İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

XML TABANLI UYGULAMALARIN YAPAY ZEKA MODELİYLE GERÇEK ZAMANLI ANALİZİ

YÜKSEK LİSANS TEZİ

Müh. Muhsin GEMİCİ

Anabilim Dalı: ENDÜSTRİ MÜHENDİSLİĞİ

Programı: MÜHENDİSLİK YÖNETİMİ

OCAK 2007

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

XML TABANLI UYGULAMALARIN YAPAY ZEKA MODELİYLE GERÇEK ZAMANLI ANALİZİ

YÜKSEK LİSANS TEZİ

Müh. Muhsin GEMİCİ

(507001303)

Tezin Enstitüye Verildiği Tarih : 25 Aralık 2006

Tezin Savunulduğu Tarih : 24 Ocak 2007

Tez Danışmanı : **Doç. Dr. Cengiz GÜNGÖR**

Diğer Jüri Üyeleri **Prof. Dr. Demet BAYRAKTAR (İ.T.Ü.)**

Doç. Dr. Y.İlker TOPÇU (İ.T.Ü.)

OCAK 2007

ÖNSÖZ

Sürekli teşvikiyle bu tezin tamamlanmasına değerli katkılar sağlayan tez danışmanım Doç. Dr. Cengiz GÜNGÖR'e, çalışmalarım sırasında maddi ve manevi desteklerini esirgemeyen aileme, katkıları ve tavsiyeleri ile bu tezin değer yaratan yenilikçi bir ürüne dönüşmesini sağlayan çalışma arkadaşlarıma teşekkürlerimi sunarım.

Ocak 2007 Muhsin Gemici

İÇİNDEKİLER

KISALTMALAR

YSA : Yapay Sinir Ağları
PSTN : Public Switched Telephone Network
MPLS : Multiprotocol Label Switching
VRF : Virtual Routing Forwarding Instance
PVS : Periyodik Veritabanı Sorgulama
MYE : Merkez Ofis Yazılımlarına Eklenti
XML : eXtensible Markup Language

TABLO LİSTESİ

ŞEKİL LİSTESİ

ÖZET

Bu çalışmada, bir akaryakıt dağıtım şirketinin, akaryakıt istasyonlarıyla oluşturduğu özel bayi ağına 7/24 hizmet veren kritik sistemlerini, yenilikçi bir yöntem ile analiz edecek, altyapı arızalarını yada satış sırasında çalıntı kredi kartı kullanımı gibi iş süreçlerinin doğal sonucu olarak oluşan, kurum tarafından gecikmeden anlaşılması ve müdahale edilmesi gereken sorunları zamanında tespit edecek, problemleri analiz ederek sonuçları ilgili birim yada çağrı sistemlerine iletecek bir yazılım ürünü oluşturulmuştur.

Problemin çözümü için, ilk etapta merkez uygulama sunucularına gelen ve sunuculardan istasyonlara giden verileri yenilikçi bir yöntem ile elde edecek, XML Dinleyici adı verilen bir yazılım geliştirilmiştir. Şimdiye kadar ağ izleme, trafik analizi yada saldırı tespit sistemlerinde kullanılmış yöntem, bu kez bir hat üzerinden geçen XML tabanlı verileri dinlemek ve yakalamak amacıyla kullanılmıştır.

İkinci etapta, bir bulanık yapay nöron modeli oluşturularak bir Yapay Sinir Ağı(YSA) içerisinde, birinci etapta elde edilen verileri analiz ederek bilgiye dönüştüren Akıllı Ajan adı verilen yazılım geliştirilmiştir.

Çalışmanın sonucunda, kurum ihtiyaçlarının karşılanmasıyla beraber, merkez sunucular üzerinden tespit edilemeyen bir takım sorunlar da keşfedilmiştir.

Uygulama C# programlama dili kullanılarak oluşturulmuştur.

ABSTRACT

In this thesis, a software product developed which use innovative approach to analyze critical application systems which gives 7/24 service to sellers' network that consist of thousand fuel stations, in a petrol distributor company.

The problem is, natural result of some work process at application level like "stolen credit card usage" or communication/automation infrastructure faults and others have to be known immediately by company to interfere to the problem on time. This solution analyzes XML transactions and produces meaningful result and delivers them to the outsourced call center systems, technical experts of the company, sellers, company customers or related company unit.

First of all, Intelligent Sniffer software is developed which innovatively collects and combines incoming/outgoing transactions from the central application servers to realize solution. Although, this network sniffing approach is used in network monitoring, network traffic analyzing or network intrusion detection till now, at his time it is used to get XML based traffics by listening communication line.

Secondly, a fuzzy neuron model is designed to constitute an Artificial Neural Network that aims to analyze collected transactions and transforms them to the form of knowledge.

At the end of this thesis project, not just company requirements is fitted, some undiscovered problems also come into sight.

Application is developed with C# programming language on .NET environment.

1. GİRİŞ

Bu bölüm de, tez çalışmasının yapıldığı akaryakıt dağıtım şirketinin merkez ofis uygulaması tanıtılacak, çözümü hedeflenen problemler açıklanacak ve çözüm alternatifleri belirtilecektir.

İkinci bölüm, tezin dayandığı bilimsel ve teknolojik temellerin bulunduğu, Yapay Sinir Ağları(YSA), biyolojik ve yapay nöronlar, YSA'nın tarihsel gelişimi, YSA ile üretilen çözümler, bulanık mantık, dilsel değişkenler, XML iletişimi ve TCPI/IP konularının literatür araştırmalarını içermektedir.

Üçüncü bölümde, bu tez ile oluşturulan çözüme alternatif olabilecek yöntemler tanımlanmıştır.

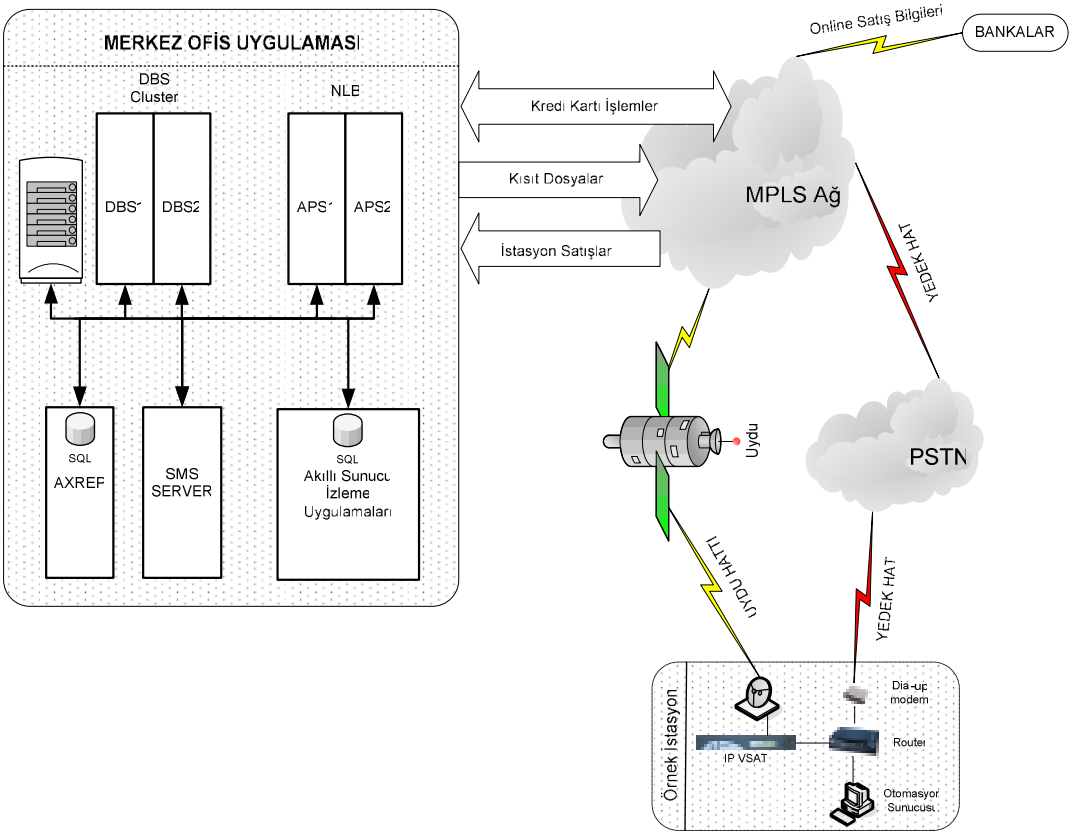
Dördüncü bölümde, tez ile tasarımı gerçekleştirilen Yapay Zeka Modeli açıklanmış, bulanık nöron tasarımı gerçekleştirilmiş, çalışma şekli açıklanmış ve YSA Modeli oluşturulmuştur.

Beşinci bölümde, akıllı sunucu izleme yapısı'nın içerdiği uygulamaların özellikleri, tasarım modelleri ve akış diyagramları bulunmaktadır.

Altıncı bölümde, 7 Aralık 2006 tarihi itibariyle devreye alınan akıllı sunucu izleme Yapısının 23 Aralık 2006 tarihine kadar tespit ettiği bulgular incelenmiş, çözümü hedeflenen problemlerin tespit edilebildiği gösterilmiştir.

1.1 Merkez Ofis Uygulaması

Merkez ofis uygulaması, kredi kartının fiziksel olarak bulunmadığı, ancak kredi kartı ile eşleştirilmiş bir RFID kartın kullanıldığı, yüksek güvenlik altyapısına sahip bir akaryakıt alışveriş altyapısıdır. İstasyon → akaryakıt dağıtım şirketi → banka arasında uçtan uca otomasyon altyapısına sahip, iletişimin uydu ağı üzerinden gerçekleştiği, iletilen verinin işlemin her aşamasında şifrelendiği bu son derece güvenli sistem, işlem boyunca herhangi bir insan müdahalesi olmaksızın çalışmaktadır (Şekil 1.1.1).



Şekil 1.1.1: Merkez Ofis Uygulama Altyapısı

Şekil 1.1.1 de görüleceği üzere, akaryakıt istasyonlarında ki satış bilgileriyle beraber, merkez ofise istasyonlarda ki akaryakıtın fiziki olarak depolandığı tanklardan da veri alınmaktadır. Bu bilgiler özet olarak, tank içerisindeki akaryakıt tipi(benzin, mazot, LNG vs..), dibe çökmüş su seviyesi, ortam sıcaklığı, nem oranı, seviye bilgisi gibi tank durumunu belirten detaylı ve anlık verilerdir.

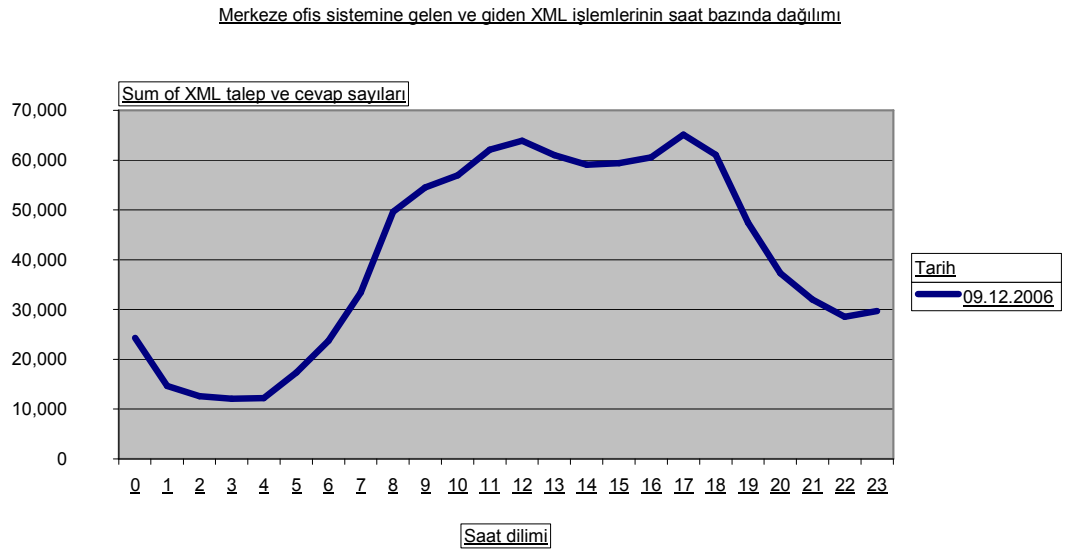
Merkez ofiste konumlandırılmış sistem içerisinde, yedekli ve yük paylaşımlı çalışan sunucular bulunmaktadır. Uygulamaların yük paylaşımlı çalıştığı uygulama sunucuları(NLB), verilerin saklandığı yedekli ve eşzamanlı hizmet veren veritabanı sunucuları (DBS Cluster), bireysel müşterilerle SMS iletişimini sağlayan SMS sunucusu(SMSSERVER), veritabanının güncel kopyasını bulunduran ve raporlama amaçlı kullanılan veritabanı sunucusu (AXREP), ve bu tez çalışması ile oluşturulan uygulamaların bulunduğu Akıllı Sunucu İzleme Yapısı bulunmaktadır.

Bankalar ile merkez ofis sistemi arasında, sanal pos (VPOS-Virtual POS), tahsilat (kurumsal müşteri hesaplarından) ve ödeme (akaryakıt istasyonlarına) işlemleri gerçekleştirilmektedir.

Şekil 1.1.1 de gösterilen yapıda ki örnek istasyon, akaryakıt şirketine ait otomasyon altyapı çalışmaları tamamlanmış 1000 istasyondan bir tanesidir. Her istasyonda çift yönlü uydu iletişimini sağlayan VSAT + çanak sistemi bulunmaktadır. Router cihazı uydu bağlantısı ile herhangi bir nedenle merkeze ulaşılamadığında yedek hattı aktifleştirme görevini gerçekleştirir. Yedek hat PSTN(Public Switched Telephone Network) altyapısını kullanır. Pompa ve Tank otomasyon sistemi ile veriler istasyon otomasyon bilgisayarında oluşturulduktan sonra, XML veri formatıyla merkez ofise aktarılmaktadır.

Ağ iletişiminde MPLS(Multiprotocol Label Switching) teknolojisi kullanılıyor olması nedeniyle, akaryakıt dağıtım şirketi uygulamaları için özel bir VRF (Virtual Route Forwarding Instance) bulunmaktadır. Bu sayede, istasyonlar, bankalar ve merkez ofis arasında özel ve güvenli bir iletişim altyapısı sağlanmıştır.

Merkez ofis uygulaması ile istasyonlar, veri aktarımını XML(eXtensible Markup Language) ile gerçekleştirmektedir.



Veri tablosu	
Saat dilimi	İşlem adedi
0	24333
1	14631
2	12542
3	12076
4	12164
5	17357
6	23703
7	33410
8	49660
9	54490
10	56972
11	62106
12	63916
13	61022
14	59136
15	59394
16	60584
17	65136
18	61078
19	47422
20	37307
21	31980
22	28535
23	29688
Toplam	978642

Şekil 1.1.2: Merkeze ofis sistemine gelen ve giden XML işlemlerinin saat bazında dağılımı

Merkez ofis uygulamasına 06.12.2006 itibariyle yaklaşık yarım milyon işlem gelmekte, ve her bir işlem için merkezden talep noktasına(banka veya istasyon)

cevap gönderilmektedir. Merkez Ofis Uygulaması ağına her geçen gün otomasyon altyapısı tamamlanan yeni istasyonların eklenmesine paralel, merkez ofis işlem yükü de sürekli artmaktadır. Şekil 1.1.2 de 9 Aralık 2006 itibariyle merkezde oluşan işlem yükü saatlik dilimlere ayrıştırılarak gösterilmiştir. Görüleceği üzere günün yoğun saatlerinde bir dakikada oluşan XML talep ve cevap mesajları 1000 adetten fazladır.

Sisteme dahil olan istasyonlar, merkez ofis ve bankalar arasında Ek-4 de belirtilen XML mesajlarını kullanmaktadır.

1.2 Problem Tanımı

Merkez ofis uygulamasına ulaşan talepler, yaklaşık 470 adet farklı hata kodu ile sonuçlanabilmektedir. Hata kodları, altyapı arızalarından, merkez ve istasyon sistemlerindeki eksik/hatalı tanımlamalardan, bankalardan, iletişim servisini sağlayan şirketin iletişim omurgasında oluşabilecek arızalardan, veya operasyonun doğal sonucu olarak; çalıntı kredi kartı kullanılması, müşteri yakıt alım limitinin yetersiz olması, müşteri kredi kartının kullanım süresinin dolması, kredi kartı bakiyesinin yetersiz olması, taşıt tanıma yöntemiyle yakıt alım talebinde bulunan araç üzerindeki kitin(VIU) kuruma ait bir ekipman olmaması gibi çeşitli nedenlerle oluşmaktadır. Örnek problemler Tablo 1.2.1 de gösterilmiştir.

Tablo 1.2.1: Hata Örnekleri

Problem Açıklaması
İstasyon otomasyon altyapısında arıza oluşması
Uydu iletişiminde kesinti/kalitesizlik
Uydu iletişiminde bölgesel problem oluşması
Merkezi uygulama yazılımlarında hata oluşması
Banka iletişiminde problem yaşanması, satışlara provizyon verilmemesi
Çalıntı kredi kartı kullanımı
Müşteri limit aşımı problemleri
Yakıt alımı yasaklanmış müşterilerin alım yapmayı denemesi
Kredi kartı kullanım süresinin sona ermesi
Veritabanında yoğunluk oluşması
Merkez ofis yazılımlarının taleplerin tamamını karşılayamaması

"Ek-3 Pompa ve Tank Otomasyon sistemi hata kodları" kısmında belirtilen 49 adet işlem sonucu, 2005 yılı başlangıcında merkez ofis uygulamasının devreye alınmasından bu yana karşılaşılan başarısız işlemleri göstermektedir.

Operasyon gereği ihtiyaç duyulan önlemlerin yada aksiyonun alınabilmesi için, hata durumuna bağlı olarak aşağıdaki birim kişi yada sistemlere oluşan hataların en kısa sürede bildirilmesi gerekmektedir.

- Otomasyon ve iletişim hizmetinin alındığı dış kaynak firmalarının çağrı sistemine,
- Merkez Ofis sisteminden sorumlu uzmanlara
- İstasyonlardan sorumlu Bölge Müdürlüklerine/Saha Müdürlerine
- Şirket içi çağrı sistemine
- Müşterilere
- İstasyon sahiplerine

Mevcut durumda, bilginin talep edilme zamanına bağlı olarak problem tespiti yapıldığı için bazı verimsizlikler ve riskler söz konusudur;

- Probleme geç müdahale edilmesi,

- Probleme müdahale etmek için artık geç olması,
- Problemin hiç tespit edilememesi,
- Probleme zamanında müdahale edilemediği için, problemlerin büyümesi ve çözümün zaman alması,
- Sistem izleme amaçlı ayrılan insan kaynağı ve maliyeti,
- Problem tespiti için harcanan zaman,
- Mesai saatleri dışında problemlerin ancak şikayetler sonrasında fark edilebilmesi,
- Müşteri ve istasyonlarda oluşacak memnuniyetsizlik.

2. LİTERATÜR ARAŞTIRMASI

2.1 Yapay Sinir Ağları

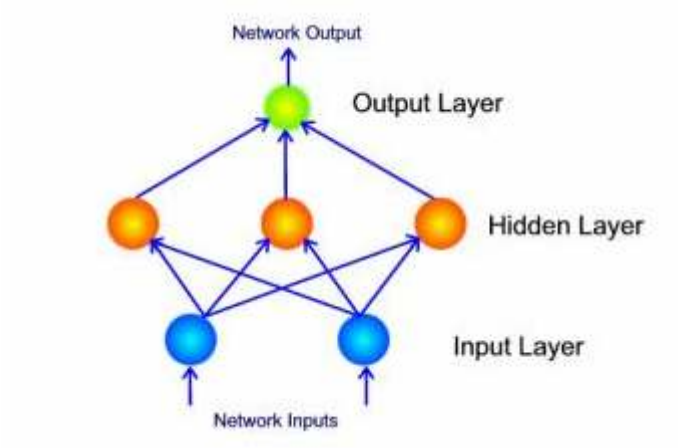
YSA'nın insan beynini andıran ilginç özellikleri vardır. Bilgi depolayabilirler ve çağrışımsal bellek gibi görev görebilirler. YSA yeni şeyler öğrenebilme yeteneğine sahiptirler ve bunun için eğitilebilirler. Genelleştirme yapabilirler, yani kendilerine gösterilen az sayıdaki örneğe dayanarak bunların ortak özelliklerini öğrenebilirler[3].

