve dış bağlaşım tanımlamaları ile bağlaşık modelde yer alan tüm bileşen modellerin tanımlamalarını içerir. (Şekil 3.4)

Parametre seti tanımlama bloğu, bir ya da biden çok parametre setinin tanımlandığı bloktur. Her bir parametre seti, model hiyerarşisinde yer alan tüm bileşen modellerin parametre değerlerini içerir.

Deney koşulları tanımlama bloğunda, sırasıyla, entegrasyon koşulları ve hiyerarşide yer alan belekli bileşen modellere ait durum değişkenlerinin ilk değerleri yer alır.
Çıkış ortamı tanımlama bloğu, hiyerarşide yer alan modellerin benzetim işleminin sonucu olarak, iki boyutlu grafik ortama aktarılacak değişkenlerinin tiplerini ve

isimlerini içerir. Bu değişkenler bellekli modeller için giriş, çıkış ve durum, belleksiz ve bağlaşık modeller için de giriş ve çıkış değişkenleridir. (Bölüm 4.4)

Benzetim işleminin çalışma anı kontrol tanımlamaları ise, benzetim modeline göre üretilen programın, hangi parametre setine, hangi deney koşullarına ve hangi çıkış ortamı tanımlamasına göre çalışacağını gösterir.

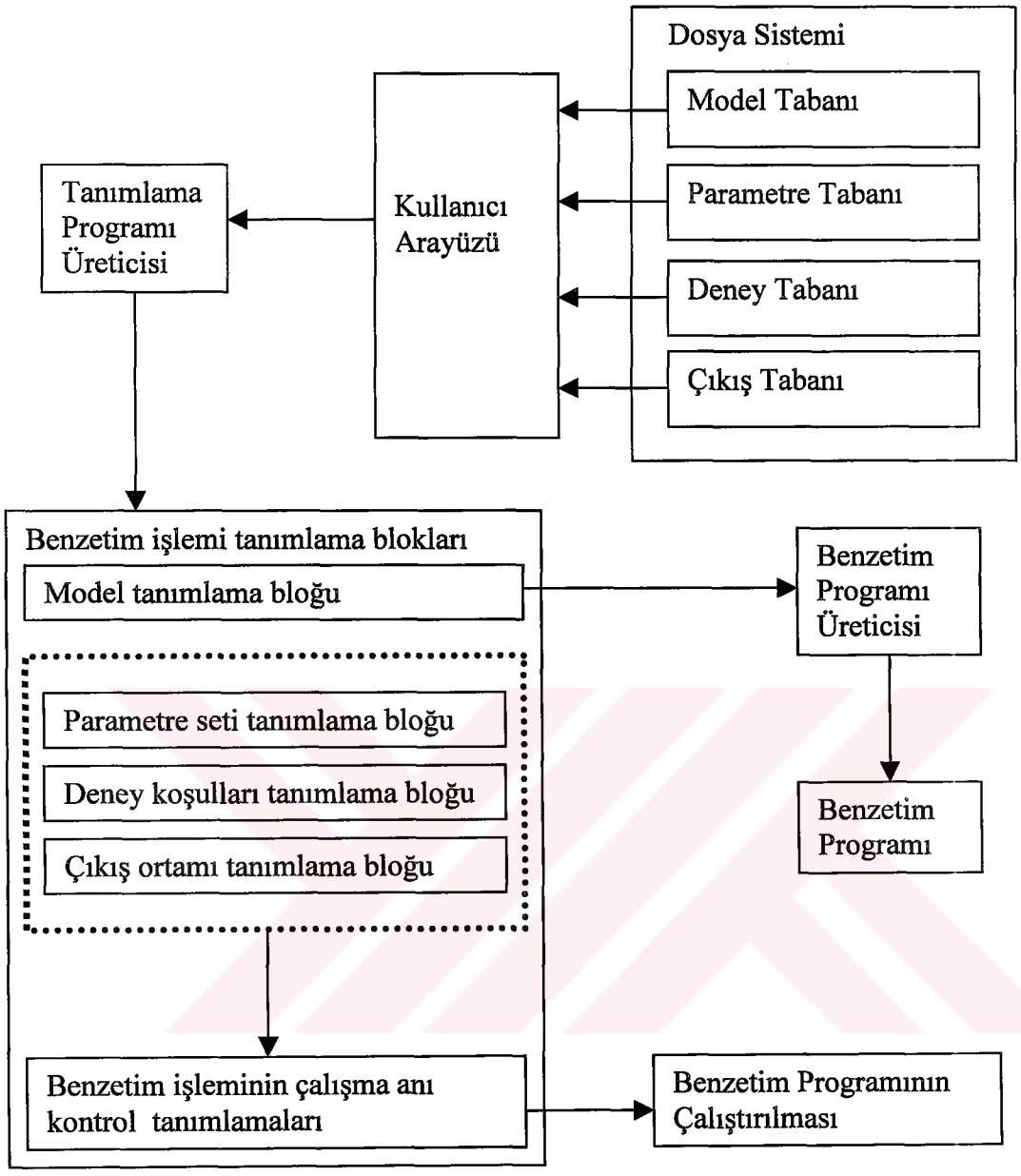
Bu tez çalışmasında göz önüne alınan benzetim dilinde, Ören tarafından geliştirilen "GEST (GEneral System Theory)" benzetim dili esas alınmıştır. Bilgisayar ortamında bu tanımlamalar otomatik olarak gerçekleştiğinden kullanıcının benzetim dilinin gramerini bilmesine gerek yoktur.

5.3 Benzetim Programının Üretilmesi

Benzetim programı, ilk aşamada üretilen "tanımlama programına" göre üretilir ve çalıştırılır. Yukarıda ifade edildiği gibi, benzetim algoritmasının işleyişi dolayısıyla, benzetim programının yapısı, modelin yapısına bağlıdır.

Buna göre, benzetim programı, model tanımlama bloğunda tanımlanan modele göre üretildikten ve derleme işleminden sonra, "seçilen çalışma anı kontrol tanımlamasına" göre çalıştırılır. Dolayısıyla, tanımlama bloklarından, model tanımlama bloğu, benzetim programının üretilmesinde, diğer bloklar ise, çalıştırılmasında kullanılır.

Şekil 5.1'de, gösterilen yapıda, kullanıcı arayüzü ile yapılan tanımlamalar, daha önce belirtildiği gibi, deneyden önce kullanıcı arayüzünün sağladığı görsel tanımlama olanakları ile ya da dosya sistemi yardımıyla yapılır.



Şekil 5.1 Benzetim programının üretilmesi ve çalıştırılması

5.3.1 Benzetim Programının Genel Mimarisi

Modele göre üretilen benzetim programı, mimari olarak ele alındığında, tamamıyla modüler olup, tablo tabanlı bir yapıya sahiptir.

Programın tablo tabanlı bir yapıda olması, benzetim modeli üzerinde yapılan deneyin dolayısıyla benzetim programının, farklı çalışma zamanlarında, farklı deney koşullarına, farklı parametre setlerine ve farklı çıkış ortamı tanımlamalarına göre çalışabileceğini ifade eder. Benzetim programı belli bir anda çalıştırıldığında, sonraki

bölümlerde anlatımı yapılan "dosya erişim modülleri" yardımıyla, seçilen değerler ve tanımlamalar dosyadan okunarak programa veri olarak aktarılır.

Benzetim programının modüler bir yapıda olması iki farklı açıdan değerlendirilir:

- Programı üretilmesi
- Programın çalıştırılması

Programın üretilmesi bakımından ele alındığında, modüler bir benzetim programında, tanımlanan modele ait her bir bileşen, modelin kendisi, erişim modülleri ve diğer yardımcı modüller (çalışma anı kütüphaneleri), "nesne tabanlı programlamanın" yapı taşı olan "nesneler" ile temsil edilir.

Bir programın üretiminde ya da yazılmasında, her bir program bileşeninin bir nesne ile temsil edilmesi, modüler programlamanın sağladığı "yeniden kullanım" ilkesinin temelini teşkil eder. Buna göre bir program kodu üretildiğinde, programa ait yardımcı modüller ve daha önceden oluşturulmuş nesne kodları tekrar üretilmez. Örneğin, bir bileşen modeli temsil eden bir nesne kodunu ele alalım. Bu bileşen modeli içeren farklı bağlaşık modeller için kod üretildiğinde, her defasında, aynı bileşen modeli temsil eden kod üretilmeyip, genel program koduna ilave edilir.(Bölüm 5.3.2)

Programın modülerliği, çalıştırılması bakımından ele alındığında, programın çalıştırılması için gereken parametre seti, deney koşulları ve çıkış ortamlarına ilişkin tanımlama ve değerler, farklı şekilde elde edilebilir. Örneğin bu değerler bir dosya sisteminden elde edilebileceği gibi, üretilen programda, sabit değer olarak alınabilir. Dosya erişim modüllerinin ayrı bir program bileşeni olarak ele alınmasıyla, algoritmanın işlendiği modülden bağımsız olarak, farklı formata sahip veritabanlarından değer okunabilir. Veritabanının farklı formata sahip dosya yapılarından oluşması, algoritmanın işlendiği modülü etkilemez.

Bir benzetim programının çalışması, çalıştırdığı algoritmanın işleyişine paralel olarak, üç ana safhadan oluşur. Bunlar, bölüm 3.4.8'de anlatıldığı gibi

- Deney öncesi
- Deney süresi
- Deney sonrası

safhalarıdır.

Deney öncesinde modele ait parametre seti, deney ve çıkış ortamı tanımlaması seçilir. Yapılan seçime göre elde edilen parametre setindeki değerler, ilgili model nesnesinin "parametre seti değer dosyasına" yazılır ve ilk çalışma anında bu dosyadan okunur.

Deney tanımlaması, daha önce belirtildiği gibi, bir genel modül ve bileşen modeller ait deney modüllerinden oluşur. Bu deney modülleri, bellekli bileşen modellere ait durum değişkenlerinin ilk değerlerini içerir. Genel modülde yer alan tanımlamalar, ana programa ait "genel bir dosyaya", bileşen modellerin durum değişkenlerinin ilk değerleri de, ilgili model nesnesinin "ilk değer dosyasına" yazılır ve yine ilk çalışma anında bu dosyadan okunur.

Deney süresinde, bölüm 3.4.8'de anlatılan algoritma, "algoritma işleme modülü" ile çalıştırılır. Bu algoritma bileşen modellerde yer alan durum geçiş algoritmalarını, bağlaşık modellerde ise, bağlaşım algoritmalarını kapsar. Bağlaşım algoritması, hangi model nenesinin, hangi model nenesine ve hangi giriş değişkenine değer göndereceğini belirtir.

Deney sonrasında ise, yine bir genel dosya olan "çıkış dosyasına" çıkış ortamı tanımlamasına göre yazılan ve hangi modellerin, ve bu modellere ait hangi değişkenlerin, iki boyutlu grafik ortamda görüleceğini belirten ifade okunur ve sonuçlar, her modelin kendine özel "sonuç dosyasına" aktarılır.

Burada dikkat edilmesi gereken nokta, bileşen modellere ait tanımlamaların, o bileşen modelin kendi dosya sisteminde saklandığı, genel tanımlamaların ise, ana programın kullandığı genel dosya sisteminde saklandığıdır. Bu yapı, benzetim programının modülerliği yanında, dosya sisteminin modülerliğini de yansıtır.

5.3.2 Model Tanımlarının Program Koduna Dönüştürülmesi

Bir benzetim modeli, parametrik yapıda olup, birden fazla bileşen model içerebilir. Bu tür, bağlaşık model özelliği gösteren bir model, benzetim algoritmasıyla işlendiğinde, bu algoritmayı çalıştıran benzetim programının, "nesne tabanlı" bir yapıda olması, yukarıda anlatılan "programın modüler bir yapı teşkil etmesi" kavramını destekler.

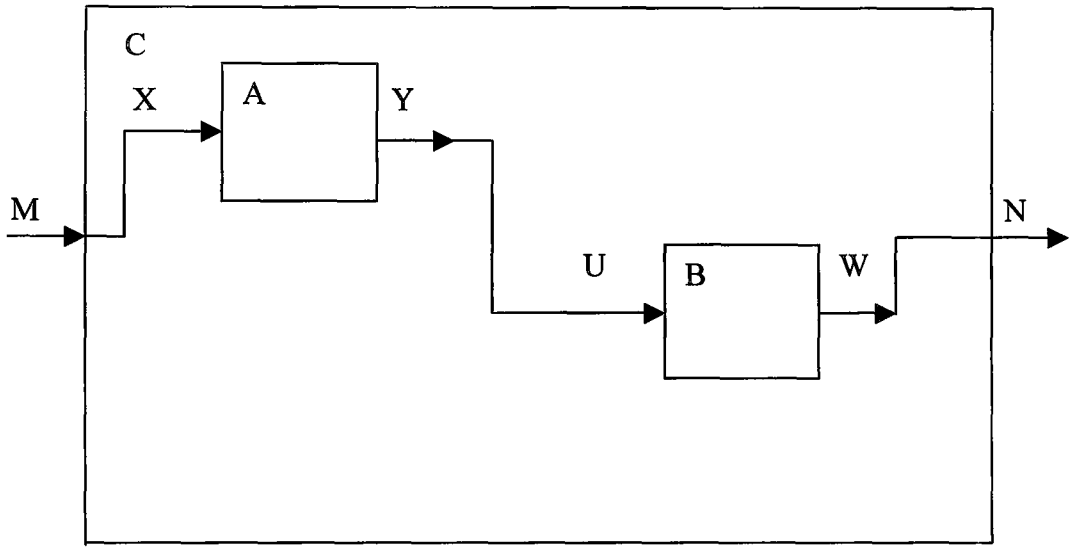
Bu tür bir program mimarisi, ilk kez Zeigler tarafından ortaya atılmıştır. Buna göre, model hiyerarşisindeki tüm modeller, programda bir "nesne" ile temsil edilir.

Modeller arasındaki giriş-çıkış ilişkileri (bağlaşım) ile nesne tabanlı programlamadaki, "nesneler arasındaki mesaj alış verişi" kavramları arasında birebir paralellik vardır.

Hiyerarşideki bileşen modellere ait "nesne kodlarında", eğer model bellekli ise durum geçiş algoritmaları (örneğin, sürekli bileşen modeller için sayısal entegrasyon algoritmaları) ve durum değişkenlerinin değerlerinin, çıkış değerine dönüştürüldüğü, çıkış fonksiyonları çalıştırılır. Eğer model belleksiz ise, modele ait nene kodlarında, durum geçiş algoritmaları bulunmaz.

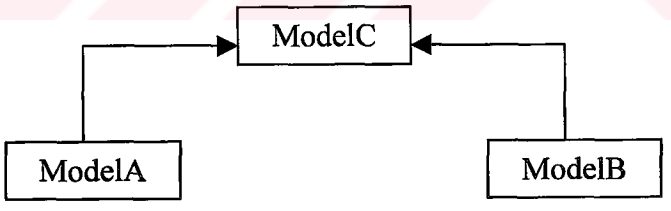
Bellekli modellerde, ayrıca, "internete bağlantı modülü" de yer alır. Bu modül de programda bir "statik sınıfla" temsil edilir. Buradaki statik sınıf, nesnesi oluşturulamayan ve sadece metotları (fonksiyonları) kullanılabilen sınıfları ifade eder. İnternete bağlantı modülü, sürekli bir bileşen modelin içerdiği diferansiyel denklemin internet ortamındaki veritabanında, analitik çözümünün bulunup bulunmadığını kontrol eder. Eğer analitik çözüm varsa, durum geçiş algoritmasında, çözüm olan fonksiyonunun her t anı için değeri hesaplanır. Eğer analitik çözüm yoksa, sayısal entegrasyon algoritması çalıştırılır.

Model hiyerarşisinde yer alan bağlaşık modelleri temsil eden nesne kodları, bağlaşık modelin içerdiği bileşen modelleri temsil eden nesneler arasındaki mesaj alış verişini yönetir. Bağlaşık bir modeli temsil eden bir nesne, bağlaşık modelde yer alan bileşen modelleri temsil eden nesneleri "veri yapısı olarak" içerir. Nene tabanlı programlamada bu kavram, "nesnelerin birbirini içermesi" olarak adlandırılır. Diğer nesneleri içeren bir nesne, içerdiği nesnelere "mesaj göndererek", o nesnelere ait kodları çalıştırır.



Şekil 5.2 Bağlaşık modellerin bileşen modelleri içermesi

Şekil 5.2'de, bir bağlaşık model olan C bağlaşık modeli ile, bu modelin içerdiği, A ve B bileşen modellerinin tanımlama diyagramları görülmektedir. Buna göre C modelini temsil eden nesneye ait sınıf, ModelC, A ve B modellerini temsil eden sınıflar da, sırasıyla, ModelA ve ModelB olmak üzere, ModelC sınıfına ait bir nesne, sırasıyla ModelA ve ModelB sınıflarına ait nesneleri içerir. Bu ifadenin şematik yapısı aşağıdaki "sınıf diyagramında" gösterilmiştir.



Şekil 5.3 ModelA ve ModelB sınıflarına ait nesneleri içeren ModelC nesnesinin sınıf diyagramı

Nesne tabanlı sistem analiz ve tasarımında, kullanılan metodolojiye göre, sınıf diyagramları farklı yapıda olabilir. Bu tez çalışmasında, herhangi bir metodolojiden bahsedilmeyip, sadece sınıf nesnelerinin birbirlerini içermesi ve sınıfların türetilmesi, basit olarak ele alınmıştır.

