T.C.
MALTEPE ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

UZMAN SİSTEMLERİN YASAL DÜZENLEMELERE UYGULANARAK AKILLI VERİ TABANLARININ GELİŞTİRİLMESİ

HİKMET TOSYALI

Yüksek Lisans Tezi

Tez Danışmanı

PROF. DR. E.MURAT ESİN

İSTANBUL – 2008

T.C.

MALTEPE ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

UZMAN SİSTEMLERİN YASAL DÜZENLEMELERE UYGULANARAK AKILLI VERİ TABANLARININ GELİŞTİRİLMESİ

HİKMET TOSYALI

Yüksek Lisans Tezi

Tez Danışmanı

PROF. DR. E.MURAT ESİN

İSTANBUL – 2008

Bu tez çalışması, Maltepe Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ..... / ..... / ..... tarih ve ..... / ..... sayılı kararıyla oluşturulan jüri tarafından ***Bilgisayar Mühendisliği Yüksek Lisansı Tezi*** olarak kabul edilmiştir.

JÜRİ

Prof. Dr. E.Murat ESİN
Danışman

Prof. Dr. Kemal KÖYMEN
Üye

Prof. Dr. Mesut RAZBONYALI
Üye

ÖZET

Günümüzde teknoloji alanında oldukça hızlı gelişmeler yaşanmaktadır. Tüm bu hızlı gelişmelere rağmen teknolojinin tam anlamıyla cevap veremediği birçok soru mevcuttur. Bunlardan birisi de "İnsan zekâsı nasıl çalışır ve makineler insanlar gibi düşünüp karar verebilirler mi ?" Cevabı evet ya da hayır olarak net bir şekilde verilemeyecek olan bu sorudan hareketle bir bilgisayar bilim dalı olan **"Yapay Zekâ (Artificial Intelligence)"** doğmuş ve gelişimini sürdürmektedir.

Yapay zekâyı basit anlamda insanı taklit etmeye çalışan makineler olarak da açıklayabiliriz. Günümüzde, yapay zekâ'nın alt bileşenlerinden birisi olan uzman sistemler, insanın yerini almaya en yakın aday sistemlerdir. Uzman sistemler, yapay zekâ'nın bir dalı olup, bir probleme uzman insan düzeyinde bir çözüm bulmak için uzman bilgisini kullanırlar. Uzman sistemler kısaca kural tabanlı bir sistem olarak nitelendirilebilirler.

Uzman sistem yazılımlarıyla ilgili olarak Türkiye ve Dünya' da yoğun çalışmalar yapılmakta, hemen her alanda uzman sistemlerden yararlanılmaktadır. Özellikle kurumların bağlı oldukları yasal düzenlemeleri oluşturan çok sayıda hükmün bir arada değerlendirilmesini gerektiren ihtiyaçların karşılanması amacıyla kural tabanlı uzman sistemler tasarlanmaktadır.

Bu çalışmada yapay zekâ tekniklerinden uzman sistemler tanıtılarak, yasal düzenlemelere bağlı olarak geliştirilmiş veri tabanlarının sorgulanmasında alışılmış yöntemlerin kısıtlarına bir çözüm olarak uzman sistemlerin kullanılması ve bu amaçla hazırlanmış dillerden Prolog ile yazılmış bir uzman sistemin çalışma mantığı araştırılmıştır. Ayrıca Prolog'un MS-SQLServer, ORACLE gibi harici veri tabanlarıyla bağdaştırılarak, uzman sistemlerin ilişkisel veritabanlarıyla birlikte kullanımına ilişkin örnekler verilmiştir.

Anahtar Kelimeler: Uzman Sistemler, Yapay Zekâ, Visual Prolog, Veri tabanları.

ABSTRACT

Today, there is a rapid growth in the technological areas. Despite of this rapid growth, there are lots of questions which technology cannot answer. One of them is how to operate human mind and whether machines think and decide just like human being or not. By starting this question whose answer cannot be known well, "Artificial Intelligence", a computer science branch, was born and it has been still developed.

We can explain the artificial intelligence that is the machine which imitates human mind. The only base of these machines can be human mind because other minds which are not based on human being, has not been found.

Today the expert system, one of the subset of the artificial intelligence, is the most imminent nominee system that can supersede the human being. According to the general definition ,the expert system is an adviser programme which imitates the process of the expert knowledge and comprehension in the solving of some problems.

The expert systems, one of the branch of the artificial intelligence, aim to use the expert knowledge in order to find a solution at the level of expert person. It can be characterized that the expert systems are rule-based systems. The rules are formed by the experience of the experts. From these rules, depending on reason-result relation of human being, it draws a conclusion at the end of these logical process.

In Turkiye and in the world, related to the expert systems softwares, there are lots of mass study and the expert systems are being used in every fields. The expert systems especially are designed to provide the needs that evaluate the commands together formed the legal regulations of institutions.

In this study, by introducing the expert system, an artificial intelligence tecnique, the usage of the expert system as a solution for the restriction of the conventional

methods in the database query and the workings of the expert system written in prolog have been researched.

Key Words: Expert systems, Artificial Intelligence, Visual Prolog, Databases.

TEŞEKKÜR

Yüksek lisans tezimi hazırlamamda bana yardım eden, zaman ayıran ve her koşulda bana destek olan tez danışmanım, T.C Maltepe Üniversitesi Elektrik-Elektronik Bölüm Başkanı hocam Prof. Dr. E. Murat ESİN' e, yardımlarından, ilgisinden, alakasından, sonsuz hoşgörüsünden ve desteğinden dolayı teşekkürlerimi sunarım.

İÇİNDEKİLER

ŞEKİLLER

TABLOLAR

1 GİRİŞ

Yapay Zekâ; insan zekâsıyla sonuca ulaşılabilecek problemlerin ve durumların, makineler veya bilgisayar yazılımlarıyla yapılmasını sağlayacak yöntemlerin geliştirilmesi konusunda çalışan bir bilgisayar bilim dalıdır. Başka bir tanıma göre Yapay Zekâ; düşünme, anlama, kavrama, yorumlama ve öğrenme yapılarının programlamayla taklit edilerek problemlerin çözümüne uygulanmasıdır [3]. Diğer bir tanıma göre Yapay Zekâ; "insanlar tarafından yapıldığında zekâ gerektiren şeyleri gerçekleştiren makineler yapma bilimidir." (Minsky, 1968).

Tüm tanımlarda görüldüğü gibi yapay zekânın temel amacı en basit anlamda, makinelerin insan zekâsını taklit ederek düşünme ve karar verebilmelerini sağlamaktır. Böylece insan zekâsının ve gücünün yetersiz kaldığı durumlarda yapay zekâ teknikleri sayesinde daha sağlıklı, doğru ve hızlı bir şekilde sonuca ulaşılabilir.

1956 yılında başladığı kabul edilen yapay zekâ çalışmalarıyla, ilk ortaya çıktığı yıllardan itibaren oldukça ilgi çekici ve faydalı ürünler geliştirilmiştir. Bu çalışmalarda çeşitli yapay zekâ teknikleri kullanılmış ve farklı durumlarda sorunun çözümüne ilişkin olarak bu teknikler kullanılmıştır.

Yapay zekâ tekniklerinden birisi olan **Kural Tabanlı Uzman Sistemler (Rule Based Expert Systems)**, bilgisayar programları vasıtasıyla uzmanlık gerektiren karmaşık problemlerin çözümünü sağlamaktadır.

Kurumların bağlı oldukları yasal düzenlemelerin sıklıkla değişmesi ve bu yasal düzenlemeleri oluşturan çok sayıda hükmün bir arada değerlendirilmesini gerektiren durumlarda, uzman sistemlerden faydalanarak mevcut veri tabanının taranarak her bir üye için bağımsız ve akıllı sorgular yapılabilmesi mümkün olmaktadır. Bu çalışmayı oluşturan bölümlerde, yapay zekâ ve uzman sistem kavramları, kuralların prolog diliyle ifade edilmeleri, prolog' un ilişkisel veritabanlarıyla birlikte kullanımı ve yasal düzenlemelere dayalı bir uzman sistemin nasıl geliştirileceği anlatılacaktır.

2 ZEKÂ VE YAPAY ZEKÂ

Nesneler arasındaki ilişkileri düşünüp anlayabilme, karşılaştırma yapabilme ve bu işlevleri amaca yönelik olarak kullanabilme yeteneğine zekâ adı verilmektedir. Zekâya ilişkin kuramların tümü, zekânın geliştirilebilir olduğu ve biyolojik temellerinin olduğu noktalarında birleşirler. Buna göre zekâ, kişinin kalıtımla doğuştan edindiği ve deneyim, öğrenme ve çevre faktörleriyle biçimlenen bir bileşimdir. Latince "intellectus" kelimesinin karşılığı olan zekâyı biyologlar; çevreye uyum kabiliyeti, eğitimciler; öğrenme, psikologlar; ilişkileri anlayabilme, bilgisayarcılar bilgiyi işleme kabiliyeti olarak değerlendirmişlerdir. Farklı branşlar zekâyı farklı şekillerde tanımlasalar da hepsi zekânın nitelikleri konusunda hemfikirdirler. İster yapay olsun ister biyolojik, zeki bir sistemin taşıması gereken nitelikler şunlardır:

- İnanç, arzu, tercih gibi tavırlar sergilemelidir
- Öğrenme yeteneğine sahip olmalıdır
- Büyük sorunları daha ufak parçalara ayırarak çözümler üretebilmelidir
- Anlama, plan yapma, öngörüde bulunma gibi yeteneklere sahip olmalıdır
- Olaylar ve durumların benzer ve farklı yanlarını ayırt edebilmelidir
- Lisan ve sembolik gösterimleri anlayabilmeli ve kullanabilmelidir.

Bu nitelikleri sağlayan sistemler (yapay veya biyolojik olabilir), zeki sistemler olarak isimlendirilebilirler.

Yapay zekâ ya da İngilizcesiyle "Artificial Intelligence" , insanın düşünme yapısını anlamak ve bunun benzerini ortaya çıkaracak bilgisayar işlemlerini geliştirmeye çalışmak olarak tanımlanır. Yani programlanmış bir bilgisayarın düşünme girişimidir. Daha geniş bir tanıma göre ise, yapay zekâ, bilgi edinme, algılama,

görme, düşünme ve karar verme gibi insan zekâsına özgü kapasitelerle donatılmış bilgisayarlardır. Yapay zekâyı basit anlamda insanı taklit etmeye çalışan makineler olarak da açıklayabiliriz. Bu makinelerin zekâ bakımından tek dayanağı insan zekâsı olabilir; çünkü henüz insan zekâsına bağlı olmayan ve kanıtlanabilen bir zekâya ulaşılmamıştır [3].

Yapay zekâ tartışmaları, genel olarak birbirinin zıttı iki görüş barındırır. Görüşlerden biri yapay zekânın ilk ortaya çıktığı yıllarda ileri sürülmüştür ve düşünmenin sadece bilgiyi işlemeden ibaret olduğunu belirtmektedir. Bu görüşe katılanlar, yapay zekânın bir gün insan zekâsı seviyesine gelebileceğine inanırlar ve beynin çalışma prensipleri herhangi bir makine de tamamen sağlanırsa o makinenin da bizim gibi düşüneceğine inanırlar. İkinci görüş ise insan düşüncesinin herhangi bir yapay düşünceden çok farklı olduğunu ve insanların sezgi, akıl yürütme, olaylara anlam verme, duyarlı olma gibi yeteneklerinin hiçbir cansız varlık tarafından kazanılamayacağını savunurlar. Onlara göre düşünce kendiliğinden ve hiçbir zorlama olmadan gerçekleşen soyut bir eylemdir. Bu yüzden hiçbir bilgisayar insan gibi sevinemez, üzülemez, heyecan duyamaz; çünkü bilgisayarlar veya robotlar için bu gibi insana özgü soyut duygular sadece işlenecek verilerdir. Bu duyguların onlar için başka bir anlamı olamaz [3].

Yapay Zekâ, insanlar tarafından yapıldığında zekâ gerektiren şeyleri gerçekleştiren makineler yapma bilimidir (Minsky, 1968).

Yapay Zekâ, zeki davranışları taklit eden bilgisayar programlarının yapımı boyunca insan zekâsının doğasını anlamayı amaç edinen disiplindir (Bonnet, 1985).

2.1 Yapay Zekânın Tarihçesi

Yapay zekâ ile ilgili ilk araştırmalar, II. Dünya Savaşından sonra yapılmıştır. İngiliz matematikçi Alan Turing, 1947 yılında verdiği bir konferansta yapay zekânın en iyi bilgisayarları programlayarak araştırılması gerektiğini savunmuş ve Turing Testi' ni ortaya çıkarmıştır. Daha sonraları 1950'lerin sonunda Turing'in görüşünü haklı çıkarırcasına araştırmaların temeli olarak bilgisayar programcılığı alınmıştır. Bundan

sonraki yıllarda mantığın hâkim olduğu çalışmalar yapılmış ve programcılar yazdıkları programların başarılarını kanıtlamak için bir takım simülasyonlar yaratmışlardır. Daha sonraları bu sorunlar gerçek yaşamı hiçbir şekilde temsil etmeyen oyuncak dünyalar olmakla suçlanmış ve yapay zekânın yalnızca bu alanlarda başarılı olabileceği ve gerçek yaşamdaki sorunların çözümüne yardımcı olamayacağı ileri sürülmüştür. Bunun üzerine her sorunu çözecek genel amaçlı bilgisayar programları yerine tek bir meslek dalı üzerinde yoğunlaşan, özel programlar yazılmaya başlanmıştır. Her ne kadar bu olay durgun olan yapay zekâ dünyasına bir hareketlilik katmış olsa da bazı sorunlara yol açmıştır. Mesela; otomobili tamir etmek için öneri veren program, otomobilin işlevini tam olarak bilmemektedir [4].

60'lı yıllar yapay zekânın altın dönemini yaşadığı yıllardır. Bu yıllarda yapay zekâ konusundaki araştırmalara önde gelen üniversiteler destek vermiş, çalışmalar kurumsallaşmıştır. 1956 yılında ABD de Darmount Koleji'nde proje çalışmaları başlatılmış ve bilgiler sistemleştirilerek bu çalışmaya Amerikalı bilim adamı McCarty tarafından "Yapay Zekâ" adı verilmiştir. Bu yıllarda bilgisayarlar satranç, dama gibi çeşitli zekâ oyunlarını oynayabilecek kapasiteye getirilmiş; doku tanıma programları geliştirilmiştir [4].

70 ve 80'lerde de yapay zekâ fırtınası tüm hızıyla sürmüştür. Bu dönemde çalışmalar bir sistem üzerine oturtulmuş ve daha derin araştırmalar yapılmaya başlanmıştır. Özellikle 80'lerde büyük bilgisayar firmaları yapay zekâ konusuna büyük yatırımlar yapmışlar ve insan beynindeki sinir ağlarına benzeyen yapay sinir ağları oluşturulmuştur. Uzmanların başarıları ilk ticari uygulamaları da getirmiştir. Yapay zekâ sayesinde müşteri ihtiyaçlarına göre donanım seçimi yapan R1 adlı bir uzman sistem şirketi bir yılda 40 milyon dolar tasarruf etmiştir. Bunun sonucunda diğer ülkelerde yapay zekâyı yeniden keşfetmişler ve araştırmalara büyük kaynaklar ayrılmaya başlanmıştır. 1988 yılında yapay zekâ endüstrisi üzerinde dönen para yaklaşık 2 milyar dolara ulaşmıştır. Yapay zekâ, artık hayatın her alanındaydı [4].

90'lı yıllarda da gelişme sürmüştür; ancak üzücü olan yapay zekânın insanlığın yararı yerine savaş teknolojisi üzerine yoğunlaşması ve 1990'da ülkemizin yanı başında olan Körfez Savaşı'nda yapay zekâ uygulamalarının ciddi anlamda ilk kez askeri alanda kullanılmasıdır. 90'ların sonuna doğru IBM'in yarattığı ve "DEEP BLUE" adını verdiği saniyede yaklaşık 10 milyon hamle hesaplayabilen satranç bilgisayarı, dünyanın en büyük satranç dehası olarak kabul edilen, dünya şampiyonu Garry Kasparov' u mağlup ederek yapay zekânın bazı konularda insan zekâsından çok daha üstün olduğunu kanıtlamış oldu [4].

2.2 Yapay Zekâ Çeşitleri

Yapay zekâ konusundaki araştırmalar şu gruplar altında toplanabilir.

- Bilgi tabanlı yapay zekâ ve uzman sistemler
- Doğal diller (bilgisayar ile doğrudan iletişim)
- Beşeri algılama yeteneklerinin simülasyonu (görme, konuşma, işitme, koklama vs.) [5].