İnsanlığın doğayı araştırma ve taklit etme çabalarının en son ürünlerinden bir tanesi, yapay sinir ağları (YSA) teknolojisidir. YSA, basit biyolojik sinir sisteminin çalışma şekli simüle edilerek tasarlanan programlama yaklaşımıdır[1]. YSA biyolojik nöronların matematiksel modelleri olup, birbirlerine çok sayıda sinaps ile bağlı bir grup işlem biriminden oluşur. Simüle edilen sinir hücreleri (nöronlar) içerirler ve bu nöronlar çeşitli şekillerde birbirlerine bağlanarak ağı oluştururlar(Şekil 2.1.1). Bu ağlar bir haç farklı seviyeden oluşabilir. Öğrenme, hafızaya alma ve veriler arasındaki ilişkiyi ortaya çıkarma kapasitesine sahiptirler. Diğer bir ifadeyle, YSA'lar, normalde bir insanın düşünme ve gözlemlemeye yönelik doğal yeteneklerini gerektiren problemlere çözüm üretmektedir. Bir insanın, düşünme ve gözlemleme yeteneklerini gerektiren problemlere yönelik çözümler üretebilmesinin temel sebebi ise insan beyninin ve dolayısıyla insanın sahip olduğu yaşayarak veya deneyerek öğrenme yeteneğidir.

Biyolojik sistemlerde öğrenme, nöronlar arasındaki sinaptik (synaptic) bağlantıların ayarlanması ile olur. Yaşayıp tecrübe ettikçe sinaptik bağlantılar ayarlanır ve hatta yeni bağlantılar oluşur. Bu sayede öğrenme gerçekleşir. Bu durum YSA için de geçerlidir. Öğrenme, eğitme yoluyla örnekler kullanarak olur; başka bir deyişle, gerçekleşme girdi/çıktı verilerinin işlenmesiyle, yani eğitme algoritmasının bu verileri kullanarak bağlantı ağırlıklarını bir yakınsama sağlanana kadar, tekrar tekrar ayarlamasıyla olur.

YSA'lar, birbirlerine bağlanmış birçok işlem biriminden (nöronlar) oluşan matematiksel sistemlerdir. Bir işlem birimi, aslında sık sık transfer fonksiyonu olarak

anılan bir denklemdir. Bu işlem birimi, diğer nöronlardan sinyalleri alır; bunları birleştirir, dönüştürür ve sayısal bir sonuç ortaya çıkartır. Genelde, işlem birimleri kabaca gerçek nöronlara karşılık gelirler ve bir ağ içinde birbirlerine bağlanırlar; bu yapı da sinir ağlarını oluşturmaktadır.



Şekil 2.1.1 : Yapay Sinir Ağı Örneği

Sinirsel hesaplamanın merkezinde dağıtılmış, adapte olabilen ve doğrusal olmayan işlem kavramları vardır. YSA'lar, geleneksel işlemcilerden farklı şekilde işlem yapmaktadırlar. Geleneksel işlemcilerde, tek bir merkezi işlem birimi her hareketi sırasıyla gerçekleştirir. YSA'lar ise her biri büyük bir problemin bir parçası ile ilgilenen, çok sayıda basit işlem birimlerinden oluşmaktadır. En basit şekilde, bir işlem birimi, bir girdiyi bir ağırlık kümesi ile ağırlıklandırır, doğrusal olmayan bir şekilde dönüşümünü sağlar ve bir çıktı değeri oluşturur. İlk bakışta, işlem birimlerinin çalışma şekli yanıltıcı şekilde basittir. Sinirsel hesaplamanın gücü, toplam işlem yükünü paylaşan işlem birimlerinin birbirleri arasındaki yoğun bağlantı yapısından gelmektedir.

Çoğu YSA'da, benzer karakteristiğe sahip nöronlar tabakalar halinde yapılandırılırlar ve transfer fonksiyonları eş zamanlı olarak çalıştırılırlar. Hemen hemen tüm ağlar, veri alan nöronlara ve çıktı üreten nöronlara sahiptirler.

YSA'nın ana öğesi olan matematiksel fonksiyon, ağın mimarisi tarafından şekillendirilir. Daha açık bir şekilde ifade etmek gerekirse, fonksiyonun temel yapısını ağırlıkların büyüklüğü ve işlem elemanlarının işlem şekli belirler. YSA'ların davranışları, yani girdi veriyi çıktı veriye nasıl ilişkilendirdikleri, ilk olarak nöronların transfer fonksiyonlarından, nasıl birbirlerine bağlandıklarından ve bu bağlantıların ağırlıklarından etkilenir.

YSA' örüntü tanıma, ses tanıma, el yazı tanıma, satış tahmini, endüstriyel iş akışı kontrolü ve sistem izleme süreçlerinde yaygın olarak kullanılır[8]. Diğer taraftan, lineer sistemlerden farklı olarak nöron ağırlıklarının yakınsamasına bağlı olarak aktivasyon fonksiyonunun belirlenmesi işinde bulanık mantık teorisi yakınsama işini etkinlikle gerçekleştirebilir.

2.1.1 Tarihsel Gelişimi

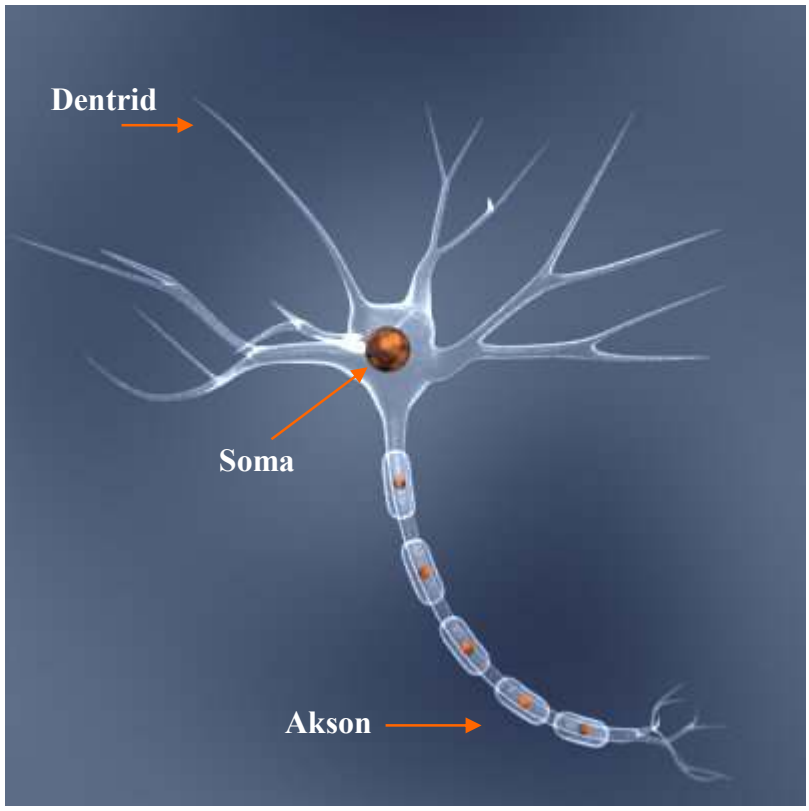
Zihinsel süreçlerin beyinde nasıl oluştuğunu açıklamak isteyen McCullogh ve Pitts tarafından 1943'te ortaya atılmıştır. Bu araştırmacıların önerdiği, yapay sinir hücrelerini kullanan hesaplama modeli, önermeler mantığı, fizyoloji ve Turing'in hesaplama kuramına dayanıyordu. Her hangi bir hesaplanabilir fonksiyonun nöronlardan oluşan ağlarla hesaplanabileceğini ve mantıksal işlemlerinin gerçekleştirilebileceğini gösterdiler. Bu ağ yapılarının uygun şekilde tanımlanmaları halinde öğrenme becerisi kazanabileceğini de ileri sürdüler [1].

Hebb, sinir hücreleri arasındaki bağlantıların şiddetlerini değiştirmek için basit bir kural önerince, öğrenebilen YSA gerçekleştirmek de olası hale gelmiştir [2].

1950'lerde Shannon ve Turing bilgisayarlar için satranç programları yazıyorlardı.İlk yapay sinir ağı temelli bilgisayar SNARC, MIT'de Minsky ve Edmonds tarafından 1951'de yapıldı. Çalışmalarını Princeton Üniversitesi'nde sürdüren Mc Carthy, Minsky, Shannon ve Rochester'le birlikte 1956 yılında Dartmouth'da iki aylık bir açık çalışma düzenledi. Bu toplantıda birçok çalışmanın temelleri atılmakla birlikte, toplantının en önemli özelliği Mc Carthy tarafından önerilen Yapay zeka adının konmasıdır. İlk kuram ispatlayan programlardan Logic Theorist (Mantık kuramcısı) burada Newell ve Simon tarafından tanıtılmıştır[2].

2.1.2 Nöronların Yapısı

Nöron ya da sinir hücresi, sinir sisteminin temel fonksiyonel birimidir. Çeşitli biçim ve büyüklüklerde olabilir. Sinirsel uyarıları elektriksel ve kimyasal yolla iletir. Bir nöron, soma, dentrid ve akson denilen üç ana kısımdan oluşur (Şekil 2.2.1). Başka hücrelerden gelen uyarılar dendiritlerin uçlarından alınır ve aksonların uçlarından diğer hücrelere iletilir[4].



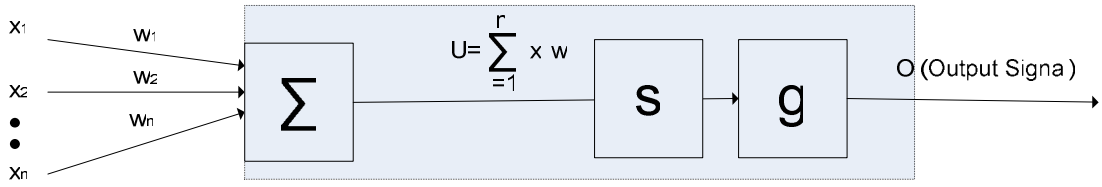
Şekil 2.1.2.1: Biyolojik Nöron

Çekirdek ve çekirdekçiği ihtiva eden esas hücre "soma" dır. Yapı olarak bir ağacın dallarını andıran, diğer nöronlardan gelen uyarıları alıp nöron gövdesine ileten, soma içerisinden çıkan çok sayıda dallanma görüntüsündeki kısım "dentrid" lerdir. Akson ise, gövdeden çıkan sitoplazmik uzantı kısmıdır. Gövdeden çıkan "akson", sinir hücresinden gelen elektriksel sinyalleri çevreye taşımakla görevli olup, böylece sinir hücresini diğer sinir hücreleri veya bir kas hücresi veya bir salgı bezi gibi iş yapan (efektör) hücrelerle bağlar. Mesaj iletiminde önemli rolü vardır[1].

İnsan beyninde 10^{11} adet nöron bulunmakta, ve aralarında yaklaşık olarak 10^{15} adet bağlantı bulunmaktadır [2]. Dentridlerden gelen uyarılar toplanırlar ve axon'a iletilirler. Axon'a ulaşan uyarılar belirli bir eşik seviyesinin üzerinde ise uyarılar axondan diğer nöronların dentrid'lerine iletilir.

2.1.3 Yapay Nöronlar

Genel olarak yapay nöron Şekil 2.1.3.1 de gösterildiği kısımlardan oluşur [5].



Şekil 2.1.3.1: Yapay Nöron Modeli

- Giriş değerleri : x_1, x_2, …, x_n. Giriş değerleriyle ilişkilendirilmiş ağırlıkları bulunur: w_1, w_2, …, w_n.
- Giriş fonksiyonu(2.1.3.1.b) net giriş değerlerinin(2.1.3.1.a) toplamıdır.

  $u = f(x,w)$ **(2.1.3.1.a)**

  $u = \sum_{i=1,n} x_i.w_i$ **(2.1.3.1.b)**
- Aktivasyon fonksiyonu (2.1.3.2) nöronun aktivasyon seviyesini hesaplar.

  $a = s(u)$ **(2.1.3.2)**
- Çıkış fonksiyonu (2.1.3.3) nöronun çıkış değerini hesaplar.

  $o = g(a)$ **(2.1.3.2)**

Biyolojik nöronlar ile yapay nöronlar yapı ve çalışma bakımından benzerlik gösterirler. Biyolojik nöronlarda ki dentrid bölümü, giriş değerleri ve ağırlıklar ile, soma, giriş fonksiyonu ile, axon aktivasyon fonksiyonu ile, synapsis ise çıkış fonksiyonu ile yapay nöron içerisinde betimlenmiştir. En çok kullanılan aktivasyon fonksiyonları Tablo 2.1.3.1 de gösterilmektedir.

Tablo 2.1.3.1: Aktivasyon Fonksiyonu Örnekleri

Function	Definition	Range
Identity	X	(-inf,+inf)
Logistic	$\frac{1}{1-e^{-1}}$	(0,+1)
Hyperbolic	$\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	(-1,+1)
-Exponential	e^{-x}	(0, +inf)
Softmax	$\frac{e^{x}}{\sum_{i} e^{x_i}}$	(0,+1)
Unit sum	$\frac{x}{\sum_{i} x_i}$	(0,+1)
Square root	$\sqrt{x}$	(0, +inf)
Sine	sin(x)	[0,+1]
Ramp	$\begin{Bmatrix} -1 & x \le -1 \\ x & -1 < x < +1 \\ +1 & x \ge +1 \end{Bmatrix}$	[-1,+1]
Step	$\begin{Bmatrix} 0 & x < 0 \\ +1 & x \ge 0 \end{Bmatrix}$	[0,+1]

2.1.4 Nöronlardan Sistemlere

Tek bir nöronun yaptığı iş tanımlı ve basit olmasına karşın, YSA'nın gücü nöronların birbirlerine bağlanarak oluşturulan ağın hesaplama gücünden meydana gelir[5]. Küçük işlem elemanlarının(nöronların) bir araya getirilerek kompleks YSA oluşturmanın arkasındaki düşünceyi daha iyi anlamanın bir yolu da bu gelişimin tarihine bakmaktır.

Donald Hebb 1949 yılında ki psikolojik çalışmasında sinaps'ların öğrenme yeteneğine dikkat çekmiştir 1959 yılında ise Hebb, McCulloch ve Pitts'in fikirlerini

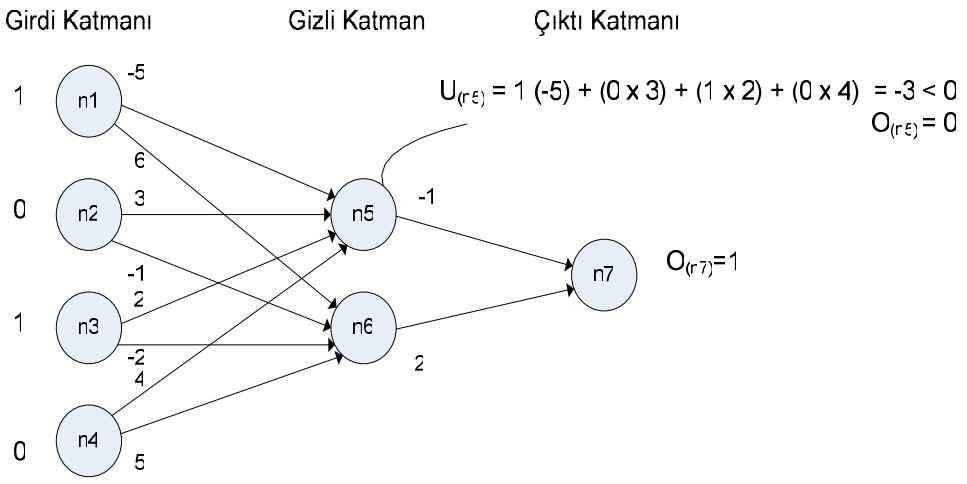
birleştiren Rosenblatt ilk operasyonel sinir ağını açıklamıştır. Rosenbalt'in ortaya koyduğu algı visual sistem çalışmalarında kullanılmıştır[5]. Minsk ve Papert 1969 yılında Rosenbalt'ın çalışmasının teorik limitlerini göstermesinin ardından, bir çok araştırmacı operasyonel sinir ağlarını bir kenara bırakıp sembolik yapay zeka sistem ve yöntemleri üzerine çalışmalar gerçekleştirmiştir. Yeni bağlantılı modeller ve içerisindeki bağımlı bellekler[6], çok katmanlı sinir ağı ve öğrenme algoritması olarak kullanılan geriye doğru yayılma yöntemi[8], adaptive resonans teorisi[9], kendi kendini organize edebilen ağlar [7], ve daha sonra geliştirilen yöntem ve algoritmalar, araştırmacıları tekrar sembolik sinir ağlarına yönlendirmiştir.

Daha sonra, bir çok yapay sinir ağı modeli tasarlanmış ve kullanılmıştır. 1988 yılında Kosko tarafından eş yönlü ilişkilendirilmiş bellek[12], 1989 yılında Moody ve Darken tarafından oluşturulan radyal tabanlı fonksiyonlar[10], yine aynı yıl olasılık teorisine dayanan RAM sinir ağları[13], 1990 yılında Yamakawa tarafından açıklanan bulanık mantık teorisi ile yapay sinir ağlarının bir araya getirilmesiyle tasarımı oluşturulan bulanık nöron ve bulanık sinir ağları[11], oluşturulan ağ modellerine birkaç örnektir.

YSA dört parametre ile tanımlanan bir hesaplama yöntemidir[5].

1. Nöron tipi
2. Bağlantı yapısı
3. Öğrenme algoritması
4. Anımsama algoritması

Basit YSA Şekil 2.1.4.1 de gösterilmektedir. Dört adet giriş nöronu, gizli katmanda iki adet gizli nöron ve bir çıkış nöronu bulunmaktadır. Bağlantılara bağlı olan ağırlıklar, öğrenme algoritması sonucunda oluşmaktadır. Şekil 2.1.4.1 de ki giriş değerleri için eşik seviyesi 0 olarak belirlendiğinde, o(n5) = 0, o(n6)=1 ve o(n7) = 1 olacaktır .



Şekil 2.1.4.1: Dört nöron girişi, iki gizli nöron ve bir çıktı nöronu içeren sinir ağı modeli

2.2 Bulanık Mantık

Bulanık Mantık, 1965 yılında Lotfi Zadeh'in yayınladığı "Fuzzy Sets" (Bulanık Kümeler) konulu makale sonucu oluşmuş bir mantık yapısıdır[16].

Bulanık mantığın temeli bulanık küme ve alt kümelere dayanır. Klasik yaklaşımda bir varlık ya kümenin elemanıdır ya da değildir. Matematiksel olarak ifade edildiğinde varlık küme ile olan üyelik ilişkisi bakımından kümenin elemanı olduğunda 1 kümenin elemanı olmadığı zaman 0 değerini alır. Bulanık mantık klasik küme gösteriminin genişletilmesidir. Bulanık varlık kümesinde her bir varlığın üyelik derecesi vardır. Varlıkların üyelik derecesi, [0,1] aralığında herhangi bir değer olabilir ve üyelik fonksiyonu M(x) ile gösterilir .

Örnek olarak normal oda sıcaklığını 23 derece olarak kabul edersek, klasik küme kuramına göre 23 derecenin üzerindeki sıcaklık derecelerini sıcak olarak kabul ederiz ve bu derecelerin sıcak kümesindeki üyelik dereceleri 1 olur. 23 altındaki sıcaklık dereceleri ise soğuktur ve sıcak kümesindeki üyelik dereceleri 0 olur. Soğuk kümesini temel aldığımızda bu değerler tersine döner. Bulanık Küme yaklaşımında üyelik değerleri [0,1] aralığında değerler almaktadır. Örneğin 14 derecelik sıcaklık için sıcak kümesine üyelik derecesi 0, 23 derecelik sıcaklık için sıcaklık kümesine üyelik değeri (0,25) olabilir.

Klasik kümelerin aksine bulanık kümelerde elemanların üyelik dereceleri [0, 1] aralığında sonsuz sayıda değişebilir. Bu durum üyelik derecelerinin devamlı ve aralıksız kılar. Klasik küme teorisinde soğuk-sıcak, hızlı-yavaş, aydınlık-karanlık gibi ikili değişkenler ile ifade edilen durumlar, bulanık mantıkta biraz soğuk, biraz sıcak, biraz karanlık gibi esnek niteleyicilerle yumuşatılarak gerçek dünya şartlarına daha uygun ifadelerle avantaj sağlar.