Nesne tabanlı programlama yöntemi kullanılarak üretilen bir benzetim programında, bir bağlaşık modeli temsil eden nesnenin, içerdiği ve bileşen modelleri temsil eden nesneler mesaj göndermesi yapılan bağlaşımın, iç ya da dış bağlaşım olmasına göre iki şekilde olabilir.

Yapılan bağlaşımın iç bağlaşım olması durumunda, daha önce de belirtildiği, giriş-çıkış değer aktarımları, sadece, bileşen modellerin kendi aralarında gerçekleştirilir. Örneğin şekil 5.2'deki A modelinin Y çıkış değerinin, B modelinin U giriş değerine aktarılması bir iç bağlaşımdır.

Bağlaşık ve bileşen modelleri temsil eden nesneler, giriş ve çıkış değişkenlerine değer atayan ve bu değerleri okuyan "erişim fonksiyonları" içerirler. Aynı durum bellekli modellerde durum değişkenleri için de geçerlidir.

Buna göre, yukarıda belirtilen iç bağlaşımda, A modeline ait erişim fonksiyonuyla, Y çıkış değişkeninin değeri alınır. Alınan bu değer, daha sonra, B modelinin erişim fonksiyonuyla, U giriş değişkeninin değerine atanır. İki aşamalı olan bu iç bağlaşım işleminde, çıkış değerinin okunması ve bu değerin giriş değişkenine atanması belli bir t anı için gerçekleşir. Nesne tabanlı programlamada, bir nesneye mesaj gönderme işlemi, o nesnenin fonksiyonlarını çalıştırarak yapılır.

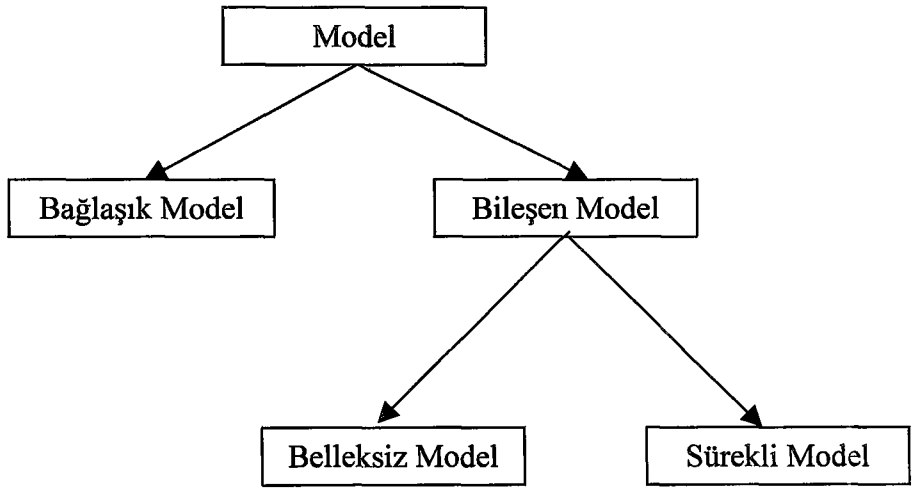
Bağlaşık modelde, dış bağlaşıma göre yapılan değer aktarımlarında, dış bağlaşım oluşturan bileşen modellerin erişim fonksiyonları kullanılarak o bileşen modellerin giriş ve çıkış değişkenlerinin değerleri okunur ya da bu değişkenlere değer atanır.

Yukarıdaki C modelinde, C modelinin M giriş değişkeninin değeri, A modelinin, erişim fonksiyonları kullanılarak, X giriş değişkenine atanır. B modeline ait erişim fonksiyonları kullanılarak elde edilen W çıkış değişkeninin değeri C modelinin N çıkış değişkenine atanır.

Benzetim programında modelleri temsil eden nesnelere ait sınıflar ortak özelliler taşır. Örneğin, her model nesnesi giriş ve çıkış değişkenlerinin değerlerini okuyan ya da bu değişkenlere değer atayan erişim fonksiyonları içerir. Bu ortak özellikler, bir üst düzey sınıfta toplanıp, "türetme" ya da "miras alma" yoluyla, bu sınıfın özel türleri olan yeni sınıflar, elde edilebilir

Nesne tabanlı programlamada, sınıfların miras alma yöntemi kullanılarak, türetilen (veri yapısı ve fonksiyonları miras alınan) sınıflar ile bu sınıfların özel bir türünü

teşkil eden yeni sınıflar, bir "sınıf hiyerarşisi" oluştururlar. Bu hiyerarşide, yukarıdan aşağıya inildikçe bir önceki sınıfın özel bir türü elde edilir.



Şekil 5.4 Model sınıf hiyerarşisi

Yukarıdaki şekilde, benzetim programında kullanılan model sınıflarına ait sınıf hiyerarşisi gösterilmektedir. Bu yapıdaki, "Model" sınıfı, tüm modellere ait özellikleri içerir. Bunlar, giriş-çıkış değişkenlerine değer atayan ve bu değerleri okuyan erişim fonksiyonlarıdır. "Model" sınıfı, "Bileşen Model" ve "Bağlaşık Model" olmak üzere, iki özel tür sınıfa ayrılır. "Bileşen Model" sınıfı da bileşen modellerin farklı türlerini temsil eden, "Belleksiz Model" ve "Sürekli Model" sınıflarına ayrılır. Ayrık modeller, bu tez konusunun dışında olduğu için, bu modelleri temsil eden sınıf, sınıf hiyerarşisinde yer almamıştır.

Benzetim programı üretilirken, modeli temsil eden nesne sınıfı, tanımlanan modelin türüne göre, daha önceden kodları oluşturulmuş ve hiyerarşik yapısı, şekil 5.4'de gösterilen sınıflardan türetilir. Türetildikten sonra, tanımlama bloklarındaki model tanımlama bloğunda yer alan ve modelin kendine has özelliklerini belirten tanımlamalar, program kodu şeklinde, nesne koduna ilave edilir.

Şekil 5.3'de sınıf diyagramları görülen sınıflardan, "ModelC" sınıfı, "Bağlaşık Model" sınıfından, "ModelA" ve "ModelB" sınıfları da, "Belleksiz Model" ya da "Sürekli Model" sınıflarından türerler.

Benzetim programı üretiminde, bir modelin nesne kodu, benzetim dilinin gramer yapısına göre oluşturulan, tanımlama programının (dosyasının) içerdiği model tanımlama bloğuna göre üretilir.

Bu işlemde, tanımlama programının yazıldığı dosyada yer alan ifadeler okunarak, çözümlenir. Benzetim dili, bloklu bir yapıya sahip olduğundan, her bir blok için ayrı çözümleyici kullanılır. Örneğin, model tanımlama bloğu için kullanılan çözümleyici, model tanımlamalarının program koduna dönüştürülmesinde, parametre seti bloğu için kullanılan çözümleyici de parametre değerlerini, model nesnelerinin dosya sistemine yazılmasında kullanılır.

Model tanımlamalarının program koduna dönüştürülmesi, modelin bileşen ya da bağlaşık olmasına göre farklılık gösterir.

5.3.2.1 Bileşen Model Tanımlarının Program Koduna Dönüştürülmesi

Bir bileşen model, benzetim programında, sınıf diyagramı şekil 5.4'de gösterilen, sınıf hiyerarşisindeki "Bileşen Model" sınıfından türeyen bir sınıfla temsil edilir. Daha önce de belirtildiği gibi, bileşen model iki ana yapıdan meydana gelir. Bunlar, statik yapı ve dinamik yapıdır.

Statik yapıda, modele ait giriş, çıkış ve durum değişkenleri, yardımcı değişkenler, parametreler, yardımcı parametreler ve sabitlere ilişkin tanımlamalar yer alır.

Bileşen modelin statik yapısında yer alan değişken, parametre ve sabit tanımlamaları, model nesnesine, veri yapısı olarak aktarılır.

Dinamik yapıda, durum geçiş fonksiyonları (model bellekli ise) ve çıkış fonksiyonları yer alır. Durum geçiş fonksiyonlarının yer aldığı blok, yardımcı değişkenlerin hesaplandığı matematiksel ifadeler ile durum değişkenlerinin karakterize ettiği diferansiyel denklemleri içerir. Çıkış fonksiyonlarında ise, çıkış değişkenlerinin değeri, durum değişkenleri ve yardımcı değişkenlere bağlı olarak hesaplanır.

Örneğin, aşağıdaki ifadelerde, giriş değişkeni X, durum değişkeni, Y, çıkış değişkeni Z, yardımcı değişkenleri, U ve V, ve parametresi de P olan bir modelin durum geçiş fonksiyonları,

U=P+X

V= U/3-P (5.1)

Y'=Y+X-U*V+P

ifadelerinden oluşsun. Bu ifadelerdeki birinci ve ikinci eşitliklerde yardımcı değişkenler, giriş değişkenleri ve parametre değerlerine bağlı olarak hesaplanır. Üçüncü ifadede ise, durum değişkeninin karakterize ettiği diferansiyel denklem yer alır. Bu diferansiyel denklem, parametrik yapıda olup, giriş değişkenlerini, yardımcı değişkenlerini modele ait parametre değerlerini, parametre olarak kabul eder. Giriş ve yardımcı değişkenlerine değeri her t anında yeniden değer atanır.
Modelin belli bir t anındaki çıkış değeri de,

Z=P*Y (5.2)

ifadesiyle hesaplanabilir.

Durum geçiş ve çıkış fonksiyonlarında yer alan matematiksel ifadeler, tanımlama programında bir metin dosyası halinde saklanır. Program üretilirken bu ifadeler, hesaplama işlemlerinde, çözümleyici modüller yardımıyla, matematiksel olarak anlamlı hale getirilir. Şöyle ki, (5.2) eşitliği bir "metin dosyası satırı" olarak okunur. Daha sonra bu bilgi çözümlenir. Bu çözümlemeye göre belli bir t anında, Z değişkeninin değerine (programlama anlamında), P ve Y değişkenlerinin değerlerinin çarpımına eşitlenir.

5.3.2.2 Bağlaşık Model Tanımlarının Program Koduna Dönüştürülmesi

Bir Bağlaşık model, benzetim programında, sınıf diyagramı şekil 5.4'de gösterilen, sınıf hiyerarşisindeki "Bağlaşık Model" sınıfından türeyen bir sınıfla temsil edilir.

Bağlaşık modelleri temsil eden nesneler, daha önce de belirtildiği gibi, bağlaşık modeldeki bileşen modelleri temsil eden nesneleri içerirler. Bir bağlaşık modelin tanımlama bloğu, daha önce de ifade edildiği gibi, üç tanımlama bloğu içerir. Bunlar, dış bağlaşım tanımlamaları, bileşen modellerin tanımlama blokları ve iç bağlaşım tanımlamalarıdır. (Şekil 3.4)

MODEL C

...

MODEL A

... (5.3)

END MODEL A

...

MODEL B

...

END MODEL B

...

END MODEL C

Şekil 5.2' de tanımlama ve bağlaşım diyagramı verilen C modelinin, GEST benzetim diline göre, tanımlama programındaki ifadeleri, (5.3)'deki gibidir.

İngilizce terimler kullanılarak yazılan bu ifadeler, model hiyerarşisine göre, iç içe tanımlama blokları içerirler. En dıştaki blok, ana bağlaşık modeli, içteki bloklar da, bu bağlaşık modele ait bileşen modelleri tanımlar.

En iç bloktan başlayarak, bileşen modellere ait tanımlama bloklarının program koduna dönüştürülmesi bir önceki bölümde anlatıldığı şekilde gerçekleştirilir. Dıştaki bloklara ait nesne kodları,bileşen modelleri temsil eden nesne kodlarını, daha önce belirtilen "içerme" yoluyla içerirler.

Bağlaşık modelin iç ve dış bağlaşım tanımlamalarının program koduna dönüştürülmesiyle, bağlaşık modele ait nesnenin, bileşen modellere ait nesneler ile (dış bağlaşım) ve bileşen modellere ait nesnelerin birbirleriyle (iç bağlaşım) erişim fonksiyonları yardımıyla, mesaj alış verişi sağlanır. Dış bağlaşımda kullanılan, bağlaşık modele ait giriş-çıkış değişkenleri ve içerdiği bileşen modellerin nesneleri,, bağlaşık modele ait nesne koduna veri yapısı olarak yazılır.

5.3.3 Parametre Değerlerine Erişim Modülleri

Çalışma anı kontrol tanımlamalarında, bir parametrik model ile, bir parametre seti, bir deney koşulları ve bir çıkış ortamı tanımlaması eşleşir.
Seçilen parametre setine ait değerler ve parametre isimleri, seçim işleminden sonra, oluşturulan model sınıfına ait "parametre değer dosyasına" yazılır.

Dosyadaki parametre değerleri, "parametre değerlerine erişim modülleri" tarafından okunarak, model nesnesine veri olarak aktarılır. Bu modüller statik bir sınıftır.

5.3.4 İnternet Bağlantı Modülü

Bellekli modellerde davranış üretiminden önce, durum geçiş fonksiyonlarında yer alan diferansiyel denklemlerin analitik çözümleri, internet ortamındaki bir veritabanından taranır. Eğer, çözüm varsa, belirlenen koşula uygun çözüm davranış üeticiye verilir. Davranış üreteci de, her t anı için çözüm fonksiyonunun değerini hesaplar.

İnternet ortamı ile bağlantı, "internet bağlantı modülü" ile yapılır. Diferansiyel denklemin ifadesi "sıralama modülü" ve diğer dönüşüm modülleri ile veritabanına uygun hale getirilir. Eğer çözüm yoksa, sayısal entegrasyon işlemi uygulanır.

İnternet bağlantı modülü de, parametre değerlerine erişim modülleri gibi statik yapıda bir sınıftır.

5.3.5 Deney Koşulları Tanımlamalarına Erişim Modülleri

Deney koşulları tanımlamaları, genel modül ve bileşen modellere yönelik deney modüllerinden oluşur. Genel modülde, sayısal entegrasyona yönelik tanımlamalar yer alır. Bu tanımlamaların yer aldığı dosyaya, ana program (nesne sisteminin koordinasyonunu sağlayan) tarafından erişilir. Bu erişim, statik yapıya sahip, "deney koşulları tanımlamalarına erişim modülleri" tarafından sağlanır.

Bileşen modellere yönelik deney modülleri, bellekli modeller için durum değişkenlerinin ilk değerlerini içerir. Bu bilgilere erişim yöntemi, parametre değerlerine erişim yöntemiyle aynıdır.

5.3.6 Çıkış Ortamı Tanımlamalarına Erişim Modülleri

Model nesnelerinin hangi değişkenlerinin değerlerinin, grafik ortamda görüleceği, "çıkış ortamı tanımlama dosyalarına" yazılır. Bu dosyalara erişim, statik sınıf yapısındaki "çıkış ortamı tanımlamalarına erişim modülleri" tarafından sağlanır.

5.3.7 Program Sonuçlarının Dosya Sistemine Aktarılması

Benzetim programı sonunda elde edilen sonuçlar,"çıkış ortamı sonuç dosyalarına" yazılır. Bu dosyalarda her bir model nesnesinin üretilen değişken değerleri yazılır. Bu değerler daha sonra, okunarak grafik ortama aktarılır.

5.3.8 Çalışma Anı Sınıf Kütüphaneleri

Bu sınıf kütüphanelerini oluşturan program kodları, program üretilirken yeniden üretilmeyip, nesne kodlarında hazır olarak kullanılırlar. Bunlar şu şekilde sıralanabilir:

- Parametre değerlerine, deney koşulları ve çıkış ortamı tanımlamalarına erişim modülleri
- Model çözümleyiciler
- Sayısal entegrasyon algoritmaları
- İnternete erişim modülleri

5.4 Benzetim Programının Derlenmesi ve Çalıştırılması

Model tanımına göre üretilen benzetim programı, sistem tarafından, kullanılan derleyiciyle göre, derlenip çalıştırılır. Derleme anında hangi sınıf dosyasının derlendiği, derleme editöründe görüntülenir.

Program çalıştığında, "çıkış ortamı sonuç dosyalarından", üretilen değişken değerleri grafik ortama aktarılır.

Eğer model yeni tanımlanmıyorsa, program üretilip derlenmez, sadece çalıştırılır. Bu tezde, tasarlanan sistemde, program, java programlama dilinde üretilir.

6 MODEL DAVRANIŞININ ÜRETİLMESİ VE İŞLENMESİ

6.1 Adi Diferansiyel Denklemler ve Çözüm Yöntemleri

Modern matematiğin önemli dallarından bir olan diferansiyel denklemler, fiziksel sistemlerin modellenmesinde ve sistem benzetiminde büyük yer tutar.

Bir diferansiyel denklem,

$$y' = \frac{dy}{dt} \tag{6.1}$$

olmak üzere, bir bilinmeyen y=y(t) fonksiyonu, ve bu fonksiyonun en az birinci mertebeden türevini içeren matematiksel ifadedir. Örneğin,

$$y'' - y' * (t-1) = 0 \tag{6.2}$$

ifadesinde olduğu gibi.