2.2.1 Bilgi Tabanlı Yapay Zekâ ve Uzman Sistemler

Bilgi tabanlı yapay zekâ sistemi, belli bir uygulama alanına (bilgisayar onarımı gibi) ilişkin pratik çözüm veya yordamlama bilgilerinden (sezgi, yargı ve çıkarımlar) oluşmuş bir bilgi tabanına dayalı olarak çalışır. İnsanların kendilerine ait bilgi tabanı sistemindeki EGER- 0 ZAMAN (IF-THEN) kurallarını kullanarak belirli soruları çözme kabiliyeti bu yapay zekâ türüne ilham kaynağı olmuştur. Bilgi tabanlı sistemlerin en gelişmiş örneği uzman sistemlerdir. Belli bir soruna ilişkin uzmanlık bilgileri bir uzman sistemin bilgi tabanına yerleştirildikten sonra, kullanıcıların bu bilgiden yararlanmak amacıyla uzman sistemle kurduğu iletişim bir uzman şahısla kurulan iletişimin bir benzeridir. Sorun çözülene kadar kullanıcı ile bilgisayar tabanlı uzman sistem arasında karşılıklı soru-cevap türünde bir iletişim oluşur.

Yıllık vergi iadesi formunun hazırlanmasında bireylere yardımcı olmak üzere hazırlanan "DAN" isimli yazılım, bilgi tabanlı sisteme güzel bir örnektir. Sistem kullanıcının veri girmesine yol gösterici olacak bir vergi iadesi formunu içermektedir. Girilen verilere bağlı olarak gerekli olan hesaplamalar sistem tarafından otomatikman yapılmaktadır. Elde edilen çıktı doğrudan resmi makamlara sunulabilecek formatta olduğundan, herhangi bir uzmanın yardımına gereksinim duyulmadan vergi iadesi formu bireylerce hazırlanabilmektedir [5].

2.2.2 Doğal Diller

Doğal diller nihai kullanıcının doğal dili ile (İngilizce gibi) bilgisayarla iletişim kurmasını sağlayan yazılımlar için kullanılan isimdir. Doğal dil yazılımlarındaki nihai amaç, geleneksel program dillerinde kullanılan komutlara olan gereksinimi ortadan kaldırmaktır. Örneğin insan kaynakları ve satın alma ile ilgili araştırma ve rapor hazırlama faaliyetlerinde doğal dil uygulaması oldukça gelişmiş düzeyde olup kullanıcı normal İngilizce konuşur gibi bilgisayar ile iletişim kurabilmektedir [5].

2.2.3 İnsan Algılama Yeteneklerinin Simülasyonu

Bu yapay zekâ türü, insanî yeteneklerin simülasyonu ile ilgili olup bilgisayar sistemlerini görme, işitme, konuşma ve hissetme (dokunma) yetenekleri ile donatma çabasındadır. İnsan algılama yeteneklerine sahip bilgisayarlar tıpkı insanlar gibi çevre ile iletişim kurma becerisine sahip olabilmektedir. [5].

2.3 Yapay Zekâ Uygulama Alanları

Yapay zekâ bugün, birçok alanda araştırmaların yapıldığı bir bilim ve teknoloji dalı olarak ortaya çıkmış bulunmaktadır. Bu alanlardan bazıları aşağıda belirtilmiştir.

2.3.1 Askeri Alanlar

Günümüzde yapay zekâ uygulamaları öncelikli olarak askeri alanlarda yeni sistemlerin geliştirilmesine, tatbikat çalışmalarına ve eğitime yönelik simülasyonların geliştirilmesinde kullanılır. Askeri araştırmalar, zeki benzetim sistemleri, üretim, lojistik, eğitim, bakım-onarım ve istihbarat edinme yapay zekânın kullanıldığı diğer bölümlerdir. Yapay zekânın askeri alanda kullanılmaya başlaması ilk olarak 1980'lerde başladı. ABD'nin bu alandaki başarıları diğer ülkelerin de dikkatini çekti ve çeşitli ülkeler bu teknolojiden yararlanmak için araştırmalar başlattı. Askeri alanda yapay zekânın görüldüğü en yakın örnek ABD'nin 1990'da Irak'a karşı kullandığı hedefe kilitlenen Patriot füzeleridir. O günden bugüne yapay zekânın gelişimiyle ve yeni araştırmalarla birlikte bu silahların ve savaş araçlarının nitelikleri inanılmaz derecede artmış ve önüne geçilemez bir hal almıştır. Eğer yapay zekâ askeri eğitim ve üretim gibi olumlu konular dışında silah üretimine yoğunlaşırsa bu teknolojik gelişim insanlık için bir felaket haline gelebilir.

2.3.2 Oyun Teorisi

Yapay zekânın belki de çağımızda en popüler olan ve en çok ilgi gören kullanım alanı bilgisayar oyunlarıdır. İlk bilgisayar oyunları yapılmaya başlanıldığından beri, yapımcılar sürekli bize karşımızdakinin bir makine değil de bir insan olduğu hissini vermek istemişler ve yıllardan bu yana bu konuda büyük aşamalar kaydetmişlerdir. Aslında oyunlarda kullanılan yapay zekâ, bilgisayarın düşünebilmesine yönelik değildir. Sadece bilgisayar, bizim yaptığımız davranışlara göre kendisine oyun programcıları tarafından önceden yüklenmiş kurallara göre hareket eder. Bilgisayar, kesinlikle kendisine önceden girilmemiş bir davranışta bulunmaz. Her şey insan zekâsı tarafından planlanmış ve uygulanmıştır.

2.3.3 Robotlar

Yapay zekâ uygulamaları üzerinde çalışan araştırmacılar, öğrenme yeteneğine sahip olan ve iletişim kurabilen makineler icat etme üzerinde çalışmaktadır. İnsana yakın davranış sergileyen ve bilgisayar tarafından yönetilen makinelere robot adı verilir. Günümüzde robotlar yeteri kadar olmasa da birçok alanda kullanılmaktadır. Örneğin; araba boyama, vida sıkma, ağır yüklerin taşınmasında, tehlikeli işlerin yapılmasında, malzeme taşıma ve hatta kusurlu parçaları tespit etme gibi daha karmaşık davranışları yapan robotikleri günümüzde görmek mümkündür. [6].

Günümüzde otomotiv sektörü robotlardan en fazla yararlanan sektördür. Bu sektörde robotlar daha çok boyama ve montaj işlemlerinde kullanılmaktadır. Elektronik sektörü bu konuda ikinciliği tutmaktadır. Elektronik devrelerin testi ve yongaların yerleştirilmesi işlemlerinde robotlardan istifade edilmektedir. Bugün, artık cerrahi de bile robotlar kullanılabilmektedir. Örneğin, bir beyin cerrahına yardımcı olan robotları hastanelerde görmek mümkündür. Robotlar büyük bir doğruluk yüzdesi ile biyopsi yapabilmekte ve böylece ameliyatın daha hızlı, daha doğru ve daha güvenli yapılmasını sağlamaktadır.

Robotlara işin nasıl yapılacağı bilgisayar tarafından öğretilir. Bir bilgisayar programı ile robotları kontrol etmek mümkündür. Bu program robota hareketin zamanı, yönü, mesafesi gibi konularda komut veren bir programdır. Bir kere programlandıktan sonra, robotların hareketlerini kontrol etmeye fazla ihtiyaç yoktur [6].

2.3.4 Uzman Bilgisayar Programları

Yapay zekânın bir başka kullanım alanı da konusunda uzman olan, işleyiş ve düzen hakkında önerilerde bulunan programlardır. Bu programlar, otomotiv sanayisinde, muhasebe kayıtlarında ve hatta tıp alanında kullanılır. Bu programların genel çalışma düzeni şöyledir: Programa ürünün veya konunun özellikleri girilir ve program kendisine önceden kaydedilmiş verilerden yararlanarak en uygun seçimi ortaya

çıkarır ve kullanıcıya sunar. Bu gibi programlar üretimde hem zamandan hem de paradan kazandırır. Bu sebepten dünyanın önde gelen yazılım şirketleri yapay zekânın hâkim olduğu yazılımlar hazırlamakta ve satışa sunmaktadır.

2.3.5 Sanal Gerçeklik Uygulamaları

Sanal gerçeklik bilgisayar ortamında oluşturulan bir gerçekliktir ve orijinal ismi "cyberspace"dir. Sanal gerçeklik ortamında gözlük, kulaklık, eldiven gibi çeşitli araç ve gereçler kullanılır ve insanın kendisini sanal, üç boyutlu dünyayı gerçekmiş gibi hissetmesi amaç edilir. Bu ortamda aslında var olmayan nesnelere dokunabilir, onları görebilirsiniz. Sanal gerçeklik, sizin bilgisayar kaynaklı her türlü objeyle etki içinde olmanızı sağlar. Sanal gerçeklik uygulamalarının oldukça geniş bir kullanım alanı vardır. Bu uygulamalardan hastalıkların teşhis ve tedavisinde, hastanın modelinin oluşturulup incelenmesinde, bilgisayarla yapılan üç boyutlu tasarımlarda, bazı videolarda ve eğitim amaçlı simülatörlerde yararlanılır. Örnek vermek gerekirse 1998 Dünya Kupasının final maçının yapıldığı stadyum olan Stad de France, IBM Fransa tarafından yapımından önce sanal olarak inşa edilmiştir. Bunun amacı stadı henüz plan aşamasındayken, stada gelecek insan davranışlarını incelemek ve ona göre davranmaktır. Sanal gerçekliğin kullanımı bu uygulamada en üst düzeydedir.

3 UZMAN SİSTEMLER

Yapay zekânın en önemli uygulama alanlarından birisi olan uzman sistemler, belirli bir uzmanlık alanındaki önemli problemleri çözmek için düşünen uzmanı taklit eden kural tabanlı bir sistemdir (Jackson, 1986).

İngiliz Bilgisayar Birliği Uzman Sistem Grubu; uzman sistemleri; uzman bir kişinin becerilerinden oluşan bilgiyle donatılmış bir bilgisayarın içindeki öyle bir yapıdır ki, sistem akıllıca önerilerde bulunabilir veya bir işlemin işlevleri hakkında kararlar verebilir şeklinde tanımlamıştır [8].

Uzman Sistem, çözümleri için önemli ölçüde insan uzmanlığı gerektiren karmaşık problemleri çözmek için bilgi, mantıksal çıkarım prosedürleri kullanan akıllı bir bilgisayar programıdır (Feigenbaum, 1982).

Uzman sistemlerle ilgili yapılan diğer bazı tanımlar şöyledir:

“Gerçekleşmekte olan bir olay yada durum hakkında zeki kararlar alan veya zeki öneriler teklif edebilen sistemlerin düzenlenmesi gibi, uzmanların yetenekleri sayesinde bilgi tabanlı elemanların bilgisayar içinde düzenlenmesidir” [9].

“İnsan bilgisi ve tecrübelerine dayalı olan davranışların bir bilgisayar ortamına aktarılarak tasarlanmış sistemlerdeki karşılaşılan problemlere uzman bir kişinin gereksinimi olmaksızın çözümler arayan bilgi tabanlı sistemlerdir” [10].

Bir uzman sistemin en belirleyici özelliği, oldukça büyük bir bilgi tabanına sahip olmasıdır. Bu konuda dikkat edilmesi gereken nokta ise; değişme ve gelişmeye açık olması gereken bilgi tabanı bölümü ile mümkün olduğunca statik olması gereken program bölümünün birbirinden ayrılmasıdır [11].

Uzman bir sistemin bilgisi gerçekler ve sezgisel bilgiden oluşur. Gerçekler; genel kabul görmüş ve söz konusu alandaki uzmanların üzerinde mutabık oldukları bilgi

setinden oluşur. Sezgisel bilgi ise; daha çok uygulamayı yapan kişi özelinde olup, iyi bir kararın göreceli olarak az tartışılan kuralları; akıl yürütme yeteneği, sorgulama kuralları gibi söz konusu alandaki uzmanlardan elde edilen bilgi setini karakterize eder [11].

Uzman sistemler bilgisayar temelli sistemler olup konusunda uzman olan insanların çözebildikleri karmaşık problemlere çözüm sağlamaktadırlar. İnsan uzmanların yaptıkları gibi belirli bir alana yönelik çözümler getiren bilgisayar yazılımlarıdır. Bu yazılımlarla insanların düşünme işlemlerini taklit edilerek onların yerlerine geçecek olan sistemler kurulmaya çalışılmaktadır.

Uzman sistem tasarımcılarının bu sistemleri geliştirmede göz önünde bulundurdukları esas ilkeler arasında; o an için mevcut olmayan bir uzmanın yerini alabilmesi, birçok insan uzmanın bilgi ve tecrübe birikimini bir araya getirebilmesi, yeni uzman adaylarını eğitebilmesi, uzmanları ilgilendirmeyen veya onlara cazip gelmeyen veya uzmanların pahalı olduğu projeler için gerekli uzmanlığı sağlamak gelmektedir.

Uzman sistemler insanlardan daha çok bilgi depolayıp bunları daha hızlı işleyebilirler. Bu durum uzman sistemlerin insanların zorlanabileceği durumlarda kolaylıkla sonuca gidebileceklerini göstermektedir.

Schalcof "Yapay Zekâ: Bir Mühendislik Yaklaşımı (Artifical Intelligence an Engineering Approach)" adlı kitabında şunu ileri sürmektedir: " Uzman sistemler genellikle özel bir alanda uzmanların davranışlarını göstermeye çalışan programlardır."

Jakson'a göre uzman sistem; "Özel bir konu üzerine problemleri çözmek veya önerilerde bulunmak için gerekli bilgileri bulunduran bir bilgisayar programıdır''.

Her iki çalışmada sistemlerin özel ve dar bir alana daha uygun oldukları konusunda hemfikirdirler. Uzman sistem' ler genelde uzmanların yerini tutar gibi görünürler, bu bazı durumlar için doğru olabilir, öyle ki bazı uzmanlara ait bilgilerin organizasyon

içinde rahatça kullanımına imkan verirler. Bazı kararlar bir uzman gibi her zaman ve her yerde sabit bir kaliteyle verilebilir. Uzman deneme yanılmaya dayalı kararlara bağımlı olup, keşiflere zamanı yoktur. Sıklıkla Uzman sistem, uzman için bir akıllı yardımcı olarak nitelendirilebilir, büyük bir görev içinde özel bir sahada yer alır ve kararların kısa zamanda alınmasına ve kalitesinin iyileştirilmesinde kullanılır. Ancak Uzman sistem' lerin içerdikleri bilgiler kesin ve tamamlanmış değillerdir. Bu bilgiler ancak elde edilebilir bilgilerden oluşurlar. Uzman sistemler ana kurallarla inşa edilmiş olup kurallar ve çerçeve yapılarla mantığa uygun sonuç üretirler. Sonuç üretme mekanizması ve bilgi alanlarının farklı olması çok önemli bir özelliktir ve bilgi temeli sonuç üretme mekanizmasından ayrı olarak değiştirilebilir.

Yine Jakson "An introduction to expert systems'' adlı kitabında genel yapay zekâ programları ve uzman sistemlerin üç yolla ayrıldıklarını belirtmiştir;

- Uzman sistemler gerçek problemleri çözerler.
- Uzman sistemler' in sonuçları kabul edilebilir sonuçlardır.
- Uzman sistem' in sonuçları bir uzmanınki kadar doğrudur.

Uzman sistemler, gerekli oldukları durumlarda gerçekleştirilmeleri gereken sistemlerdir. Gerek tasarım, gerekse uygulama aşamaları zor ve uzun bir süreç olan bu sistemleri hazırlamaya başlamadan önce uzman sistem ihtiyacının net ve doğru bir şekilde ortaya konmuş olması gerekmektedir. Aşağıda uzman sistemlerin kullanılabileceği durumlardan bazıları belirtilmiştir.

- Çözüm için basit bir bilgisayar programı yeterli değil ve karar verme süreci karmaşıksa,
- Karar verme mekanizması bir kurallar zincirine dönüştürülebiliyorsa,
- Bir olaya ilişkin çok sayıda doğru sonuç çıkıyorsa,

- Çok miktarda dikkate değer veriyi anlamlandırmanın ya da bilgilerin geri çağrılması işlemlerinin yapılmasının gerektiği durumlarda

Uygulama öneri, sınıflama, teşhis, yorum, açıklama, çözüm yolu seçme, durumu değerlendirme ve tahmin etme üzerinde yoğunlaşıyorsa uzman sistemlerin kullanılması uygundur [13].