2.2.1 Matematiksel Temeli

Bulanık kümeler klasik kümelerin devamı olarak düşünülebilir [14]. Klasik kümelerde ki objeler, küme elemanları yada küme üyeleri olarak tanımlanır. *A* kümesine ait eleman *x*, $x \in A$ ile, ait olmayan elaman *x*, $x \notin A$ ile ifade edilir. *x'*in karakteristik fonksiyonu yada üyelik fonksiyonu $\mu_A(x)$, evrensel küme U içerisinde 1 yada 0'dır. Matematiksel ifade ile klasik kümelerde aşağıdaki ifade kullanılabilir;

$\forall x \in$ U için,

$$\mu_A(x) = \begin{cases} 1 & x \in \text{A}, \\ 0 & x \notin \text{A}. \end{cases} \qquad \textbf{(2.2.1.1)}$$

Diğer bir şekildeki gösterimde aşağıdaki gibidir;

$$\mu_A(x) \in \{0,1\} \qquad \textbf{(2.2.1.2)}$$

Klasik kümelerde üyelik 1 veya 0 değerini alırken, bulanık kümelerde ise üyelik [0,1] aralığında sonsuz sayıda değer alabilir. Bu durumda bulanık kümeler için [0, 1] aralığındaki değer kümesi, üyeliğin üyelik derecesi olarak referans edilir[15].

Bir *A* bulanık kümesi $\mu_A(x)$ üyelik derecesini göstermek kaydıyla aşağıdaki şekilde tanımlanır:

$$A = \{(x, \mu_A(x)) \mid x \in A, \mu_A(x) \in [0,1]\} \qquad \textbf{(2.2.1.3)}$$

Lotfi Zadeh tarafından oluşturulan bulanık küme işlemlerinde bazıları aşağıdaki şekilde ifade edilir[16, 17];

1. Ave B kümelerinin birleşimi C:

   $C=A \cup B$

$$\mu_C(x) = \max(\mu_A(x), \mu_B(x)) = \mu_A(x) \vee \mu_B(x), \quad \forall x \in X \qquad \textbf{(2.2.1.4)}$$

2. A ve B kümelerini kesişimi:

   $C = A \cap B$ veya $C = A$ *AND* B

$$\mu_C(x) = \min(\mu_A(x), \mu_B(x)) = \mu_A(x) \wedge \mu_B(x), \quad \forall x \in X \qquad \textbf{(2.2.1.5)}$$

3. A kümesinin ters A' olarak gösterilen ters fonksiyonu:

$$\mu_{A'}(x) = 1 - \mu_A(x), \quad \forall x \in X \qquad \textbf{(2.2.1.6)}$$

4. A ve B kümesinin kartezyen çarpımı:

$$\mu_{A \times B}(x, y) = \min(\mu_A(x), \mu_B(y)) = \mu_A(x) \wedge \mu_B(y), \quad \forall x \in X \text{ and } \forall y \in Y$$

(2.2.2.7)

Örneğin: P ve Q kümeleri üç farklı değer için aşağıdaki üyelik derecelerine sahip olsun.

$$P = (0.6 \quad 0.7 \quad 0.8)$$

$$Q = (0.9 \quad 0.3 \quad 0)$$

Bu durumda,

$$P \cap Q = (\min(0.6, 0.9) \ \min(0.7, \ 0.3) \ \min(0.8, 0))$$

$$= (0.6 \quad 0.3 \quad 0)$$

$$P \cup Q = (\max(0.6, 0.9) \ \max(0.7, \ 0.3) \ \max(0.8, 0))$$

$$= (0.9 \quad 0.7 \quad 0.8)$$

$$P' = (1\text{-}0.6 \quad 1\text{-}0.7 \quad 1\text{-}0.8)$$

$$= (0.4 \quad 0.3 \quad 0.2)$$

$$P \cap P' = (\min(0.6, 0.4) \ \min(0.7, \ 0.3) \ \min(0.8, 0.2))$$

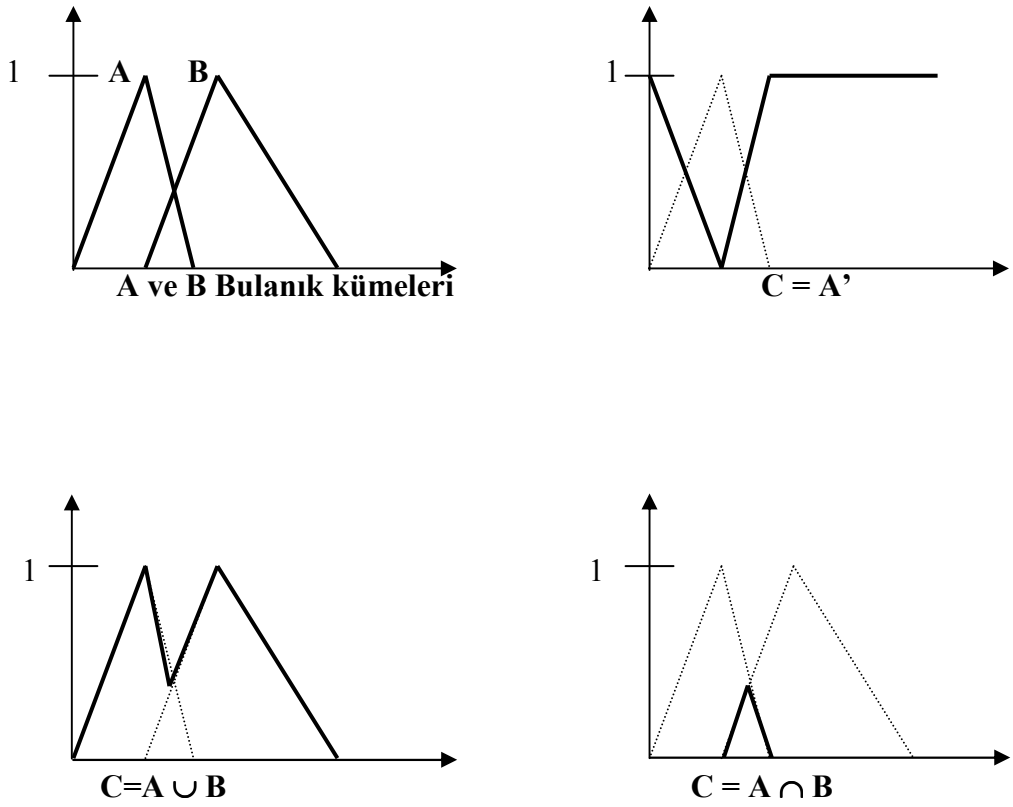
$$= (0.4 \quad 0.3 \quad 0.2)$$

$$P \cup P' = (\max(0.6, 0.4) \ \max(0.7, \ 0.3) \ \max(0.8, 0.2))$$

$$= (0.6 \quad 0.7 \quad 0.8)$$

değerlerini alır.

Kesişim, birleşim ve ters alma işlemlerine dair grafik gösterim Şekil 2.2.1.1 de belirtilmiştir.

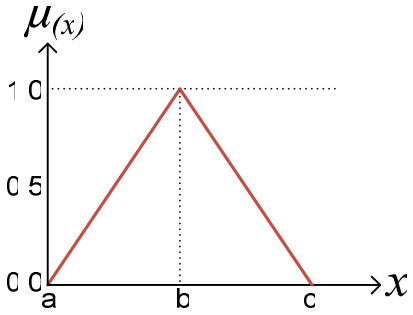


Şekil 2.2.1.1: Bulanık küme işlemlerinin grafiksel gösterimi

Üyelik derecesi fonksiyonları sistem tasarımcıları tarafından belirlenebilmesine rağmen, yaygın olarak kullanılan üyelik derecesi fonksiyonları (triangular, trapezoidal, singleton, gaussian) Şekil 2.2.1.2 de gösterilmiştir.

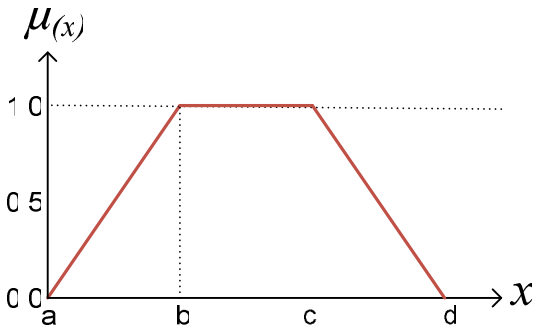
Triangular:

$$\mu(x) = \begin{cases} 0, & c < x < a \\ \dfrac{x-a}{b-a}, & a \le x \le b \\ \dfrac{c-x}{c-b}, & b \le x \le c \end{cases}$$



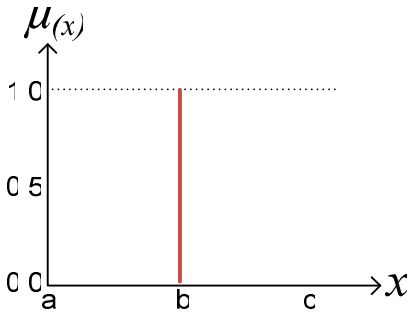
Trapezoidal:

$$\mu(x) = \begin{cases} 0, & d < x < a \\ 1, & b \le x \le c \\ \dfrac{x-a}{b-a}, & a \le x \le b \\ \dfrac{d-x}{d-c}, & c \le x \le d \end{cases}$$



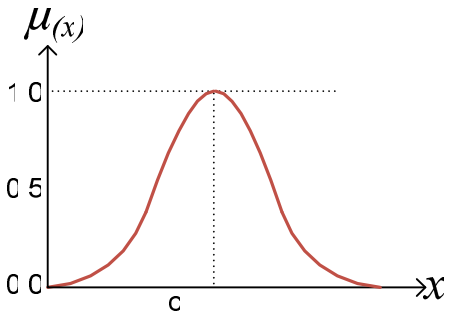
Singleton:

$$\mu(x) = \begin{cases} 0, & x \neq b \\ 1, & x = b \end{cases}$$



Gaussian

$$\mu(x) = e^{-\frac{(x-c)^2}{2\sigma^2}}$$



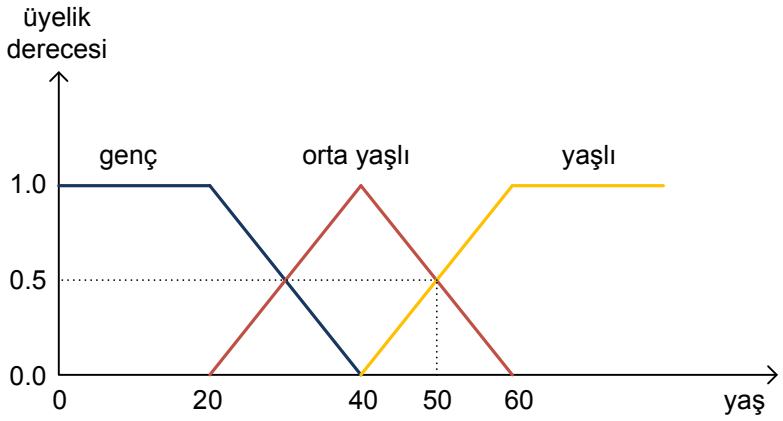
Şekil 2.2.1.2: Yaygın üyelik derecesi fonksiyonları

2.2.2 Dilsel Değişkenler

İstatistik ve olasılık kurumında, bilindiği gibi belirsizliklerle değil kesinliklerle çalışılır. Fakat insanın yaşadığı ortam belirsizliklerle doludur. Bu yüzden

insanoğlunun sonuç çıkarabilme yeteneğini anlayabilmek için de belirsizliklerle çalışılması gereklidir.

Kelimeler bulanık kümelerin içeriğini tanımlar[18]. Bir dilsel değişken l terim kümesi T içerisinde T(l) olarak ifade edilir. Örneğin Şekil 2.2.3.1 de genç, orta yaşlı ve yaşlı bulanık kümelerinden oluşan terim kümesini düşünelim. Bu durumda, T(Yaş) = { genç, orta yaşlı, yaşlı } olacaktır. Şekilde görüleceği üzere [0, 40] yaş aralığı genç bulanık kümesinde, [20, 60] orta yaşlı bulanık kümesinde, [40, ∞] yaşlı bulanık kümesinde üyelik derecelerine sahiptir. Örneğin 50 yaşındaki bir kişi için, 0.0 üyelik derecesinde genç olduğu, 0.5 üyelik derecesinde orta yaşlı olduğu ve 0.5 üyelik derecesinde yaşlı olduğu söylenebilmektedir.



Şekil 2.2.2.1: Dilsel değişken(Yaş)

Üyelik derecesine bağlı olarak, “az, çok az, çok, biraz, hiç“ gibi dilsel niteleyici terimler, üyelik derecesini betimlemekte kullanılır. 50 yaş örneği için, hiç genç değil, biraz orta yaşlı ve biraz yaşlı olarak tanımlanacaktır. Bulanık kümelerin gerçek dünya problemlerinin modellenmesinde ki yeteneği de buradan gelmektedir. Bu nedenle Zadeh dilsel terimlerin bulanık mantığın merkezini oluşturturduğunu belirtmektedir[19, 20].

2.3 XML

XML'in mucidi ve geliştiricisi, HTML'i de (Hyper Text Markup Language) icad etmiş olan Tim Berners Lee'dir[2]. Bağımsız bir kuruluş olan W3C (World Wide Web Consortium) organizasyonu tarafından tasarlanan ve herhangi bir kurumun tekelinde bulunmayan XML (eXtensible Markup Language), kişilerin kendi sistemlerini oluşturabilecekleri, kendi etiketlerini tanımlayarak çok daha rahat ve

etkin programlama yapabilecekleri, bu belirlenen etiketleri kendi yapıları içerisinde standardize edebilecekleri esnek, genişleyebilir, kolay uygulanabilir ve işletim sistemi bağımsız(windows, linux, mMAC OS vs...) bir meta dildir. Bu özelliği ile veri saklamanın yanında farklı sistemler arasında veri alışverişi yapmaya yarayan bir ara format görevi de görür.

XML, elektronik ticaret, elektronik veri değişimi, tedarik zinciri bütünleştirmesi, iş akışı, veri yönetimi, akıllı arama makinaları gibi bir çok alanda stratejik bir araç olarak kullanılan/kullanılacak basit ve esnek metin biçimi teknolojisidir aynı zamanda. XML'in en fazla ilgi çeken tarafı elektronik iş web uygulamalarında evrensel bir veri değişim formatı olarak kullanılmasıdır.

```
<?xml version="1.0" encoding="UTF-8" ?>
<book isbn="0836217462">
        <title>Being a Dog Is a Full-Time Job</title>
        <author>Charles M. Schulz</author>
        <character>
                <name>Snoopy</name>
                <friend-of>Peppermint Patty</friend-of>
                <since>1950-10-04</since>
                <qualification>extroverted beagle</qualification>
        </character>
        <character>
                <name>Peppermint Patty</name>
                <since>1966-08-22</since>
                <qualification>bold, brash and tomboyish</qualification>
        </character>
</book>
```

Şekil 2.3.1: XML dokümanı örneği

Şekil 2.3.1 de XML doküman örneği gösterilmektedir. Örnek doküman görüleceği üzere bir kitabın bilgilerini içermektedir. XML dokümanları XML işlemcisi (yazılımsal) tarafından işlenirler. İşlemciye ulaşan dokümanın ayrıştırılabilmesi için doküman gramerinin yani şemasının işlemciye sunulması gerekmektedir. Yukarıdaki örnek XML dokümanının şeması Şekil 2.3.2 de gösterilmiştir.

XML Şemaları DTD(Document Type Definitions) standartları ile oluşturulur. Standartlar W3C oluşumu tarafından belirlenmiştir. Şema standartları XML dokümanı içerisindeki elamanların özelliklerini ve niteliklerini belirtirler. Şekil 2.3.2