Buradaki diferansiyel denklem, "adi" diferansiyel denklem, olup, tek bir değişken bağlı fonksiyon ve türevlerini içerir. Bir diferansiyel denklemin mertebesi, ifadesinde yer alan en yüksek mertebeden türevin derecesine eşittir.

Bir sistemin modelini karakterize eden bir diferansiyel denklem ve o diferansiyel denklemin bilinmeyen fonksiyonunun ve türevlerinin değerleri, "tek bir noktada" verilebilir. Bu tür matematik modellere, "başlangıç değer problemi" denir. Örneğin,

$$y'' - y' - \cos(t-1) = 0$$
$$y(1) = 0 \; ; \; y'(1) = 1 \tag{6.3}$$

ifadelerinde, diferansiyel denklemin yanı sıra, t=1 noktasındaki ilk değerleri de verilmiştir. Eğer diferansiyel denklem ve o diferansiyel denklemin bilinmeyen fonksiyonunun ve türevlerinin değerleri, birden çok noktada verilirse, Bu tür modellere ise, "sınır değer problemi" denir. Örneğin,

y'''+2*y''-y'-cos(t-1)=0
y(0)=1 ; y'(0)=2
y(1)=0 ; y'(1)=1 (6.4)

ifadelerinde ise t=0 ve t=1 noktalarındaki değerler verilmiştir.

Birinci mertebe lineer diferansiyel denklemler, aşağıdaki üç tür formattan birinde olabilir,

$$y'=f(t,y) \tag{6.5}$$

$$M(t,y)dt+N(t,y)dy=0 \tag{6.6}$$

ve

$$y'+p(t)*y=0 \tag{6.7}$$

(6.5) denklemi ile temsil edilen modele göre, bağımsız değişkenin türev değeri, bağımlı ve bağımsız değişkenlerin fonksiyonuna eşittir.

(6.6) denkleminde, f=f(t,y) olmak üzere bu fonksiyonunun kısmi türevleri M ve N fonksiyonları olup, fonksiyonunun diferansiyeli sıfırdır. Son fonksiyonda ise, p(t) sürekli bir fonksiyondur.

Daha yüksek mertebe lineer diferansiyel denklemler, bir fonksiyonun en az ikinci mertebe türevini içeren diferansiyel denklemlerdir.

Lineer olmayan diferansiyel denklemler bir fonksiyonun kendisinin ve türev değerlerinin üstel olarak en az ikinci mertebe veya bağımsız değişkenle bağımlı değişkenin ya da türev ifadelerinin çarpım terimleri oluşturduğu denklemlerdir.

(6.7) denkleminde, eşitliğin sağ tarafı sıfıra eşittir. Bu tür diferansiyel denklemlere,"homojen diferansiyel denklem" adı verilir. Eğer denklemin sağ tarafının sıfırdan farklı olması durumunda bu tür tür diferansiyel denklemlere ise,"homojen olmayan diferansiyel denklem" adı verilir.

Bir diferansiyel denklemin analitik çözümü, diferansiyel denklemi karakterize eden çözüm fonksiyonun parametrik ifadesidir. Bir diferansiyel denklemin analitik çözümü,

verilen başlangıç ya da sınır değerlerine bağlı olarak birden fazla olabilir. Çözümün parametrik yapıda olmasının nedeni, ifadesinde denklemin başlangıç koşulu kadar entegrasyon sabiti içermesidir. Bu sabit değerleri, başlangıç koşullarına bağlı olarak bulunur ve parametrik yapıda olan çözümün fonksiyonun kesin ifadesi bulunmuş olur.
Bir diferansiyel denklemin analitik çözümü, her zaman bulunamaz. Bu durumda, sayısal entegrasyon işlemi uygulanır.

Sayısal entegrasyon işlemlerinde amaç, diferansiyel denklemin kesin çözüm fonksiyonunu elde etmek değil, muhtemel çözüm fonksiyonlarının verilen bir sayı aralığında alabileceği değerleri hesaplamaktır.

Sayısal entegrasyon işlemlerinde farklı sayısal entegrasyon algoritmaları kullanılır. (Runge Kutta, Euler..) Sayısal entegrasyon algoritmalarında, önce, sayı aralığının ilk değeri için çözüm fonksiyonunun değeri ve sayı aralığındaki iki değer arasındaki fark (adım uzunluğu) verilir. Daha sonra kullanılan ayısal entegrasyon algoritmasına göre, çözüm fonksiyonunun bir sayı aralığındaki bir sonraki nokta için değeri hesaplanır. Bu işlem, aralığın son noktasına kadar devam eder.

Örneğin, y(t) fonksiyonunu ele alalım. Bu fonksiyonun karakterize ettiği, (6.5) diferansiyel denklemini sayısal entegrasyon yöntemiyle çözmek istediğimizde, $[t_0, t_n]$ aralığı için önce $y(t_0)=y_0$ başlangıç değeri ve $h=y_i-y_{i-1}$ adım uzunluğu verilir.Daha sonra, kullanılan ayısal entegrasyon algoritmasına göre, ardışık olarak, $t_1, t_2,..., t_n$ değerleri için, $y_1, y_2, ..., y_n$ değerleri hesaplanır.

Ayrıca, diferansiel denklemrin analitik ya da sayısal çözümlerinin elde dilmesinde, bilgisayar destekli problem çözme ortamları da kullanılır..

6.2 Model Davranışının İşlenmesi

Bir benzetim sisteminde, modelin temsil ettiği sistemin davranışları hakkında bilgi edinebilmesi için model davranışının, bir "davranış üretici" tarafından üretilmesi gerekir. Buradaki davranış üretici tabiri, modelin işlendiği benzetim algoritması ve bu algoritmayı çalıştıran benzetim programını ifade eder. Bir modelin davranışı, belirli deney koşulları ile deneye tabi tutulan model üzerinde, davranış üretici tarafından üretilen veriyi ifade eder.

Daha önce de belirtildiği gibi bellekli modeller durum değişkeni ve buna bağlı olarak durum geçiş fonksiyonlarına sahiptir. Benzetim programı, bu durum geçiş

fonksiyonlarını, belli bir zaman aralığında, bir benzetim algoritmasıyla işler. Bu işlem sırasında, aralığın "deney koşulları ile belirlenen" noktaları (t_1, t_2,..., t_n) için, durum değişkenlerinin ve buna bağlı olarak da çıkış fonksiyonları ile çıkış değişkenlerinin değeri hesaplanır. Hesaplanan bu değerler, deney sırasında davranış üretici tarafından model üzerinde üretilen veriyi dolayısıyla modelin davranışını ifade eder. Davranış üretici tarafından üretilen değerler zamana bağlı olarak belirlenir.

Bir modelin davranış üretici tarafından üretilen gözlenebilen davranışı üç türlüdür. Bunlar,

- Yörüngesel davranış
- Yapısal davranış
- Noktasal davranış

olarak sınıflandırılabilir.

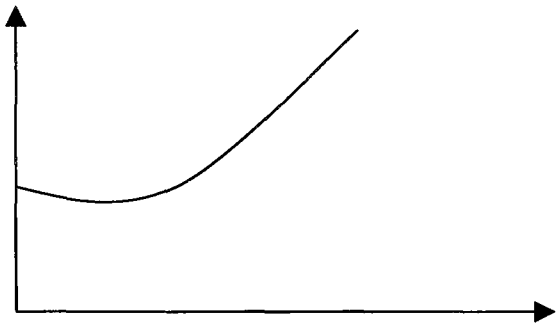
Verilen bir y değişkeninin , [t_0, t_n] aralığında, belirlenen t_0, t_1..., t_n zaman aralığında almış olduğu y_0, y_1,..., y_n değerler kümesine, o değişkenin yörüngesi denir Benzetim ortamlarında en çok karşılaşılan davranış türü, yörüngesel davranıştr. Örneğin diferansiyel denklemlerle modellenen sistemlerde sayısal entegrasyon algoritmaları, yörüngesel davranış üretir.

Yapısal davranış modelin, fiziksel yapısının değişimi karakterize eder. Bu tür davranışların üretilmesinde, modelin dinamik yapısında durum geçiş fonksiyonlarına ek olarak modelin temsil ettiği sistemin yapısal hareketin yansıtan fonksiyonlar da bulunur.

Yörüngesel ve yapısal davranış, modelin dinamiğini yansıtırken, noktasal davranış modelin statik yapısındaki değiikliklere ne şekilde cevap vereceğiniölçmek için üretilir. Örneğiin modelin parametre değerleri değiştirilip yörüngesel veya yapısal davranışı üretildiğinde, bu değer değişimine göre modelin göstereceği davranış noktasal davranıştır. Optimizasyon işlemlerinde kullanıla noktasalş davranış üretimi, çözüm aramaya değil, varolan modelin statik yapısında değişklikler yaparak ve yeni model alternatifleri üreterek, optimum çözümü bulmaya yöneliktir.

6.2.1 Sonuçların Gösterilmesi

Davranış üretici tarafından, modelin işlenmesiyle üretilen veriler deney anında ya da deney sonrasında kullanıcıya sunulur. Benzetim tanımlamalarından, deney koşullarına ve çıkış ortamları tanımlamalarına uygun olarak, üretilen veriler iki boyutlu zaman ekseninde görüntülenir. (Şekil 6.1)



Şekil 6.1 Davranış üretiminde, sonuçların kullacıya gösterilmesi

6.2.2 Sonuçların Analizi

İki boyutlu eksende t zaman değişkenine göre değişimi gösterilen veriler analiz edilerek modelin temsil ettiği sistem hakkında bilgi edinilir.

7. BİR GÖRSEL MODELLEME VE BENZETİM ORTAMI : GestCell

7.1 GestCell'in Genel Yapısı ve Yazılım Mimarisi

Bölüm 2'de bilgisayar destekli benzetim ortamlarının genel yapısına değinilmişti. Bu bölümde, bir görsel modelleme ve benzetim ortamı olan GestCell yazılımının genel yapısı ve yazılım mimarisi hakkında bilgi verilecektir.

GestCell, Ören tarafından geliştirilen GEST benzetim dilinin bir uygulama yazılımıdır. Daha önceleri farklı tez çalışmalarında ele alınan GestCell'in bu son sürümünde birçok eklentiler içerir.

GestCell'in amacı, sürekli ya da belleksiz modeller ve bu tür modelleri, bileşen model kabul eden bağlaşık modellerin temsil ettikleri sistemlerin davranışlarını incelemektir Bir sistemin davranışı, bölüm 2.1'de de ifade edildiği gibi, sistemi temsil eden modelin benzetim deneyine tabi tutularak anlaşılır.

Belirli deney koşulları, parametre setleri ve çıkış ortamı tanımlamalarına göre gerçekleştirilecek olan benzetim deneyi, bilgisayar ortamında, benzetim programı yardımıyla yapılır. Benzetim programı sabit bir program değildir.

Sistemi temsil eden model, sistem elementleri veya alt sistemler arasındaki ilişkileri yansıtan ifade ve tanımlamaları içerir. Bu tanımlamalar değiştiğinde, sistem davranışlarını ölçen benzetim deneyinin dolayısıyla benzetim programının yapısı da değişir. Bu durumda, yeni bir model tanımlandığında ya da model üzerinde değişiklik yapıldığında, her defasında, yeni bir program yazmak yerine, bu değişiklik ya da tanımlamalara uygun bir programın otomatik olarak üretilmesi büyük esneklik sağlar.

Üretilen program çalış tığında elde edilen sonuçlar,üretilen programın dışındaki bir grafik ortama aktarılır.

Ayrıca, sürekli bir modelin benzetiminde, modele ait diferansiyel denklemin çözümünün internet ortamındaki bir veritabanından alınması, sayısal entegrasyon işlemi yapmadan, çözüm fonksiyonunun değerlerinin elde edilmesini sağlar.

Bütün bu anlatılanlar ışığında, GestCell, genel yapısı itibarıyla, aşağıdaki işlevlere sahiptir:

- Görsel olarak model tanımlama
- Benzetim
- Program üretme
- Dış grafik ortamla bağlantı
- Internete bağlanabilme

GestCell ortamında, model, parametre seti, deney ve çıkış ortamı tanımlama işlemlerinde yazılım sisteminin dosyalama olanaklarından yararlanılır. Benzetim programının çalışması sırasında da aynı durum söz konusudur.

Buna göre, GestCell ortamında, bir bileşen model ve bu modelin içeriği olan matematik model ve birden fazla bileşen model içeren bağlaşık model görsel olarak tanımlanabilir. Bu modele ilişkin benzetim tanımlamaları yapılabilir, bu tanımlamalara uygun benzetim programı üretilebilir, derlenebilir ve çalıştırılabilir, diferansiyel denklemlerin çözümleri internet ortamından sorgulanabilir ve son olarak, benzetim sonuçları grafik ortama aktarılabilir.

GestCell yazılımı, etkin kullanıcı arayüzü tasarlama olanakları sunan, Visual Basic ortamında tasarlanmış ve yazılmıştır.

7.1.1 Model Tanımlama

GestCell ortamında, model tanımlama işlemleri etkin kullanıcı arayüzü sayesinde, tamamıyla "görsel" olarak gerçekleştirilir.

Bilgisayar destekli sistem teorisinde önemli bir yer tutan görsel modelleme veya model tanımlama ortamı, kullanıcıya tanımlanacak olan modelin türüne göre, farklı arayüz olanakları sunar.

Bileşen bir modelin görsel olarak tanımlanması da, modelin giriş-çıkış ya da sürekli bileşen model olmasına göre iki farklı şekilde olabilir.

Eğer tanımlanacak olan model, giriş-çıkış modeli ise, sadece, o modele ait giriş ve çıkış değişkenleri tanımlanır. Bir modelin giriş ve çıkış değişkenleri, görsel olarak, modele giriş ve çıkışları sembolize eden "ok işaretleriyle" gösterilir. (Şekil A.1)

Bileşen model sürekli bir model olduğunda, giriş ve çıkış tanımlamalarının yansıra, modelin statik yapısı da tanımlanır. Modelin statik yapısında, modele ait durum

değişkenlerin (model bellekli ise), parametrelerin, sabitlerin, yardımcı değişkenlerin, yardımcı parametrelerin (Şekil A.2) ve modelin dinamiğini oluşturan, durum geçiş (model bellekli ise) ve çıkış fonksiyonları tanımlanır. (Şekil A.3)

Görsel modlleme ortamında, bu tanımlamalar, kullanıcı arayüzünün dokunmatik denklem editörleri ile yapılır. (Şekil A.4)

Programlama ortamında, bir model, bir "model nesnesi" ile temsil edilir. Bu model nesnesi, bir kullanıcı arayüzü elamenti olan "Visual Basic form penceresi" ve modelin çeşitli bileşenlerini (değişkenler, parametreler, bağlaşık modellerin içerdiği bileşen modeller..) içeren veri yapıları ve diğer nesnelerin çeşitli kombinasyonlarından oluşur. Model nesneleri, görsel olarak, herbiri, ana programa ait ve birden çok Visual Basic form penceresi içeren bir uygulama penceresindeki bir Visual Basic form penceresi ile temsil edilir. (Şekil A.5)

Bir bileşen modelin tanımlanması tamamlandığında modele ilişkin tanımlamalar, tanımlama işlemi bitince, bir model dosyasına kayıt edilir. Bir bileşen ya da bağlaşık model, dosyaya kayıt edildikten sonra, bir "model listesine" eklenir.

Model listesi, programlama ortamında, bir içeren ya da biriktirici (collection) sınıfına ait bir nesnedir. Bu listedeki her bir model, model nesnesine ait, "model adı" değişkeni ile indeks oluşturarak eklenir ya da silinir.

Benzetim ortamına ilişkin tanımlamalar da benzer şekilde, parametre seti, deney koşulları ve çıkış tanımlamaları listelerine eklenir ve dosyaya kayıt edilir.

GestCell ortamında, model tanımlama, sadece görsel modelleme ortamında oluşmaz. Aynı zamanda bu tanımlamalar dosyadan da okunabilir.

GestCell ortamında, bağlaşık modelin oluşturulası üç aşamadan oluşur: İlk aşamada, bağlaşık modele ait giriş ve çıkışlar bileşen modellerde olduğu gibi tanımlanır. Daha sonra, model listesindeki bileşen modeller seçilir ve bağlaşık modele eklenir. Bileşen modellerin bağlaşık modele eklendikten sonra, modeller arasındaki bağlaşım tanımlamaları görsel olarak, fare yardımıyla, çıkış gönderen modelden, bu çıkışı, giriş kabul eden modele doğru, bir çizgi çizerek tanımlanır. (Şekil A.6)

7.1.2 Benzetim

GestCell ortamında, oluşturulan modelin benzetimine yönelik tanımlamalar, sırasıyla, parametre setleri, deney koşulları ve çıkış ortamlarına ilişkin tanımlamalardır.

Model tanımlamalarında olduğu gibi, benzetime yönelik tanımlamalar da GestCell ortamının sağladığı kullanıcı arayüzü ve dosyalama olanakları ile yapılır.Bir modele yönelik benzetim tanımlamaları, , programlama ortamında, bir "benzetim nesnesi" ile temsil edilir.

Benzetim nesnesi, model listesini temsil eden biriktirici (collection) sınıfına ait bir nesne ile aynı yapıda olan, bir parametre setleri listesi , bir deneyler listesi ve bir de çıkış ortamları tanınmları listesi içerir. Her benzetim modeline, bir benzetim nesnesi karşıklık gelir.