Uzman sistemler ancak gerekli olduğu zamanlarda kullanılmalıdır. Bazı işletmelerin bulunduğu koşullar, uzman sistemin kullanılmasını gerek maliyet, gerekse sağlanan faydanın önemsiz olması yüzünden haklı kılmayabilir. Uzman sistemlerden aşağıdaki koşullar altında yararlanılması tavsiye edilmektedir:

- İşin tekdüzeliğine bağlı olarak uzman sistemin sık sık kullanılmasına gereksinim duyuluyorsa ve kullanıcı sayısı uzman sistemin kullanılmasını maliyet boyutunda ekonomik kılacak kadar fazla sayıda ise,

- Karar verme durumu karmaşıksa (basit durumlar için basit bir bilgisayar programından da yararlanılabilir.) ,

- Karar verme mantığı bir kural hiyerarşisine dönüştürülebiliyorsa,

- Gözden geçirmenin çok uzun zaman alacağı, çok sayıda mümkün kombinasyonun olduğu problemlerde,

- Çok miktarda dikkate değer veriyi anlamlandırmanın ya da bilgilerin geri çağrılması (retrieval) işlemlerinin yapılmasının gerektiği durumlarda

- Uygulama öneri, sınıflama, teşhis, yorum, açıklama, çözüm yolu seçme, durumu değerlendirme ve tahmin etme üzerinde yoğunlaşıyorsa uzman sistemlerin kullanılması uygundur [18].

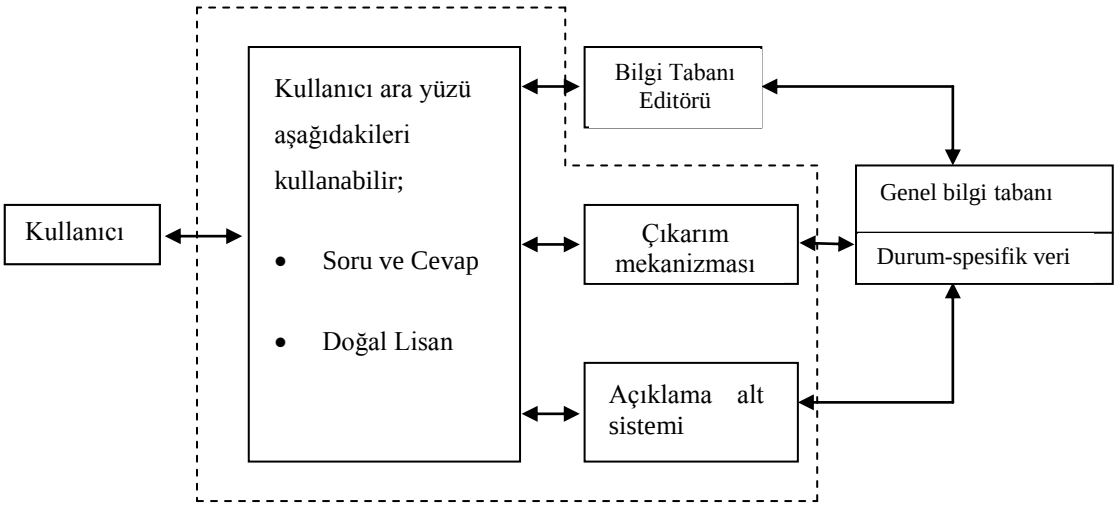
3.1 Uzman Sistemlerin Gelişim Süreci

Uzman sistem alanındaki öncü proje DENDRAL'dir. Bu proje 1965'te E.Feigenbaum ve meslektaşları tarafından Birleşik Devletler Standford Üniversitesinde bir kimyagere, organik bir bileşiğin yapısını, kitle spektrogramının ve ham kimyasal formülünün verileriyle bulması için, yardımcı olmak üzere başlatılmıştır. Fizik-kimya alanına özgü bilgiler usa vurma mekanizmalarına sıkı sıkıya bağlıydı. Dolayısıyla projeden, bir alanın özgül bilgilerini, yorumlama mekanizmalarında açık bir şekilde ayırmanın gerekli olduğu düşüncesi çıktı. Bilgi tabanlı sistemlerin ve uzman sistemlerin asıl temeli zamanla, verili bir problemi çözmek için bir bilgiler ve olgular bütününü kullanan çıkarım mekanizması kavramıyla birlikte doğmuştur. Daha sonra tıp alanında yeni programlar geliştirilmiştir [Allahverdi, 2003]

3.2 Uzman Sistemlerin Yapısı

Genel olarak uzman sistemlerde aşağıdaki bileşenler vardır. (Jacksoni 1990);

- Belli bir problem hakkındaki gerçekler, kurallar ve bilgileri içeren; Bilgi Tabanı,
- Problemlere çözümler üretmek üzere depolanan bilgiyi ustalıkla kullanan; Çıkarım Mekanizması,
- Kullanıcı ile iletişimi sağlamak üzere; Kullanıcı Arayüzü,
- Bilgi tabanını geliştirmeye yardım etmek üzere; Bilgi Edinim Modülü.



Şekil 3.1. Tipik bir uzman sistemin yapısı (Luger, 1980)

Uzman sistemler, belli bir algoritma yapısı kullanan ve kesin sonuçlar üreten yordamsal programlardan çok farklıdır. Uzman sistemler de kesin sonuç olmayabilir. Bilgi ve kural tabanını kullanarak yorum yaparlar.

Uzman sistemler de "Bilgi – Kural Tabanı" olarak adlandırılan bileşen, problemin çözümüne ilişkin bilgi ve kuralları içerir. Bu bilgiler, insanlar, kitaplar, veri tabanları, özel araştırma raporları ve kullanıcının kendi deneyimleri vb. kaynaklardan temin edilerek uzman bilgisayar sistemleri tarafından tanınabilecek formatlara çevrildikten sonra bilgi tabanına aktarılırlar. Uzman sistemlerle yordamsal programları birbirinden ayıran en önemli yapı bilgi tabanıdır. Çünkü yordamsal programlar geniş bir bilgi tabanı kullanmak yerine sınırlı miktarda bilgiyi girdi parametresi olarak kullanan basit veya gelişmiş algoritmalarla probleme çözüm ararlar. Algoritmaların kullanacakları bu bilgilerin de program kodunun içerisine yerleştirilmesi gerekmektedir. Bilgi ve kurallarda değişiklik olduğu zaman veya sisteme yeni kurallar eklenmesi gerektiği zaman da program kodunun değiştirilmesi ve yeniden derlenmesi gerekmektedir. Uzman sistemler ise, ufak bilgi parçalarını bilgi tabanına toplarlar ve buna uygun bir problemin değerlendirilip çözülmesinde kullanılırlar. Farklı bir problemle karşılaşıldığında ise, yordamsal programlamanın aksine, bilgi

tabanının sınırları dahilinde tekrar programlama yapılmadan probleme çözüm aranabilir.

Çıkarılan sonuçların gerekçelerini açıklaması, güven seviyeleriyle uğraşması ve kuşku gibi özellikler uzman sistemlerin yordamsal programlara göre artı özellikleridir. [14]

Bir uzman sistem iki ana parçanın birleşiminden oluşur. Geliştirme çevresi ve görüşme çevresi. Gelişme çevresi sistemin bileşenlerini kurmak ve uzman insan bilgilerini bilgi tabanına girmek için uzman sistemi kuranlar tarafından kullanılır. Görüşme çevresi ise uzman bilgi ve nasihatlerine ulaşabilmek için uzman olmayanlar tarafından kullanılır.

Bilgi Tabanı: Problemlerin anlaşılması, formülasyonu ve çözümü için gerekli olan tüm bilgileri içerir. Olaylar ve durumlar hakkında bilgi ve bunlar arasındaki mantıksal ilişki yapılarını ihtiva eder. Ayrıca standart çözüm ve karar alma modellerini de içerir.

Çıkarım Mekanizması: Uzman sistemin beynidir. Bilgi tabanı ve çalışma alanında bulunan bilgiler üzerine düşünmek için bir metodoloji sunan ve sonuçları biçimlendiren bir bilgisayar programıdır. Bir başka deyişle problemlere çözümler üreten bir mekanizmadır. Burada sistem bilgisinin nasıl kullanılacağı hakkında karar alınır.

Çalışma Alanı: Giriş verileri tarafından belirlenmiş problem tanımları için hafızanın bir köşesinde bulunan çalışma alanıdır. Bu alan işlemlerin ara seviyelerindeki sonuçları kaydetmek için de kullanılır.

Kullanıcı Arabirimi: Kullanıcı ile bilgisayar arasında bir çevirmen rolünü üstlenmiştir.

Açıklama: Uzman sistemleri diğer sistemlerden farklı yapan modüllerden birisidir. Açıklama modülünden kasıt, kullanıcıya çeşitli yardımların verilmesi ve soruların

açıklanması olduğu kadar, uzman sistemin çıkardığı sonucu nasıl ve neden çıkardığını açıklayabilmesidir. Burada uzman sistem karşılıklı soru cevap şeklinde davranışlarını açıklar.

Düşünme kapasitesini iyileştirme: Bir uzman insan kendi performansını analiz edebilir, öğrenebilir ve gelecekteki kullanım için onu iyileştirebilir. Sistemlerin de bu tip davranışlar göstermeye ihtiyacı vardır. Sistemin kendini iyileştirmesi öğrenme ile ilgili bir konudur. Sistemlerin bir uzman insan gibi öğrenebilmelerine yönelik çalışmalar sinirsel ağlar üzerinde sürdürülen araştırmalarla devam etmektedir. Amaç bir insan beyni gibi çalışan yapay zekâyı geliştirebilmektir

Son zamanlarda uzman sistemlerin geliştirilmesinde uzman sistem kabukları denilen sistemlerden de istifade edilmektedir. Bunlar hazır hale getirilmiş, çıkarım mekanizması ve bilgi saklama özellikleri ile donatılmış sistemler olup sadece alan bilgisi olmayan içi boş uzman sistemlerdir. Ayrıca kullanıcının kendisinin özel çıkarım mekanizması geliştirmesine imkân veren daha gelişmiş sistemler de vardır.

3.3 Uzman Sistemlerin Temel Çalışma Prensibi

Uzman sistemlerin temel çalışma prensipleri şöyledir. Programı kullanan kişi uzman sisteme gerçekleri (facts) verir ve karşılığında uzman tavsiyesi veya uzmanlık alır. Uzman sistemler genelde iki ana unsurdan oluşur. Bunlardan birisi bilgi ve kural tabanı olup, doğruluğu önceden bilinen gerçekleri içerir. İkinci unsur olan karar motoru ise, bilgi tabanında bulunan bilgiyi kullanarak kullanıcının sorduğu sorulara uygun sonuçlar çıkarır.

Kurallar (rules), problemin teoride veya pratikte çözümüne ilişkin bilgileri ifade eder. Örneğin; “Bir banka için, müşterinin yaşı 18 den küçükse ve çekeceği para miktarı 1000 ytl üzeri ise ebeveyn imzası gerekmektedir.”

3.4 Uzman Sistem Tasarımı

Uzman sistemler tasarlanırken ilk sorulması gereken soru probleme geleneksel programlama ile çözüm bulunup bulunamayacağıdır. Çünkü geleneksel programlama ile çözüme ulaşılabilecek bir problem için uzman sistem kullanılması gereksiz olacaktır. Bu konuda bir karara varabilmek için göz önünde bulundurulması gereken bazı kriterler mevcuttur. Bunlardan bazıları şunlardır.

- Problem yordamsal programlama ile çözülebilir mi ?

Eğer bu sorunun cevabı "Evet" ise, bu durumda Uzman Sistemlerle uğraşmanın en iyi çözüm olmadığı barizdir. Örneğin arıza yapan bir cihazdaki hatayı tespit etmek için, bir arızaya yol açan bütün nedenler bilindikten sonra yapılması gereken tek şey, sadece bir hata ve çözüm tablosuna bakmak yeterli olur. Uzman Sistemler daha çok algoritmik çözümü olmayan, eldeki verileri kullanarak sonuç çıkarmak gereken problemler için uygun olur.

- Uygulama yapılacak olan alanın sınırları kesin olarak belli mi ?

Tasarlanacak Uzman Sistemin bilmesi gerekenler ve yeteneklerinin sınırları tam olarak belirli olmalıdır. Örneğin baş ağrılarına teşhis koymak için kullanılan bir Uzman Sistem düşünelim. Böyle bir Uzman Sistem için bir tıp doktorunun bilgisine başvurmak gerekir. Fakat baş ağrılarının nedenlerini çok ayrıntılı bir biçimde araştırmak, tavsiyede bulunmak için nörologun da bilgisine ihtiyaç duyulur. Daha ayrıntıya girildikçe biyokimya, kimya, moleküler biyofizik vs. gibi pek çok alan ile ilgili bilginin de bilgi tabanına eklenmesi gerekir. İşte böyle durumlara düşmemek için, Uzman sistemin alanının mutlaka iyi bir şekilde sınırlandırılması gerekir. Aksi takdirde Uzman Sistem çok karmaşık hale gelir.

- Uzman sisteme ihtiyaç var mı ?

Pek çok uzmanın bulunduğu bir alanda uzman sistem hazırlamak pek de mantıklı olmaz. Çünkü böyle bir uzman sistemin dayanak noktası az bulunan uzman bilgisinden ziyade, çok sayıda uzmandan rahatlıkla alınabilecek bilgiye dayanır.

- İşbirliği yapabilecek en az bir uzman kişi mevcut mu ?

Tasarlanan projeye ilgi duyan en az bir uzman kişinin bulunması gereklidir. Çünkü herkes, bilgisinin doğruluğunun test edilmesini kabul etmeyebilir. Bir projeye çok fazla sayıda uzman kişinin katılması da bazen sakıncalı olabilir. Bir problemin çözümü için iki uzman farklı testler ve çözümler önerebilir. Bazen, farklı sonuçlara da varabilirler.

- Uzman kişi bilgisini anlaşılabilir derecede açıklayabiliyor mu ?

Bir Uzman Sistem hazırlanırken, bilgisine başvurulan uzman kişinin kendisine net bir biçimde ifade edebilmesi çok önemlidir. Tasarım yapan mühendis, uzman kişinin söylediklerini net bir şekilde anlayamazsa, aldığı bilgiyi bilgisayar koduna dönüştürmesi çok zor olur.

Uzman sistem tasarımına karar verdikten sonra aşağıdaki 6 adım uygulanmalıdır.

1. Uzman Sistem tasarımı için bir araç seçilir.

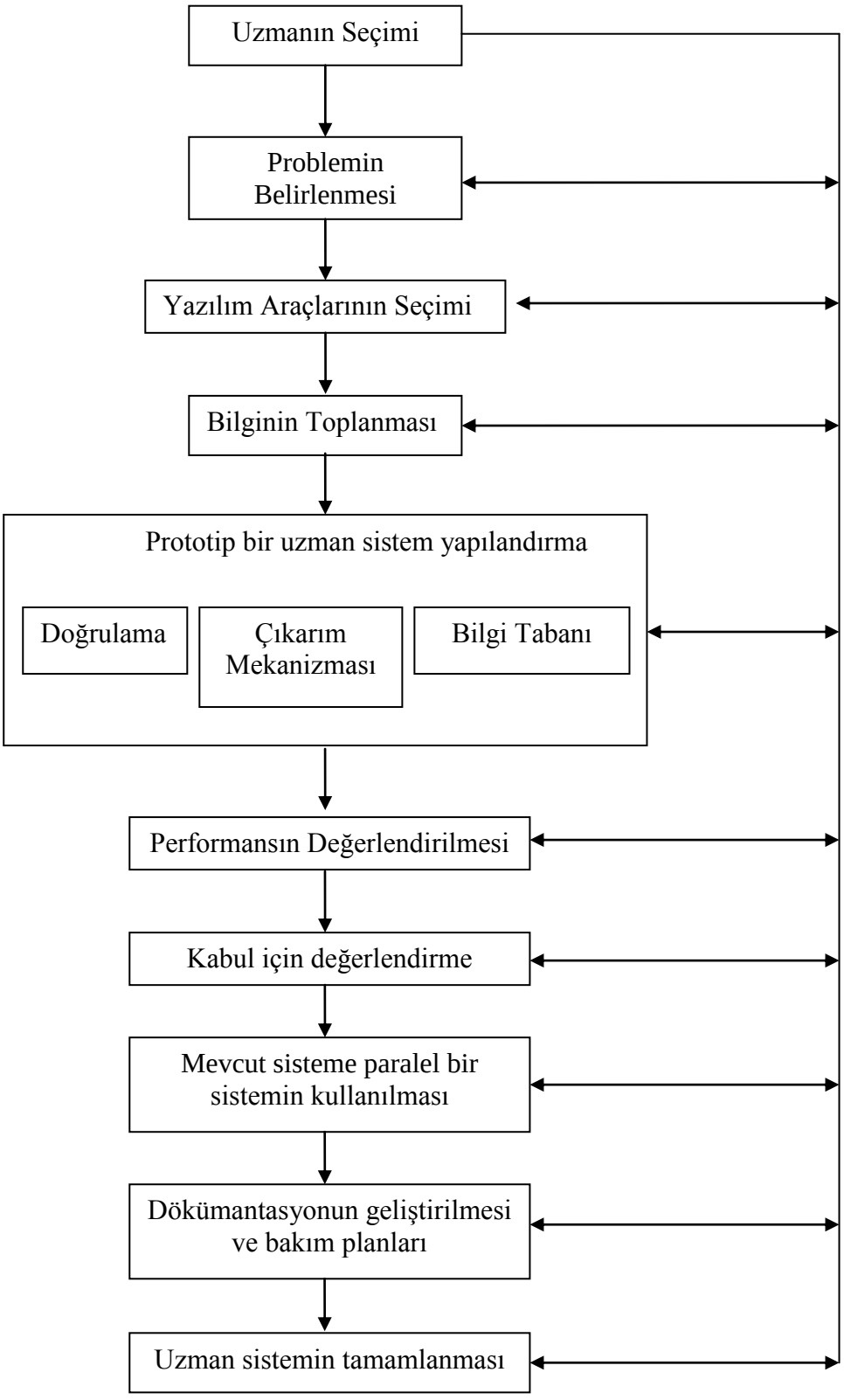
2. Problemin tam olarak ne olduğu belirlenir, sistemin içermesi gereken bilgi analiz edilir.

