EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

(YÜKSEK LİSANS TEZİ)

JAVA İLE YAPAY ZEKÂ MEKANİZMASINA SAHİP BİR AĞ YÖNETİM SİSTEMİ GELİŞTİRİLMESİ

Erkan BİNİCİ

Bilgisayar Mühendisliği Anabilim Dalı

Bilim Dalı Kodu: 619.01.00

Sunuş Tarihi: 05.12.2006

Tez Danışmanı: Yrd. Doç. Dr. Aybars UĞUR

BORNOVA – İZMİR

Erkan BİNİCİ tarafından YÜKSEK LİSANS TEZİ olarak sunulan "Java İle Yapay Zekâ Mekanizmasına Sahip Bir Ağ Yönetim Sistemi Geliştirilmesi" başlıklı bu çalışma E.Ü. Lisansüstü Eğitim ve Öğretim Yönetmeliği ile E.Ü. Fen Bilimleri Enstitüsü Eğitim ve Öğretim Yönergesi'nin ilgili hükümleri uyarınca tarafımızdan değerlendirilerek savunmaya değer bulunmuş ve 05.12.2006 tarihinde yapılan tez savunma sınavında aday oybirliği/oyçokluğu ile başarılı bulunmuştur.

Jüri Üyeleri: **İmza**

Jüri Başkanı : Yrd. Doç. Dr. Aybars UĞUR .....

Raportör Üye: Doç. Dr. Mustafa Murat İNCEOĞLU .....

Üye : Yrd. Doç. Dr. Muhammet G. CİNSDİKİCİ .....

IV

ÖZET

JAVA İLE YAPAY ZEKÂ MEKANİZMASINA SAHİP BİR AĞ YÖNETİM SİSTEMİ GELİŞTİRİLMESİ

BİNİCİ, Erkan

Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü

Tez Yönetici: Yrd. Doç. Dr. Aybars UĞUR

Aralık 2006, 135 sayfa

Bu tez çalışmasında; günümüzde teknolojinin hızla gelişmesi ve ihtiyaçları çeşitlendirmesi sonucu, gelişen bilgi sistemleri ağlarının yönetim ihtiyaçlarını karşılayacak ağ yönetim yazılımları ve bu yazılımlarda yapay zekâ tekniklerinin kullanımı araştırılmıştır. Hiç bir bilgi ağı sistemi sorunsuz olmayacağı gibi yoğun kullanım ihtiyacının bu sistemlerdeki çok dinamik ve hızlı değişmelere ve gelişmelere sebep olması ile birlikte ağ yönetim sistemlerinin gereksinimi daha çok önem kazanmıştır. Tez çalışmasında ağ yönetiminde yapay zekâ yöntemleri kullanılarak çok büyük ağ sistemlerinin insan müdahalesine daha az ihtiyaç duyacak şekilde nasıl yönetilebileceği üzerinde durulmuştur.

Anahtar Sözcükler: Ağ Yönetim Sistemleri, SNMP, Hata Yönetimi, Kullanıcı Yönetimi, Güvenlik Yönetimi, Yapay Zekâ, Yapay Sinir Ağları, Olay Tabanlı Çıkarsama, Alarm Korelâsyonu.

ABSTRACT

IMPLEMENTATION OF A NETWORK MANAGEMENT APPLICATION SUPPORTED BY ARTIFICIAL INTELLIGENCE TECHNIQUES WITH JAVA

BİNİCİ, Erkan

MSc., Fen Bilimleri Enstitüsü

Supervisor: Asst. Prof. Dr. Aybars UĞUR

December 2006, 135 pages

Nowadays communication environment and technology are changing and growing up with a serious speed. These changes and progress, produce new requirements and do variations on use of the technology especially on communication networks. In this thesis, researches are made in solving the network management requirements on such a fast growing and improving communication network world with less human force usage by the support of Artificial Intelligence techniques.

Keywords: Network Management Systems, SNMP, Fault Management, User Management, Security Management, Artificial Intelligence, Artificial Neural Networks, Case Based Reasoning, Alarm Correlation.

VIII

TEŞEKKÜR

Yüksek lisans eğitimim ve tez çalışmam süresince yardımlarını esirgemeyen değerli danışmanım Sayın Yrd. Doç. Dr. Aybars Uğur'a ve hayatım boyunca hep yanımda olan, tez çalışmam süresince de maddi ve manevi desteklerini esirgemeyen aileme ve eşime teşekkür ederim.

İÇİNDEKİLER

İÇİNDEKİLER (DEVAM)

İÇİNDEKİLER (DEVAM)

ŞEKİLLER DİZİNİ

ŞEKİLLER DİZİNİ (DEVAM)

ÇİZELGELER DİZİNİ

SİMGELER VE KISALTMALAR

ISO	: International Organization for Standardization
WWW	: World Wide Web
API	: Uygulama Geliştirme Ara yüzü
URL	: Uniform Resource Locator
YSA	: Yapay Sinir Ağları
OTÇ	: Olay Tabanlı Çıkarsama
OSI	: Open Systems Interconnection
MIB	: Management Information Base
SNMP	: Simple Network Management Protokol
SMI	: Structure of Management Information
OID	: Object Identifier
UDP	: User Datagram Protocol
TCP	: Transmission Control Protocol
IP	: Internet Protocol
YZ	: Yapay Zeka
BBN	: Bayesian Belief Networks
MPLS	: Multiprotocol Label Switching
CMIP	: Common Management Information Protocol
IDS	: Intrusion Detection System

SİMGELER VE KISALTMALAR (DEVAM)

IPS	: Intrusion Prevention System
FW	: Firewall
NIDS	: Network Intrusion Detection System
QoS	: Quality of Service
RSVP	: Resorce Reservation Protocol
OSPF	: Open Shortest Path First
VPN	: Virtual Private Network
ATM	: Asynchronous Transfer Mode
LRS	: Label Switch Router
LER	: Label Edge Router
VCI/VPI	: Virtual Path Idenfifier / Virtual Cannel Identifier
DLCI	: Data Link Connection Identifier
RIP	: Routing Information Protocol
BGP	: Border Gateway Protocol
FEC	: Forwarding Equivalence Class
OSS	: Operational Support Systems
XML	: Extended Markup Language
CLI	: Komut Satırı Ara yüzü
RMI	: Uzak Metot Çağırımı

1 GİRİŞ

Internet büyük bir hızla gelişmekte ve büyümektedir. Bununla beraber toplumların iletişim yapısı da sürekli değişikliğe uğramaktadır. İletişim ve haberleşme çok eski zamanlardan beri sürekli gelişmiş ve günümüz Internet'ine kadar büyük bir yolculuk geçirmiştir. Bu gelişim hala devam etmektedir. Bu büyük gelişimin ana kaynağı Internet'tir. Bir süre öncesine kadar iletişim ağlar telefon şebekeleri, radyo ağlar, televizyon yayınları gibi tek bir hizmet için işletilmekteydiler. Teknolojinin ve internetin gelişimi ile bu tek hizmet iletişim ağları yerine, Internet bilişimin gücünü kullanarak tek bir iletişim ağını birçok uygulama için kullanmaktadır. Bu sistem aynı anda mesajlaşma, genel yayın, ses ve video, gerçek zamanlı paylaşım ve daha birçok uzlaşma gerektiren uygulamayı desteklemektedir. İletişim ve bilişimin bu birlikteliği, iletişim dünyasında büyük bir değişime neden olmuştur ve bu kapsamda farklı iletişim hizmet sektörleri arasındaki sınırlar giderek belirginliğini yitirmeye başlamıştır.

Benzer bir şekilde bu birliktelik; bilişim dünyasına da değişiklikler getirmiştir. Genelde veri işleme cihazı olarak kabul edilen bilgisayar, yerini sayısal asistanlara, Web TV'lere, ağ kameralarına ve iletişim yeteneklerini kullanan diğer aygıtlara bırakmıştır. Bu değişimin en büyük itici gücü Internet bilgisayar ağı ve onunla beraber gelen iç ağ, elektronik ticaret, Internet servis sağlayıcılığı gibi yeni oluşumlardır. Bu da bize gelecekte, iletişim ve haberleşmenin giderek daha büyük bir hızla bilgisayar ağları üzerine kayacağını göstermektedir. Elektronik ticaret gibi gerçek zamanda yapılan ve para transferinin söz konusu olduğu

işlerde; performansın, doğruluğun ve güvenliğin en üst düzeyde olması gerekir. Bu nedenle, bu tip uygulamaların yapılacağı Internet bağlantısı olan iç ağların olduğu her kuruluşta konunun yani bilgisayar ağı yönetiminin öneminin en iyi şekilde anlaşılması, kolay, hızlı ve uygulanabilir olması gerekmektedir.

Bunu sağlayabilmek için bilgisayar ağı yönetimi konusunun bileşenlerinin ve öneminin çok iyi anlaşılması gereklidir. Ayrıca, bilgisayar ağı hizmetlerinin kalitesi, sanal özel ağlar (VPN), Internet hizmetleri mühendislik ve işletme stratejileri ile Internet ve toplum politikaları gibi konular da herhangi bir bilgisayar ağının yönetimini etkileyen faktörleri oluşturmaktadır.

Ağ yönetim işleri genellikte insan müdahalesi ile yapılmaktadır. Fakat günümüz bilgi sistemleri ağlarının hızla büyüyen ve heterojen yapısı daha yoğun uzmanlık ve iş gücü gerektirmeye başlamıştır. Bu nedenlerden dolayı bu tür ağ yönetim işlerini otomatikleştirmek oldukça çok önem kazanmıştır. Bu bağlamda, Ağ sistemlerinin hata yönetimi aktivitelerinde Yapay zekâ teknikleri; problem çözümü ve çıkarsama kabiliyetleri ile öne çıkmaktadır. Yapay zekâ günümüzde çoğu yazılımda önemli bir işlevi yerine getirmektedir. En küçük yazılımda bile çok basitte olsa bir yapay zekâ tekniği kullanılmaktadır. Bu tekniklerin bazıları yıllardır var olan teknikler olmasına rağmen bazıları da yeni sayılabilir.

Bu tezin kapsamında internet ağlarının yönetim ihtiyaçları ve bu ihtiyaçların yapay zekâ teknikleri kullanılarak nasıl karşılanabileceği araştırılmış ve örnek bir ağ yönetim sistemi geliştirilmiştir.

Tezin 2. bölümünde ağ yönetimi konusu ele alınmıştır. Ağ yönetim ihtiyaçları, ağ yönetim protokolleri ve mimarileri üzerinde durulmuştur.

3. Bölümde Yapay Zekâ konusu araştırılmıştır. Yapay Zekâ teknikleri üzerinde detaylı araştırma yapılmıştır. Zaman içerisindeki gelişimi, çeşitleri, modelleri ve kullanım alanları üzerinde durulmuştur.

4. Bölümde ise Ağ yönetim sistemlerinde Yapay Zekânın nasıl kullanılabileceği üzerinde durulmuştur. Bu konuda ağ yönetimde sistemlerinin en önemli işlevi olan Hata Yönetimi üzerinde duruldu. Hata yönetiminde yapay zekânın nasıl kullanılabileceği üzerine yapılan çalışmalar derlenmiştir. Ayrıca diğer ağ yönetim işlevlerinde yapay zekânın nasıl kullanılabileceği üzerinde de kısaca durulmuştur.

Tez çalışmasında geliştirilen ağ yönetim yazılımı, MPLS ağlarında kullanılan Etiket Anahtarlayıcı Yönlendirici (LSR) ağ cihazlarının yönetimi için tasarlanmıştır. Bu nedenle 5. Bölümde MPLS ağları ve bu ağlarda kullanılan ağ cihazları üzerinde durulmuştur.

6. Bölümde gerçekleştirilen ağ yönetim yazılımı anlatılmış, benzer ticari paket programlara değinilmiş, geliştirilen sistem ile farklarına yer verilmiştir.

7. Bölümde ise ağ yönetim sistemleri ve yapay zekâ konularını hedefleyerek geliştirilen bu çalışmadan elde edilen sonuçlar açıklanmıştır.

2 BİLGİSAYAR AĞLARINDA AĞ YÖNETİMİ

1980'li yılların başında bilgisayar ağları yaygınlaşmaya başlamıştır. Kurumlar bilgisayar ağlarının faydalarını gördükçe teknolojinin hızlı gelişmesinin sunduğu faydaları da kullanarak büyük bir hızla mevcut ağlarını büyüttüler ve bunlara yeni bilgisayar ağları eklediler. 1980'li yılların ortalarına gelindiğinde ise bazı şirketler bu hızlı büyümenin ve çeşitli ağ cihazlarını bilgisayar ağlarına eklemiş olmanın getirdiği sıkıntıları hissetmeye başladılar. Bu hızlı büyüme ile ortaya çıkan problemler günlük ağ işletimini etkilediği gibi ileriye yönelik ağ planlamasında da sıkıntılar doğurdu. Her yeni teknoloji o konuda uzman işgücünü gerektirdi. Hatta bu yoğun uzman işgücü gereksinimi birçok şirkette krize dahi neden olmuştu. Dolayısı ile 1980'li yıllarda birçok bütünleşmiş cihazın birlikte yönetiminin yapılabileceği otomatikleştirilmiş ağ yönetiminin arayışları başlamıştır.

Bilgisayar ağı yönetimi, bir bilgisayar ağı tasarımında hedeflenen hizmet kalitesini sağlayan, mükemmel hizmet verme konusunda yoğunlaşabilmek için işletme, performans, kurulum ve kullanıcı yönetimlerini bir araya getiren bir yönetim şeklidir. Günümüze birçok kritik işlem bilgisayar ağları üzerinden gerçek zamanlı olarak yapılmaktadırlar. Bu nedenle bilgisayar ağlarında performansın, güvenilirliğin ve güvenliğin en üst düzeyde olması gereklidir.

Bu başlıkta ISO Bilgisayar Ağı Yönetimi Forumu (OSI Network Management Forum) tarafından geliştirilen ağ yönetimindeki 5 temel işlevsel alanı incelenerek açıklanmıştır. Sözü edilen bu 5 temel alan;

Hata Yönetimi, Yapılandırma Yönetimi, Performans Yönetimi, Kullanıcı Maliyet Yönetimi ve Güvenlik Yönetimi olarak listelenebilir (ISO, 1998).

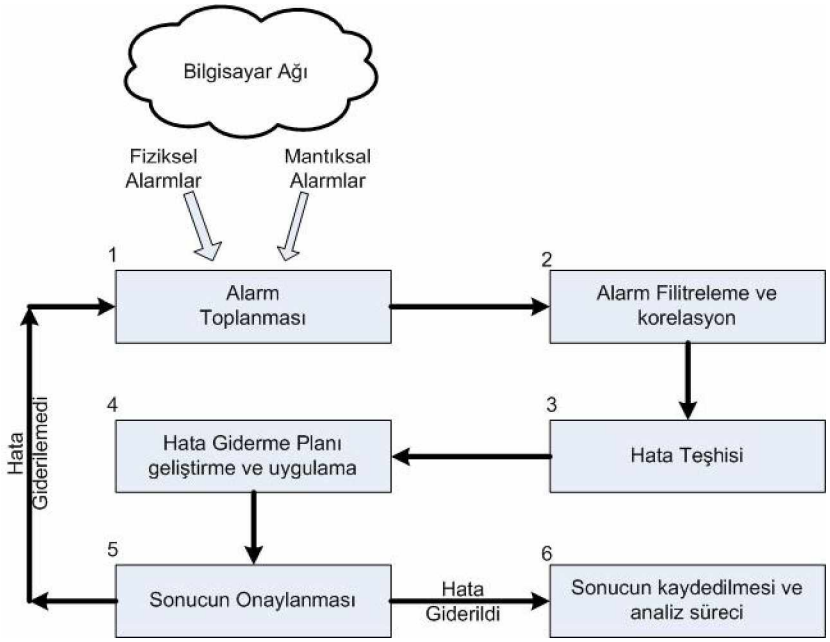
2.1 Ağ Yönetim İşlevleri

2.1.1 Hata Yönetimi

Bilgisayarlar ağlarında cihazlar ve işletim sistemleri bozulabilir, servisler yanlış çalışabilir veya tamamen kesilebilir. Günümüzdeki gelişmiş ve karmaşıklığı artmış ağlarda, bu hatalar çok kısa sürede ağ işleyişinde büyük sorunlar çıkarabilmektedirler. Internet/iç ağ üzerinde çalışan iş yerlerinde, kullanıcılar yaptıkları iş gerçek zamanlı olsun ya da olmasın, yaşadıkları en ufak bir soruna dahi anında çözüm bekleyeceklerdir. Çıkan sorunlar anında çözülmediği takdirde kullanıcıların memnuniyet seviyeleri, sorunlar istedikleri hızda çözümlenmediği için çok çabuk bir biçimde düşecektir. Bu uzun vadede, çalışanların işlerini gerektiği gibi yapamamaları ve müşterilerin de giderek azalması anlamına gelecektir (Shayman et al., 2000).

Hatalar donanımsal ve yazılımsal hatalar olmak üzere iki sınıfta değerlendirilebilirler. Ağ cihazlarında oluşabilecek bu hatalar ağ cihazın yanlış çıktılar üretmesine ve genel olarak ağın yanlış çalışmasına neden olurlar. Donanım hatalarına örnek olarak güç ünitesinin yetersiz kalması ve durması verilebilir. Yazılım hatasına örnek olarak ise yazılımdaki bir hatadan dolayı yavaş çalışma ve yanlış sonuçlar üretmek verilebilir.

Hata yönetim akışı; aşağıda, şekil 2.1 de verilmiştir.



Şekil 2.1: Hata yönetim akışı

Hata yönetim sürecindeki 1. adım alarmların toplanmasıdır. Bu adımda ağ üzerinde olabilecek fiziksel ve mantıksal alarmlar toplanır. Fiziksel alarmlar donanımsal olan problemlerde üretilirler. Hat kopması örnek olarak verilebilir. Genellikle element yöneticisi tarafından raporlanırlar. Mantıksal alarmlar ise daha çok istatistiksel hatalardır. Örneğin ağ performansının düşmesi gibi (Chetan et al., 2000).

Hata yönetim sistemleri, ağ elemanlarından topladığı fiziksel ve mantıksal alarmları dört farklı grupta değerlendirir. Birincisi iletişim alarmlarıdır. İletişim alarmları ağ cihazlarında iletişim kopması veya

kesintisi üzerine üretilirler. İkincisi servis kalitesi alarmlarıdır. Bunlar herhangi bir ağ servisinde performans düşmesi veya kesilmesi sonucunda üretilirler. Eşik değeri aşımı, bant genişliğinin yetersizliği örnek olarak verilebilir. Üçüncü grup alarmlar ise ağ cihazı üzerinde yazılımda oluşabilecek işlem alarmlarıdır. Son olarak da donanımsal problemler sonucu olaşabilecek donanım alarmlarıdır.

2. adımda ise alarmların filtrelemesi ve korelâsyonu yapılır. Filtrelemede amaç alarmların mantıklı kümelere eşlenebilmesi için tekrarlananların elenmesi, benzer olanların bir araya getirilmesidir. Korelâsyondaki amaç ise birbirlerinden kaynaklanan alarmların ilişkilendirilmesidir. Bu sayede hata giderim sürecinde alarmlar mantıksal olarak düzenlenmiş olurlar.

3. adımda ise mantıksal olarak düzenlemesi yapılmış bu alarmların hangi hatalardan dolayı oluşabileceğinin adreslenmesidir. Bu aşamada hata teşhisi konur.

4. adımda ise teşhis edilmiş hata için belirlenmiş giderme yordamının işleme koyulması vardır.

5. adımda giderme planının işe yarayıp yaramadığı kontrol edilir. Eğer hata artık giderilmişse 6. adıma, yani bu yapılan akışın istatistiksel inceleme için kaydedilmesi yapılır, eğer hata giderilememişse 1. adıma yani yeni alarmların toplanması ve incelenmesi sürecine geri dönülür.

Genel olarak bilgisayar ağına sahip her kurumun aşağıdaki işlevleri yerine getiren bir hata yönetimi yaklaşımı olması gerekir:

- Servis ortamındaki hataları anında tanımlama
- Hatayı diğer birimlerden en kısa sürede ayrıştırma
- Bu tür hatalara anında cevap verebilme
- Hatayı en kısa sürede çözme

Hata yönetimi; kullanıcıların ve müşterilerin hatanın ne olduğu, sistemi nasıl etkilediği ve tahmini olarak ne zaman sorunun giderileceği konusunda haberdar edilmesini de sağlamalıdır. Bunlara ek olarak, hata yönetim sistemi, aynı hataların tekrar etmesini de önleyecek şekilde tasarlanmalıdır.

Hata yönetimi, en basit şekliyle, bir hata durumunda yapılması gereken bir dizi süreç olarak tanımlanabileceği gibi; çok daha uzmanlaşmış bir grup tarafından yalnızca hataların izlendiği ve çözümlerinin bulunduğu ayrı bir sistem olarak da tanımlanıp işletilebilir.

Bir hata yönetim sisteminde bilgisayar ağının ağ izleme sistemlerince sürekli izlenmesi, ağda oluşan olayların bu sistemlerce arşivlenmesi ve kontrol edilmesi gereklidir. Ancak bu şekilde, ortamdaki hatalar anında tanımlanabilir. İyi bir hata yönetim sistemi;

- Oluşan hataları oluşan hataları olabildiğince çabuk ve kolay ilişkilendirebilme

- Ortalama hata giderim sürecini

Günümüzde ticari olarak piyasada Bilgisayar Ağı Yönetim programı olarak satılan ürünlerin büyük çoğunluğu bu işi yerine getirmektedir. Fakat yalnızca bu programın satın alınması ve bilgisayar ağının izlenmesi, ağın yönetildiği anlamına gelmez. Bu sistemler ağı izlememize ve hatayı oluştuğu anda fark etmemize yardımcı olurlar. Hata oluştuğunda hatayı fark edip tanımladıktan sonra yapılması gereken ilk şey, hatayı diğer birimlerden en kısa sürede ayrıştırmaktır. Bunu yapabilmek için o ağın kurumuyla ilgili özellikleri ve detayları; tasarımını ve işlevlerini iyi anlamak gereklidir çünkü ancak bu şekilde, hata durumunu bir ağ bileşeni ya da ağ üzerindeki belli bir konumla ilişkilendirmek ve dolayısıyla ayrıştırmak mümkün olabilir. Hata diğer sistemden ayrıştırıldıktan sonra yapılması gereken hataya en kısa sürede cevap verebilmek ve çözümü bulmaktır.

2.1.2 Yapılandırma Yönetimi

Yapılandırma yönetimi, çok genel olarak ağ üzerinde yer alan her cihazın durum bilgisinin yönetimidir. Durum bilgisi altında;

- Ağ topolojisi ve durumu,
- Ağ elemanlarının donanım bilgileri,
- Çalışma yapılandırma bilgileri,
- Kurulum özellikleri,
- Yazılım bilgileri sayılabilir.

2.1.2.1 Ağ topolojisi ve genel durum yönetimi

Ağ durum bilgisini temel şekliyle; ağın mevcut topolojisini, ağın elemanlarının listesini ve bu elemanların bağlantı şekillerinden oluşur. Bu bilgiler ağ durum bilgisinin statik kabul edilen bölümünü oluşturur. Ayrıca bir de ağ durum bilgisinin çalışma durumu adı verilen dinamik bölümü bulunmaktadır. Çalışma durumu, ağ elemanlarının işlerliklerini gösteren bir rapordur. Ağ elemanının çalışıp çalışmadığının ve eğer çalışıyor durumdaysa iş yükünün anlık olarak gösterimidir.

Ağ topolojisi ve işlev durumu yanında ağ durum bilgisi; her ağ elemanının mevcut kurulum özellik tanımlarını da kapsar. Bu tanımlar; işlemci, bellek gibi ünitelerin fiziksel kurulum özelliklerinin yanı sıra ağ cihazı üzerinde çalışan yazılım yapılandırmasının özelliklerini de içerir.

Yapılandırma yönetiminin bir parçası olan çalışma kontrolü, bilgisayar ağ elemanlarının kontrolü demektir. En temel haliyle ağ işlemlerini yapan personelin ağ elemanlarını durdurması ya da başlatması gibi özellikleri içerir. Bunu yapabilmek için çalışma yönetimi araçlarına gereksinim duyulur. Böylece, herhangi bir ağ elemanının kurulması, kurulum parametrelerinin yenilenmesi ve gerektiğinde değiştirilmesi sağlanmış olur.

2.1.2.2 Stok yönetimi

Yapılandırma yönetimi sırasında döküm kontrolünün yapılması ve değişim yönetiminin gerçekleştirilmesi de söz konusudur. Bu konuları

sırasıyla incelemek gerekirse, stok kontrolü; bilgisayar ağı elemanlarından oluşan bir veri tabanının yönetilmesi demektir. Bu veri tabanın da ağ elemanlarının yapılandırma özellikleri, bulundukları konumlar, geçirdikleri değişimler ve oluşan hatalar bilgi olarak saklanır. Bu tür bir stok yönetimi yalnızca ticari açıdan değil, hata yönetimi açısından da önemlidir.

2.1.2.3 Değişim yönetimi

Değişim yönetimi ise sürekli büyüyen ve değişen bir bilgisayar ağı açısından çok önemlidir. Bu sürekli değişimin etkilerini biraz olsun azaltabilmek için değişim yönetimi büyük önem taşır. Bunu sağlayabilmek için yapılması gerekenler kısaca şöyle özetlenebilir: Fiziksel hatların detaylı ve güncel kayıtlarının tutulması, yapılacak değişikliklerin önceden planlanması, ağ elemanlarını kurmak için gereken komut ve parametrelerin birer kopyalarının saklanması, yapılan değişikliklerin gözden geçirilmesi.

2.1.3 Performans Yönetimi

Bilgisayar ağı yönetiminin en çok mücadele gerektiren alanlarından biri de performans yönetimidir. Bunun için ağ ve servislerine ilişkin performans ölçütlerinin saptanması, sonra da bunların farklı ağ elemanlarından toplanan veriyle karşılaştırılması gereklidir. Bu alandaki etkinlikler genel olarak şu başlıklar altında incelenebilir:

- Veri toplama

- Verinin mevcut performans ölçütleri ve eğilimleri için incelenmesi
- Performans için eşik değerlerin saptanması
- Saptanan eşik değerleri ile sürekli toplanan performans verilerinin karşılaştırılması
- Performans bilgileri ışığında kapasite planlaması yapılması

Performans yönetiminin amacı; bilgisayar ağı yöneticisinin ağ da ortaya çıkan sıkışmaları, servislerde meydana gelebilecek hizmet düşüşünü en aza indirmek ve tüm kullanıcılar için tutarlı bir performans düzeyi sağlamaktır. Bilgisayar ağı işletmeni, performans yönetimi araçlarını kullanarak ağdaki iletişimin, anahtarlamanın ve servis verilen ortamların performansını izler. Onun topladığı veriler yönetim tarafından performans problemlerini çözmek için kullanılır.

Veri toplamak için performans yönetiminin en temel kaynağı SNMP'nin (Simple Network Management Protocol) performans ölçütleri için sorgulanmasıdır. Bu sorgulamada verilen bir zaman sürecinde bilgisayar ağı yöneticisinin gerekli bulduğu veriler toplanır. Bunu sağlamak için yöneticinin hangi ağ elemanlarından ne kadar süreyle hangi değişkenlerin toplanacağına karar vermesi gereklidir (Bohoris et al., 2001).

2.1.4 Kullanıcı Maliyet Yönetimi

Kullanıcı hesapları yönetimi, bilgisayar ağı kullanıcılarının aldıkları tüm hizmetlerin ki; bu en genel anlamıyla kullandıkları bilgisayar ağı uygulamalarıdır, izlenmesi ve gerekli bilgilerin tutulmasıdır. Bu alanda yapılan en belirgin iş kullanılan servislerin örneğin; Internet Servis Sağlayıcı konumundaki bir işletmede müşteriler için ücretlendirilmesidir. Diğer kurumlarda da kullanıcılardan gerçekte ücret alınmasa dahi hangi kullanıcının ağı ne için, ne kadar ve nasıl kullandığının izlenebilmesi için benzer yöntemlerin kullanılması hararetle önerilmektedir. Böylece, bilgisayar ağının hizmetlerinin ve kaynakların nasıl kullanıldığı hakkında bilgi sahibi olmak mümkün olacaktır. Bu da kapasite planlamasında daha gerçekçi tahminler yapmayı mümkün kılacaktır. Bu gibi durumlarda her kullanıcının ağdaki tüm hizmetleri ne kadar süreyle ve ne zamanlar kullandığı bilgisinin izlenmesi en çok kullanılan yöntemlerden biridir.

2.1.5 Güvenlik Yönetimi

Internet güvenli bir ortam değildir. Bu nedenle Internet'e bağlı olan iç ağlar nedeniyle hem ilgili kurumlar hem de kurumların kullanıcıları her an Internet'teki saldırılara maruz kalabilirler ve bu saldırıların bir kısmı başarılı da olabilir. Bu gibi saldırılara anında cevap verebilmek için güvenlik yönetimi konusunda bilgi sahibi olmak ve bunların uygulanabilmesi için politikalar oluşturmak gereklidir.

Geniş bir alana yayılmış büyük bir bilgisayar ağında güvenliği sağlayabilmek oldukça zor bir iştir. Bu ancak ağa bağlı her ekipmanın güvenlik ayarlarının ve yönetiminin düzenli bir şekilde yapılması ve birçok güvenlik sisteminin işbirliği içerisinde çalıştırılması ile sağlanabilir.

Temel ağ güvenlik sistemlerine güvenlik duvarları, saldırı tespit sistemleri, saldırı önleme sistemleri, ağ şifreleme sistemleri ve güvenlik yama yönetim sistemleri verilebilir. Büyük bir bilgisayar ağında güvenliğin sağlanabilmesi için tüm bu sistemlerin uygun konumlarda yerleştirilmesi ve yönetilmesi gerekir.

Ayrıca ağ güvenliği için her bir donanımın bireysel güvenliğinin de sağlanması çok önemlidir. Ağ donanımlarının yönetimi için yapılan bağlantıların şifreli olması, kullanıcı doğrulaması ve yetkilendirilmesi, kullanıcı hareket günlüklerinin yapılması ve yönetilmesi gereklidir.

Gittikçe karmaşıklığı artan ağ sistemlerin bu kadar çok kontrolün birlikte insan kontrolü ile yapılabilmesinin yetersizliği nedeni ile birçok güvenlik sistemi içerisine yapay zekâ yetenekleri eklenmiştir. Bu sayede saldırılar anında tespit edilmeye çalışılmakta ve anında gereken önlemler alınmaya çalışılmaktadır.

2.2 Ağ Yönetim Mimarisi ve Protokolleri

Ağ yönetim sistemleri iki önemli eleman içerirler. Bunlar;

- Ağ yöneticisi ve

- Yönetilen aygıt üzerindeki ajandır.

Ajanlar yönetilen ağ cihazı üzerinde çalışırlar ve cihaz hakkındaki bilgilere direk erişimleri vardır. Burada yönetilen aygıta çalışmamızda bir yönlendiriciyi örnek verebiliriz. Diğer örneklere Windows işletim sistemi, sunucu bilgisayar, HUB verilebilir.

Ajanın yönetilen cihaz üzerinde eriştiği bilgiler;

- Donanım parametreleri,
- Yazılım bilgileri,
- Performans verileri,
- Yapılandırma parametreleri gibi

Direk olarak cihazı işletimsel açıdan ilgilendiren bilgileri sayabiliriz. Bu bilgiler ajan üzerinde sanal bir veri tabanı içerisinde nesne sıradüzeni şeklinde yönetim veri tabanı (MIB) diye isimlendirilen yapılarda tutulurlar.

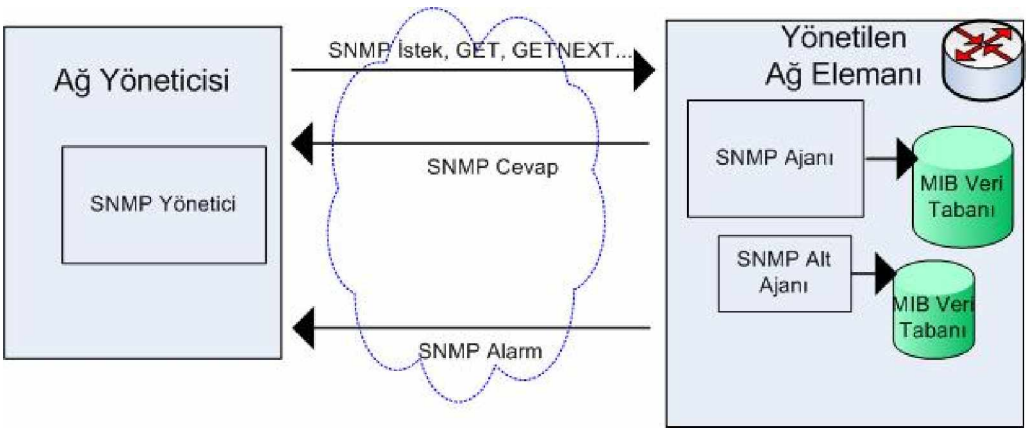
Ağ yönetim protokolleri içerisinde ilk geliştirilen Basit Ağ yönetim protokolüdür (SNMP). Bilgisayar ağları ilk gelişmeye başladığı zamanlarda bu ağların yönetim ihtiyacının olduğu görülmüştür. Fakat henüz ağ sistemleri olgunlaşmadığı ve standartlar oturmadığı için tüm işlevleri ile tam bir yönetim protokolü yerine üreticiler tarafından kolay uygulanabilecek basit bir protokol geliştirilmiştir. Bu basit protokol uzun yıllar birçok üretici tarafından uygulanmış ve desteklenmiştir. Sonradan RMON ve CMIP gibi ağ yönetimi açısından daha kabiliyetli protokollerde geliştirilmiştir. Fakat bu protokoller karmaşık olmaları,

daha fazla sistem kaynağı tüketmeleri çok sonradan uygulamaya girmiş olmaları gibi nedenlerden dolayı beklenen ilgiyi görememişlerdir.