deki *<xs:element name="title" type="xs:string" />* tanımlaması, doküman içerisinde *title* adında bir element bulunduğunu ve içerisindeki veri tipinin *string* yani karakter katarından oluşan kelime, cümle yada cümleler olduğunu belirtir.

```
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="book">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="title" type="xs:string" />
                <xs:element name="author" type="xs:string" />
                    <xs:element name="character" minOccurs="0" maxOccurs=
                                                        "unbounded">
                    <xs:complexType>
                        <xs:sequence>
                          <xs:element name="name" type="xs:string" />
                          <xs:element name="friend-of" type="xs:string"
                                  minOccurs="0" maxOccurs="unbounded" />
                          <xs:element name="since" type="xs:date" />
                          <xs:element name="qualification" type="xs:string"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
            <xs:attribute name="isbn" type="xs:string" />
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Şekil 2.3.2: XML şema örneği

Bilgisayar iletişimindeki büyük gelişimlere ve birikime rağmen, bilgisayar sistemlerindeki ve veritabanlarındaki farklı formatlardaki verilerin şirket içi ve şirketler arası taşınması ve işlenmesi en büyük problemlerden birisidir. XML'in özellikleri, veri yapılarını, içeriklerini ve kavramlarını platform, şirket ve dilden bağımsız bir yapıda temsile imkan vermektedir.

Çeşitli endüstriler için geliştirilmiş olan şemaların dünya çapında erişilebilip kullanılabilmesi için XML kayıt ve depolama (XML registry and repository) hizmeti veren web siteleri bulunmaktadır. Bunlardan bir tanesi OASIS (Organization for the

Adcancement of Stuctured Information Standards) tarafından işletilen www.xml.org sitesidir. Diğer bir site www.biztalk.org, Microsoft tarafından desteklenmekte ve işletilmektedir. Genellikle çeşitli yatay ve dikey sektörler için geliştirilmiş olan şemalar bu sitelerde kayıt edilip saklanmakta ve diğer kuruluşların hizmetine sunulmaktadır.

Çeşitli uygulamalar ve endüstriler için tanımlanmış şema sayısı her gün artmaktadır. Değişik endüstriler için geliştirilmiş olan bir çok şemanın olması bu şemaların hangisinin kendi uygulamalarında baz alınması konusunda kullanıcıları tereddüt'e düşürmektedir. Zamanla bu şemalar bir kısmı birleşerek açık bir standart kümesi oluşacaktır. Bu aşamada 2000 yılında Gartner Group şema kullanıcılarına şunları tavsiye etmişti[20]:

- Kendi endüstriniz için geliştirilmekte olan şemaları takip edin.
- Kurumlara özel XML şemalarından kaçının. Bunlar ilerde ortak ve ortaklarınızla olan iletişimi engelleyecektir.
- 2001 yılı sonunda XML standartlarının oturmaması ile oluşabilecek ve kısa sürecek kullanıcı bir hayal kırıklığına hazır olun.
- 2003 yılı sonunda XML'den önce işlerin nasıl yapıldığını hatırlamak bugün Web'ten önce işlerin nasıl yapıldığını hatırlamak kadar zor olacaktır.
- 2005 yılında XML şema ve standartlar kümesi elektronik ticaretin genel bir dili olarak ortaya çıkacaktır.
- 2005 yılında bilgi ve uygulama entegrasyonu, platformlar arası içerik taşıma, içerik tarama, anında (just-in-time) yazılım taşıma ve teslimi gibi imkanlar sağlayarak elektronik iş işlevsellik ve kapsamını ilerletecek temel teknoloji XML olacaktır.

2.4 TCP/IP

TCP/IP (Transmisson Control Protocol / Internet Protocol) bir ağ iletişim protokolüdür. TCP gönderilen verilerin, gönderildiği sırayla, hedefine ulaşmasını, dolayısıyla verinin güvenli iletimini sağlar [21].

A.B.D Department of Defense Advanced Research Projects Agency (DARPA) projesi adıyla 1970'li yıllarda temelleri atılan TCP/IP'nin ilk amacı, farklı bilgisayar sistemlerinin birbirleriyle iletişimini sağlamaktı. İlk çalışmaların sonucunda, bilgisayar ağındaki kısmi çöküş tüm ağın çalışırlığını etkiliyor, ağ çalışmıyordu. Bu nedenle DARPA ağ iletişiminde heterojen bir yapı kurmak amacıyla Stanford Üniversitesinden Bolt, Beranek, ve Newman (BBN) dan oluşan bir araştırma ekibi kurdu. Ekip hedeflenen çalışmaları 1970'li yılların sonunda tamamladı. Çalışmanın ardından, öncelikle Berkley Software Distribution (BSD) UNIX işletim sisteminde bir standart olarak kullanılmaya başlandı. Takip eden yıllarda diğer işletim sistemleri tarafından kabul edilmesinin ardından Internet'in temelini oluşturan protokol TCP/IP ağ iletişiminde en popüler iletişim standardı olmuştur.

Bağlantılı (Connection-oriented) TCP protokolü, iki bilgisayar arasında veri transferi yapılmadan önce bağlantının kurulması, transfer anında uçtan uça verinin doğru içerikte hedefine ulaştırılması, hata varsa hatalı parçayı tekrar iletmesi esaslarına dayanır.

TCP iki makine arasında kurulan sanal bir bağlantı üzerinden çalışır. Öncelikle istemci sunucuya, kendi kimlik bilgilerini içeren bir bağlantı isteği gönderir. İsteği alan sunucu bu isteğe karşılık onay ve kendi kimlik verilerini içeren bir paket gönderir. En sonunda da istemci makine sunucuya bir onay paketi gönderir ve bağlantı sağlanmış olur (Şekil 2.4.1).

Şekil 2.4.1: Connection-oriented(Bağlantılı) iletişim

Şekil 2.4.2 detayları belirtilen TCP paket yapısında, bayrak(flag) adı verilen kısım verilerin hedefine eksiksiz ve doğru olarak iletimini sağlar. Bayrak aşağıdaki değerleri alabilir.

- URGENT(URG) Bayrağı: Urgent pointer alanının geçerli olduğunu gösterir. Özel durumlarda kullanılmak üzere tasarlanmıştır.

- ACKNOWLEDGEMENT(ACK) Bayrağı: Onay alanının geçerli olduğunu gösterir. Diğer bir tabirle bir önceki gönderinin hedefine ulaştığını göndericiye bildirmek üzere kullanılır .
- PUSH(PSH) Bayrağı : Gönderilen paket içerisinde veri olduğunu ve bir üst katmana iletilmesi gerektiğini gösterir
- RESET(RST) Bayrağı: Bağlantıyı özellikle anormal durumlarda başlangıç durumuna getirir yada kapatır.
- SYNCHRONIZE(SYN) Bayrağı: Gönderen ve alanın sanal bağlantı isteğinde bulundukları anlamını taşır.
- FINISH(FIN) Bayrağı : Gönderimin tamamlandığını gösterir.

SYN (synchronize) her bağlantı isteğinin başında gönderilen paketlerde bulunur ve gönderen sistemin bağlantı isteğini ve kimlik bilgilerini içerir. ACK (acknowledge) bayrağı ise alınan bütün paketlerin ardından, alıcının paketi aldığını onaylamak için gönderilen paketlerde yer alır. FIN (finish) bayrağı taraflardan birinin bundan sonra veri göndermeyeceğini, ancak almaya devam edeceğini bildirir. RST (reset) bayrağı ise taraflardan birinin bağlantıyı tamamen sonlandırdığını bildirir.

TCP/IP tek bir protokol değildir. Bir protokol kümesine verilen, genel isimdir. Daha ayrıntılı bir anlatımla TCP/IP bilgisayarların kendi aralarında nasıl iletişim kuracaklarını belirleyen kurallar topluluğudur. Tablo 2.4.1 de yedi katmanlı OSI modeli içerisinde kullanılan farklı protokoller bulunmaktadır.

Tablo 2.4.2 : OSI Modeli ve Internet Protokolleri

Internet Protokolleri		
Katman		**Protokoller**
7	Uygulama tabakası	HTTP, HTTPS, SMTP, FTP, TFTP, UUCP, NNTP, SSL, SSH, IRC, SNMP, SIP, RTP, Telnet, ...
6	Sunum tabakası	ISO 8822, ISO 8823, ISO 8824, ITU-T T.73, ITU-T X.409, ...
5	Oturum tabakası	NFS, SMB, ISO 8326, ISO 8327, ITU-T T.6299, ...
4	Taşıma tabakası	TCP, UDP, SCTP, DCCP, ...
3	Ağ tabakası	IP, IPv4, IPv6, ICMP, ARP, IGMP, ...
2	Veri bağ tabakası	Ethernet, HDLC, Wi-Fi, Token ring, FDDI, PPP, ...
1	Fiziksel tabaka	ISDN, RS-232, EIA-422, RS-449, EIA-485, ...

TCP segment yapısı Ek-1 de IP packet yapısı Ek-2 de detaylı olarak gösterilmiştir.

3. ÇÖZÜM ALTERNATİFLERİ

Birinci bölümde bahsedilen problem ve aksaklıkların, zamanında tespit edilerek ilgili kişi, birim yada sistemlere iletilmesi için bu tezde yapılan çalışmanın dışında çeşitli yazılımsal çözüm alternatifleri bulunmaktadır. Aşağıdaki kısımda alternatifler ve ilk etapta öngörülmüş dezavantajları bulunmaktadır.

3.1 Periyodik Veritabanı Sorgulama(PVS)

Şekil 1.1.1 de gösterilen Merkez Ofis Uygulama altyapısı içerisindeki AXREP (veritabanının güncel kopyası) sunucusunu belirli aralıklarla sorgulayarak, hatalı işlemleri tespit edecek ve ilgili kişi yada birimlere bildirecek uygulama geliştirilmesi.

PVS yöntemi mevcut uygulama yazılımlarından bağımsız bir uygulamanın geliştirilmesine olanak sağlamaktadır. Bu nedenle Merkez Ofis uygulamasının, istasyon ve bankalara verdiği hizmetin performansını PVS yöntemi azaltmayacaktır.

Diğer taraftan bu yöntem ile belirli bir aralık için belirli sürelerde AXREP'in sorgulanması gerekmektedir. Sorgulamada kullanılan zaman aralığı ile sorgulamanın çalıştırılacağı periyodik süre, performans kısıtları nedeniyle ters orantılıdır. Diğer bir tabirle, daha doğru bir rapor oluşturulması için sorgulama zaman aralığının arttırılması zorunluluğu, problemin daha hızlı tespit edilmesi için sorgulamanın çalıştırılacağı periyodik sürenin azaltılması gerekmektedir. Bu durumda daha sık periyotlarda daha uzun zaman aralığı için veritabanı sorgulanması verimli olmayacağı gibi, hiçbir zaman problemlerin gerçek zamanlı tespitine olanak vermeyecektir.

PVS yönteminin diğer bir dezavantajı ise, veritabanına kaydedilememiş hataları analiz edememesidir.

3.2 Merkez Ofis Yazılımlarına Eklenti(MYE)

Şekil 1.1.1 de gösterilen Merkez Ofis Uygulama altyapısı içerisindeki uygulama yazılımlarına, hata kodlarını dinleyerek analiz gerçekleştiren, müdahale edilmesi

gereken problemleri belirleyen ve gerekli aksiyonu alan eklentilerin entegre edilmesidir.

Bu durumda hataların gerçek zamanlı tespit edilmesi mümkün olacaktır. Ancak, bu durumda, amacı öncelikle istasyonlara hizmet vermek olan merkez ofis uygulamaların da temel amaçların dışında ek yük oluşacaktır. Hizmetin kesintiye uğrama riski yanında, MYE yapısındaki her bir yazılım güncelleme işi için, tüm uygulamanın güncelleme sırasında durdurulması zorunluluğu bulunmaktadır.

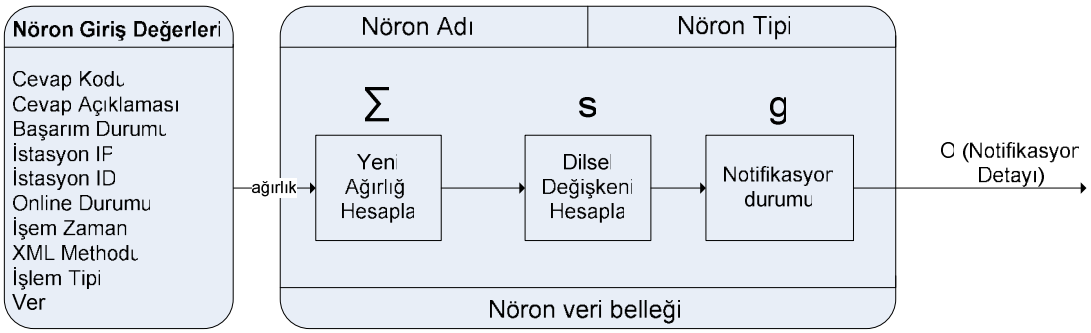
MYE yönteminin diğer bir dezavantajı ise, veritabanına kaydedilememiş hataları analiz edemeyecek olmasıdır.

4. YAPAY ZEKA MODELİ

Giriş bölümünde tanımlanmış problemlerin çözümü için detayları aşağıdaki kısımlarda belirtilen modelin tasarlanmış olmasında ki temel amaç, YSA içerisinde ki basit işlem birimlerinin büyük bir problemi küçük problemler halinde analiz edebilir olması, işlem birimlerinin durumlarına göre genel durumun yorumlanabiliyor olması, bu sayede güçlü bir kontrol ve izleme sürecinin oluşturulabilmesidir.

Yazılım tasarımında YSA modellinin tercih edilmesindeki bir diğer önemli neden de sadece ileriye doğru beslenen bir sistem ile izleme olanağını sağlamasıdır. Detayları aşağıda belirtilen tasarım ile yazılımın üzerinde çalıştığı sunucunun işlemci kaynağının sadece %2 si kullanılarak günde ortalama bir milyona yakın mesaj işlenilmekte ve yorumlanmaktadır.

4.1 Bulanık Nöron Tasarımı



∑ = eğer (Nöron Giriş Değerleri Başarılıysa) {ağırlığı düşür} değilse {ağırlığı arttır}

S = T(l) , T(l) ∈ Dilsel küme { Sağlıklı, Az problemli, Biraz problemli, Kritik problemli}

g = eğer (Nöron dilsel değişken değişimi varsa) {eşik değeri aşımı→uyarı üret} değilse {devam et}

Şekil 4.1.1: Bulanık nöron tasarımı

Tez çalışmasında tasarlanan bulanık nöron, Şekil 4.1.1 de görüleceği üzere, yapay nörondan farklılıklar göstermektedir. Aşağıdaki bölümlerde fonksiyon ve nöron özellikleri gösterilmektedir.

4.1.1 Giriş Fonksiyonu

Biyolojik nöronlarda dentrid olarak adlandırılan kısım, yapay nöronlarda bilginin muhafaza edildiği giriş ağırlıklarıyla giriş değerlerinin çarpımıydı. Burada ki tasarımda ise, giriş değerleri Nöron giriş değerleri adındaki bir değer kümesi olmuştur. Tablo 4.1.1.1 de, Nöron giriş değerleri içerisindeki kısımların açıklamaları bulunmaktadır.

Tablo 4.1.1.1: Nöron Giriş Değerleri

Cevap Kodu	İşlemin akıbet kodunu içerir. "0000" dışındaki tüm kodlar başarısız sayılır
Cevap Açıklaması	Akıbet kodunun açıklamasını
Başarım Durumu	İşlemin başarım durumunu gösterir {doğru, yanlış}
İstasyon IP	İşlem talebini yapan istasyonun IP adresi