3. Sistem dizayn edilir. İlk olarak bu dizayn sistemin kağıt üzerinde tasvir edilmesini içerir. Yani akış diyagram ve matrisleri, sistemin kurallarının taslakları bu dizaynın içerisinde yer alır.

4. İlk adımda belirlenen araç kullanılarak sistemin bir prototipi oluşturulur. Bu prototip, bilgi tabanının oluşmuş halini, test edilmesini ve birçok denemenin gerçekleştirilmesini içerir.

5. Sistem istenilen hale gelene kadar üzerinde genişletmeler ve testler yapılarak sistem gözden geçirilir.

6. Sistem son halini alır ve yeri geldikçe üzerinde yeni düzenlemeler yapılabilir.



Şekil 3.2. Uzman sistem oluşturulurken izlenecek prosedür (Vural, 1998)

3.5 Uzman Sistemlerin Faydaları

Maliyet Azalması: Uzman sistem kullanımı ile karşılaştırıldığında insanların incelemeleri daha pahalı görülmektedir. Böylece kullanıcı başına düşen uzmanlık maliyeti azalmış olur.

Hazır Bilgi: Hazırlanan uzman sistem programı sayesinde uzman bilgisi herhangi bir bilgisayara yüklenebilir. Bilgi almak için uzman kişiyi beklemeye gerek kalmaz.

Verimlilik artışı: Uzman sistemler insanlardan daha hızlı çalışır. Artan çıktının anlamı, daha az sayıda insan ve daha düşük maliyettir.

Kalıcı Bilgi: Zamanla emekli olabilen veya hayata veda eden insan uzmanların aksine, uzman sistem bilgisi kalıcıdır.

Açıklama: Uzman sistem, varılan sonucun nedenlerini ayrıntılı olarak açıklar. Oysa bir insan bunu her zaman yapamayabilir.

Kalite iyileştirmesi: Uzman sistemler tutarlı ve uygun nasihatler vererek ve hata oranını düşürerek kalitenin iyileştirilmesini temin ederler.

İşleyiş hatalarını azaltma: Bir çok uzman sistem hatalı işlemleri tespit etmek ve onarım için tavsiyelerde bulunması için kullanılır. Uzman sistem ile bozulma sürelerinde önemli bir azalmanın sağlanması mümkündür.

Esneklik: Uzman sistemlerin kullanımı üretim aşaması ve servis sunulması sırasında esneklik sağlar.

Daha ucuz cihaz kullanımı: İzleme ve kontrol için insanların pahalı cihazlara bağlı kaldığı durumlar vardır. Fakat Uzman Sistemler ile aynı görevler daha ucuz cihazlarla yerine getirilebilir.

Tehlikeli çevrelerde işlem: Bazı insanlar tehlikeli çevrelerde çalışırlar. Uzman sistemler ise insanların tehlikeli çevrelerin dışında kalmasına imkân sağlar. Uzman

sistemler, insanlar için zararlı veya tehlikeli olan bütün ortamlarda rahatlıkla kullanılabilir.

Güvenilirlik: Uzman sistem güvenilirdir. Uzman Sistem bilgilere ve potansiyel çözümlere üstün körü bakmaz, tüm detayları yorulmadan ve sıkılmadan dikkatlice gözden geçirir.

Cevap verme süresi: Bazı durumlarda hızlı veya gerçek zamanlı cevap vermek gerekebilir. Kullanılan yazılım ve donanıma bağlı olmak şartıyla, bir Uzman sistem, özellikle verilerin büyük bir kısmının gözden geçirilmesi gerektiğinde bir insandan çok daha hızlı cevap verecektir.

Tam ve kesin olmayan bilgi ile çalışma: Basma kalıp bilgisayarlar ile karşılaştırıldığında, uzman sistemlerin insanlar gibi tam olmayan bilgi ile çalışabildiği görülmektedir. Bir görüşme sırasında sistemin bir sorusuna kullanıcı "bilmiyorum" veya "emin değilim" şeklinde bir cevap verdiğinde, uzman sistem kesin olmasa bile bir cevap üretebilecektir.

Eğitim: Uzman sistemin açıklayabilme özelliği bir öğretim cihazı gibi kullanılarak eğitim sağlanabilir.

Problem çözme kabiliyeti: Uzman Sistemler, uzmanların yargılarını bütünlemeye imkân sağlayarak problem çözme kabiliyetlerini yükseltirler. Bu sistemler bilgileri nümerikten ziyade sembolik olarak işledikleri için bir çok yöneticinin karar alma stilleri ile uyumludur.

Sınırlı bir sahada karışık problemlerin çözümü: Uzman sistemler insan yeteneklerini aşan karışık problemlerin çözümünde kullanılabilir.

Duygusallıktan Uzak Cevaplar: Stres veya kırgınlıktan dolayı verimli olarak çalışamayan bir insanın aksine, bir Uzman Sistem gerçek zamanlı sorunlara duygusallıktan uzak gerçekçi cevaplar verebilir.

Akıllı Veritabanı: Uzman sistemler, bir veritabanı dosyasına akıllıca erişebilir.

3.6 Uzman Sistem Örnekleri

Uzman sistemlerin ortaya çıktıkları yıllar olan 1970 ‘ lerden bugüne yapılan çalışmalardan bazıları şunlardır.

• **DENDRAL (Kimyasal Analiz Uzman Sistemi):** Genelde ilk uzman sistem olarak kabul edilen DENDRAL başarılı bir uzman sistem olduğu için halen dünyanın pek çok yerinde kimyagerler tarafından kullanılmaktadır. Hatta bazı işleri yapmada DENDRAL’ in, herhangi bir uzmandan daha başarılı olduğu kabul edilmiştir. DENDRAL Uzman Sistemi, önce geleneksel programlama yaklaşımıyla hazırlanmıştı. Daha sonra sonuç almak için geçen zamanı düşürmek için buluşsal metotlar denenmiştir. DENDRAL, varılan sonucun nedenini açıklamak yerine, sadece vardığı sonucu kullanıcıya sunardı.

• **HEARSAY (Konuşma Tanıma Uzman Sistemi):** Carnegie-Mellon Üniversitesi’nde geliştirilen HEARSAY I (1969) ve HEARSAY II (1971) uzman sistemleri, bilgisayara ses dalgası olarak verilen insan konuşmasını yazılı bir çıktı halinde ekranda görüntülemek amacıyla hazırlanmıştı. Bu ses dalgasından, konuşmanın ne olabileceği hakkında hipotezler türetilmekte ve daha sonra en iyi tahmin ise sonuç olarak sunulmaktaydı. Bu Uzman Sistemin gelişmesinde kullanılan önemli bir yenilik, birden fazla bilgi tabanının kullanılmış olmasıydı. Bu bilgi tabanları, bütün bilgi tabanlarınca işleyen hafıza olarak kullanılan bir kara tahta vasıtasıyla birbirleriyle iletişim kuruyorlardı. Kullanılan bilgi tabanlarının her biri sırasıyla, çözümlenmesi istenen konuşmanın bir yönü üzerinde yoğunlaşıyordu. Günümüzde kullanılan Uzman Sistem kabuklarının bazıları, bu kara tahta kavramını kullanmaktadır. 1975 yılında tamamlanan bu projenin kelime kapasitesi 1000 civarında idi. Sistem, verilen konuşmanın yaklaşık %75’ini başarılı bir şekilde anlayabiliyordu. Bu işlemi başarması için gereken süre, bir insanın aynı konuşmayı anlaması için gereken zamandan sadece birkaç kat daha uzundu. HEARSAY Uzman Sisteminin ortaya koyduğu sonuçlardan biri, daha önce konvansiyonel metotlar

kullanılarak yapılmaya çalışılan bir iş için, buluşsal yaklaşımın daha üstün olması idi. Çünkü geleneksel yöntem, istatistiksel araç kullanılan analitik yönteme dayanıyordu.

• **MYCIN (Kan Enfeksiyonları Uzman Sistemi):** MYCIN, 1970'li yıllarda Standford Üniversitesi'nde geliştirildi. Bugüne kadar hiçbir zaman kullanıma sokulmamasına rağmen, MYCIN, bugüne kadar geliştirilen Uzman Sistemlerden en çok bilinenidir. Uzman Sistemler konusunda hazırlanan yazıların çoğu MYCIN üzerine yoğunlaştığından, bu Uzman Sistem bu alanda yapılan sonraki çalışmaları teşvik edici bir rol oynamıştır. MYCIN, Uzman Sistemlerin büyük ve karmaşık gerçek problemleri çözmek üzere nasıl tasarlanması gerektiğini göstermek amacıyla yapılan bir araştırma sonucu ortaya çıkmıştır. Çalışma, menenjit ve bakteriyel enfeksiyonların teşhis ve tedavisinde doktorlara yardımcı olmak amacıyla gerçekleştirilmiştir. MYCIN' in kullandığı bilgi veritabanında, doktorların bazı enfeksiyonları teşhis ederken kullandıkları buluşsal kurallar vardı. MYCIN' in kullandığı bu veritabanı silinip yerine başka grup hastalıkların teşhisi için kullanılan kurallar konulduğunda ortaya başka grup hastalıkların tanınmasında kullanılabilecek başka bir uzman sistem ortaya çıkmış oluyordu. Böylece uzman sistem kabuğu olarak tanımlanan eklenti (plug-in) kavramı ortaya çıkmış oldu.

• **XCON (Bilgisayar Konfigürasyon Uzman Sistemi):** Rutin işlerde kullanılan uzman sistemlere verilen en klasik örnek, Digital Equipment Corp. (DEC) tarafından geliştirilen XCON isimli uzman sistem, DEC'in bilgisayar sistemlerinin konfigürasyonunda kullanılmak için hazırlanmıştır. Bir bilgisayar sisteminin konfigürasyonu yapılırken, müşterinin istediği özellikleri taşıyan parçalar bir araya getirilir ve üretim yapılan yerde testi yapılır. Bilgisayarı oluşturan parçaların çok fazla alternatifi olduğu için, sadece istenilen sayıda parçayı paketleyip yollamak mümkün değildir. Kullanılmakta olan Uzman Sistemlerin en başarılılarından biri olan XCON, üreticisi olan firmaya milyonlarca dolar kazandırmaktadır. Aynı zamanda alınan siparişleri hazırlamak için gerekli zaman oldukça azalmakta ve siparişin hatasız olarak yerine getirilmesini sağlamaktadır. XCON, bir konfigürasyonu bir insandan yaklaşık olarak 15 kat daha hızlı yapabilmektedir.

Yapılan testlerde insanların yaptığı konfigürasyonlarda yaklaşık %70'lik bir doğruluk saptanırken, XCON ile bu oran %98 olarak tespit edilmiştir. Kullanılmakta olan ilk ve tek sistem elbette XCON değildir. Ancak XCON'un hazırlanması ve ne kadar başarılı olduğunun duyulmasından sonra bu alanda yapılan çalışmalara hız verildiği için, XCON'un önemi büyüktür.

• **GATES (Havaalanı Pist Tayini ve İzleme Uzman Sistemi):** GATES, JFK (New York) havaalanında halen kullanılmakta olan bir uzman sistemdir. Sistem, TWA şirketinde çalışan yer kontrolü personeline, gelen ve giden uçaklara pist tayininde yardımcı olmak amacıyla hazırlanmıştır. Sistemde kullanılan bilgi tabanı, bu işi günlük olarak yapan bir yer kontrol uzmanının bilgisine başvurarak hazırlanmıştır. Pist tayin problemi uçuş saatlerindeki gecikme, kötü hava şartları, mekanik arızalar ve benzeri durumlarda çok karmaşık bir hale gelebilir ve çok hızlı çözüm gerektirir. Önceleri doğrusal programlama metoduyla çözülmeye çalışılan işlem, gerçek zamanlı probleme yeterince cevap veremediğinden, aynı sistem daha sonra mantıksal programlama dilleri kullanılarak bir PC üzerinde çalışacak şekilde hazırlandı. Sistem TWA' nın uçuşlarla ilgili veritabanına doğrudan erişerek, günde 100'den fazla uçuşa yönelik pist tayinini yaklaşık 30 saniye gibi kısa bir zamanda yapmaktadır. Aynı iş, daha önce uzman bir insan tarafından yaklaşık 10-15 saatte hazırlanıyor ve günlük değişiklikler için yaklaşık 1 saat vakit harcanıyordu.

• **HESS (Petrokimya Sanayisi İçin Uzman Sistem Planlama):** HESS, büyük bir petrokimya firmasının rafinerisinde üretilen ürünlerin planlanması için Houston Üniversitesi'nde geliştirilen bir Uzman Sistemdir. Sistemde kullanılan bilgi tabanı, firmanın rafinerisinde çalışan iki planlama uzmanının bilgisine başvurularak hazırlanan buluşsal kurallardan oluşmaktadır. Bu uzmanların görevi, hangi ürün veya ürünlerin hangi zamanda ve hangi süreçlerde üretileceğine karar vermekti. Uzmanların performansı, üretim maliyeti, imalat hataları ve kaybedilen müşteri satışlarına göre değerlendiriliyordu. Yaklaşık 12 ayda hazırlanan ve PC üzerinde çalışan HESS, üretimle ilgili 400 kadar kural içeriyordu. Sistem, daha önce uzman insanlar tarafından yapılan planlama işini çok daha tutarlı bir şekilde yapmaktadır.

Rafineride çalışan personelin ifadelerine göre, böyle bir sistemin hazırlanması ve kullanılmasıyla elde edilen yıllık kazanç milyon doları bulmaktadır. Sistem, firmanın veritabanına erişebilecek şekilde yeniden tasarlanmıştır.

• **DustPro (Maden Ocağı Güvenliği Uzman Sistemi):** ABD Maden Ocakları Bürosu tarafından geliştirilen DustPro, madencilik işlerinde hava kalite kontrolü yapabilen sınırlı sayıdaki uzmanın yerine kullanılmak üzere hazırlanmıştır. Madencilik operasyonlarındaki güvenlik şartları, havadaki kömür ve silis miktarına dayanarak düzenlenmektedir. ABD'de yaklaşık 200 maden ocağı bulunduğu dikkate alınırsa, her bir maden ocağında bir kişi olmak üzere, sürekli gözetim yapması gereken 200 uzmana ihtiyaç olduğu açıktır. Metan gazı emisyonu ve toz izleme sistemleri ile ortak çalışan DustPro, yaklaşık 15 dakikada sonuca ulaşmaktadır. Bilgi tabanı yaklaşık 200 kuraldan oluşan DustPro, dünyadaki pek çok maden ocağında kullanılmaktadır.

• **TOP SECRET (Güvenlik Sınıflandırma Uzman Sistemi):** ABD Enerji Bakanlığında, nükleer silah sistemleri güvenlik verilerine erişmek için yaklaşık 100 kadar sınıflandırma mevcuttur. Bakanlık bünyesindeki zahmetli işlerden biri, bir belgenin bu sınıflandırma kriterlerin doğrultusunda doğru biçimde sınıflandırılmasıdır. Bu sınıflandırmada yapılan şey, bir belgeyi kimin görebileceği veya göremeyeceğine karar vermektir. Oluşturulan bilgi tabanında, bu güvenlik sınıflandırmalarına dayanarak hazırlanan kurallar bulunmaktadır. Sistem belgeleri güvenlik derecelerine göre sınıflandırmakta ve daha önce bu işi yapan personelin yükünü oldukça hafifletmektedir.