Bu çalışma kapsamında yaygın kullanımı ile her türlü ağ elemanı tarafından standart olarak desteklenen SNMP protokolünü kullanan bir ağ yönetim sistemi geliştirilmesi yapılmıştır.

2.2.1 Basit Ağ Yönetim Protokolü (SNMP)

SNMP protokolü ise temel olarak ağ yöneticisinin, ağ elemanı üzerindeki bu yönetim bilgilerine ulaşımını ve yönetimi sağlamaya yarar. Bu sistemin mimarisi genel olarak şekil 2.2 de verilmiştir.



Şekil 2.2: SNMP yönetim mimarisi

Tipik bir SNMP Ajanı şunları barındırır:

§ SNMP protokolüne tümüyle uygular.

- § MIB veri tabanında tuttuğu yönetim verilerini saklar ve istek olduğu zaman iletir.
- § Asenkron bir şekilde yöneticiye alarm gönderebilir.
- § SNMP desteği olmayan bir ağ elemanı için vekâlet yönlendirme yapabilir.

Tipik bir SNMP yöneticisi, (ağ, elaman yönetici) ise

- § SNMP protokolünü tümüyle uygular.
- § Yönetim işlevlerini içerir.
- § Ajanları SNMP GET ve GetNext komutları ile sorgulayabilir.
- § Ajanlar üzerindeki değişkenleri SET komutu ile belirleyebilir.
- § Ajanlardan gelen alarmları algılayabilir.

SNMP protokolünün SNMPv1, SNMPv2 ve SNMPv3 olmak üzere üç sürümü bulunmaktadır. Temelde tüm bu sürümlerin protokol işlemleri aynıdır. Tüm sürümler endüstride kullanılmaktadır fakat en yaygın olarak en eski sürüm olan SNMPv1'dir. SNMPv1 de en temel eksiklik güvenlik anlamındadır. Neredeyse güvenlik yok denebilir. Tüm mesajlaşma açık metin olarak yapılmaktadır ve ağı dinleyen herhangi biri tarafından kolaylıkla izlenebilirler.

SNMPv2 de MIB ve PDU yapısında bazı değişiklikler olmuştur. Bunun yanında en önemli değişiklik güvenlik anlamında gelmiştir. Bu sürümde mesaj trafiği MD5 ve DES algoritmaları ile

şifrelenebilmektedir. Aslında sürüm 2'nin SNMPv2c ve SNMPv2u olarak iki farklı alt sürümü de bulunmaktadır. SNMPv2'nin yaygın olarak kullanılamayışının temel sebeplerinden birisi getirdiği karmaşıklıktır. Karmaşıklığı nedeni ile SNMPv1 kullanımı devam etmiştir.

SNMPv1 deki güvenlik açıkları ve SNMPv2 deki karmaşıklık nedeni ile yeni bir sürüm, yani SNMPv3 geliştirilmesi kararlaştırılmıştır. Güvenlik, doğrulama aşamasında HASH ve zaman damgaları kullanılarak artırılmıştır. Ayrıca SNMPv2 de bulunan karmaşıklık da SNMPv3 de azaltılmaya çalışılmıştır.

Tüm bu yapılan geliştirmelere rağmen endüstride en yaygın olarak kullanılan SNMP protokolü SNMPv1'dir.

2.2.1.1 Yönetim Bilgisinin Yapısı

Ağ yönetiminde yönetici, ajan açısından yönetilen ağ elemanı erişilebilir olmalıdır. Bunun yanı sıra ağ elamanı için yönetim bilgileri bir yerde saklanıyor erişilebiliyor ve istendiğinde değiştirilebiliyor olmalıdır. SNMP protokolü aslında bu erişim ve değiştirilme işlemlerini yapabilmek için tasarlanmıştır. Yönetim bilgisinin yapısı RFC 1155 de OSI SMI (Structure of Management Information) tanımlanmıştır (Pras, 2000).

SMI yapısı isimlendirmeyi ve erişilecek bilgileri tanımlar. Bu tanımlamaya göre yönetim veri tabanı (MIB) içerisindeki her yönetilecek nesne bir isme, sözdizimi ve kodlamaya sahip olmalıdır.

- İsim, nesne tanımlayıcısı OID (Object Identifier) olarak açılanır ve her nesne bir OID ye sahiptir.
- Sözdizimi nesnenin veri tipini tanımlar, Integer, String gibi.
- Kodlama nesnenin erişim ve iletim sırasında nasıl formatlanacağını tanımlar.

SNMP MIB dosyalarında kullanılan sözdizimi ASN1 formatındadır. ASN1 OSI içerisinde birçok şekilde tanımlama amaçlı olarak kullanılmıştır. Örneğin birçok RFC dokümanı da ASN1 formatındadır.

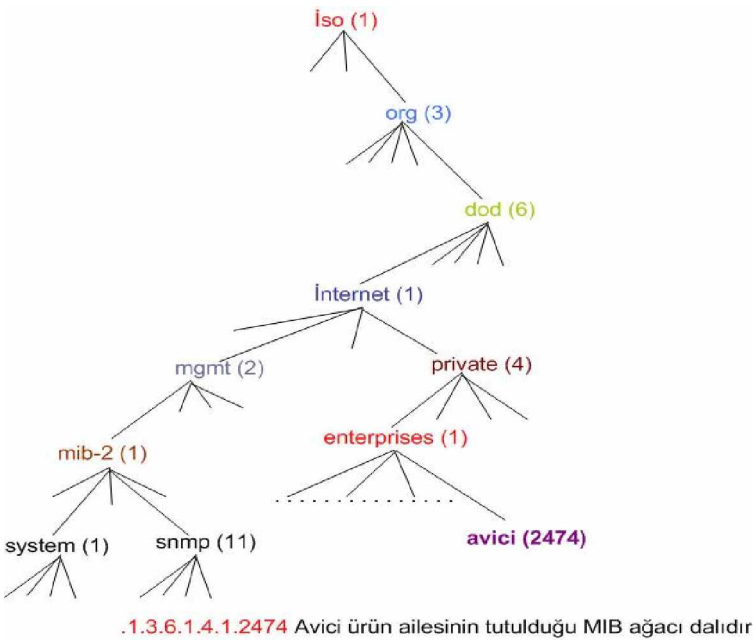
MIB içerisinde her ağ elemanı veya yönetilecek birimi nesne olarak tanımlanır ve bir isme sahiptir. Bu isimlendirme aralarında noktalar olan sayı dizileri şeklindedir. Örneğin 1.3.6.1.2.1.1.1.0 “mgmt” alt ağacı içerisindeki “system” grubunu temsil eder.

2.2.1.2 Yönetim Bilgi Tabanı (MIB)

MIB, ağ elemanının yönetilmek için kullanılacak tüm parametrelerinin birlikte tutulduğu tanımlar bütünüdür. Her ağ elamanı MIB veri tabanı içerisinde tanımlanan parametrelerinin değerlerinden oluşan bir veri tabanı tutarlar (Cisco Systems University, 2006). Bu gerçek bir veri tabanı olmayabilir, uygulamaya göre hafızada, dosyada

veya veritabanında tutulabilir. MIB tanımı RFC 1155 ta verilen SMI tanımına uyar. En son tanımı yapılmış olan internet MIB tanımı MIB-II olarak adlandırılır.

Şekil 2.3 de örnek bir MIB ağacı görülebilir. Burada üretici firmalar için ayrılmış olan ağaç "iso.org.dod.internet.private.enterpise" altıdır. Buradan sonra her ağ elemanı üretici firmanın kendine özel sahip olduğu ağaç dalı altında geliştirdiği ürünler hakkındaki yönetim parametrelerinin tanımları durur. Şekildeki 1.3.6.1.4.1.2474 düğümü tez çalışması için kullanılan LSR ağ elemanının üreticisi olan Avici firmasına ait ağaç düğümüdür.



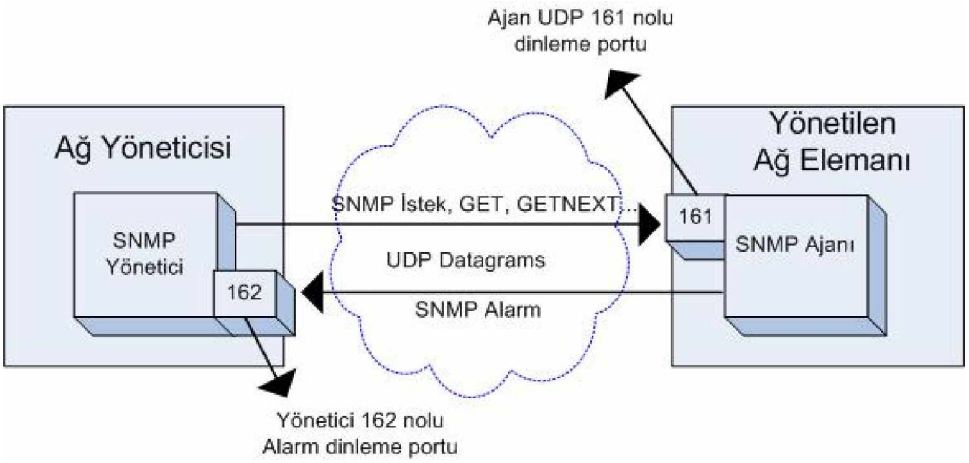
Şekil 2.3: Örnek bir MIB ağacı

2.2.1.3 SNMP Protokol Komutları

SNMP daha önce de söylediğimiz gibi yönetici, ajan ilişkisine dayanır. Basit diye adlandırılır çünkü ajan üzerinde çok küçük bir yazılım bulunması yeterlidir. SNMP de tanımlanmış çok basit ve küçük bir komut dizisi vardır. Bunlar;

- § GET: Bir parametre değerinin ajandan alınması.
- § GETNEXT: Bir sonraki parametre değerinin ajandan alınması
- § GetResponse: Parametre değerinin yöneticiye iletilmesi.
- § SET: Bir parametre değerinin ajan yönetim veri tabanında güncellenmesi
- § TRAP: Yönetilen ağ cihazında oluşan bir alarmın yöneticiye senkronize olmayan bir şekilde iletilmesi.

Burada GET, GETNEX ve SET komutları yöneticiden ajana iletilen komutlardır. Ajan ise yöneticinin gönderdiği komutlara cevap dışından bir de TRAP ile yöneticiye alarm gönderir. Aşağıdaki şekilde SNMP komutlarının UDP üzerinden yönetici üzerinde 162, ajan üzerinde 161 numaralı kapı ile karşılandığı gösterilmektedir. Tüm SNMP trafiği SNMP Datagram paketleri olarak iletilmektedir.



Şekil 2.4: SNMP komutlarının işleyişi

2.2.2 Uzak Ağ Yönetim Protokolü (RMON)

RMON protokolü SNMP ye alternatif bir protokol değildir. Temel olarak SNMP protokolüne dayanır. Ağ elemanları üzerindeki MIB bilgisi alt yapısı SNMP ile aynıdır ve ağ yöneticisi ile yapılan haberleşme SNMP protokolü üzerinden yapılır. RMON protokolü yukarı tanımladığımız temel ağ yönetim işlevlerinin daha efektif yapılabilmesi için SNMP'nin yeniden tasarlanması ile genişletilmiş bir hali olarak oluşturulmuştur.

SNMP sadece üzerinde çalışmakta olduğu cihaz hakkında bilgiler vermektedir. Fakat ağ yönetimini daha işlevsel olarak yapabilmek için cihazın diğer ağ elemanları ile olan etkileşimi ve bunların istatistiksel bilgilerini de takip etmek gerekmektedir. RMON SNMP de ki bu açığı kapatmak üzere geliştirilmiştir. Ana amaç ağ elemanının ve dolayısı ile ağın yönetimini uzaktan yapabilmeyi sağlamaktır (Pras, 2000).

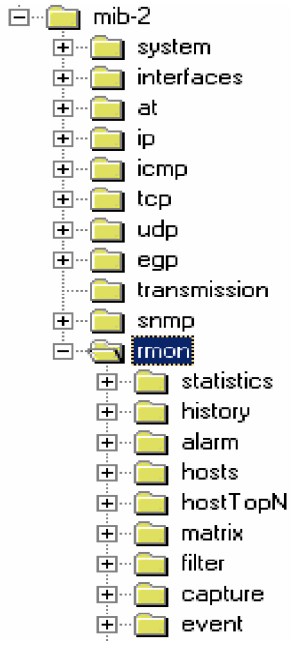
SNMP'nin tersine birçok ağ yönetim işlevi RMON sayesinde ağ elemanı üzerinde yönetilebilmektedir. SNMP istemci / sunucu mimarisine dayalıdır. Yani veriler SNMP ile ağ donanımları üzerinden ağ yönetim elemanına yollanır ve burada değerlendirilirler. Fakat RMON protokolünde ise verilerin ağ elemanına gönderilmeden cihaz üzerinde tutulması ve istatistiksel amaçlı olarak kullanılması mümkün kılınmıştır. Bu özellik de ağ yönetim katmanı üzerindeki veri işleme yükünü azaltmaktadır.

Günümüzde birçok ağ anahtarı üreticisi cihazlarına RMON protokolünü uygulamıştır. (Cisco Systems University, 2006).

RMON protokolünün birinci sürümü RFC1757 içerisinde tanımlanmıştır. Bu RFC de temel olarak Ethernet ağları hedef alınmıştır. RFC1513'te ise Token Ring ağları için gerekli adreslenmeler tanımlanmıştır. RMON Sürüm 2 de ise anahtar ve hızlı Ethernet ağları için gelişmiş tanımlamalar eklenmiştir.

2.2.2.1 RMON Ağ Bilgisi Grupları

Yukarıda belirtildiği gibi RMON protokolü SNMP yi genişletme amaçlı geliştirilmiştir. Bu amaçla protokol içerisinde SNMP MIB ağaç yapısı altında MIB–2 içerisinde aşağıda görüldüğü gibi 9 farklı grup tanımlanmıştır (Şekil 2.5 RMON MIB Ağaç yapısı).



Şekil 2.5: RMON MIB ağaç yapısı

Bu 9 farklı gruba değinirsek;

1- İstatistik (Statistic) : İzlenen Ethernet ara yüzleri üzerindeki istatistiksel bilgileri verir.

2- Geçmiş (History) : Bu tablo üzerinden geçmişe yönelik istatistiksel bilgilerin nasıl toplanması gerektiğine yönelik tanımlamalar yapılır.

3- Alarm: Bu tablo ile alarmların eşik değeri aşım kontrolleri yapılır.

4- Sunucu (Host): Sunucu tablosu üzerinde ağ üzerinde keşfedilmiş sunuculara yönelik tanımlamaya yönelik ve istatistiksel bilgiler tutulur.

5- İlk N Sunucu (HostTopN) : Bu tablo ile herhangi bir statiksel bilgisine göre listelenmiş ilk N sunucu bilgisi bulunur.

6- Matris (Matrix) : Matris tablosu üzerinde iki adres arasındaki iletişime ait istatistiksel bilgiler tutulur.

7- Filtre (Filter) : Filtre tablosu ile paketlerin belli bir değere eşlenmesi ile filtrelenmesi sağlanır. Filtre ile belirlenen paketin yakalanması sonucu sonraki trafiğin kaydedilmesi veya bilgilendirme olay bilgisi gönderilmesi sağlanabilir.

8- Paket Yakalama (Packet Capture) : Bellenen bir kanaldan akan paketlerin kaydedilmesi için kullanılır.

9- Olay (Events) : Olay tablosu ile bilgilendirme olaylarının yaratılması ve gönderilmesi kontrol edilir.

2.2.3 Genel Yönetim Bilgi Protokolü (CMIP)

CMIP protokolü de SNMP de bulunan eksiklikleri kapatmak ve ağ yönetim sistemlerinin yeteneklerini daha fazla desteklemek amaçlı geliştirilmiştir. Ağ yönetim sistemi ve ağ elemanı arasında bilgi alışverişini tanımlayan OSI tabanlı bir ağ yönetim protokolüdür. Protokol içerisinde erişim kontrolü, yetkilendirme ve güvenlik kayıtları desteklenmektedir.

Birçok eklentisi olması sebebi ile CMIP protokolü SNMP ye oranla daha karmaşıktır. Bu nedenle SNMP tüm internet ağına yayılmasına

karşın CMIP sadece belli telekomünikasyon servis sağlayıcıları tarafından kullanılmaktadır.

Yönetim bilgisi ağ yöneticisi ve ağ elemanı arasında yönetim nesneleri üzerinden paylaşılır. Ağ yöneticisi aşağıdaki işlemler ile ağ elemanı üzerinde işlem başlatabilir.

- ACTION: Yönetim nesnesini içerisinde tanımlanan aksiyonun istenmesi
- CANCEL_GET: Gönderilmiş bir istek komutunun iptal edilmesi
- CREATE: Bir yönetim nesnesinin yaratılması
- DELETE: Bir yönetim nesnesinin silinmesi
- GET: Bir yönetim nesnesi değerinin istenmesi
- SET: Bir yönetim nesnesinin değerinin belirlenmesi

Ağ elemanı EVENT_REPORT işlemini kullanarak ağ elemanı tarafından daha önceden belirlenmiş bir işlemin başlatılmasını sağlayabilir. Ağ yönetim birimi ACTION komutu ile daha önceden herhangi bir durumun sağlanması halinde bunu kendisine bildirmesini isteyebilir. Bu bilgilendirme işlemi SNMP de ki TRAP e karşılık gelen EVENT_REPORT ile yapılır.

CMIP in SNMP ye göre başlıca avantajları;

- CMIP değişkenleri ile bilgi taşımanın dışında işlem gerçekleştirilmesi yapılabilir. Bu SNMP de mümkün değildir.
- CMIP kendi içerisinde bütünleşik güvenlik altyapısına sahip tasarlanmıştır. Yetkilendirme, erişim kontrolü ve güvenlik kayıtları bulunmaktadır. Bu sayede SNMP ye göre daha güvenlidir. SNMP protokolünde ise sürüm 3 de güvenlik açıklarını kapatacak eklentiler yapılmıştır.
- CMIP değişkenleri ile ağ yönetim sistemi ağ elemanı üzerinde birden fazla işlemi tek bir istek ile gerçekleştirebilir.
- SNMP ye oranla CMIP daha iyi raporlama yetenekleri bulunmaktadır.

Bu avantajlara karşın CMIP protokolü SNMP ye oranla daha karmaşık bir yapıya sahiptir. Bu daha çok sistem kaynağı tüketmesini doğurmuştur ve ayrıca yönetim sistemini programlamak için daha uzman insan kaynağına ihtiyaç duyulmaktadır.

3 YAPAY ZEKÂ VE YAPAY SİNİR AĞLARI

3.1 Yapay Zekâ

Yapay Zekâ terimi Stanford Üniversitesi'nde Profesör olan John McCarthy tarafından ortaya atılmıştır.

Yapay zekâ'nın tanımını yapmak önemli olduğu kadar da zordur. Çünkü yapay zekâ çok kapsamlı bir konu olup pek çok alt alan içermektedir ve alt alanı olarak düşünülemeyecek pek çok alanla da ilintilidir. Yapay zekânın içerdiği ve ilintili olduğu tüm alanları kapsayacak şekilde yapılmaya çalışılacak tek bir tanım ya çok uzun olacaktır, ya çok soyut olacaktır yada eksik olacaktır. Dolayısıyla yapay zekânın tanımını yapmaya çalışan kişiler genellikle kendi uzmanlık alanlarını ön plana çıkaran tanımlar yapmışlardır.

Bu nedenle tek bir tanım yapmak yerine, birden çok tanıma yer vererek zihnimizde yapay zekâ kavramının daha eksiksiz oluşmasını sağlamak en uygunu olacaktır:

- Yapay zekâ yapay bir varlığın (genellikle bir bilgisayar) sergilediği zekâdır.
- Yapay zekâ bir makine yada insan eliyle üretilmiş otonom bir sistem kullanarak insan zekâsının benzetimini yapmaya çalışan bir araştırma alanıdır.

- Yapay zekâ, insanın düşünme yöntemlerini analiz ederek bunların benzeri yapay yönergeleri geliştirmeye çalışan araştırma alanıdır.
- Yapay zekâ, canlılarda (özellikle insanlarda) bulunan algılama, öğrenme, çoğul kavramları bağlama, düşünme, fikir yürütme, sorun çözme, iletişim kurma, çıkarım yapma ve karar verme gibi yüksek bilişsel işlevleri ve otonom davranışları sergilemesi beklenen yapay bir sistemdir.
- Slage'e göre yapay zekâ; sezgisel programlama temelinde olan bir yaklaşımdır.
- Yapay zekâ; insanların yaptıklarını bilgisayarlara yaptırabilme çalışmasıdır.
- Genesereth ve Nilsson'a göre yapay zekâ, akıllı davranış üzerine bir çalışmadır. Ana hedefi, doğadaki varlıkların akıllı davranışlarını yapay olarak üretmeyi amaçlayan bir kuramın oluşturulmasıdır.
- Tesler'e göre yapay zekâ; şu ana kadar yapılamayanlardır.
- Axe'a göre ise yapay zekâ; akıllı programları hedefleyen bir bilimdir.

Bu programlar aşağıdakileri yapabilmelidir.

- İnsanın düşünmesini taklit ederek karmaşık problemleri çözebilmeli.

- Yorumlarını açıklayabilmeli, yeni bir durum karşısında kişiye yanıt verebilmeli.
- Öğrenerek uzmanlığını geliştirebilmek ve eski bilgilerini yenilerle uyumlu biçimde kullanarak bilgi tabanını genişletmek.

Yapay zekâ özetle; bir bilgisayarın ya da bilgisayar denetimli bir makinenin, genellikle insana özgü nitelikler olduğu varsayılan akıl yürütme, anlam çıkartma, genelleme ve geçmiş deneyimlerden öğrenme gibi yüksek zihinsel süreçlere ilişkin görevleri yerine getirme yeteneği olarak tanımlanmaktadır.

Zekâ, insanın düşünme, akıl yürütme, nesnel gerçekleri algılama, kavrama, yargılama, sonuç çıkarma, soyutlama, öğrenme yeteneklerinin tümü. Ayrıca Soyutlama, öğrenme ve yeni durumlara uyma gibi yetenekler de zekâ kapsamı içindedir. Yapay zekâ ise, bu özelliklere sahip organik olmayan sistemlerdeki zekâdır.

Yapay zekâ dört kategoriye ayrılabilmektedir (Russel, 1995):

- İnsan gibi düşünen sistemler
- İnsan gibi davranan sistemler.
- Mantıklı düşünen sistemler.
- Mantıklı davranan sistemler.

Yapay zekânın temelleri birçok farklı alanlardan beslenmektedir. Felsefe, Matematik, Algoritma, Ekonomi, Psikoloji, Bilgisayar

Mühendisliği, Sinir bilimleri, Kontrol teorisi ve Sibernetik ve Dilbilim başlıca sayılabilmektedir (Uğur, 2002).

Yapay Zekâ kavramının geçmişi modern bilgisayar bilimi kadar eskidir. Fikir babası, "Makineler düşünebilir mi? " sorusunu ortaya atarak Makine Zekâsını tartışmaya açan Alan Mathison Turing'dir. 1943 yılında İkinci dünya savaşı sırasında Kripto Analizi gereksinimleri ile üretilen Elektro-Mekanik cihazlar sayesinde Bilgisayar Bilimi ve Yapay Zekâ kavramları doğmuştur.

Alan Turing, Nazi'lerin Enigma makinesinin şifre algoritmasını çözmeye çalışan matematikçilerin en ünlenmiş olanlarından biriydi. İngiltere, Bletchley Park'ta şifre çözme amacı ile başlatılan çalışmalar, Turing 'in prensiplerini oluşturduğu bilgisayar ilk örneği olan Heath Robinson, Bombe ve Colossus bilgisayarları, Boole cebrine dayanan veri işleme mantığı ile Makine Zekâsı kavramının oluşmasına sebep olmuştu. (Sondak et al., 2003)

Modern bilgisayarın atası olan bu makineler ve programlama mantıkları aslında insan zekâsından ilham almışlardı. Ancak sonraları, modern bilgisayarlarımız daha çok uzman sistemler diyebileceğimiz programlar ile gündelik hayatımızın sorunlarını çözmeye yönelik kullanım alanlarında daha çok yaygınlaştılar. 1970'li yıllarda büyük bilgisayar üreticileri olan Apple, Xerox, IBM gibi şirketler kişisel bilgisayar modeli ile bilgisayarı popüler hale getirdiler ve yaygınlaştırdılar. Yapay Zekâ çalışmaları ise daha dar bir araştırma çevresi tarafından geliştirilmeye devam etti.

Alan Turing'in adıyla anılan Turing Testi yazılımların insan gibi düşünüp düşünmediğini ölçmektedir. Testin içeriği kısaca şöyledir: birbirini tanımayan birkaç insandan oluşan bir denek grubu birbirleri ile ve bir Yapay Zekâ diyalog sistemi ile geçerli bir süre sohbet etmektedirler. Birbirlerini yüz yüze görmeden yazışma yolu ile yapılan bu sohbet sonunda deneklere sorulan sorular ile hangi deneğin insan hangisinin Makine Zekâsı olduğunu saptamaları istenir. İlginçtir ki, şimdiye kadar yapılan testlerin bir kısmında Makine Zekâsı insan zannedilirken gerçek insanlar Makine zannedilmiştir (Uğur, 2002).

Loebner Ödülünü kazanan Yapay Zeka Diyalog sistemlerinin dünyadaki en bilinen örneklerinden biri A.L.I.C.E'dir. Carnegie üniversitesinden Dr.Richard Wallace tarafından yazılmıştır. Bu ve benzeri yazılımlarının eleştiri toplamalarının nedeni, testin ölçümlediği kıstasların konuşmaya dayalı olmasından dolayı programların ağırlıklı olarak diyalog sistemi (chatbot) olmalarıdır (Uğur, 2002).

Türkiye'de de Makine Zekası çalışmaları yapılmaktadır.Bu çalışmalar Doğal Dil işleme, Uzman sistemler ve Yapay Sinir Ağları alanlarında Üniversiteler bünyesinde ve bağımsız olarak sürdürülmektedir. Bunlardan biri, D.U.Y.G.U - Dil Uzam Yapay Gerçek Uslamlayıcıdır.

Yapay zekâ’nın araştırma ve çalışma alanları çok geniştir. Birçok farklı alanda Yapay zekâ kullanımıyla karşılaşmak mümkündür. Yapay zekâ ile günümüzde birçok alanda karşılaşılmaktadır. Bu alanlara aşağıdaki örnekleri verebiliriz:

- Oyunlar (Satranç, dama, strateji, ...)
- Yapay yaşam
- Teorem ispatlama (Prolog, Paralel Prolog, Cebir Mantık Programlama)
- Doğal dil anlama, işleme ve çeviri
- Bilgi tabanlı sistemler (Bilgi gösterimi, uzman sistemler, bilgi tabanlı benzeştirme, genel bilgi sistemleri, ...)
- Makine öğrenmesi (Bilgi düzeyinde öğrenme, sembol düzeyinde öğrenme, aygıt düzeyinde öğrenme)
- Makine buluşları (Veri madenciliği, bilimsel buluşların modellenmesi)
- Robotik (Görev planlama, robot görmesi)
- Şekil tanıma (Nesne tanıma, optik harf tanıma, ses tanıma, ...)

Kısaca Yapay Zekâ alanındaki kilometre taşları aşağıdaki gibi sıralanabilir:

Yunan Mitolojisi: akıllı makineler, mekanik araçlar ve yapay zekâ.

Abbasiler döneminde (750–1256), 9. yy'da hidrolik prensiplere dayalı otomatik makineler geliştirildi. İlk otomat sistemlerden sonra, bu çalışmalara Selçuklular döneminde "Ebul-İz" devam etmiştir.

Rönesans döneminde otomatik makineler konusundaki çalışmalara yenileri eklenmiştir (Leonardo da Vinci)

Pascal, ilk hesap makinesini geliştirmiştir.

Babbage, ilk programlanabilir bilgisayarı geliştirmiştir (19. yy).

Sembolik Mantık çalışmaları (Boole, Frege, Russel, Whitehead).

Turing'in hesaplanabilirlik teorisi (İlk zeki sistemler çalışmaları).

1940: Sibernetik (İnsan-makine)

1950: Alan Turing "Bilgisayar Mekanizması ve Zekâ" 'yı yayınladı.

1956: John McCarthy Dartmouth'taki bilgisayar konferansına "Yapay Zekâ" terimini ortaya attı.

1956: Carnegie Mellon Üniversitesi'nde ilk çalışan YZ programı tanıtıldı.

1958: John McCarthy Massachusetts Teknoloji Enstitüsü'nde (MIT) YZ programla dili olan LISP dilini icat etti.

1964: Danny Bobrow bilgisayarların kelime programları ve cebir problemlerini çözebilmek için yeteri derecede doğal dili anladığını gösterdi.

1965: Joseph Weizenbaum etkileşimli diyalog kurabilen ELIZA programını inşa etti.

1979: İlk bilgisayar denetimli otonom araç (Stanford Cart) yapıldı.

1983: Danny Hillis çok büyük ölçüde paralel çalışabilen Düşünen Makineleri üretti.

1985: Harold Cohen tarafından yapılan çizim programı Aaron, YZ konferansında tanıtıldı.

1990'lar: Makine öğrenmesi, bulanık mantık, uzman sistemler, veri madenciliği, takvimleme, doğal dil işleme, çeviri, sanal görsellik gibi YZ'nin bir çok alanında önemli ilerlemeler kaydedildi.

1997: Dünya satranç şampiyonu Garry Kasparov IBM bilgisayarı Deeper Blue'ya yenildi.

1990'ların sonu: Ağ tabanlı YZ uygulamalarında büyük artış görüldü.

2000: Etkileşimli robot ev hayvanları ticarî olarak satılabilir hale geldi. Honda, Sony gibi firmaların geliştirdiği insan benzeri robotlarda başarı oranı arttı (Vikipedia, 2006).

Yapay zekâ'nın temel yaklaşımları Yapay Sinir Ağları, Genetik Algoritmalar, Etmen Sistemler olarak sayılabilir. (Uğur, 2002)

3.2 Yapay Sinir ağları

Yapay sinir ağları, insan beyninin özelliklerinden olan öğrenme yolu ile yeni bilgiler türetebilme, yeni bilgiler oluşturabilme ve keşfedebilme gibi yetenekleri herhangi bir yardım almadan otomatik olarak gerçekleştirmek amacı ile geliştirilen bilgisayar sistemleridir. Bu yetenekleri geleneksel programlama yöntemleri ile gerçekleştirmek oldukça zor veya mümkün değildir. O nedenle, yapay sinir ağlarının, programlanması çok zor veya mümkün olmayan olaylar için geliştirilmiş uyarlamalı bilgi işleme ile ilgilenen bir bilgisayar bilim dalı olduğu söylenebilir. (Haykin, 1999)

3.2.1 Yapay Sinir Ağı Temel Görevi ve Genel Özellikleri

Yapay sinir ağları, insanlar tarafından gerçekleştirilmiş örnekleri kullanarak olayları öğrenebilen, çevreden gelem olaylara karşı nasıl tepkiler üretebileceğini belirleyebilen bilgisayar sistemleridir. İnsan beyninin işlevsel özelliklerine benzer şekilde,

- Öğrenme
- İlişkilendirme
- Sınıflandırma
- Genelleme
- Özellik belirleme
- Optimizasyon

Gibi alanlarda başarılı bir şekilde uygulanmaktadır. Örneklerden elde ettikleri bilgiler ile kendi deneyimlerini oluşturur; ve daha sonra, benzer konularda benzer kararları verirler (Yurtoğlu, 2005).

Yapay sinir ağlarının genel özellikleri aşağıdaki gibi listelenebilir:

- Yapay sinir ağları makine öğrenmesi gerçekleştirirler.
- Programları çalışma stili bilinen programlara benzemez.
- Bilginin saklanması bağlantı değerlerinde olur.
- Yapay sinir ağarlı örnekleri kullanarak öğrenirler.