Kullanıcı benzetim işlemine ilişkin, parametre, seti deney ve çıkış ortamı tanımlamalarını, "benzetim tanımlama diyalog penceresinden" yönetir (Şekil A.7)

7.1.2.1 Parametre Seti Tanımlama

Daha önce de belirtildiği gibi, benzetim modeli, bir parametrik model olup, çeşitli türdeki bileşen modelleri ve bunlar arasındaki bağlaşım tanımlamalarını içerir. Parametrik modelin tanımlanmasında, ayrıca, parametrik modelde yer alan bileşen modellerin parametre değerlerini içeren, parametre setleri de tanımlanır.

GestCell ortamında bir parametre seti, model hiyerarşisinde yer alan tüm atomik bileşen modellerin parametre değerlerinin verilmesiyle tanımlanır.

Bit parametre seti tanımlanırken önce, "parametre seti tanımlama diyalog penceresi" ekrana gelir. (Şekil A.8) Eğer model atomik bir bileşen model ise, parametreler listelenir. Kullanıcı, listeden parametreyi seçip, değer atama işlemine geçtiğinde, karşısına çıkan diyalog penceresinden parametre değerini girer (Şekil A.9). Eğer model bağlaşık bir bileşen model ise (iç içe hiyerarşik bir bağlaşık model ise), modelin bağlaşık model olduğu mesajı kullanıcıya verilir ve bu modelin bileşen modelleri için parametrelere değer atama işlemine geçilir. Girilen parametre değerleri, paramtre listesinde parametre adıyla görüntülenir. (Şekil A.10)

Bir modelin parametre değerleri tanımlanırken, bir diğer modele geçmek için, modelin tüm parametrelerine değer verilmesi gerekir. Aksi durumda, bir uyarı mesajı çıkar.

Bir parametre setinin tanımlanma işleminin tamamlanası için, tüm atomik bileşen modellerin parametre değerlerinin tanımlaması gerekir

Parametre seti tanımlandıktan sonra, dosyaya kayıt edilir ve aktif parametre listesine eklenir (Şekil A.11)

Bir parametre setinin tanımlaması, parametre setinin bir dosyadan okunmasıyla da gerçekleştirilebilir.

Parametre seti, dosyadan okunduğunda, dosyanın, dosyada tanımlanan parametre setinin model ile uyumlu olup olmadığı kontrol edilir. Bir parametre seti, dosyaya kayıt edilirken, modelin adı da dosyaya yazılır. Daha sonra möodelin içerdiği bileşen modellerin isimleri ve parametreleri değerleriyle birlikte, dosyaya yazılır. Dosya okunduğunda, eğer modelin adı dosyada varsa, dosya bir çözümleyici yardımıyla okunarak bir paramere setiioluşturulur ve parametre seti listesine eklenir.

Dosyada tanımlanan parametre seti, model ile uyumlu değilse, parametre seti dosyaya eklenmez ve kullanıcıyua uyarı mesajı verilir.

7.1.2.2 Deney Tanımlama

Deneyin tanımlanması, genel deney modülü ve bileşen modellere yönelik deney koşullarının tanımlanmasından oluşur.

Genel deney modülü, sayısal entegrasyon koşullarını içerir. Bileşen modellere yönelik deney koşulları da, bileşen modellere ait (model bellekli ise) durum değişkenlerinin ilk değerlerini içerir.

Yeni bir deney tanımlandığında, "deney tanımlama diyalog penceresi" ekrana gelir. Bu diyalog penceresinden, entegrasyon koşullarına yönelik tanımlamalar yapılır. Bunlar, zaman aralığının başlangıç ve bitiş değerleri, ve entegrasyon adım uzunluğudur. (Şekil A.12)

Aynı diyalog penceresinden bileşen modellere ait durum değişkenlerinin ilk değer tanımlamalarına geçilir. Bileşen modellere ait durum değişkenlerinin ilk değer tanımlamalarında, bir önceki bölümde anlatılan, parametre seti tanımlamalarında

uygulanan yöntem bu kez, durum değişkenleri için uygulanır. (Şekil A.13) Eğer model bellekli model ise, durum değişkenleri listelenir ve kullanıcı bu listeden durum değişkenlerini seçer ve değerini girer. Bir sonraki modele geçebilmek için, bütün durum değişkenlerine değer atanmlıdır. Bileşen modelin bağlaşık bir model olması durumunda, yukarıdaki yöntem uygulanır. Girilen durum değişkenlerinin ilk değeri, listede görüntülenir. (Şekil A.14)

Tüm bu tanımlamalar bittiğinde, deney, bir dosyaya ve programda yer alan bir deney listesine kayıt edilir. (Şekil A.15)

Bir deney yukarıda anlatılan yöntemle tanımlanabileceği gibi, bir dosyadan da okunabilir. Eğer deney, bir dosyadan okunuyorsa, yukarıda olduğu gibi, deneyin modele uygunluğu kontrol edilir. Deney modele uygunsa, dosya okunarak yeni bir deney nesnesi oluşturulur ve deney listesine eklenir, uygun değilse, kullanıcıya uyarı mesajı verilir.

7.1.2.3 Çıkış İşleminin Tanımlanması

Çıkış işlemlerinin tanımlamalarında, benzetim programı çalıştığında, iki boyutlu grafik ortamda, hangi değişkenlerin görüleceği belirtilir.

Bu değişkenler, daha önce de belirtildiği gibi, bellekli bileşen modeller için, giriş, çıkış ve durum değişkenleri, belleksiz ya da bağlaşık modeller için ise, sadece giriş ve çıkış değişkenleridir

GestCell ortamında, parametrik bir modelin benzetiminde, benzetim programının çalışması sonunda, program sonuçlarının ekrana nasıl yansıyacağı, çıkış modülü tanımlamaları ile belirlenir.

Bir çıkış modülü, programlama ortamında, bir çıkış modülü nesnesi ile temsil edilir. Bu nesne, parametrik modelin ve parametrik modelde yer alan bileşen modellerden, hangi modelin, hangi değişkenlerinin, iki boyutlu grafik ortama yansıyacağını belirten veri yapılarının bir biriktirici (collection) nesnesi içerir.

GestCell ortamında, bir çıkış modülü tanımlanırken, "çıkış modülü tanımlama diyalog penceresi" ekrana gelir. Bu dialog penceresinde, herbir bileşen model için, önce, grafik ortama aktarılacak değişkenlerin hangi değişken gurubuna ait olduğu belirlenir. Daha sonra, seçilen gurba ait değişkenler listelenir ve bu listeden değişken seçilir. (Şekil A.16)

Çıkış modülü tanımlandıktan sonra, oluşturulan çıkış modülü nesnesi, çıkış ortamları tanınmları listesine eklenir ve yapılan tanımlamalar bir dosyaya yazılır.

Diğer tanımlamalarda olduğu gibi, gerektiğinde, çıkış işlemlerinin tanımlaması da bir dosyadan okunur ve uygunluk kontrolünden sonra, yapılan tanımlama, bir çıkış modülü nesnesi olarak çıkış ortamları tanınmları listesine eklenir.

7.1.3 Benzetim Programının Üretilmesi

Bölüm 5'de anlatıldığı gibi, bir benzetim programı, benzetim algoritmasını çalıştırır. Bu programın yapısı, modelin yapısına bağlı olup, program tablo tabanlı olarak çalışır.Buna göre, bir modele göre üretilen benzetim programı, farklı parametre setlerine, deney koşullarına ve çıkış ortamı tanımlamaları için çalıştırılabilir.

Her bir benzetim modeli ve bu modelin içerdiği bileşen modeller, bir nesne ile temsil edilir. Her bir model nesnesi, parametre değerlerinin ve durum değişkenlerinin (model bellekli ise) yazılabildiği bir dosya sistemine sahiptir. Ayrıca programın kendisi de, global tanımlamaların yer aldığı bir dosya sistemi içerir

GestCell'de öncelikle, bölüm 5'de anlatıldığı şekilde, önce, model, parametre seti, deney koşulları ve çıkış ortamı tanımlamalarının yer aldığı, tanımlama programı üretilir. GestCell ortamında görsel olarak yapılan bu tanımlamalar, GEST benzetim dilinde, tanımlama programı dosyasına yazılır.

Daha sonra, tanımlamaların yer aldığı tanımlama programı dosyasından, çözümleyiciler yardımıyla, model tanımlamaları, program koduna dönüştürülür.

Kullanılan benzetim dili, bloklu bir yapıdadır. Model tanımlama bloklarında, her bir model, bir blokta tanımlanır. Program üreticide yer alan çözümleyiciler, her bir model tanımlama bloğunu, java programlama dilinde yazılan ve programlama ortamında bir model nesnesini temsil eden program koduna dönüştürürler.

Programın çalışma anında kullanacağı ve hazır olarak kodlanmış, çalışma anı kütüphaneleri de java programlama dilinde yazılmıştır.

Programı derleyen java derleyicisi sistem komut satırından, GestCell tarafından çalıştırılır. Aynı durum, programı çalıştıran java yorumlayısı için de geçerlidir.

7.1.4 Grafik Ortam

Bernzetim programı çalıştırıldıktan sonra, çıkış ortamı tanımlamalarında belirtilen modeller ve modellere ait değişkenler, iki boyutlu grafik ortamada görüntlenir. Programın ürettiği sonuçlar, ana programın erişebileceği sonuçlar dosyasınna yazılır. Daha sonra, bu değerler, GestCell tarafından okunarak, GestCell programındaki bir OLE (Object Linking Embeding) nesnesi olan Excel nesnesi tarafından iki boyutlu grafik olarak ekranda görüntülenir.

OLE nesneleri sayesinde, Microsoft Windows ortamında, bir pogramın içinden, bir başka program bir nesne olarak alınıp çalıştırılır.

GestCell ortamında Visual Basic içinden bir OLE nesnesi olarak Excel programı çalıştırılır.

7.1.5 İnternet Bağlantısı

Benzetim algoritmasında, sürekli modellere ait diferansiyel denklemler üzerinde sayısal entegrasyon işlemi yapılır. Eğer denklemin analitik çözümü varsa, sayısal entegrasyon işlemine gerek kalmaz.

Bu noktadan hareketle, internet ortamında, adi diferansiyel denklemlerin analitik çözümlerini içeren bir veritabanı oluşturulmuştur.

Bu veritabanına erişim, java dilinin internet ortamında yer alan CGI programlarıyla iletişim kurabilme yetenekleriyle sağlanır. İnternet ortamındaki veritabanın genel yapısı ve işleyişi, bölüm 8'de detaylı olarak anlatımıştır.

Benzetim programı çalışırken, internet ortamına erişim model nesnelerinde yer alan internet bağlantı modülleriyle sağlanır. Modelin içerdiği diferansiyel denklem, sıralama algoritmasıyla (bölüm 8) veritabanınına uygun hale getirilip, internet ortamında yer alan veritabanı erişim modüllerine aktarılır. Veritabanı erişim modülleri de, SQL sorgulama diliyle, veritabanını tarayıp, çözüm varsa koşula uyan çözümü, sisteme akarılar.

8. İNTERNET ORTAMINDA ADİ DİFERANSİYEL DENKLEMLERİN ÇÖZÜMLERİNİ İÇEREN BİR VERİTABANI YÖNETİCİSİ:ODENET

8.1 Giriş

Bu bölümde, internet ortanımda çalışan ve birinci, ikinci ve üçüncü mertebeden lineer veya nonlineer bir adi diferansiyel denklemlerin çözümünü içeren ve yerel bir kişisel bilgisayarda bulunan GestCell benzetim ortamı ile ileşim kurabilen, "ODENET" veritabanı yönetimi sistemi tanıtılmaktadır.

Öncelikle, sistem için önerilen yaklaşım ve veritabanının genel yapısı tanıtıldıktan sonra, sistemin işevsel yapısını oluşturan, sıralama algoritması, arama işlemleri ve kullanıcı arayüzü hakkında bilgi verilecektgir. Son olarak, sistemin yazılım mimarisi ele alınmıştır.

8.1.1 Önerilen Yaklaşım

Sürekli bir sistemin, matematik modeli, diferansiyel denklemlerin çözümüne indirgenir. Elde edilen diferansiyel denklemin çözüm fonksiyonları, sistemin matematik model tarafından yansıtılan dinamiğini ortaya koyar.

Ancak, sürerkli sistem benzetiminde ve mühendislik problemlerin çözümünde önemli bir yer tutan adi diferansiyel denklemlerin, analitik çözümlerini bulmak her zaman mümkün olmamaktadır. Bu durumda, diferansiyel denklemleri karakterize eden çözüm fonksiyonlarının, denklemin bağımsız değişkeni için belirlenen sayı aralığında ilk değerleri verilerek, diferansiyel denklemler üzerinde, sayısal entegrasyon işlemi uygulanır.

Ancak, sayısal entegrasyon işleminde kullanılan yöntem, her zaman kesin sonuç vermeyebilir. Bu durumda, diferansiyel denklemin analitik olarak çözmek ya da sayısal entegrasyon işlemi uygulamak yerine, diferansiyel denklemin analitik çözümünü içeren bir veritabanından elde etmek büyük bir esneklik sağlar.

Bu esneklik sayesinde, hem diferansiyel denklemin analitik çözümünü bulmak için gereken zaman azalacak, hem de elde edilemeyen çözümler için, uyulanan ayısal entegrasyon yönteminin kesin sonuç verememe riski ortadan kalkacatır.
Bu noktadan haraketle, Kanada'nın Ottawa Üniversitesi'nde, birinci, ikinci ve üçüncü mertebeden lineer ve nonlineer adi diferansiyel denklemlerin, analitik çözümlerini içeren bir veritabanın tasarımını konu alan bir tez çalışması yapılmıştır. Yerel bir kişisel bilgisayarda, Excel ortamında, çalıştırılan bu sistem, kullanıcı arayüzü ile, girilen bir diferansiyel denklemi önce, bir sıralama algoritmasıyla veritabanına uygun hale getirdikten sonra, veritabanını sorgulayıp, sonucu kullanıcıya iletir

Bu tez çalışmasında tasarlanan sistem, daha önce tasarlanan sistemi esas almakla beraber, birçok ek işlevsel özelliği içerir.

Her şeyden önce sistem internet ortamında çalışıyor olup, temelde, iki amaca yöneliktir: Buna göre, sistem internet ortamında çalışan ve adi diferansiyel denklemlerin, analitik çözümlerini içeren sanal bir el kitabıdır. Kullanıcı, internet ortamından, birinci, ikinci ve üçüncü mertebeden lineer veya nonlineer bir adi diferansiyel denklemin çözümünü elde edebilir.

Sistemin ikinci işlevi de, bir önceki bölümde anlatılan, yerel bir kişisel bilgisayarda çalışan GestCell benzetim ortamıyla iletişim kurabilmektir. GestCell benzetim ortamında, bir benzetim programı çalıştırıldığında, atomik yapıya sahip, bileşen modellerin içerdiği diferansiyel denklemlerin analitik çözümleri, öncelikle veritanından taranır.Eğer çözüm varsa, davranış üretimi, denklemin çözüm fonksiyonlarının aldıkları değerlere göre yapılır. Çözüm yoksa, sayısal entegrasyon işlemini uygulanlır.

8.1.2 Veritabanı

Veritabanı ilişkisel bir yapıda olup, İnternet ortamında, ekleme, arama ve silme işlemlerini yapabilmektedir. İlişkisel veritabanları, tablo yapısındadır ve yönetimleri, standart, SQL sorgulama dili ile yapılır. Bu sistem için kullanılan veritabanı, Microsoft Access 97 dir.

Veritabanı, denklemlerin derecelerine göre, her biri ayrı bir mertebe için olmak üzere, dört adet tablo içerir.

Tablolardaki, bir kayıta(bir diferansiyel denkleme ait tüm bilgiler) erişmek için kullanılan anahtar değişken, denklem ifadesi kullanılır.

8.2 Sistemin İşlevsel Yapısı

8.2.1. Sıralama

Bir sistemin modeli oluşturulduğunda, model, bir diferansiyel denklemin çözümüne indirgendiği zaman, denklemi oluşturan terimler toplama ve çarpma işlemlerinin değişme özelliğine bağlı olarak farklı sırada olabilir. Örneğin;

$$y''+t+y'''+y=0 \tag{8.1}$$

denklemi ile,

$$y'''+y''+y+t=0 \tag{8.2}$$

denklemi matematiksel ifade bakımından eşdeğerdir.

Buna karşılık, bilgisayar ortamı açısından bakıldığında, bu iki ifade, string (karakter dizisi) olarak, birbirlerinden farklılık gösterir. Dolayısıyla, bu iki ifadenin, veritabanında doğru sorgulama yapılabilmesi için, bilgisayar ortamının algılayabileceği şekilde, eşdeğer bir formata dönüştürülmesi gerekir.

Bu dönüşüm kullanıcıya, sisteme giriş yaparken, diferansiyel denklemin terimlerini, istediği terim sırasına göre girebilme imkanı sağlar. Kullanıcı, diferansiyel denklemi girdikten sonra, denklemin terimleri arasında bir sıralama işlemi yardımıyla dönüşüm gerçekleştirilmiş olur.