• **Codecheck (Bilgisayar Programı Değerlendirme Uzman Sistemi):** Abraxas Yazılım şirketi tarafından hazırlanan bu uzman sistemin temel fonksiyonu, C dili ile hazırlanan programların değerlendirilmesidir. Kural-tabanlı bir uzman sistem olan Codecheck'in yaptığı değerlendirme neticesinde program içerisinde karmaşık olan bölümler, biçimleme, standartlara uygunluk tespit edilmektedir. Böylece programın sonradan bakımı kolaylaşmakta ve program başka ortamlara taşınabilmektedir.[14]

4 UZMAN SİSTEMLERİN PROGRAMLANMASI

Akıl yürütme, felsefe ve matematiksel mantığın konusudur. Geçerli mantıksal tartışmaların ardındaki ilkelerin açık ve kesin ifadeleri, Eski Yunan'da başlamış olup günümüzde de hala matematik ve felsefenin bir konusu olarak devam etmektedir. Tamamen Boolean mantığı (Boolean logic) üzerine kurulmuş olan bilgisayarlar, 20.yüzyıl mantığının uygulamada gerçekleştirdiği çok büyük bir başarıdır. Klasik mantığın bir ürünü olan bilgisayarlar, zeki programlar ve makineler yapma gayesiyle bu mantığın daha ileri biçimlerini kullanan yapay zekâ için uygun altyapılar olarak görülmektedir.

Önermesel Mantık: Mantık (logic), sıradan dildeki durumlarda kullandığımız akıl yürütme işlemlerinin özünü ortaya çıkarma teşebbüsüdür. Doğal dildeki hesaba kattığımız durum özelliklerinin sayısı, ürettiğimiz resmi mantık türlerini sınırlandırır. Mantığın en basit biçimlerinden biri önermesel mantıktır. Aşağıda önermesel mantık yöntemiyle yorumlanmış bir durum örneği verilmiştir:

A -> B
AND B -> C
THEN A -> C

-> işareti, ise, o halde yada gerektirir gibi anlamlara gelir. İse işareti, önermesel mantıktaki "Eğer A doğruyasa B doğrudur" ifadesini temsil eder.

Yüklem Mantığı: Önermesel mantık, akıl yürütmenin artan karmaşık görünümünü yakalamış mantık çeşitlerinden en basit olanıdır. Önermesel mantığa dönüştürmenin güç olduğu ifadeler bulmak hiç de zor değildir. Klasik mantığın en kullanışlı biçiminin yüklem mantığı (predicate logic) olduğuna dair genel bir görüş mevcuttur. Yüklem mantığı, önermesel mantığın üzerine inşa edilmiştir.

"Tüm babalar erkektir."

Yukarıdaki durumun önermesel mantıkla temsil edilebilmesi için yeryüzündeki tüm babalar için birer önerme oluşturulması gerekir. Ama, bir insanın baba olması onun erkek olduğunu da gösteren bir durum olduğu bilinen bir gerçektir. Bu durum, geri dönüş değeri boolean (doğru/yanlış) olan bir yüklem işlevi (predicate function) vasıtasıyla karar verilebilir. Boolean değerler, DOĞRU ve YANLIŞ olmak üzere iki tanedir. Örneğin;

baba(x).

x değişkeninin bir işlevidir ve eğer x babaysa DOĞRU; diğer tüm durumlarda da YANLIŞ değerini döndürür. Yüklemler kullanarak "Tüm babalar erkektir." ifadesi şimdi kolayca aşağıdaki gibi temsil edilebilir.

baba(x) −> erkek(x).

Daha karmaşık fikirleri de yüklemler yardımıyla temsil etmek mümkündür. Örneğin;

ebeveyn(x) AND erkek(x) −> baba(x).

önermesi, "x erkek ve ebeveyn ise babadır" anlamındadır. İlişkileri temsil etmek için de yüklemler kullanılabilir. Örneğin;

ebeveyn(x,y) −> cocuk(y).

önermesi, " x, y' nin ebeveyni ise y bir çocuktur" anlamındadır. ebeveyn (x,y) yüklem işlevi, şayet x, y' nin ebeveyniyse DOĞRU; diğer durumlarda da YANLIŞ değeri döndürür. Ayıraç içindeki x ve y gibi değişkenler, ait oldukları işlevin içsel değişkenleridirler. Bu değişkenlere değerler atayarak işlevler önermelerin içinden çağrılabilir.

4.1 Uzman Sistemlerde IF … THEN Mantığı

Önermesel mantık veya yüklem mantığı üzerine kurulu bir mantıksal durum, itiraz kaldırmaz. Ancak, seyrek de olsa insanların klasik mantık kullanma ihtiyacı duydukları durumlar olabilir. Klasik mantık, sadece doğru ve yanlış ile sınırlı değildir; çünkü gerçek dünyada, kesin olmama durumu ve edinilmiş deneyimler vardır. Yani insanlar, problemleri sadece ince ayrıntılar yardımıyla çözümlemeyip kendi deneyimlerinden de sıklıkla faydalanırlar. İnsanlar, problemlerin bazı parçalarını geçmişte gördükleri birtakım şeylerle karşılaştırırlar. [19]

İnsanların deneyimlerini kullanarak akıl yürüttükleri fikri, uzman sistemlerin ilham kaynaklarından birisi olmuştur. Birtakım sonuçları ve kurallarıyla beraber bilgi depolama yollarını düşünmek zor değildir. Örneğin, hava ile ilgili bilgiler şu şekilde olsun:

IF gök gürültüsü var

THEN yağmur yağma ihtimali çok yüksek

Bir parça bilgi, bir şartlar ve sonuçlar listesi olarak temsil edilebilir. Örneğin;

IF sonbahar mevsimi

AND gök gürültüsü

AND kara bulutlar

THEN fırtına

bilgileri, yıldırımlar hakkında bilinenler olsun. Yukarıda kullanılan IF … THEN ifadesi, eğer bir şeyler doğru ise bir şeyler yapılması için bir komut değil farklı olgular arasındaki ilişkinin ifadesidir ve mantıktaki ise (−>) ilişkisine benzer. Yukarıda ki ifade yüklem mantığının kurallarıyla yazılırsa,

kara(x) AND bulut(x) AND sonbahar(y) AND gokgurultusu(z) −> fırtına(t)

Ama IF . . . THEN yapısındaki kural, daha kolaydır, çünkü daha az resmidir.

Belirli bir IF . . . THEN kuralı kümesi ve bir kural veritabanı varsa şartlar topluluğunun sonucunu bulmak, basit bir veritabanı sorgulamasıdır. İşlemleri daha tutarlı kılmak amacıyla kısmi karşılaştırmalar yapılabilir ve hatta şartlar kümesine birden fazla kural eklenebilir. Örneğin, gök gürültüsü sesi varsa, yukarıda verilen, havayla ilgili her iki IF . . . THEN kuralı da karşılaştırılabilir. Bunlardan ilkinde, yağmurlu, ikincisinde ise fırtınalı olma ihtimalinden bahsediliyordu. Bunlardan hangisinin daha muhtemel olduğunu bulabilmek için daha fazla ayrıntı tanımlamak gerekecektir. [19]

Bir uzman sistemin üstünlüğü, kural tabanının (rule base) ne kadar iyi olduğuna bağlıdır. Oluşturulacak olan bu kural tabanı algoritma tabanlı dillerde kullanılan ve yukarıda bahsedilen IF … THEN yapılarıyla kurulabileceği gibi daha iyisi, sadece uzman sistemler için özel hazırlanmış diller kullanılarak istenilen programlar oluşturulabilir. [19]

4.2 Kural Tabanlı Dil Örneği – Prolog

Yordamsal programlama da problemin nasıl çözüleceği, bilgisayara adım adım yapılması gerekenlerle birlikte bir algoritmaya bağlı olarak tanıtılmaktadır. Bu yönteme alternatif olarak ortaya çıkan mantıksal programlama da ise bilgisayara işlemleri nasıl yapacağı değil ne yapacağı söylenmektedir.

Prolog bir yapay zekâ programlama dilidir ve alışık olduğumuz programlama dillerinden oldukça farklıdır. Bu farklılık yapay zekâ dillerinde program ve veri ayrımının olmamasından ileri gelir. Yapay zekâ dillerinde program da veriler de bir çeşit semboller bütünü olarak algılanır. Bunların yanı sıra geleneksel programlama dilleri sıralı, yordamlara bağlı bir yapı gösterirken, yapay zekâ dilleri semboller

(program ve veriler) arası ilişkilere dayalı olduğundan bu yapıya uymazlar, yani belirgin bir sıra takip etmezler. Bu ise daha dinamik bir yapı sağlar.

Mantıksal programlama dillerinden en çok kullanılan prolog dilinde hazırlanan programlar klasik programlama dillerinden farklı olarak, sadece komut satırlarından değil doğruluğu önceden kabul edilen gerçeklerden ve bu gerçeklerden sonuç elde etmeye yarayan kurallardan oluşur.

Prolog' a günlük konuşma diline benzer yapıda oluşturulmuş cümlecikler temel teşkil etmektedirler. Prologda ki çıkarım mekanizması, cümlecikler halinde verilen bilgileri kullanarak, cevap aranan problem için mantıksal bir şekilde karar vermeyi sağlamaktadır. Çıkarım mekanizması, programın sorduğu sorulara verilen cevapları eşleştirerek bilgi tabanında saklanan bilgiyi geri çağırır. Prolog sorulan sorulara verilen cevapları doğrulamak için, bilgi tabanında ki verileri tarayarak cevapların doğruluğu hakkında karar vermeye çalışır.

Prolog' un en önemli özelliklerinden birisi, sorulan sorulara ve problemlere tek bir çözüm getirmek yerine tüm alternatifleri inceleyerek mümkün olan tüm çözümleri bulmasıdır. Prolog, programı satır satır inceleyerek mümkün olacak tek bir sonucu bulmak yerine, gerektiğinde geri dönüşler yaparak birden fazla sonuçlar geri döndürebilir.

Prologda kullanılan cümlelerin (clause) yapısı, doğruluğu önceden bilinen gerçeklerin belirli formlarda yazılı bir biçimde ifade edilmesini sağlayacak şekilde geliştirilmiştir. Günlük konuşma ve yazı dilinde ki gereksiz kelimelerin atılmasıyla belirli bir forma sokulan cümleler sayesinde nesneler arasındaki ilişkiler tanımlanır.

4.2.1 Gerçekler (Facts)

Prolog' da nesneler arasındaki ilişkiler yüklem (predicate) olarak isimlendirilirler. Yüklem mantığına göre çalışan prolog da ilişkiler yazılırken, ilişkinin ismi ve hemen

yanında parantez içinde ilgili nesneler belirtilir. Gerçek, nesne veya nesneler arasındaki ilişkinin yalnızca tek bir yönünü temsil eder.

Ali Ayşe' nin ebeveynidir gerçeği prologda; *"**ebeveyn(ali,ayşe).**"* **ş**eklinde yazılır.

4.2.2 Kurallar (Rules)

Gerçekler arasında ilişkiler tanımlayarak kurulan yapılara kural (rule) denir. Kurallar gerçekleri (facts) kullanarak sonuç elde etmek için kullanılır. Kurallar iki kısımdan oluşur; baş ve gövde. Baş kısma aynı zamanda bağımlı ilişkide denir. Kuralların gerçeklerden farkı açıklayıcı bilgi taşıyan gövde kısmının olmasıdır.[20]

cocuk(ayşe,ali) :- ebeveyn(ali,ayşe)

İkinci kısmın doğru olması halinde, ilk kısmın geçerli olacağını ifade etmektedir. Yukarıda ki kural "eğer ali ayşe' nin ebeveyni ise ayşe ali' nin çocuğudur" cümlesini ifade etmektedir.

İki bölüm eğer anlamı taşıyan **":-"** işareti ile birbirinden ayrılırlar. İkinci kısımda bir veya birden fazla yüklemin doğruluğu kontrol edilebilir. Tüm yüklemlerin doğru olması durumunda o kuralın doğruluğu ispat edilmiş olur.

erkekkardes(Y,X):-ebeveyn(E,X),ebeveyn(E,Y), erkek(Y).

İkinci kısımda belirtilen her bir yüklem birbirinden **","** ile ayrılmaktadır. Tüm yüklemlerin doğruluğu tek tek kontrol edilir ve hepsinin doğru olması durumunda kuralın doğruluğu sağlanmış olur. Sadece bir yüklem bile yanlış cevap geri döndürürse bütün kural yanlış olur.

4.2.3 Hedefler (Goals)

Prologda gerçekler ve kurallar belirtildikten sonra bu verilere göre prolog' a soru sormak ve cevap almak mümkündür. Böylece prolog' a yerine getirmesi gereken bir hedef gösterilmiş olur. Aşağıdaki gerçekleri prolog' a tanıtmış olalım.

ebeveyn(ali,ayşe).

ebeveyn(ahmet,veli).

ebeveyn(ali,ahmet).

ebeveyn(suzan,ayşe).

ebeveyn(ayşe,cihan).

ebeveyn(murat,meral).

- ali ayşe' nin ebeveynimi ? sorusu prolog' a aşağıdaki şekilde sorulur ve tek bir satırlık yes cevabı alınır.

? – ebeveyn(ali,ayşe).

Prolog' un cevabı : YES.

- veli mehmet' in ebeveynimi ?

? – ebeveyn(veli,Mehmet).

NO.

- veli' nin ebeveyni kimdir ?

? – ebeveyn(X,veli).

X=Ahmet

- ayşe' nin ebeveyni kimdir ?

? – ebeveyn(X,ayşe)

X=ali ;

X=suzan ;

- Kimler kimlerin ebeveynidir ?

? – ebeveyn(X,Y).

X=ali

Y=ayşe;

X=suzan

Y=ayşe;

X=ahmet

Y=veli;

Prolog' a yukarıda ki gerçeklere dayalı olarak farklı sorularda sormak mümkündür. Prolog bu sorulara tüm gerçekleri ilkinden başlayarak tek tek değerlendirir ve uygun cevapları verir.

cihan' ın büyük ebeveyni kimdir ? sorusu, büyükebeveyn(X,cihan) şeklinde sorulursa, bilgi tabanında büyükebeveyn isminde bir ilişki olmadığı için cevap alınamaz. Ancak büyükebeveyn ilişkisi iki ebeveyn ilişkisinin iki kere kullanılmasıyla bulunabilir.

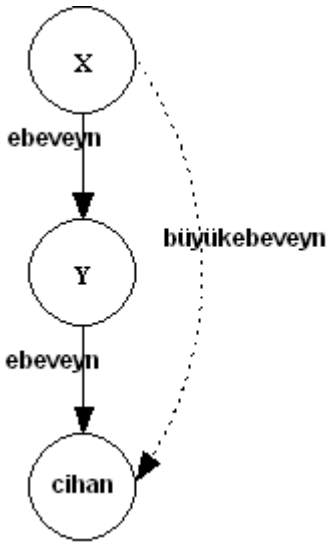
?– ebeveyn(Y,cihan), ebeveyn(X,Y).

Y=ayşe

X=ali;

Y=ayşe

X=Suzan



Hedefler ebeveyn(X, ayse) gibi basit olabileceği gibi, yukarıda ki örnekte ki ebeveyn(Y, cihan), ebeveyn(X, Y) hedefi gibi daha karmaşık da olabilir. Birden fazla parçadan oluşan hedefe Birleşik Hedef, her bir parçaya da alt hedef denir. Birleşik bir hedefin çözümünü bulmak için her iki alt hedefin doğru olması gerekir. Bu durumda, iki alt hedef arasında (ve) anlamına gelen ‘,’ kullanılır. Fakat istenilen durumlarda alt hedeflerden sadece birinin doğru olması şartı da aranabilir. Bu durumda alt hedefler arasında (veya) anlamına gelen ‘;’ kullanılması gerekir. Bu duruma Ayırma işlemi denilmektedir. [20]

4.2.4 Değişkenler

Prologda sorgulamalardan elde edilecek cevapları saklamak üzere kullanılan nesnelere değişken ismi verilmektedir. ebeveyn(Kim,ayse) cümlesi ile ayse’ nin ebeveyninin kim olduğu öğrenilebilir. Kim değişken olarak kullanılmıştır. Prologda değişkenler büyük harf veya “_” karakteri ile başlamak zorundadır. Bu şarta bağlı kalmak suretiyle değişkenler istenilen uzunlukta rakam, büyük küçük harfle belirtilebilir.