- Güvenle kullanılabilmeleri için önce eğitilmeleri ve performanslarının test edilmesi gerekir.
- Görülmemiş örnekler hakkında bilgi üretebilirler.
- Algılamaya yönelik olaylarda kullanılabilirler.
- Şekil ilişkilendirme ve sınıflandırma yapabilirler.
- Eksik örüntüleri tamamlayabilirler.
- Doğrusal olmayan yapılar üzerinde çalışabilirler.
- Kendi kendini organize etme ve öğretebilme yetenekleri vardır.
- Eksik bilgi ile çalışabilirler.
- Hata toleransına sahiptirler.
- Öğrenme ve hafıza yetenekleri vardır.
- Belirsiz, tam olmayan bilgileri işleyebilmektedirler.
- Dereceli bozulma gösterirler.
- Dağıtık belleğe sahiptirler.
- Sadece nümerik bilgi ile çalışabilirler. (Saraç, 2004)

Bu özelliklerin bazılarını açıklayacak olursak;

Doğrusal Olmayan Yapı:

YSA'ların en önemli özelliklerinden birisi gerçek hayattaki olası doğrusal olmayan yapıları da dikkate alabilmesidir. YSA'ların doğrusal

olmayan modeller olarak görülebileceğine dair bulgular ortaya koymuştur.

Öğrenme:

YSA'ların diğer bir önemli avantajı en önemli özelliğinden kaynaklanmaktadır. Esin kaynağı insan beyninin çalışma sistemi olan bu yöntem, eğitme veya başlangıç tecrübesi sayesinde veriyi kullanarak öğrenme yeteneğine sahiptir. Bu özelliği sayesinde ise geleneksel teknikler için çok karmaşık kalan problemlere çözüm sağlayabilmektedirler. Ayrıca, insanların kolayca yapabildiği ama geleneksel metotların uygulanamadığı basit işlemler için de oldukça uygundurlar (Yurtoğlu, 2005).

Yerel İşlem ve Esneklik:

YSA'lar geleneksel işlemcilerden farklı şekilde işlem yapmaktadırlar. Geleneksel işlemcilerde, tek bir merkezi işlem elemanı her hareketi sırasıyla gerçekleştirir. YSA modelleri, her biri büyük bir problemin bir parçası ile ilgilenen çok sayıda basit işlem elemanlarından oluşma ve bağlantı ağırlıklarının ayarlanabilmesi gibi özelliklerinden dolayı önemli derecede esnek bir yapıya sahiptirler. Bu esnek yapı sayesinde ağın bir kısmının zarar görmesi modelde sadece performans düşüklüğü yaratır. Modelin işlevini tamamen yitirmesi söz konusu olmaz. Ayrıca, toplam işlem yükünü paylaşan işlem elemanlarının birbirleri arasındaki yoğun bağlantı yapısı sinirsel hesaplamanın temel güç kaynağıdır. Bu yerel işlem yapısı sayesinde, YSA yöntemi en

karmaşık problemlere bile uygulanabilmekte ve tatminkar çözümler sağlayabilmektedir (Yurtoğlu, 2005).

Gerçek Zamanlı İşlem:

YSA hesaplamaları paralel olarak yürütülebildiğinden gerçek zamanlı işlem yapabilir.

Genelleme:

Yine öğrenme yeteneği sayesinde bilinen örnekleri kullanarak daha önce karşılaşılmamış durumlarda genelleme yapabilmektedir. Yani, hatalı veya kayıp veriler için çözüm üretebilmektedir. YSA'lar, tanımlanmamış girdi veriler hakkında karar verirken genelleme yapabildikleri için iyi birer desen tanımlayıcısı ve sağlam sınıflandırıcıdırlar.

Hafıza:

Bunlara ek olarak, işlem elemanları arasındaki ağırlıklı bağlantılar sayesinde dağıtılmış hafızada bilgi saklayabildikleri söylenebilir.

Kendi İlişkisini Oluşturma:

Yapay sinir ağları, verilere göre kendi ilişkilerini oluştururlar, denklem içermezler.

Sınırsız Sayıda Değişken ve Parametre:

Diğer taraftan, YSA modelleri sınırsız sayıda değişken ve parametre ile çalışabilmektedir. Bu sayede mükemmel bir öngörü doğruluğu ile genel çözümler sağlanabilmektedir. Karmaşık veya sorunlu veriden bile anlam çıkarabilmek gibi dikkate değer yetenekleriyle YSA'lar, insanlar veya bilgisayarlar tarafından anlaşılması zor eğilimleri belirlemek veya desen çıkartmak için kullanılabilirler. Tam eğitilmiş bir Yapay Sinir Ağı modeli, analiz ettiği bilgi kümesi (veri tabanı) için uzman olarak düşünülebilir. Bu uzman, değişik durumlar ve '... olsa ne olur?' türünde simülasyon problemlerine projeksiyonlar sağlamak için kullanılabilir. Bununla birlikte, YSA'ların kullanımında göz önünde bulundurulması gereken bazı dezavantajlar da bulunmaktadır. Bunlar arasında en önemlisi geniş veri seti gereksinimidir. Sinir ağlarının eğitilebilmesine ve test edilebilmesine yetecek genişlikte veri setine ihtiyaç duyulmaktadır. Yine de, yeterli veri seti genişliği için kesin bir kıstas yoktur; bir noktada uygulamaya bağlıdır. Dezavantaj sayılabilecek diğer bir nokta ise basit olarak görülebilecek modelleme yapılarına rağmen uygulamanın zor ve karmaşık olabilmesidir. Bazı durumlarda, bir yakınsama sağlamak bile imkansız olabilmektedir fakat bu durum da uygulama alanına bağlıdır ve genellikle çok karmaşık problemlerde ortaya çıkmaktadır(Yurtoğlu, 2005).

3.2.2 Yapay Sinir Ağlarının Uygulama Alanları

Teorik Uygulama Alanları:

- Olasılık işlevi kestirimleri
- Sınıflandırma

- § İlişkilendirme ve örüntü eşleştirme
- § Zaman serileri analizleri
- § Sinyal filtreleme
- § Veri sıkıştırma
- § Örüntü tanıma
- § Doğrusal olmayan sinyal işleme
- § Doğrusal olmayan sistem modelleme
- § Optimizasyon
- § Zeki ve doğrusal olmayan kontrol

Pratik Uygulama Alanları:

Biyoloji: Beyni ve diğer sistemleri daha iyi anlama, Retina ve kornea'yı modelleme

İş Dünyası: Petrol ve jeolojik yapı değişimlerinin tahmini, Özel durumlar için toplum eğilimlerinin tanımı, Veri tabanı oluşturulması, Hava yolları ve ücret düzenlemesi, El yazısı karakterini tanıma.

Çevresel: Numuneleri analiz etme, Hava tahmini.

Finans: Kredi riski değerlendirilmesi, Sahte para ve evrak tanımı, El yazısı formların değerlendirilmesi, Yatırım eğilimleri ve portföy analizi

Üretim: Robot ve kontrol sistemlerini otomatikleştirme, Üretim işlem kontrolü, Kalite kontrolü, Montaj hattında parça seçimi.

Tıp: Sağırlar için ses analizi, Semptom hastalıkların teşhis ve tedavisi, Ameliyat görüntüleme, İlaçların yan etkilerinin analizi, X-ışınlarını okuma, Epileptik felcin nedenlerini anlama

Askeri: Radar sinyallerini anlama, Yeni ve gelişmiş silahlar yaratma, Keşif yapma, Kıt kaynakların kullanımını optimize etme, Hedef tanıma ve izleme. (Saraç, 2004)

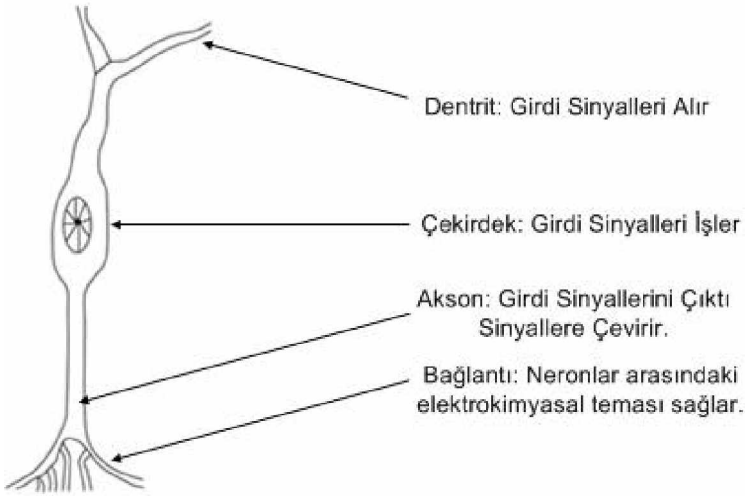
3.2.3 Doğal ve Yapay Sinir Hücreleri

Biyolojik sinir sistemi, merkezinde sürekli olarak bilgiyi alan, yorumlayan ve uygun bir karar üreten beynin (merkezi sinir ağı) bulunduğu 3 katmanlı bir sistem olarak açıklanır. Alıcı sinirler organizma içerisinden ya da dış ortamlardan algıladıkları uyarıları, beyine bilgi ileten elektriksel sinyallere dönüştürür. Tepki sinirleri ise, beyinin ürettiği elektriksel darbeleri organizma çıktısı olarak uygun tepkilere dönüştürür. Şekil 3.1 de bir sinir sisteminin blok gösterimi verilmiştir (Yurtoğlu, 2005).

Şekil 3.1: Biyolojik sinir sisteminin blok gösterimi

Merkezi sinir ağında bilgiler, alıcı ve tepki sinirleri arasında ileri ve geri besleme yönünde değerlendirilerek uygun tepkiler üretilir. Bu yönüyle biyolojik sinir sistemi, kapalı çevrim denetim sisteminin karakteristiklerini taşır. Merkezi sinir sisteminin temel işlem elemanı,

sinir hücresidir (nöron) ve insan beyninde yaklaşık 10 milyar sinir hücresi olduğu tahmin edilmektedir. Sinir hücresi; hücre gövdesi, dendritler ve axonlar olmak üzere 3 bileşenden meydana gelir. Dendriteler, diğer hücrelerden aldığı bilgileri hücre gövdesine bir ağaç yapısı şeklinde ince yollarla iletir. Axonlar ise elektriksel darbeler şeklindeki bilgiyi hücreden dışarı taşıyan daha uzun bir yoldur. Axonların bitimi, ince yollara ayrılabilir ve bu yollar, diğer hücreler için dendriteleri oluşturur. Şekil 3.2'de görüldüğü gibi axon-dendrite bağlantı elemanı synapse olarak söylenir (Yurtoğlu, 2005).



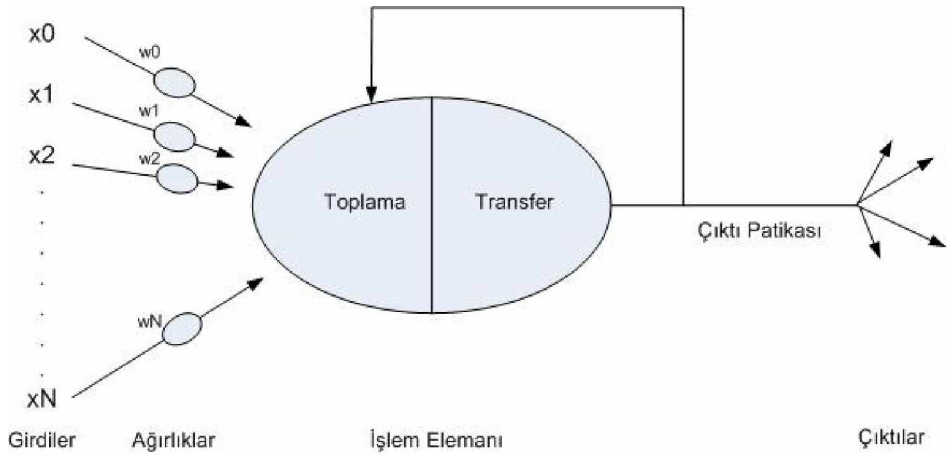
Şekil 3.2: Biyolojik sinir hücresi ve bileşenleri.

Synaps'e gelen ve dendriteler tarafından alınan bilgiler genellikle elektriksel darbelerdir ancak, synapsedeki kimyasal ileticilerden etkilenir. Belirli bir sürede bir hücreye gelen girişlerin değeri, belirli bir eşik değerine ulaştığında hücre bir tepki üretir. Hücrenin tepkisini artırıcı

yöndeki girişler uyarıcı, azaltıcı yöndeki girişler ise önleyici girişler olarak söylenir ve bu etkiyi synapse belirler.

İnsan beyninin 10 milyar sinir hücresinden ve 60 trilyon synapse bağlantısından oluştuğu düşünülürse son derece karmaşık ve etkin bir yapı olduğu anlaşılır. Diğer taraftan bir sinir hücresinin tepki hızı, günümüz bilgisayarlarına göre oldukça yavaş olmakla birlikte duyusal bilgileri son derecede hızlı değerlendirebilmektedir. Bu nedenle insan beyni; öğrenme, birleştirme, uyarlama ve genelleştirme yeteneği nedeniyle son derece karmaşık, doğrusal olmayan ve paralel dağılmış bir bilgi işleme sistemi olarak tanımlanabilir.

Şekil 3.3'de yapay sinir ağı hücresi görülmektedir. Temel olarak 5 bileşeni vardır. Girdi değerleri, ağırlıklar, toplama işlemi, aktivasyon işlemi ve çıktı. Bu bileşenler dışında bias ve yerel alan değerleri de kullanılmaktadır (Yurtoğlu, 2005).



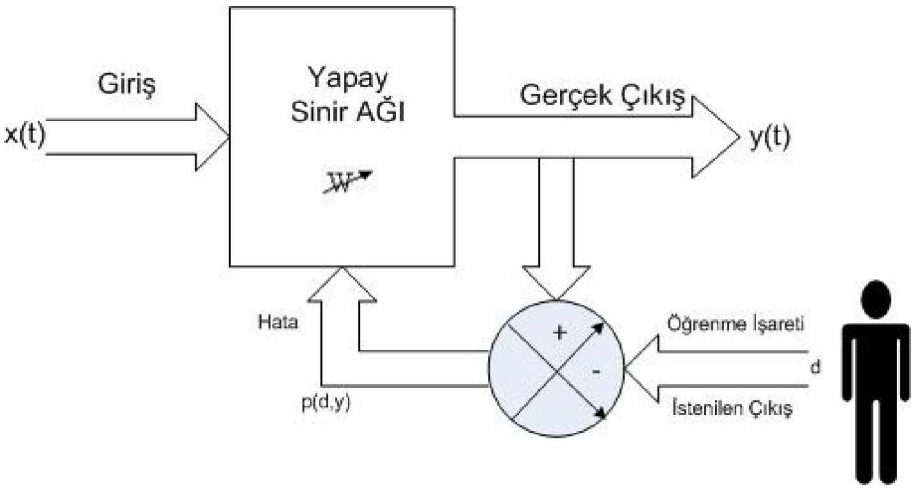
Şekil 3.3: Bir yapay sinir hücresi

Yapay sinir ağının işleyişi kabaca şöyledir: Girdi değerleri ağa verilir ve bu değerler ağırlıklarla çarpılır. Sonra toplama işlemine ardından aktivasyon işlemine gönderilerek çıktı değeri elde edilir. Ağın yapısına ve isteklerimize göre bias değeri belirlenebilir ve karşılaştırma yapılabilir. Ayrıca ağın yapısına göre çıktı sonucu değerlendirmeye alınır ve ağırlıklar sonuca göre değiştirilebilir.

3.2.4 Öğrenme Modelleri

3.2.4.1 Öğreticili Öğrenme

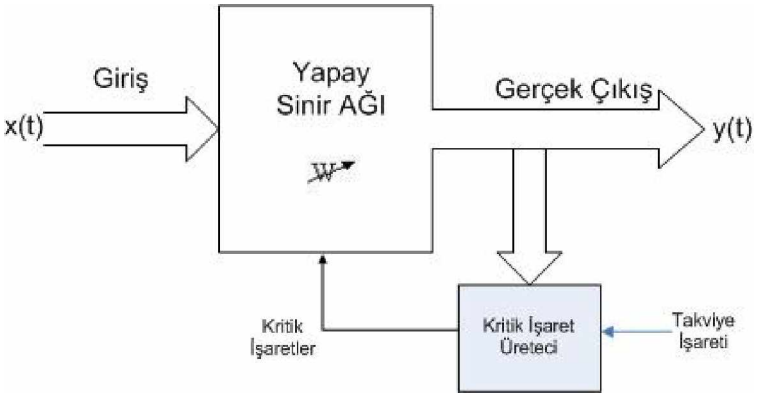
Öğreticili öğrenme kipiyle, yapay sinir ağının eğitimi için eğitici veriler (eğitim seti) kullanılmaktadır. Eğitim seti giriş bilgileri ve istenen(hedef) bilgiler olmak üzere iki ayrı vektör gibi düşünülebilir. Vektörlerin her bir karşılıklı elemanları bir eğitim çiftini oluşturmaktadır. Eğitim seti ağın eğitimine başlamadan önce belirlenmektedir. Ağın eğitimi için, öncelikle bağlantı ağırlıklarına rasgele değerler atanmaktadır. Daha sonra eğitim çiftlerine bağlı olarak bir algoritma dahilinde ağırlıklar yenilenmektedir. İstenilen bilgiler ve ağın çıkışı arasındaki fark(hata) azalıncaya kadar eğitim sürdürülmektedir. Ağ çıkışındaki, hatanın azalması ağırlıkların kararlılık kazanması demektir. Ağırlıklar istenilen kararlılığa ulaştığında eğitim bitirilmektedir. Geri yansıtmalı yapay sinir ağı modeli de öğreticili öğrenme kipini kullanmaktadır. Şekil 3.4'te öğreticili öğrenme yapısı gösterilmiştir. (Haykin, 1999)



Şekil 3.4: Öğreticili öğrenme durumu

3.2.4.2 Destekleyici öğrenme

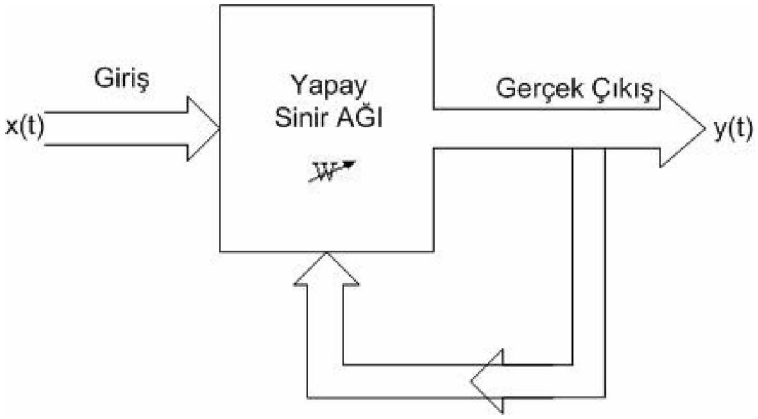
Bu öğrenme kuralı öğreticili öğrenmeye yakın bir metottur. Denetimsiz öğrenme algoritması istenilen çıkışın bilinmesine gerek duymaz. Hedef çıktıyı vermek için "öğretmen" yerine, burada YSA'ya bir çıkış verilmemekte fakat elde edilen çıkışın verilen girişe karşılık iyiliğini değerlendiren bir kıstas kullanılmaktadır. Şekil 3.5'te destekleyici öğrenme yapısı gösterilmektedir. Optimizasyon problemlerini çözmek için Hilton ve Sejnowski'nin geliştirdiği Boltzman kuralı veya genetik algoritma tasdikli öğrenmeye örnek olarak verilebilirler. (Haykin, 1999)



Şekil 3.5: Destekleyici öğrenme yapısı

3.2.4.3 Öğreticisiz Öğrenme

Öğreticisiz öğrenme kipine "Kendi kendine öğrenilebilen kip" da denilmektedir. Bu öğrenme kipinde eğitim seti kullanılmamaktadır. Ağ, birbirine benzer giriş bilgilerini gruplamakta veya giriş bilgisinin hangi gruba ait olduğunu göstermektedir. Ağ eğitimi için sadece giriş bilgileri yeterli olmakta, referans alınacak(eğitici) bilgiye ihtiyaç duyulmamaktadır. Ağın performansını kendiliğinden izlenmesi söz konusudur. Ağ, giriş sinyallerinin yönüne veya düzenine bakmakta ve ağın işlevine göre ayarlama yapmaktadır. Ağ kendini nasıl organize edeceği hakkında bir miktar bilgiye sahip olmalıdır.(İnal 1996). Şekil 4.6'te öğreticisiz öğrenme yapısı gösterilmiştir.



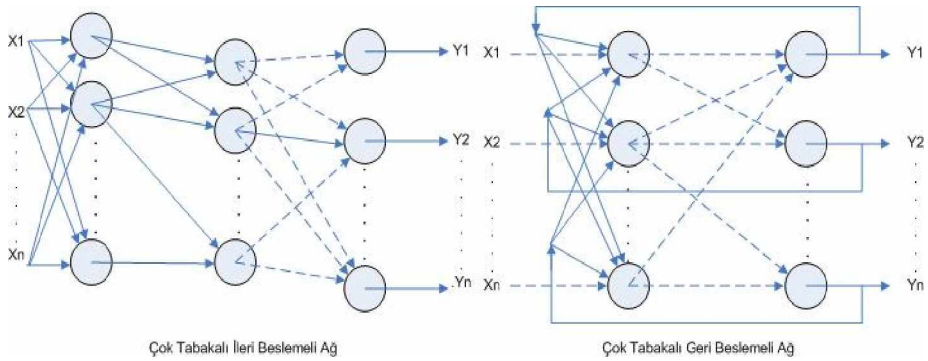
Şekil 3.6: Öğreticisiz öğrenme yapısı

3.2.5 Yapay Sinir Ağları Çeşitleri

YSA'ların çok sayıda farklı çeşitleri vardır. Bu farklılıkların kaynağı mimarisi, öğrenme yöntemi, bağlantı yapısı vb. olabilmektedir. Genel olarak, YSA'lar üç ana kıstasa göre sınıflandırılmaktadırlar. Bu kıstaslardan biri öğrenme yöntemidir. Önceki bölümde belirtildiği gibi, temel olarak iki çeşit öğrenme algoritması vardır: yönlendirmeli öğrenme ve yönlendirmesiz öğrenme. Her yöntemin kullandığı öğrenme kuralı değişebilmekteyse de, YSA'lar bu iki algoritmaya göre sınıflandırılırlar. (Haykin, 1999)

İkinci bir sınıflandırma, ağın kullandığı veriye göre yapılmaktadır. Temel olarak, niteleyici ve niceleyici olmak üzere iki tür veri vardır. Kalitatif verilerle çalışan ağlar, ister yönlendirmeli ister yönlendirmesiz öğrenme kullansın, sınıflandırma ağları olarak bilinirler. Niteleyici veriler kullanan yönlendirmeli eğitme ise regresyon olarak adlandırılmaktadır.

Son sınıflandırma ölçütü ise ağın yapısıdır. Bazı ağlar ileri besleme şeklinde yapılandırılırken, bazı ağlar ise geri besleme yapısı içermektedir. İleri besleme sinir ağlarında, işlem elemanları arasındaki bağlantılar bir döngü oluşturmazlar ve bu ağlar girdi veriye genellikle hızlı bir şekilde karşılık üretirler. Geri beslemeli ağlarda ise bağlantılar döngü içerirler ve hatta her seferinde yeni veri kullanabilmektedirler. Bu ağlar, döngü sebebiyle girdinin karşılığını yavaş bir şekilde oluştururlar. Bu yüzden, bu tür ağların eğitme süreci daha uzun olmaktadır. Ayrıca, hem ileri besleme hem de geri yayılma olarak tanımlanabilecek ağ yapıları da mevcuttur. Şekil 3.7'de, bir kıyaslamaya imkan tanıyabilmek için, çok tabakalı ileri besleme ağ yapısı ile birlikte çok tabakalı geri besleme ağ yapısı örneklenmektedir (Yurtoğlu, 2005).



Şekil 3.7: İleri beslemeli ve geri beslemeli ağ yapıları

Bu çok geniş YSA çeşitleri yelpazesinde en çok bilinen ve kullanılan ağlar arasında hata algoritması genellikle geri yayılma ile eğitilen çok tabakalı Perceptron (Geri-Yayılma Ağ – Backpropagation

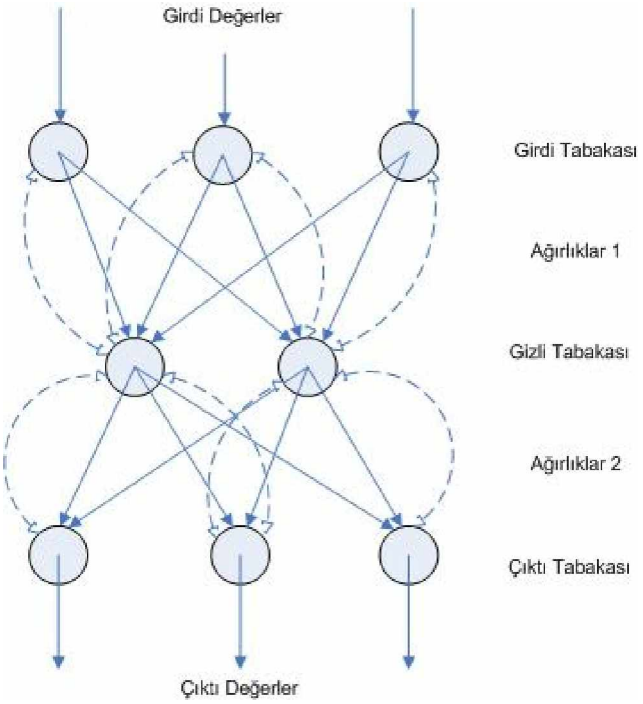
Network), Radyal tabanlı fonksiyon (radial basis function), Hopfield ve Kohonen sayılabilir. Çok fazla çeşit ve yoğun bir yayın bulunmaktadır. Burada en yaygın kullanılanlarından biri olan Geri Yayılma Yapay Sinir Ağı hakkında bilgi verilmemektedir.

Geri yayılma ağlar, çok tabakalı Perceptron ile aynı yapıya sahiptirler ve öğrenme yöntemi olarak geri yayılma algoritması kullanırlar. Dolayısıyla, bu ağlar ileri besleme ağlar sınıfına girmektedirler. Ayrıca, çalışmada kullanılan ağ niceleyici verilerle çalışmaktadır ve yönlendirmeli öğrenme yöntemi kullanmaktadır. Bu YSA türünün seçilmesinin temel sebebi öngörü ve sınıflandırma işlemleri için oldukça uygun olmasıdır. Diğer bir önemli neden ise doğrusal olmayan yapılar için de oldukça kullanışlı olmasıdır.

İleri beslemeli Geri Yayılma mimarisi 1970'li yıllarda geliştirilmiştir. Bu mimarinin geliştirilmesinde birbirlerinden bağımsız olarak birkaç araştırmacının katkıları olmuştur. Asıl katkı ise Rumelhart, Hinton ve Williams (1986) tarafından yapılmıştır. Ortaya çıkışından sonra, hem etkili hem de çok kullanışlı olmasından dolayı büyük bir popülerlik kazanmıştır ve hala en çok kullanılan ağ türü olarak bilinmektedir. Çok sayıda farklı uygulama alanında kullanılmaktadır ve en büyük özelliği doğrusal olmayan yapı içeren problemlerde de etkili olabilmesidir.

Tipik bir geri yayılma ağ mimarisinde bir girdi tabakası, bir çıktı tabakası ve bu iki tabaka arasında en az bir adet gizli tabaka bulunur. Gizli tabaka sayısı için herhangi bir kısıt yoktur fakat genellikle bir ya da

iki gizli tabaka kullanılmaktadır. Bu ağ çeşidinin genel yapısı Şekil 3.8'de örneklenmektedir (Yurtoğlu, 2005).



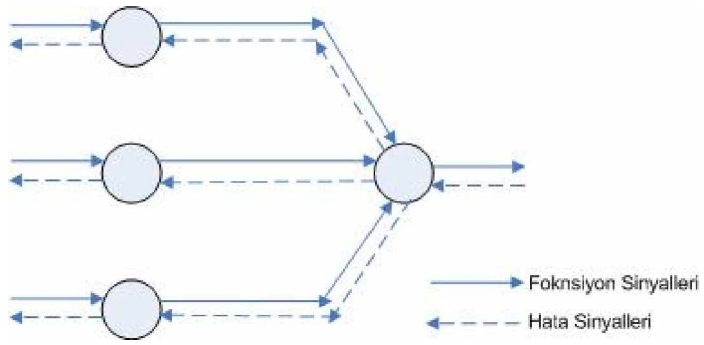
Şekil 3.8: İleri beslemeli geri yayılma ağların genel yapısı

Şekilde, bir girdi tabakası, bir gizli tabaka ve bir çıktı tabakası içeren bir geri yayılma ağ yapısı örneklenmektedir. Tabakalar halinde düzenlenmiş daireler işlem elemanlarını yani nöronları temsil etmektedir. Girdi tabakasında üç nöron bulunmaktadır, yani ağa girdi olarak üç değişken tanıtılmaktadır. Bunun dışında gizli tabakada iki nöron, çıktı tabakasında ise üç nöron bulunmaktadır. Dolayısıyla, ağdan üç değişken olarak çıktı alınmaktadır. Girdi tabakasından, gizli tabakaya iletilen değerler "Ağırlıklar 1" ağırlık seti ile gizli tabakadan çıktı tabakasına

iletilen değerler ise “Ağırlıklar 2” ağırlık seti ile değerlendirilmektedir. Ağ yapısında, kalın oklar anımsama sırasındaki bilgi akışını simgelemektedir. Anımsama, eğitilmiş bir ağa yeni girdi verilerin sunulması ve çıktısının alınması işlemidir. Bu yüzden, anımsama işlemi sırasında geri yayılma işlemi kullanılmaz.

Geri yayılma sadece eğitme sürecinde kullanılır, dolayısıyla eğitme sürecindeki bilgi akışı şekildeki tüm oklar ile gösterilmektedir.

Sonuç olarak, bu ağ yapısının eğitme sürecinde iki çeşit sinyal tanımlanmaktadır. Birbirine zıt yönde çalışan bu iki sinyal çeşidi fonksiyon sinyalleri ve hata sinyalleri olarak adlandırılmaktadırlar (Parker, 1987). Şekil 3.9’da küçük bir ağ bölümü kullanılarak bu sinyallerin yönleri gösterilmektedir. Şekilde, sol taraf girdi veri tarafını, sağ taraf çıktı tarafını ve yuvarlaklar ise nöronları göstermektedir.



Şekil 3.9: Eğitme sürecindeki sinyal çeşitleri

Fonksiyon sinyali, ağa girdi tabakasından giren, ağ içinde ileri doğru yayılan ve çıktı tabakasından çıktı sinyali olarak çıkan bir girdi

sinyalidir. Bu tür bir sinyalin fonksiyon sinyali olarak adlandırılmasının iki temel sebebi vardır. İlk olarak, ağın çıktısı için gerekli fonksiyonları uyguladığı kabul edilmektedir. İkinci sebep ise fonksiyon sinyalinin geçtiği her nöronda, sinyal girdilerin ve o nörona uygulanan ağırlıkların bir fonksiyonu olarak hesaplanır. Fonksiyon sinyalleri, girdi sinyalleri olarak da adlandırılmaktadırlar. Hata sinyali ise, fonksiyon sinyalinin tersine, çıktı tabakasından başlar ve tabaka tabaka geriye doğru yayılır. Hata sinyali olarak adlandırılmasının sebebi, tüm nöronlarda bir hata tabanlı fonksiyon ile hesaplanmasıdır.

Ağın genel yapısına dönersek, tabaka sayısı ve tabakaların içerdiği işlem elemanı sayısı ağın performansı açısından önemli ve zor kararlardır. Zor karar olmalarının sebebi ise herhangi bir uygulama için net bir seçim ölçütünün olmamasıdır. Bunun yerine, uygulamalar sonucunda ortaya çıkmış ve araştırmacılar tarafından benimsenmiş bazı kurallar vardır.

Bu kurallar şu şekilde özetlenebilir:

Kural–1: Girdi ve çıktı veriler arasındaki ilişkinin karmaşıklık derecesi arttıkça, tabakaların içerdiği işlem elemanı sayısı da artmalıdır.

Kural–2: Modellenen konu değişik safhalara ayrılabiliyorsa, tabaka sayısının artırılması gerekebilir.

Kural–3: Eldeki eğitme verisinin genişliği, gizli tabakalardaki toplam nöron sayısı için bir üst limit ölçütleri oluşturur.