İstasyon ID	İşlem talebini yapan istasyonun kurum kodu
Online Durumu	İşlemin asli hattan mı(uydu), yedek hattan mı(dial-up) geldi?
İşem Zamanı	İşlemin talep edilme zamanını
XML Methodu	Ek-4 de belirtilen XML mesaj tipini belirtir
İşlem Tipi	İşlem eğer pompa satış bilgisiyse, işlem tipini belirtir {Nakit, kurumsal, bireysel}
Veri	XML Dinleyici den alınan verinin tamamı

Nörona her bir nöron giriş değer kümesi ulaştığında, nöron giriş değerleri başarılı bir işleme aitse (başarım durumu kısmı), nöron'un ağırlığı yükseltilir, değilse azaltılır. Tasarıma göre bir nöron'un ağırlığı 0 dan az, azami sınırdan fazla olamaz. Bir nörona azami sınırının olması dilsel değişken tasarımı bölümünde bahsedilmiştir. Giriş fonksiyonu Yeni Ağırlığı Hesapla(CalculateNewWeight)'nın yazılımsal kodu Ek-5.1 de gösterilmiştir.

$\sum$ = eğer (Nöron Giriş Değerleri Başarılıysa) {ağırlığı düşür} değilse {ağırlığı arttır} **(4.1.1.1)**

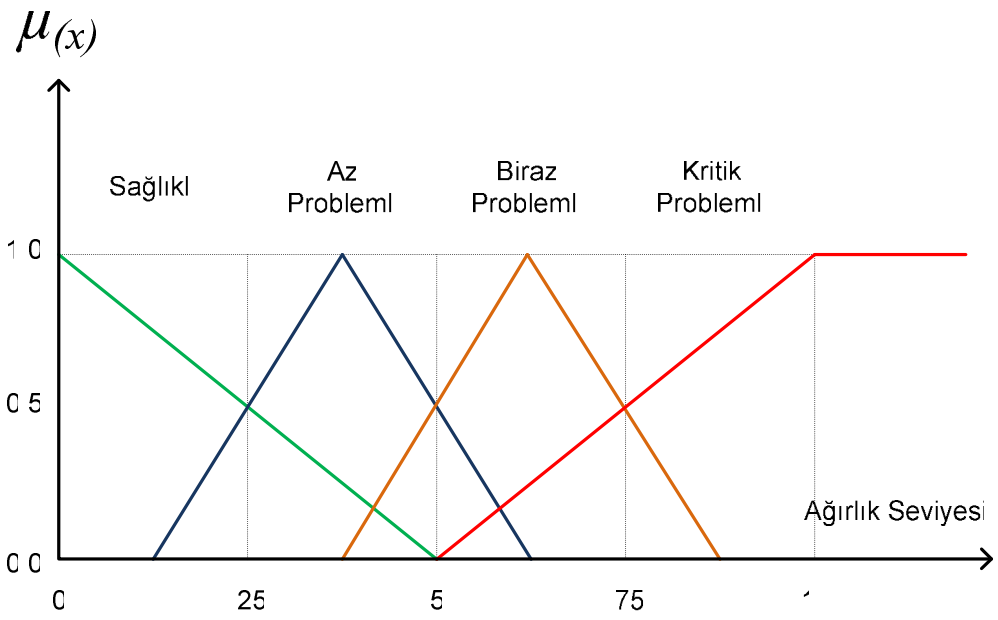
4.1.2 Dilsel Aktivasyon Fonksiyonu

Tasarlanan nörona bulanık nöron denilmesinin nedeni, aktivasyon fonksiyonun dilsel değişkenlerle ifade edilmiş olmasıdır. Aktivasyon fonksiyonları 2. bölümde belirtildiği üzere aktivasyonun derecesini hesaplıyordu. Dilsel aktivasyon fonksiyonu, nöronun ağırlığına bağlı olarak nöron durumunun dilsel durumunu ifade etmek üzere kullanılmıştır.

Bu çalışmada Şekil 4.1.2.1 de belirtildiği gibi nöronun aktivasyon durumunu belirlemek için T(l) = { Sağlıklı, Az problem, Biraz problem, Kritik problem } dilsel değişken kümesi kullanılmıştır. Dolayısıyla dilsel aktivasyon fonksiyonu çalıştırıldığında dört farklı dilsen değişkenden birisi elde edilir.

S = (l) , T(l) ∈ Dilsel küme {Sağlıklı, Az problemli, Biraz problemli, Kritik problemli} **(4.1.2.1)**

Dilsel Aktivasyon fonksiyonu nöron da ki problemin seviyesini belirlemek için kullanılmıştır.



Şekil 4.1.2.1: Dilsel üyelik derecesi fonksiyonu

Nöron ağırlığının bulanık kümelere olan üyelik derecesinin tespiti için kullanılan c# yazılım kodu Ek-5.2 de bulunmaktadır.

Şekil 4.1.2.1 de dilsel aktivasyon fonksiyonu içerisindeki bulanık kümeler gösterilmiştir. Nöron ağırlığından, bulanık kümelere üyelik derecelerine bakılarak, en fazla üye olunan bulanık küme(dilsel değişken) belirlenir. Ağırlık Seviyesi ağırlığın [0,1] aralığındaki ifadesi olarak tanımlanmıştır. Her nöron için tanımlı bir hassasiyet derecesi değeri bulunmaktadır. Hassasiyet derecesi, nöronun bir özlük bilgisidir aslında. Oluşturulmasındaki amaç, nöronların incelediği diğer bir tabirle ajanı olduğu problem parçalarının farklılıklar göstermesidir. Örneğin, çalıntı kredi kart kullanımı olduğu anda ajanın problemi bildirmesi gerekirken, merkez ofis banka iletişiminde ki problemi inceleyen ajanın en azından ard arda 5 işlemin bankadan red aldığını gördükten sonra ilgili bankaya arızanın giderilmesi amacıyla bir e-posta göndermesi gerekmektedir. Ağırlık Seviyesi aşağıdaki denklem ile hesaplanır.

Ağırlık Seviyesi = Ağırlık / Hassasiyet Derecesi **(4.1.2.2)**

Ağırlık Seviyesi değerinin azami 1 değerini alması için 4.1.1 bölümünde de belirtildiği gibi ağırlık değerinin artışı azami hassasiyet derecesinin değeri olabilecek şekilde sabitlenmiştir.

Hassasiyet derecesi nöronun problemler karşısındaki hassasiyetini belirlemek için oluşturulmuştur. Örneğin hassasiyet derecesi 30 olarak tanımlanmış bir nöronun aktivasyon derecesinin kritik problemli olması için ardı ardına 30 tane başarısızlıkla sonuçlanan Nöron giriş değeri alması gerekmektedir. Uygulamada hassasiyet derecesi için varsayılan değer 20 olarak belirlenmiştir. Nöron tipine bağlı olarak Ek-5.3 de belirtilen yazılımın konfigürasyon dosyası içerisindeki (SettlementScale tanımında olduğu gibi) gibi nöronun hassasiyetini azaltmak için varsayılan değerden daha yüksek yada daha düşük hassasiyet derecelerinin tanımlanması mümkündür.

4.1.3 Çıkış Fonksiyonu

Çıkış fonksiyonu, nöronun tespit ettiği problemlerden uyarı üretilmesi işini yapar. Biyolojik nöronlarda olduğu gibi bir eşik değeri ile çalışır. Nöronun dilsel aktivasyon seviyesindeki değişim eşik değerinin aşıldığı ve uyarı üretilmesi gerektiği anlamına gelir.

g = eğer (Nöron dilsel değişken değişimi varsa) {eşik değeri aşımı oluştu, uyarı üret} değilse {devam et} **(3.1.3.1)**

Bulanık nöron tasarımında özel olarak tanımlanmış bazı değerler bulunmaktadır. Bunlar, Nöron adı, Nöron tipi ve Nöron veri belleğidir. Her bir nöronun bir adı vardır ve Nöron Adı alanı içerisinde tanımlanır. Ayrıca nöronun bulunduğu katman ve nöronun tipi nöron tipi alanında tanımlanmaktadır. Nöron veri belleği, nöron ağırlığının yükselmesiyle beraber problemin analiz edilmesi ve bilgiye erişilebilmesi için talep ve cevap mesajlarını içeren işlemlerin nöron üzerinde saklandığı bellektir.

Eşik değerinin aşılmasıyla beraber, Nöron analiz edilmek üzere Ek-5.5 de yazılımsal kodları gösterilmiş Nöron Analiz fonksiyonuna gönderilir ve uyarı detayları bu şekilde oluşturulur.

Problemin düzelmesine paralel olarak, nöron ağırlığı da sıfırlanmış olur. Bu durumda Nöron Veri Belleği de, yazılımın performansı öngörülerek boşaltılmaktadır.

4.1.4 Nöron Ağırlığı

Bulanık nöron da ağırlık, izlenen problem parçasının durumunu belirlemektedir. Nöron giriş değer kümesinin, nöron ağırlığını belirlemesi sayesinde, sistem sadece ileriye doğru beslenerek, diğer bir tabirle geriye doğru hiç bakmaksızın problemleri tespit edebilmektedir. Bu sayede yazılımın yüksek performans ve verimlilik ile hizmet vermesi sağlanmıştır.

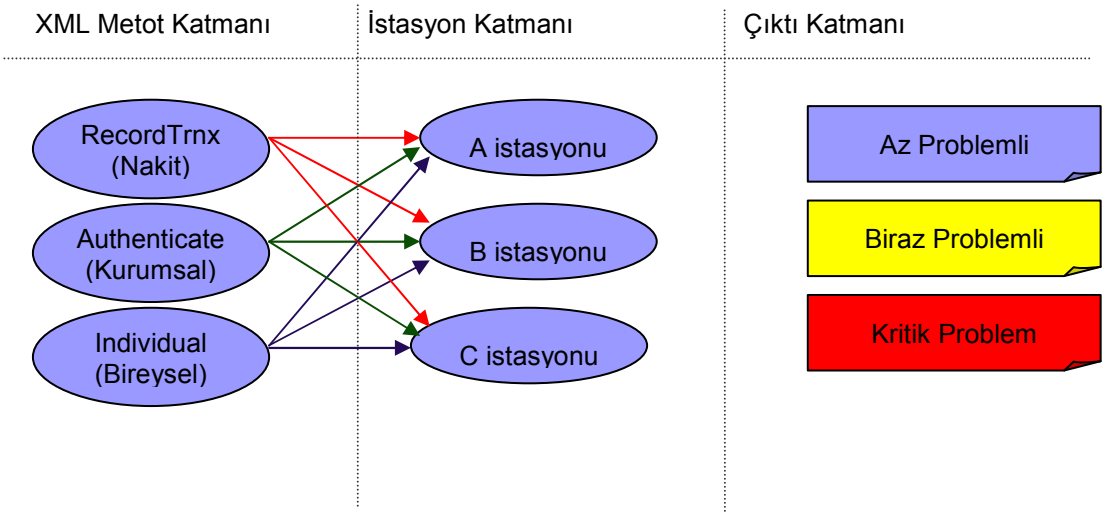
Nöronun ağırlığındaki artış, dilsel değişkenlerin işaret ettiği problem seviyesini de yükseltir. Her bir değişim sistemde, problem seviyesini ve problemle ilgili analiz sonuçlarını içeren bir çağrı açılmasını tetiklemektedir. Nöron ağırlığı azami değerine ulaştığında problemde en kritik seviyeye ulaşmış olur.

4.2 YSA Modeli

Yapay sinir ağlarının gücü, oldukça basit çalışan nöronların bir araya getirilmesi ile ortaya çıkar. Tasarlanan model de merkezi uygulamanın izlenmeye başlanmasıyla beraber Şekil 4.2.1 de bir kısmı gösterilmiş YSA hızlıca oluşmaya başlar.

Şekil 4.2.1 de üç katmanlı bir YSA modeli gösterilmiştir. Birinci katman XML mesaj katmanıdır; EK-4 de belirtilen XML mesajlarından merkez ofis yönüne doğru olanları barındırır(11 adet). İkinci katman istasyon katmanıdır; istasyon bazındaki

problemlerin tespit edilmesi için oluşturulmuştur. Üçüncü katman ise çıktı katmanıdır. Nöronların ürettiği uyarılardır(çağrılardır). YSA'nın bağlantılı bir yapıya sahip olması sayesinde, ikinci katmandaki bir nöronda bir problem tespit edildiğinde problemin genel mi yoksa sadece istasyona ait bir problem mi olduğunu söylemek mümkün hale gelir. Örneğin merkez ofis banka arasındaki iletişimde problem olması durumunda, hem birinci katmandaki bireysel işlemleri takip eden nöron, hem de bireysel işlemlerin gerçekleştirildiği istasyon nöronları problem tespit edecektir. Sistem bu durumda ilgili nöronun bağlantılı olduğu daha üst katmanlardaki nöronların durumuna bakarak sorunun istasyona özel mi yoksa genel bir problemi olduğunu belirleyebilmektedir.



Şekil 4.2.1: YSA tasarımı

Merkez ofis yazılımlarının dinlenmesiyle elde edilen her bir işlem EK-4 de belirtilen XML mesaj tipi ve işlemin talep edildiği istasyon bilgisi ayrıştırıldıktan sonra, öncelikle YSA içerisinde XML metot katmanı ve istasyon katmanı içerisinde varlığı kontrol edilerek, ilgili nöron(ajan) yaratılmış mı kontrol edilir, nöron henüz oluşturulmamışsa nöron yaratılır ve katmanlar arası eksik bağlantılar kurulur.

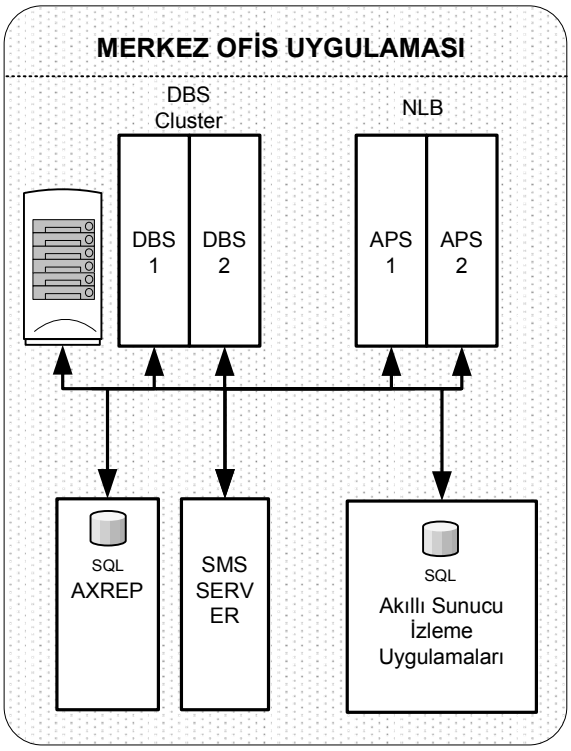
Nöron yaratma sürecine örnek olarak, Tablo 4.2.1 de belirtildiği sırada XML metodu ve istasyon ilişkisinin ayrıştırılarak sisteme gönderildiği durumda, Şekil 4.2.1 deki ağ modeli oluşmaktadır. Tablo içerisinde, oluşturulan nöron ve katmanı kolonu nöron yaratılma işlemini, bağlantı kurma kolonu nöron yaratıldıktan sonra oluşturulan bağlantıları göstermektedir. Öncelikle, A istasyonundan RecordTrnx XML metodu

sisteme gönderilmiş, sonuç olarak XML metot katmanında RecordTrnx, istasyon katmanında A nöronları yaratılmış ve RecordTrnx den A'ya bir bağlantı kurulmuştur. İkinci olarak, B istasyonundan Authenticate XML metodu sisteme gönderilmiş, akabinde XML metot katmanında Authenticate, istasyon katmanında B nöronları yaratılmış ve RecordTrnx den B'ye, Authenticate den A'ya, Authenticate den B'ye bağlantılar kurulmuştur. Üçüncü sırada B istasyonundan RecordTrnx XML metodu sisteme gönderildiği ve her iki nöronunda ağda mevcut olması nedeniyle, yeni bir nöron yada bağlantı yaratılmamıştır. Takip eden işlemlerde de aynı yöntem ile nöron ve bağlantıların yaratılma işlemi devam etmiştir.

Tablo 4.2.1: YSA Oluşumuna Örnek

Sıra No	**XML Metodu**	**İstasyon**	**Oluşturulan nöron ve katmanı**	**Bağlantı kurma**
1	RecordTrnx	A	XML Metot	-
			İstasyon	RecordTrnx→A
2	Authenticate	B	XML Metot	-
			İstasyon	RecordTrnx→B Authenticate→B Authenticate→A
3	RecordTrnx	B	-	-
4	RecordTrnx	A	-	-
5	Individual	C	XML Metot	-
			İstasyon	RecordTrnx→C Authenticate→C Individual→C Individual→B Individual→A
6	Authenticate	C	-	-

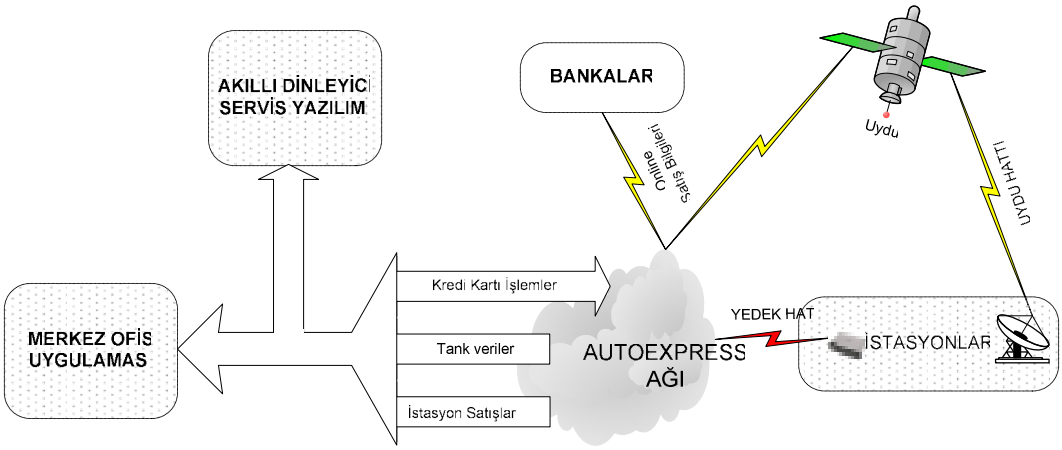
5. AKILLI SUNUCU İZLEME YAPISI



Şekil 5.1.1: Akıllı Sunucu İzleme Yapısı

Akıllı Sunucu İzleme Yapısı, merkez uygulama sunucularına gelen ve sunuculardan istasyonlara giden verileri yenilikçi bir yöntem ile elde eden XML Dinleyici adı verilen uygulamayı, 4. bölümde bulanık nöron tasarımını ve YSA modeli açıklanan Akıllı Ajan adı verilmiş uygulamayı, ve tespit edilmiş bulguları içeren bir veritabanı sunucusunu içerir.

5.1 XML Dinleyici Servis Yazılımı



Şekil 5.1.1: Trafik izleme yöntemi

Şimdiye kadar ağ izleme, trafik analizi yada saldırı tespit sistemlerinde kullanılmış ağ izleme yöntemi, bu kez bir hat üzerinden geçen XML tabanlı verileri dinlemek amacıyla kullanılmıştır.

Merkez ofis uygulamasına gelen ve giden XML mesajlarını dinleyerek bir araya getirip, sonuçları akıllı ajana aktarır. Bu işi gerçekleştirebilmek için, ağ üzerinden geçen parçalı verileri birleştirip önce XML mesaj talebini, daha sonra XML Mesajına sunuculardan verilen cevabı algılar.

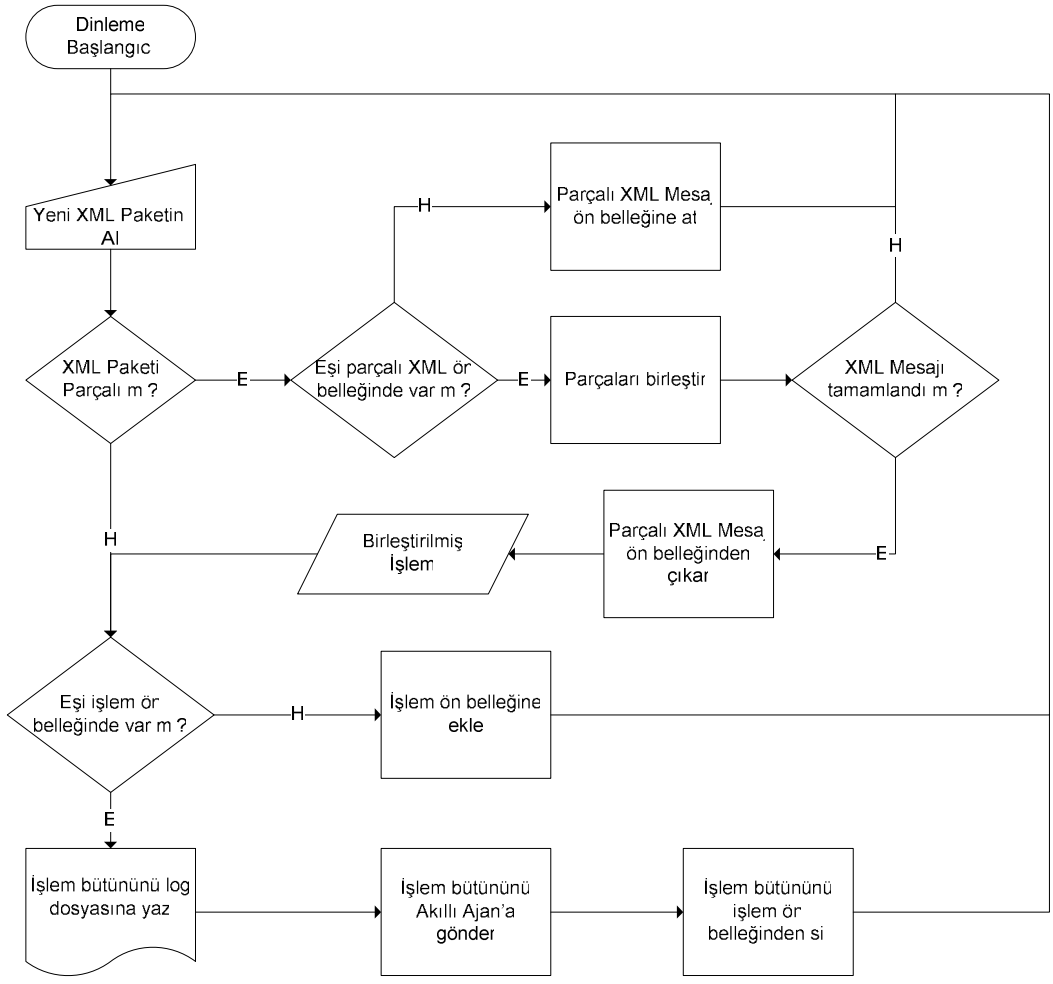
Ayrıca, talebi gelen fakat sunucudan cevabı alınamayan, yada istasyon iletişimindeki aksaklıklar nedeniyle tekrar gönderilen XML cevaplarını arşivler. Bu yöntem ile bölgesel yada istasyon bazında, merkez iletişiminde kalitesizlik yaşayan istasyonlar yada bölgeler tespit edilir.

Dinlemenin etkinlikle gerçekleştirilebilmesi için Akıllı Sunucu İzleme Yapısı üzerindeki ağ kartlarından bir tanesinin dinleme konumuna alınması gerekmektedir. Promicious mode olarak da adlandırılan bu yöntem ile dinlenen ip paketleri(EK-2) ve tcp segmentleri(EK-1) TCP/IP OSI modelinin ağ ve tcp katmanlarında tanımlanan (3. ve 4. katmanında) standartlara göre ayrıştırılırlar.

Yazılımın en kritik noktası, bir günde ortalama 980,000 XML mesajının gidip geldiği, bir ortamda, talepleri ve taleplere verilen cevapları hatasız olarak doğru eşleyebilmektir.

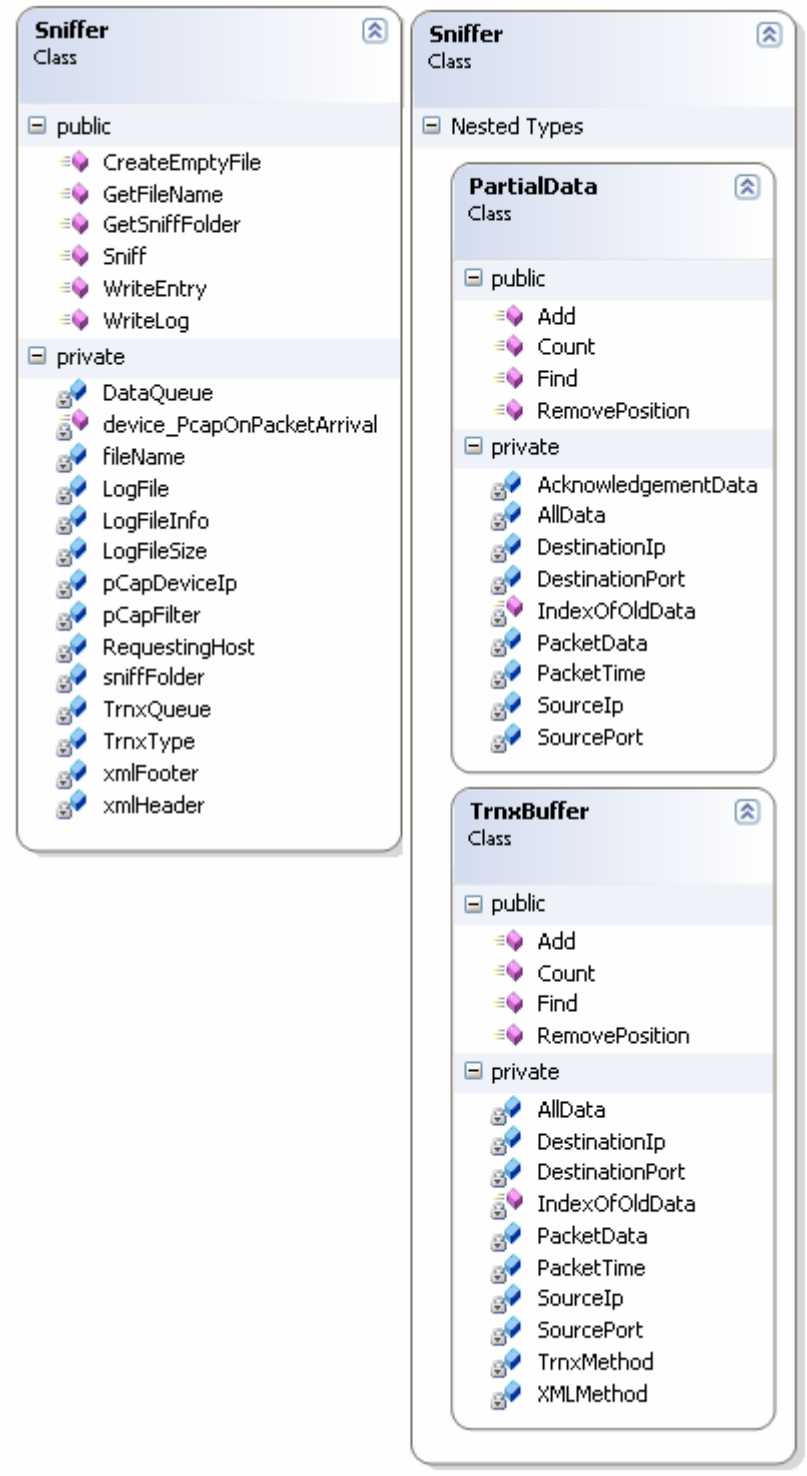
Merkez ofis uygulaması, bankalar ve istasyonlar arasında ki XML veri akışında bağlantılı iletişim yöntemi kullanılır. Bağlantılı iletişimlerde, bağlantıyı başlatan istasyonun, bağlantıda kullandığı kaynak port adresi, bağlantı sona erene kadar değiştirilmeden kullanılır. Diğer bir şekilde XML mesajına yanıt alana kadar aynı kaynak port adres bilgisiyle merkeze ile konuşur. Merkez uygulama talebe cevap verirken hedef port adresi olarak, bağlantıyı açan istasyonun kaynak port adresini belirler. Bu sayede talep ve taleplere verilen cevaplar eşlenebilmektedir.

XML dinleyici servis yazılımı içerisinde oluşturulmuş sanal bellek yapısıyla, gelen talepler ön belleğe atılır. Bir cevap algılandığında ise, bellek içerisindeki talebiyle eşleştirilerek akıllı ajan yazılımına değerlendirilmek üzere aktarılır. XML dinleyici uygulamasının akış diyagramı Şekil 5.1.2 de gösterilmiştir.