Aşağıda açıklanan algoritma, orijinal algoritmayla analiz olarak aynı, ancak, nesne tabanlı programlama teknikleri kullanıldığı için, tasarım olarak farklıdır.

Sıralama Algoritması :

Sıralama algoritmasında izlenen temel yöntem, operatör tabanlı sıralama yöntemidir. Bu yöntemde, diferansiyel denklemi oluşturan terimlerin, bir operatöre ve bir de terimsel ifadeye sahip oldukları göz önüne alınır. Örneğin;

$$y+y'+y''-t=0 \tag{8.3}$$

denkleminde, denklemi oluşturan terimlerin operatörleri sırasıyla, "+","+", "+" ve "-" iken terimsel ifadeler ise, "y", "y'", "y''" ve "t" şeklindedir.

Yukarıdaki (8.3) denklemini oluşturan terimler birinci mertebe terimlerdir ve denklem, bu terimlerin lineer toplamı olarak düşünülebilir. Sıralama işlemi, bu terimleri eleman kabul eden liste üzerinde yapılır.

Böylece, sıralama işleminin sonunda, (8.3) denklemi,

$$y''+y'+y'-t=0 \tag{8.4}$$

şekline dönüşür.

Sıralama işlemi, terimleri, toplama veya çarpma işlemlerini karakterize eden ifadeler için yapılır. Bir liste üzerinde yapılan sıralama işleminde, terimlerin sahip oldukları toplama veya çarpma indeksleri kullanılır. Hangi indeksin kullanılacağı, listenin karakterize ettiği işleme göre belirlenir. Örneğin, (8.4) denklemin terimlerin operatörleri "+" veya "-" olduğundan bu liste bir toplama işlemini karekterize eder ve toplama indeksleri kullanılır. Diğer taraftan,

$$y'*t*y''''+y''+y=0 \tag{8.5}$$

şeklindeki bir denklem göz önüne alındığında, denklemdeki "y'*t*y''''", "y''" ve "y" terimleri yine birinci mertebe terimlerdir ve işlem bir toplama işlemidir. Ancak, ilk terim olan "y'*t*y''''" terimi, kendi bünyesinde, terimsel ifadeleri sırasıyla "y'", "t" ve "y''''" olan ve bir çarpma işlemini karakterize eden alt terimlerinden oluşur. Bu terimler ikinci mertebe terimlerdir ve operatör tabanlı sıralama yöntemine göre operatörleri, "*" işaretidir. Sıralama işlemi, bu kez, önce ilk terim olan "y'*t*y''''" teriminin elemanları üzerinde sıralama yapar. Bu işlemin sonunda terim, "t*y'*y''''" şeklini alır. Daha sonra bu terim, yeni şekliyle bir üst mertebedeki terimler listesinde yerini alır. Sıralama işleminin sonunda, denklem,

$$t*y'*y''''+y''+y=0 \tag{8.6}$$

şekline dönüşür.

Denklemin tamamı, toplama işlemini karakterize eden terimlerden oluşur. Çarpım terimleri ise alt terim olarak kabul edilir. Bunun nedeni çarpma işleminin toplama işleminden önce gelmesidir. Bu nedenle,

t*y*y'=0 (8.7)

şeklindeki bir denklem aslında bir elemanlı bir liste ile karakterize edilir. ve eleman terimsel ifadesi "t*y*y'" dir. Buna karşılık, terimin kendisi, üç elemanlı bir liste ile karakterize edilir ve bunlar sırsıyla, "t", "y" ve "y'" terimleridir. Bu üç terim arasında sıralama işlemi yapıldıktan sonra, terimlerin çarpımlarından oluşan terim, bir üst mertebedeki terimler listesine(denklemin kendisi) eklenir. Benzer şekilde, aşağıdaki denklemde altları çizili olan terimler, bir çarpma işlemini karakterize eden alt terimlerdir.

y''*y+y+y*y'*t+y''''=0 (8.8)

sıralama işleminin sonunda denklem,

y''''+ y*y''+ t*y*y'+y =0 (8.9)

şekline dönüşür.

Yukarıda sözü geçen mertebe kavramı, bir terimin hangi seviyedeki listeye ait olduğunu belirler. Eğer bir terim, bir alt terimi karakterize eden bir listeye ait ise, mertebe değeri artar. Örneğin, alt terim içermeyen (8.4) denkleminin tüm terimleri birinci mertebe terimler iken, diğer denklemlerdeki çarpım işlemini karakterize eden listelere ait terimler ise, birer alt terim olup, ikinci mertebe terimlerdir.

Alt terim kavramının gündeme geldiği diğer terimler ise parantezli terimlerdir. Parantezli terimlerin ürettiği alt terim türleri sırasıyla,

- transandantal(trigonometrik, logaritmik, exponensiyel...) fonksiyonların parametreleri, örneğin,

y'''- ln(y+t)+sin(y''''+y+t)=0 (8.10)

- rasyonel ifadelerin pay ya da paydası, örneğin,

y'''-y'+y+t/(y'+y+t)+sin(y'+a) /(y+t)+1=0 (8.11)

- üstel ifadelerin üst ya da tabanı, örneğin,

y''-(y''+y+t)^(y'+t*y+a)+cot(y''+t)^b-c=0 (8.12)

- çarpma işlemine katılan terimler, örneğin,

y'''-t*(y'+t*y+a)+(t+4*y)*(y'''+t*y) =0 (8.13)

şeklinde sıralanabilir.

Yukarıda verilen parantez içindeki olan terimler, ikinci mertebe terimlerdir ve kendi aralarında sıralama işlemine tabi tutulduktan sonra, birleştirilerek, bir üst mertebedeki terimin ifadesinde yeni şeklini alır. Örneğin,

y'''+sin(y-y''+y'''+t)+a=0 (8.14)

ifadesindeki altı çizili olan, "sin" fonksiyonunun parametresi olan, "y-y''+y'''+t" ifadesi, sıralama işleminden sonra, "y'''-y''+y+t" şekline dönüşür. Sonuçta tüm denklem,

y'''+sin(y'''-y''+y+t)+a=0 (8.13)

şeklini alır.

Ancak, bölme, üstel işlem ve transandantal fonksiyonların parametrelerini karakterize eden parantezli terimler dışındaki terimler, yani (8.13) denkleminde olduğu gibi çarpım terimleri, önce "çarpma" işlemine, tabi tutulur, daha sonra, işlem sonunda oluşan çarpım terimleri, kendi aralarında sıralanarak, denklem ifadesine eklenir. Örneğin

y'''+(t+y''-y)*(y'''+sin(y+t)) =0 (8.13)

şeklindeki bir denklem, sıralama işleminden sonra,

t*y'''+y''*y'''-y*y'''+y'''+sin(y+t)*y''-sin(y+t)*y+sin(y+t)*t=0 (8.14)

şeklini alır. Altları çizili terimler, çarpma işlemi sonucunda oluşan ve kendi aralarında sıralanarak, denklem ifadesine eklenen, çarpım terimleridir.

Ayrıca sıralama algoritması, çarpma, toplam-çıkarma, bölme ve üs alma işlemlerinin etkisiz(birim) ve yutan eleman, dağılma, birleşme ve negatif alma özelliklerini destekler.

Parantezli terimlerin iç içe olduğu durumlarda, denklem ifadesinde "(" şeklinde açılmış bir parantez karakterine rastlandığında bir alt mertebeye geçilir. ")"şeklinde kapanmış bir parantez karakterine rastlandığında ise bir üst mertebeye geri dönülür. Bir mertebedeki en dıştaki "(" ve ")" terimleri dışındaki terimler, o mertebedeki listenin elemanlarını teşkil eder. Bu terimler, açılan ve kapanan parantez sayısı birbirine eşit olduğunda, sıralama işlemine tabi tutulduktan sonra, birleştirilerek, bir üst mertebedeki denklemin ifadesinde yeni şeklini alır. Bu yapı, "öz yineleme" yöntemi ile en dıştaki terimler listesine ulaşıncaya tekrarlanır. Örneğin,

y'''+y''+t*(y''''+cos(y+t+y''')-y)+pi=0 (8.15)

şeklindeki bir denklemde, altı çizili olan terim iç içe iki parantezli terim içerir. Sıralama işlemi, önce, en içteki "cos" fonksiyonunun parametresi olan, "y+t+y''''" ifadesini sıralar ve sonuçta terim, "cos(y'''+y+t)" şeklini alır. Daha sonra, "y''''+cos(y'''+y+t)-y" ifadesi sıralama işlemine tabi tutulur ve "y''''-y+cos(y'''+y+t)" şeklini alır. Sonraki adımda, "t*(y''''-y+cos(y'''+y+t))" terim sıralanır ve "t*y''''-t*y+cos(y'''+y+t)*t" şeklini alır. En sonunda denklemin birinci mertebe terimleri sıralanarak,

t*y''''+y'''+y''-t*y+cos(y'''+y+t)*t+pi =0 (8.16)

halini alır.

Terimler arasındaki sıralama işlemi, işlemin türüne bağlı olarak, toplama ya da çarpım indekslerine göre yapılır.

Bir mertebeden, bir üst, ya da bir alt mertebeye geçiş, operatör değişimi ya da transandantal(trigonometrik, logaritmik, exponensiyel...) fonksiyonların parametrelerini karakterize eden parantezli terimler ile belirlenir. Örneğin,

t*y''+y-cos(y''+y'''+t+1) +a=0 (8.17)

denklemindeki ilk terim olan "t*y''" terimi çarpım elamanlarından oluşan bir çarpma işlemidir ve bu işlemi karakterize eden liste, çarpım indekslerine göre sıralanır. Bir sonraki operatör, "+" olduğundan, buna bağlı olarak, bir üst mertebedeki listenin türü değişir ve toplama indekslerine göre sıralanır. Benzer mertebe değişimi, denklemin üçüncü terimi olan, "cos" fonksiyonunun parametresi , "y''+y'''+t+1" ifadesi için de geçerlidir.

Bir alt terimin kendi bünyesinde sırlandıktan sonra, bir üst mertebedeki listede konumlanması, yani, yeni indeksinin belirlenmesi şu şekilde olur :

Eğer, alt terim, bir (8.17) denklemindeki "t*y''" terimi gibi, bir çarpma işlemi ise liste, çarpım indekslerine göre sıralandıktan sonra, en son terimin (y'' terimi) "toplama indeksi" tüm alt terimin (t*y'' terimi) bir üst listedeki indeksi olur.

Eğer, alt terim, "cos(y''+y'''+t+1)" terinde olduğu gibi, bir transandantal fonksiyonun parametrelerini karakterize ediyorsa, (y''+y'''+t+1 terimi gibi) önce parametreyi temsil eden liste sıralanır, sonra transandantal fonksiyonun, ("cos" fonksiyonu gibi) eğer işlem toplama ise toplama indeksi, çarpma ise çarpım indeksi," tüm alt terimin bir üst listedeki indeksi olur.

Bu işlem, eğer ifade rasyonel ise; ifadenin "pay" kısmı için, ifade üstel ise; ifadenin "taban" kısmı için aynen uygulanır.

Herhangi bir listedeki terimler, bir alt seviyedeki terimlerin kombinasyonu nedeniyle indeksleri eşit ise, terimlerin string ifadeleri karşılaştırılır ve karakter sayısı fazla olan önce gelir. Eğer indeksleri eşit ifadeler, tek karakter ise, ascii kodlarına bakılır.

Terimler arasında sıralama toplam ve çarpım terimleri arasında yapılabilirken, aynı durum, "/" ve "^" operatörlerinin karekterize ettiği, rasyonel ve üstel ifadeler için geçerli değildir. Bunun nedeni, bir rasyonel ifadenin pay ile paydasının, ve bir üstel ifadenin üst ile tabanının yer değişmesi durumunda, ifadenin yapısı değişir. Örneğin aşağıdaki denklemler birbirinden farklıdır.

y''+(y''''+y+t)/(y+t-1)-t*y+ln(y+t)^(y''+y)-a=0 (8.18a)

y''+(y+t-1)/(y''''+y+t)-t*y+(y''+y)^ln(y+t)-a=0 (8.18b)

Bu tür terimler, pay ve paydalarında veya, üst ve taban kısımlarında, bir ya da birden çok alt mertebe terim içerdiklerinde, bu terimler alt terim olarak sıralanır ve rasyonel ifadelerde "/" operatörü, üstel ifadelerde ise, "^" operatörü ile birleştirilerek, bir üst mertebedeki yerini, yeni terim olarak alır. Bu terimlerin birleştirilmiş terim olarak ele alınmasının nedeni, bölme ve üst alma işlemlerinin toplama ve çarpma işlemlerinden önce gelmesidir.

Homojen olan ya da olmayan diferansiyel denklemlerde, birinci mertebeden terimler göz önüne alındığında, denklemin herhangi bir yerinde, "=" işareti bulunabilir. Bu durumda, denklemin tüm terimleri, denklemin sol tarafında toplanır ve sıfıra eşitlenir. Bu işlem yapılırken, sol taraftaki terimler aynen terim listesine eklenirken, sağ taraftaki terimler, toplama işlemine göre tersleri alınarak, terim listesine eklenir. Bu şekilde oluşturulan liste, sıralama işlemine tabi tutulur ve denklem sonuna "=0" ifadesi eklenir. Örneğin,

y''+y''''+t*sin(t+y'-y+y'')=y'''+(pi+y''')^(1/2)-pi (8.19)

şeklindeki bir denklem, sıralama işleminden sonra,

y''''-y'''+y''+sin(y''+y'-y+t)*t-(y'''+pi))^(1/2)+pi =0 (8.20)

şeklini alır altı çizili işaret değiştiren terimlerdir.

Eğer denklem, "=" işareti içermiyorsa, sıralama işleminden sonra, denklem sonuna sadece "=0" ifadesi eklenir.

Terimler arasında sıralama işlemi bittikten sonra, eğer ilk terim negatif ise, kendisinden sonra gelen ilk pozitif terim ile yer değiştirir.

Yukarıda belirtildiği gibi, sıralama işleminde amaç, veritabanında doğru sorgulama yapabilmek için, diferansiyel denklemi ifade eden string bilgiyi, bilgisayarın anlayabileceği bir formata sokmak. Bu işlem, terimler arsında sıralama işleminin yanı sıra, alfabetik sabitlerin düzenlenmesi işlemini de içerir. Şöyle ki,

y*t+y'+y''+m*y''+y''''-n+k*y=0 (8.21a)

ve

y*t+y'+y''+p*y''+y''''-q+r*y=0 (8.21b)

denklemlerini göz önüne alalım. Bu iki denklem, matematiksel olarak aynıdır. Ancak, altları çizili alfabetik sabitler açansından bakıldığında, string olarak, birbirinden farklılık gösterir. Bu farklılığı gidermek için, önce bu sabitler tespit edilir ve daha sonra alfabenin ilk karakterinden başlayacak şekilde yeniden düzenlenir ve denklem,

y''''+y''+y'+a*y+b*y''+t*y-c=0 (8.23)

şeklini alır. Eğer denklemde birden fazla aynı isimde alfabetik sabit varsa, sabitin ilk karşılaşıldığı noktadan sonrakiler aynı ismi korur. Örneğin,

y'''+m*y''-t+m+n*y'=0 (8.24)

denkleminin sıralanmamış şekli olan,

y'''-t+a*y''+b*y'+a=0 (8.23)

denkleminde olduğu gibi.

Sıralama işlemi bitikten sonra, en yüksek mertebeden türev tespit edilip, denklemim mertebesi bulunur.

8.2.2 Veritabanında Arama

Veritabanında yapılan arama işlemi sonunda, bir diferansiyel denkleme ait tüm bilgiler kullanıcıya sunulur.
Bu işlem, iki farklı arama yöntemi ile yapılır. Bunlar, "Denklem İfadesine Göre" ve "Denklem Derecesine Göre" yöntemidir.

Denklem İfadesine Göre Arama Yöntemi :

Bu yöntemde, kullanıcı, diferansiyel denklemin ifadesini girer. Daha sonra, bu denklem, sıralama işlemine tabi tutulur ve sıralama işleminden sonra, derecesi

belirlenir. Belirlenen dereceye göre denklem, veritabanındaki, derecelere göre düzenlenmiş dört tablodan birinde aranır.

Denklem Derecesine Göre Arama Yöntemi :

Bu yöntemde, kullanıcıdan, aradığı denklemin derecesini seçmesi istenir. Kullanıcı seçme işlemini yaptıktan sonra, derecelere göre düzenlenmiş dört tablodan, o dereceye ait tablodaki tüm denklemler, listelenir ve kullanıcı, istediği denklemi seçip, o denklem hakkındaki tüm bilgilere ulaşır.

Arama işleminin sonunda, denklem bulunduysa, denklem hakkındaki tüm bilgilerin kullanıcıya sunuluşu, denklemin çözüm sayısına bağlı olarak, iki farklı şekilde olur :

Eğer denklem, bir tek çözüm içeriyorsa, o çözüm kullanıcıya sunulur.

Eğer denklem, koşullara bağlı olarak, birden çok çözüm içeriyorsa, tüm koşullar listelenir. Kullanıcı, herhangi bir koşulu seçtiğinde, o koşula ait çözüm, kullanıcıya sunulur.

8.2.3. Arayüz

Sistemin kullanıcı arayüzü, kullanıcıya veritabanının tüm işlevlerini yerine getirebilme imkanı sağlar. Internet ortamın çalışan bu sistemde, arama işlemleri, java "applet"lerinin baz alındığı, arayüz sayesinde, etkin bir şekilde yapılabilmektedir.