Geri yayılma ağlarda çok çeşitli öğrenme kuralı, hata fonksiyonları ve transfer fonksiyonları kullanılabilmektedir. Öğrenme kuralı olarak genellikle Delta Kuralı'nın bir varyantı kullanılmaktadır. Delta kuralı, ağın çıktısı ile istenilen çıktı arasındaki farkın hesaplanması ile başlar. Bu hata kullanılarak bağlantı ağırlıkları belirli bir doğruluk derecesi faktörüne göre güncellenir. Bu öğrenme mekanizmasının karışık olan tarafı, hatalı çıktı üretilmesinde hangi işlem elemanının daha etkili olduğunun belirlenmesi ve hatanın düzeltilmesi için bu işlem elemanının nasıl değiştirileceğidir. Bu noktada aktif olmayan bir uç hataya sebep olamaz ve dolayısıyla ağırlıklarını değiştirmeye gerek yoktur. Bu sorunun çözümü için, eğitme setine ait girdi veriler ağın girdi tabakasına sunulur ve istenilen çıktılarla karşılaştırma çıktı tabakasında gerçekleştirilir. Öğrenme işlemi süresince, ağ içinde ileri doğru bir bilgi akışı vardır ve tabaka tabaka her işlem elemanının çıktısı hesaplanır. Çıktı tabakasına ulaşıldığında, bu tabakanın çıktısı ile istenilen çıktı arasındaki fark hesaplanır ve bu hata önceki tabakalara iletilir (geri yayılma). Bu süreçteki önemli nokta ise, hata önceki tabakalara iletilirken, transfer fonksiyonunun türevi ile bir transformasyon işlemi uygulanmasıdır. Hatanın iletilmesi, tabaka tabaka geriye doğru olur ve bu süreçte Delta Kuralı ile bağlantı ağırlıkları ayarlanır. İşlem, girdi tabakasına ulaşılana kadar devam eder ve bu noktada yeni bir döngüye başlar.

Geri yayılma algoritmasında, Delta Kuralı ile bağlantıların ayarlanmasının matematiksel gösterimi şu şekilde özetlenebilir. Delta Kuralı, temel olarak, ilgili bağlantı ağırlığının ayarlanması için gerekli

olan düzeltme miktarını formüller. Buna göre, nöron(i) ve nöron(j) arasındaki bağlantı için düzeltme miktarı şu şekilde hesaplanır (Yurtoğlu, 2005):

Çizelge 3.1: Sinir hücreleri ara bağlantı düzeltme miktarı

Ağırlık Düzeltme Miktarı	=	Öğrenme Oranı Parametresi	*	Yerel Değişim (Gradient)	*	Nöron(j) İçin Girdi Sinyali

Veya matematiksel formül olarak:

$$\Delta wji(n) = \eta * \delta j(n) * Yi(n)$$

Burada dikkat edilmesi gereken nokta, yerel değişimin hesaplanma şeklinin nöron(j)'nin çıktı veya gizli nöron olmasına göre değişiklik gösterdiğidir. Buna göre:

• Nöron(j) bir çıktı nöronu ise, yerel değişim, nöron(j)'ye ait hata sinyali ve fonksiyon sinyalinin türevinden hesaplanmaktadır.

• Nöron(j) bir gizli nöron ise, yerel değişim, fonksiyon sinyalinin türevi ve bir sonraki tabakadaki nöronlara ait değişimlerin ağırlıklı toplamı kullanılarak hesaplanır.

Son olarak, geri yayılma ağların bazı kısıtlamalara da sahip olduğu belirtilmelidir. Geri yayılma mekanizması oldukça geniş bir girdi-çıktı veri seti ile geniş çaplı bir yönlendirmeli eğitmeye ihtiyaç duyar. Ek olarak, içsel haritalama yapısı tam olarak anlaşılamadığından sistemin istenilen doğruluk derecesine yakınsayamaması ihtimali de mevcuttur (Sondak et al., 1989).

4 AĞ YÖNETİMİNDE YAPAY ZEKÂ KULLANIMI

Teknolojinin ilerlemesi ile birlikte günümüz ağ sistemleri yüksek hız sunan ve çok değişik tipte ağ elemanları ile heterojen bir yapı içermektedirler. Bu yapıları, ağ sistemlerinin yönetiminde klasik insan operatörlerinin kullanımından farklı çözümler gerektiren çok daha karmaşık ve yoğun veri işleme ortamları haline getirmiştir (Denise et al., 1990). Ağ yönetiminin otomatikleştirilmesinde yapay zekâ tekniklerinden hata yönetiminde, performans yönetiminde ve trafik yönetiminde etkin olarak kullanılabilirler.

Bu tez çalışması kapsamında ağ yönetim ihtiyaçlarından hata yönetimi alanı üzerine yapay zekâ tekniklerinin kullanımı araştırılmıştır. Ayrıca diğer ağ yönetim ihtiyaçlarının da yapay zekâ teknikleri ile nasıl karşılanabileceğine ise bir başlık altından değinilmiştir.

4.1 Hata Yönetiminde Yapay Zekâ Kullanımı

Ağ yönetimindeki en önemli fonksiyonlardan birisi hata yönetimidir. Hata yönetimi benzeri ağ yönetim işleri, genellikte insan müdahalesi ile yapılmaktadır. Fakat günümüz bilgi sistemleri ağlarının hızla büyüyen ve heterojen yapısı daha yoğun uzmanlık ve iş gücü gerektirmeye başlamıştır. Bu nedenlerden dolayı bu tür ağ yönetim işlerini otomatikleştirilmek oldukça çok önem kazanmıştır (Wang, 1989). Bu bağlamda, ağ sistemlerinin hata yönetimi aktivitelerinde yapay zekâ teknikleri; problem çözümü ve çıkarsama kabiliyetleri ile öne çıkmaktadır.

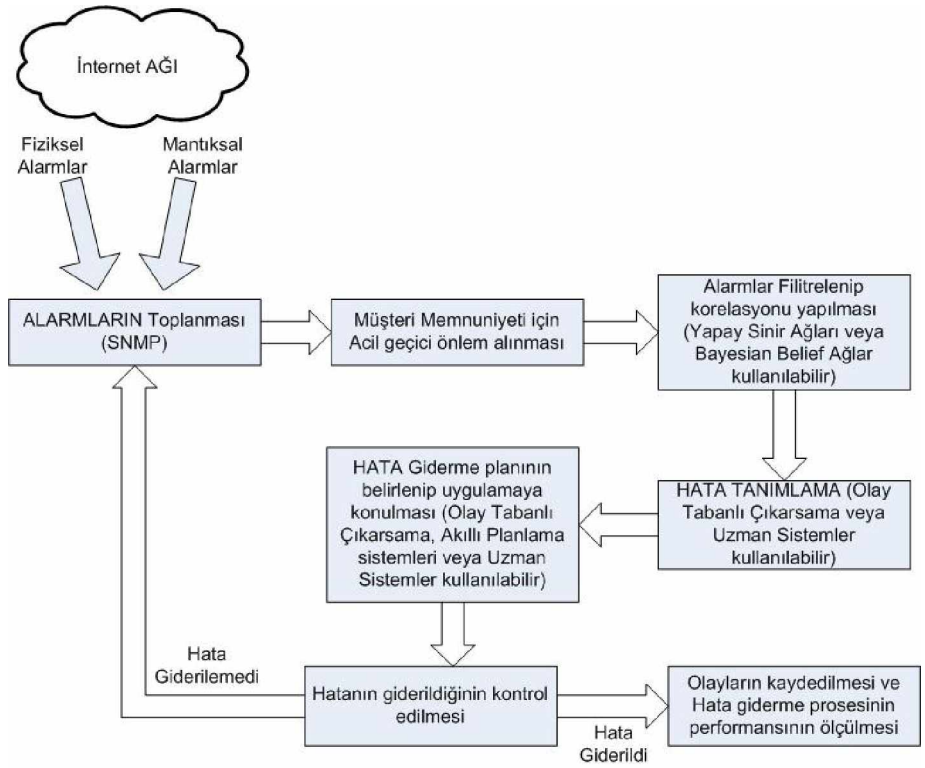
Uzman sistemler birçok hata yönetimi sistemine başarı ile uygulanmışlardır. Fakat bu sistemler günümüz bilişim ağlarının gelişen ihtiyaçlarını karşılayabilecek kadar esnek değillerdir. Bu nedenle Yapay Sinir Ağları ve Olay Tabanlı çıkarsama tekniklerini bir arada içeren melez yaklaşımlar hata yönetim sistemleri için daha verimli sonuçlar vermektedirler.

4.1.1 Hata Yönetim Sistemleri

Hata yönetiminde hedef aktif olarak ağ üzerindeki anormallikleri izleyerek nedenlerini tanımlamak ve eğer mümkün ise gerekli düzeltici adımları atmaktır. İlk olarak hata yönetimi işlemini inceleyip daha sonra bunların yapay zekâ yöntemleri ile nasıl otomatik gerçekleştirilebileceğini listeleyeceğiz. Özellikle Yapay Sinir Ağları ve Olay Tabanlı Çıkarsama yöntemleri üzerinde duracağız.

Temel olarak ağ hataları donanım ve yazılım hataları olarak ikiye ayrılabilir (Denise et al., 1990). Bu hatalar ağ elemanlarının yanlış sonuçlar üretmesine neden olup hataların birbirini tetiklemesi ile birlikte tüm ağın birden tıkanmasına neden olabilirler. Donanım hatalarına örnek olarak bir ağ elemanı parçasındaki tasarımından kaynaklı bir sebeple yetersiz kalması ve zayıflık göstermesi; kaza gibi dış etkenler sebebi ile fiziksel parçaların zarar görmesi; kötü kullanımları; yanlış kurulumları gösterilebilir. Yazılım hatalarına örnek olarak; yazılımdaki yanlış veya eksik tasarımlar sonucu ağ elemanı işlemlerinin hatalara düşmesi; değişken ağ elamanı ve yanlış bilgiler (yanlış yönlendirme tabloları)

sonucu kaynaklanan hatalar; ağ elemanı üzerindeki yazılım içerisindeki hatalar gösterilebilir.



Şekil 4.1: Hata yönetimi akışı

Şekil 4.1 de gösterilen hata yönetimi akışı su şekilde tanımlanabilir;

1- Alarmları toplama

2- Acil aksiyon ile müşteri memnuniyetini sağlama

3- Gelen alarmları filtreleme ve korelasyon

4- Analiz ve test ile hataları teşhis etme

5- Düzeltme için bir plan tanımlama

6- Hatanın giderildiğini doğrulama

7- Bilgilerin kaydedilmesi ve hata yönetimi işlevinin etkinliğini tanımlama.

Hata yönetiminde ilk adım hata alarmlarının ve performans alarmlarının toplanmasıdır. Alarmlar ağ elemanları tarafından üretilebileceği gibi istatistiksel olarak ağ elemanının performans değerlerinin izlenmesi ve belli eşik değerlerini aşması neticesinde ağ yönetim sistemi tarafından da oluşturulabilir. Alarmlar fiziksel ve mantıksal olarak ikiye ayrılabilir. Fiziksel alarmlar (bağlantı düşmesi gibi) önemli hatalardır ve genellikle ağ elamanı yöneticisi tarafından raporlanır. Mantıksal alarmlar ise istatistiksel hatalardır (tıkanıklık nedeni ile performans düşmesi gibi) (Denise et al., 1990).

Hatalar ağ yönetimi tarafından toplandığı ve raporlandığı anda ilk olarak hatanın giderilebilmesi için gerekli servis işleme koyulmalıdır. Bu işlem hatanın incelenip asıl nedeni bulunana kadar müşterinin aldığı hizmet kalitesindeki düşüş meydana gelmeden yapılan geçici bir çözümdür. Örnek olarak bir anahtarda meydana gelen bozukluk sonucu trafiğin başka bir yola yönlendirilmesi verilebilir.

Ağı kullanmakta olan müşterilerin normal bir şekilde servislerden faydalanması sağlandıktan sonraki adım toplanan alarmlarının filtrelenerek korelasyonudur. Alarm filtreleme aşamasında toplanan alarmların sayıları analiz edilir ve tekrarlanan alarmlar elenir. Alarm korelasyonu aşamasında ise alarmlar işlenerek gerekli mantıksal çıkarsamalar yapılır ve gerekli ise bu çıkarsamalar sonucu yeni alarmlar

yaratılır. Alarmların filtrelenmesi ve korelasyonu sonucunda hatalar tanımlanır ve ağ elemanı yöneticilerinden durum hakkında gerekli ekstra bilgiler sağlanır ve gerekli ise test aksiyonları almaları istenir(Denise et al., 1990).

Hatanın tanımlanmasından sonra hatanın sebebinin giderilmesi için gerekli düzeltici işlemler devreye alınabilir. Ağ yönetim sisteminin korelasyon aşamasındaki görevi önlem alma aşamasında atılacak olan adımları belirlemek ve ağ üzerindeki yapılması gereken diğer işlemlerle birlikte bu aşamaların işletilmesini sağlamaktır. Korelasyon işleminin ağ yönetiminde ki oranı arttıkça insan müdahalesine gereksinimde o kadar azalabilecektir. Bazı zamanlarda da teknisyenin giderek ağ elemanı üzerindeki bir parçayı değiştirmesi veya bir yazılım hatasının giderilmesi için yazılımcı tarafından kod geliştirilmesi gerekli olabilecektir. Korelasyon işlemi ağ eleman yöneticilerine test ve durum bilgilerinin sorulması ile doğrulanmalıdır. Eğer hata alınan aksiyon sonucunda düzelmemiş ise daha fazla verinin analiz edilmesi ile birlikte hata giderilme aşaması tekrarlanır.

Hata yönetimindeki bir başka adım ise hata yönetim sürecinin etkinliğinin takip edilmesi için gerekli verinin toplanmasıdır. Bu aşamada toplanan veriler ile hatanın neden olduğu zararın ve hata giderimi için harcanan güç gibi verilere ulaşılır. Ağ üzerinde ne kadar hata oluştuğu ve servis kalitesini ne kadar hatanın etkilediği ağ büyüklüğü ve müşteri sayısı ile orantılıdır (Byrne,1994). Ağ üzerinde oluşan hatalar sonucu servis kesintilerinin süreleri, ne kadar hata oluştuğu gibi istatistiksel

veriler ağ yönetim sisteminin performansının ölçülmesinde kullanılırlar. Ayrıca hataların daha detaylı analizleri sonucu elde edilen veriler ağ ve ağ elemanlarının performans ve güvenilirlikleri hakkında veriler verir.

4.1.2 Yapay Zekâ Uygulanması

Basit bir hata yönetimi süreci sırasında alarmların filtrelenmesi, korelasyonu, hatanın tanımlanması ve hatanın giderilmesini gibi adımlar bulunmaktadır. Bu adımlar; analiz, korelasyon, desen tanımlanması, gruplama ve kategorilere ayırma, problem çözme, planlama ve verilerin ağ elemanlarının ve ağ topolojisi hakkında veriler içeren bir bilgi tabanı üzerinden yorumlanması gibi aşamalar içerir. Yapay zekâ teknikleri bu türden gereksinimler için oldukça uygundur.

Şu an birçok sistemde hata tanımlanması için yapay zekâ tekniklerinden uzman sistemler kullanılmışlardır (Corn et al., 1988). Bu sistemler oldukça başarılı bir şekilde kullanılmalarına rağmen bazı kısıtları bulunmaktadır. Genel olarak bunlardan bahsedecek olursak;

- Uzman sistemler yeni ve değişen verileri yönetemezler. Kurallar öngörülmeyen koşullar için (örneğin değişen ağ haritası sonucu yeni alarmların ortaya çıkması) güçlü değillerdir.

- Tecrübeler sonucu öğrenme yetenekleri yoktur. Geçmiş veriler ve tecrübeleri kullanarak geçmiş başarı ve başarısızlıkları değerlendirip örnekleme yolu ile neden

çıkarsaması yapamazlar. Kurallar sisteme geliştirme aşamasında dahil olurlar ve bu da ağın gelişme hızına yetişecek bir hızda sisteme kuralların güncellenmesini engeller.

- Ağ sistemlerinin hızında gelişemedikleri için ağ sistemi büyüdükçe ölçeklenemezler. Bu ölçekleme için teknisyenlerin bu sistemi anlayacak kadar bilgi birikimine sahip olmalılar, mevcut kural tabanı analiz edip yeni hatalar sonucu gerekli olan kuralları sisteme ekleyebilmelidirler ve yeni kuralın da mevcut sistemi nasıl etkileyeceğini tespit edebiliyor olmaları gerekmektedir. Dolayısı ile teknisyenler ile kural eklemeleri gerektirecek uzman sistem tabanlı bir ağ yönetim sisteminin yaşatılması oldukça zordur.

- Ağ sistemi değiştikçe oldukça kapsamlı bakım çalışmaları gerekir. Yeni kurallar eklenmelidir, mevcut kurallar ya bunlara göre adapte edilmeli veya çıkarılmalıdırlar.

- Belirsizlik ve olasılık koşullarını yönetmekte başarılı değillerdir. Bulanık mantık teknikleri ile bulanık kurallar sisteme dahil edilebilir. Fakat bu sistemlerin de yukarı listelediğimiz dezavantajları bulunmaktadır.

- Uzman sistemler büyük miktardaki korelasyonu yapılmamış veriler ile çalışmada güçlük çekerler. Uzman sistemler için

alan tam olarak anlaşılmış olmalıdır. Fakat hata yönetimi gibi alanlarda bunun tamamen sağlanması zordur.

Bütün bu dezavantajlar hata yönetiminde farklı yapay zeka tekniklerinin ayrı veya uzman sistemlere destek olarak birlikte kullanılmalarının gereksinimi doğurmaktadır (Denise et al., 1990). Yapay Sinir Ağları gibi olasılık metotları korelasyon için oldukça uygundurlar. Simgesel yöntemler olarak sayabileceğimiz Olay Tabanlı Çıkarsama ve Uzman Sistemler ise hata tanımlaması aşaması için daha verimli kullanılabilirler. Dolayısı ile genellikle bu sistemleri kooperatif olarak birlikte kullanmak daha verimli sonuçlar verecektir.

Yapay zekâ tekniklerinin çok faydalı olarak ağ yönetiminde kullanılabileceği bir başka alan ise hata giderilmesidir. CBR sistemleri, ES sistemleri veya akıllı planlama sistemleri belirlenmiş ve tanımlanmış hatanın giderilmesi için planlar geliştirilebilirler (Li et al., 2001).

4.1.2.1 Hata Tanımlama

Günümüz büyük ve dinamik bilgi ağı sistemlerinde ağ yönetim sisteminin ağ haritasında ve ağ elemanlarında meydana gelen değişimlere uyum gösterebilmesi çok önemlidir. Bunun için gerekli olan filtreleme, korelasyon ve genel hata kategorilerinin oluşturulması eğitilmiş bir yapay sinir ağı sayesinde veya geliştirilmiş bir BBN ile gerçekleştirilebilir. Fakat hatanın tam olarak yerinin saptanabilmesi için gerekli testlerin yapılması ve sonuçlarının analiz edilmesi gibi bir yordam geliştirilmesi gerekmektedir. Bu yordamlar insanın tecrübelerine dayanan bir kararlar

serisi içermektedirler. Bu nedenle sadece bir yapay sinir ağı veya BBN ile bu işlemin gerçekleştirilmesi olanaksızdır. Dolayısı ile temel hata modellerinin saptanması için, yapay sinir ağının eğitilmesi veya BBN oluşturulması yanında simgesel işleme gerektiren detaylı analizler ve testler ile hatanın tam yerinin saptanması Olay Tabanlı çıkarsama yöntemleri ile gerçekleştirilebilir (Denise et al., 1990).

Bu karma kullanım doğası gereği çok çeşitlilik içeren hata yönetim işi için idealdir. Tüm işlemin tek bir teknik ile analiz edilmesi çeşitlilik karşısında gerekli aksiyonun yeterince sağlanabilmesi açısından verimli olmayacaktır. Bu yaklaşım ile yapay zekâ tekniklerinin farklı avantajları bir araya geldiği gibi birbirlerinin dezavantajlarını da kapatmaktadırlar. Karma yaklaşımın dezavantajı olarak yapay zekâ teknikleri için gerekli olan veri tabanını farklı şekillerde birden çok kere oluşturulması gerektiği söylenebilir (Denise et al., 1990). Örneğin bir yapay sinir ağı çok büyük bir girdi çıktı veri örneklemesi ile eğitilmeli ve Olay Tabanlı Çıkarsama sistemine uzmanlar tarafından temel olaylar beslenmelidir. Bu da ağ yönetimi sisteminin geliştirilme süresini uzattığı gibi çok farklı temel iki konuda uzmanlık gerektirmektedir.

Yapay zekâ teknikleri kullanılarak yapılacak bir hata tanımlama işleminde iki temel adım bulunmaktadır. Bunlar hatanın filtrelenip korelasyon yapılması, ve hatanın tanımlanmasıdır. İlk olarak hata yönetim sistemine gelen yüklü miktarlardaki alarmlar filtrelenmeli ve yönetilebilecek şekilde farklı kategoriler altında korelasyonu yapılmalıdır. Daha sonra bu kategorilere ayrılmış alarmlar analiz edilerek

hatanın temel nedeninin tam olarak tanımlanabilmesi için ağ elemanı üzerinde gerekli testler başlatılır.

4.1.2.2 Hata Filtreleme ve Korelasyon

Hata filtreleme dört farklı süreçten oluşur. Bunlar özetleme, sayım, eleme ve genellemedir (Jakobson et al., 1993). Özetleme, birçok kez tekrar eden alarmların elenerek tek bir alarma düşürülmesidir. Sayım benzer alarmların veya alarm kategorilerindeki yeni alarmların meydana gelme sıklığı için sayılmasıdır. Eleme aşamasında alarmlar derecelerine göre sıralanır ve en kritik olan alarmların işlenmesine başlanır. Genelleme aşamasında ise alarmın daha önceden uzmanlar tarafından tanımlanan genel tanımlar ile karşılaştırılarak herhangi bir genel tanıma uyup uymadığına bakılır. Bu dört aşama tanımlanacak kurallar bütünü veya uzman sistemlerin uygulanması ile sağlanabilir.

Her ne kadar filtreleme yapılsa da, alarm sayıları yüksek olacağından dolayı korelasyon yapılmalıdır. Alarmlar her zaman tam olarak kendilerini tanımlamaya yarayacak veri içermezler. Bu nedenle korelasyon yapılması oldukça zordur. Her ne kadar çok fazla veri olsa bile, ağ sistemlerindeki çeşitlilikten kaynaklanan önemli miktarda belirsiz ve uyumsuz alarm olabilmektedir. Genellikle belli bir grup alarmın meydana gelmesi için birden fazla açıklama bulunmaktadır. Örneğin, uzak bir birimden servis alacak bir sistem isteğine cevap alamadığında zaman aşımına uğrayacaktır. Bu cevap alamama durumunun birçok sebebi olabilir. Cevap isteyen sistem de problem olabilir, tıkanıklık nedeni ile cevap gecikmiştir veya sistem saatinde bir

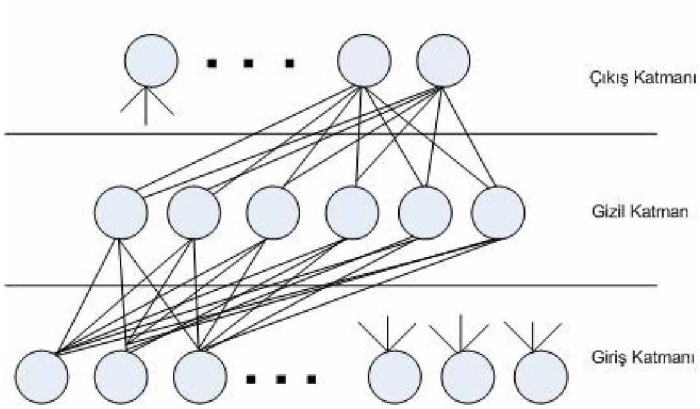
hata bulunabilir. Sistem çökmesi ve arızası gibi durumlar sonucu sürecin yarım kalması alarmların korelasyonunda zorluk çıkaran başka bir sebeptir. Bazı ağ elemanları üzerindeki hatalar sonucu belli bir hata seviyesinden sonra kendisini otomatik olarak kapatıp açacak mekanizmalar içermektedir. Bu gibi işlemler, sistemin kapatılıp açılması sırasında hataların tanımlanmasını sağlayacak alarmların ve verilerin kaybedilmesinden dolayı hatanın hala mevcut olmasına rağmen sistemde her şeyin normal olarak görünmesine neden olacaktır.

Korelasyon işlemindeki en karmaşık işlem yeni gelen bir alarmın daha önceden tanımlanmış alarm setine atanmasıdır (Denise et al., 1990). Yukarıda örneklendiği üzere bu işlem belirsiz, eksik ve uyumsuz veriler içermektedir. Genel olarak örnekleme problemi olarak tanımlanabilir. Bu nedenle olasılığa dayalı yapay zekâ teknikleri olan yapay sinir ağları veya BBN sistemleri alarm korelasyonu için çok uygundurlar. Bu teknikler ile genel davranışlar üzerinden örnekler analiz edilebilmektedir ve belirsizlikler ve eksik, uyumsuz veriler ile çalışabilirler.

4.1.3 Alarm Korelasyonu için Yapay Sinir Ağı kullanımı

İleri beslemeli yapay sinir ağı sistemleri kullanımı tıbbı teşhis sistemleri, çok duyarlı iz takibi, görüntü-veri tanımlama ve sıkıştırmasında etkin olarak kullanımını ispatlamıştır. Alarm korelasyonunun benzer problemlere sahip olmasından dolayı yapay sinir ağlarının alarm korelasyonunda kullanılması mantıklıdır. Çok katmanlı, ileri beslemeli yapay sinir ağlarının bu tür problemleri için oldukça güçlü olmasını şu özellikleri sağlamaktadır:

- Yapay sinir ağları çözümünü bildiği durumlar üzerinden örnekleme yaparak benzer durumları ayırt edebilmektedir.
- Sinir ağının doğru-yanlış ikilileri ve katmanları arasında sınıflandırılması ile fonksiyonlara yaklaşım yapabilmektedirler. Bu sinir ağlarının farklı alarm desenlerinde eğitilebilmeleri için büyük esneklik sağlamaktadır.
- Çok iyi genelleme yapabilmektedirler ve bir fonksiyonun yaklaşımını öğrenebilmektedirler. Bunun için üzerinde çalışılan ortam hakkında çok derin bilgi birikimine sahip olmalarına gerek yoktur. Bu özellikle üzerinde çok fazla bilgiye hakim olunamayacak yeni nesil ağ sistemleri gibi yeni teknolojik alanlarda büyük bir avantaj sağlar.
- Alarmların analiz edilebilmesi için hızlı ve etkin bir metot sunarlar.
- Tam olmayan, belirsiz ve uyumsuz veriler ile çalışabilirler.



Şekil 4.2: İleri beslemeli yapay sinir ağı

Şekil 4.2de görülen ileri beslemeli yapay sinir ağında, sinir hücreleri birbirlerini besleyen katmanlar olarak dizilmişlerdir. Bu model tek bir giriş katmanı, tek bir çıkış katmanı ve hiç olamayacağı gibi, bir veya daha çok gizli katman da içerebilmektedir. Ağ üzerinde tüm bağlantılar ileri yönlüdür ve geriye doğru bir veri besleme yoktur. İleri beslemeli yapay sinir ağları yeterince sinir hücresi ile herhangi bir fonksiyona yaklaşımda bulunabildikleri ve basit girdi çıktı örnek bilgileri üzerinden öğrenebildikleri ve genelleme yapabildikleri için oldukça kullanışlıdırlar. Öğrenme işlemi örnek veriler üzerinden hücreler arası bağlantıların ağırlıklarının ayarlanması üzerinden sağlanır. Öğrenme aşaması sistem geliştirme aşamasında olabileceği gibi gerçek sistem kullanım zamanında da olabilir. Öğrenmenin yapılma şekline göre yapay sinir ağlarının çeşitlerine üstteki bölümlerde değinmiştik.

Öğreticili yapay sinir ağlarında doğruluğu bilinen girdi ve çıktı örneklerinin ağa yüklenmesi ve sinir hücreleri arası ağırlıkların belirlenmesi ile öğrenme sağlanır. Girdi veri kümesi, bu çalışma için ağ üzerinden toplanan alarmlar olarak düşünülebilir, sinir ağı üzerine yüklenir ve bunların çıktıları ile olması gereken doğru sonuçlar karşılaştırılır. Sonuçlara göre aradaki farkı kapatacak şekilde sinir hücreleri arasındaki ağırlıklar ayarlanır. İyi eğitilmiş bir sinir ağı eğitim seti içerisinde yer almayan alarmlar ile karşılaştığında büyük bir oranda doğru sonucu bulacak şekilde çıktılar üretir.

Öğreticisiz yapay sinir ağlarının öğrenmesinde doğruluğu bilinen girdi-çıktı örnekleri içeren bir eğitim veri seti kullanılmaz. Sinir ağı girdi verilerindeki desenleri, düzenleri, korelasyonları veya kategorileri keşfeder ve bunların çıktılarını hesap eder. Örneğin öğreticisiz bir Yapay sinir ağı çıktısındaki değişkenlik düşürüldükçe alarmları farklı kategoriler altında toplayan bir sınıflama servisi olarak çalışır.

Öğreticisiz öğrenmenin uygulanması denetlemeli öğrenmeye göre oldukça hızlıdır. Kullanımı sırasında işletim personeline çeşitli desenlerin gösterilebilmesine olanak sağlar ve bunlar arasından çıktılar arası ilişkilerin belirlenmesinde kullanılabilir. Bu avantajlarından dolayı öğreticili öğrenmenin mümkün olduğu durumlarda dahi denetlemesiz öğrenme tercih edilebilir.

Alarm korelasyonu alanında örnek girdi/çıktı eğitim verileri çok kolay oluşturulabildiği için öğreticili öğrenme metodu tercih edilebilir.

Alarm korelasyonu için farklı bir yapay sinir ağı yaklaşımı ise yapay sinir ağlarının doğrusal olmayan dinamik sistemlerin olasılık tahmin edilmesinde kullanılır (Patton et al., 1994). Bu yaklaşım sinir ağı geçmiş gözlemlerden ve şu anki durum bilgisinden faydalanarak sistemin davranışını belirler. Sistemin şu anki durumu ve tahmin edilen hareketleri karşılaştırılarak elde edilen veriler ile ikinci bir sinir ağı eğitilir. Bu sinir ağı iki durumun karşılaştırılması ile öğrendiği bilgilerle gelen ağ alarmlarını temel kategorilere ayırır.

4.1.4 Alarm Korelasyonu için Bayesian Belief Ağları

Bayesian belief ağları alarm korelasyonu için belirsiz durumlar ile çalışabilmeleri ve sebep ve sonuç ilişkilerini yansıtabilmesinden dolayı diğer bir alternatiftir. Bayesian-belief ağları; hücreler arası sebep sonuç bağlantıları ile belirli olmayan değişkenlerin, hücreler bütünü şeklinde gösterilmesidir. Bir hücre üzerindeki bilgi sebep bilgisini taşıyan kendisinden önceki hücrenin taşıdığı bilgiye bağlıdır. Bu ilişki her sonuç hücresi için kendisinden önceki hücrenin üzerinde taşıdığı olası değişkenlerin olasılık dağılımı ile ifade edilir. Bir sonuç hücresi bir sonraki durumda iletişimi yönlendiren bir sebep hücresi konumunda olabilir. BBN in en önemli avantajı ağ üzerindeki her hücre için bir permutasyon sayısı büyüklüğünde bağlantı dağılım tablolarının geliştirilmesine gerek bırakmamasıdır. Daha doğrusu sadece hücrenin kendinden önceki sebep hücresi için olası durumları ve bunların sonuçlarını bilmesi yeterlidir (Denise et al., 1990).

Bu şekilde BBN ağlarının bilgiyi tutma formu sayesinde çok büyük miktarlardaki birbiri ile sebep sonuç ilişkisi ile bağlı bilgi tanımlanabilir.

Genel olarak;

- BBN ağları kendi bilgi modelleme fonksiyonu ile iletişim ağlarındaki ağ elemanları arasında sebep sonuç ilişkisine dayanan ağ davranışı ve hatalarını, yoğun bilgi gösterimi ile tanımayabilmektedirler.
- Bu sayede hata tanımlamada yol göstericilik yapabilirler. Bir hata alarmı alınması durumunda BBN ağı üzerinde yapılacak

hesaplar ile alarmın varlığı ve hangi nedenlerden kaynaklanabileceği, nerelerin incelenmesi gerektiğine dair detaylı analiz yapılabilir.

- Olasılık teorisi üzerine kurulmuş temelleri sayesinde geçici, belirsiz ve net olmayan veriler ile rahatlıkla çalışabilirler.
- Diğer olasılık temelli metotlara göre parçalı, küçük ve basit anlaşılabilen bir gösterimleri vardır.
- BBN ağı ile hatalar veya hataların farklı kombinasyonlardaki ifadeleri üzerine kesin sonuç metodu kullanılır, bu sayede tanımladıkları problem uzayı etkin ve iyi tanımlanmıştır.