Mantıksal programlama dillerini, diğer dillerden ayıran en önemli özelliklerden birisi kullanılan değişkenlere atama ifadelerinin yer almamasıdır. Kullanılan değişkenler, almaları gereken değerleri atama ifadeleriyle değil gerçekler ve kurallardan elde edilen cevaplara göre alırlar. Diğer önemli bir fark da değişkenlerin tek bir değerle sınırlı kalmayıp kurallardan ve gerçeklerden elde edilen tüm sonuçları içerebilmeleridir. Yani bir kurala bağlı olarak elde edilen bir sonuç değişkene atandıktan sonra program tekrar başa dönerek tüm gerçekler ve kuralları yeniden inceleyerek mümkün olan diğer sonuçları da değişkene atayabilir.

Bir değişken, değer almadığı müddetçe serbest değişken olarak kalır. Fakat herhangi değer aldığı andan itibaren sınırlı hale gelir. Bu sınırlılık bir sorguya cevap almak için gerekli olan süre kadar sınırlı kalır. Bu işlem bittikten sonra değişken yeniden serbest hale gelir, program başa döner ve alternatif çözümler arar. Dolayısıyla bir

değişkene bir değer vererek bilgi depolanamaz. Değişkenler bilgi depolamak için değil, eşleştirme ve işlemenin bir parçası olarak kullanılır. [20]

derskayit(ayse,bil_151)

derskayit(ali,bil_151)

derskayit(mehmet,ing_151)

derskayit(ali,ing_152)

Yukarıda ki gerçekler prolog' a tanıtılmış olsun. bil_151 ve ing_152 derslerinin her ikisine de kayıt yaptırmış olan öğrenci kimdir sorusunun cevabını bulmak için prolog aşağıdaki sorguyu kullanır.

?- derskayit(Sahis, bil_151), derskayit(Sahis, ing_152).

Prolog bu kuralı çözmek ve önce derskayit(Sahis, bil_151) kısmının doğru olup olmadığını bulmak için bütün gerçekleri baştan sonra kadar inceler. Çözüm bulunmadan önce Sahis değişkeninin değeri yoktur, yani serbesttir. Öte yandan bil_151 kısmı bilinmektedir. derskayit(ayse,bil_151) gerçeğindeki bil_151 kısmı sorguya uyduğu için Sahis **'ayse'** değerini alır. Prolog aynı zamanda aşağıya doğru nereye kadar tarama yaptığını göstermek için kuralın başıyla eşleşen ilk noktaya bir işaret (pointer) koymaktadır.

Sorgunun ilk kısmı doğrulandıktan sonra ikinci kısmının da doğrulanması gerekir. Yani ing_152 dersine kayıt yaptıran kişinin de bulunması gerekir. Sahis değişkeni 'ayse' değeri aldığı için artık derskayit(ayse, ing_152) gerçeğinin doğru olup olmadığı araştırılır. Gerçekler incelenirse, 'ayse' isimli şahsın ing_152 dersine kayıt yaptırmadığı görülür. Bu durumda Prolog Sahis değişkenine atadığı 'ayse' değerini etkisiz hale getirir ve Sahis yeniden serbest hale gelir. Kuralın ilk kısmını doğrulayan gerçeği bulmak için Prolog bu kez kuralların başından değil, gerçekler listesine daha önce yerleştirmiş olduğu işaretten aşağıya kadar doğru taramaya başlar. İlk gerçek

gereken şartları sağlayamadığı için artık dikkate alınmaz. Bu işleme "**Geriye İz Sürme"** denir.

Yapılan taramada bil_151 dersine kayıt yaptıran kişinin "ali" olduğu görülünce Sahis değişkeni bu kez 'ali' değerini alır. Kuralın ikinci kısmının doğrulanması için tarama yapılırsa, ing_152 dersi için gereken şartın yine 'ali' ile sağlandığı görülür. Dolayısıyla Prolog'un vereceği cevap şu olur:

Sahis=ali

1 Solution

4.2.5 Anonim Değişkenler

Programların gereksiz satırlarla karmaşık hale gelmelerini engellemek için anonim değişkenler kullanılabilir. Böylece bir sorgulamadan beklenilen bilgileri alabilir ve ihtiyaç olmayan değerler iptal edilmiş olur. [20]. Anonim değişkenler prologda "_" karakteri ile gösterilir.

?- ebeveyn(Kim,_)

Yukarıda ki sorgu çalıştırıldığında, ebeveyn olan kişilerin isimleri listelenirken, çocukların isimlerinin yerine anonim değişken kullanıldığı için bu bilgiler gelmeyecek ve gereksiz satırlar engellenmiş olacaktır. Ayrıca anonim değişkenler gerçeklerin tanımlanmasında da kullanılabilir. [20]

4.3 IF … THEN ve Yüklem Mantığı Yapılarının Karşılaştırılması

Prolog da kurallar yazılırken kullanılan, baş ve gövde kısmını ayıran **':-'** sembolü **IF** anlamına gelir. Örneğin Pascal'daki IF komutu, öncelikle IF komutundan sonra gelen ifadenin doğru olup olmadığını kontrol eder. Eğer ifade doğrulanırsa, komut THEN ifadesinden sonraya geçer. [20]

Yani IF x>10 THEN writeln("Sayı ondan büyük");

satırında öncelikle x değişkeninin 10'dan büyük olup olmadığı kontrol edilir. Eğer sonuç doğru ise 'Sayı ondan büyük' satırı görüntülenir. Aksi takdirde program bir alt satırdan itibaren çalışmaya devam eder. Bu tip ifadeye "**IF … THEN"** şartı denir. Prolog ise bunun tam tersi olan bir sistem uygular. Öncelikle gövdedeki alt hedeflerin doğru olup olmadığına bakılır. Tamamı olumlu sonuç verirse, kuralın gövde kısmının doğruluğu ispatlanmış olur. Bu ise Prolog' da "**THEN … IF"** şartının geçerli olduğunu gösterir. [20]

5 UZMAN SİSTEMLER VE İLİŞKİSEL VERİTABANLARI

Genel amaçlı programlama dilleri ile yaratılan büyük veri tabanları' nın başarıyla kullanıldığı iyi bilinmektedir. Klasik veri tabanları çok büyük olabilir; çok sayıda kullanıcı tarafından paylaşılabilir. Veri tabanı uygun bir kaynak programla kolayca güncelleştirilebilir; veriye kolayca erişilebilir. Günümüzde çok gelişmiş olan ve gene de gelişmesini sürdüren bu sistemde sorgulama yapmak, yani mevcut bilgilere erişmek için, istenen işlevi yapacak kaynak programın yazılması zorunludur. Sorgulama eylemini yapacak kaynak program, ilgili veriye giden en kısa yolu izleyebilir; dolayısıyla sorgulama hızlıdır. Ancak sorgulamayı yapacak kaynak programın öngörmediği hiçbir bilgiyi bu sistemden elde etme olanağı yoktur. Sistemin kendi kendisine, mevcut bilgilere dayalı usavurmalar yaparak yeni bilgiler üretme yeteneği yoktur. Sistemden doğru bilgiyi almak için, kaynak program belirli önlemler koyabilir ve bu önlemlerin etkinliği ölçüsünde doğru verilere erişilir.

Mantıksal programlama sistemlerinde ise, sisteme önceden girilen veriler (dış veri tabanı) arasında var olan mantıksal ilişkiler, ayrı bir program yazımına gerek kalmadan, sistem tarafından keşfedilip kullanıcıya sunulmaktadır. Başka bir deyişle, yapay bir zekâ yaratılmaktadır. Bu zekâ, temel veri tabanındaki bilgilere dayalı usavurmalar yardımıyla yeni bilgiler üretebilmekte ve bir insan zekâsına benzer olarak, mantıksal sonuçlar çıkarabilmektedir. Veri tabanına yanlış bilgi girişi sistem tarafından önlenir. Sorgulama baştan sona ya da sondan başa kadar bütün bilgiler taranarak yapıldığı için, bilgilerin doğruluğu ve tamlığı kesindir. Doğal olarak, bu yetenek mantıksal programlama sistemlerinin üstünlüğünü ve önemini artırmaktadır.

Son yıllarda mantıksal programlama ile veri tabanlarını birleştirerek etkileşimli ya da bütünleşik yeni sistemler yaratmak için yapılan çalışmalara ilginin büyük ölçüde arttığı gözlenmektedir. Bir sorgulama dili olarak mantıksal programlama dilinin ne kadar etkili olduğu açıktır. Dolayısıyla iyi eşlenmiş bir mantıksal programlama ile bir veri tabanının oluşturacağı akıllı bir veri tabanının çekiciliği açıktır.

Bu sistemlerin her birisin de, mantıksal programlamanın işlevlerini yüklenen bir mantıksal programlama makinesi; kullanılacak veri tabanındaki ilişkileri tanıyacak bir mantıksal programlama arayüzü; veri tabanı ile aralarındaki iletişimi kuracak bir veritabanı arayüzü; veri tabanına erişimi ve güncellemeyi sağlayacak bir veritabanı makinesi vardır. Kuruluş açısından iki ana yapıya ayrılabilirler:

- Mantıksal Programlama ile veri tabanı iki ayrı alt sistemmiş gibi düşünülerek bu ikisi arasında etkileşimi sağlayacak bir arayüz yaratılır. Her iki sistem kendi varlıklarını bağımsız olarak sürdürmeye devam eder. Arayüz anabelleğe gerekli veri akışını sağlar. Prolog'a dayalı sistemler genellikle bu yöntemi kullanır.

- Büyük bir veri tabanı üzerine bir mantıksal programlama yerleştirilerek bütünleşik bir sistem yaratılır. Veri yapıları ve algoritmalar tamamen bu sistem için yeniden yaratılır.

Birinci tür sistemin kuruluşu daha kolaydır; ama ikinci tür sistemler daha etkindir.

Aşağıda ilişkisel veritabanlarıyla birlikte kullanılan uzman sistem örnekleri bulunmaktadır.

PRO-SQL: Yorktown Heights IBM Araştırma Merkezinde geliştirilmiştir. Prolog ile SQL/DS veritabanı sistemini eşlemektedir. Bu sistemde kullanıcının hem prologu hem de SQL dillerini bilmesi gerekmektedir.

EDUCE: Munich' de ki Avrupa Bilgisayar Endüstrisi Araştırma Merkezi tarafından geliştirilmiştir. Prolog ile Ingres veritabanı sistemini eşlemekte ve QUEL dili ile ilişki kurmaktadır.

ESTEAM: Prolog ile veritabanları arasında eşleme yapmak amacıyla Espirit projesi çerçevesinde yaratılmıştır.

BERMUDA: Wisconsin Üniversitesinde geliştirilmiş bir prototypedır. Prolog ile Britton-Lee Intelligent Database Machine IDM 500' ü eşlemektedir.

PRIMO: İtalya'daki Modena Üniversitesinde geliştirilmiştir. Arity Prolog ile ORACLE veritabanı sistemini eşlemektedir.

QUINTUS: Quintus Prolog ile Unify Database System arasında bir arayüzdür. Kaliforniya'da Quintus Computer System tarafından geliştirilmiştir.

5.1 Prolog ile SQL' in İlişkilendirilmesi

Visual Prolog ve piyasada bulunan diğer bir çok prolog sürümünde harici veritabanlarına (MS-SQL Server,ORACLE,FoxPro vb.) bağlantı sağlayabilmek amacıyla bir **ODBC (Open Database Connectivity/Açık Veritabanı Bağlanırlığı)** paketi bulunmaktadır. ODBC değişik veritabanı sistemlerine erişebilmek için geliştirilmiş bir arayüzdür. Örneğin, Oracle veritabanına bağlanan bir program ile ODBC sayesinde, bağlantı cümlelerinde ve sorgu cümlelerinde hiçbir değişiklik yapmadan MS-SQL Server veritabanıyla çalışılabilir. Bunun gerçekleşebilmesi için sisteme ODBC Sürücüleri (ODBC Drivers)' ni yüklemek yeterlidir. ODBC bu erişimler için tüm veritabanları için bir standart haline gelmiş olan **SQL (Structured Query Language/Yapısal Sorgulama Dili)'** i kullanır.

SQL herhangi bir veritabanı ortamında kullanılan bir alt dildir. SQL ile sadece veritabanı üzerinde işlem yapılabilir. SQL cümleleri ile veritabanına kayıt eklenebilir, var olan kayıtlar değiştirilebilir, silinebilir, güncellenebilir, sorgulanabilir ve bu kayıtlardan listeler oluşturulabilir.

Bir veritabanından veri almak gerektiğinde, bu isteği belirtmek için SQL dili kullanılır. DBMS (Database Management System), SQL isteğini işler, istenen veriyi elde eder ve bunu geri döndürür. Bir veritabanından veri isteyip sonucu alma işlemine, veritabanı sorgusu adı verilir. Yapısal Sorgu Dili (Structured Query Language) adı da buradan gelir.

SQL dilinin tarihçesinin başlangıcı, ilişkisel veritabanlarının gelişimiyle yakından ilgilidir. İlişkisel veritabanı kavramı ilk olarak bir IBM araştırmacısı olan Dr.E.F.Ted

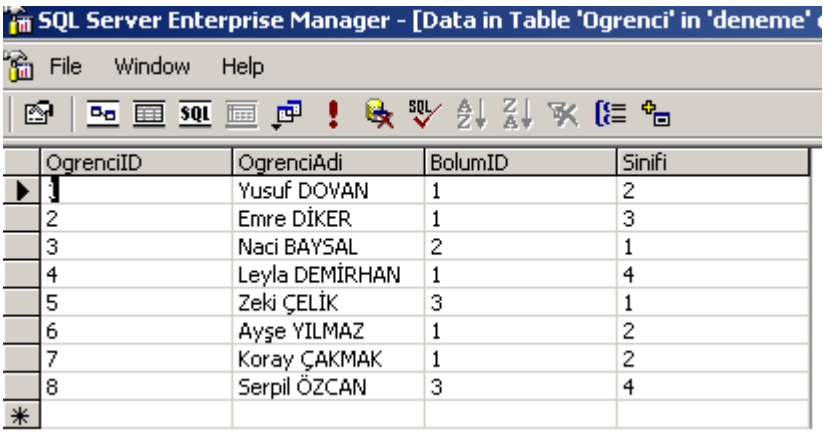
Codd tarafından geliştirilmiştir. Haziran 1970' te Dr. Codd " Paylaşılan Büyük Veri Bankaları İçin İlişkisel Bir Veri Modeli. " adlı makaleyi yayımlamıştır. Bu makale, verinin tablolu bir yapı kullanılarak nasıl depolanabileceği konusunda matematiksel bir teori ortaya koymuştur. İlişkisel veritabanları ve SQL' in tarihi bu makaleyle başlar.

SQL, ilişkisel veri tabanlarındaki bilgileri sorgulamak için kullanılan bir dildir. SQL, tüm kullanıcıların ve uygulamaların veri tabanına erişmek için kullandıkları komutlar bütünüdür. Uygulama programları ve veri tabanı araçları kullanıcılara çoğu durumda SQL kullanmadan veri tabanına erişim imkânını sunmaktadırlar fakat bu uygulamalarda geri planda SQL kullanmaktadırlar. SQL, ilişkisel veri tabanları ile uygulamaların diyaloğunu sağlamaktadır. SQL temelde verilerle mantıksal seviyede çalışmaktadır. Yani, bir tablodan birkaç kayıt seçebilmek için, o kayıtları seçebilecek bir koşul belirtilir. Koşula uyan tüm kayıtlar bir basamakta gelir ve bunlar kullanıcıya gösterilebildiği gibi, bir başka SQL'e veya uygulamaya da gönderilebilir. Kayıtların tek tek nasıl geldiği ve fiziksel olarak veri tabanının neresinde ve nasıl tutulduğu ile SQL ilgilenmektedir.

ODBC mimarisi 4 bileşenden oluşur:

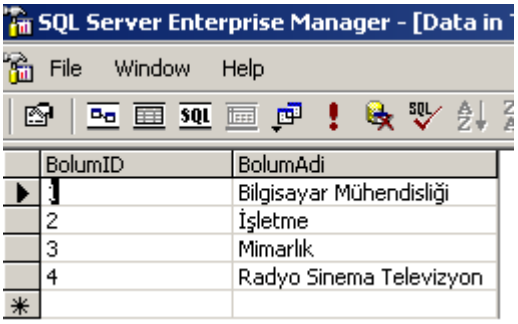
- Uygulama programı: Veritabanındaki verilere dayalı işlem yapmak üzere geliştirilmiş program.
- Sürücü yöneticisi: Uygulama programından gelen bağlantı tipine göre uygun veritabanı sürücüsünü aktif yapan modüldür.
- Sürücü: SQL isteklerini alarak veritabanına iletir ve sorgulama sonuçlarını uygulama programına geri döndürür.
- Veri Kaynağı: Uygulama programıyla erişilmek istenen veritabanı.

Visual prolog dan erişmek üzere MS-SQL Server da aşağıdaki tabloları oluşturalım.