Şekil 5.1.2: XML Dinleyici Uygulaması Akış Diyagramı

5.1.1 XML Dinleyici Nesne yapısı



Şekil 5.1.1.1: XML Dinleyici Nesne Yapısı

5.1.2 XML Dinleyici Servis Yazılımının Başarım Testi

Yazılım, test ortamında geliştirilirken, Ek-4 de belirtilen XML mesajlarının tamamının dinleyici servis yazılımıyla algılanabildiği ve elde edilen mesajın Akıllı Ajan'a aktarabildiği gözlemlendi. Yazılım gerçek sistem üzerinde devreye alındıktan iki hafta sonra, 08 Aralık 2006 tarihinde, merkez ofis sunucularına gelen nakit satış talepleri, dinleyici servisin edinimleriyle karşılaştırıldıktan sonra Tablo 5.1.2.1 sonuç elde edilmiştir. Sonuç olarak, adetsel ve içerik olarak veritabanı üzerinde gözlemlenen tüm satış bilgilerinin, XML dinleyici servis ile edinilmiş olmasının gözlemlenmesiyle beraber, merkez ofis uygulamalarının 321 adet talebe yanıt veremediği, dinleyici servisin merkez ofis uygulamalarının hizmet vermediği/veremediği durumları da tespit edebildiği teyit edilmiştir. Yapılan detaylı incelemede 257 adet satışın, merkez sunucularda yapılan günlük veritabanı bakımı sırasında yanıtlanamadığı, 64 adet satışında istasyonlardan merkeze tekrar gönderildiği dolayısıyla, bir hat bağlantı problemi yaşanıldığı durumlarında öngörüldüğü gibi tespit edilebildiği görülmüştür.

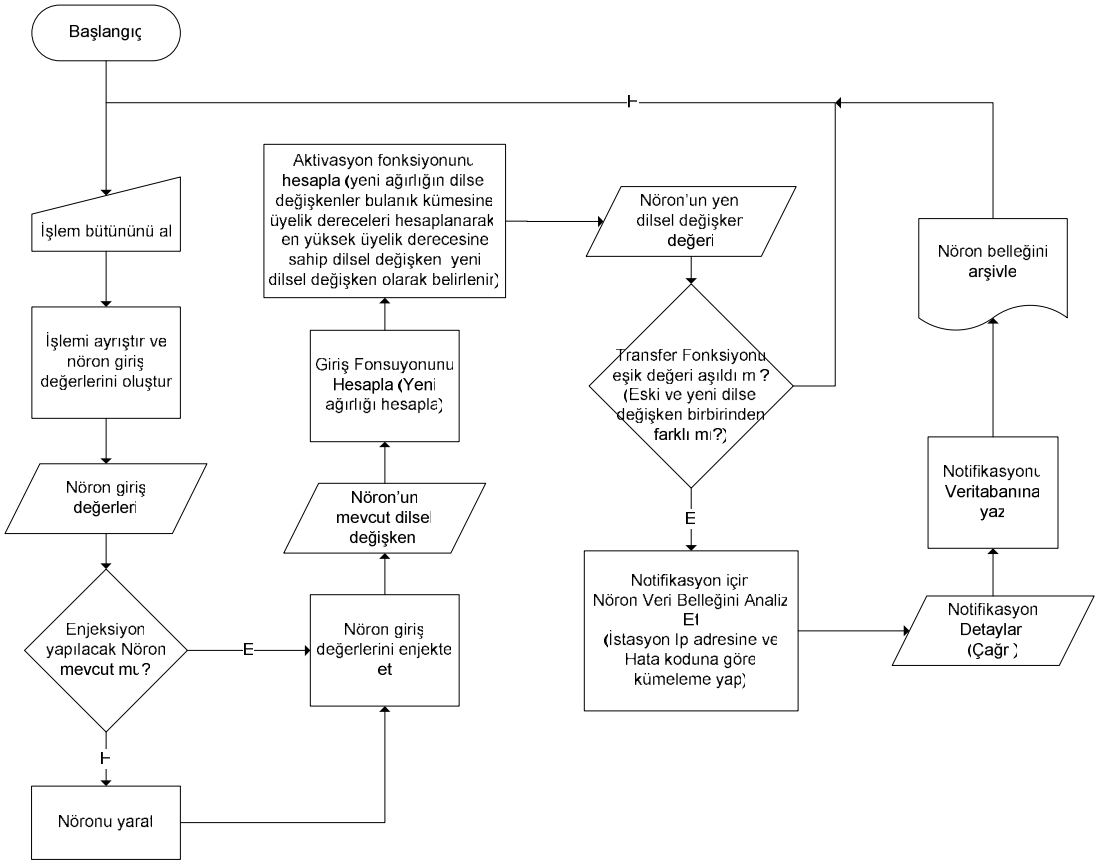
Tablo 5.1.2.1: XML Dinleyici Servis Başarım Tablosu

Tablo	**İşlem adedi**
Veritabanı üzerindeki nakit satış	295,874
XML Dinleyici ve veritabanı üzerinde eşleşen kayıt sayısı	295,874
XML Dinleyici ile elde edilen nakit satış	296,195
XML Dinleyici ile edinildiği halde veritabanı üzerinde bulunmayan kayıt sayısı	321

5.2 Akıllı Ajan Servis Yazılımı

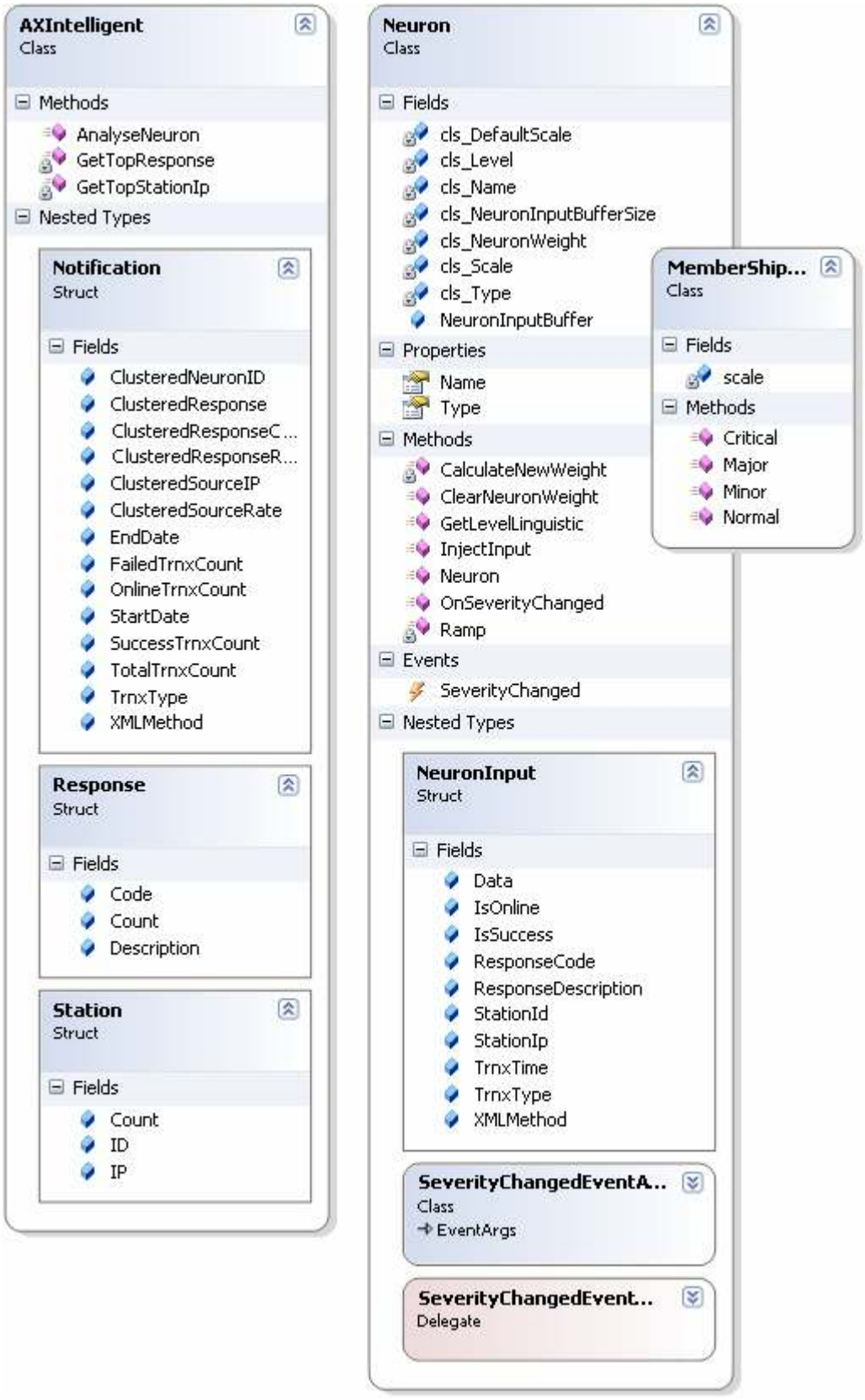
Bu tez çalışmasına konu olan ve tasarımı dördüncü bölümde açıklanmış Yapay Zeka Modeli'nin kurgulandığı uygulamadır. Şekil 5.2.1 de uygulamanın akış diyagramı gösterilmiştir.

XML Dinleyici uygulamasından alınan işlem bloğu, ayrıştırılarak nöron giriş değerler kümesi oluşturulur. Nöron giriş değerler kümesinin gönderileceği nöron sistemde varsa gönderim işlemi yapılır, yoksa nöron yaratıldıktan sonra, yaratılan nörona oluşturulan giriş değerleri enjekte edilir. Daha sonra nöronun mevcut(eski) dilsel değişken değeri alınarak bir kenara kaydedilir. Giriş fonksiyonunun hesaplanmasıyla yeni ağırlık hesaplanır. Hesaplanan ağırlık değerinin dilsel aktivasyon fonksiyonu içerisindeki bulanık kümelere üyelik dereceleri hesaplandıktan sonra, nöronun yeni dilsel aktivasyon derecesi belirlenir. Bu aşamadan sonra, transfer fonksiyonu(çıkış fonksiyonu) eşik değerinin aşılma durumunu hesaplamak üzere eski ve yeni dilsel değişken değerlerini karşılaştırır. Eğer dilsel değişken değerinde bir değişme olduysa, uyarı mesajı üretmek üzere nöron veri belleği analiz edilir, sonuçlar akıllı sunucu izleme yapısı üzerindeki arşiv kayıtlarına ve veritabanına işlenir. Süreç dinleyiciden alınan bir sonraki işlem bloğunu ilgili nörona enjekte ederek devam eder.



Şekil 5.2.1: Akıllı Ajan Akış Diyagramı

5.2.1 Akıllı Ajan Nesne Yapısı



Şekil 5.2.1.1: Akıllı Ajan Nesne Yapısı

5.2.2 Akıllı Ajan Servis Yazılımının Başarım Testi

Yazılım gerçek sistem üzerinde devreye alındıktan iki hafta sonra, 08 Aralık 2006 günü içerisinde akıllı ajan sistemi ile tespit edilen 36 adet çağrının doğruluğu test edildi.

"Ödeme ve işlem sayısı uyumsuz" konulu 12 farklı istasyon için açılan çağrılar merkez ofis veritabanıyla karşılaştırıldı ve ilgili istasyonların merkez ile mutabakatlarında hata oluştuğu, sadece hata olan istasyonlar için çağrı üretildiği görüldü.

"Sistem hatası" konulu açılan çağrılarla belirlenen ve merkeze aktarılamayan satışlar içerisinden 10 tane örnek alınarak, istasyonların yerel veritabanında ki durumları kontrol edildi (bu hata tipi için merkez veritabanında kayıt olmadığı için, işlem akıbetlerinin istasyon veritabanından teyidine ihtiyaç duyulmuştur). Sonuçta, çağrılar içerisinde belirtilen zaman dilimlerinde ilgili satışların istasyon tarafında merkeze aktarılmadığı ve ayna hata kodu ile istasyon tarafında sonuçlandığı görülmüştür.

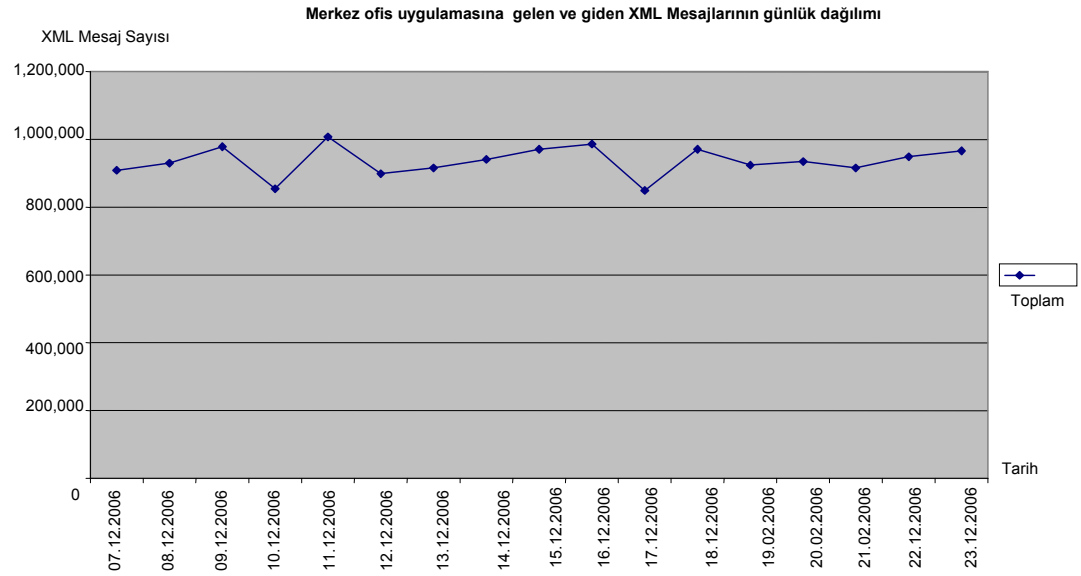
"Müşteri araç tanımı bulunamadı" konusuyla açılan çağrıların merkez ofis veritabanıyla kontrolü sonucunda da, tasarımı yapılan yöntemin beklenildiği gibi üretmesi gereken çağrıları ürettiği görülmüştür.

Tablo 5.2.2.1: Akıllı Ajans Servis Başarım Tablosu

Tablo	Çağrı adedi
ÖDEME VE İŞLEM SAYISI UYUMSUZ	12
SİSTEM HATASI	16
MÜŞTERİ ARAÇ TANIMI BULUNAMADI	8

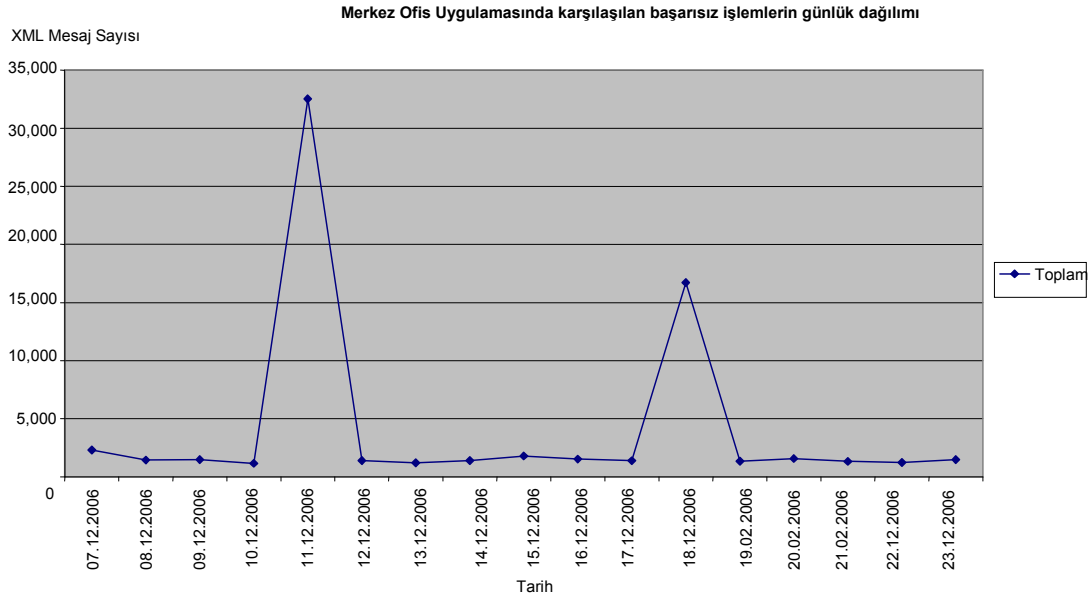
6. SONUÇLAR

XML Dinleyici ve Akıllı Ajan uygulamalarının tamamlanmasının ardından, oluşturulan Akıllı Sunucu İzleme Yapısı 7-24 Aralık 2006 tarihleri arasında 17 gün boyunca merkez ofis uygulamasını izlemek üzere devreye alınmıştır. Çalışmalar sırasında 15,904,637 adet XML Mesajı dinlenmiş ve analiz edilmiştir. XML mesajlarının günlük dağılımı Şekil 6.1 de gösterilmektedir.



Şekil 6.1: Merkez Ofis Uygulamasına gelen ve giden XML mesajlarının günlük dağılımı

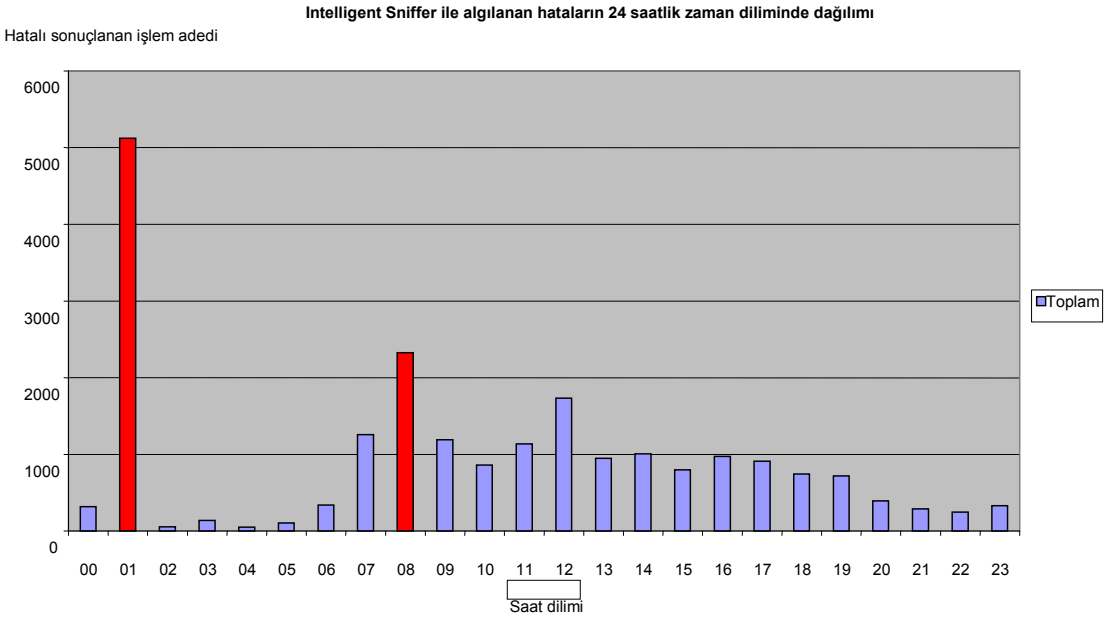
Diğer taraftan işlenen 15,904,637 adet işlemden 71,236 sı başarısızlıkla sonuçlanmıştır. Şekil 6.2 de günlük dağılımı gösterilen hatalı işlemlere dikkat edildiğinde 11 ve 18 Aralık tarihlerinde günlük alınan hata sayısından anlamlı derecede arttığını görmekteyiz. Bu günlerde, merkez uygulama sistemleri güncelleme yapmak amacıyla 10 dakikalık süreyle hizmet vermeyi kestiği için, bu süre zarfında gelen bütün taleplere hatalı cevap verilmiştir. Özel durumları belirtilen bu iki günü çıkardıktan sonra geriye kalan 15 gün içerisinde alınan hataların 24 saatlik zaman dilimine göre dağılımı Şekil 6.3 de gösterilmiştir.



Şekil 6.2: Merkez Ofis Uygulamasında karşılaşılan başarısız işlemlerin günlük dağılımı

Şekil 6.3 de görüleceği üzere, gece yarısı [01:00-02:00) ve sabah [08:00-09:00) aralıklarında karşılaşılan hatalar daha önceki zaman aralıklarına nazaran oldukça yüksektir. Hata detayı akıllı ajan yazılımının ara yüzünden incelendiğinde, veritabanında 1801 hata kodu ile genel bir zaman aşımı problemi oluştuğu, anlamlı yükselişlerin bu nedenle oluştuğu görülmüştür. Veritabanına erişim talepleri zaman aşımı ile sonuçlandığı için, NLB tarafından oluşturulan hata kodu istasyon ve bankalara gönderilmesine rağmen, veritabanı erişilemediği için, veritabanı sunucularına hatalı işlem kayıtları yazılamamaktadır. Bu nedenle problem şimdiye kadar bu kadar açık bir şekilde tespit edilememişti. Dolayısıyla bu noktada akıllı ajan, keşfedilmemiş bir bilgiyi çıkarmış oldu. Daha sonra yapılan teknik incelemelerde, ilgili zaman dilimlerinde veritabanında yapılan yedekleme ve bakım işlemlerinin, veritabanı sunucusu üzerinde yüksek seviyede yoğunluk oluşturduğu görülmüştür.

Alınan hataların hata kodlarına göre dağılımına (Tablo 6.1) dikkat edildiğinde, istasyonların merkez ile mutabakatlarında da yoğun hatalar dikkat çekmektedir.



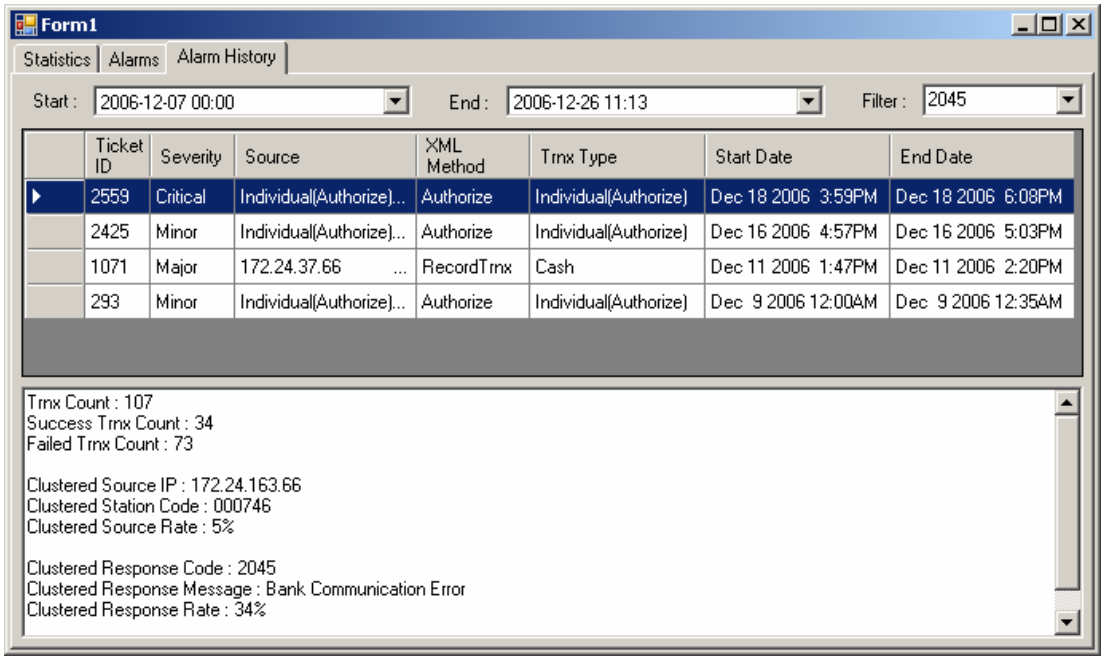
Şekil 6.3: Alınan hataların saat bazında dağılımı

Tablo 6.1: Alınan hataların İşlem Tipine Göre Dağılımı

	Hata Kodu	Açıklama	Toplam Adet
Veritabanı hatası	1002	İşlem tipi veritabanına yazılamadı (Zaman aşımı)	68
Kurumsal müşteri onay hataları	1204	Satış bilgileri verilen onaydan farklı	15
	1207	Müşteri için tanımlı aktif PAN bulunmuyor	217
	1210	Tanımsız VIU(Araç Tanıma Ünitesi)	1858
	1211	Onay verilmediği halde sonlandırılan satış(Reddedilir)	2
	1212	VIU kara listede	138
	1213	VIU için tanımsız ürün	775
	1215	Kurumsal filoya satış izni verilmedi	848
	1217	VIU aktif değil	8
	1218	Müşteri aktif değil	66
	1220	Miktar hatası	2
	1227	Araç/filo için gün saat kısıtı mevcut	439
	1228	İlgili istasyondan araç/filo yakıt alamaz	113
	1229	İstasyon banka tanımı bulunmuyor	4
	1232	Vardiya alım limiti aşıldı	469
	1233	Haftalık alım limiti aşıldı	454
	1234	Aylık alım limiti aşıldı	206
	1235	Yıllık alım limiti aşıldı	14
Zaman aşımı hataları	1801	Zaman aşımı	4682
	1809	İstasyon banka tanımı kontrol edilirken hata oluştu	6
	1814	Tüketim limiti kontrol edilemedi (Zaman aşımı)	1
	1815	VIU(Araç Tanıma Ünitesi) bilgisi kontrol edilirken zaman aşımı oluştu.	19
	1819	Mutabakat başarısız. Zaman aşımı	4
İstasyon mutabakat hataları	1902	Mutabakat başarısız. Bireysel miktar toplamı	2
	1904	Mutabakat başarısız. Kurumsal işlem sayısı	185
	1905	Mutabakat başarısız. Kurumsal miktar toplamı	16
	1907	Mutabakat başarısız. Nakit İşlem	4564
	1908	Mutabakat başarısız. Nakit miktar	1189
	1909	Mutabakat başarısız. Nakit hacmi	6
Bankalardan alınan hatalar	2012	Banka tanımı bulunamadı	3
	2045	Banka iletişim hatası	141
	2046	Banka cevap mesajı bozuk	1
	2047	İstasyon banka bilgisi, bankadan onay almadı	2
	2060	Talep edilen işlemi gerçekleştirmek için yetersiz izin seviyesi	136
	2062	Geçersiz işlem	13
	2080	Banka sistem hatası	2
	2081	Banka finansal kurumu tanımıyor	36
	2082	Kredi kart bakiyesi yetersiz	250
	2083	Kredi kart vade aşımı	143
	2084	Kredi kartı limiti aşıldı	8
	2085	Çalıntı kredi kartı	41
	2086	İşlem tipi kart sahibine kapalı	2
Sistem hataları	9998	Sistem hatası	22843
	9999	Sistem hatası	22290
	Toplam		71236

6.1 Nöronların Ürettiği Çağrıların Değerlendirilmesi

Akıllı Ajan sisteminde modellenmiş YSA modeliyle, 17 günlük süre boyunca 164'ü kritik, 975'i biraz problemli, 737'si az problemli 582'si sağlıklı seviyede olmak üzere toplam 2458 adet çağrı açılmıştır. Açılan çağrılar, ve aktif alarmlar akıllı ajan uygulaması arayüzünden takip edilmektedir(Şekil 6.1.1).



Şekil 6.1.1: Akıllı Ajan uygulama arayüzü

Arayüz ile aktif olan yada kapatılmış çağrılar sorgulanabilmekte, hata koduna, istasyon koduna, istasyon IP adresine, XML Mesaj tipine, işlem tipine göre süzülebilmektedir. Şekil 6.1.1 de, 2045 hatası(banka iletişim hatası) için 2006-12-07 00:00 ile 2006-12-26 11:13 aralığı arasında oluşan çağrılar gösterilmektedir.

2599 numaralı çağrı yorumlandığında;

- 18 Aralık 2006 15:59 tarihinde bireysel otorizasyon işlemlerinde kritik seviyede bir problem oluştuğu,
- Problemin aynı gün 18:08' e kadar devam ettiği
- Problem açıklamasına bakıldığında, banka ile iletişim hatası olduğu,
- Problem boyunca ilgili nörona gelen 107 işlemden 73'ünün başarısızlıkla sonuçlandığı,

- En fazla hatanın 5% gibi düşük bir oran ile 000746 numaralı istasyonda gerçekleştiği, problemin istasyon spesifik bir problem olmadığı,
- Hata kodlarının %34 oranında 2045 kodunda kümelendiği, oranın düşük olması nedeniyle, ilgili nöronun farklı hata kodlarıyla sonuçlanan işlem içerdiği,

Sonuçlarına varılmıştır.

Açılan çağrılar incelendiğinde 17 gün boyunca aşağıdaki hatalar için belirtilen adet kadar çağrı üretildiği görülmüştür (Tablo 6.1.1)

XML dinleyici yazılımının tekrar transfer edilen ya da tamamlanmayan işlemleri arşivlediğini daha önceki bölümlerde belirtmiştik. Yapay Zeka Modelinin bu verileri kullanarak, kalitesiz bağlantıya sahip istasyonların tespit edilebildiği de görüldü. 000437 numaralı istasyonun merkeze arka arkaya aynı işlemi gönderdiği, zaman zamanda merkez cevabının istasyona ulaşmaması nedeniyle merkez cevabının tekrarlanarak gönderildiği görüldü. İstasyon hattının problemli olduğu düşünülerek istasyon ile görüşüldüğünde, istasyonun uydu kablo bağlantısının gevşediği, bu nedenle istasyonun merkeze yedek hat ile bağlandığı, yedek hattın ise paralel bir telefon şeklinde görüşme amaçlı kullanılabildiği, tekrar gönderilen yada tamamlanmayan işlemlerin paralel hattın kullanılmasıyla oluştuğu anlaşıldı.

Sonuç olarak, akıllı ajan uygulamasıyla, başarısızlıkla sonuçlanan işlemleri değerlendirmek için bu tezde önerilen yöntem, giriş bölümünde tanımı yapılan problemlerin gerçek zamanlı tespitini yapılabilmiş, 17 günlük çalışma süresince Tablo 6.1.1 de listelenmiş problemler için çeşitli seviyelerde uyarı ürettiği görülmüştür.

Tablo 6.1.1: Çağrı Üretilen Hata Kodları İçin Üretilen Çağrılar

Hata Kodu	Açıklama	Açılan çağrı adedi
1233	HAFTALIK KOTA AŞIMI	35
1210	MÜŞTERİ ARAÇ TANIMI BULUNAMADI	163
1907	ÖDEME VE İŞLEM SAYISI UYUMSUZ	106
1801	SİSTEM HATASI	207
1215	FİLOYA SATIŞ İZNİ VERİLMEDİ	48