BBN ağları problem uzayını yoğun bir şekilde tanımlaması ve hesaplamaları tam olarak yapması sayesinde otomatikleştirilmiş hata tanımlama ihtiyacı çok uygundurlar (Denise et al., 1990). Bir sistemin analiz modelini tanımlayabilmenin en kısa ve direk yolu olayların hata belirtileri ve hataların altında yatan problemler ile oluşturulmuş bir BBN ağıdır. BBN ağı gözlenebilen belirtiler ve gözlenememiş problemler arasındaki sebep ve sonuç ilişkilerinin gösterilmesidir. Bu sayede belli bir set belirti gözlemlendiğinde en olası problemler bunların sebebi olarak tanımlanabilir. Pratikte BBN ağı, hataları birbirine en çok benzeyen sebeplerle ilişkilendirilmesi ile oluşturulur. BBN ağı bir hata tanımlama işlemi olarak kullanılır ise sistem sonuçlardan sebep çıkarımı yapacak şekilde çalışacaktır.

Tanımlama için geliştirilecek bir BBN ağı üzerinde çalışılacak alanda, o konunun uzmanları tarafından sağlanmış, sebep ve sonuç

ilişkileri hakkında derin bilgilere sahip olmayı gerektirir. Bu hem avantaj hem de dezavantaj olarak görülebilir. Avantajı, yapay sinir ağlarında olduğu gibi bilgi üstü kapalı şekilde sistemde tutulmamaktadır. Bu sayede insanların kolayca anlayabileceği tanımlama sonuçları üretilebilir. Dezavantajı ise yeni nesil ağlar veya MPLS ağları gibi henüz uzmanlığın çok olgunlaşmadığı yeni teknolojiler için hata tanımlarını yapacak sebep sonuç ilişkilerini sisteme uygulayabilecek uzmanların bulunabilmesi oldukça zordur.

4.1.5 Olay Tabanlı Çıkarsama ile Hata Teşhisi

Filtreleme ve korelasyon hata tanımlamasının ilk aşamasıdır. İkinci aşama daha derin araştırma ile alarmların veya hataların asıl sebebini teşhis etmeye yöneliktir. Bu kademeli bir süreçtir. Alarm verileri analiz edilip kararlar verildiğinde, daha fazla veriye ihtiyaç olabilir, daha detaylı bir analiz yapmak gerekebilir veya problem bulunmuştur ve çözümüne gidilebilir (Denise et al., 1990). Daha fazla veri toplanması ağ sistemine ihtiyaca yönelik test çağrılarının yapılması ile veya performansa yönelik verilerin istenmesi ile olabilir. Problem çözümü aşaması ise alan hakkında uzmanlık derecesinde bilgiye gereksinim duyar. Örneğin ağ elemanları hakkında, ağın genel haritası hakkında, genel hatalar ve bu hataların genel olarak nasıl giderildiği hakkındaki bilgiler gibi.

Dolayısı ile sembolik yapay zekâ yöntemleri olarak nitelendirebileceğimiz Olay Tabanlı Çıkarsama veya Uzman sistemler bu fonksiyon için çok uygundurlar. Uzman sistemler için daha önce üzerinde durduğumuz sınırlamaları düşünüldüğünde Olay Tabanlı

Çıkarsama tekniğinin hata teşhisi için en uygun seçenek olabileceğini söyleyebiliriz. Olay Tabanlı Çıkarsama tekniği Uzman Sistemlere karşı şu yönleri ile daha güçlüdürler;

- Benzerlik durumlarını kullanabilmelerinden dolayı yeni ve değişken veriler ile çalışabilirler.
- Yeni olayların edinimi ile bilgi sistemlerini genişletip öğrenebilirler.
- Bilgiyi tanımlama yapılarındaki birleştirme ve katlama kabiliyetleri sayesinde büyük bilgi alanlarında da çok rahat ölçeklenebilirler.
- Değişimler karşısında çok derin ve kapsamlı bakıma ihiyaç duymazlar.
- Bilgi kazanım süreleri Uzman sistemlerin kural geliştirme sürelerine oranla oldukça düşüktür.

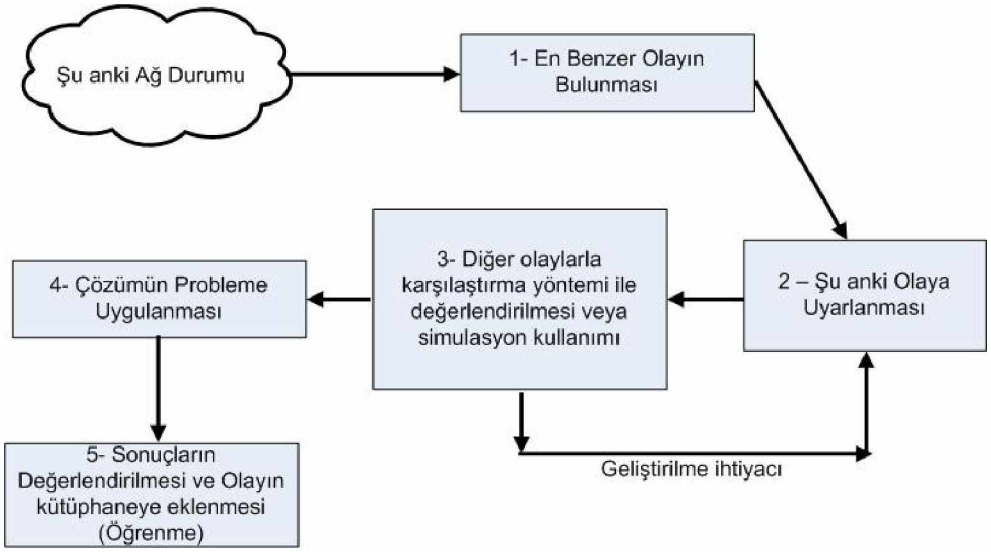
Olay Tabanlı Çıkarsama durumların belli bir düzende tekrarlaması üzerine dayanmaktadır (Denise et al., 1990). Uzmanların çalışmaları ve problem çözme yaklaşımları araştırıldığında problemlere geçmiş tecrübelerden edinilen bilgiler ile çözüme gittikleri görülmüştür. Olay tabanlı çıkarsama yöntemi de bir uzmanın bir olayı teşhis etmek için olay kütüphanesindeki benzerliklerini kullanması olarak görülebilir.

Olay Tabanlı çıkarsama yönteminin önemli bir parçası olayın asıl bileşenlerinin neler olduğunun kararlaştırılması ve bilgi bankasında hangi bileşenleri ile adresleneceğidir. Genellikle bir Olay Tabanlı Çıkarsama yöntemi kullanan bir uygulamanın ön yüzü, bir uzmanlık ve sistemin tüm

altyapısının bilinmesi gerek kalmadan yeni olayların kolayca girilmesini sağlar. Örneğin yeni olaylar tamamen metin ve bağımsız olarak sisteme girilirler ve sistem bu yeni olayların kendi üzerinde adreslenmesini otomatik olarak yapar. Olay tabanlı çıkarsama yöntemlerinin en büyük avantajlarından biri daha önce de söylemiş olduğumuz gibi öğrenebilmeleridir. Bu öğrenmenin bir şekli sisteme yeni olayların sebep ve sonuç bilgileri ile girilmesi şeklinde olur. Diğer bir şekli ise Olay Tabanlı çıkarsama sisteminin olaylar arasında bileşenlerin benzerliğine göre olayları birleştirmesi veya ayırması sayesinde belli bir olaya adreslemesi şeklinde olur.

Olay Tabanlı Çıkarsama ile problemin çözülmesi Şekil 4.3 de görüldüğü gibi beş adım olarak açıklanabilir.

1- Alınma
2- Açıklama ve uyarlama
3- Değerlendirme ve çözüm
4- Uygulama
5- Değerlendirme ve öğrenme



Şekil 4.3: Olay tabanlı çıkarsama süreci

İlk adım şu anki olaya en çok benzeyen olayların veri tabanından alınmasıdır. Bu nedenle karar ağaçları veya en yakın komşu uyumu ile olayların düzgün bir şekilde adreslenebilmesi çok önemlidir. Yeni bir olay alındığında açıklanmalı ve uyarlanmalıdır. Değerlendirme aşaması alınan örnek olaylar ile yeni olay arasındaki karşılaştırmadır. Uyarlama aşaması ise uzmanlık isteyen karmaşık bir adımdır. Alan bağımlı bu süreç aralarındaki benzerlik ve farklılıklara göre olayın problem durumuna uyarlanması ve çözümün geliştirilmesidir. Bir sonraki adım değerlendirme ve çözüm adımıdır. Bu adımda önerilen çözüm benzer olaylar ile karşılaştırılır veya simülasyon kullanılır, ve çözüm bu verilere göre yeniden değiştirilir. Olay Tabanlı Çıkarsama sistemi en uygun çözümü bulduktan sonra çözüm uygulanır ve sonuçlar değerlendirilir. Sonuçların değerlendirilmesi, çözüm adımları ve problem bilgileri yeni

bir olay olarak sisteme kaydedilir, adreslenmeleri yapılır ve bu sayede sistemin öğrenmesi gerçekleştirilir.

4.1.6 Hata Tanımlaması ve Yapay Zekâ'nın Melez Kullanımı

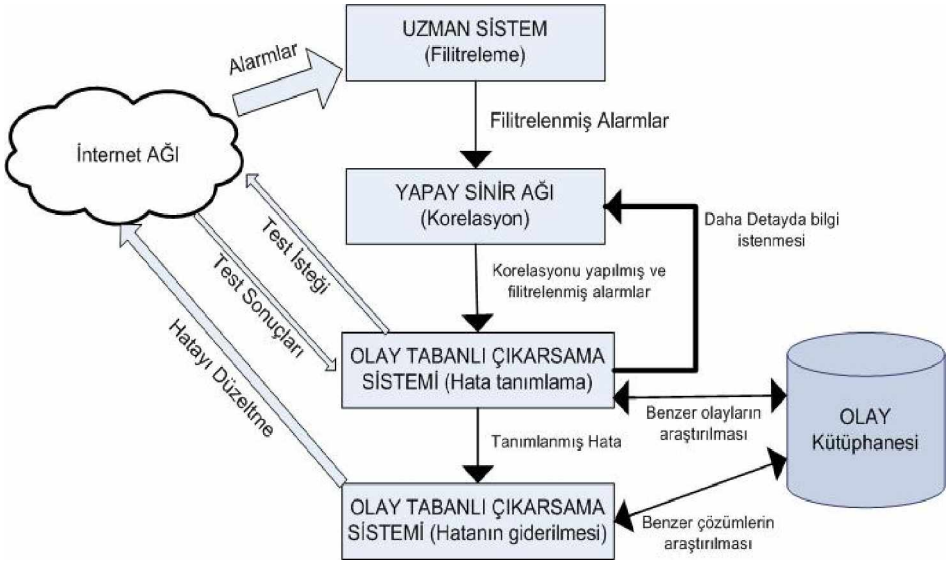
Buraya kadar yaptığımız incelemelerimiz sonrasında hata tanımlama sürecini en iyi şekilde yapacak çalışma olasılığa dayalı ve semboliksel problem çözme tekniklerini birlikte barındıracak melez bir yapay zekâ kullanımı olacaktır. Sekil 4.4'de Yapay Sinir Ağları ve Olay Tabanlı Çıkarsama tekniklerinin birlikte kullanımı ile tasarlanmış bir hata tanımlama sürecini bulunmaktadır.

Ağ sistemi üzerinden toplanan alarmlar bir Uzman Sisteme beslenmektedir. Bu aşamada daha önce detaylarına değindiğimiz sıkıştırma, sayım ve genelleme ile filtreleme gerçekleşir. Yapay sinir ağlarının desen tanımlama fonksiyonu ile alarm korelasyonu ve desenlerinin çıkarılması için kullanılabilir. Filtrelenmiş alarmlar genel alarm kategorilerine ayırmak üzere eğitilmiş yapay sinir ağına beslenirler. Çıktı bilgisi alarmın hata tipi, protokol tipi ve coğrafi bilgisidir. Buradan elde edilen bilgi Olay Tabanlı Çıkarsama sistemine beslenir. Burada daha detayda analiz ile test aşamaları gerçekleşir. OTÇ sistemi geçmiş tecrübelerine dayanarak daha fazla veri toplamak, problemin çözümünü bulmak, yeniden yapay sinir ağına beslemek veya daha detaylı analiz yapacak başka bir yapay sinir ağı mekanizmasına beslemek gibi kararlar alabilir.

Hata yönetimi için bir yapay sinir ağı oluşturmanın ilk adımı öğrenme aşamasıdır. Yapay sinir ağı örnekler üzerinden genellemelere giderek mevcut desenleri tanımlamayı öğrenir. Çıktıları bilinen birçok hata verisinin elde olduğu durumlarda en etkin yol öğretmenli öğrenim tekniği ile yapay sinir ağının geliştirilmesidir. Yapay sinir ağı hazır olduktan sonra OTÇ sistemi ile bütünleştirilebilir (Hanemann et al., 2004).

Yapay sinir ağı sistemi sayesinde alarmların filtrelenmesi ve korelasyonunun yapılmasının ardından OTÇ sistemi elde edilen veriler üzerinden daha ileri nasıl analizlere ihtiyaç olduğuna karar verir. Bu kararı verirken de kendi olay kütüphanesinden benzer olayları araştırarak faydalanır. En benzer alarmları kendi kütüphanesinde bulduktan sonra benzerlikler ve farklılıkları kullanarak yeni bir çözüm üretir. Bu işlem hatanın tanımlandığına karar verene kadar tekrarlamalı olarak devam eder.

Hata tanımlamasının ardından tüm problem çözme süreci analiz edilmelidir. Test sonuçlarının incelenmesi, işletilen adımlar, girilen döngüsel durumlar, çıkmaz durumlar ve başarılı sonuçlar yeni bir olay olarak OTÇ sistemi olay kütüphanesine kaydedilir. Bu sayede hata tanımlama sistemi kendi kendine gelişecektir.



Şekil 4.4: Hata yönetimi için melez yapay zeka uygulaması

4.1.7 Hata Giderilmesi

Otomatik hata tanımlama sistemlerinin diğer bir hedefi de otomatik olarak hatanın giderilmesidir. Bu işlem ağ elemanlar, ağ haritası, eski ve yeni teknolojiler hakkında uzmanlık gerektirmektedir. Hataların giderilmesinde karşılaşılan genel teknik problemler şu şekilde listelenmişlerdir (Dupuy et al., 1989). İş birliği olmadan düzeltme, elle yapılan düzeltmeler ve eski teknolojiler.

İş birliği olmadan düzeltmeler bir set ağ elemanında performans çakışmasına neden olabilir. Bu da dalgalanarak başka ağ elemanlarının da performans problemlerine neden olabilir ve dolayısı ile başka hatalar oluşur. Örneğin fiber optik kablo kopar ise iş birliği yapılmadan bu bozukluğun düzeltilmeye çalışılması başka hatların, bölümlerin ve

servislerin bozulmasına ve hiç birinin tam olarak düzeltilememesine neden olabilir. Bu gibi düzeltme çalışmalarında genellikle ağ üzerinde sıkışıklığın oluşması ortaya çıkmaktadır.

Hata giderilme aşamasında en çok problem çıkan çözümlerden bir diğeri de bir lojistik problemi diyebileceğimiz elle yapılan düzeltmelerdir. Çoğu zaman bu tür düzeltme işleminde cihazların kapatılıp yeniden açılmakta. Ve bu da aslında kapatıp açma işlemini gerektirmediği halde genellikle yapıldığı için başka problemlerin tetiklenmesine neden olmaktadır. Eğer cihazlar erişim problemleri ile uzaktan erişime müsait durumda olmayabilir. Bu koşullarda sahadaki bir çalışanın bulunması ve uzaktan erişim problemine müdahale etmesi kaçınılmazdır. Özellikle çok büyük ağlarda bu türden müdahaleler çok daha sık yapılmaktadır.

Eski teknolojiler başka bir teknik problem olarak karşımıza çıkıyor. Zaman içerisinde ağlar geliştikçe ve büyüdükçe birçok yeni teknoloji cihazlar ve standartlar alınıyor ve ağa ekleniyorlar. Çoğu zaman yeni ve geliştirilmiş olan cihazlar eski cihaz ve standartlarla uyum problemi yaşamaktadırlar. Örneğin eski ve yeni tipte anahtar her ikisi de problem durumunda yeniden ayağa kalkabilmek üzere fonksiyonlar içerebilir. Problem olduğu zaman yeni olan anahtar çok daha hızlı ayağa kalkabildiği için eski olan anahtar hala işleve geçmeye çalışırken birden ağ üzerinde çok fazla miktarda veri akışı ile karşı karşıya kalabilmekte ve yeni problemler oluşabilmektedir.

Bu tür sorunlar, ağ alanı hakkında bilgi birikimi sahibi olan ve problem çözme metotları ile sembolik yapay zekâ tekniklerinin kullanımını ortaya çıkarır. Örneğin sıradüzensel olarak planlanmış uzman sistem ve OTÇ içeren bir yapay zekâ sistemi olabilir.

Yapay zekâ planlaması ile karmaşık lojistik planlamaları, sıralamalar, düzeltmeler ve özellikle zamanlama ile ilgili ihtiyaçlar; iş birliği yapılmaması dolayısı ile ortaya çıkabilecek veya eski teknolojilerden kaynaklanabilecek karışıklıkların giderilmesinde kullanılabilir. Günümüzdeki planlama sistemlerinin bu ihtiyacın giderilmesindeki en büyük güçlüğü gerçek ağlarda uygulanırken hala çok yavaş kalmasıdır. Bu nedenle hala ilk örnek uygulamalar olarak kullanılabilmektedirler. Yakın bir zamanda yapay zekâ içeren planlama sistemleri geliştirildikçe ve makineler hızlandıkça tercih edilecek sistemler olacaktır.

Ayrıca OTÇ sistemleri hata giderimi planlanması aşamasında eski tecrübelerden faydalanabilmektedirler. Örneğin cihaz manüel düzeltmelere cevapsız kaldığı durumlarda, daha önceki benzer olayların incelenmesi ve uygun planlamanın yapılması sayesinde OTÇ sisteminin problemi gidermek üzere aksiyonda bulunması sağlanabilir.

Uzman sistemler hata giderilmesi için OTÇ den sonra en uygun tekniklerden biridir. Yukarıda listelediğimiz nedenlerden dolayı OTÇ sistemleri daha iyi bir seçenek olacaktır. Bunun en önemli nedeni OTÇ sistemlerinin yapılacak hatalar üzerinden öğrenebilmesidir. Örneğin iş birliği yapılmadan yapılan bir düzeltme sonucu oluşan sıkışıklık OTÇ

sistemi tarafından negatif bir örnek olarak kaydedilecek ve bir sonraki hatalarda yeniden tekrarlanmayacaktır. Kural tabanlı sistemler ise sürekli, bir uzman tarafından mevcut ve yeni kurallar düşünülerek elle güncellenmelidir.

Geçmiş tecrübelerin kullanımına dayalı hata düzeltilmesi eski teknolojilerden kaynaklanacak hataların giderilmesinde de çok kullanışlıdırlar. Çoğu zaman problemlerde yeni teknolojilere uygulanan çözümler eski teknolojiler ile hemen hemen aynı işlevleri içermelerinden dolayı benzerdirler. Farkları ise teknolojiler arası performanstır. Örneğin eski ve yeni anahtarların bir hata durumdan toparlanması için uygulanacak adımlar aynıdır. Farklı olan eski anahtarın yavaş olmasından dolayı sadece zamanlamadır. Burada koyulacak kurallar ile eski anahtar üzerine bir anda çok yük binmesini engelleyecek şekilde gereken zaman farkı çıkarılarak anahtarların üzerinden veri geçirecek zamanlama düzenlenir.

4.2 Diğer Ağ Yönetim İşlevlerinde Yapay Zekâ

4.2.1 Performans Yönetiminde Yapay Zekâ Kullanımı

Hata yönetim aşamasında ağ üzerindeki donanım veya yazılım hataları sonucu oluşan alarmlar üzerine hata teşhisi yapılmaya çalışılıyordu. Performans yönetiminde ise amaç ağ üzerinde ciddi bir donanım veya yazılım arızası olmadan elde edilen performans verilerindeki düşüklüklerin takip edilmesi ile büyük bir kayba engel olmaya yönelik yapılan yönetim işidir (Hanemann et al., 2004).

Performans düşüklüğü probleminin birçok farklı sonucu olabilir. Bir noktada oluşan performans düşüklüğü başka bir noktayı tetikleyebilir. Bu durumda performans düzeltmesi ancak bir önceki problemin yani performans düşüklüğünün giderilmesi ile sağlanabilir.

Performans yönetiminde yapay zekâ uygulamalarından yapay sinir ağları ve gezici ajan uygulamaları yapılmıştır. Her ne kadar tam olarak istenen sonuçlar elde edilemese de yapılan çalışmalar gün geçtikçe daha iyi sonuçlar vermektedir (Bieszczad et al., 1998).

Temel olarak performans yönetimi ağ elemanları üzerindeki performans verilerinin periyodik olarak toplanmasına dayanır. Toplanan bu istatistiksel veriler ağın durumu hakkında yorumlar yapmak üzere kullanılır ve ayrıca ileride olacak durumlar için ağ kapasite planlamasında kullanılır. Performans yönetiminde kullanılacak bu veriler klasik yaklaşımda ağ elemanına periyodik sorgularla toplanırlar. Bu sorguların yapılırken ve verilerin toplanırken ağın normal işleyişini aksatmaması çok fazla hat işgali yapmaması çok önemlidir. Bu performans yönetiminde en önemli problemlerden biridir. Bu probleme cevap olarak gezici etmen kullanımı başlamıştır. Gezici etmenler vasıtası ile performans verileri tüm ağ hattı boyunca ağ elemanından ağ yöneticisine kadar taşınmadan ağ elemanı üzerinde işlenirler. Gezici etmenlerin genel davranışları ağ yöneticisi tarafından istenen performans gereksinimlerine göre değiştirilebilir. Ağ elemanı üzerine yüklenen bir gezici etmen yönetim ajanı sayesinde ağ yöneticisi direk ağ elemanına müdahale edebilecek şekilde bir fonksiyon zenginliği kazanır. Elbette bu

imkân performans yönetimi konusunda da oldukça faydalı bir şekilde kullanılabilmektedir. Her ağ elemanı üzerine yeni bir yazılım (gezici etmen) yüklenmesine imkân verecek yapıda olmayabilir. Bu durumda ara yönetim sistemleri üzerinde gezici etmenlerin yüklenmesi sayesinde çözüme ulaşılabilir (Bohoris et al., 2001).

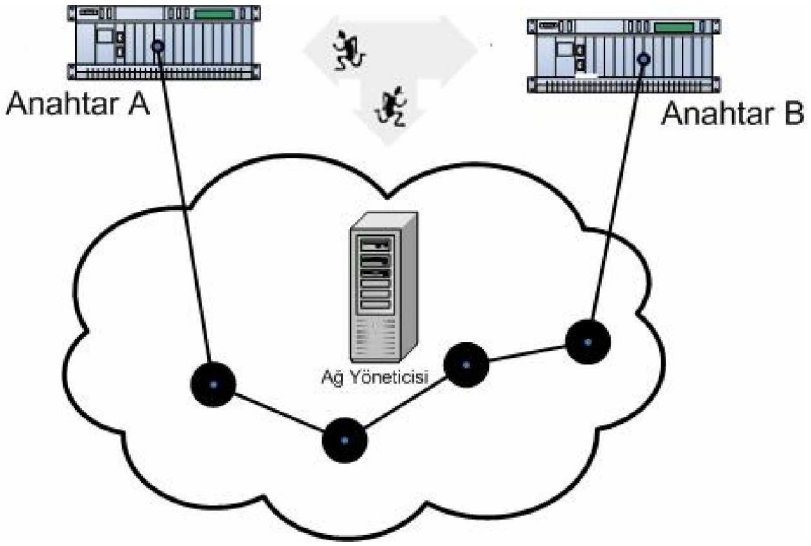
4.2.2 Yapılandırma Yönetiminde Yapay Zekâ Kullanımı

Yapılandırma yönetimi ağ durumunun ağ yöneticisine raporlanması ve ağ üzerindeki kaynakların yapılandırma bilgilerinin takip edilmesidir. Bu işlem SNMP veya CMIP gibi klasik ağ yönetimi protokolleri ile de rahatça yapılabilmektedir.

Fakat ağ üzerinde yeni bir cihaz eklenmesi veya kaldırılması gibi fiziksel bir değişim sonucu meydana gelecek, ağın durumundaki değişiklikler; örneğin yeni bir bağlantının eklenmesi, kaldırılması veya bir ağ düğümünün iptal edilmesi gibi ağın yeniden yapılandırılması da ağ yapılandırma yönetimin bir parçasıdır (Bieszczad et al., 1998). Böyle bir çalışma için yapay zekâ teknikleri etkin olarak kullanılabilmektedir Ağın yeniden yapılandırılması çalışması ağ hata yönetimi işi kadar önemli bir çalışmadır. Böyle bir ihtiyaçta ağ yönetim sisteminin ağ üzerindeki hataları tanımlayıp düzeltmeden önce ağda meydana gelen yeniden yapılandırma çalışmasını tamamlamalıdır.

Yapılandırma yönetimi altıda yer alan diğer bir aktivite ise müşterilere ilgili ağ servislerinin sağlanmasıdır. Bu işlem içinde genellikle birden çok eleman katılır. Servis sağlanması işlemini daha

etkin yapabilmek için yapay zekâ tekniklerinden gezici etmenler kullanılabilmektedir. Müşteriye bir ağ servisinin verilebilmesi için ağ üzerindeki birden çok ağ elemanında ve birden çok yerde yapılandırma yapılması gerektiği için gezici etmenler bu ihtiyaca tam olarak cevap verebilmektedir (Marzo et al., 1999). Buna örnek olarak ATM temelli bir ağ üzerinde geçici bir sanal devre (PVC) ihtiyacı için birçok ayar yapılması gerekir. Eğer bu devrenin kurulacağı iki anahtar ağ elemanı farklı üreticilerden alınmış ise işlem daha da zorlaşır. Her ağ elemanının da bu yeni devre için gerekli yapılandırmasının ağ yöneticisi aracılığı ile yapılması gerekir. Çok farklı üreticilerin ağ cihazlarını barındıran heterojen ağ ortamlarında ağ yönetiminin tek bir ağ yöneticisi tarafından yapılabilmesi genellikle olanaksızdır. Dolayısı ile gerekli devrenin yapılabilmesi için gereken tüm ağ yöneticileri üzerinde operatörün gerekli ayarları girmesi gerekir. Fakat bu işlem aşağıdaki şekilde görüldüğü gibi gezici etmenler ile otonom bir şekilde gerçekleştirilebilir (Marzo et al., 1999).



Şekil 4.5: Gezici etmenler ile otonom ağ yapılandırması

4.2.3 Güvenlik Yönetiminde Yapay Zekâ Kullanımı

İletişim ağları genellikle bir veya birçok noktadan internete açık ağlardır. İletişimin gereği ve gelişmeleri sonucunda bu açılım artarak devam edecektir. Hepimizin bildiği üzere internet bir çok zararlı yazılımın (virüs, spam, truva atı gibi..) bulunduğu ve gün geçtikçe arttığı bir ortamdır.

Ağ yönetim sistemleri içerisinde güvenlik önlemleri genellikle ağ yönetimine olan erişimlerin düzenlenmesi ve yetkilendirilmesi şeklinde gerçekleştirilir. Ağ elemanlarının güvenliği ve ağın genel güvenliği ise ağ yönetim mimarisi içerisine güvenlik için özelleştirilmiş cihazların ve yazılımların eklenmesi ile sağlanır. Bu yazılım ve cihazlara anti virüs sistemleri, anti spam sistemleri, ateş duvarları (FW), saldırı tespit sistemleri (IDS), saldırı önleme sistemleri (IPS) örnek verilebilir.

Günümüzde anti spam, IDS, IPS ve anti virüs sistemlerinde yapay zekâ tekniklerinden oldukça çok faydalanılmaktadır. Doğası gereği bu gibi güvenlik önleme sistemlerinin geleneksel yazılım geliştirme metotlarının dışına çıkarak, az verinin bulunduğu ortamlarda sezgisel ve bulanık mantık yürüterek tahminle ve tecrübeye dayalı yeteneklerinin bulunması gerekir. Çünkü virüs ve spam gibi yazılımlar aynı biyolojik virüsler gibi gün geçtikçe çeşitlenmekte ve şekil değiştirmektedirler. Gerçekleştirilecek olan güvenlik yazılımlarının bu değişme ve çeşitlenmeyi atak desenleri üzerinden fark edebilmesi gerekir. Dolayısı ile yapay zekâ yöntemleri güvenlik yönetimi için etkin olarak kullanılabilecek bir alandır.

Ağ sistemlerinde kullanılan bazı güvenlik sistemlerini açıklayacak olursak:

Ağ Saldırı Tespit Sistemleri (NIDS) : Bu sistemler ağa ulaşan paketleri dinlerler ve kötü niyetli kişilerin sisteme sızmak isteyip istemediklerini tespit etmeye çalışırlar. Bu saldırıların tipik bir örneği hedef makineye yapılan birçok TCP bağlantı isteğidir (SYN). NIDS sistemi sadece tespit işleminin yapılacağı hedef makinede çalışabileceği gibi, bu iş için adanmış bir cihaz üzerinde (yönlendirici, anahtar, vs.) de çalışıp bütün ağa yapılan saldırıları dinleyebilir. Bu cihazların son yıllarda geliştirilmiş olan bir ileri modeli Ağ saldırı önleme sistemleridir (NIPS). Bu sistemlerin kullanılması ile birlikte tespit edilen saldırıların insan müdahalesine bırakılmadan çok hızlı bir şekilde az zarar ile

giderilmesi mümkün olabilmektedir. Bu sistemler de büyük bir verimle yapay zekâ tekniklerini kullanmaktadırlar.

Anti virüs Sistemleri: Ağ üzerindeki virüs hareketlerini tespit ederek virüs kaynağını bildirir. Ağ yönetim sistemleri ile bütünleşmesi sağlanarak eğer gerekiyorsa virüs yaymakta olan sistemleri veya hatları karantina altına alınarak ağın geri kalan bölümünü virüslerden korumaya çalışılır.

Sistem Bütünlük Doğrulayıcılar (SIV): Bu tarz sistemler, kullandığımız ortamdaki kritik dosyaların değişip değişmediğini kontrol ederler.

Aldatma Sistemleri (DS): Bazı çok bilinen açıkları taklit ederek sisteme sızmak isteyen kişileri yanıltıp, onların tespit edilmesini sağlayan sistemlerdir.

5 ÇOK PROTOKOLLÜ ETİKET ANAHTARLAMA

Çok Protokollü Etiket Anahtarlama (MPLS) IETF tarafından ağ üzerindeki trafik akışının yönlendirme, iletme, anahtarlama esnasında etkin işaretlenme için belirlenmiş bir protokoldür. Genel olarak işlevi şu şekilde sıralanabilir (MPLS, 2006);

- Birbirinden farklı cihazlar, makineler hatta uygulamalar arasında ki çeşitli büyüklüklerde olabilecek trafik akışının yönetilebilmesi için mekanizmalar sunar.
- 2. Katman ve 3. Katman protokollerden bağımsızdır.
- Birbirinden farklı paket yönlendirme ve paket iletme teknolojileri tarafından kullanılabilecek, IP adreslerinin yönlendirilmesi için sabit uzunlukta etiketler ile imkân sağlar.
- Mevcut yönlendirme protokolleri (RSVP, OSPF) için bir önyüz imkânı sunar.
- IP, ATM ve Frame-Relay 2. Katman protokollerini destekler.
- Paket-temelli ağlarda trafik mühendisliği yeteneği sağladığı gibi, IP'ye Servis Kalitesi QoS (Quality of Service) gücünü de vermektedir. Ayrıca IP-temelli VPN (Virtual Private Network) uygulama olanağını da getirmektedir.

Bu özellikler, özellikle çoklu servis, çoklu kullanıcı sağlamayı amaçlayan kuruluşlar için çok önemlidir.

MPLS, klasik uçtan uca işlem yapan metotlardan çok farklıdır. Kısa, sabit uzunlukta ve kolaylıkla iletilebilecek etiketleri, IP paket başlıklarının kısa yoldan tanıtımını sağlamaktadır (MPLS, 2006).

TCP/IP protokolü tüm anonim ve özel veri ağlarının temelini oluşturmaktadır. Veri, ses, görüntü ve çoklu ortam ağlarının yakınlaşması sonucu oluşacak ağlarda da IP-temelli protokoller egemen olmaya başlamıştır. Etiket anahtarlama, bu gelişimde endüstrinin bir desteği olarak ortaya çıkmaktadır.