Sistem ilk açıldığında, sisteme giriş karakterize eden "home page" ekrana gelir. Bu sayfadan bir sonraki sayfaya geçildiğinde, arama işleminin başlangıç noktası olan, "Arama Türünü Seçme Appleti" çıkar. Buradan, yapılan seçim sonucunda, kullanıcı, bir diferansiyel denklem ve o denklem ait tüm bilgilere aşağıdaki arama yöntemi ile erişebilir. Bunlar;

Denklem Giriş Editörü İle Arama:

Eğer kullanıcı, "Denklem Giriş Editörü İle Arama" seçeneğini seçerse, arama işlemi için, ekrana yeni bir sayfada "Denklem Giriş Editörü" nün yer aldığı applet gelir. Denklemin girilme işlemi bittikten sonra, denklem sıralama işlemine tabi tutulur ve derecesi belirlenir ve belirlenen dereceye göre ilgili tabloda arama işlemi yapılır.

Denklem Derecesine Göre Arama:

Eğer kullanıcı, "Denklem Derecesine Göre Arama" seçeneğini seçerse, bu kez, ekrana, "Derece Seçme Diyalog Penceresi" gelir ve yeni bir sayfada seçilen dereceye uyan tüm denklemler listelenir ve kullanıcı, seçimini yapar.

Her iki işlemin sonunda da, yeni bir sayfada, arama işleminin sonucunu kullanıcıya sunan "Sonuç Appleti" ekrana gelir. Bu applet kullanıcıya, denklemin çözüm sayısına bağlı olarak, o denklem hakkındaki bilgileri iki farklı şekilde sunar :

Eğer, tek bir çözüm varsa, o çözüm kullanıcıya sunulur.

Eğer, birden çok çözüm varsa, çözümlerin bağlı oldukları koşullar listelenir. Kullanıcı, bu listeden bir koşul ifadesi seçtiğinde, o koşula uyan çözüm kullanıcıya sunulur.

8.3 Sistemin Yazılım Mimarisi

Tasarlanan bu sistem, Kanada'nın Ottawa Ünirversitei"nde yapılan çalışmanın İnternet ortamına uyarlanmış hali olup, temelde, işlevsel olarak aynı, fakat, mimari tasarım olarak, farklıdır.

Bu farklılık, kullanılan yazılım elamanlarının(veritabanı, programlama dili, analiz ve tasarım metodolojisi gibi) farklılığından ileri gelir.

Sistemin, sıralama algoritması ve arayüz modüllerinin tasarımında, nesne tabanlı analiz ve tasarım metodolojisi kullanılmış ve java programlama dili ile programlanmıştır.

Veritabanı modülü ise Microsoft Internet Information Server (IIS) ın kendine özgü dinamik Web tasarımı olanakları ile tasarlanmıştır.

8.3.1. Sistemin Genel Yapısı

Sistem, en genel anlamda, İnternet ortamında çalışabilen, birinci,ikinci ve üçüncü mertebeden lineer olan ve olmayan diferansiyel denklem ve bunların analitik çözümlerini içeren bir veritabanı yönetim sistemidir.

Sistemin işlevsel özellikleri,

- denklemin sıralama işlemine tabi tutulması ve derecesinin belirlenmesi,
- denklem ifadesini ve çözüm takımları ile bunlara bağlı olan koşul ve entegrasyon sabitlerinin girilmesinde kullanıcının arayüz yardımı ile yönlendirilmesi,
- veritabanının internet ortamına aktarılması,

işlemleridir.

Sistemlerin tasarımında kullanılan nesne tabanlı analiz ve tasarım metodolojileri, sistemin birden çok, bileşenden meydana geldiğini ve buların gerek alt sistemler, gerekse, tüm sistem bazında, etkileşim halinde olduğunu kabul eder.
Bu parçacıkların her birine, "nesne", bu nesnelerin bağlı olduğu tanımlama yapılarına da "sınıf" adı verilir.

Her nesne "öznitelikler" ve "metotlar" olmak üzere, iki özellik ile karakterize edilir. Nitelikler, bir nesnenin bünyesinde bulunan veri yapılarını, metotlar ise, nesnenin bünyesindeki veri yapıları üzerinde işlem yapmak için nesneye gönderilen, mesaj veya komutları temsil eder.

Eğer bir nesneye bir komut gönderilirse, nesne "kendi" niteliklerine göre komutu icra eder. Yani her nesne, kendi bünyesindeki verilerin değerini bilir.

Sistemin fonksiyonel özellikleri karakterize eden modüller, nesne tabanlı yaklaşımlar çerçevesinde, tasarlanmış olup, bünyelerinde bulunan sınıf yapıları, aşağıdaki bölümlerde açıklanmıştır.

Aşağıda, sistemin tasarımında kullanılan sınıf yapıları ve bunların bünyelerinde barındırdıkları veri yapıları ile fonksiyonellikleri, "kavramsal" olarak açıklanmıştır.

Denklem :

Bu sınıfa ait bir nesne, bir diferansiyel denklemi temsil eder. Bünyesinde, nitelik olarak, sırasıyla,

- denklem ifadesi,
- denklem derecesi,
- denklem lineritesi,
- çözümler vektörü,
- koşullar vektörü,

- entegrasyon sabitleri vektörü,
- sabit terimler vektörü
- alfabetik olarak sıralanmış sabit terimler vektörü

bulunur.

Terim :

Denklemdeki, herhangi bir seviyedeki terim ve alt terimleri temsil eder. Bünyesinde, nitelik olarak, sırasıyla,

- terim ifadesi,
- terim derecesi,
- terim lineritesi,
- rasyonel ya da üstel ifade ise, ifadenin pay ya da taban kısmı,
- rasyonel ya da üstel ifade ise, ifadenin payda ya da üst kısmı,
- çarpım indeksi,
- toplama/çıkarma indeksi,
- işlem türü indeksi,
- operatör

bulunur.

İsimGurubu :

Değişken ve bunlara bağlı fonksiyonların isimlerini bir arada tutan yardımcı bir sınıftır. Bünyesinde, nitelik olarak, sırasıyla,

- bağımlı değişkenin ismi,
- bağımsız değişkenin ismi,
- bağımlı değişkene bağlı fonksiyonun isimi,
- bağımsız değişkene bağlı fonksiyonun isimi,

bulunur.

LinearityOrder :

Denklemin lineritesini ve derecesini bir arada tutan yardımcı bir sınıftır. Bünyesinde, nitelik olarak, sırasıyla,

- denklemin lineritesi,
- denklemin derecesi

bulunur.

SabitTerim :

Denklem ifadesindeki, alfabetik sabitleri ve bunların denklem içindeki konumunu temsil eder. Bünyesinde, nitelik olarak, sırasıyla,

- sabitin adı,
- sabitin denklem içindeki konumu,

bulunur.

SabitBulucu :

Bu sınıfın tek bir metot (operation)u vardır ve bu metot, "statik" yapıdadır. Bu metotlar diğer sınıflara ait nesneler tarafından, "yardımcı servis" olarak kullanılır. Bu sınıf, denklem ifadesindeki, alfabetik sabitler ve bunların denklem içindeki konumunu tespit edip, alfabenin başından itibaren başlayacak şekilde konumlanmış karşılıklarını bulan yardımcı bir sınıftır.

Çözümleyici :

Denklemin string ifadesini, operatör tabanlı yönteme göre parçalara ayıran(parsing) ve terim listesini oluşturan iç içe(nested) bir sınıftır. Bünyesinde, java dilinin bir sınıfı olan, "Hashtable" sınıfına ait bir nesne bulunur. Bu nesne, referans terimler tablosunu temsil eder. Bu sayede, denklemin string ifadesinden "pars" edilen karakter kümelerinin standart bir terim olup olmadığı ve toplama/çıkarma ve çarpma indeksleri belirlenir. İç içe bir yapıda olmasının nedeni, parantezli terimleri sıralanmasında, algoritmanın "öz yineleme" yöntemini kullanıyor olmasıdır.

VektörYardımcısı :

Bu sınıfın metot(operation)ları, "SabitBulucu r" sınıfında olduğu gibi, "statik" yapıdadır. Bu metotlar diğer sınıflara ait nesneler tarafından, "yardımcı servis" olarak kullanılır. "Parser" sınıfın oluşturduğu, terim listesi java dilinin bir sınıfı olan, "Vektör" sınıfına ait bir nesne ile edilir. Bu sınıfa ait bir nesne, dinamik bir diziyi ifade eder. VektörYardımcısı sınıfı, bu nesne üzerinde işlemler yapar. Bunlar,

- sıralama,
- eğer ilk terim negatif ise, kendisinden sonra gelen ilk pozitif terimle yer değiştirme,
- listedeki en küçük işlem indeksini bulma,

- sıralanmış terimlerden yeni denklem ifadesinin elde edilmesi

işlemleridir.

SonSıralayıcı :

Bu sınıf da "statik" yapıda olup, "SabitBulucu" sınıfının tespit ettiği, alfabetik sabitler ve bunların alfabenin başından itibaren başlayacak şekilde konumlanmış karşılıklarını yer değiştiren, yardımcı bir sınıftır.

Sıralayıcı :

"Denklem" sınıfı ile, "Çözümleyici", "SonSıralayıcı" ve "SabitBulucu" sınıfları arasındaki ilişkiyi kurar.

Dönüştürücü :

String dönüşüm işlemlerini yapar.

DenklemEntryApplet :

Denklem giriş editörünü temsil eder.

8.3.1.2 Sıralama Modülü

Daha önce de bahsedildiği gibi, bir denklem için veritabanında, doğru sorgulama yapabilmek için, denklemin, sıralama işlemine tabi tutulması gerekir.

Bu işlem gerçeklenirken, önce, denklem editörü ile giriş yapıldıktan sonra, denklemi karakterize eden string ile bir "Denklem" nesnesi oluşturulur.

Daha sonra, bu nesne, daha önceden "DenklemEntryApplet" nesnesi bünyesinde oluşturulmuş, "Sıralayıcı" nesnesine gönderilir. "Sıralayıcı" nesnesi, "Denklem" nesnesini önce, "Parser" nesnesi ile ilişkilendirir.

Bu ilişkilendirmede, "Denklem" nesnesinin bünyesinde bulunan ve denklemi karakterize eden string ifade, "pars" edilerek, terimleri karakterize eden alt stringlere ayrılır ve terimler listesi oluşturulur. Bu listeler, "VectorUtility" sınıfının sağladığı ve yukarıda açıklanan servisler aracılığı ile sıralama işlemine tabi tutulur. Bu işlem, öz yineleme yapısına sahiptir.

Daha sonra, "Sıralayıcı" nesnesi, "Denklem" nesnesini, "SabitBulucu" ile ilişkilendirir. Bu safhada, denklem ifadesindeki, alfabetik sabitler ve konumları

belirlenir. Daha sonra, bu sabitler, "PostSıralayıcı" sınıfının servisleriyle, alfabenin başında başlayacak şekilde, yeniden isimlendirilir.

Böylece, denklem ifadesi, veritabanının anlayabileceği bir duruma gelmiş olur.

8.3.1.3. Veritabanı Modülü

Sistem, tamamıyla Internet ortamında çalışıyor olup, veritabanı, Microsoft Internet Information Server (IIS) ın kendine özgü dinamik web tasarımı olanakları ile tasarlanmıştır.

Dinamik web sayfaları sayesinde, kullanıcılar, bir web sitesindeki "CGI(Common Gateway Interface)" modüllerini, veritabanlarını ve diğer programları web sayfalarındaki arayüz yardımıyla kullanabilir. Bu arayüzler, bir "web formu" ya da bir "java appleti" olabilir. Kullanıcı, arayüz yardımıyla bir veri girişi yaptığında, bu veriler, "server" tarafından işleme tabi tutulup, web sayfasının tasarımına uygun bir şekilde, kullanıcıya sunulur.

IIS de, benzer yapıyı kullanarak, "server"da bulunan bir veritabanı üzerinde işlem yapabilir. Bu işlem, IIS ile Microsoft firmasının, standart veritabanı arayüzü olan, "ODBC(Open Database Conectivity)" ile bağlantı kurularak yapılır.

ODBC, herhangi bir veritabanı(Access, SQL Server, FoxPro...) ile bir "müşteri(client)" program arsında bağlantı kurarak, müşteri programın, bu veritabanına erişmesini sağlar. Müşteri program da, standart sorgulama dili olan, SQL yardımıyla veritabanı üzerinde işlem yapar.

IIS, ODBC ile bağlantı kurarak dinamik web sayfalarını oluştururken, şu yapıyı izler.

Bu işlemde iki farklı tipte dosya kullanılır. Bunlar sırasıyla ".htx" ve ".idc" dosyalarıdır. ".htx" dosyaları bilinen "HTML" dilinde yazılan, ancak, IIS in kendine özgü dil özelliklerini de içeren, bir "şablon(template)" dosyadır. Bu dosya, kullanıcıya, veritabanında yapılan işlemin sonucunu web sayfası olarak sunar. ".idc" dosyaları ise, ODBC deki "veri kaynağı (datasource)na bağlantıyı sağlar. Veri kaynağı bir veritabanını temsil eder. Her ".idc" dosyasına karşılık, bir ".htx" dosyası karşılık gelir.

Kullanıcı, bir "web formu" vasıtasıyla veri girişi yaptığında, yapılan bu veri girişi, ".idc" dosyasına gönderilir. ".idc" dosyasındaki SQL komutu yardımıyla, sonuç,

".htx" dosyasındaki, tasarıma bağlı olarak, yeni bir web sayfası formatında kullanıcıya sunulur.

Tasarlanan bu sistemde, veri giriş işlemleri, "java applet"leri yardımıyla yapılır. Veri giriş işlemlerinden sonra appletlerin bünyesinde oluşturulan veriler, dinamik web dili, "JavaScript" vasıtasıyla aynı sayfadaki "form"a aktarılır ve ".idc" dosyalarına gönderilir.

9 SONUÇ

Bu tezde birbirinden bağımsız, ancak, birbiriyle etkileşim halinde olan iki ayrı sistem geliştirilmiştir. Ayrıca, bu iki sistem, kendi bünyelerinde, yine, birbirinden bağımsız ve birbiriyle etkileşim halinde olan alt sistemler içerirler. Bu modüler zenginlik ve çeşitlilik, tezde aktarılmak istenilen bilgi miktarını zengin kılmakla birlikte, zaman yetersizliği nedeniyle, tez kapsamında olan bazı konulara değinilmemiştir.

Ancak yine de, genel sistem teorisi kapsamında, bir sistemin modelleme ve benzetiminde ya da genel bir sistem probleminin benzetim yoluyla çözümünde izlenecek süreç ayrıntılı olarak ele alınmıştır. Bunun yanında, benzetim programının, benzetim ortamında otomatik olarak üretilmesi, bilgisayar destekli yazılım mühendisliğinde önemli bir yer tutan, yapılan bir sistem tasrımına uygun program kodunun kendiliğinden üretilmesi kavramıyla paraleldir.

Diferansiyel denklemlerin, internet ortamında yer alan veritabanına, uygun hale getirilmesini sağlayan sıralama algoritması, aslında çok fonksiyonlu bir terim taanınma ve matematik model yorumlama algoritmalarını içeren bir çözümleyici modüldür. Bu modülde yer alan matematik model yorumlama algoritmaları, eksisiz olarak çalışmakla birlikte, terim tanıma algoritmalarında, ortak paranteze alma ve aynı terimlerin toplanıp, terim sayısını katsayı olarak atama (a+a+a=3*a) fonsiyonları yer almaz. Bunun dışında bu algoritmalar da eksisiz olarak çalışmaktadır.

Veritabanında birinci, ikinici ve üçüncü mertebeden lineer ve nonlineer diferansiyel denklemler ve çözümleri yer alır. Sayısal entegrasyon algoritması birinci mertebe

denklemler için geeçerlidir. Daha yüksek mertebeden denklemleri, birinci mertebeden denkleme indirgeme algoritmaları sisteme henüz ilave edilmemiştir.

Veritabanının daha etkin bir hale getirilebilir. Şöyle ki, homojen olmayan diferansiyel denklemlerde, denklemin sağ tarafında kalan kuvvet fonksiyonu seçimlik hale getirilebilir. Bu sayede, homejen kısmı aynı olan diferansiyel denklemlerin, veritabanınıda, aynı tabloda birden fazla yer alması gerekmez. Ayrıca, sembolik türev ve entegrasyon algoritmaları sisteme ilave edilerek, diferansiyel denklemlerin çözümleriyle ilişkisi kontrol edilebilir. Böylece, kendi kendini kontrol edebilen güvenli bir sistem tasarlanmış olur. Bunun yanında, arayüz daha etkin hale getirilebilir.

Sistemin internet ortamı ile bağlantı modülü tasarlanmış ancak, yine zaman yetersizliğinden dolayı, aktif hale getirilmemiştir. Buna ek olarak, sıralama algoritması ve diğer modüller, gözden geçirilerek, daha etkin bir hale getirilebilir.

Bütün bunların yanında, tezin asıl amacının bazı kavramları dile getirmek olduğu unutulmamalıdır.