1234	AYLIK KOTA AŞIMI	9
1908	ÖDEME VE TUTAR UYUMSUZ	18
1213	KART/ARAÇ İÇİN TANIMSIZ ÜRÜN	24
2060	BANKA İLETİŞİM HATASI	4
1227	GÜN/SAAT KISITLANMIŞ	15
2045	BANKA İLETİŞİM HATASI	3
9998	SİSTEM HATASI	651
1232	VARDİYA KISIT KOTASI AŞIMI	23
1228	iSTASYONDAN ALIMA İZİN YOK	4
1904	KURUMSAL İŞLEM SAYISI UYUMSUZ	2
9999	SİSTEM KAPALI	1052
1218	MÜŞTERİ TANIMI AKTİF DEĞİL	1
9998	SİSTEM HATASI	651
1212	ARACA SATIŞ İZNİ VERİLMEDİ	2
1207	KREDI KARTI BİLGİSİ BULUNAMADI	4
1002	ISTASYON KODU KONTROL EDİLEMEDİ(ZAMAN AŞIMI)	9
1815	SİSTEM HATASI	2

6.2 PVS ve MYE Alternatiflerinin Akıllı Sunucu İzleme Yapısı ile Karşılaştırılması

PVS yöntemi ve MYE yöntemleri akıllı sunucu izleme yapısına nazaran bir takım dezavantajlara sahiptir. MYE Yönteminin dezavantajları;

- Merkez Ofis uygulamasına entegre edilerek çalıştırılacağı için, istasyon ve bankalara verilen hizmet kalitesi düşürecektir,
- Yazılım güncelleme sırasında merkez ofis uygulamasının durdurulma zorunluluğu,
- İhtiyaç duyulan sistem kaynağının NLB sunucularından kullanılması,
- Kalitesi düşük istasyon bağlantılarının tespit edilememesi,
- Bölgesel problemlerin algılanamaması,
- Merkez Ofis uygulaması çalışmadığında MYE'nin de hizmet verememesi,

PVS yönteminin dezavantajları;

- Gerçek zamanlı analizi yapılamasının mümkün olmaması,
- Sistemde problem olmasa bile periyodik olara AXREP sunucusunun belirli bir aralık için sorgulanması ile uygulamanın gereksiz yere sistem kaynağı kullanması,
- İşlem hızının raporlama aralığıyla(geriye dönük zaman dilimi) ters orantılı olması,
- Özellikle istasyon ve müşteri bazındaki hataların tespit edilememesi,
- Bölgesel problemlerin algılanamaması,

Akıllı sunucu izleme yapısında her iki alternatif çözümün zayıflıklarıyla karşılaşılmaması, daha önceden farkında olunmayan bir takım problemlerin tespit edilmiş olması, teze konu olan projenin diğer alternatifler önünde anlamlı bir şekilde öne çıktığını göstermiştir. Karşılaştırma tablosu, Tablo 6.2.1 de gösterilmektedir.

Tablo 6.2.1: Çözüm Alternatifleri Karşılaştırma Tablosu

Kriter	IS	PVS	MYE
Mevcut Sistem Performansına Etkisi	✔	✔	✘
Gerçek zamanlı analiz	✔	✘	✔
Yazılım güncelleme kolaylığı	✔	✔	✘
İhtiyaç duyulan sistem kaynağı	✔	✘	✘
İşlem Hızı	✔	✘	✔
Rapor Kapsamı	✔	✘	✘
Rapor doğruluğu	✔	✔	✔
Bölgesel iletişim problemlerinin algılanması	✔	✘	✘
Merkez Ofis Uygulaması çalışmadığında hizmet verebilir mi?	✔	✔	✘

6.3 Projede Devam Eden Çalışmalar

Merkez Ofis Uygulama sunucularında gerçekleşen işlemler için yaklaşık 470 adet farklı cevap mesajı üretilmektedir. Hata mesajları aşağıdaki kişi yada sistemlerin probleme müdahale etmesi için bölüştürüldükten sonra, Akıllı Ajan tarafından oluşturulan çağrılar problemin tipine göre SMS ve/veya e-posta yoluyla ilgili kişi yada sistemlere aktarılacaktır. Çağrıların yönlendirileceği kişi yada sistemler;

- Otomasyon firması çağrı sistemi
- Uydu iletişim firması çağrı sistemi
- Merkez Ofis Uygulamasının çalışırlılığından sorumlu teknik uzmanlar
- Saha Müdürleri
- Bölge Müdürleri
- Kurum içi çağrı sistemi

Ayrıca, istasyon iletişim bilgileri ve müşteri özlük bilgileri Akıllı Ajan uygulamasına aktarılacaktır. Bu şekilde, istasyon ve müşteri için açılan çağrılara müdahale ve çözüm sürelerinin daha da kısalacağı hedeflenmektedir.

Çağrı açılırken, nöron veri belleğinde bulunan veriler klasik kümeleme yöntemiyle, istasyon ve cevap koduna göre kümeleniyor. Projenin devamında Neuron buffer'ı

üzerindeki veriler içerisinde, veri madenciliği yöntemleri ile kümeleme, sınıflandırma ve kolerasyon analizleri yapılacaktır.

Verilerin gerçek zamanlı olması nedeniyle, Akıllı Ajan üzerinde Karar Destek Sistemlerinin bir parçası olan kokpit tasarımlar gerçekleştirilecektir.

KAYNAKLAR

[1] **Sorias, S.**, 2005. Yapay Nöron Ağları Bilişsel İşlevleri Anlamamıza Yardım Edebilir Mi?, *Journal of Internal Medical Sciences,* 40**,** 8-16.

[2] **Yapay Sinir Ağları**, ***http://tr.wikipedia.org***

[3] **Lin, C.,** 1991. Neural Network Based Fuzzy Logic Control and Descision System, *IEEE Trans. Comput. 40*, 1320-1336.

[4] **Ermis K.,** Midilli A., Dincer I., Rosen M. A., 2005. Artificial neural network analysis of world green energy use, *Energy Policy. Kidlington* Vol. 35, Iss. 3; pg. 1731.

[5] **Nikola K. K.,** 1996. Foundations of Neural Networks Fuzzy Systems and Knowledge Engineering, *A BradFord Book The MIT Pres Cambridge Massachusetts* London, England.

[6] **Amari S., Arbib M.,** 1977. Competition and cooperation in neural nets. *System Neuroscience.* Newyork, Academic Pres, pp 119-165.

[7] **Kohonen T.,** 1982. Self orginesed formation of topologically correct feature maps. *BiologicalSsybernetics.* 43, 59-69.

[8] **Werbos P.,** 1990. Backpropagation through time: What it does and how to do it. *Proceeding of the IEEE*, 87-10.

[9] **Carpenter G., Grossberg S.,** 1987. ART 2: Stable self organization of pattern recognition codes for analog input patterns. *Applied Optics*, 26:4919-4930.

[10] **Moody T., Darken C.,** 1989. Fast learning in networks of locally tuned processing unit. *Neural Computation*, 1:281-294.

[11] **Yamakawa T.,** 1990. Pattern recognition hardware system employing a fuzzy neuron. *In proceedings of the International Conference on Fuzzy Logic and Neural Networks*, Iizuka, Japan, July 1990, pp 943-948.

[12] **Kosko B.,** 1988. Bidirectional associative memories. *IEEE Transactions on Systems, Man and Sybernetics*, 18:49-60.

[13] **Taylor J., Mannion C.,** 1989. New Developments of Neural Computings. Bristol, England, Adam Hilger.

[14] **Klir, G.J., and Yuan, B.,** 1995. Fuzzy Sets and Fuzzy Logic: Theory and Applications, *Prentice Hall Inc*., Upper Saddle River, NJ.

[15] **Bojadziev M.,** 1995. Fuzzy Sets, Fuzzy Logic, Application, *World Scientific,* New Jersey.

[16] **Zadeh L.,** 1965. Fuzzy Sets, *Information and Control* 8:338:358.

[17] **Zadeh L.,** 1968. Fuzzy Sets, *Information and Control* 12:94-102.

[18] **Zadeh L.,** 1975-1976. The concept of linguistic variable and its application to approximate reasoning, *Information Sciences,* 8:199-249, 8:301-357, 9:43-80.

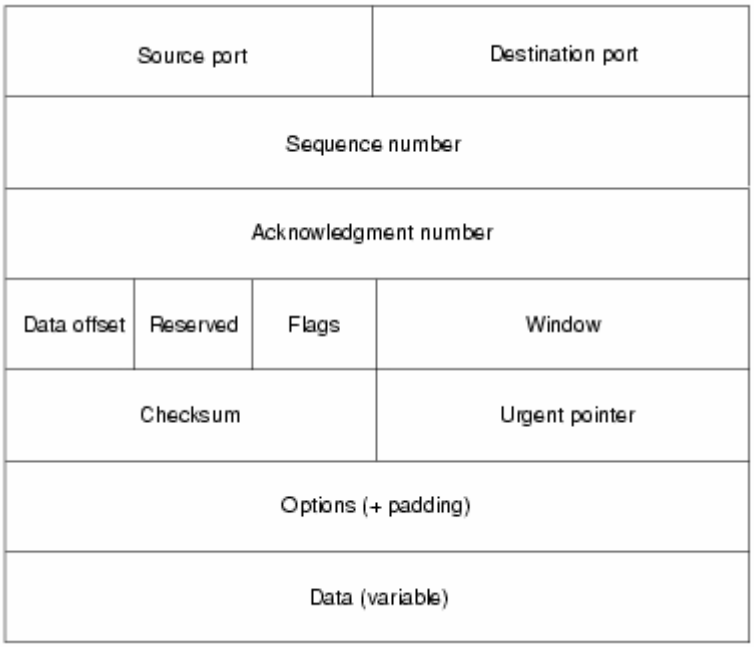
[19] **Zadeh L.,** 1973. Outline of a new approach to the analysis of complex systems anddecision processes, *IEEE Trans. Sys. Man Cybernet,* 28-44.

[20] **Gartner Group,** 2000. XML: The life-booster for Web-business applications http://www.softwareag.com/xml/presentations/Agenda2000Gartner/hochberg.ppt.

[21] **Birkmaier C., 2002.** Broadcast Engineering, *Overland Park,* Vol. 44, Iss. 9, p.14.

EKLER

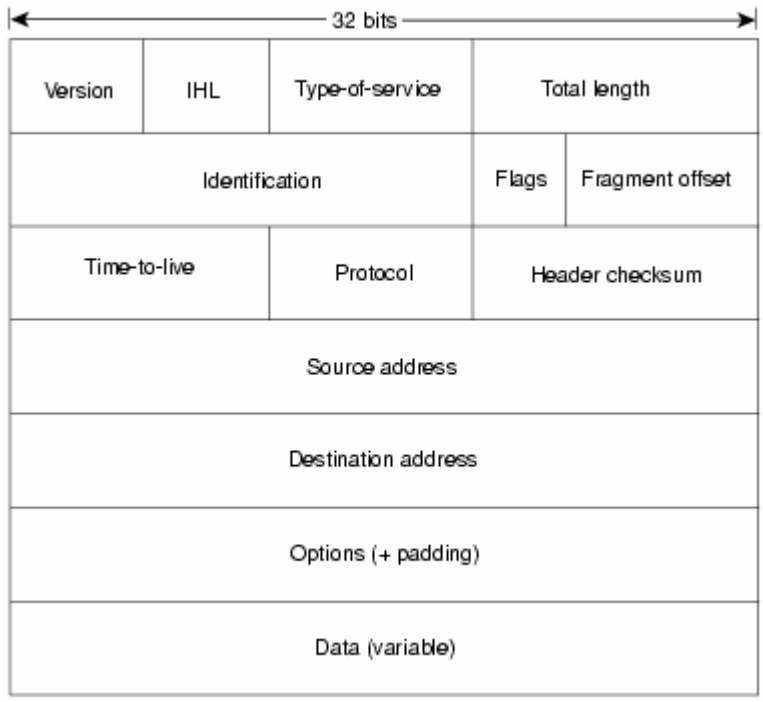
EK-1 TCP Segment Yapısı



Aşağıda TCP başlığı içindeki ana alanlar açıklanmaktadır:

Alan	**İşlevi**
Source Port	Gönderenin TCP portu.
Destination Port	Alanın (hedefin) TCP portu.
Sequence Number	TCP segmenti içindeki birinci baytın sıra numarası.
Window	TCP ara bellek (buffer) alanının şu anki mevcut büyüklüğü.
TCP Checksum	TCP header ve TCP datanın bütünlüğünü kontrol etmek için kullanılır.

EK-2 IP Paket Yapısı



Aşağıda IP başlığı içindeki ana alanlar açıklanmaktadır:

IP Header alanı	**İşlevi**
Source Address	Kaynak verinin IP adresi.
Destinetion Address	Gideceği yerin IP adresi.
Identification	Bir spesifik IP datagramını tanımlamak için kullanılır.
Protokocoll	Paketlerin TCP, UDP, ICMP ya da diğer protokollerle iletişimi ile ilgili.
Checksum	IP header'ın bütünlüğünü kontrol etmek için kullanılan basit bir matematiksel hesaplama.
Time-to-Live (TTL)	Datagramın dolaşacağı network sayısını belirler. TTL sayesinde paketlerin sürekli olarak dolaşması engellenir.

EK-3 Pompa ve Tank Otomasyon Sistemi Hata Kodları

Tablo Ek-3.1: Pompa ve Tank Otomasyon Sistemi Hata Kodları (1.kısım)

SıraNo	Kod	Açıklama	Aylık ortalama oluşma adedi
1	0001	Sanal POS iletişiminde hata	0
2	1002	İşlem tipi veritabanına yazılamadı (Zaman aşımı)	7
3	1202	Kredi kart tanımı kontrol edilirken hata oluştu	1
4	1204	Satış bilgileri verilen onaydan farklı	11
5	1207	Müşteri için tanımlı aktif PAN bulunmuyor	1164
6	1208	Kredi kartı onay alımında hata oluştu (zaman aşımı)	1
7	1210	Tanımsız VIU(Araç Tanıma Ünitesi)	2317
8	1211	Onay verilmediği halde sonlandırılan satış(Reddedilir)	3
9	1212	VIU kara listede	262
10	1213	VIU için tanımsız ürün	1280
11	1214	Onay ve işlem sonu bilgileri işleşmiyor	2
12	1215	Kurumsal filoya satış izni verilmedi	532
13	1217	VIU aktif değil	27
14	1218	Müşteri aktif değil	539
15	1220	Miktar hatası	0
16	1221	Hatalı Plaka	1016
17	1223	Araç kotası tükenmiş	513
18	1225	Araç/filo için özel gün kısıtı bulunuyor	21
19	1226	Araç/filo için resmi tatil kısıtı bulunuyor	11
20	1227	Araç/filo için gün saat kısıtı mevcut	466
21	1228	İlgili istasyondan araç/filo yakıt alamaz	215
22	1229	İstasyon banka tanımı bulunmuyor	96
23	1230	Limit tükendi	11
24	1231	Özel limit tükendi	0

Tablo Ek-3.2: Pompa ve Tank Otomasyon Sistemi Hata Kodları (1.kısım)

SıraNo	Kod	Açıklama	Aylık ortalama oluşma adedi
25	1232	Vardiya alım limiti aşıldı	351
26	1233	Haftalık alım limiti aşıldı	323
27	1234	Aylık alım limiti aşıldı	395
28	1235	Yıllık alım limiti aşıldı	18
29	1801	Zaman aşımı	bilinmiyor
30	1802	İşlem sonucu veritabanına yazılamadı (Zaman aşımı)	113
31	1804	Ürün tipi kontrolü tamamlanamadı (zaman aşımı)	2
32	1809	İstasyon banka tanımı kontrol edilirken hata oluştu	1
33	1814	Tüketim limiti kontrol edilemedi (Zaman aşımı)	3
34	1815	VIU(Araç Tanıma Ünitesi) bilgisi kontrol edilirken zaman aşımı oluştu.	2
35	1820	Müşteri bilgisi kontrolü tamamlanamadı(Zaman aşımı)	0
36	1821	Müşteri PAN bilgisi alınamadı (Zaman aşımı)	0
37	2012	Banka tanımı bulunamadı	1
38	2045	Banka iletişim hatası	505
39	2046	Banka cevap mesajı bozuk	2
40	2047	İstasyon banka bilgisi, bankadan onay almadı	3
41	2060	Talep edilen işlemi gerçekleştirmek için yetersiz izin seviyesi	319
42	2062	Geçersiz işlem	48
43	2080	Banka sistem hatası	18
44	2081	Banka finansal kurumu tanımıyor	124
45	2082	Kredi kart bakiyesi yetersiz	774
46	2083	Kredi kart vade aşımı	399
47	2084	Kredi kartı limiti aşıldı	47
48	2085	Çalıntı kredi kartı	168
49	2086	İşlem tipi kart sahibine kapalı	3

EK-4 XML Mesaj Tipleri

Sistem	XML Mesajları	Açıklama
Pompa Otomasyon sistemi	RecordTrnx	Nakit satışların merkeze aktarır
	RecordTrnxResponse	Nakit satış akıbetinin istasyona bildirir
	LiveTest	Merkez sunucuların erişilebilirliğini kontrol etmek için istasyonlar tarafından kullanılır
	LiveTestResponse	Merkez sunucu durumunu istasyona bildirir
	Settlement	İstasyonlardaki satışlarla merkeze yansıyan satışların mutabakatı yapılır
	SettlementResponse	Mutabakat işlem sonucu istasyona bildirilir
	Authenticate	Onay gerektiren satış tiplerinde kullanılır
	AuthenticateResponse	Onay işleminin akıbetini istasyona bildirir
	Authorize	Onay almış ve yakıt verme işlemi tamamlanmış satışların merkeze gönderiminde kullanılır
	AuthorizeResponse	Onay almış ve yakıt verme işlemi tamamlanmış satışların akıbetlerini istasyona bildirir
	SendSMS	Akaryakıt dağıtım şirketi bireysel müşterine yakıt alımı sonrasında SMS göndermek üzere kullanılır
	SendSMSResponse	SMS gönderim talebine verilen cevabı istasyona iletir
	Reversal	Bankalara sanal POS işlemi için talep yapıldığında ve bankadan cevap alınamadığında,bankanın işlemi gerçekleştirmiş olabileceği düşünülerek, tahsilat işlemini geri iade etmesi talep edilir
	ReversalResponse	Merkezin Reversal isteğine verilen cevabı içerir
Tank Otomasyon Sistemi	SendTankStatusInfo	Tank durum bilgisini merkeze iletir
	SendTankStatusInfoResponse	Merkezin istasyona verdiği tank durum bilgisi cevabıdır
	SendTankFillingInfo	Tank dolum bilgisini merkeze iletir
	SendTankFillingInfoResponse	Merkezin istasyona verdiği tank dolum bilgisi cevabıdır
	SendTankLeakInfo	Tank sızıntı durumu merkeze iletilir
	SendTankLeakInfoResponse	Merkezin istasyona verdiği sızıntı durum cevabıdır
	SendStationTankInfo	İstasyonlardaki tankların tamamı için merkeze durum bilgileri iletilir
	SendStationTankInfoResponse	Merkezin tüm tank durumlarına verdiği cevaptır

EK-5 Uygulama Kodları

EK-5.1 Giriş Fonksiyonu C# Kodu