Orijinal TCP/IP yapısının geliştirilmesinde, yalnızca firma ürünleri arasındaki farklılıkları arttırmakla kalmayıp, tümleşik anonim ağların doğmasına neden olmakta ve böylece endüstrinin bir önemli katkısı olarak ortaya çıkmaktadır. Örneğin, IP ağları gerçek zamanlı paket gönderimini sağlayabilecek şekilde gelişmesi gerekmektedir. Bu durum da, IP ile ATM protokolünün, VPN'lerin ve çok geniş çaptaki ağların tümleşmesini gündeme getirmektedir. Bunun sonucu, pek çok sistem ve yönlendirici ağa bağlanabilir ve tüm kullanıcılar için gerekli bant genişliği sağlanabilir. Bu etken gelişme, anahtarlamada fiyat/başarım yüzdesini arttırır ve genel giderlerde büyük düşüşlere neden olur (MPLS, 2006).

QoS desteği için etiket anahtarlamanın kullanılması, trafik mühendisliği için bazı özelliklerin sağlanması çözümün bir parçası olarak ortaya çıkmaktadır.

Etiket anahtarlama çözümleri, IP kontrol protokolleri ile etiket değiş tokuşu paket gönderimi ile birleştirilmesi ve etiket dağıtım mekanizmasının kullanımı ile karakterize edilebilir.

Etiket anahtarlama, etiket anahtarlama teknolojisinin önemli bir parçası olan IP ve ATM protokol modelleri arasında oluşturulacak haritalama işlemlerinin güçlüğü nedeniyle, IP ile ATM arasında tümleşmenin ortaya çıkardığı sorunlarla boğuşmaktadır (MPLS, 2006).

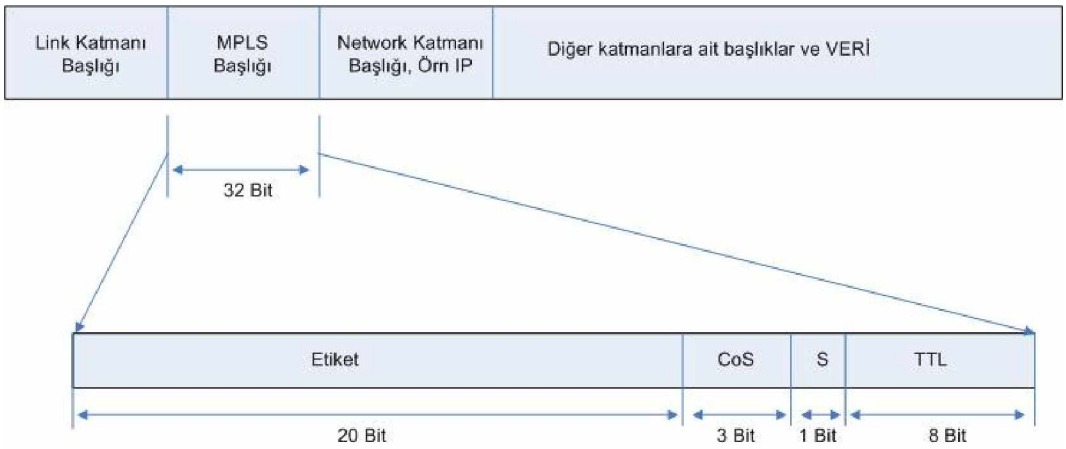
5.1 MPLS Bileşenleri

LER (Label Edge Router). Ağlarda en uç noktalarda bulunurlar ve paketlerde ilk seviye işlemleri ve sınıflandırmaları başlatırlar. Yönlendirici veya anahtar olabilirler.

LRS (Label Switch Router). Ağın merkezinde bulunurlar ve hazırlanmış anahtarlama tablolarındaki bilgilere uygun olarak etiketleri anahtarlar. Yönlendirici veya anahtar olabilirler.

Etiket. Etiket, oldukça kısa, sabit uzunlukta olan ve gönderme işlemlerine yardım için kullanılır başlıklardır. Şekil 5.1'de genel MPLS etiket yapısı görülmektedir. Etiketler yaratılışları aşamasında FEC'lerle sıkı ilişkidedirler. Etiketler gerçekte bir adres değillerdir. Onlar yerel ve tek bir veri-bağlantı içindedirler ve geniş çapta hiç bir anlamları yoktur. Etiketler Frame Relay ağlarda DLCI (Data Link Control Identifier) veya ATM ağlarda kullanılan VPI/VCI'ra (Virtual Path Idenfifier/Virtual Cannel Identifier) benzerler. ATM, anahtarlama kararlarının kolaylıkla

verilebilmesi için yaratılmış bir teknoloji olduğundan, etiket anahtarlamada ATM üzerinden IP paketlerinin iletilmesinde kullanılabilecek etken bir yol olarak görülmektedir (MPLS, 2006).



Şekil 5.1: MPLS genel etiket yapısı

Etiket Anahtarlama. Evvelce konu ettiğimiz gibi çok etken bir etiket değiş tokuş algoritması kullanması nedeniyle etiket anahtarlama, klasik yönlendirmeden çok daha etken bir dağıtım mekanizmasıdır.

Etiket Dağıtım Protokolü (LDP). LDP, LER ile LSR aygıtları arasındaki iletişimi sağlar. Bunun için, LSP (Label Switched Path, uç noktalar arasında oluşan yol) oluşturmak üzere LER ve LSR'larda etiketleri görevlendirir

Yönlendirme. Yönlendirme, ağ tarafından paketlerin kendi içinden geçirilip gönderilmesini tarif etmek için kullanılır. Bir bilgisayar ağında pek çok yönlendirici çok çeşitli şekilde ağ'a bağlanabilmektedir. OSPF, RIP ve BGP (Border Gateway Protocol) örnek bazı yönlendirme protokolleridir. Yönlendiriciler, yönlendirme tablolarını oluşturmada yönlendirme protokollerini kullanırlar. Bu tablolar, paketin ulaştığı yönlendiricide paketin daha sonra gideceğe adresin yönüne uygun olarak gönderilecek daha sonraki ucun belirlenmesini sağlar. Tabloların oluşturulması ve gidilecek yönün belirlenmesi için harcanan zaman mantıksal işlemler için ayrılan zamandan farklıdır.

Anahtarlama. Anahtarlama her hangi bir aygıtta ikinci katman temel alınarak (örneğin ATM VPI/VCI gibi) verinin giriş kapısından girdikten sonra çıkış yapısına yönlendirilmesi, ifade edilmektedir.

Kontrol Yapıtaşı. Düğüm için gönderme tablosunun yaratılması ve bakım ve onarımının yapılaması işinde kullanılır. Yönlendirme bilgilerini düzenli ve sağlıklı olarak dağıtımında diğer düğümlerin kontrol yapıtaşları ile birlikte çalışır. Ayrıca bu yapı taşları gönderme tablolarının yaratılmasında kullanılan düzenli yerel işlemlerin güven altına alınmasını da sağlar.

Gönderme Yapıtaşı. Bunlar direkt olarak paketlerin gönderilmesinde görev yaparlar. Yönlendiriciler tarafından bakım ve onarımları yapılan gönderme tablolarındaki bilgileri kullanırlar Klasik yönlendiricilerde, çok büyük bir algoritma, gönderme tablosuna yerleştirilmiş paketteki gidilecek adresleri karşılaştırarak en uygun

kullanılabilir yolu seçinceye kadar, çalışmasını sürdürür. Bu karar-destek işlemleri paket kaynağa varıncaya kadar her düğümde tekrarlanır. Buna karşılık LSR'larda, ki bunlar ağın uçlarında veya çekirdeğinde bulunabilirler, etiket değiş tokuş algoritması, paketlerdeki etiketler ve etiket-temelli gönderme tablolarını kullanarak, paket için yeni etiketler ve çıkış ara yüzleri elde edilmesini sağlarlar.

Gönderme Tabloları. Gönderme yapı taşlarına anahtarlama fonksiyonunu yerine getirmesi aşamasında gerekli desteği sağlayan bilgi topluluğudur. Gönderme tablosu, gelen her paketin gideceği adres ile ilişkide olması gerekir. Böylece paketin bundan sonra gideceği yer için gerekli bilgiyi sağlamış olacaktır.

FEC (Forwarding Equivalence Class). Bu tanım; gönderim aşamasında eşit tarzda işlem görecek paket grupları için kullanılır. FEC; gidecekleri adresleri özel bir IP adresleri unvanı altında birleştirilmiş paketler seti olarak ta tanımlanabilir. Bir diğer FEC ise kaynak ve gidilecek adresleri ayni olan paketler grubu olarak tanımlanır.

6 YAZILIMIN İŞLETİLMESİ VE DEĞERLENDİRİLMESİ

Bu tez çalışması kapsamında çok protokollü etiket anahtarlama (MPLS) ağlarında kullanılan Etiket Anahtarlayıcı Yönlendirici (LSR) ağ elemanlarının yönetimi için bir ağ yönetim yazılımı gerçekleştirilmiştir. Yazılım daha önce işlemiş olduğumuz tüm ağ yönetim ihtiyaçlarını karşılayacak şekilde JAVA programlama ortamında geliştirilmiştir. Geliştirme sırasında AVICI model LSR'lar temel alınmıştır. LSR'lardan oluşan ağ elemanlarının hazır bulunabileceği bir laboratuar ortamı mevcut olmadığı için yazılım geliştirme süresi boyunca simülatörler kullanılmıştır. MIMIC (MIMIC, 2006) yazılımı bu simülatörlerden biridir.

6.1 Geliştirme Ortamı ve Kullanılan Kütüphaneler

Yazılım Java 1.5 kullanılarak geliştirilmiştir. Kullanılan birimler ve kütüphaneler aşağıdaki şekildedir;

Yazılım geliştirme Ortamı: Eclipse (Eclipse, 2006), Borland Jbuilder.

Kod sürüm kontrol sistemi: CVS (CVS, 2006)

Sürüm ve derleme sunucusu: Redhat Linux A.S3

Yazılım Dili: Java

Veri Tabanı Sunucusu: Oracle v9.2.0.1

Uygulama Sunucusu: Apache Tomcat v5.x

İstemci Arayüzü Java Kütüphanesi: Swing

SNMP Kütüphanesi: SNMP4J (Snmp4j, 2006)

Uzman Sistem Kütüphanesi: JESS

Web Service Kütüphanesi: Apache Axis (Axis, 2006)

Raporlama Kütüphanesi: JfreeChart

Log Kütüphanesi: Log4J

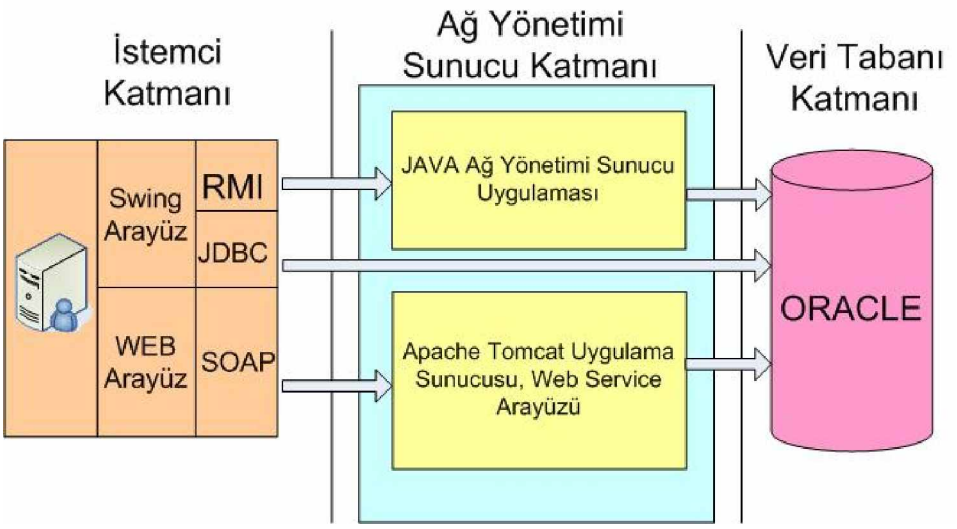
Derleme Kütüphanesi: Apache ANT

Network Simülatörü: MIMIC, Adventnet (AdventNet, 2006), Net-SNMP (Net-SNMP, 2006) , MGSOFT (Mg-Soft, 2006)

6.2 Yazılımın Tasarımı

Yazılım temel olarak 3 katmandan oluşmaktadır. İstemci, Uygulama Sunucusu ve Veri Tabanı katmanları. İstemci ara yüzü Java Swing Kütüphanesi kullanılarak zengin istemci anlayışı ve avantajlarını kullanarak işletimsel ihtiyaçlara etkin cevap verebilecek şekilde geliştirilmiştir.

Yazılımın istemci ve Sunucu katmanları Java ile geliştirildiklerinden ortam bağımsız olarak Java Çalışma Ortamı yüklü her türlü işletim sistemi ile çalışabilir şekildedir. Bu tez çalışması sırasında istemcilerde kullanım kolaylığı açısından Windows işletim sistemi, sunucu katmanında ise Redhat Linux işletim sistemi kullanılmıştır. Yazılımın genel mimarisi şekil 6.1 de görülebilir.

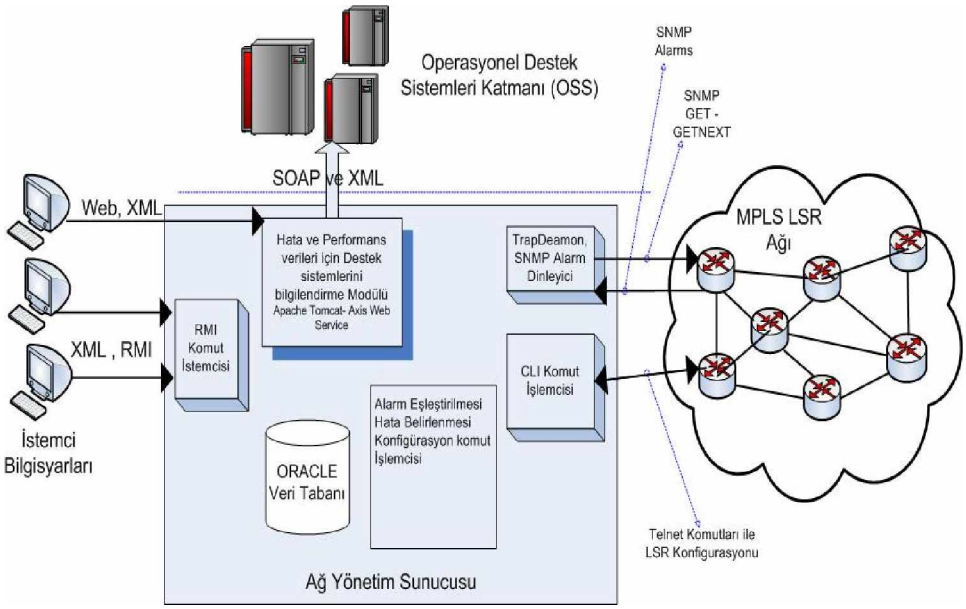


Şekil 6.1: Uygulama mimarisi

6.3 Geliştirilen Sisteminin Mimarisi ve Parçaları

Ağ Yönetimi işlevlerini daha önce ele almıştık. Bu işlevlerin birçoğu uygulama içerisinde parçalar halinde geliştirilmiştir. Bu ihtiyaçların yanında ağ yönetim sistemlerinin birbirleri ile haberleşmesi için kullanılan İşletim Destek Sistemlerine (OSS) olan bağlantıları da yapılmıştır.

Ağ yönetim sistemi 3 katmandan oluşmasından dolayı ölçeklenebilir ve erişilebilirdir. Aşağıdaki şekilde sistemin mimarisini iç parçalarını ve dış dünya bağlantıları ile birlikte geliştirilmiş hali bulunmaktadır.

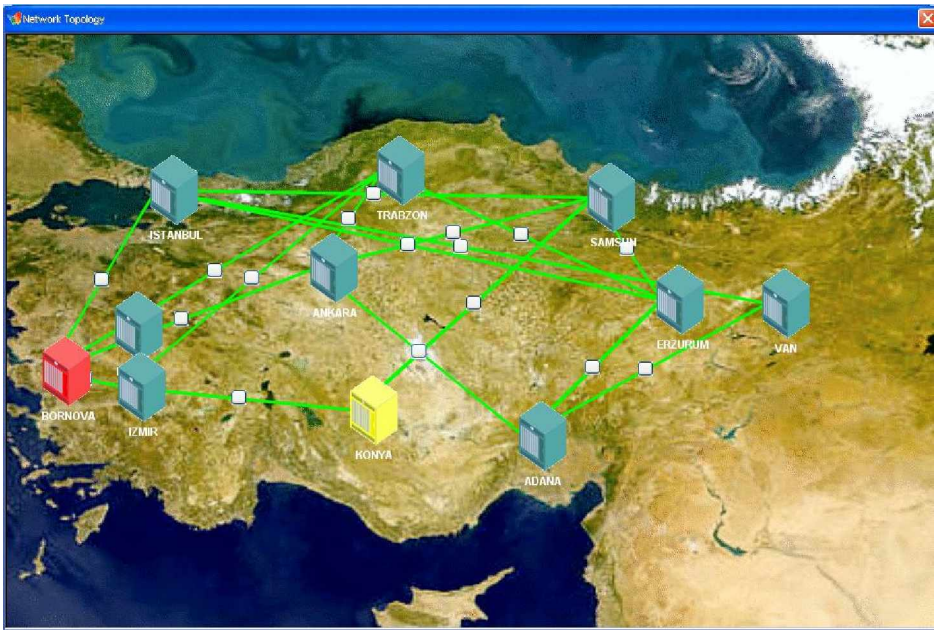


Şekil 6.2: Ağ yönetim sistemi mimarisi

6.3.1 Genel Ağ Yapısı Yönetimi

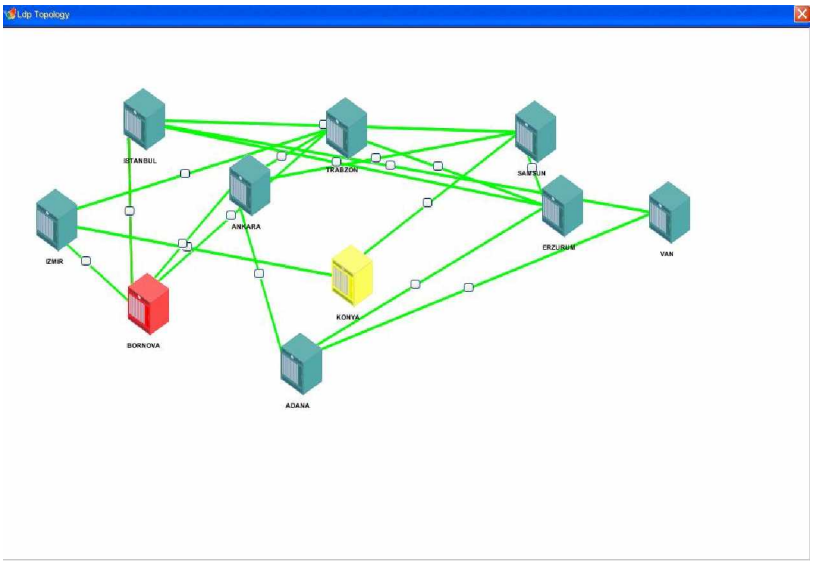
MPLS ağ yapısı içerisinde ihtiyaç duyulan LDP yönlendirme ağ haritası ve genel LSR yönlendirici ağ haritalarının geliştirilmiştir. Topoloji haritalarının geliştiriminde başka bir yardımcı kütüphane kullanılmamıştır.

Ağ işletimi açısından bu haritalar hangi bölgelerde veya hangi ağ elemanlarında hata olduğunu daha genel anlamda takip edebilmek açısından çok önemlidir. Haritalar üzerindeki bağlantılar ve ağ elemanlarının renkleri ile genel anlamda ağın durumu hakkında işletime bilgi verilir. Örneğin yeşil bağlantılar normal durumu, kırmızı bağlantılar ise orada bir problem olduğunu işaret etmektedir. Şekil 6.3 de MPLS ağı üzerindeki LSR yönlendiricilerin genel topoloji haritası görülmektedir.



Şekil 6.3: MPLS LSR yönlendiricileri genel ağ topoloji haritası

Aşağıdaki şekil 6.4 üzerinde LSR yönlendiriciler ve LER yönlendiricilerin birlikte gösterildiği LSR üzerindeki LDP yönlendirme tablolarındaki bilgiler ile oluşturulmuş olan LDP topoloji haritası görülmektedir.



Şekil 6.4: MPLS LDP topoloji haritası

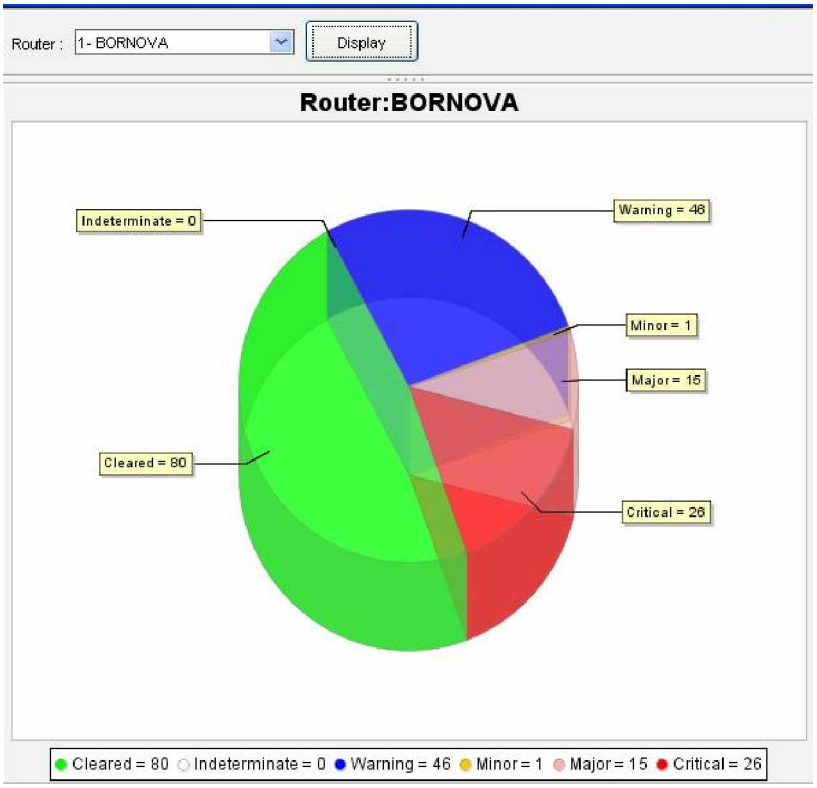
6.3.2 Performans Yönetimi

Performans yönetimi ağ yönetim sistemi içerisinde donanım istatistikleri menüsü altında geliştirilmiştir. Performans yönetimi LSR yönlendiricilerden SNMP üzerinden periyodik olarak toplanan performans verilerinin yanında alarmlar için yine SNMP üzerinden ağ elemanı tarafından gönderilen veriler kullanılmıştır.

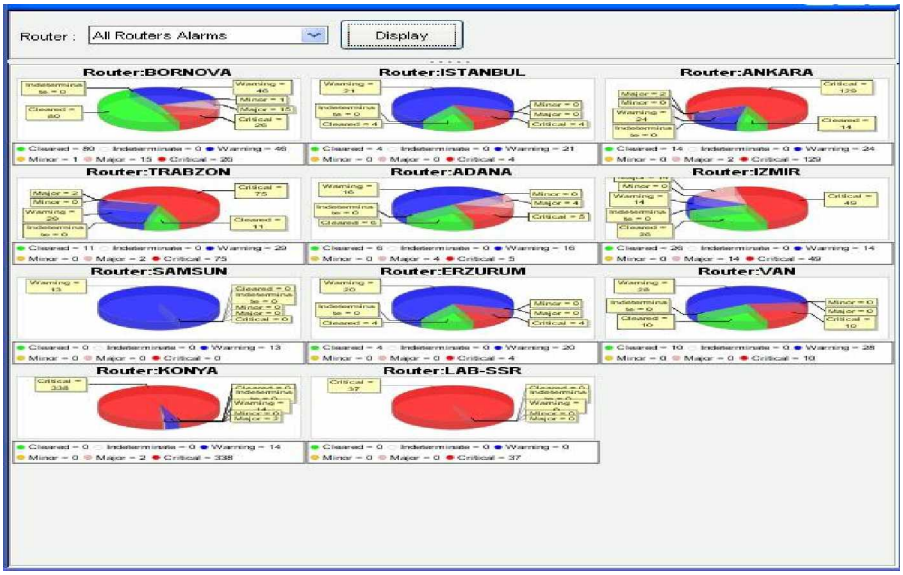
Cihaz yönetimi için en önemli olanlarından başlıcaları; Hata, CPU, FAN, Hafıza istatistikleri grafiksel gösterimlerle güçlendirilerek işlemsel açıdan verilerin takibi kolay kılınmıştır.

Belli bir tarih aralığında bir veya birden fazla yönlendirici biriminin istatistikleri aynı anda görülebilmektedir. Aşağıdaki şekillerde

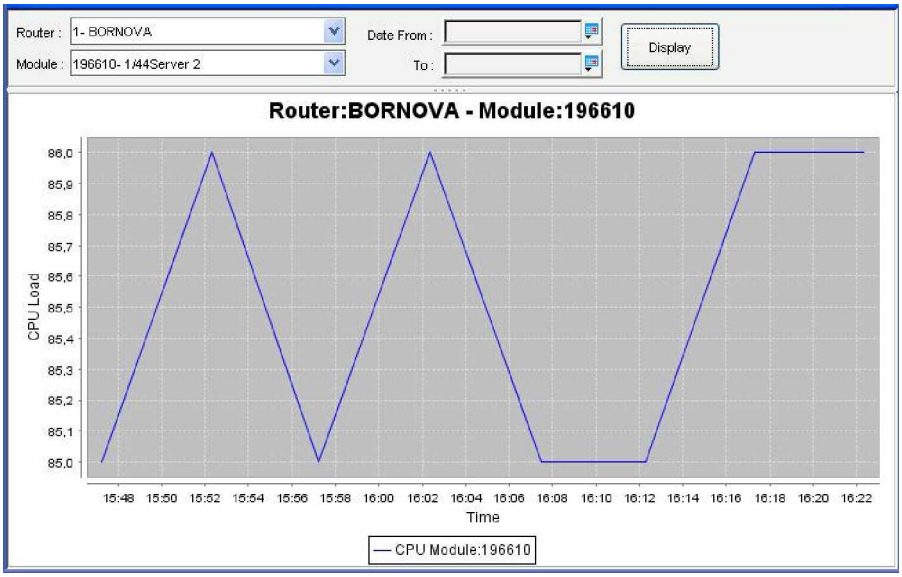
uygulama içerisinde geliştirilmiş olan performans ekranlarını bulabilirsiniz.



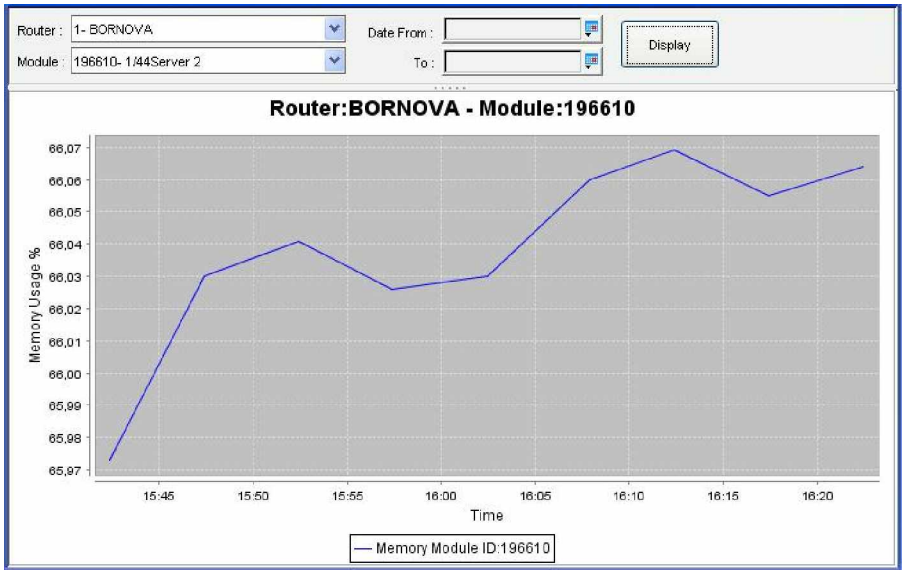
Şekil 6.5: Tek bir LSR için hata istatistik ekranı



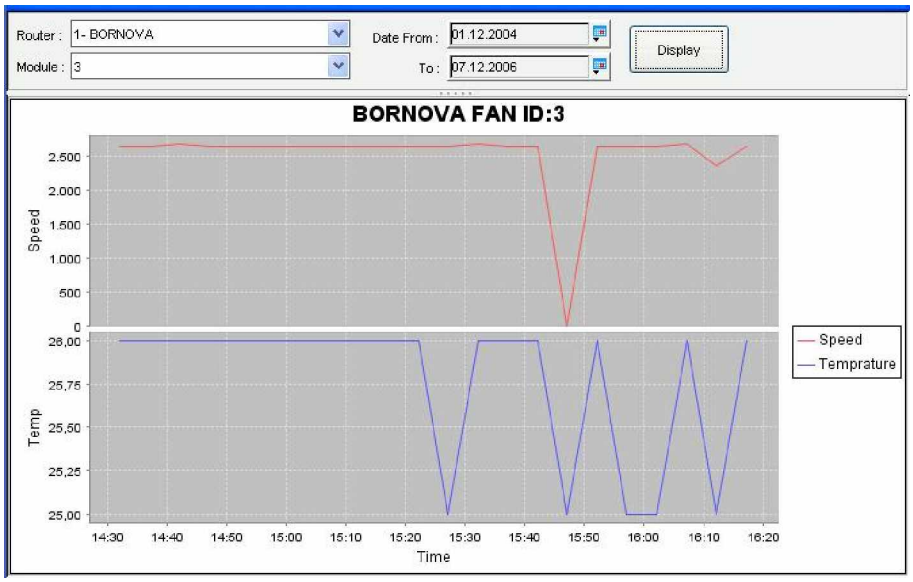
Şekil 6.6: Ağdaki tüm LSR'lar için hata istatistik ekranı



Şekil 6.7: Yönlendirici CPU performans grafiği



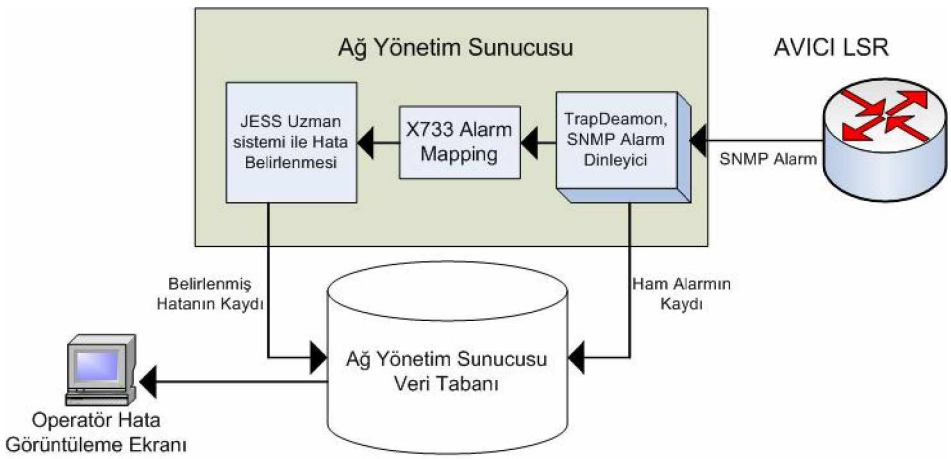
Şekil 6.8: Yönlendirici hafıza performans grafiği



Şekil 6.9: Yönlendirici fanı sıcaklık derecesi ve hız grafiği

6.3.3 AVICI LSR Hata Yönetimi

Hata Yönetimi için LSR yönlendiricinin MIB bilgisi içerisinde yer alan alarm bilgileri sistem üzerinde geliştirilmiştir. MIB dosyaları içerisinde yer alan NOTIFICATION-TYPE nesnelerin SNMPv2 Trap geliştirmesi yapılmıştır. Yapılan geliştirmede alarmları ITU X733 alarm raporlama fonksiyonu standardı kullanılmıştır. Yani SNMP üzerinden AVICI yönlendiriciden gelen alarmların X733 standardına göre dönüşümü gerçekleştirilmiştir. Aşağıdaki şekilde uygulama içerisindeki Hata yönetimi sürecini bulabilirsiniz.



Şekil 6.10: Hata yönetim işlemi

6.3.3.1 X733 Uyum Sınıfı

Gelen alarmların X733 dönüşümü yapılırken TrapDeamon paketi içerisinde geliştirilen ITUX733Compatibility sınıfı kullanılmıştır. Bu

sınıf içerisinde alarmların seviyeleri, tipleri ve görsel gösterimde kullanılmak üzere renklendirme gibi değişkenleri tutulmuştur.