KAYNAKLAR

[1] **Ahluwalia, S. D., Callegari, J. A., ve Reiss, L. E., (1976),** Ordinary Differential Equations with Aplications, The Maple Press Company, USA.

[2] **Balintfy, J. L., Burdick, D.S., Chu, K. ve Naylor, T.H., (1966),** Computer Simulation Techniques, John Wiley ve Sons, Inc., New York, London, Sydney.

[3] **C. J. Brennan, ve Koskossidis, D. A, (1984),** A Rreview of Simulation Techniques ve Modelling, Proceedings of the 1984 Summer Computer Simulation Conference (W. D.Wade, ed.), Vol. 1, Boston, pp. 55-8.

[4] **Bronson, R.** Schaum's Outline of Modern Introductory Differential Equations width Laplace Transforms – Numerical Methods – Matrix Methods – Eigenvalue Problems

[5] **Durling, A,. (1974),** Computational Techniques, Intext Educational Publishers, 257 Park Avenue South New York, 10010

[6] **Fogiel, M., (1978),** The Differential Equations Problem Solver, Research and Education Association, 505 Eighth Avenue, New York, N. Y. 10018

[7] **Karba, R., Matko, D. ve Zupançiç, B., (1992),** Simulation ve Modelling of Continuous Systems, A Case Study Approach, Prentice Hall International (UK) Ltd, Campus 400, Maylves Avenue Hemel Hempstead Hertfordshire, HP2 7EZ,.

[8] **Ladd , S.C ., (1998),** Java Algorithms

[9] **Neelamkavil F,. (1987),** Computer Simulation Modelling, John Wiley. NY

[10] **Neil, K. (1995),** A thesis submitted to The School of Graduate Studies Research of The University of Ottawa in partial fulfillment of the requirements of Master of Science in System Science

[11] **Ören,T.İ.. ve Klir, G.J., (1994),** Computer Aided Systems Theory - CAST'94, 4th İnternational Workshop Ottawa, Ontario, Canada, May 1994, Selected Papers

[12] **Ören,T.İ.. (1984),** GEST - A modeling and simulation language based on system theoretic concepts, Computer Science Department, University of Ottawa, Canada, NATO ASI Series, ,Vol F10, Simulation and Model-Based Methodologies : An İntegrative View

[13] **Ören,T.İ..** Dynamic and smantic rules for smulation advisers and certifiers, Computer Science Department, University of Ottawa, Canada.

[14] **Ören,T.İ..** Computer Aided Modelling Systems , Computer Science Department, University of Ottawa, Canada.

[15] **Ören,T.İ..** Computer Aided Systems Thecnology : Its Role in Advanced Computerization, Computer Science Department, University of Ottawa, Canada.

[16] **Ören,T.İ..** Implememtation of Synchronous Variables and Another Parallel Computation Feature of GEST, Computer Science Department, University of Ottawa, Canada.

[17] **Ören,T.İ.., Birta, L.G., Abou-Rabia, O.** Reqirements for the Documentation of Simulation Studies, (1989), Simulation Research Group, Computer Science Department, University of Ottawa, Canada, Bell Canada, Breau 226 Quality Engineering Desing Integrity 2265, bouled, Roland-Therrien Longueuil Quebec J4N 1C5

[18] **Ören,T.İ.., (1987),** Simulation Methodology: Top-Down Approach In: Systems and Conrol Encyclopedia, M.G. Singh (ed) Pergamon Press, Oxford, England, pp.4319-4323

[19] **Ören,T.İ.., (1987),** Simulation: Taxonomy In: Systems and Conrol Encyclopedia, M.G. Singh (ed) Pergamon Press, Oxford, England, pp.4411-4414, (Chinese translation Acta appeard in: Simulata Systematica Sinica, 1989, 1, 60-63)

[20] **Ören,T.İ.., (1987),** Simulation Models: Taxonomy In: Systems and Conrol Encyclopedia, M.G. Singh (ed) Pergamon Press, Oxford, England, pp.4381-4388, (Chinese translation Acta appeard in: Simulata Systematica Sinica, 1990, 1, 34-43)

[21] **Ören,T.İ.., (1987),** Simulation Models Symbolic Proccessing: Taxonomy In: Systems and Conrol Encyclopedia, M.G. Singh (ed) Pergamon Press, Oxford, England, pp.4377-481, (Chinese translation Acta appeard in: Simulata Systematica Sinica, 1990, 4, 59-62)

[22] **Ören,T.İ.., (1987),** Model Behavior: Type, Taxonomy, Generation and Proccessing: In: Systems and Conrol Encyclopedia, M.G. Singh (ed) Pergamon Press, Oxford, England, pp.3030-3035, (Chinese translation Acta appeard in: Simulata Systematica Sinica, 1990, 2, 53-59)

[23] **Ören,T.İ.., (1987),** Simulation and Model-Oriented Languages In: Systems and Conrol Encyclopedia, M.G. Singh (ed) Pergamon Press, Oxford, England, pp.4303-4306 (Chinese translation Acta appeard in: Simulata Systematica Sinica, 1990, 3, 59-62)

[24] **Ören,T.İ.., (1990),** Simulation Methodology: Top-Down Approach In: Consise Encyclopedia of Information Processing in Systems and Oeganizations, A. P. Sage (ed), Pergamon Press, Oxford, England, pp.421-425

[25] **Ören,T.İ.., (1993),** Simulation Languages : Taxonomy In: Consise Encyclopedia of Information Processing in Systems and Oeganizations, A. P. Sage (ed), Pergamon Press, Oxford, England, pp.306-312

[26] **Pichler, F., Schwartzel, H, (1992),** CAST Methods in Modelling, Computer Aided Systems Theory for the Design of Intelligent Machines, Contributors : Mittelmann, R., Müller-Wipperfürth, T., Praehofer, H.

[27] **Ross, S. L., (1984),** Differential Equations, Hamilton Printing Company, USA,.

[28] **Stroud, K.A., (1989),** Further Engineering Mathematics, Pogrammes and Problems, Sole distributors in the USA and its dependencies Springer-Verlag New York Inc. 175 Fifth Avenue, New York NY 10010 USA

[29] **Tam, K.M. (1997),** ODEASY: A Query Tool for Ordinary Differential Equations, A Master Thesis Submitted to The University of Ottawa in Partial Fulfillment of the Requirements for Degree of Master of Computer Science

[30] **Zeigler, B. P. (1990),** Simulation: Model-Base Organization and Utilization In: Consise Encyclopedia of Information Processing in Systems and Oeganizations, A. P. Sage (ed), Pergamon Press, Oxford, England, pp.425-429

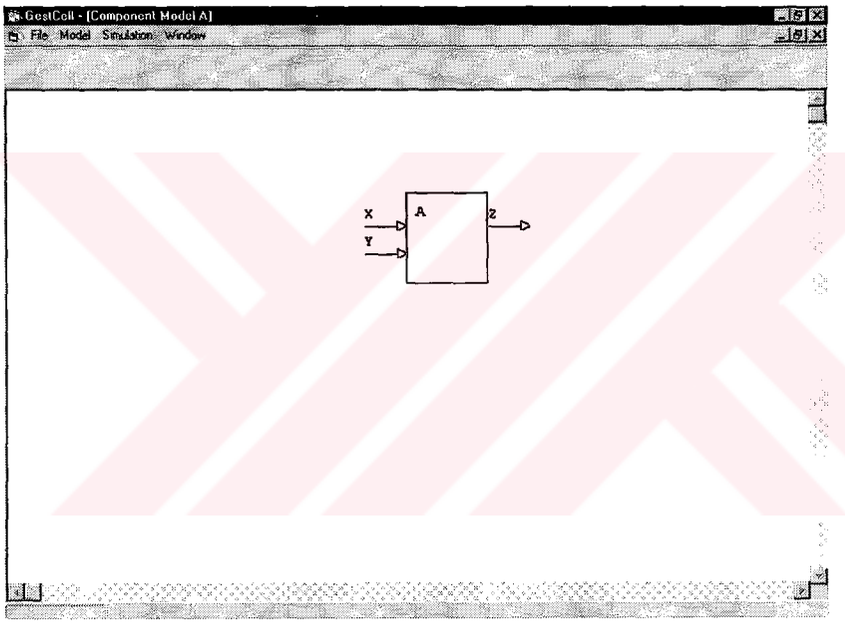
[31] **Zeigler, B. P. (1990),** Simulation Methodology and Model Manipulation In: Consise Encyclopedia of Information Processing in Systems and Oeganizations, A. P. Sge (ed), Pergamon Press, Oxford, England, pp.419-421 14

EKLER

EK A

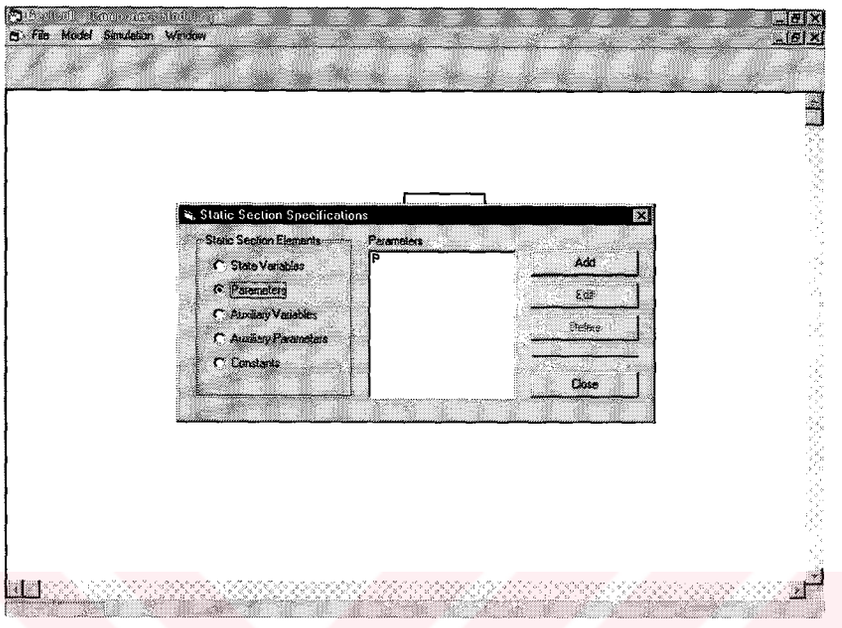
GestCell MODELLEME VE BENZETİM ORTAMININ KULLANICI ARAYÜZÜ ÖRNEKLERİ

1 Bir Modelin Giriş ve Çıkışları



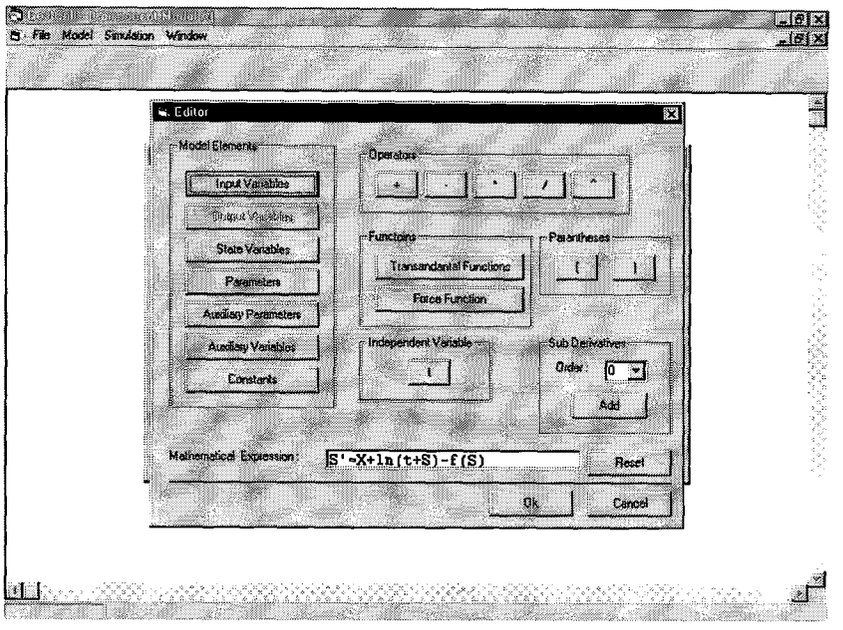
Şekil A.1 Bir modelin giriş ve çıkışları

2. Modelin İç Yapısının Tanımlanması



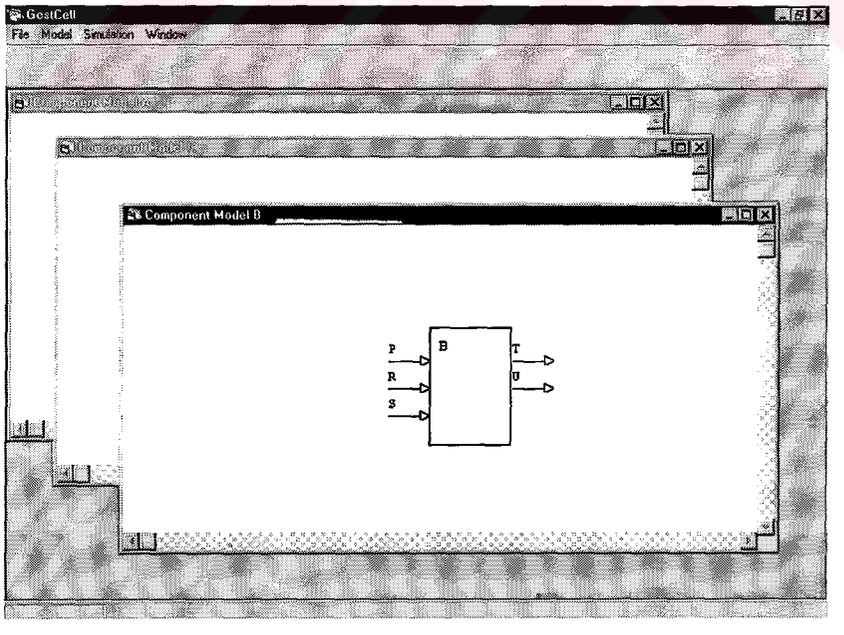
Şekil A.2 Modelin statik yapısını oluşturan elementler

Şekil A.3 Durum geçiş fonksiyonlarının tanımlaması



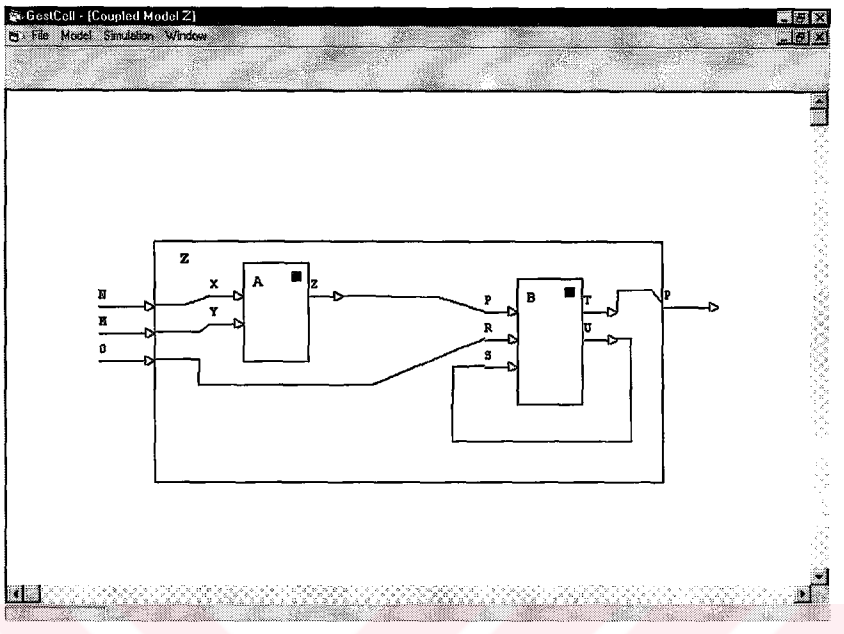
Şekil A.4 Dokunmatik denklem editörü

3. Model Nesnelerinin Grafik Arayüzde Temsil Edilmesi



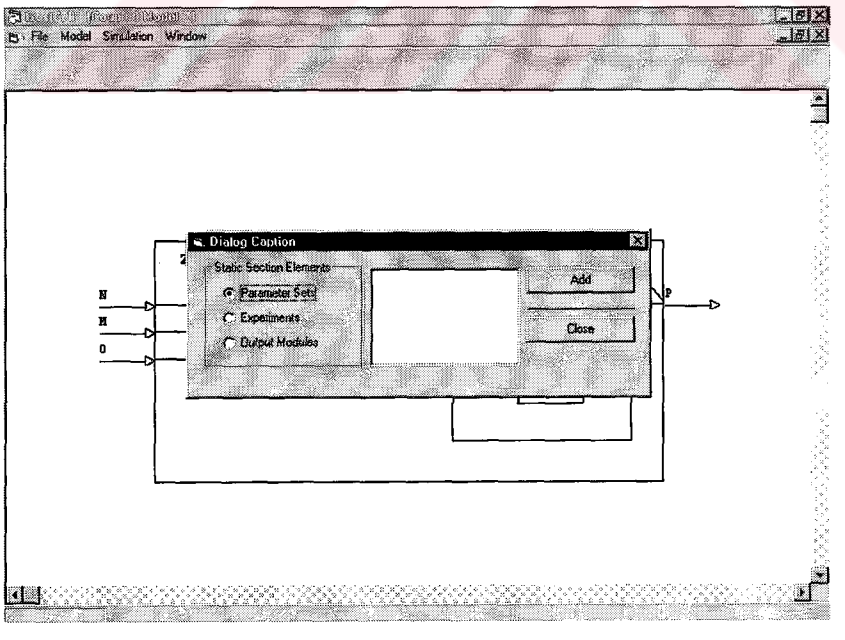
Şekil A.5 Visual Basic form penceresi ile modelin temsil edilmesi