OgrenciID	OgrenciAdi	BolumID	Sinifi
1	Yusuf DOVAN	1	2
2	Emre DİKER	1	3
3	Naci BAYSAL	2	1
4	Leyla DEMİRHAN	1	4
5	Zeki ÇELİK	3	1
6	Ayşe YILMAZ	1	2
7	Koray ÇAKMAK	1	2
8	Serpil ÖZCAN	3	4

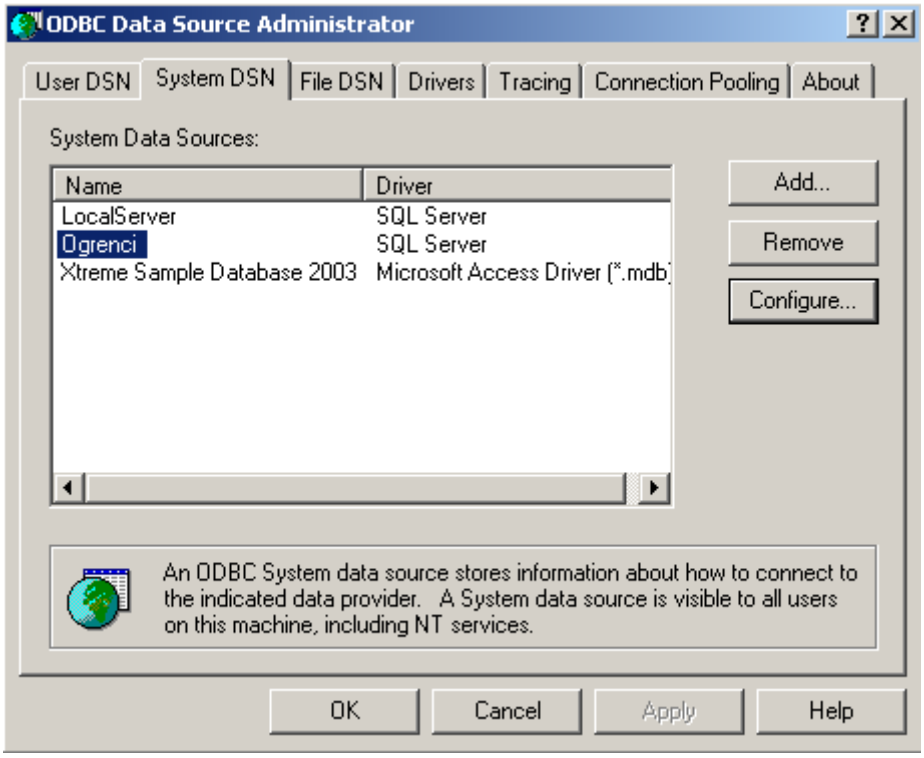
Şekil 5.1. Öğrenci bilgilerinin tutulduğu **"Ogrenci"** tablosu



BolumID	BolumAdi
1	Bilgisayar Mühendisliği
2	İşletme
3	Mimarlık
4	Radyo Sinema Televizyon

Şekil 5.2. Bölüm bilgilerinin tutulduğu **"Bolumler"** tablosu

Yukarıdaki tabloları oluşturduktan sonra Visual Prolog dan veritabanına bağlanabilmek için Windows platformunda Yönetimsel Araçlarda bulunan ODBC Veri Kaynağı Yöneticisi ekranından gerekli bağlantı ayarlarının yapılması gerekmektedir.

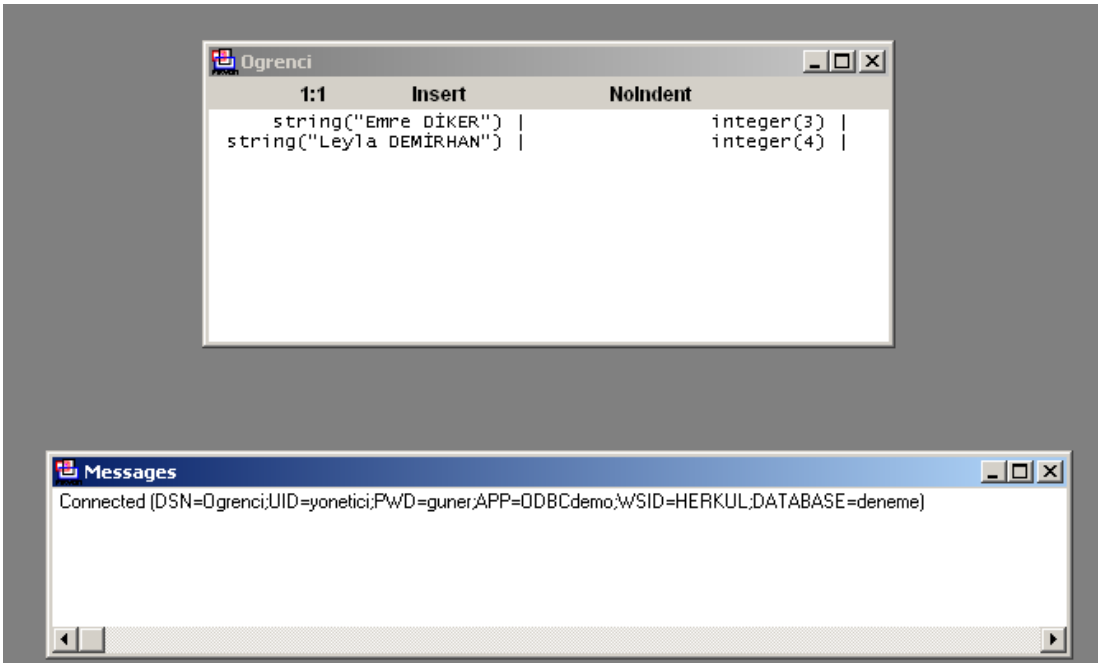


Şekil 5.3. ODBC Veri Kaynağı Yöneticisi

Tüm bu ayarlar yapıldıktan sonra Visual Prolog da gerekli bağlantı ve sorgu cümlelerini yazarak veritabanına erişip sorgu sonuçlarını listeleyebiliriz.

```
class facts
   baglanti : (odbcConnection Baglanti)
clauses
  BaglantiKur() = Baglanti :-
     Baglanti = odbcConnection::new(), assert(baglanti(Baglanti)).
     Listele() = Stmt :- Baglanti = BaglantiKur (),
     Stmt = odbcStatement::new(Baglanti),
     StmtString=string::format("SELECT Ogrenci.OgrenciAdi, Ogrenci.Sinifi
       FROM INNER JOIN Bolumler ON Ogrenci.BolumID=Bolumler.BolumID
       WHERE (Bolumler.BolumAdi = 'Bilgisayar Mühendisliği")
       AND Ogrenci.Sinifi>2 ,
     Stmt:execDirect(StmtString).
```

Yukarıda veritabanına bağlantı kurmak için kullanılan ve SQL sorgusu gönderilen kod bölümleri verilmiştir. Bu sorguyla programdan elde edilen ekran görüntüsü aşağıdadır.



Şekil 5.4 Sorgu çalıştırıldıktan sonra elde edilen ekran görüntüsü

6 YASAL DÜZENLEME METİNLERİNİN UZMAN SİSTEM BİÇİMİNE DÖNÜŞTÜRÜLMESİ

Önerme mantığı kurallarına göre herhangi bir cümle, büyük bir harfle temsil edilebilir. Bir basit cümle özne ve yüklemden oluşan herhangi bir cümledir. Örneğin; "Ayşe iyi bir öğrencidir.", "Ali yeni bir radyo aldı."..vb.

"Mehmet burslu bir öğrencidir." cümlesi "M" harfi ile sembolize edilebilir. Mantıksal değişkenleri düşünüldüğünde, bu büyük harf iki mantıksal değerden birini varsayar: 1 veya 0. Eğer bu cümle doğru olarak düşünülürse, M'nin mantıksal değeri birdir; eğer yanlışsa mantıksal değer sıfırdır. M cümlesi prolog için bir gerçek ifade eder ve bu gerçek *Burslu(Mehmet)* şeklinde prologa tanıtılır. Bu ifade prologa verildikten sonra, *?- Burslu(X)* şeklinde bir sorgulama yapılırsa X in alacağı değerler içerisinde Mehmet de olacaktır.

Önerme mantığı kurallarına göre, altı tane mantıksal operatör vardır: "NOT, AND, OR, XOR, Conditional (Koşullu), Biconditional (Birden fazla koşullu) operatörlerdir. Bu operatörler sırasıyla şu sembollerle gösterilir: **" ~ "** , **" . "** , **" v "** , **" ^ "** , **" ⊃ "** ve **" ≡ "**. "Mehmet burslu bir öğrenci değildir." cümlesi *"~M"* şeklinde gösterilebilir. Tablo 6.1 bu bileşik cümlenin doğruluk tablosunu tanımlar.

Tablo 6.1 ~M cebirsel ifadesinin doğruluk tablosu

M	**~M**
1	0
0	1

Bir bileşik cümle, en az bir basit cümle ve mantıksal operatör NOT veya en az bir mantıksal operatör tarafından oluşturulmuş iki basit cümleden oluşur. Örneğin, "Öğrencilerin yeni eğitim öğretim döneminde derslere devam edebilmeleri için ders

seçme dilekçelerini doldurmuş olmaları ve eğitim ücretini ödemiş olmaları gerekmektedir.” yönetmelik maddesi, *“D.M”* ile biçimlendirilir. D harfi “Ders seçme dilekçesi doldurulmuş olmalıdır” cümlesini, M harfi ise “Eğitim ücretini ödemiş olmalıdır” cümlesini sembolize eder. “**.**” bu işarette AND mantıksal operatörünün karşılığıdır. D**.**M mantıksal ifadesinin doğruluk tablosu Tablo 6.2’ de gösterilmiştir.

Tablo 6.2 D**.**M mantıksal ifadesinin doğruluk tablosu

D	**M**	**D . M**
1	1	1
0	1	0
1	0	0
0	0	0

D.M yönetmelik maddesi prolog için bir kural ifade etmektedir. Bu kural iki gerçeğin doğruluk değerlerine göre 1 veya 0 değeri almaktadır. Bu gerçeklerden birisi öğrencilerin ders seçme dilekçesi doldurmuş olmaları diğeri ise eğitim ücretlerini ödemiş olmalarıdır. Bu kural prologda ;

Derslere_Devam_Edebilir(X):-Dilekce_Doldurdu(X),Odeme_Yapti(X). şeklinde ifade edilir.

Aynı şekilde, “Öğrencilerin birinci sınıf derslerini alabilmeleri için, her eğitim yılı başında yapılan yabancı dil muafiyet sınavında başarılı olmaları veya bir yıl süreyle yabancı dil hazırlık okumaları gerekmektedir.” yönetmelik maddesinin mantıksal ifadesi *“MvH”* şeklindedir. M harfi “Yabancı dil muafiyet sınavında başarılı olmalıdırlar” cümlesini “v” mantıksal operatör olan OR’ u H harfi ise “Yabancı dil hazırlık okumalıdırlar” cümlesini sembolize eder. MvH mantıksal ifadesinin doğruluk tablosu Tablo 6.3’ de gösterilmiştir.

Tablo 6.3 MvH cebirsel ifadesinin doğruluk tablosu

M	**H**	**M v H**
1	1	1
0	1	1
1	0	1
0	0	0

MvH yönetmelik maddesi prologda;

Birinci_Sinifa_Gecebilir(X):-Muafiyet_Sinavinda_Basarili(X);Hazirlik_Okudu(X). şeklinde ifade edilir.

OR bağlacı yalnızca düzenli cümlelerde kullanılabilir. Örneğin yönetmelikte belirtilen "Öğrenciler birinci sınıf güzel sanatlar dalı seçmeli derslerinden Fotoğraf veya Müzik derslerinden sadece birisini almalıdırlar." cümlesi OR bağlacı kullanılarak v ile sembolize edilemez. Bu cümle ancak *"F^M"* mantıksal ifadesiyle sembolize edilebilir. F, "Öğrenciler fotoğraf dersi almalıdırlar" cümlesini, "^" mantıksal operator XOR' u ve M ise "Öğrenciler Müzik dersi almalıdırlar" cümlesini sembolize eder. Tablo 6.4 bu cümlenin doğruluk tablosunu göstermektedir.

Tablo 6.4 F^M mantıksal ifadesinin doğruluk tablosu

F	**M**	**F ^ M**
1	1	0
0	1	1
1	0	1
0	0	0

“Eğer öğrencinin birinci sınıftan başarısız dersi bulunuyorsa ikinci sınıftan ders alamaz” cümlesi ise *“$B \supset A$”* mantıksal ifadesi ile gösterilir. B, “Birinci sınıftan başarısız dersi bulunuyorsa” cümlesini, “⊃“ IF … THEN mantıksal operatörünü, A “İkinci sınıftan ders alamaz” cümlesini sembolize etmektedir. Tablo 6.5 bu ifadenin doğruluk tablosudur.

Tablo 6.5 B ⊃ A mantıksal ifadesinin doğruluk tablosu

B	**B ⊃ A**
1	1
0	0

Son olarak “Öğrencilerin yeni eğitim öğretim yılında derslere devam edebilmeleri için, her yıl belirlenen tarihler arasında mali yükümlülüklerini yerine getirerek kayıt yenilemiş olmaları gerekmektedir. Burslu statüde olan öğrenciler bu uygulamanın dışındadırlar” maddesinin mantıksal ifadesi *“ $E \equiv M \vee (B)$ ”* şeklindedir. E, “Öğrenciler yeni eğitim öğretim yılında derslere devam edebilirler” cümlesini, “≡” IF AND ONLY IF (Ancak ve ancak), M “Mali yükümlülüklerin yerine getirilmesi” cümlesini ve B, burslu öğrencileri ifade etmektedir.

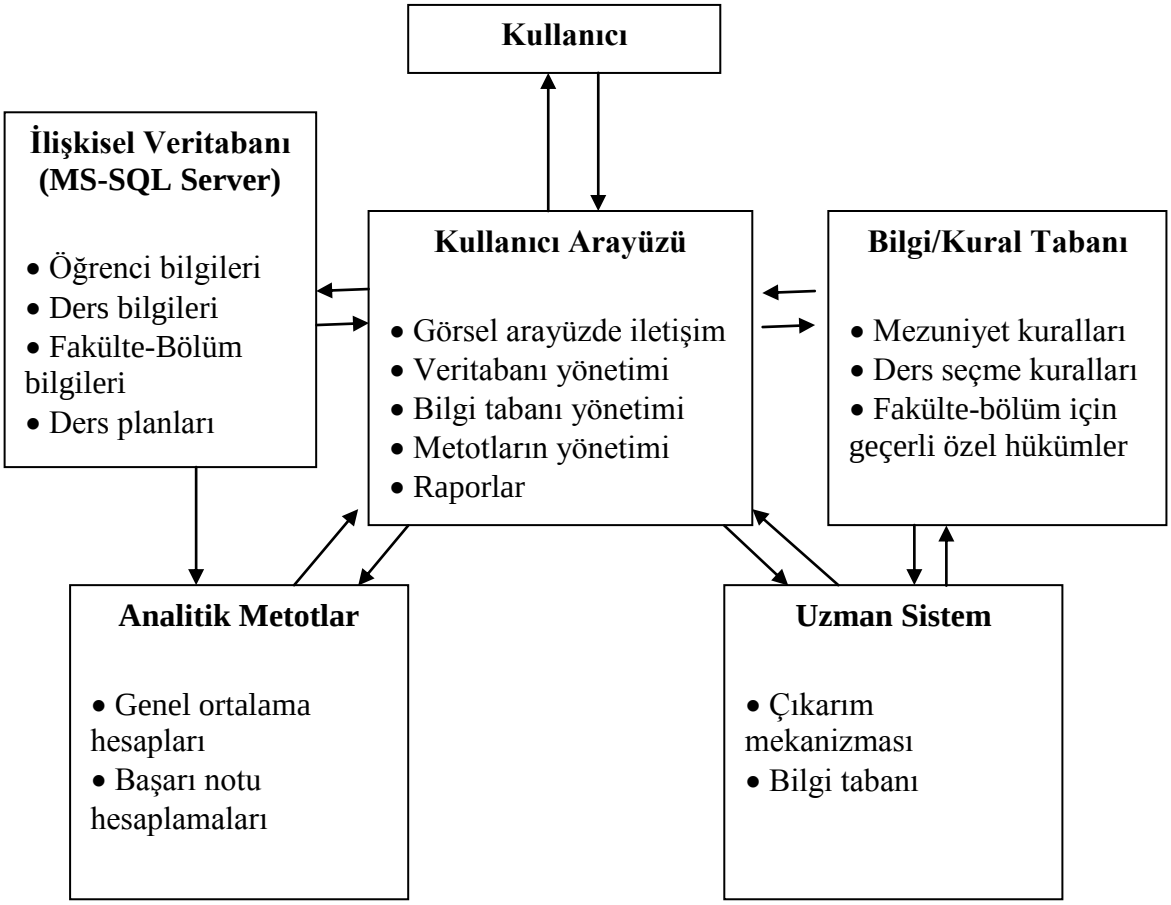
Tablo 6.6 E ≡ M v (B) mantıksal ifadesinin doğruluk tablosu

M	(B)	E ≡ M v (B)
1	1	1
1	0	1
0	0	0
0	1	1

6.1 İlişkisel Veritabanıyla Birlikte Çalışan Bir Uzman Sistem Modeli

Genelde tüm işlemlerin kağıt üzerinde yapıldığı üniversitelerde, verilerin kağıt ortamından elektronik ortama aktarılmaları zor ve masraflı olmaktadır. Bununla birlikte, yönetmelik maddeleri de sürekli olarak değişebilmektedir.