Çizelge 6.1: ITUX733Compatibility sınıfı

```
Public class ITUX733Compatibility {
    public static int CLEARED_SEVERITY = 0;     //Renksiz
   public static int INDETERMINATE_SEVERITY = 1;  //Beyaz
   public static int WARNING_SEVERITY = 2;  //Mavi
   public static int MINOR_SEVERITY = 3;  // Sarı
   public static int MAJOR_SEVERITY = 4;  //Turuncu
   public static int CRITICAL_SEVERITY = 5;   //Kırmızı

   public static String CLEARED_SEVERITY_STR = "CLEAR";
   public static String INDETERMINATE_SEVERITY_STR = "INDETERMINATE";
   public static String WARNING_SEVERITY_STR = "WARNING";
   public static String MINOR_SEVERITY_STR = "MINOR";
   public static String MAJOR_SEVERITY_STR = "MAJOR";
   public static String CRITICAL_SEVERITY_STR = "CRITICAL";

   public static int ENVIRONMENTAL_ALARM_TYPE = 11;
```

```
   public static int EQUIPMENT_ALARM_TYPE = 12;

   public static int PROCESSING_ERROR_ALARM_TYPE = 13;

   public static int QUALITY_OF_SERVICE_ALARM_TYPE = 14;

   public static int COMMUNICATIONS_ALARM_TYPE = 14;

   public static int UNKNOWN_ALARM_TYPE = 15;


   public static String ENVIRONMENTAL_ALARM_TYPE_STR =
"ENVIRONMENTAL ALARM";

   public static String EQUIPMENT_ALARM_TYPE_STR =
"EQUIPMENT ALARM";

   public static String PROCESSING_ERROR_ALARM_TYPE_STR =
"PROCESSING ERROR ALARM";

   public static String QUALITY_OF_SERVICE_ALARM_TYPE_STR
= "QUALITY OF SERVICE ALARM";

   public static String COMMUNICATIONS_ALARM_TYPE_STR =
"COMMUNICATIONS ALARM";

   public static String UNKNOWN_ALARM_TYPE_STR =
"UNKNOWN ALARM"; }
```

6.3.3.2 AVICI Alarm Listesi ve Tanım Yapılandırması

Alarm eşleştirmeleri X733 standardını destekleyecek şekilde XML formatında bir yapılandırma dosyasında tutulmuştur. XML yapısına göre dosya MappingTable içerisinde her alarm için bir eşleme bilgisi tutulmaktadır.

Çizelge 6.2: MappingTable içerisindeki bir kaydın yapısı

```
<Mapping>  aviciBayControllerWarmStart
  <MappingRule>
    <Oid>1.3.6.1.4.1.2474.1.4.3</Oid>
    <Generic>6</Generic>
    <Spesific>2</Spesific>
  </MappingRule>
  <ManagedObject>router = #router# , type = avici , module = bay , bay = #vb1# , controller = #vb2#</ManagedObject>
  <EventType>EQUIPMENT ALARM</EventType>
  <EventName>Event-aviciBayControllerWarmStart</EventName>
  <PercivedSeverity>CLEAR</PercivedSeverity>
  <AdditionalText>Controller #vb2# on Bay #vb1# has been restarted.</AdditionalText>
  <Help> The aviciBayControllerWarmStart notification signifies
```

```
that the
            bay controller has been restarted.    </Help>
        </Mapping>
```

Mapping: Eşleme kuralının genel adı

Mapping Rule: Alarmın belirlenebilmesi için gerekli objeleri içerir.

OID: Alarmın SNMP formatında MIB bilgisi içerisinde yer alan nesne belirleyici numarası.

Generic: Alarmın bulunduğu genel sınıf numarası

Specific: Alarmın özel numarası.

Event-Type: Alarmın genel Tipi

Event-Name: Alarmın genel Adı

PercivedSeverity: Alarmın kural içerisindeki derecesi.

Bu ağ yönetim sistemi için geliştirilmesi yapılan ve TrapMapping XML dosyasına işlenmiş olan AVICI LSR tip yönlendiricinin alarm listesi çizelge 6.1 de verilmiştir. Toplam 100 AVICI alarmı mevcuttur.

Çizelge 6.3: AVICI SNMP alarm listesi

ospfIfStateChange	aviciServerUserAuthenticationFailure
ospfVirtIfStateChange	aviciServerCpuThresholdExceeded
ospfNbrStateChange	aviciServerCpuThresholdCleared
ospfVirtNbrStateChange	aviciServerMemoryThresholdExceeded

ospfIfConfigError	aviciServerMemoryThresholdCleared
ospfVirtIfConfigError	aviciServerDefaultConfigFileChanged
ospfIfAuthFailure	aviciServerDisqualified
ospfVirtIfAuthFailure	aviciServerFailedToDetectBackupServer
ospfIfRxBadPacket	aviciServerDetectedBackupServer
ospfVirtIfRxBadPacket	aviciServerStateTransition
ospfTxRetransmit	aviciServerLdpFecVerifyInconsistent
ospfVirtIfTxRetransmit	aviciServerLdpFecVerifyConsistent
ospfOriginateLsa	aviciQosConfigBWChange
ospfMaxAgeLsa	aviciQosConfigBWRestored
ospfLsdbOverflow	aviciQosConfigBufferSpaceChange
ospfLsdbApproachingOverflow	aviciQosConfigBufferSpaceRestored
aviciSonetSectionEvent	aviciBgpSessionPolluted
aviciSonetLineEvent	aviciBgpSessionCleared
aviciSonetFarEndLineEvent	aviciMemberLinkAdd
aviciSonetPathEvent	aviciMemberLinkRemove
aviciSonetFarEndPathEvent	aviciCompositeLinkActualBWIncrease
aviciBayMonitorUp	aviciCompositeLinkActualBWDecrease
aviciBayMonitorDown	aviciCompositeLinkUp
aviciBayControllerColdStart	aviciCompositeLinkDown
aviciBayControllerWarmStart	aviciGbeEvent
aviciBayControllerIncompatibleSW	aviciGbeMemberLinkEvent
aviciBayFanNotification	aviciTeTunnelUp
aviciBayMultiFanFailures	aviciTeTunnelDown
aviciBayMultiBayControllerFailures	aviciTeTunnelChange
aviciBayMultiBayMonitorFailures	aviciTeMainLSPFullyProtected
aviciModuleColdStart	aviciTeMainLSPPartiallyProtected
aviciModuleWarmStart	aviciTeTunnelBackupInUse
aviciModuleDown	aviciTeTunnelBackupNotInUse
aviciModuleTemperatureNormal	aviciBayControllerDown

aviciModuleTemperatureMinor	aviciBayCommLinkNotificationUp
aviciModuleTemperatureMajor	aviciBayCommLinkNotificationDown
aviciModuleTemperatureCritical	aviciBayVoltageNotificationOutOfSpec
aviciModuleMisconfigured	aviciBayVoltageNotificationInSpec
aviciModuleIncompatibleSW	aviciBayBreakerNotificationUnknown
aviciModuleCommunicationUp	aviciBayBreakerNotificationTripped
aviciModuleCommunicationDown	aviciBayBreakerNotificationClosed
aviciModuleDormant	aviciBayBreakerNotificationOpen
aviciServerAccessModuleDown	aviciMplsLdpFailedInitSessionThresholdExceeded
aviciServerAccessModuleUp	aviciMplsLdpPathVectorLimitMismatch
aviciAllServerAccessModulesDown	aviciMplsLdpSessionUp
aviciAllServerAccessModulesDownClear	aviciMplsLdpSessionDown
aviciServerUp	netrightDatabaseSwitchoverAlarm
aviciServerDown	netrightApplicationServerSwitchoverAlarm
aviciServerHeartbeatUp	netrightCommunicationsLostNotification
aviciServerHeartbeatDown	netrightDatabaseConnectionNotification
aviciServerMisconfigured	netrightDiscoveryFailureAlarm
aviciServerPkgFileReadFailure	netrightLaunchExternalFailureAlarm
aviciServerIncompatibleSW	netrightPollingFailureAlarm

6.3.3.3 Yapay Zekâ ile Alarm Eşleştirmesi

Projede yapay zekâ kullanımı olarak hata yönetiminde alarm tanımlama ve eşleştirilmesi kısmı geliştirildi. Yapay zekâ yöntemlerinden Uzman sistemlerde kullanılan kural tabanlı çıkarsama yöntemi kullanıldı. Java yapay zekâ kütüphanesi olarak JESS kural motoru kütüphanesi kullanıldı (JESS Rules).

Java ile geliştirilmiş olan JESS kural motoru, yorumlayıcı programlama dillerine benzemektedir. Bir uzman sistem kabuğu olarak nitelendirebileceğimiz JESS boş bir bilgi tabanı, bir karar mekanizması, kullanıcı erişim ara yüzü ve uygulama erişim ara yüzü içermektedir. Temel olarak ticari bir üründür, kaynak kodu açık değildir fakat akademik çalışmalar için tam sürüm lisanssız kullanımı mümkündür. JESS'in kendine has bir kural dili vardır. Bu kural dili ile standart olarak geri zincirleme ile çıkarsama yöntemini kullanmaktadır. Fakat JESS aynı zamanda ileri zincirleme yöntemi ile çalışacak şekilde de yapılandırılabilmektedir.

Yazılımımızda JESS (JESS, 2006) kütüphanesinin JAVA uygulama programlama ara yüzü kullanılarak ağ yönetim sisteminin hata yönetimi bölümüne yapay zekâ yeteneği kazandırıldı. Hata yönetimi eşleşmesinin JESS tarafından eşleme yöntemi ile yapılabilmesi için AVICI ağ elemanına ait SNMP alarmlarının tanımları JESS kural dili formatı ile geliştirilerek bir dosyaya kaydedildi. Bunun içerisinde veri tabanı veri tanımı ve alarm verileri bulunmaktadır. Dosyanın içeriğine örnek olarak aşağıdaki çizelgeyi inceleyebiliriz.

Çizelge 6.4: JESS hata eşleme veri tabanı

```
(clear)
(deftemplate alarm
   "A specific alarm."
   (slot name)
   (slot type (default settt))
```

```
   (multislot alarms-to-clear)
)
(deffacts alarm-Knowledge-Base
        "All Alarms Catalog"
        (alarm (name Event-aviciModuleDown) (type set) (alarms-to-clear none))
        (alarm (name Event-aviciModuleWarmStart) (type clear) (alarms-to-clear Event-aviciModuleDown))
        (alarm (name Event-aviciBayMonitorDown) (type set) (alarms-to-clear none))
        (alarm (name Event-aviciBayMonitorUp) (type clear) (alarms-to-clear Event-aviciBayMonitorDown))
        (alarm (name Event-aviciModuleTemperatureCritical) (type set) (alarms-to-clear none))
        (alarm (name Event-aviciModuleTemperatureMajor) (type set) (alarms-to-clear none))
        (alarm (name Event-aviciModuleTemperatureMinor) (type set) (alarms-to-clear none))
        (alarm (name Event-aviciModuleTemperatureNormal) (type clear) (alarms-to-clear Event-aviciModuleTemperatureCritical/Event-aviciModuleTemperatureMinor/Event-aviciModuleTemperatureMajor))
)


(defquery search-alarms-by-name
  "Finds alarm with a given alarm name"
  (declare (variables ?name))
  (alarm (name ?name) (type ?type) (alarms-to-clear ?alarms-to-clear)))
```

Bu veri tabanında kullanılan tanımları inceleyecek olursak;

(clear) : JESS kural veri tabanı hafızasını temizlemek için kullanılıyor.

(deftemplate alarm: Hata yönetiminde kullanılan alarmın parametreleri ile tanımlanması.

(deffacts alarm-Knowledge-Base: Alarm veri tabanının ve barındırdığı kuralların tanımlanması

(alarm (name Event-aviciModuleDown) (type set) (alarms-to-clear none)) : Bir alarm kuralı. Bu kurala göre Event-aviciModuleDown alarmı set tipinde yani bir hata alarmdır ve herhangi bir hata alarmını gidermemektedir.

(defquery search-alarms-by-name: Alarm eşleşme veri tabanının sorgulanabilmesi için kullanılan fonksiyonun tanımı.

Buraya kadar gördüğümüz JESS uzman sistemi ve kurallarının java programımız içerisinden çağırımı aşağıdaki şekilde yapılmaktadır.

Çizelge 6.5: JESS'in uygulama içerisinde örnek kullanımı

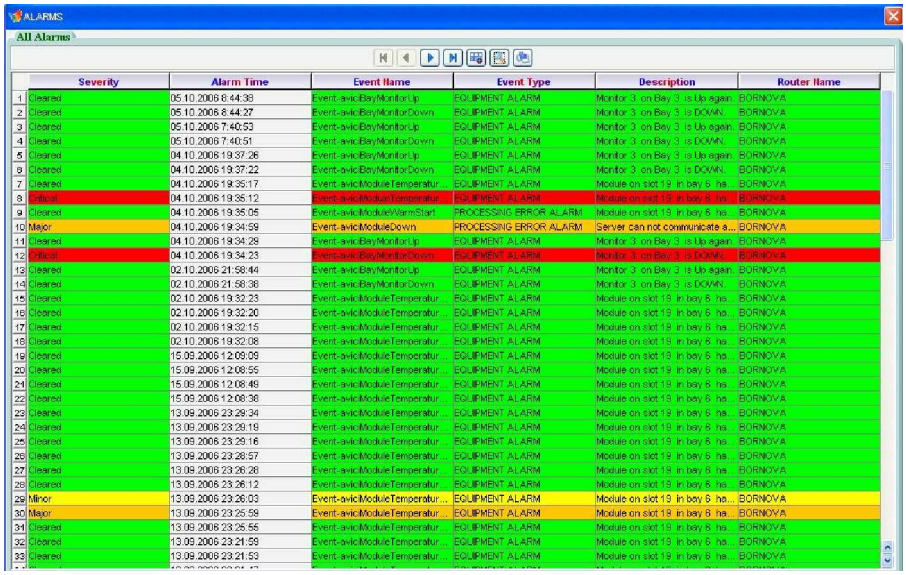
```
import jess.*;


engine = new Rete();
engine.batch("FirstToJess.clp");
engine.reset();


QueryResult result = engine.runQueryStar("search-alarms-by-name",
new ValueVector().add(a.getEventName()));
while (result.next()) {
   String eventName = result.getString("name");
```

```
    String eventType = result.getString("type");
    String alarmsToClear = result.getString("alarms-to-clear");
    LOG.debug("Event: " +eventName+ ", " + eventType
            + ", " + alarmsToClear);
}
```

6.3.3.4 Hata Takip Ekranı

Uygulama içerisinde gelen alarmların ilgili hataya ilişkilendirilmesinden sonra listelenebilmesi için hata yönetim ekranları geliştirilmiştir. Bu ekranlar üzerinden operatörler ağ üzerindeki tüm LSR yönlendiriciler için veya tek bir LSR için ilgili hataları gerçek zamanlı olarak takip edebileceklerdir.

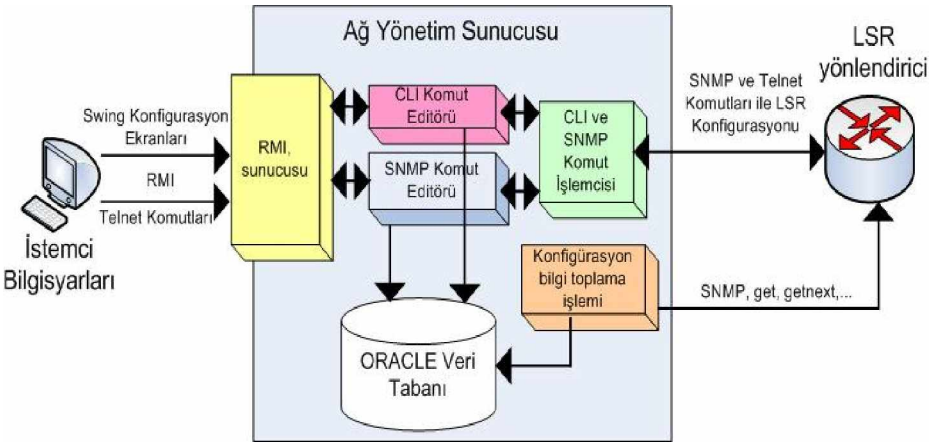
Şekil 6.11 den görülebileceği gibi gelen alarmlar gösterimde renklendirilmiştir. Bu alarmlar içerisinden yeşil olanlar hatanın giderilmiş olduğu alarmları, mavi olanlar uyarı hataları, kırmızı olanlar ise kritik durumdaki hataları göstermektedirler.



Şekil 6.11: Hata takip ekranı

6.3.4 Yapılandırma Yönetimi

Yapılandırma yönetimi, ağın tasarımı ve yapılandırması ile ilgili bilginin yönetimidir. Ağ elemanlarının ve elaman parçalarının stok listesi, kurulum özellikleri ve görevleriyle de ilişkilidir. Uygulama içerisinde tüm ağ elemanları ve birimleri hakkında bilgiler tutulmuş ve bir MPLS LSR ağının yönetilebilmesi için gerekli yapılandırma ekranları geliştirilmiştir. Bu ekranlar vasıtası ile operatörler LSR ağı yönetimi için ağ elemanlarına gerekli komutlar ağ yönetim sistemi aracılığı ile gönderilmektedir. Ayrıca sistemin klasik işletim ekranları aracılığı ile ağ yönetimi yapamadığı durumlar için bir komut işlemcisi aracılığı ile operatörün istediği komutları uç sistemler üzerinde koşturması sağlanmıştır.



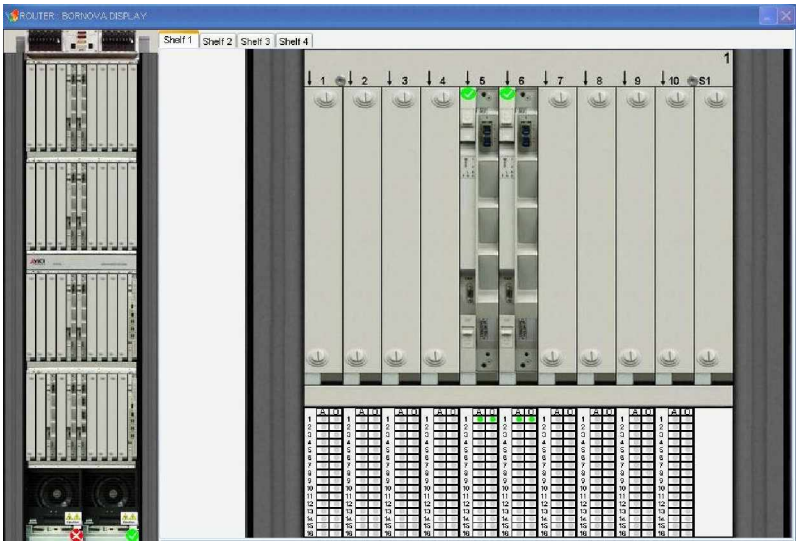
Şekil 6.12: Yapılandırma yönetim mimarisi

Yukarıdaki mimaride ağ yönetim sisteminin yapılandırma yönetimi ve komut işlemcilerinin tasarımı görülmektedir. Ara yüzden alınan komut parametreleri RMI üzerinden sunucu üzerindeki RMI sunucu tarafından karşılanıyor. Buradan ilgili komutun tanımına göre CLI ve SNMP komut editörü tarafından ayrıştırılarak LSR tarafından algılanacak Telnet ve SNMP komutları haline getiriliyorlar. Aynı zamanda veri tabanı üzerine komutlar hakkında bilgiler saklanıyor. Yine SNMP ve CLI komut işlemcileri tarafından komutlar Telnet ve SNMP ara yüzü aracılığı LSR yönlendiriciye gönderilerek komutun sonuçları alınıyor. Tüm bu işlem süresince işlem bütünlüğünü takip edebilmek için veriler veri tabanı üzerinde işlem sırasına göre kaydedilerek takip ediliyor. Gerçekleştirimi yapılan yapılandırma işlevleri şu şekildedir:

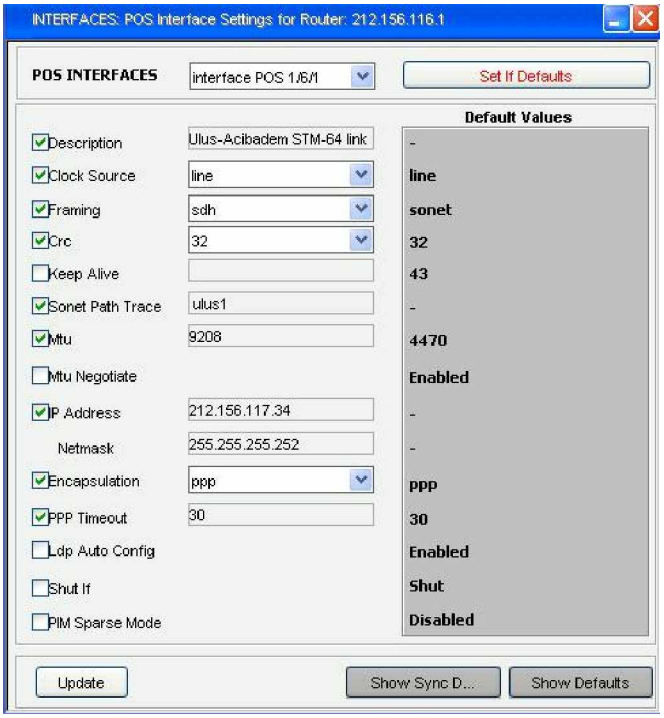
- LSR Ara yüz Yapılandırmaları
    - POS Ara yüzü
    - GBE Ara yüzü

    - VLAN Ara yüzü
- MPLS Yapılandırmaları
    - LSP Ayarları
    - Trafik Mühendisliği
        - Tünel ayarları
        - Yol ayarları
        - LDP Ayarları
- LSR Yönlendirici Yapılandırmaları
    - İşlemci, Hafıza, Ara yüz Bilgileri
    - Fan Bilgileri
    - Şasi, kart bilgileri
- LSR Yönlendirici
    - Dosya yükleme, indirme
    - Yapılandırma dosyası yükleme ve indirme

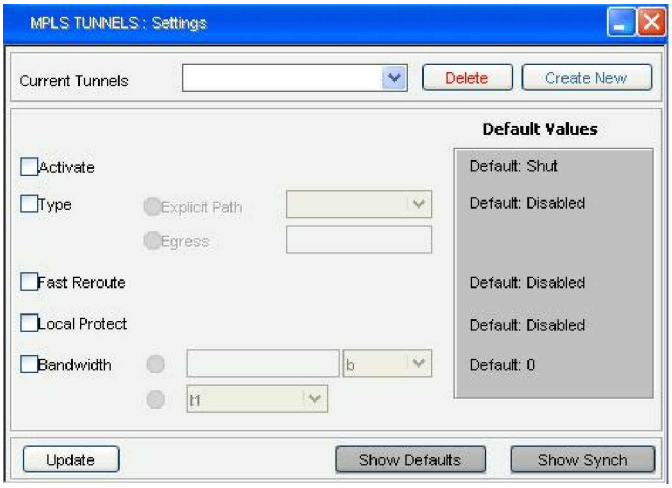
Bu yapılandırma ekranlarından bazıları şu şekilde gösterilebilir.



Şekil 6.13: Yönlendirici şasi görünümü



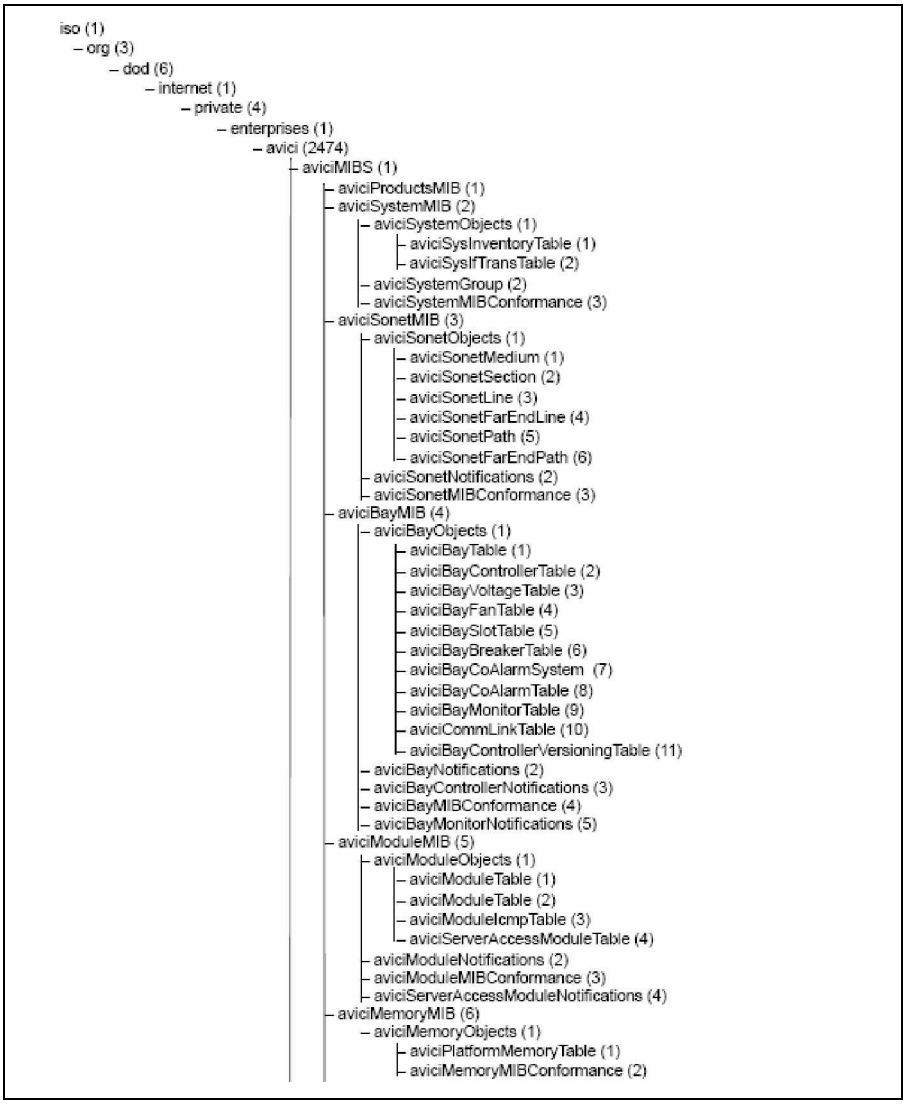
Şekil 6.14: POS ara yüz ayarları



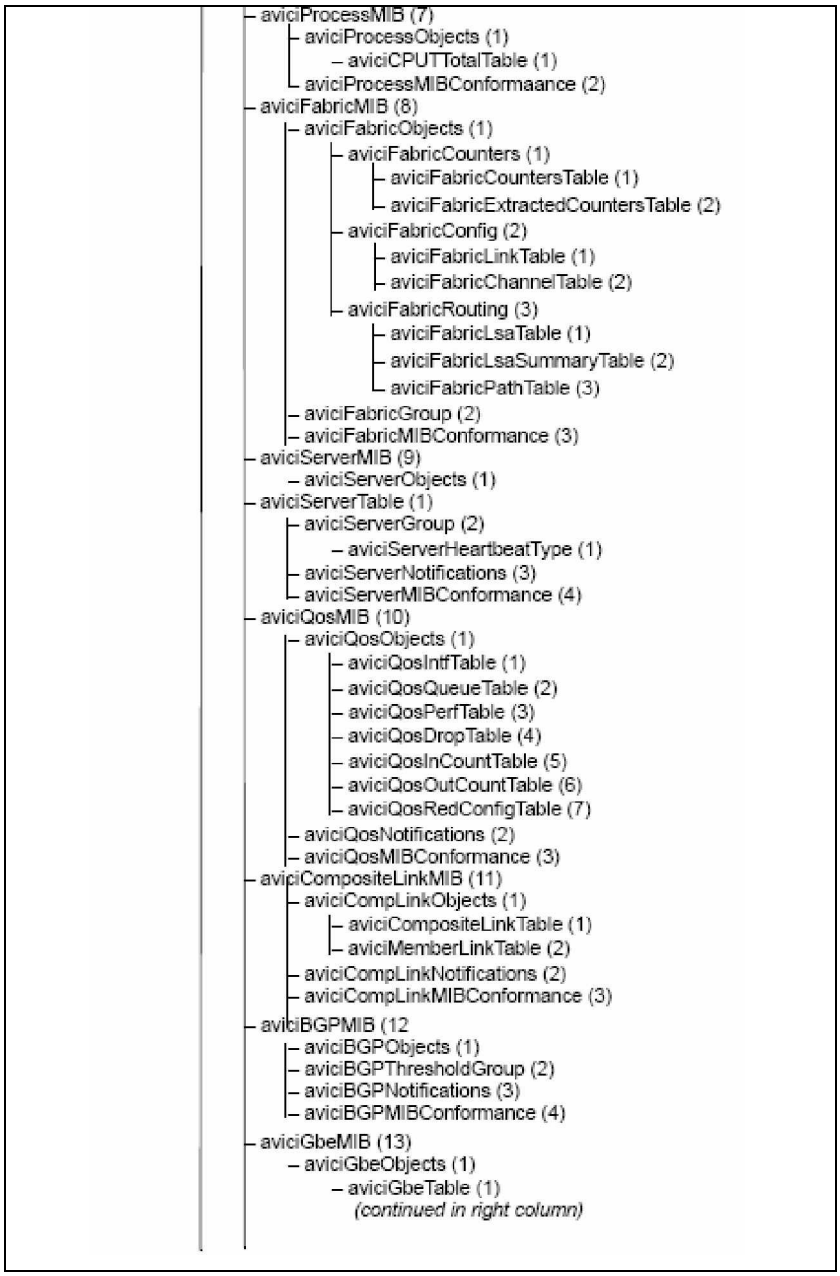
Şekil 6.15: Trafik mühendisliliği yapılandırma ekranı

Yapılandırma yönetimi ve performans yönetimi sırasında veriler AVICI LSR yönlendiricinin SNMP MIB veri tabanı üzerinden sağlanmıştır. Bu veri tabanının ağaç yapısı aşağıda verilmiştir.