4 Bağlaşık Model



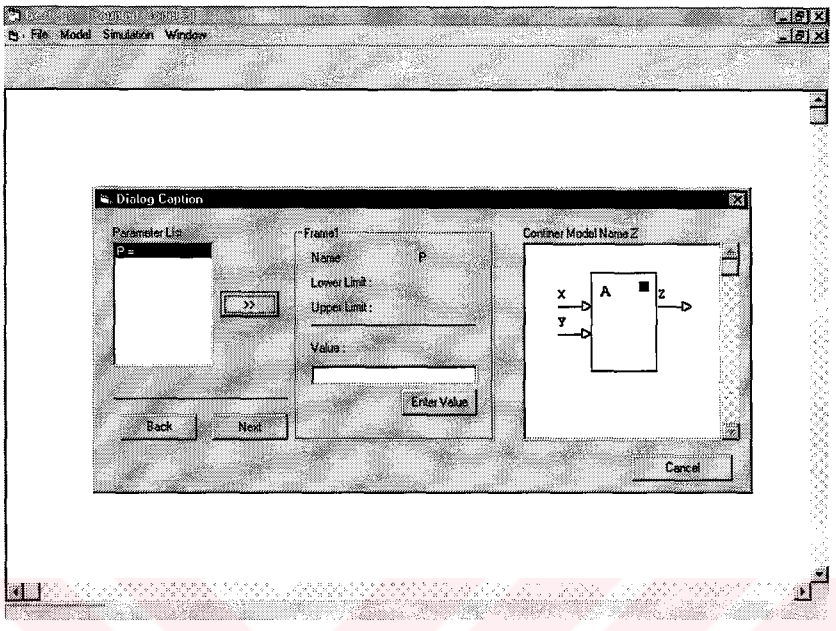
Şekil A.6 Bağlaşık model

5 Benzetim Tanımlamaları

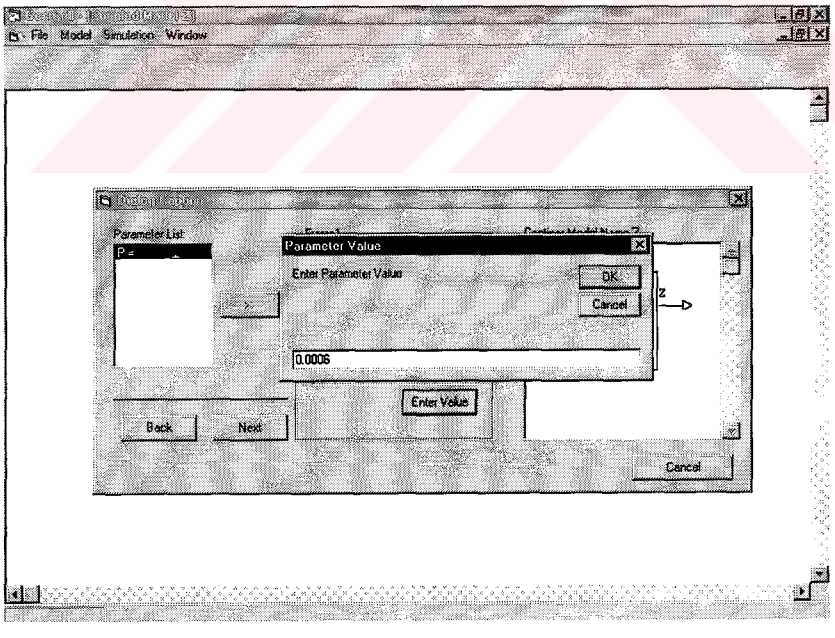


Şekil A.7 Benzetim tanımlama diyalog penceresi

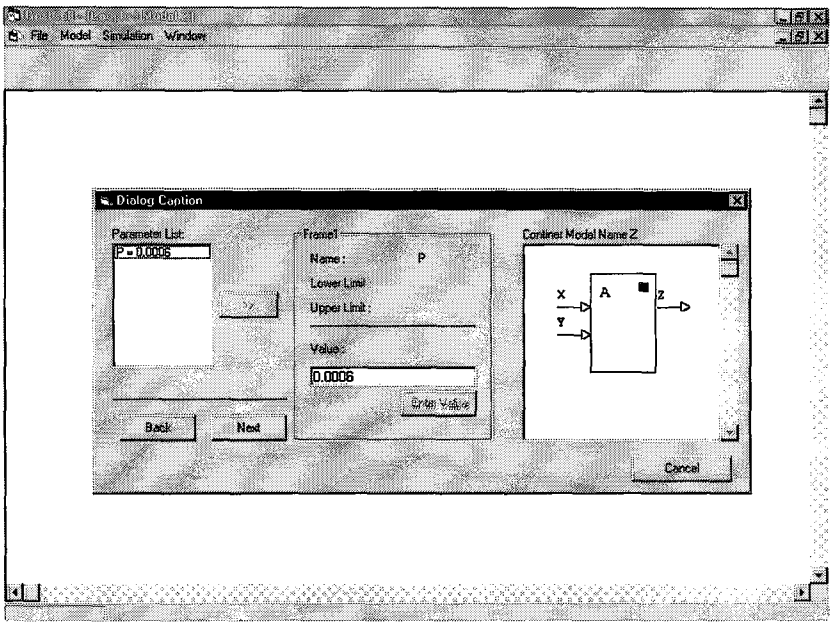
6 Parametre Seti Tanımlama



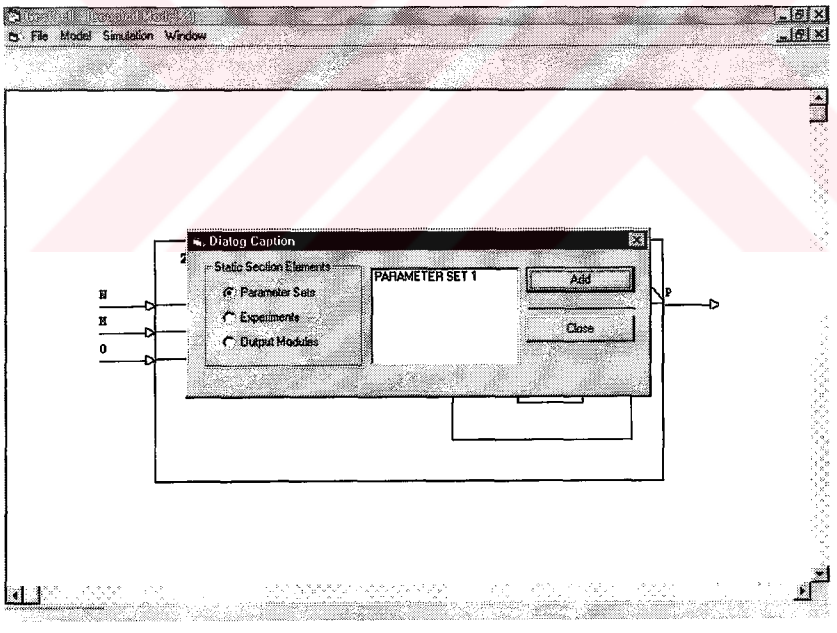
Şekil A.8 Parametre seti tanımlama diyalog penceresi



Şekil A.9 Parametre değerlerinin girilmesi

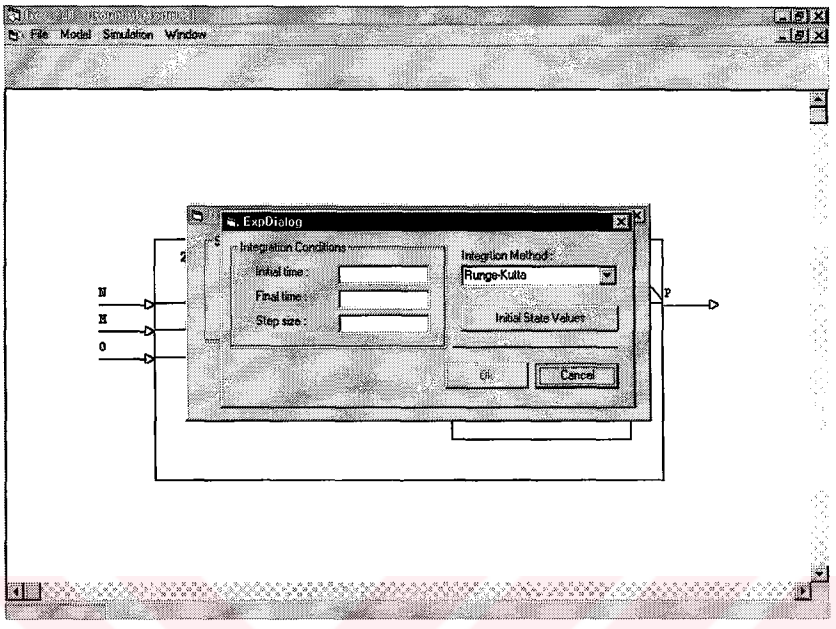


Şekil A.10 Girilen parametre değerinin listede görüntülenmesi

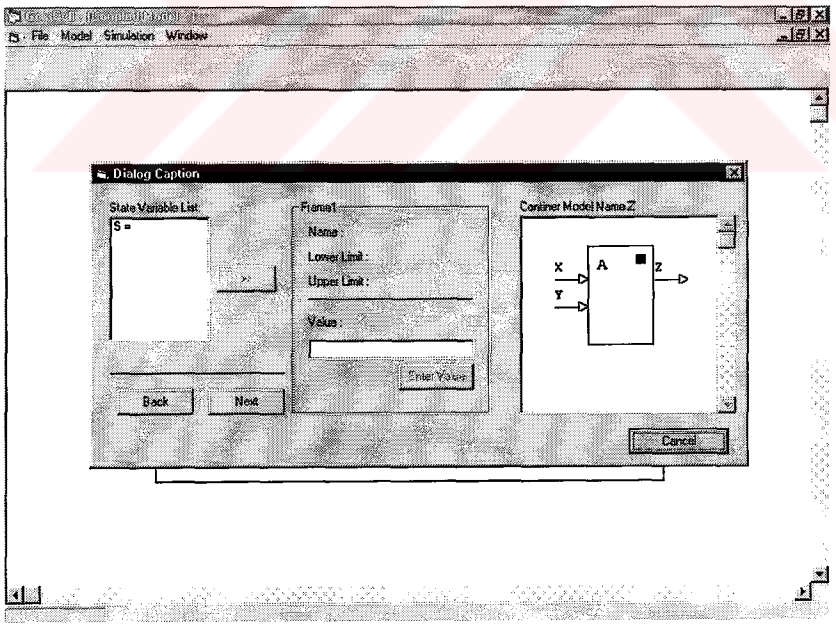


Şekil A.11 Parametre setinin parametre listesine eklenmesi

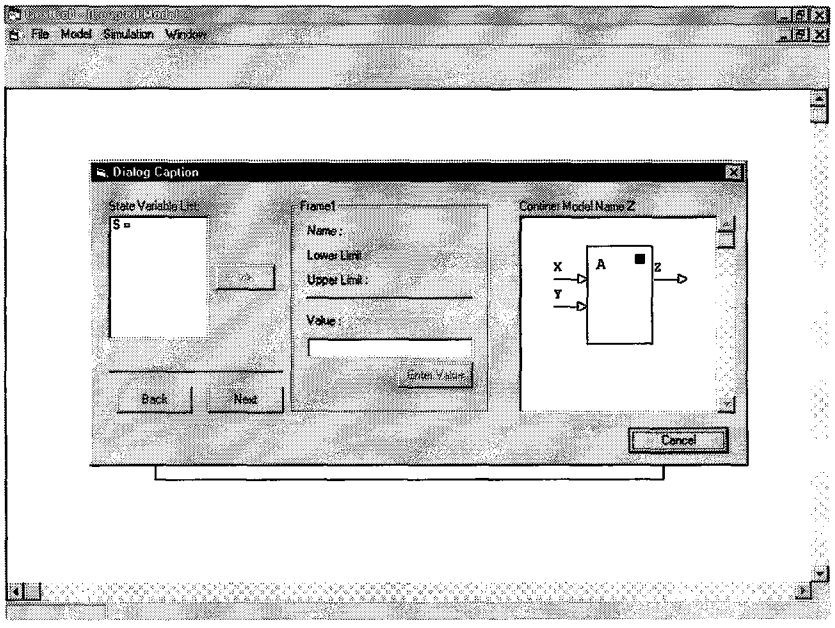
7 Deney Tanımlama



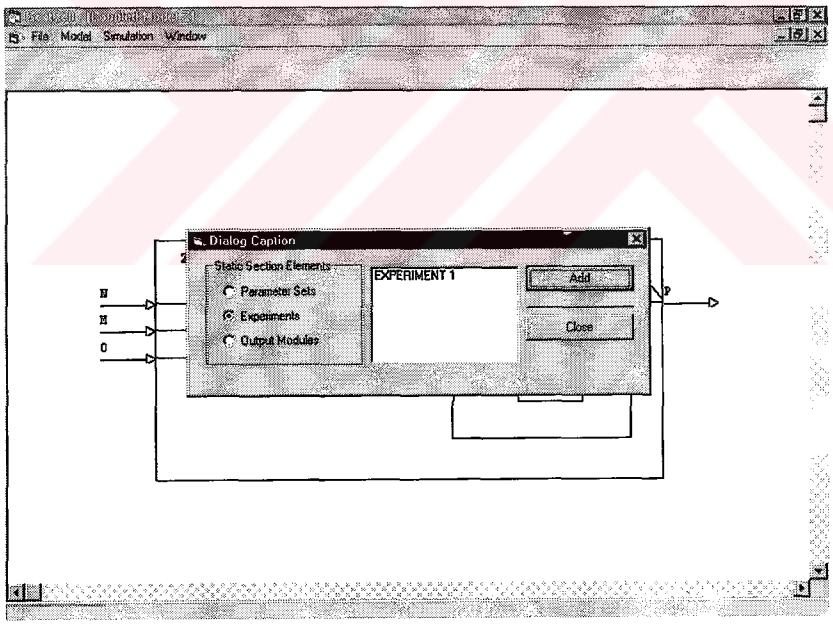
Şekil A.12 Deney tanımlama diyalog penceresi



Şekil A.13 Durum değişkenlerinin ilk değer tanımlamaları

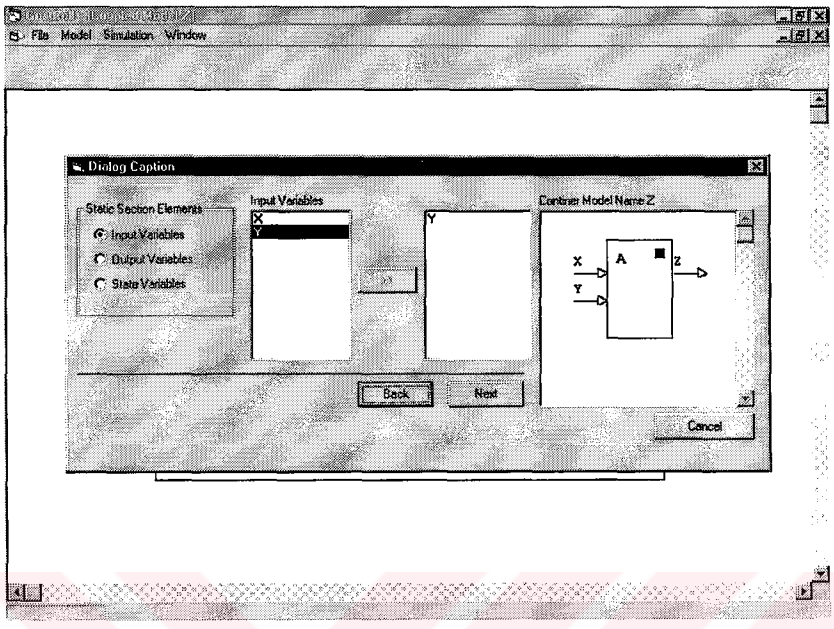


Şekil A.14 Girilen durum değişkenlerinin ilk değerinin listede görüntülenmesi



Şekil A.15 Deneyin deney listesine eklenmesi

8 Çıkış Ortamı Tanımlama



Şekil A.16 Çıkış modülü tanımlama diyalog penceresi

ÖZGEÇMİŞ

13.06.1971 yılında İzmit'te doğdu. İlk ve orta öğrenimini İzmit'te tamamladıktan sonra 1991 yılında İstanbul Teknik Üniversitesi Matematik Mühendisliği Bölümü'ne girdi. 1995-1996 öğretim yılında mezun olduktan sonra, 1996 yılında İstanbul Teknik Üniversitesi Mühendislik Bilimleri Anabilim Dalı Sistem Analizi programına yüksek lisans eğitimi için başladı. Halen TUBİTAK-Marmara Araştırma Merkezi'nde araştırmacı olarak görev yapmaktadır.