Bu sebeple bilgisayar destekli öğrenci kayıt işlemlerinde farklı yaklaşımlar kullanılarak, uzman sistemlerle ilişkisel veritabanı sistemlerinin birleştirilmesi ve kural esnekliğini içinde barındıran bir uzman sistem geliştirilmesi hedeflenmiştir.



Şekil 6.1 İlişkisel veritabanı sistemi ve Uzman sistem

Şekil 6.1’ de temel olarak iki kısım bulunmaktadır. Bunlardan şeklin sol yanında yer alan kısım aslında geleneksel bir ilişkisel veritabanı sistemi yapısıdır. Şeklin sağındaki kısım ise bir uzman sistem yapısıdır. Şekil 6.1’ in sol yanında yer alan ilişkisel veritabanı sisteminde öğrencilere, derslere, fakülte ve bölümlere ait tüm bilgiler bir veri tabanında tutulmaktadır. Bu çalışmada MS-SQL Server veri tabanı kullanılmaktadır. Bu bilgiler öğrencilerin kişisel bilgileri, öğrenim süreleri boyunca kazanılmış gerekli bilgiler, derslerle ilgili bilgiler ve fakültelerin oluşturdukları ders planlarının bilgileridir. Analitik Metotlar olarak ifade edilen bileşende ise ortalama ve başarı notu hesaplamalarını yapan modüller yer almaktadır. Bu modüller Visual Basic 6.0 da hazırlanmıştır. Şekil 6.1’ in sağ tarafında yer alan uzman sistem kısmında ise mezuniyet, ders seçme ve diğer özel hükümleri kapsayan kurallardan oluşan bilgi tabanı ve bir çıkarım mekanizması yer almaktadır.

Şekil 6.1’deki bu iki yapının birlikte çalıştırılması kullanıcı arayüzü tarafından sağlanmaktadır. Uzman sistem çalıştırıldığında, arayüz vasıtasıyla kullanıcıdan öğrenci bilgileri, ders bilgileri, ders talepleri, ortalama bilgileri gibi parametre değerlerinin girilmesi istenmektedir. Bu sorulara kullanıcının doğru ve kesin cevaplar verebilmesi için veritabanından bu bilgileri sorgulayabilecek bir başka sisteme ihtiyaç duyulmaktadır. Bu amaçla ilgili analitik metotlar kullanılarak sonuçlar alınmakta daha sonra bu sonuçlar doğrultusunda uzman sistem tarafından sorulan sorulara cevaplar verilmektedir. Uzman sistem tarafından kullanıcıya yöneltilen tüm sorular cevaplandıktan sonra, uzman sistem kullanıcıya seçilebilecek derslerin listelendiği önerilerde bulunmaktadır.

Konuyla ilgili olan uzmanlar, öğrenci ve derslerle ilgili tam bilgileri, özel istek ve uzun vadeli planları ve gelecek dönem için öğrenci programlarını ilişkisel veritabanı sistemine kaydederler.

Sistem için gerekli olan teknik bilgi uzmanların kullandığı bazı kurallar çerçevesinde bir bütün haline getirilmiştir. Bu kurallar başarı şartlarının yerine getirilmesi için istenilen özellikleri ve bunun ne kadar zaman alacağını içermektedir.

Sistemin birinci hedefi veriyi değil bilgiyi işlemektir. Sistem alan uzmanlarının belirli kurallara bağlı olarak, ilişkisel veritabanından alınan bilgileri göz önünde bulundurarak öğrencilere tavsiyelerde bulunmaktadır. İkinci olarak sistemin, gerektiğinde yeni kurallar eklemek veya değiştirmek amacıyla yeterli derece de esnekliğe sahip olması hedeflenmiştir.

Burada uzmanlık sürecinin asıl hedefi, bazı fakülte ve belirli bölümler tarafından belirlenen mezuniyet şartlarının sağlandığının kontrol edilmesidir. Bu hedef üniversite, fakülte ve bölümle ilişkili olan amaçların birleştirilmesini kapsamaktadır.

Üniversite yönetmeliğindeki kurallar, öncelikle önerme mantığı esasları kullanılarak biçimlendirilmiş, daha sonra bunlar prolog cümlelerine çevrilmişlerdir.

Ayrı olarak çalışan veritabanı yönetim sistemi kayıtlı her öğrencinin niteliksel ve niceliksel bilgisini sürdürür. Prolog sürücü programı her öğrenci için bu kayıtlara erişebilir.

6.1.1 Yönetmelik Maddelerinin Prolog Cümleleriyle İfade Edilmesi

Maltepe Üniversitesi eğitim-öğretim ve sınav yönetmeliği, üniversiteye bağlı fakülte ve yüksekokullarda kayıt, eğitim-öğretim ile sınavlarda uygulanacak genel esasları düzenlemektedir.

Yönetmeliğin 6. maddesi öğrencilerin eğitim-öğretimlerine başlama ve devam etmelerini, "Öğrencilerin yeni eğitim öğretim döneminde derslere devam edebilmeleri için ders seçme dilekçelerini doldurmuş olmaları ve eğitim ücretini ödemiş olmaları gerekmektedir." kuralına bağlamıştır. Bu durumda öğrencilerin iki şartı yerine getirmiş olmaları gerekmektedir. Bunlardan birisi ders seçme dilekçelerini doldurmuş olmaları, diğeri ise eğitim ücretlerini ödemiş olmaları. Bu şartlar prolog için gerçekleri ifade etmekte ve her bir üye için tanımlanmış bilgileri ifade etmektedir. Ders seçme dilekçesinin doldurulması gerçeği prologda; *Ders_Secme_Dilekcesi_Doldurma(ogrenci).* şeklinde ve eğitim ücretinin ödenmesi

gerçeği ise, *Harc_Yatirma(ogrenci).* şeklinde ifade edilebilir. Yönetmeliğin 6. maddesi bu iki gerçeği içermek koşuluyla prolog için bir kural olarak tanımlanacak olursa aşağıdaki cümle oluşturulabilir.

Egitim_Hakki_Kazanma(ogrenci):-
Ders_Secme_Dilekcesi_Doldurma(ogrenci),Harc_Yatirma(ogrenci)

Yönetmeliğin 7. maddesi mali yükümlülüklerle ilgili detayları belirtmektedir. Buna göre burslu öğrencilerin mali yükümlülükleri bulunmamaktadır. Öğrencilerin burslu olduklarını belirten prolog gerçeğini *Burslu(ogrenci)* şeklinde ifade edersek, yukarıdaki *Egitim_Hakki_Kazanma* kuralını aşağıdaki şekliyle değiştirebiliriz.

Egitim_Hakki_Kazanma(ogrenci):-Ders_Secme_Dilekcesi_Doldurma(ogrenci),

(Harc_Yatirma(ogrenci);Burslu(ogrenci)).

Böylece bir öğrencinin eğitim hakkı kazanabilmesi; dilekçe doldurması ve burslu olmayan bir öğrenci ise harç yatırmış olması şartlarına bağlanarak yukarıdaki kuralla kontrol edilebilir.

Eğitim hakkı kazanan öğrencinin ders seçme dilekçesiyle seçmiş olduğu dersler proloğa aşağıdaki cümlelerle tanıtılabilir.

Derse_Yazilma(ogrenci,ders).

Yönetmeliğin 4. bölümü; dersler, ara sınavlar, yarıyıl/yıl sonu sınavları, bütünleme ve değerlendirme ile ilgili esasları kapsamaktadır. Yönetmeliğe göre ders geçme notu 50 olarak belirlenmiştir. Öğrencilerin bir dersten başarılı olabilmeleri için o ders için dönem içinde katıldıkları sınavlardan elde ettikleri ortalamanın 50 ve üzeri olması gerekmektedir. Öğrencinin aldığı derslere ilişkin notları prolog gerçekleri olarak aşağıdaki gibi belirtebiliriz;

Basari_Notu(ogrenci,ders,ortalama).

Bu durumda bir dersten başarılı olma durumunu kontrol eden prolog kuralı aşağıdaki gibi olur.

Dersten_Basarili(ogrenci,ders):-Basari_Notu(ogrenci,ders,ortalama),ortalama>=50

Eğitim-öğretim dili ve yabancı dil esaslarını düzenleyen 17. maddeye göre öğrencilerin birinci sınıfa başlayabilmeleri için, kayıt oldukları dönem başında yapılan yabancı dil muafiyet sınavında başarılı olmaları veya bir yıl süreyle yabancı dil hazırlık okumaları gerekmektedir. Bunu prolog ifadesine çevirecek olursak;

Birinci_Sinifa_Gecme(ogrenci):-
Derse_Yazilma(ogrenci,hazirlik);Dersten_Basarili(ogrenci,haz_muafiyet).

Son olarak, ilişik kesme esaslarını düzenleyen madde 7 ye göre öğrencilerin aşağıdaki durumlarda okulla ilişikleri kesilmektedir:

- Yükseköğretim kurumundan çıkarılma cezası almış olması,
- Kaydının silinmesini yazılı olarak istemiş olması.
- 6. madde de belirtilen eğitim-öğretime hak kazanma şartlarını yerine getirmemiş olması.

Madde 7 prolog kuralı olarak ifade edilecek olursa;

İlisik_Kesme(ogrenci):-Cikarilma_Cezasi(ogrenci);Kayit_Silme_Dilekcesi(ogrenci);
NOT(Egitim_Hakki_Kazanma(ogrenci)). şeklinde olacaktır.

6.1.2 Örnek Sorgulamalar

Ders_Secme_Dilekcesi_Doldurma(ahmet).
Ders_Secme_Dilekcesi_Doldurma(ayse).
Ders_Secme_Dilekcesi_Doldurma(ali).
Harc_Yatirma(ahmet).

Burslu(ayse).
Derse_Yazilma(ahmet,bil151).
Derse_Yazilma(ayse,hazirlik).
Basari_Notu(ahmet,bil151,70).

Yukarıda ki gerçekler proloğa aktarıldıktan sonra aşağıdaki sorgulamalar ve sonuçları elde edilecektir.

?- Egitim_Hakki_Kazanma(ogrenci).
ogrenci=ahmet;
ogrenci=ayse.
?- Derse_Yazilma(ogrenci,ders).
ogrenci=ahmet
ders=bil151;
ogrenci=ayse
ders=hazirlik.
?- Derse_Yazilma(ogrenci,bil151).
ogrenci=Ahmet.
?- Dersten_Basarili(ogrenci,bil151).
ogrenci=Ahmet.

?- Birinci_Sinifa_Gecme(ogrenci).
ogrenci=ayse.
?- İlisik_Kesme(ogrenci).
ogrenci=ahmet.

SONUÇ

Bilginin elde edilmesi ve bu bilgiye doğru zamanda kolay bir şekilde erişebilmek önemlidir. Bu bilgiye sahip olan konuyla ilgili insan uzmanlara, istenildiği zaman kolay bir şekilde erişebilmek her zaman mümkün olmamaktadır. Böyle durumlarda uzman sistemler, bilgiyi saklayarak bunu kullanmak isteyen herkesin istifadesine sunarlar. Uzman sistemler, içerdikleri ana kurallarla çıkarımlar yaparak mantığa uygun sonuçlar üretirler.

Uzman sistemler günümüzde birçok alanda kullanılmakta ve farklı sektörler için yeni uzman sistemler geliştirilmektedir.

Uzman sistemlerin geliştirilmesinde kullanılan mantık dillerinden en popüler olanı prologdur. Önceleri DOS işletim sistemi üzerinde çalışan prolog, Windows işletim sistemine de aktarılarak Visual Prolog versiyonu olarak son halini almıştır. Bu gelişmelerle birlikte, uzman sistemlerin ilişkisel veritabanları ve diğer uygulama geliştirme programlarıyla birlikte kullanılması çalışmaları da giderek artmaktadır.

Klasik veritabanı sistemleri ile mantıksal programlama sistemlerinin birleştirilerek her iki sistemin iyi niteliklerini taşıyan kullanışlı bütünleşik sistemlerin yaratılmasıyla çok sayıda bilgi içeren büyük veritabanlarından kaynak programın önceden öngörmediği mantıksal ilişkileri, hızlı ve kolay bir şekilde elde etme olanağı doğacaktır.

Bu çalışmada uzman sistemlerin yapısı ve uzman sistem geliştirme dillerinden olan prolog' un yapısı araştırılmış, prolog' un ilişkisel veritabanları ile birlikte kullanımı ve yasal düzenleme metinlerinin uzman sistem biçimine dönüştürülmesine ilişkin örnekler sunulmuştur.

KAYNAKLAR

1. ANONYMOUS, 2003a, "**Yapay Zekâ Nedir**",
2. TEKTAŞ, M., AKBAŞ, A., TOPUZ, V., 2003, "**Yapay Zekâ Tekniklerinin Trafik Kontrolünde Kullanılması Üzerine Bir İnceleme**" , www.trafik.gov.tr/icerik/bildiriler/C4-7.doc
3. http://www.bahcesehir.k.12.tr/webyarisma/2003/files/WT2003-87/yapay_zekâ.htm
4. http://www.bahcesehir.k.12.tr/webyarisma/2003/files/WT2003-87/gecmis.htm
5. ADACIK, M., 2002, " **Yapay Zekâ**", http://user1.7host.com/mehmetadacik/bilgibir/yzekâ.asp
6. ADACIK, M., 2004, " **Yapay Zekâ**", www.yapay-zekâ.org
7. SAĞIROĞLU, Ş., BEŞDOK, E., ERLER, M., "**Mühendislikte Yapay Zekâ Uygulamaları-1 Yapay Sinir Ağları**", Ufuk Yayıncılık; Kayseri, 2003
8. KAYA, İ., GÖZEN, Ş., ENGİN O., "**Kalite Kontrol Problemlerinin Çözümünde Uzman Sistemlerin Kullanımı**", Havacılık ve Uzay Teknolojileri Dergisi, 2004
9. Kurt, A., **"Uzman Sistem Nedir?"**, Gazi Üniversitesi Fen Bilimleri Enstitüsü Bülteni, Cilt:8, Sayı:3, 5-7, 1995.
10. Türker, E.S. ve Taşkın, H., **"Endüstriyel Sistemlerde Yapay Zekâ ve Uzman Sistemler Uygulamaları"**, Endüstri Mühendisliği Dergisi, Yıl:3, Sayı:14, 1991.
11. Öz, E., "**Tedarikçi Seçimi Problemine Karar Teorisi Destekli Uzman Sistem Yaklaşımı"**, Gazi Üniv. Müh. Mim. Fak. Dergisi, Cilt:19, No:3, 275-286, 2004
12. ANONYMOUS, 2003e, "**Yapay Zekâ**", www.gurayim.com/yapayzekâ/birinci_bölüm.htm
13. ANONYMOUS, 2003d, http://www.baskent.edu.tr/~eraslan/yapayzekâ.doc
14. Tatlı, E., **"Uzman Sistemler", 2000**
15. Aydırn Y.S, **"Visual Prolog ile Programlama"**, Sistem Yayıncılık, 2000
16. KUŞ, M., VAROL, A., OĞUROL, Y., "**Uzman Sistemin Dişçilik Alanında Kullanımına Ait Bir Uygulama"**, Endüstri&Otomasyon, Aylık Elektrik, Elektronik, Makine, Bilgisayar ve Kontrol Sistemleri Dergisi, Sayı: 18, 1998

17. Karaçay T., **“Akıllı Veri Tabanları Yaratmaya Doğru”**, IBM Araştırma ve Geliştirme Merkezi, Ankara
18. http://www.baskent.edu.tr/~eraslan/yapayzekâ.doc
19. http://www.geocities.com/akadirbali/yapayzekâ
20. Aydın Y.S, **“PROLOG PROGRAMLAMA DİLİ İLE MAKİNE MÜHENDİSLİĞİ ALANINDA UZMAN SİSTEMLERİN HAZIRLANMASI TEKNİKLERİ”**, Yüksek Lisans Tezi, 1998