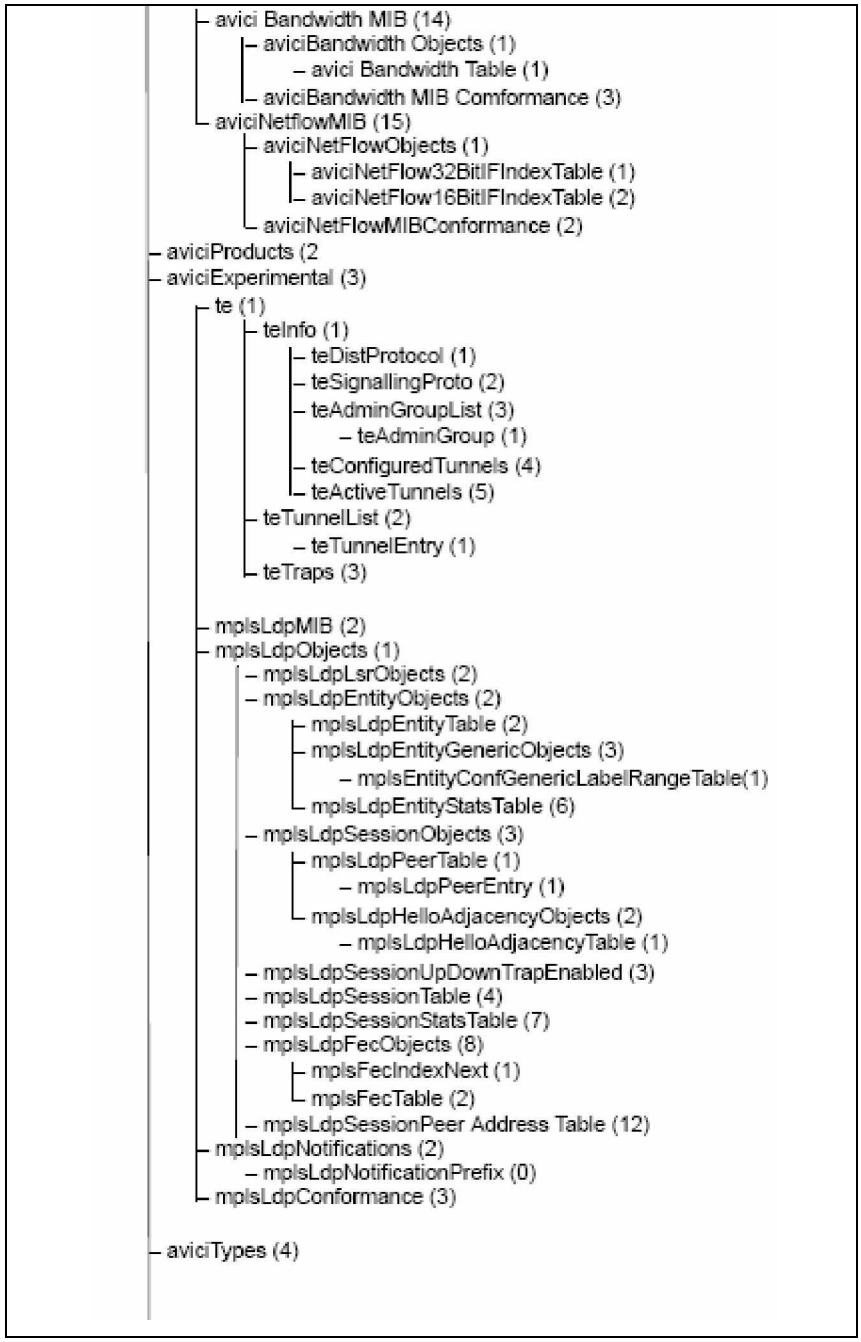
Çizelge 6.6: AVICI SNMP MIB ağaç yapısı



Çizelge 6.6: AVICI SNMP MIB ağaç yapısı (Devam Ediyor)



Çizelge 6.6: AVICI SNMP MIB ağaç yapısı (Devam Ediyor)


avici Bandwidth MIB (14)
aviciBandwidth Objects (1)
avici Bandwidth Table (1)
aviciBandwidth MIB Comformance (3)
aviciNetflowMIB (15)
aviciNetFlowObjects (1)
aviciNetFlow32BitIFIndexTable (1)
aviciNetFlow16BitIFIndexTable (2)
aviciNetFlowMIBConformance (2)
aviciProducts (2
aviciExperimental (3)
te (1)
teInfo (1)
teDistProtocol (1)
teSignallingProto (2)
teAdminGroupList (3)
teAdminGroup (1)
teConfiguredTunnels (4)
teActiveTunnels (5)
teTunnelList (2)
teTunnelEntry (1)
teTraps (3)
mplsLdpMIB (2)
mplsLdpObjects (1)
mplsLdpLsrObjects (2)
mplsLdpEntityObjects (2)
mplsLdpEntityTable (2)
mplsLdpEntityGenericObjects (3)
mplsEntityConfGenericLabelRangeTable(1)
mplsLdpEntityStatsTable (6)
mplsLdpSessionObjects (3)
mplsLdpPeerTable (1)
mplsLdpPeerEntry (1)
mplsLdpHelloAdjacencyObjects (2)
mplsLdpHelloAdjacencyTable (1)
mplsLdpSessionUpDownTrapEnabled (3)
mplsLdpSessionTable (4)
mplsLdpSessionStatsTable (7)
mplsLdpFecObjects (8)
mplsFecIndexNext (1)
mplsFecTable (2)
mplsLdpSessionPeer Address Table (12)
mplsLdpNotifications (2)
mplsLdpNotificationPrefix (0)
mplsLdpConformance (3)
aviciTypes (4)

6.3.5 Kullanıcı Yönetimi

Uygulama içerisinde ağ işletimini ve ağ yönetim uygulamasının yönetim anlamında kullanımını gerçekleştirecek şekilde 2 çeşit genel kullanıcı tanımı yapılmıştır.

Uygulama kullanıcılar arasında fonksiyonel kullanıcı grupları tanımlamaya imkân verir. Bu fonksiyonel kullanıcı gruplarına uygulama içerisinde geliştirilen esnek ve etkin hak yönetimi sayesinde uygulama içerisinde yapılabilecek her türlü işlemi farklı kullanıcı gruplarının kontrolüne vermek mümkündür.

Günümüz ağ yönetim uygulamalarında kullanıcı yönetimi açısından en büyük ihtiyaç farklı ağ elemanlarını, hatta bir ağ yönetimi elemanının farklı birimlerinin (Kart, soket vb gibi) veya yapılandırma sorumluluklarını farklı kişilere verebilmektir. Bu sayede çalışanların sadece kendi sorumluluklarında olan ağ elemanlarına müdahale etmeleri ve ağın diğer bölümleri hakkında yanlışlıkla işlem yaparak veya izni olmayan yerlere girerek problemlere neden olmaları engellenebilmelidir.

Bu ihtiyaçlardan dolayı yazılımımızdaki kullanıcı yönetimi şu işlevleri destekleyecek şekilde geliştirilmiştir;

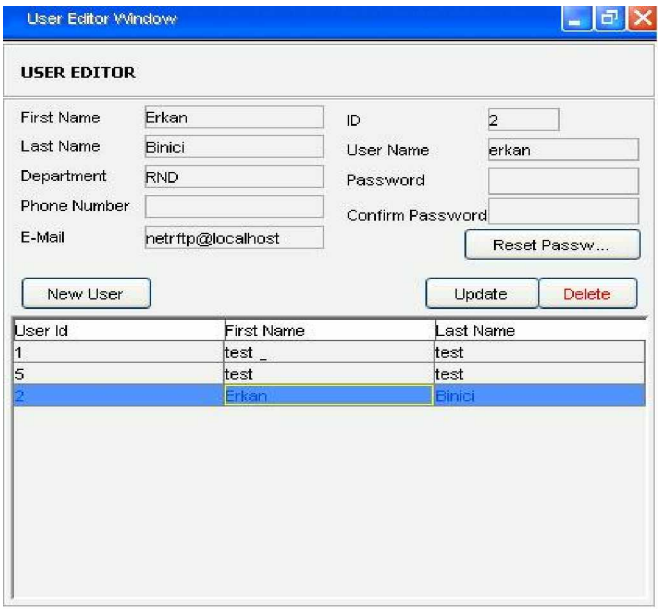
§ Kullanıcı editörü ekranı ile farklı kullanıcılar sisteme tanımlanabilmektedir.

§ Grup editörü ekranı ile farklı kullanıcı grupları yaratılabilmektedir.

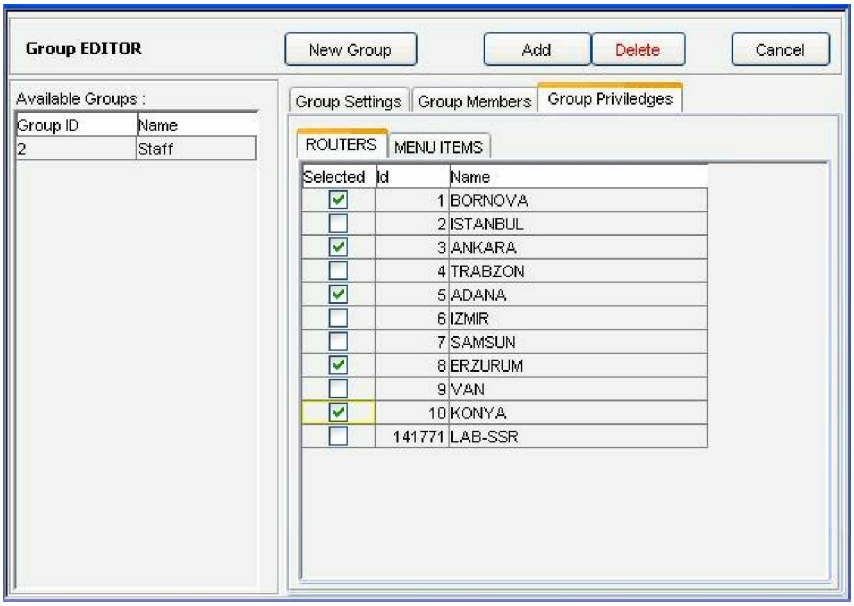
§ Bir kullanıcı birden fazla kullanıcı grubuna atanabilmektedir.

§ Kullanıcılara bir ağ üzerindeki bir yönlendiricinin farklı bölümleri üzerinde işlem yapmaları açısından detayda yetkilendirme yapılabilmektedir.

§ Kullanıcı gruplarına ağ yönetim uygulamasının sadece belli menülerini kullanabilmeleri için haklar tanımlanabilmektedir.



Şekil 6.16: Kullanıcı editör ekranı



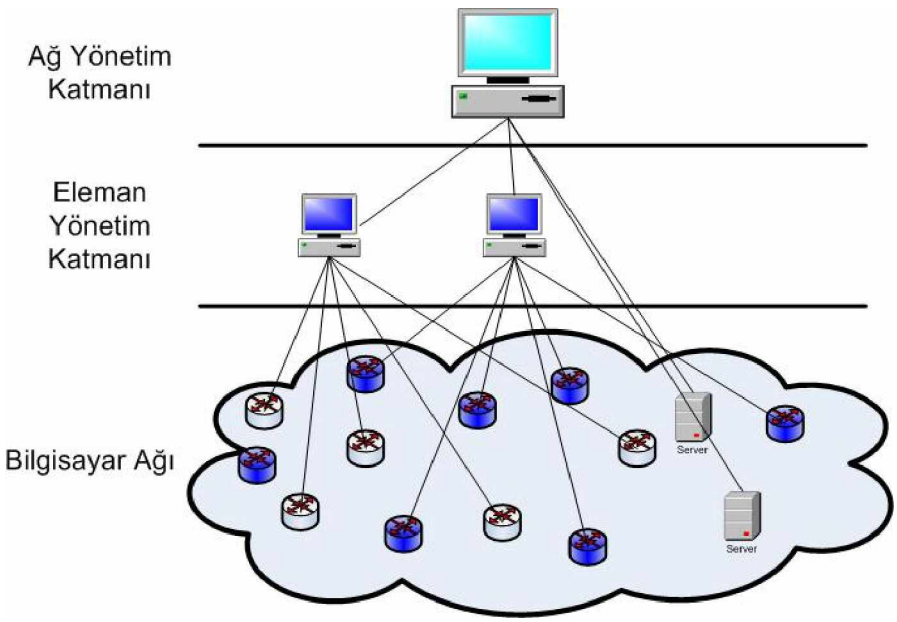
Şekil 6.17: Grup editörü ekranı

6.4 Diğer Ticari Ağ Yönetim Sistemleri

Bilgisayar ağlarındaki en ufak aksaklıklar dahi çok büyük maddi kayıplara neden olabilmektedir. Bu nedenle kuruluşlar yeni ağ sistemleri kurarken veya mevcut bilgisayar ağlarına yeni bir ağ elemanı ekleneceği zaman ilk olarak bu ağ elemanının yönetim ihtiyaçlarına nasıl cevap verebileceklerini sorgulamaktadırlar. Kuruluşlar ağ elemanından maksimum faydayı alabilmek için çok fonksiyonlu yönetim arabirimleri istemektedirler. Bu da üreticileri ürünleri ile birlikte ağ elemanı yönetim sistemini müşterilerine sunmaya çalışmaktadırlar.

Büyük bir bilgisayar ağında birçok farklı ağ elemanının farklı yönetim uygulamaları arabirimleri ile tüm ağın organize bir şekilde yönetilmesi mümkün değildir. Bu nedenle özellikle ağ sistemlerindeki ağ

elemanı çeşitliliğinin artması ile birlikte zaman içerisinde ağ yönetim mimarileri farklı şekillerde gerçekleştirilmektedir. Bunlardan ilki sıradüzensel bir yapı ile yönetim. Bu sıradüzen ağ yönetim sistemleri altında ağ elemanı yönetim sistemlerinin bulunması şeklindedir. Diğer bir yöntem ise özel eleman yönetim sistemi sunmadan bu fonksiyonu daha önce değindiğimiz standart protokoller ile destekleyerek müşterilerinin elinde bulunan genel ağ yönetim sistemlerine bırakmaktır. Genelde büyük Telekom ağlarında bu iki yöntem birlikte kullanılmaktadır. (Şekil 6.18 Ağ yönetim mimarisi)



Şekil 6.18: Ağ yönetim mimarisi

Tez çalışması kapsamında geliştirilen ağ yönetim uygulaması ağ yönetim katmanında çalışmayı destekleyecek şekilde geliştirilmiştir. Ticari paket programları içerisinde sektörde en yaygın olarak kullanılan ağ yönetim uygulamalarına IBM firmasının geliştirdiği Tivoli Netview uygulaması, HP firmasının ürünü olan HP Open View NNM uygulaması ve CA Unicenter TNG uygulaması örnek verilebilir.

6.4.1 IBM Tivoli Netview

Tivoli yönetim çatısı sistem yönetim altyapısı sunmaktadır. 1995 yılında IBM tarafından Tivoli firmasının satın alınması ile IBM bünyesine alınmıştır. Mimarisini oluşturan güçlü sistem yönetim çatısı üzerine birçok Tivoli yönetim ürünü geliştirilmiştir. Örnek olarak yazılım güncelleme sistemi, ağ yönetim sistemi, güvenlik ve kimlik yönetim sistemi, risk yönetim sistemi, veri deposu yönetim sistemi verilebilir. Bu başlık altında bir ağ yönetim ürünü olan IBM Tivoli Netview ürününe değineceğiz.

- Netview en çok kullanılan ağ yönetim ürünlerinden biridir.
- SNMP tabanlı bir ağ yönetim ürünüdür.
- Yeni eklenen birimleri ile MPLS ağları yönelik yönetim fonksiyonlara sahiptir.
- Ağ üzerindeki sunucuların otomatik keşfini yapabilmektedir.
- Ağ topoloji haritaları oluşturabilmektedir.

- § SNMP alarm korelasyon yeteneği vardır.
- § Tüm ağın genel durumu hakkında bilgi verebilmekte ve genel performans verisi toplayabilmektedir.
- § Ürünün mimarisi sayesinde büyük ağları yönetebilmek için genişletilebilmektedir.
- § CiscoWorks gibi lider ağ eleman yöneticileri ile direk bütünleşme sunmaktadır.
- § Ağ üzerinde varlık takibi yapılabilmesini sağlayan işlevleri vardır.
- § Tivoli ürün ailesi içerisindeki diğer yönetim ürünleri ile direk bütünleşmesi bulunmaktadır.
- § AIX, Solaris, Redhat, SuSe, Windows NT ve Windows 2000 platformlarında çalışabilmektedir.
- § Çok yetenekli fonksiyonlara ve ara yüzlere sahip olmasına karşın korelasyon için Prolog dilini kullanmasından dolayı kural eklenmesi uzman sistem yöneticileri gerektirmektedir.

6.4.2 HP Open View NNM

HP OV- NNM ürünü HP firmasının Ağ Yönetim ürünüdür. HP firmasının ürün içerisine direk koyduğu birçok yönetim birimi yanından ürün içerisinde çok iyi tasarlanmış ve tanımlanmış bir bütünleşme ara yüzü bulunması sebebi ile birçok biriminin geliştirilmesine olanak sağlanmıştır. Yani yeni bir ağ elemanı geliştiren bir şirket, bunun

yönetim yazılımını yeni baştan yazmak yerine, HP NNM ürününün ara yüzleri ile bütünleşecek yönetim birimini geliştirebilmektedir.

- Genel olarak SNMP tabanlı bir ağ yönetim uygulamasıdır.
- Birçok HP yönetim uygulaması ile direk bütünleşmeye sahiptir.
- Üçüncü parti yazılımlar ile sıkı bütünleşme yetenekleri vardır.
- Ağ Topoloji yönetim ara yüzlerine sahiptir.
- Ağın otomatik keşfi ve topolojiye yansıtılması işlevleri vardır.
- Hata oluşmadan önce keşfedilmesine yönelik çıkarsama sistemleri içermektedir.
- Ürün içerisinde standart olarak birçok mevcut ağ elemanı için olay korelasyonu kuralları gelmektedir.
- Yeni korelasyon kurallarının eklenmesi ara yüzler ile yapılamamaktadır.
- Unix, Windows ve Linux platformlarında çalıştırılabilmektedir.

6.4.3 CA Unicenter TNG Ağ ve Sistem Yönetimi

Computer Associates firması tarafından geliştirilen Unicenter ürünü ağ yönetim işlevleri dışında TANDEM, VAX yönetimi, Formula–

1 yarış arabası sistemleri ve kola makineleri gibi oldukça özel ihtiyaçlar içinde genişletilmiş ve kullanılan bir üründür.

- Sadece ağ yönetimi için uygulamaya koyulan bir ürün değildir.
- Genellikle kurumun tüm yönetim ihtiyaçları için seçilen ve kurulan bir üründür. Örneğin ITIL süreçlerine uyumlu zengin bir yardım masası yönetim sistemi, iş akışları yönetimi ve problem yönetimi sistemleri ürün içerisinde bütünleşik yer almaktadır.
- SNMP ve RMON protokolleri üzerinden ağ yönetim işlevleri sunmaktadır.
- Otomatik ağ keşfi ve topoloji haritası konumlandırılması gibi işlevleri bulunmaktadır.
- Korelasyon için bütünleşik bir kural tanımlama yeteneğine sahiptir. Kurallar ile alarmların çözümlenmesi ve gerekli aksiyonun alınması sağlanabilir.
- IBM Netview gibi yeni kuralların eklenebilmesi için bir işletimsel ara yüzü bulunmaktadır.
- Ürünün kurulumu kolay değildir. Ürünün farklı yanı uç sistemler üzerinde SNMP ve RMON dışında alınamayan bilgiler için kendi ajanlarının kurulabilmesidir. Bu sayede daha detaylı bilgiler korelasyon amaçlı toplanabilmektedir.
- Ürün ailesi içerisinde HP ve IBM de olduğu gibi çok farklı sistemlerin de yönetimini gerçekleştirebilmek amacı ile farklı

bileşenleri mevcuttur. (Örneğin SNA ağların yönetimi, Mainframe yönetimi gibi.)

6.4.4 Yazılımın Değerlendirilmesi

Tez kapsamında geliştirilen yazılım yukarıda değindiğimiz ticari paket programlara göre bazı artı ve eksi yönleri ile farklılık göstermektedir. Bunlara değinecek olursak;

- § Sadece ağ yönetim sistemi olması için tasarlandığından fonksiyonel ve kullanım açısından daha işe odaklıdır. Bu işletim çalışanlarının ürüne daha hakim olabilmesini sağlamaktadır.
- § Ağ sistemlerinde varlık yönetimi, iş akışı yönetimi, güvenlik yönetimi gibi fonksiyonlara da ihtiyaç duyulmaktadır. Bu fonksiyonlardan yazılım içerisinde varlık yönetimi uygulanmaya çalışılmışsa da diğer ticari sistemlere oranla yeterli değildir. Gerçek bir ağ sisteminde uygulanabilmesi için bunların geliştirilmesi veya standartlar üzerinden gerekli bütünleşme yeteneklerinin sağlanması gerekir.
- § MPLS ağ yönetimine odaklan bir uygulama örneği ile çok zor olan yönlendirici yönetiminde bile uygulanabileceği gösterilmiştir. MPLS ağlarının günümüzde yaygınlaşmaktadır ve bu alanda geliştirilecek diğer yazılımlar için bir açılım oluşturmaktadır.
- § JESS yapay zekâ kütüphanesi ile yazılama diğer ticari paket yazılımlarda olduğu gibi bir kural derleme arabirimi eklenmiştir.

JESS kural kütüphanesi IBM de bulunan prolog arabirimine göre kullanımı daha kolaydır.

§ Yazılım ağ yönetim mimarilerinde en üst katmanda, yani ağ yönetim katmanında kullanılmak üzere geliştirilmiştir. Bunun için herhangi bir TCP/IP ağ elemanının MIB bilgisinin uygulama içerisine derlenmesi ve kurallarının işlenmesi yeterlidir. Diğer ticari programlarda da bu özellik bulunmaktadır.

§ Yeni bir ağ elemanı için yazılımın geliştirilebilmesi için Java ile ilgili nesnelerin kodlanması ve yeniden derlenmesi gerekmektedir. Bu işlem diğer ticari paket programlarda yapılandırma değişikliği ile yapılmakta ve yeniden derlenmeyi gerektirmemektedir.

§ Aynı zamanda eleman yönetim katmanında da kullanılabileceği düşünülerek bir üst sisteme standartlar üzerinden raporlayabilmesi ve bağlanabilmesi için gerekli ara yüzleri gerçekleştirilmiştir. Bu sayede başka bir ağ yönetim sistemi ile bütünleşmesi kolay olacaktır.

§ Yazılım Java ile geliştirilmiştir. Bu da ortam bağımsız olarak Java derleyicisi yüklü olan her türlü sisteme kolayca kurulabilmesini sağlar.

§ Sadece SNMP ve CLI arabirimleri ile yönetim işlevlerini içermektedir. Bunlara daha kabiliyetli olan RMON ve CMIP protokolleri de eklenmelidir.

- Yazılımda gerçekleştirilen Veri Tabanı erişim katmanı soyutlaması sayesinde tek bir veri tabanına bağlı kalmadan ihtiyaca göre farklı veri tabanları ile kullanılabilir.
- Java'nın ANT ve Web Start gibi güçlü kurulum ortamları ile desteklendiği için kurulumu daha kolaydır.
- Yazılımda ara yüz olarak Java Swing arabirimi kullanılmıştır. Bu sayede zengin işletimsel fonksiyonlar eklenmiş olmasıyla birlikte Java'dan kaynaklanan bir yavaşlık söz konusudur. Bu yavaşlığı Web üzerinden yönetim ara yüzlerinin eklenmesi ile aşılması gerekmektedir.

7 SONUÇ

Bu çalışma temel olarak iki ana başlık altında yapılmıştır. Bunlar ağ yönetim sistemleri ve yapay zekâ teknikleri ile yazılım geliştirmedir. Bu iki konuda yapılan çalışmalar sonucunda, ağ yönetim sistemlerinin yapay zeka teknikleri ile nasıl desteklenebileceği üzerinde durulmuştur.

Çalışmada ağ yönetim ihtiyaçları, ağ yönetim protokolleri ve ağ yönetim sistemlerinin mimarileri detaylı olarak araştırılmıştır. Bu araştırma sonucunda etkin bir ağ yönetim sistemi geliştirilmiştir. Geliştirilen ağ yönetim sistemi, ticari bir uygulamada ihtiyaç duyulabilecek temel işletimsel işlevleri içerecek şekilde tasarlanmıştır.

Yönetim sisteminde temel ağ cihazı olarak Çok Protokollü Etiket Anahtarlama (MPLS) ağlarında yer alan Etiket Anahtarlayıcı Yönlendiriciler (LSR) seçilmiştir. Günümüz iletişim ihtiyaçlarının teknolojinin hızlı gelişmesi ile birlikte ihtiyaçları çeşitlendirmesi ve iletişim ağları bu ihtiyaçları karşılayabilecek şekilde ATM ve Frame-Relay gibi altyapılar üzerine IP temelli paket yapısına geçilmektedir. MPLS anahtarlama teknolojisi, sunduğu avantajlarından dolayı mevcut bilgisayar ağları üzerinde uygulanması gittikçe yaygınlaşmaktadır. Tez çalışmasında MPLS ağlarının seçilmiş olması, bu alanda yapılmış yeni ağ yönetim sistemlerinden biri olmasını sağlamıştır. Çalışma içerisinde MPLS ağları hakkında araştırmalar yapılmış ve bu ağların ihtiyaçlarını karşılayacak şekilde ağ yönetim sistemi geliştirilmiştir.

Geliştirilen yazılım, nesne tabanlı mimarisi ve yapılandırmaya müsait alt yapısı sayesinde, kolaylıkla başka ağ cihazlarını da SNMP ve CLI üzerinden yönetmek üzere genişletilebilir.

Çalışmada Yapay Zekâ sistemleri incelenmiş ve Yapay Zekâ yöntemlerinden çok güçlü öğrenme kabiliyetlerine sahip, ağ yönetim sistemlerinde daha yaygın olarak kullanılabilecek, Yapay Sinir Ağları üzerinde daha detaylı araştırma yapılmıştır.

Tez çalışması ile ağ yönetim sistemlerinin ihtiyaçlarını karşılayacak şekilde Yapay Zekâ tekniklerinin nasıl kullanılabileceği, şimdiye kadar bu alanda yapılan çalışmaların araştırılması ve derlenmesi şeklinde gerçekleştirilmiştir. Yapılan bu araştırmalar ile, yeni tasarlanacak ağ yönetim sistemlerinin insan müdahalesini azaltacak şekilde geliştirilmesi sağlanabilir. Geliştirilen yazılımda hata yönetim süreci içerisinde, hata korelasyonu aşamasında JESS yapay zekâ kütüphanesi ile uzman sistem kural tabanlı çıkarsama yöntemi kullanılarak yazılıma yapay zekâ ile eşleme kabiliyeti eklenmiştir. Bu sayede klasik programlama ile çok zahmetli olabilecek korelâsyon fonksiyonu sadece yapılandırma dosyalarına yeni kural eklenmesi haline getirilmiştir. Bu da yazılımın yeniden derlenmesini ve uzun if-then-else koşullarının yazılmasını ortadan kaldırmıştır.

Bir sonraki adım olarak, ağ yönetim yazılımı tüm işlevlerinde yapay zekâ eklentileri ile daha çok desteklenebilir. Hata yönetim süreci içerisinde filtreleme, hata giderimi, planlanması, çözümün test edilmesi ve bu yapılanların sisteme öğretilmesi aşamalarında uzman sistemler ve

yapay sinir ağları kullanılarak çok daha etkin bir hata yönetim sistemi tasarlanabilir. Ayrıca performans ve yapılandırma yönetimi için, geniş ağ sistemlerinde birçok işletimin ağ elemanı üzerinde yapılması tercih edilmektedir. Bunu sağlayabilecek şekilde akıllı ajanlar ile yönetim sistemi desteklenerek performans yönetimi ve yapılandırma yönetimi ihtiyaçlarına daha efektif cevap vermesi sağlanabilir. Güvenlik yönetimi için, ağ üzerindeki ateş duvarı, saldırı tespit sistemleri, doğrulama ve yetkilendirme sistemleri gibi güvenlik sistemleri yapay zekâ yöntemleri kullanılarak daha etkin yorumlanabilir ve yönetilebilir. Bu sayede güvenlik yönetiminin artırılması yönünde çalışmalar yapılabilir.

Bu tez çalışması ile hızla gelişmekte ve genişlemekte olan bilgisayar ağlarının yönetim ihtiyaçlarına, aynı hızda cevap verebilmek üzere, yarı otonom ve tam otonom ağ yönetim sistemlerinin nasıl geliştirilebileceği araştırılmış ve bu anlamda ileriye dönük temel olabilecek bir çalışma gerçekleştirilmiştir.

KAYNAKLAR DİZİNİ

AdventNet, 2006, Advent Net Network Simulator, Available at: http://www.adventnet.com

Axis, 2006**,** Apache Java Web Service Implementation Available at: http://ws.apache.org/axis

Bieszczad, A., White, T. And Pagurek, B., 1998, Mobile Agents for Network Management, IEEE Communications Surveys, Bell Laboratories , 1:2–8, 8p.

Bohoris, C., Pavlou, G. and Cruickshank, H., 2001**,** Using Mobile Agents for Network Performance Management, Center for Communication Systems Research School of Electronic Engineering and Information Technology University of Surrey, 16p.

Byrne, C.J., 1994**,** Fault management. In Aidarous, S. & Plevyak, T. (Eds.), IEEE Press, New York.

Chetan, P.C., Mohapi, S. and Hanrahan, H., 2000**,** Fault Management Service in ATM Networks Using Tina Network Resource Architecture, Centre for Telecommunications Access and Services Department of Electrical Engineering University of the Witwatersrand, 6p.

KAYNAKLAR (DEVAM)

Cisco Systems University, 2006**,** Network Management Basics**,** Internetworking Technologies Handbook, 1-58705-001-3, 6p, US, Available at: http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/nmbasics.pdf

Corn, P.A., Dube, R., McMichael, A.F. and Tsay, J.L., 1988, An autonomous distributed expert system for switched network maintenance, In Proceedings of IEEE GLOBECOM'88, NJ USA.

CVS, 2006, CVS Versioning System Available at: http://www.nongnu.org/cvs

Denise, W.G., Khan I. and Ogier R., 1990**,** An Artificial Intelligence Approach to Network Fault Management, SRI International, Menlo Park, CA 94025 USA,10p.

Dupuy, A., Schwartz, J., Yemini, Y., Barzilai, G. and Cahana, A., 1989, Network fault management a user's view. In Meandzija, B. & Westcott, J. (Eds.) Integrated Network Management, I. North Holland, Elsevier Science Publishers B.V.

Eclipse Foundation, 2006, Eclipse Available at: http://www.eclipse.org

KAYNAKLAR (DEVAM)

Hanemann, A., Sailer, M. and Schmitz, D., 2004, Towards a Framework for IT Service Fault Management, Munich Network Management Team Leibniz Supercomputing Center, Germany, 15–19, 6p.

Hanemann A., 2004, Assured Service Quality by Improved Fault Management ServiceOriented Event Correlation, New York USA, 10p.

Haykin, S., 1999, Neural Networks A Comprehensive Foundation, Prentice Hall, Second edn, IEEE, New Jersey.

ISO/IEC 10040, 1998, Open Sysems Interconnection Systems management overview. International Electrotechnical Commission Available at: http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=24406

Jakobson, G. and Weissman, M.D., 1993, Correlating Multiple Network Alarms Improves Telecommunications Network Surveillance And Fault Management, IEEE Network, 52-59.

JESS Rules, 2006, JESS Available at: http://www.jessrules.com

KAYNAKLAR (DEVAM)

Li, H., and Baras, J. S., 2001, A framework for supporting intelligent fault and performance management for communication networks, Technical Report, University of Maryland, IFIP/IEEE MMNS, Chicago, Illinois, 13p.

Marzo, L. J., Vila, P. and Fabregat, R., 1999, ATM network management based on a distributed artificial intelligence architecture, CCYT, 2p, Available at: http://citeseer.ist.psu.edu/406613.html

MG-Soft, 2006**,** MIB Browser and SNMP Agent Simulator Library Available at: http://www.mg-soft.com/

MIMIC, 2006**,** Network Simulator Available at: http://www.gambitcomm.com

MPLS, 2006, Multi Protocol Label Switching, Available at: http://www.mplsrc.com/

Net-SNMP, 2006**,** Free Open Source SNMP implementation Available at: http://net-snmp.sourceforge.net

Patton, R.J., Chen, J., and Siew, T.M. (1994) Fault diagnosis in nonlinear dynamic systems via neural networks. In Proceedings of the International Conference on Control, 2:1346–1351.

KAYNAKLAR (DEVAM)

Pras, A., 2000, SNMPv1 SNMPv2 SNMPv3 RMON Lecture Notes, University of Twente, Online Course Material Available at: http://www.simpleweb.org/tutorials/slides.html

Redhat, 2006, Redhat Operating System Available at: http://www.redhat.com

Russell S. J. and Norvig, P., 1995, Artificial Intelligence: A Modern Approach, Prentice-Hall.

Saraç T., 2004, Yapay Sinir Ağları, Gazi Üniversitesi Endüstri MühendisliĦi Bölümü Ana Bilim Dalı Seminer Projesi, Ankara, 75p.

Shayman, M.A and Fernandez, E.G., 2000, Fault management in communication networks: Test scheduling with a risk-sensitive criterion and precedence constraints, Proceedings of the IEEE Conference on Decision and Control, Sidney, Australia.

SNMP4J, 2006**,** Free Open Source SNMP API for Java, Available at: http://www.snmp4j.org

Sondak N. E. ve Sondak V. K., 1989, Neural networks and artificial intelligence, SIGCSE '89: Proceedings of the twentieth SIGCSE technical symposium on Computer science education, 241-245pp.

KAYNAKLAR (DEVAM)

Uğur A., 2002, Yapay Zeka ve Yapay Sinir Ağları Ders Notları, Ege Üniversitesi Bilgisayar Mühendisliği, Bornova

Vikipedia, 2006, Vikipedi Özgür Ansiklopedi Yapay Zeka Tarihçesi, Erişim: http://tr.wikipedia.org/wiki/Yapay_Zeka#Tarih.C3.A7e

Wang, Z., 1989, Model of network faults. In Meandzija, B. & Westcott, J. (Eds.) Integrated Network Management, I. North Holland, Elsevier Science Publishers B.V.

Yurtoğlu, H., 2005, Yapay Sinir Ağları Metodolojisi ile Öngörülü Modellemesi, Ekonomik Modeller ve Stratejik Araştırmalar Genel Müdürlüğü, DPT Uzmanlık Tezi, Dpt:2683, Ankara, 104p.

ÖZGEÇMİŞ

Adı Soyadı : Erkan BİNİCİ

Uyruğu : T.C.

Doğum Yeri / Tarihi : Erzincan / 22.09.1979

Adres : 201 Sok No:2-A Kat:2 Daire:9 Bornova/ İZMİR

Tel / E-Mail : (232) 3423688 / erbinici@yahoo.com

EĞİTİM

Yüksek Lisans: 2002 – Ege Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği A. D.

Lisans :1996 – 2001 Ege Üniversitesi
Mühendislik Fakültesi
Bilgisayar Mühendisliği Bölümü

Lise :1993 – 1996 Eskişehir Fatih Fen Lisesi

Mesleki İlgi Alanları :

- Ağ Yönetim Sistemleri
- Yapay Zekâ
- Bilgi Güvenliği
- Nesne Tabanlı Yazlımın Tasarımı ve Geliştirimi