```
private int CalculateNewWeight(int weight, bool IsSuccess)
 {
     if (!successResponse)
         if (weight < Scale)
             return ++weight;
         else
             return weight;
     else
         if(weight>0)
             return --weight;
         else
             return weight;
 }
```

EK-5.2 Dilsel Değişkenlerin Üyelik Derecelerini Hesaplayan C# Kodu

```
    class MemberShipFunctions
    {
        private double scale = 0.25;

        public double Normal(double x)
        {
            if (x > (double)2 * scale)
                return (double)0;
            else
                return (double)1 - (x / ((double)2 * scale));

        }

        public double Minor(double x)
        {
            if (x < (scale / (double)2) || x > (((double)5 /
(double)2) * scale))
                return 0;
            else
                if (x > ((double)3 / (double)2) * scale)
                    return (double)1 - ((x - ((double)3 / (double)2)
* scale) / scale);
                else
                    return ((x - ((double)1 / (double)2) * scale) /
scale);
        }

        public double Major(double x)
        {
            if (x < ((double)3 / (double)2) * scale || x >
((double)7 / (double)2) * scale)
                return 0;
            else
                if (x > ((double)5 / (double)2) * scale)
                    return (double)1 - ((x - ((double)5 / (double)2)
* scale) / scale);
                else
                    return ((x - ((double)3 / (double)2) * scale) /
scale);
        }

        public double Critical(double x)
        {
            if (x < (double)2 * scale)
                return 0;
            else
                if (x > (double)4 * scale)
                    return (double)1;
                else
                    return ((x - (double)2 * scale) / (double)2 *
scale);
        }

    }
```

Ek-5.3 Akıllı Ajan Uygulaması Konfigürasyon Dosyası

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <appSettings>
        <add key="AXIntelligenceDBConnectionString" value="Data
Source=localhost;Initial Catalog=AXIntelligenceDB;Persist Security
Info=True;User ID=*******;Password=********"/>
        <add key="SniffFolder"
value="D:\AxIntelligence\IntelligentSniffer\SniffedData"/>
        <add key="UnsuccessfullDataFolder"
value="D:\AxIntelligence\IntelligentSniffer\UnsuccessfullData"/>
        <add key="ProcessedDataFolder"
value="D:\AxIntelligence\IntelligentSniffer\ProcessedData"/>
        <add key="TicketFolder"
value="D:\AxIntelligence\IntelligentSniffer\Tickets"/>
        <add key="ProcessedDataFolderSize" value="1MB"/>
        <add key="LogFileSize" value="1MB"/>
        <add key="LogFile"
value="D:\AxIntelligence\IntelligentSniffer\ServiceLog\IntelligentAg
ent.log"/>
        <add key="OfflineNetworkNotifier" value="192."/>
        <add key="DebugModeIsOn" value="false"/>
        <add key="SettlementScale" value="30"/>
        <add key="MaxNeuronBuffer" value="1000"/>
  </appSettings>
</configuration>
```

Ek-5-4 Dilsel Aktivasyon Derecesini Hesaplayan C# Kodu

```
        public string GetLevelLinguistic()
        {
            MemberShipFunctions memberShipFunction = new
MemberShipFunctions();

            double maxMemberShipValue = 0;
            double memberShipValue;
            string maxMemberShipFunctionName = "Normal";

            memberShipValue = memberShipFunction.Normal(cls_Level);
            if (memberShipValue > maxMemberShipValue)
            {
                maxMemberShipFunctionName = "Normal";
                maxMemberShipValue = memberShipValue;
            }

            memberShipValue = memberShipFunction.Minor(cls_Level);
            if (memberShipValue > maxMemberShipValue)
            {
                maxMemberShipFunctionName = "Minor";
                maxMemberShipValue = memberShipValue;
            }

            memberShipValue = memberShipFunction.Major(cls_Level);
            if (memberShipValue > maxMemberShipValue)
            {
                maxMemberShipFunctionName = "Major";
                maxMemberShipValue = memberShipValue;
            }

            memberShipValue =
memberShipFunction.Critical(cls_Level);
            if (memberShipValue > maxMemberShipValue)
            {
                maxMemberShipFunctionName = "Critical";
                maxMemberShipValue = memberShipValue;
            }


            return maxMemberShipFunctionName;
        }
```

EK5.5 Nöronu Analiz Eden, Uyarı Detaylarını Hesaplayan C# Kodu

```
        public struct Notification
        {
            public string XMLMethod;
            public string TrnxType;
            public string ClusteredSourceIP;
            public string ClusteredNeuronID;
            public int ClusteredSourceRate;
            public string ClusteredResponseCode;
            public string ClusteredResponse;
            public int ClusteredResponseRate;
            public long TotalTrnxCount;
            public long SuccessTrnxCount;
            public long FailedTrnxCount;
            public long OnlineTrnxCount;
            public DateTime StartDate;
            public DateTime EndDate;
        }

        public void AnalyseNeuron(List<Neuron.NeuronInput> Data, out
String Result, out Notification NotificationParts)
        {
            Result = "";
            double RateToStop = 0.5;
            int position = 0;
            Notification cls_NotificationParts = new Notification();

            cls_NotificationParts.XMLMethod = Data[0].XMLMethod;
            cls_NotificationParts.TrnxType = Data[0].TrnxType;
            cls_NotificationParts.TotalTrnxCount = Data.Count;

            #region calculating SuccessTrnxCount and modifying
Result

            int SuccessCount = 0;

            for (int count = 0; count < Data.Count; count++)
            {
                if (Data[count].IsSuccess) SuccessCount++;
            }

            cls_NotificationParts.SuccessTrnxCount = SuccessCount;

            Result += "Success Trnx Rate: " +
(int)((double)((double)SuccessCount / (double)Data.Count) *
(double)100) + "%\n";

            #endregion

            cls_NotificationParts.FailedTrnxCount =
cls_NotificationParts.TotalTrnxCount -
cls_NotificationParts.SuccessTrnxCount;
            cls_NotificationParts.StartDate =
DateTime.Parse(Data[0].TrnxTime);
            cls_NotificationParts.EndDate =
DateTime.Parse(Data[Data.Count-1].TrnxTime);

            #region clustering StationIp

            List<Station> StationList = new List<Station>();
```

```
            List<String> Station = new List<string>();

            for (int count = 0; count < Data.Count; count++)
            {
                if (!Station.Contains(Data[count].StationIp))
                {
                    Station StationItem;
                    StationItem.IP = Data[count].StationIp;
                    StationItem.ID = Data[count].StationId;
                    StationItem.Count = 1;
                    StationList.Add(StationItem);
                    Station.Add(StationItem.IP);
                }
                else
                {
                    position =
Station.IndexOf(Data[count].StationIp);
                    Station StationItem;
                    StationItem.IP = StationList[position].IP;
                    StationItem.ID = StationList[position].ID;
                    StationItem.Count = StationList[position].Count
+ 1;
                    StationList[position] = StationItem;
                }
            }


            position = GetTopStationIp(StationList);

            cls_NotificationParts.ClusteredSourceIP =
StationList[position].IP;
            cls_NotificationParts.ClusteredNeuronID =
StationList[position].ID;
            cls_NotificationParts.ClusteredSourceRate =
(int)((double)((double)StationList[position].Count /
(double)Data.Count) * (double)100);

            if (((double)StationList[position].Count /
(double)Data.Count) >= RateToStop)
            {
                Result += "Station IP: " +
cls_NotificationParts.ClusteredSourceRate + "% (" +
StationList[position].IP + "[" + StationList[position].ID + "])\n";
            }
            else
            {
                Result += "Station IP: distributed. The biggest
cluster rate is  " + cls_NotificationParts.ClusteredSourceRate + "%
( " + StationList[position].IP + "[" + StationList[position].ID +
"])\n";
            }

            StationList.Clear();
            Station.Clear();

            #endregion

            #region calculating online Transaction Rate

            int OnlineTrnxCount=0;
```

```
            for (int count = 0; count < Data.Count; count++)
            {
                if (Data[count].IsOnline) OnlineTrnxCount++;
            }

            cls_NotificationParts.OnlineTrnxCount = OnlineTrnxCount;

            if ((double)OnlineTrnxCount/(double)Data.Count>=0.5)
                Result +=  "Online Trnx Rate: " +
(int)((double)((double)OnlineTrnxCount/(double)Data.Count)*(double)1
00) + "% \n";
            else
                Result += "Offline Trnx Rate: " +
(int)((double)((double)(Data.Count - OnlineTrnxCount) /
(double)Data.Count) * (double)100) + "% \n";

            #endregion

            #region clustering ResponseCode and Response Description

            List<Response> ResponseList = new List<Response>();
            List<String> ResponseCodes = new List<string>();

            for (int count = 0; count < Data.Count; count++)
            {
                if (
!ResponseCodes.Contains(Data[count].ResponseCode) )
                {
                    Response ResponseItem;
                    ResponseItem.Code = Data[count].ResponseCode;
                    ResponseItem.Description =
Data[count].ResponseDescription;
                    ResponseItem.Count = 1;
                    ResponseList.Add(ResponseItem);
                    ResponseCodes.Add(ResponseItem.Code);
                }
                else
                    {
                        position =
ResponseCodes.IndexOf(Data[count].ResponseCode);
                        Response ResponseItem;
                        ResponseItem.Code =
ResponseList[position].Code;
                        ResponseItem.Description =
ResponseList[position].Description;
                        ResponseItem.Count =
ResponseList[position].Count + 1;
                        ResponseList[position] = ResponseItem;
                    }
            }


            position = GetTopResponse(ResponseList);

            cls_NotificationParts.ClusteredResponseCode =
ResponseList[position].Code;
            cls_NotificationParts.ClusteredResponse =
ResponseList[position].Description;
            cls_NotificationParts.ClusteredResponseRate =
(int)((double)((double)ResponseList[position].Count /
(double)Data.Count) * (double)100);
```

```
            if (((double)ResponseList[position].Count /
(double)Data.Count) >= RateToStop)
                Result += "Response code is clustered, " +
cls_NotificationParts.ClusteredResponseRate + "% of the Transactions
are ended with " + ResponseList[position].Code + "(" +
ResponseList[position].Description + ")" + Environment.NewLine;
            else
                Result += "Response code is not classified. The most
appeared's rate is  " + cls_NotificationParts.ClusteredResponseRate
+ "% of the Transactions are ended with " +
ResponseList[position].Code + "(" +
ResponseList[position].Description + ")" + Environment.NewLine;

            ResponseCodes.Clear();
            ResponseList.Clear();

            #endregion

            NotificationParts = cls_NotificationParts;

        }
```

EK-5.6 XML Dinleyici Yazılımı Konfigürasyon Dosyası

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
      <appSettings>
            <add key="pCapFilter" value="tcp port 80 and (((ip[2:2]
- ((ip[0]&#38;0xf)&#60;&#60;2)) - ((tcp[12]&#38;0xf0)>>2)) != 0)
&#38;&#38; ((dst host 10.4.0.14 &#38;&#38; src net 10.180.0.0/16)
||(src host 10.4.0.14 &#38;&#38; dst net 10.180.0.0/16))"/>
            <add key="RequestingHost" value="10.180.0.3"/>
            <add key="SniffFolder" value="c:\Sniffer\SniffedData"/>
            <add key="LogFile"
value="c:\Sniffer\ServiceLog\AxSniff.log"/>
            <add key="LogFileSize" value="1MB"/>
            <add key="InterfaceIP"  value="10.4.0.14"/>
      </appSettings>
</configuration>
```

ÖZGEÇMİŞ

Muhsin Gemici, 1979 yılında Bakırköy/İstanbul da doğdu. 1996 yılında Şişli Endüstri Meslek Lisesi Bilgisayar Bölümünü tamamladıktan sonra, aynı yıl devlet bursu hakkı kazanarak Yakın Doğu Üniversitesi Bilgisayar Mühendisliği Bölümün de lisans eğitimine başladı. 2000 yılında lisans eğitimini mühendislik fakültesi birincisi olarak tamamladıktan sonra, İ.T.Ü. Fen bilimleri Enstitüsü, Endüstri Mühendisliği A.B.D.na bağlı Mühendislik Yönetimi programında Yüksek Lisans çalışmasına başladı. Akademik eğitimi devam ederken, web tabanlı uygulama geliştiricisi, veritabanı tasarım ve yöneticisi, Linux sistem yöneticisi, OLAP ve karar destek sistemi geliştiricisi olarak çeşitli projelerde görev aldı. Şu anda Petrol Ofisi A.Ş. de otomasyon uzmanı olarak görev yapmaktadır. İyi derecede İngilizce bilmektedir.